Project acronym: CS3MESH4EOSC

Deliverable D3.2: Initial Implementation of the Platform - Demonstration

Contractual delivery date	31-03-2021
Actual delivery date	09-07-2021
Grant Agreement no.	863353
Work Package	WP3
Nature of Deliverable	D (Demo)
Dissemination Level	PU (Public)
Lead Partner	CERN
Document ID	CS3MESH4EOSC-21-09
Authors	Hugo Gonzalez Labrador, Diogo Castro, Giuseppe Lo Presti, Samuel Alfageme, Ishank Arora, Pedro Ferreira (CERN), Milan Danecek , Miroslav Bauer (CESNET), Daniel Muller (WWU), Armin Burger (JRC), Piotr Wichliński, Marcin Sieprawski (AILLERON)

Disclaimer:

The document reflects only the authors' view and the European Commission is not responsible for any use that may be made of the information it contains.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 863353

Versioning and Contributions History

Version	*Date*	*Authors*	*Notes*
0.1	20.04.2021	Hugo Gonzalez (CERN)	Initial draft
0.2	21.04.2021	Milan Danecek (CESNET)	Invitation workflow
0.3	22.04.2021	Daniel Muller (WWU)	Central Service
0.4	23.04.2021	Giuseppe Lo Presti / Samuel Alfageme (CERN)	Collaborative applications
0.5	26.04.2021	Marcin Siepraswki (AILLERON)	Data Science environments
0.6	03.05.2021	Pedro Ferreira	Suggestions

Table of Contents

Table of Figures

Glossary

Node – a node of the ScienceMesh, normally run by an institution on one of the three initial supported vendor platforms (Nextcloud, ownCloud, Seafile).

API – Application Programming Interface – an interface which defines interactions between software components. In the context of this document, we will be using Web APIs, which target components communicating over a network. Those are either REST APIs, which use HTTP requests and verbs and follow a REST architecture, or APIs based on the gRPC[1] framework and Protocol Buffers[2].

API key – a unique, normally secret token which is used to uniquely identify and/or authenticate the user of an API. In the context of the ScienceMesh's "Central Service" component, this token is used for both identity and authentication and should be kept secret.

HTTP – Hypertext Transfer Protocol, one of the building blocks of the Web.

Markdown – A plain-text data format which allows for basic formatting and styling of text and can be easily converted to HTML.

REST – Representational State Transfer – a software architectural type based on HTTP and which provides guidelines on how to structure Web APIs in an clearly-defined and standard way.

Share – A resource shared between two or more users, normally a folder or a file.

[1] https://grpc.io/
[2] https://developers.google.com/protocol-buffers/

1 Introduction

This document accompanies the Work Package 3 deliverable D3.2 ("Initial implementation of the platform"), available in video format (screen recording) at https://www.youtube.com/watch?v=TnlUOG3lOu4&feature=youtu.be. The goal of this document is to provide an explanation of the various sections of the demonstration video as well as some background for the use cases and tools.

This document will also highlight the main aspects of ScienceMesh's platform and how it already enables some of the use-cases in the context of Work Package 4 (remote sharing, collaborative editing of documents and access to data science environments).

The initial platform consists of two main components:

a. The IOP (Inter-Operability Platform)
b. The "Central Service"

The IOP component is a software package which is deployed on each one of the sites, as to allow them to interact with other components of the ScienceMesh. The IOP guarantees compatibility of the use-cases across multiple vendor EFSS (Enterprise File Sync and Share) platforms and allows them to speak the common language of the CS3APIs[3], essential to the cross-integration of the various micro-services which constitute the platform. The IOP also provides monitoring information about the health of a mesh node and is continuously reporting these metrics to the Central Service component.

The Central Service is a collection of different software solutions which will be deployed as a single package. This service is a lightweight management service for the ScienceMesh federation that enables the monitoring and site registration, allowing the operation team of the ScienceMesh to:

1. register and get an API key which will allow a node to effectively become part of the Mesh;
2. have access monitoring dashboards (through Grafana) and helpful insights (Prometheus) as well as information which will help the governing bodies ensure a fair distribution of resources within the federation.

These two use-cases are implemented through the Mentix service. Another component of the Central Service is the Mesh Directory service, which allows the nodes to establish trust relationships between end-users and bootstrap the process through arbitrary communication channels, thanks to the invitation workflow.

Figure 1 depicts the ScienceMesh platform from a high-level view.

[3] For more information on the CS3APIs, please consult Deliverable D3.1 - "Initial Definition of Protocols and APIs", section 2.2

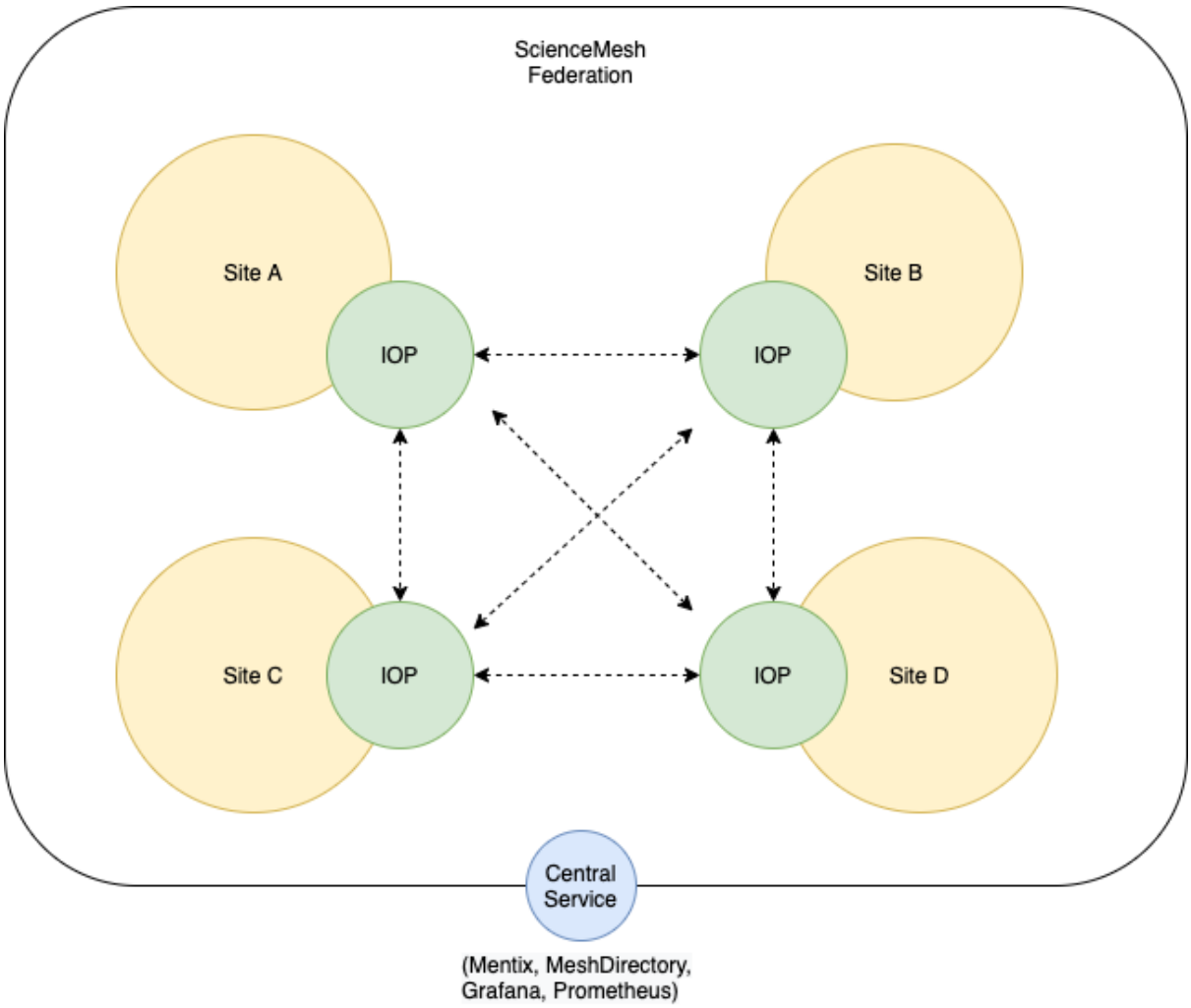

Figure 1. ScienceMesh Initial Platform

1.1 Software foundation

The reference implementation of the Science Mesh IOP component is provided by the Reva4 software package. Reva implements the CS3APIs and bridges them with standard industry protocols, working as translation layer which allows EFSS platforms, Storage Backends and Applications to work together in a vendor-neutral way, despite their differences. Reva is a modular microservice platform that supports both HTTP/REST and GRPC endpoints for communication, making it a great platform to build on top.

Figure 2 shows how Reva interacts with applications, Storage and EFSS, providing a transparent communication layer between otherwise heterogenous systems.

A more detailed explanation about Reva and the CS3APIS can be found in Deliverable 3.1 - "Initial Definition of Protocols and APIs".

[4] https://reva.link

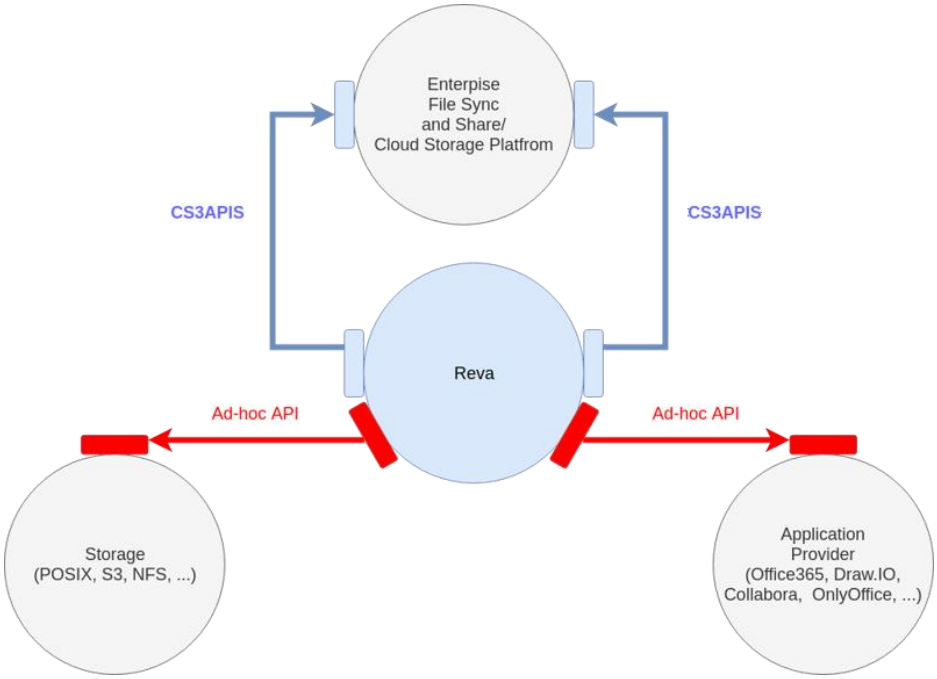

Figure 2. Reva - Bridging the gap between storage, applications and EFSS

Figure 3 shows some of the elements which constitute the IOP (yellow) and the Central Service (blue).

The Central Service was developed as part of task 3.2 with input and requirements from WP2 (QoS, monitoring and metrics). As for the IOP/Reva, it was expanded upon as part of tasks T3.1 (New OCM workflow), T4.3 (WebDAV bridge for CodiMD and CS3APIs BYOA workflow) and T4.4 (New share types for data transfers). All software produced followed the same development and configuration philosophy, which allowed the consortium to capitalize on the know-how which already existed in the teams across the various partners and maximize knowledge sharing among collaborators.

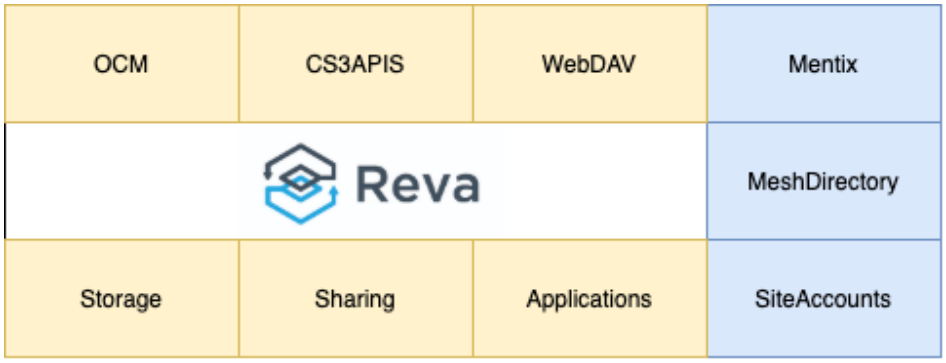

Figure 3. Reva - Software foundation for ScienceMesh platform

2 The Demo

The demonstration video is split in 4 different parts:

1. **How to Join the ScienceMesh** – the onboarding process
2. **Invitation workflow**, or how users share files between them
3. **Real-time collaboration in documents**
4. **Data Science Environments**

In this section, we will go over each one of them in detail and provide additional background whenever necessary. Links will be provided to software packages and repositories whenever relevant.

2.1 How to Join the ScienceMesh

This section highlights the steps a node administrator needs to take in order to join the federation. Those are:

Step 0: Installing the IOP

In order to connect a node to the ScienceMesh and communicate with its systems, the IOP must first be installed next to the Sync and Share platform. Detailed instructions can be found in the ScienceMesh developer docs.[5]

Step 1: Installing the ScienceMesh App for the corresponding Sync and Share platform

To connect a Sync and Share platform to the ScienceMesh, an app specific to the underlying platform system is necessary. These apps will be available on the app stores of the corresponding Sync and Share platforms. Currently, ownCloud[6] and nextCloud[7] applications are provided, with a Seafile version targeted for implementation before the end of the Project. These apps are used to register a node with the ScienceMesh, and they will also export metrics (e.g., the total number of users).

Step 2: Providing the details of the mesh node

After installation of the ScienceMesh app for the specific Sync and Share platform, various settings need to be filled out. This includes general information about the node (e.g., its name and URL), as well as various technical parameters, like the address of the IOP. For the time being, there is also a series of "statistics" which are updated by hand but will be automated before the service goes to production. The most fundamental technical configuration parameter which is required during this step is an API key which will allow the node to communicate with the "Central Service". This key is used to associate any request made by the ScienceMesh app with a user account (see below) and also adds a layer of security. How a user account is created and this key is obtained is explained in the next step. The figure below shows the settings of the ScienceMesh app for Nextcloud as an example.

[5] https://developer.sciencemesh.io/docs/iop/deployment/
[6] https://github.com/sciencemesh/oc-sciencemesh
[7] https://github.com/sciencemesh/nc-sciencemesh

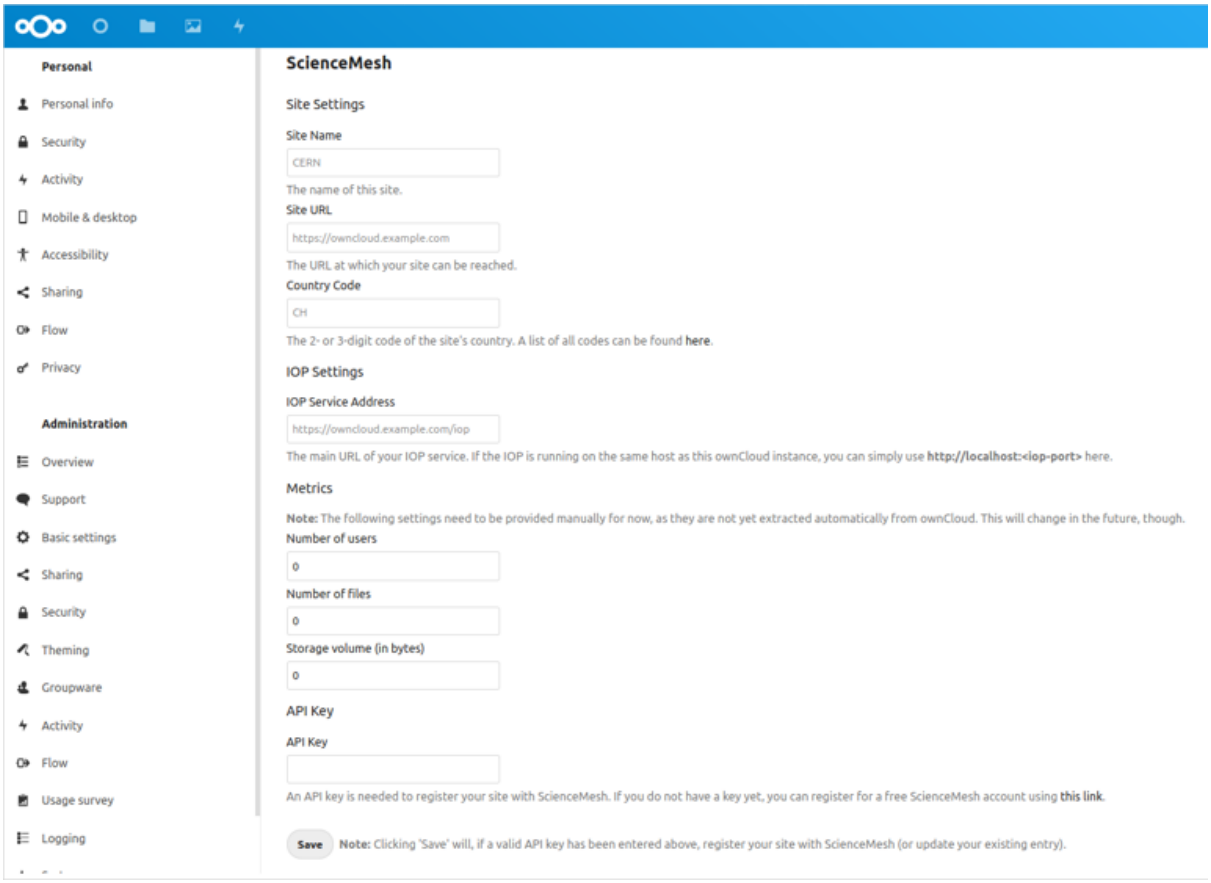

Figure 4. ScienceMesh application

Step 3: Obtaining an API key

To obtain an API key, a node administrator must first register for a ScienceMesh account. This is done through a web form that is linked to on the app settings page. To protect against invalid, fake or otherwise rogue accounts, every newly created account must first be approved by the governing bodies of the ScienceMesh. Accounts registration also may require, in the future, fulfilling a series of technical requirements or providing further documentation. Once the registration is approved, the node administrator will receive his unique API key via e-mail, which they can use to complete step 2.

Once the process is finished, the node will be automatically displayed on the ScienceMesh dashboard, which provides a high-level view of all the sites which are part of the Mesh. The new node will be continuously monitored in order to assess its health and availability. A screenshot of this dashboard can be found below in Figure 5.

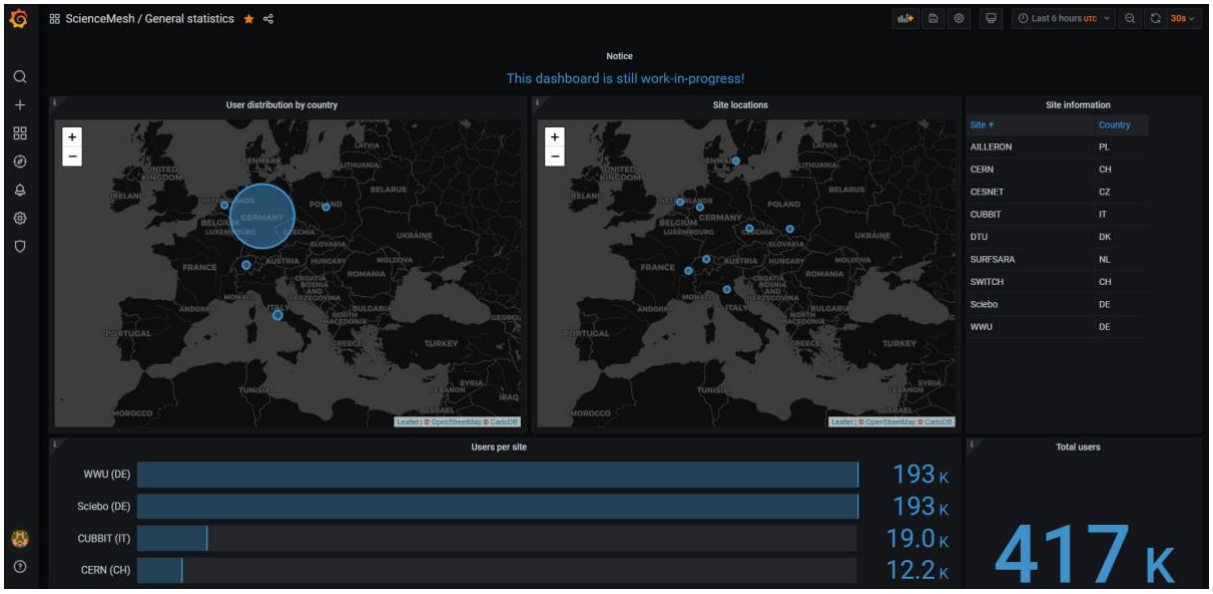

Figure 5. ScienceMesh Node Dashboard

2.2 Invitation workflow

Once a site joined the mesh, its users can start sharing data with others. This section narrates the workflow of how a user at one site can share data with another user from a different EFSS without prior knowledge of target user's identity in the system.

To establish trust between originating and target user, they use so-called "invitation workflow"[8], which provides a privacy-preserving and user-friendly discovery mechanism of user identities. To demonstrate the functionality of the invitation workflow, let us have two users: **Miroslav (originator)** and **Hugo (target)**. Note that Miroslav has no idea about Hugo's EFSS. Miroslav knows only Hugo's email address.

1. To initiate the process, **Miroslav** uses his EFSS system to send an invitation to **Hugo**'s email;
2. **Hugo** receives the email containing a link to the MeshDirectory service running at the originating (Miroslav's) system.He uses a browser to visit the service and lets it know (by picking it from a list) what is his home EFSS (i.e. the target system he will be accessing the data from);
3. **Hugo** is redirected to the target system in order to log in (in a manner similar to identity federations);
4. The systems exchange authentication tokens and the originating system takes note of **Hugo**'s identity at the target system, so that it can initiate the share;
5. **Miroslav** can now share resources with Hugo, either for a single time, or, provided Hugo consented to having his information cached in Miroslav's roster, repeatedly.

[8] The principles underlying the "Invitation Workflow" are described in greater detail in Deliverable D2.1 - "Implementation of federated identity and group management", section 3.2.

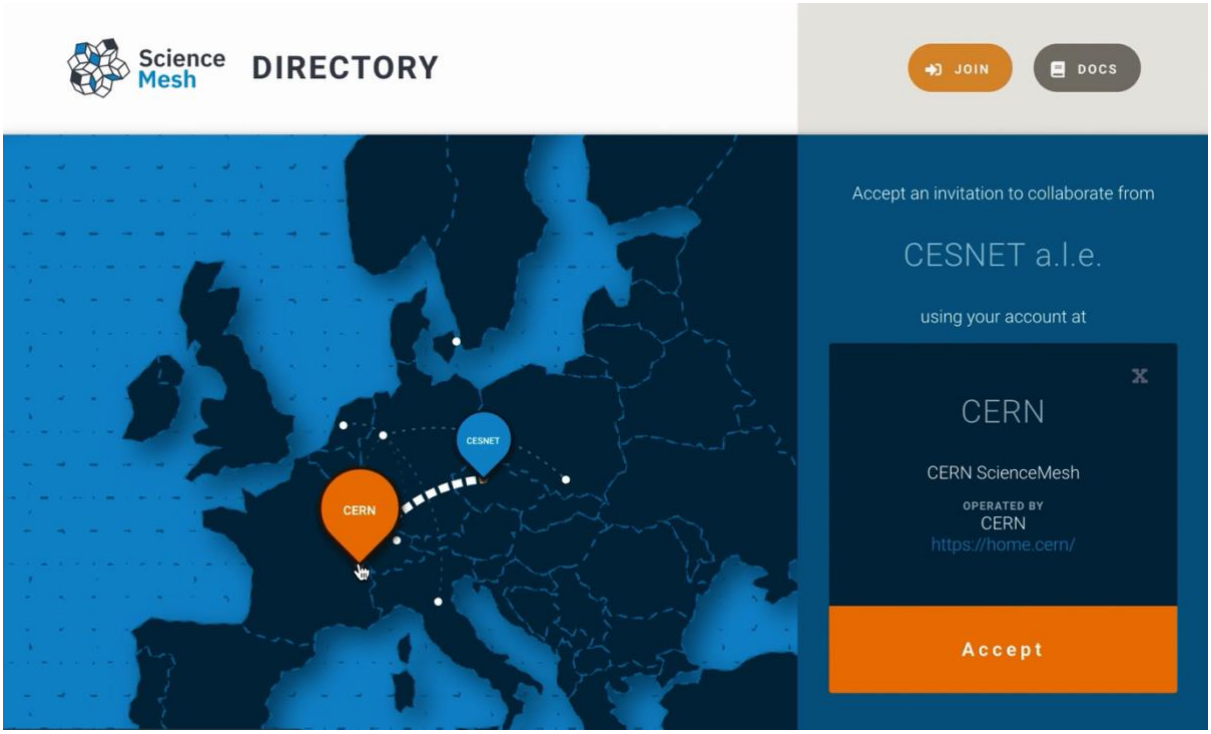

Figure 6. MeshDirectory service invitation acceptance form

During the trust establishment process, the sharing policies of both nodes are compared and their restrictions taken into account. It is important to ensure that the strictest rules on each side are met and the workflow behaves as users expect. The procedure is designed to be compliant with GDPR and other privacy laws: Hugo's e-mail address is never stored by the EFSS system and its handling is beyond scope of the system (it is up to Miroslav to ensure it remains private). The rest of the procedure is based on Hugo's actions. An explicit consent is necessary for a minimal subset of Hugo's information to be retained on the originating site for repeated use.

For more detailed description, we refer the reader to Deliverable D2.1.

2.3 Real-time collaboration in documents

Once a data-sharing relationship has been established between users on different systems, it is possible use collaborative applications to work in on shared content in real time. Web-based applications which allow for concurrent editing by two or more users are becoming increasingly popular and are often seen as a more effective means of collaboration, when compared to the traditional approach of sequential, "round-robin-style" editing of the same file by several users.

To this end, we have investigated applications as diverse as OnlyOffice[9] by Ascensio Systems, Collabora Online[10], and Overleaf, as well as more lightweight applications such as CodiMD[11], an open-source platform which offers real-time collaborative editing for simple text files in plain text or Markdown format, typically meant for taking meeting minutes or creating drafts of documents. In this demo, we

[9] https://www.onlyoffice.com/
[10] https://www.collaboraoffice.com/collabora-online/
[11] https://github.com/hackmdio/codimd

showcase the CodiMD platform integrated into ScienceMesh and used to edit a markdown document. CodiMD has been extended in order to support the integration to cloud-based storages as provided in ScienceMesh, and we plan to contribute back upstream the extensions we have developed.

The approach chosen to integrate CodiMD as an application into ScienceMesh was to include it in a meta-package containing all supported applications that integrate with the Inter-Operability Platform (IOP), and which is available for deployment at https://artifacthub.io/packages/helm/sciencemesh/meshapps.

The integration of CodiMD with the IOP sharing capabilities is possible thanks to the development of an open-source component named the "WopiBridge" that works hand to hand by the WopiServer.

Furthermore, documentation for site operators[12] has been made available for partners to test the deployment of CodiMD and engage their user community.

The figure below shows a live editing session between Munster University and CERN over a shared Markdown document.

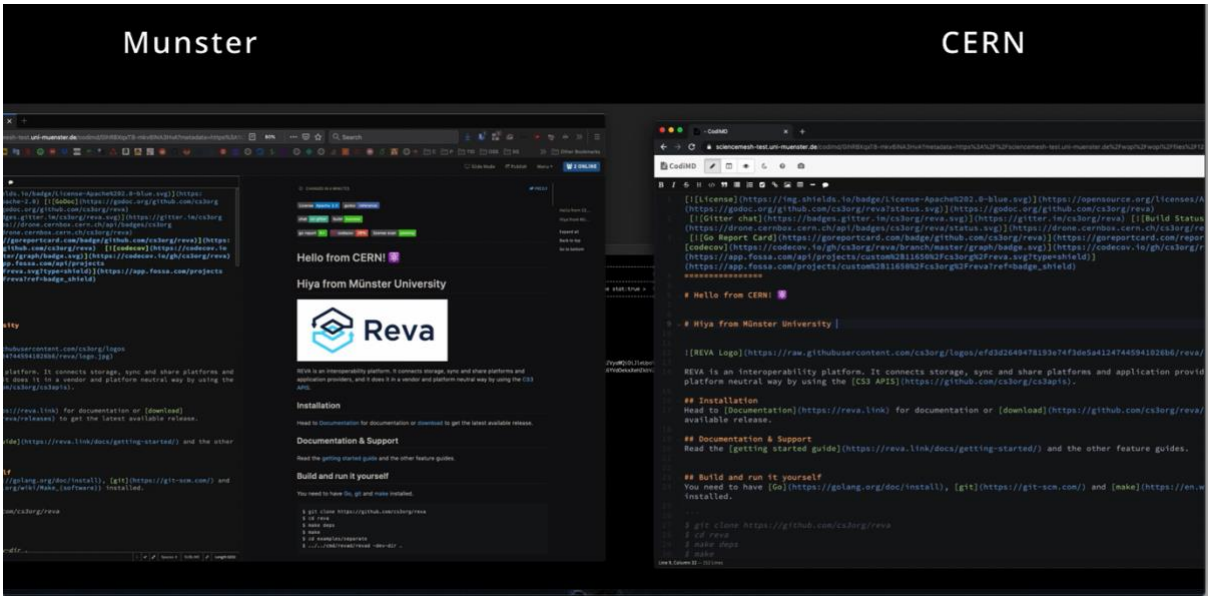

Figure 7. Live editing of a document from two remote sites

2.4 Data Science Environments

Data science environments are integrated into the federated Science Mesh in order to facilitate collaborative research and enable cross-federation sharing of computational tools, algorithms and resources. Users will be able to access remote execution environments to replay (and modify) analysis algorithms without the need for an account in the remote system.

[12] https://developer.sciencemesh.io/docs/integrations/codimd

This section shows the advanced ad-hoc integrations performed in the existing service nodes (sharing, Voilà dashboards) and goes on to explain how ScienceMesh can further enhance these services by connecting the concerned applications to the IOP.
Jupyter Notebook has become the “de facto” platform used by data scientists to build interactive applications and tackle big data and AI problems. For distributed data science environments, the Cs3api4Lab JupyterLab extension was developed, integrating with ScienceMesh which provides sharing and collaboration functionalities (full SC3 API client) directly into a JupyterLab environment. Additional sharing functionalities are available through the file browser: the extension replaces the default JupyterLab file manager, adding new UI elements for additional sharing functionalities (a “shared by/with” tab, sharing buttons and entries in the context menus) and adding new modal dialogs to display file information and sharing status.

The figure below shows the integration of JupyterLab platform with Reva.

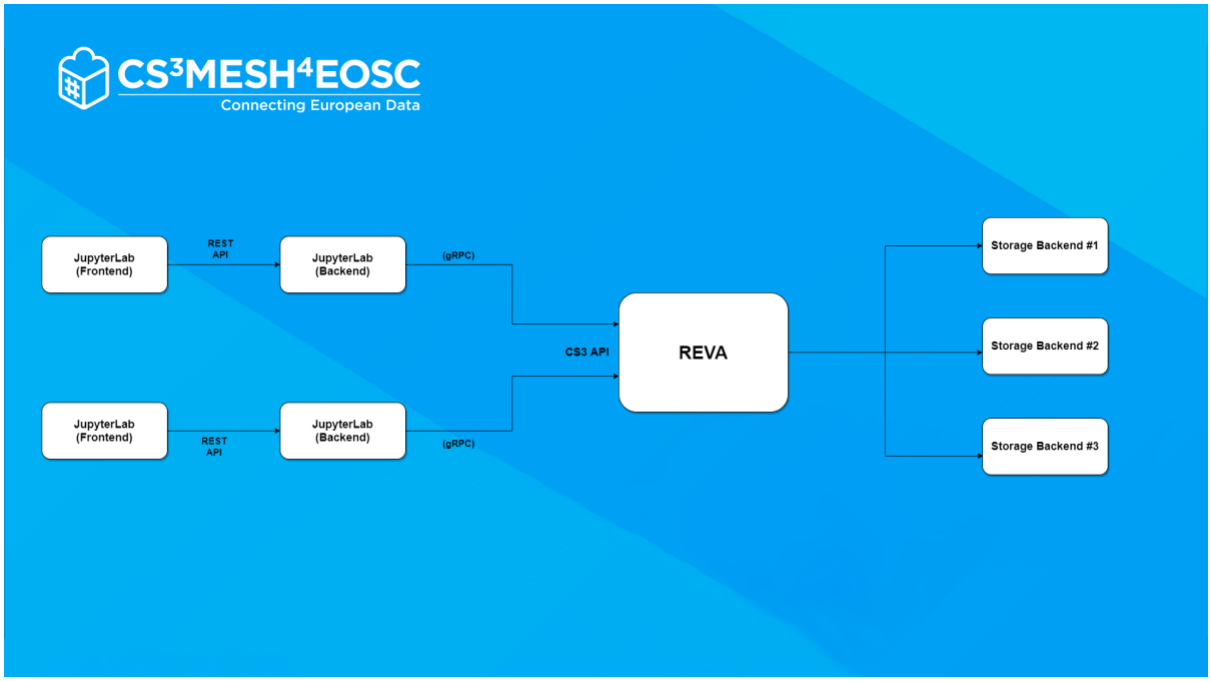

Figure 8. JupyterLab integration with IOP/Reva

3 Conclusion

This document provided additional commentary for the demonstration video. This document provides an overview of main features of the platform and interaction of different components.

It's worth noting that some of the demonstrated use cases, 2.1 and 2.2 above in particular, still lack the graphical user interfaces which will be essential to turning this concept into a widely-usable production-ready solution. This will be the object further work in the context of Task 3.3 and which involve the development of "connector" modules for the targeted EFSS systems (ownCloud, Nextcloud and Seafile).

A more in-depth explanation of the workflows and technologies demonstrated in the aforementioned use cases can be found in deliverables D2.1 - "Implementation of federated identity and group management" and D3.1 - "Initial Definition of Protocols and API".