Project acronym: CS3MESH4EOSC

Deliverable D4.1:
Demonstration of application functionality (all workflows) across federated share (M18)

Contractual delivery date	30-06-2021
Actual delivery date	04-11-2021
Grant Agreement no.	863353
Work Package	WP4
Nature of Deliverable	D (Demo)
Dissemination Level	PU (Public)
Lead Partner	PSNC
Document ID	CS3MESH4EOSC-21-12
Authors	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Piotr Wichliński (Ailleron), Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), Michał Zimniewicz (PSNC), David Antos (CESNET), Samuel Alfageme Sainz (CERN), Angelo Romasanta (ESADE), Frederik Orellana (DTU)

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	1.06.2021	Maciej Brzeźniak (PSNC)	Initial draft
0.2	18.06.2021	Maciej Brzeźniak (PSNC)	Introduction sections, overview of the applications and use-cases done
0.3	30.07.2021	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), David Antos (CESNET)	Application tasks initial descriptions
0.4	31.08.2021	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), Michał Zimniewicz (PSNC), David Antos (CESNET)	Application tasks improved descriptions
0.5	9.09.2021	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), Michał Zimniewicz (PSNC), David Antos (CESNET), Samuel Alfageme Sainz (CERN)	Application tasks complete descriptions
0.6	11.09.2021	Maciej Brzeźniak (PSNC), Angelo Romasanta (ESADE)	4.6 task covered
0.7	12.09.2021	Maciej Brzeźniak (PSNC), Angelo Romasanta (ESADE)	4.6 task cross-checked
0.8	14.09.2021	Maciej Brzeźniak (PSNC)	Demonstrators' descriptions in the draft version
0.9	25.10.2021	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), Michał Zimniewicz (PSNC), David Antos (CESNET), Samuel Alfageme Sainz (CERN)	Draft with summary; Demonstrator descriptions in the final version
1.0	29.10.2021-4.11.2021	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Peter Heiss (WWU), Holger Angenent (WWU), Ron Trompert (SURF), Antoon Prins (SURF), Michał Zimniewicz (PSNC), David Antos (CESNET), Samuel Alfageme Sainz (CERN); Ahmadova Gonzal (ESADE), Angelo Romasanta (ESADE); Remarks and review corrections by Pedro Fereira (CERN) and Jakub Mościcki (CERN)	Final version

Table of Contents

Table of figures

1 Introduction

This document presents the state of work in the WP4 of the CS3MESH4EOSC project on integrating user-level applications and usage scenarios within the ScienceMesh federation.

The CS3MESH4EOSC consortium is building a federated mesh of sync & share systems (EFSS) and applications focused on scientific and research use-cases, referred to as ScienceMesh.

The Interoperability Platform (IOP) worked out in WP3 provides the technical basis for the federation, ensuring compatibility across sync & share products, enabling data flow and user information exchange as well as implementing communication for application workflows.

The partner-provided deployments of sync & share systems at federated sites provide the testing and pre-production environments for the federation. Finally, management procedures and tools developed by WP2 constitute the operational basis for the federation.

Within WP4, a set of user-level applications is being developed with the aim of being embedded into this federated mesh of sync & share systems. This work is organized in four application-related design, development and testing tasks (T4.1-T4.4) as well as two auxiliary tasks (T4.5-T4.6).

This deliverable provides the summary of the work within WP4 related to the integration of the flagship applications identified for the ScienceMesh. It provides an overview of the state of the demonstrators prepared within the first 18 months of the Project. It also summarizes the status of work in the auxiliary tasks dedicated to ensuring that requirements and other information flow among WP4 and WP3, WP4 and WP2, while analysing the future sustainability of those Project results.

1.1 The application tasks

The application tasks address the four "flagship" use-cases of the ScienceMesh:

- **Collaborative data science environments** (T4.1) - scientific computing and data analysis activities, that leads to creating and discovering new data and knowledge;
- **Collaborative document editing platforms** (T4.3) - applications for collaborative editing of the data content (office documents, codes, documentation, visualisations);
- **On-demand data transfer solutions** (T4.4) - point-to-point data transfer mechanisms, including transfer operations between Virtual Organizations;
- **Open data systems and FAIR** (T4.2) – collaborative processes and activities related to the publishing and preservation of data, including data preparation, improvement, curation (for data re-usability and compliance to general domain standards) as well as publication and deposit in repositories (data discoverability and availability).

The four use-cases were analysed during the initial months of the project; analysis of the requirements and usage scenarios obtained from interaction with end-users, site operators and developers has been performed. This part of WP4 work is described in detail in Section 2.1 of the deliverable.

Analysis and evaluation of the technology components existing in particular application areas have also been conducted, including a review of the available software solutions and operational practices and experiences at the involved sites, as well as analysis of usage scenarios with the interested communities. This work is described in Section 2.2.

Based on this analysis, the overall approach to the integration as well as the detailed development and integration paths has been determined. This work included distilling the functional requirements regarding the IOP that were passed to WP3 in M9 of the project and are updated constantly among WP4 and WP3. Based on the initial design the integration work has been conducted along with the iterative design activities. This has been continued until M18. The work was organised in four main streams related to the four application areas. The design and technical integration work are described in Section 2.3.

The development and integration work resulted, until to M18, in preparation of the proof-of-concept implementations of integrated, federated application scenarios and workflows. The demonstrators for the integrations we worked out - they are covered in Section 3.

1.2 Auxiliary tasks

Application-focused activities conducted in Tasks 4.1-4.4 are supported by the auxiliary activities implemented within two tasks in WP4 – Tasks 4.5-4.6.

- **Task 4.5 "Common application requirements and deployment mechanisms"**– supports the analysis, design, development, and integration work on applications by coordinating the collection of the application-level requirements vis-a-vis the federation platform and communicating them to the IOP developers in WP3ands ensuring feedback coming from WP3 is addressed. This task also works on converging and applying the approaches to the application environments deployment and code management across applications of WP4.
- **Tasks 4.6 "Evaluation of the dynamics of collaborative users communities and governance of the CS3 mesh consortium"** – performs an analysis of the impact of the cloud services market, external stakeholders and consortium partners on the development strategy that should be adopted within WP4 for ensuring sustainability of the project results.

Section 4 of the document provides the summary of the activities of tasks 4.4-4.6.

Document positioning

This deliverable accompanies the WP3 deliverables D3.1 and D3.2 that present the initial design and implementation of the Interoperability Platform (IOP). D4.1 provides summary of the demonstrations of the application workflows integrated in the federated platform. The work documented in D4.1 was also impacted by the design of the user invitation workflows, group workflows and application workflows from the point of view of federated identity management than has been documented in WP2 deliverable D2.1.

This deliverable also precedes deliverable D4.2 planned for M12 that will cover the demonstration of the federated application functionality in a user group.

1.3 ScienceMesh Applications – overview

The set of "flagship" applications integrated within the ScienceMesh pertain to four basic areas: Data Science Environments, Open Data systems, Collaborative Documents and Data Transfers Solutions.

These applications aim to cover as exhaustively as possible the workflow of activities in the daily life of researchers and help scientists and other users of data-driven environments at various stages of research and the publication process and various points of the research data lifecycle. This includes: data creation, collection, and transfer, data sharing preparation and improvement, collaborative data curation, as well as data publication and deposit in repositories to make the data accessible and reusable.

Data-Science applications provide friendly interfaces to users in the field of data analytics and data-driven computations. They are right now essential in many of the research activities which lead to producing data and discovering new knowledge. In certain use-cases, especially related to scientific research data that are acquired from dispersed instruments or gathered in distributed data stores, data transfer tools enable running computations and analytics on the remote data by bringing them transparently and automatically to the user's environment. Collaborative documents solutions aid users in sharing, documenting, reviewing, and editing the intermediate results of the R&D. Finally, data and metadata-related functionality oriented at data publication, preparation, packaging, and deposit to repositories supports users in documenting their work and its results by publishing it under standard-compliant open-data systems, enabling data accessibility and re-use. Figure 1 depicts the full research data workflow and lifecycle as defined and perceived by the CS3MESH4OESC project.

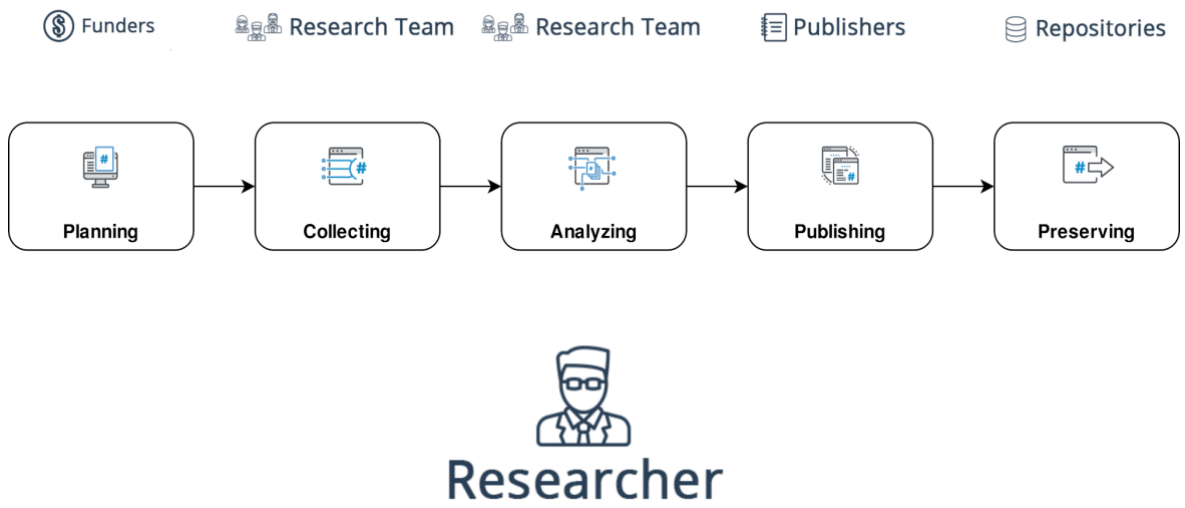

Figure 1. Illustration of the full research data flow – as defined in ScienceMesh

The application tasks within WP4 aim at **providing a comprehensive integrated set of functionalities supporting research workflow elements** including data collection, handling, analysis, processing, publishing, and preserving to the research and business users.

These different applications and features could be applied to the datasets held by end-users in a common federated data management system, which ensures efficient and seamless execution of particular processes of the data-driven research and business.

The auxiliary tasks in WP4 aim at synchronising the application integration work with project efforts on the federation platform development (WP3), deployment and operation (WP2).

While the main aim of CS3MESH4EOSC project is to integrate these applications into a federated mesh of sync & share systems, the applications themselves are improved and extended to enable their better integration – in two dimensions: with particular sync & share systems and with the federation– using IOP-provided functionality and features:

- **Data Science Environments** enable researchers to work on algorithms and data processing code interactively, providing the non-IT-experts with a convenient way to access computing and storage resources; these applications are integrated with the data management layer of the "host" EFSS systems and the ScienceMesh federation; this includes, among others, extending the JupyterLab interface by enabling it to operate on the datasets held in the EFSS and in the federation, and running the notebooks that can themselves access the data in the EFSS or federation; sharing of notebooks as well as the input and output data of the computing and data analysis among scientists is also possible ;
- **Open Data Systems** mean, in this particular context, applications that support scientists and data curators in preparing their data for publication and deposit into data repositories; they provide support for metadata enrichment, data description, data and meta-data packaging, in accordance with the relevant standards, supporting as well the process of data depositing (including assigning a PID/DOI); CS3MESH4EOSC aims to integrate these functionalities directly into EFSS , so that users can perform these processes efficiently and in an integrated way; among others, they will be able to curate data within a sync & share system and trigger the deposit to repositories from that same interface;
- **Collaborative Documents** – applications that enable collaborative, interactive editing & sharing the documents, datasets, code and other content using collaboration platforms embedded into EFSS; the prototypes currently under evaluation include office-like applications, MarkDown-based tools and simple document (notepads etc.) editing systems, online graphic editors (e.g. for graph drawing) and so on; transparent access to collaborative applications running at the remote sites is being also enabled, including enabling access to data held in federated data storage systems and local EFSS systems as well as enabling support for industry standard for collaboration applications such as the WOPI protocol.
- **On-demand data transfers** refer to mechanisms enabling transparent, automated and effective and reliable data transfer from remote to local sites; they are useful especially in the use-cases where it is not possible to extend processing to remote sites, due to compute-data locality issues; data transfer mechanisms are integrated into the sync & share system, so that data can be moved automatically among federated sites, without the need to actively perform and control the transfer; two major scenarios are covered including ad-hoc data transfer between collaborating users at the federated sites: one based on the extension and integration of general-purpose tools such as RClone; another aiming at scheduled transfers of massive datasets from and to large data repositories, driven and operated by research and scientific communities using specialised tools such as FTS/Rucio;

A common denominator of the application integration with EFSS systems and federation taken up in WP4 and in the project in general is to enable cross-site and cross-domain data sharing, exchange, and access, in order to facilitate implementing data-centric research and business processes as well as provide help in preparing research results for open access.

1.4 ScienceMesh Applications – technology

A broad set of technologies is used in the implementation and integration of ScienceMesh-based applications with the respective EFSS systems and with the federated environment based on the IOP. While the descriptions of the design and integration approaches as well as demonstrators worked out in the project for particular applications provide further details of the technologies used, this section aims to give to the reader a birds-eye view over the technological foundations which underlie this application integration work.

It is important to stress that the ScienceMesh is powered almost exclusively by Open-Source software, from its federation-enabling layers up to the higher-level application platform which is the object of this document. Figure 2 shows a simplified view of the EFSS federation and application software stack.

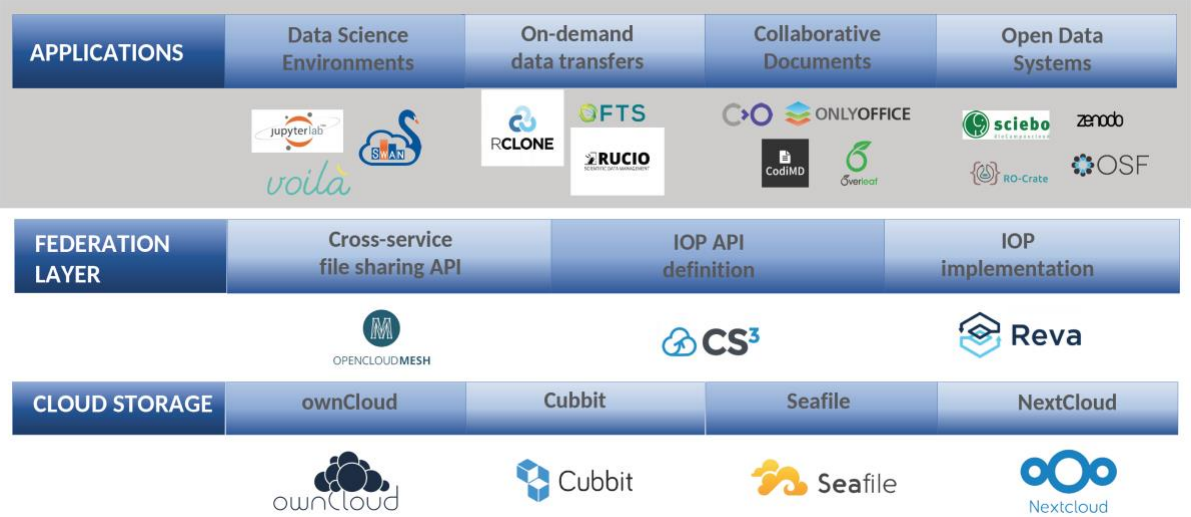

Figure 2. Illustration of the ScienceMesh software stack

In particular, the middle layer of the stack contains solely open-source components. OpenCloudMesh (OCM) with its extensions as well as the CS3 APIs (the interoperability layer) and Reva (the IOP implementation) are all fundamental to the ScienceMesh. OCM in particular, due to his centrality in this vision, transitioned from being an initiative of CS3 partners under the umbrella of the GEANT association to being a community-driven hosted directly by the CS3 community.
The upper layer depicts the application components where Open-Source systems have been adopted wherever possible. For instance, JupyterHub has been taken up for its popularity in the Data Science applications and because its open architecture that enables, among other, developing plugins and extensions and contributing them upstream. A similar approach has been applied to on-demand data transfers and open-data systems based on RClone, Rucio and FTS. Although the collaboration editing platforms integrated include closed-source systems alongside with other open-source ones, efforts have been made in order to guarantee that these products do follow at least widely-adopted standards such as WOPI, which is the industry's default protocol for on-line collaborative applications. The chosen EFSS systems that provide the basis for sync & share functionality and the federation itself include both open and closed source products. This is why it is paramount for federation-development work in WP3 and the project in general to enable these various systems to interoperate based on the CS3 APIs and OCM. The long-term plan is to foster their adoption in the sync & share industry.

2 ScienceMesh Applications – design and integration

In this section the details of the analysis, design and integration work conducted for four ScienceMesh applications are presented. The main datacentric applications of the ScienceMesh - seen in Section 1.2 - have been extended and integrated with the IOP, which provides added value related to the application improvements themselves and the benefits of running and operating these applications in the federated environment.

The application areas included in the WP4 work plan cover a potentially broad spectrum of technical solutions to be integrated and various user requirements to be addressed. Therefore, the specifics of the basic use-cases identified have been used for determining the overall direction of the integration work, work item prioritization, as well as selecting technical components to be integrated and focusing the work on critical improvements and developments.

In this section, an overview of high-level use-case-specific requirements is provided (point 2.1). The Technical solutions and components taken up for integration are discussed as well, along with a short explanation of the selection decisions made (point 2.2). Technical design and the integration work, including design assumptions and implementation work, is also discussed (point 2.3).

2.1 Use-cases overview

The four use-cases included in the project proposal as the "basic" ScienceMesh applications were studied and discussed in detail during the three initial months of the Project. The use-case and application definition included analysis of specifics and high-level functional and quality requirements and usage scenarios, through dialogue with of the developers, architects, and end-users. The results of this analysis acted as an input to the design and integration work.

These ScienceMesh basic use-cases are, as mentioned above, described in the project proposal as well as documented in detail on the CS3MESH4EOSC project website[1]. They are being addressed through integration of four main application groups ("data services") within the Mesh's application platform. Those groups are once again detailed in the project's DoW and documented on the website[2].
This section of D4.1 sheds light on the aspects of the use-cases that had an influence in the design and integration work in WP4 related to applications. This work was targeted and resulted at preparing and integrating four main applications (data services) within the ScienceMesh.

2.1.1 Data Science Environments

Jupyter[3] Notebooks are the number 1 platform used in Data Science to build interactive applications and to work with big data and AI. Jupyter is a free, open-source, interactive web-based tool which researchers can use to combine code, computational output, documentation, and multimedia resources in a single document. It has exploded in popularity over the past couple of years. This rapid uptake has been aided by an enthusiastic community of user–developers and a redesigned architecture that allows the notebook to speak dozens of programming languages.

[1] https://cs3mesh4eosc.eu/use-cases
[2] https://cs3mesh4eosc.eu/data-services
[3] https://jupyter.org/

Jupyter is also a common denominator of the Data Science environments which the ScienceMesh is focusing on: Earth Observation use-cases (as defined by JRC), High-energy physics use-cases (as defined by CERN), education-related use-cases (as defined by PSNC) and business use cases (as defined by Ailleron - Software Mind). Those organizations have identified a series of requirements which guide all the work performed within this task and are detailed below.

Requirement #0 is that the distributed Data Science environments in question are based on JupyterNotebook technology. CERN's HEP use case makes use of SWAN, while the remaining ones stick to JupyterLab. They are both Jupyter-based environments which provide slightly different storage and compute solutions. They are, however, similar enough for a common solution to be found which fits both of them.

Requirement #1 is that additional Science Mesh share functionality should be provided inside the Jupyter Lab environment (through the file browser). A user will be able to easily access, via the web interface, Data Science Environments at a remote site, to interactively work on algorithms and data processing programs shared with them. The user will not be required to create guest accounts or perform other administrative actions. This will "bring algorithms and interactivity close to data" in cases where it does not make sense to transfer data.

This extension will replace the default JupyterLab file manager, adding new UI elements for additional share functionality:

- adding a share by/with tab;
- adding sharing buttons and entries in the context menus;
- and adding new modal windows to display file information and sharing status.

A user will be able to create a notebook, share it with collaborators, who will see it through a "Shared with me" tab directly in JupyterLab, where they will find a unified view of the data owned by and shared with them.

Requirement #2 is to provide tools for concurrent collaboration on notebooks.

Data Science tasks in all aforementioned use cases are performed by distributed teams of scientists from institutes all over the world, with a variety of storage systems and processing tools. For effective collaboration in Data Science, it is necessary (apart from sharing functionality):

- Locking mechanism for concurrent updating of notebooks;
- Concurrent updating and merging of notebooks.

Requirement #3 is that the environment should provide access to computational resources and analytic services: Spark, Condor batch system and GPU resources, for example.

- HEP researchers analyse huge volumes of data from detectors, at very hight rates. The Large Hadron Collider (LHC), the world's largest particle accelerator, produces raw data stream of 600TB/s or (50 000 PB per day). Only a fraction of this is stored, which still amounts to 350PB.
- In the Earth Observation use cases (JRC), the data science environments are collocated with a petabyte-scale storage holding all required public EO data (Copernicus, USGS) for data analysis. A data catalogue with a data discovery API allows users to identify relevant datasets.

Requirement #4 is to integrate and synchronise file storages between Jupyter and EFSS systems.

PSNC has been running several instances of data-science and learning platforms consisting of Jupyter and Nextcloud. A need expressed by users was to see the same user files in both services: data science

environment and sync & share data handling system. In Jupyter, the same files should be accessible from the user interface, from the terminal but also from the running notebook kernels. The emerging requirement is also to integrate or synchronize files and datasets between Jupyter and the sync and share of use - i.e. Nextcloud. This requirement is also relevant for all other use cases.

2.1.2 Open Data Systems

Research projects often don't involve work-practices that describe the data generated over the lifetime of the project so that it meets the FAIR principles. This is especially true of smaller research projects where resource constraints may result in researchers using an application to compile and curate their data, only to find out at the end of the project that the data and metadata are effectively locked up inside the application. Or worse, the research may be completed without any plan for long-term preservation of the data.

Further complicating this issue at an institutional level is how to preserve, and make accessible and discoverable, content coming from different sources and domains when the format of the incoming metadata varies by domain and ontology. The implication is that description as defined by different communities often results in data that is difficult to preserve and leverage at cross-disciplinary level.

For many researchers and domains, resource constraints mean data description is left as a project completion activity that may not actually occur. As ScienceMesh aims to become a massive collection of live data, it presents a great opportunity to counter this problem by offering uniform, cohesive data annotation, packaging and deposit architecture, approach, and solution (tools, services).

Requirement #1 is to perform data annotation on a collections level. Annotation at the individual file level is too burdensome for human intervention. It can only be achieved at the collections level.

Requirement #2, annotation functionality is needed that is sufficiently flexible and extensible that it can deal with metadata from various disciplines, including all *schema.org* schemas or their subsets. It needs to be able to deposit to the collections format chosen in #1, and it needs to operate as a web service and be extensible to talk to the IOP. It also needs to be easy to train users into; a GUI is necessary to avoid steep learning curve and ensure that it can be use by non-IT experts.

Requirement #3, the solution needs to be able to be triggered as part of a research data management workflow, optionally ending in deposit of the collection into a repository, archive, or publishing platform. This unburdens researchers from having to keep track of precisely *when* annotation is necessary and allowing librarians and curation professionals from triggering the data management workflow – e.g., on a project basis ("it is time to publish a paper, therefore we need the attendant data collected, annotated and published as per the publishers' rules").

These three requirements and their interactions have prompted us to settle on:
#1 – ROcrate[4]
#2 – Describo Online[5]
#3 – ScieboRDS[6]

[4] https://www.researchobject.org/ro-crate/
[5] https://arkisto-platform.github.io/describo/
[6] https://www.research-data-services.org/doc/

Based on the requirements and technical components selection, Task 4.2, as of M18 was focusing on making ScieboRDS use Describo-Online as its annotation tool, and for ScieboRDS and Describo-Online to interact with the data inside the Science Mesh through the IOP.

2.1.3 Collaborative Documents

Research and business projects require document- and (more general) data-based collaboration solutions. At various stages of scientific or commercial activities, different types of documents, including typical office files, drawings, notes, spreadsheets, pre-prints, and other types of content can be shared and edited collaboratively.

On-premise enterprise file sync & share systems (EFSS) combined with collaborative document applications can serve these use-cases properly, due to the nature of the data to be edited - its scale (volume), format (variety of data types including typical documents, lightweight formats, drawings), confidentiality level requested (that may disable usage of public clouds) and other regulatory (e.g. not allowing to use public clouds for organisation data) or political factors.

Although sync & share systems themselves enable sharing files among users, however, typical desktop editors are still not able to recognize if a shared file is edited simultaneously on a remote computer nor observe and synchronise the changes in real time, nor present an up-to-date view of the document's content to the editing parties, which may lead to user confusion and data inconsistencies. There is a need to provide integrated collaborative editing applications on top of EFSS systems and their federation enabling live changes tracking of edited and shared content.

ScienceMesh aims to federate multiple sites (EFSS instances) in order to enable cross-site data (documents) sharing and integrates on-line editing applications, in order to enable real-time collaboration between users at different sites, based on the shared documents. In the most ambitious approach, the solution worked out will enable also using the on-line editor's functionality across federation, i.e., running a remotely provided editor to handle the files within the federation.

The high-level use-case definition can be translated into the following functional requirements for federated collaborative document editing platform to be provided by the CS3MESH4EOSC project:

Requirement #1 is that collaborative document editing platforms integrated into EFSS should support various type of documents and content that is handled within the scientific and business use-cases, including office documents, drawings, spreadsheets, lightweight document formats, etc.

Requirement #2 is that the document's data should be available for all the co-editing parties. While this is enabled by the EFSS federation itself at the file-level, collaborative editing platforms have to handle the concurrent access to the document contents as well as real time content synchronisation.

Requirement #3 is that - from the perspective of the user - the editors and collaborative editing functionality has to be available and usable from the EFSS's web browser interface, preferably without the need to install any additional applications on the user's workstation or mobile device.

Requirement #4 is that - the efforts invested into integrating the support for collaborative editing applications into EFSS federation should be re-usable in case new editing platforms and solutions are identified as promising candidates for integration or requested by users or other stakeholders.

The overall approach to address the requirements listed above is to integrate collaborative editing applications capable of working simultaneously across the boundaries of the single systems via shares in the EFSS systems. An important aspect of the integrations conducted is to use - wherever possible - the standard, industry recognised protocols and APIs for integrating documents editing functionality on top of the EFSS systems and their federation, as such an approach promises that the reusability of the solutions works out.
Taking into consideration the functional requirement described above the following candidate solutions to be integrated has been chosen for first round of applications-EFSS integrations:

#1, #2 and #3 - Online document editors:

- Documents, spreadsheets and presentation editors: OnlyOffice, Collabora Online
- Latex editor: Overleaf
- Light-weight documents (Markdown, plain text) editing applications: CodiMD, Etherpad
- Online graphical editors: diagrams.net (formerly draw.io), Excalidraw

#4 - collaborative editing interfaces and APIs:

- CS3 APIs - for implementing the invitation workflow, enable sharing and manage files;
- WOPI protocol (Web Application Open Platform Interface Protocol) - for handling interactive content editing with support for on-line updates processing and presentation.

Based on the requirements and components selection, Task 4.3, as of M18 was focusing on integration of a set of editing applications with IOP-based federation, using WOPI as an interaction interface among applications and EFSS and federation wherever possible (mostly Collabora Online, CodiMD as OnlyOffice support for WOPI is officially available since 3Q2021).

2.1.4 On-demand data transfers

Data-centric collaboration, research and business requires efficient and transparent access to data that may be outside of the local storage system as well as convenient solutions for data exchange.
Also ensuring data-compute resource locality is a vital challenge in distributed IT, data processing and high-performance computing in traditional research disciplines such as astronomy and modern business applications making use of ML- and AI-based data analysis and processing models.

On-premise sync & share systems (EFSS) that are used by scientists and companies for data-based research and business provide minimal support for massive or automated data ingress and download. ScienceMesh aims to enable transparent, automated, and efficient data transfer among federated EFSS sites. The use-cases targeted include spontaneous data transfers initiated by EFSS end-users, among their EFSS systems in order to conduct *ad hoc* data-based collaboration as well as data transfers among EFSS systems and external data infrastructures such as those of large scientific projects - e.g., LOFAR (RadioAstronomy), WLCG (High Energy Physics) and others.

Both ad-hoc and large-scale transfers require proper tools and services to be integrated and available in the data handling and storage systems. Within CS3MESH4EOSC such functionality is being provided on top of the federated mesh of EFSS sites. Even a simple analysis of the two use-cases presented above leads to noticing both the common and the use-case specific requirements.

Requirement #1 is that the actual data transmission - triggered by end-users - *ad hoc* or initiated automatically in a scheduled manner or in response to user's data access request or activity - has to be driven transparently, by using data transfer tools or services integrated within EFSS systems.

Requirement #2 is that the data transferred among users and sites has to be available (visible) in the EFSS interfaces or namespace presented in GUI or by other means (CLI etc.) in a manner similar to 'regular' data shares among users. For this to happen EFSS user interfaces integrations are needed.

Requirement #3 is that the data transfers have to be performed efficiently which can be achieved by applying parallel, multi-threaded transfer of the data over the long-distance network links.

Requirement #4 is that - taking into account the heterogeneity of the distributed, multi-site data storage and processing environment that may include traditional data storage systems (e.g., supporting NFS, or GridFTP access) and cloud storage (S3, Swift, WebDAV) - various data interfaces has to be supported by the data transfer solutions as the data access and storage should happen using native low-level protocols and APIs for great efficiency and resource usage optimisation.

Taking into consideration above-mentioned requirements the following approaches and tools and services has been chosen for the first round of integrations of data transfer mechanisms into EFSS:

#1 and #2:

- Data transfer control and implementation mechanics should follow the data sharing invitation workflow model and re-use its functional components wherever possible;
- The UI extensions have to be worked out for selected EFSS in order to enable triggering and controlling the data transfer and presenting to the user the datasets transferred;

#3 and #4:

- RClone - a Swiss army knife-like data transfer tool is to be used for *ad hoc* transfers, due to its wide range of data access/storage protocols supported, multi-threading capability and lightweight architecture that promises relatively easy integration with EFSS environment;
- Massive data transfers to and from the external data infrastructures will be performed using the FTS (File Transfer Service) and Rucio for its wide acceptance in large scientific projects (e.g., WLCG and others)

Based on the requirements and technical components selection, the work within Task 4.4, as of M18, was focusing on integration of RClone with EFSS systems of choice (ownCloud). Invitation workflow designed within WP2's AAI activity has been adopted as the mechanics for initiating the data transfers among EFFS spaces owned by participating users. UI extensions has been designed following the model of presenting the transferred data in similar way to 'regular' data shares.

2.2 Overview of the technical components

Important part of the work within WP4 was the analysis and evaluation of the existing implementations for applications and use-cases planned to be integrated with the federated EFSS mesh. It included review of the available software solutions and operational practices and experiences at the involved sites and with the communities. The results of this analysis acted as an input to the design and integration work, described in Section 2.3.

2.2.1 Data Science Environments

The work in the Data Science Environment spaces focused on the Jupyter[7] platform as a basic implementation of data science applications. Is also included Voilà dashboards[8] enabling interactive graphically rich data visualisation that provided a non-experts friendly interface for data analysis.

JupyteLab provides a web-based, interactive platform that combines code, text, and outputs, and contains several solutions supporting collaboration such as interactive dashboards, etc. It supports mainstream programming languages including Python, C++, R. As it was already stated JupyterLab has been selected, as it is a *de facto* standard platform used by data scientists for building interactive applications and tackling big data and AI problems both in research and business.
Therefore, the obvious decision based on the market reality was to integrate of Jupyter with IOP. Another reason for that was the fact the partners involved in the task hold knowledge and experience in deploying using, integrating, and extending the platform: CERN, JRC and PSNC use JupyterLab v3. CERN and JRC are using SWAN[9] (Service for Web based ANalysis) platform, that is a CERN-developed package of tools and solutions including Jupyter Lab integrated with CERNBox (the sync & share system based on ownCloud) and EOS (CERN-developed scalable filesystem) as the storage back-ends as well as containerized Spark, Hadoop and/or HTCondor clusters user for actual data analysis and computations. PSNC uses a stand-alone JupyterLab installation with NextCloud as a storage back-end, both deployed on Openshift and OpenStack clusters.

JupyterLab is modular and extendible that allows developing and integrating extra functionality using popular implementation languages. The projected extension allowing to connect JupyterLab to the storage back-end based on the CS3MESH-developed IOP is to be implemented as a typical JupyterLab extension in Python, with an ambition to be taken up to the upstream code.

Voilà dashboards[10] are user for preparing the interactive graphical data visualisation of the data and analysis results to be use by non-experts friendly or graphically rich interface for data analysis. They can be used in order to compile, based on the notebooks the standalone web applications, enabling performing computations using the Jupyter kernels in the back-end and updating the data view provided by Jupyter interactive widgets on the front-end based on the results of computations.

The details of the development work as well as integration and proof of concept activities conducted until M18 based on JupyterLab and other components is described in Section 3 of this document.

2.2.2 Open Data Systems

An important technical component used in Task 4.2 is a RO-Crate[11] (Research Object Crate) standard along with the tools implementing it has been applied. RO-Crate is widely adopted, community-grown standard for packaging research data sets including data with metadata. RO-Crate has been impacted by the Technical University Sydney. It has also been used in the context of the PARADISEC[12] initiative related to protecting the endangered languages, supported by AARNet that is a project partner. The main authors and developers of the RO-Crate standard and packaging tools

[7] https://jupyter.org/
[8] https://github.com/voila-dashboards/voila
[9] https://swan.web.cern.ch/swan/
[10] https://github.com/voila-dashboards/voila
[11] https://www.researchobject.org/ro-crate/
[12] https://www.paradisec.org.au/

(Describo) have been involved in the design, development, and integration activities in T4.2.

The 1st round of integrations of the proof of concept of the RO-Crate based packaging functionality as well as Describo Online data description tool with EFSS system has been conducted using on the Sciebo EFSS service provided by WWU based on ownCloud and WWU-developed ScieboRDS platform for handling research and scientific data.

Sciebo RDS[13] is the solution for integrating with the Sciebo[14] service functionality provided by external services such as those for creation and the administration of data management plans, expert tools, such as editors for special file formats or data types as well as functionalities for capturing metadata or indexing via taxonomies. Sciebo RDS integrates services, tools, and functionalities into research data management processes through using programming interfaces and connecting them to continuous, partly automatic processes and process chains.

2.2.3 Collaborative Documents

Requirements for the document types to be collaboratively edited using on-line editing platforms integrated with the federated EFSS systems are addressed by a set of applications taken up in T4.3.

Office-like content including documents spreadsheets and presentation editors can be covered by integrating Collabora Online by Collabora and OnlyOffice by Ascensio Systems. Co-editing the light-weight types of documents (based on Markdown or plain text) can be addressed by integrating CodiMD[15] that is the Open Source version of the HackMD - an online Markdown editor, used for writing project documentation and by software developers as well as by integrating Etherpad[16], an online, real-time collaboration text editor frequently used by IT teams for taking the meetings notes.

An architectural assumption in collaborative documents area was to base the integration on industry adopted standards. Therefore, the candidates taken up for the first round of integration included applications that support WOPI such as Collabora Online. OnlyOffice supports WOPI officially since 3Q2021 so its integration will be performed in the remaining part of the project.

Overleaf[17] is also an interesting option for collaborative online Latex editing, however its architecture and licensing mode would require involving its developers in the integration process.

Integration of online graphical editors was considered including Diagrams.net[18] tool, previously known as DrawIO as well as Excalidraw[19] application that provides an interactive on-line whiteboard.

The choice of the online editors considered for the integration with ScienceMesh has also been motivated by the fact that the file types covered by them are very common in the researcher's world and are typically used in the daily work. It was of importance, too, that the editors are capable of a collaborative editing possibility to make the concurrent work of researchers at single files possible.

[13] https://www.research-data-services.org/
[14] https://www.uni-muenster.de/IT/en/services/arbeitsplatz/sciebo/index.html
[15] https://github.com/hackmdio/codimd
[16] https://etherpad.org/
[17] https://www.overleaf.com/
[18] https://www.diagrams.net/
[19] https://excalidraw.com/

Last but not least, the licensing issues played an important role while prioritizing the candidate list. Along with general lines of the project, open-source tools have been used in the first round, wherever possible, and those supporting open APIs (e.g., WOPI) will be taken in the next integration steps.

2.2.4 On-demand data transfers

The work related to on-demand data transfers revolves around three tools, namely:

- RClone[20]
- FTS[21]/Rucio[22]

RClone is a rsync-type tool for about 40 different types of cloud storage. It supports standard Linux commands like cp, ls and mkdir and similar operations. It has been selected as the tool of choice for the end-user ad-hoc data transfers. RClone supports multi-threading transfer capability. It is also a lightweight client-side that enables easy integration with EFSS environment. Secure passing of user's credentials required some adaptations of the code, for which the CS3MESH4EOSC project has funded the implementation of a feature request by the leading RClone developers.

The FTS is an acronym for File Transfer Service which was developed at CERN and it is used for distributing data acquired from the Large Hadron Collider (LHC) over multi-tiered structure of data centres across the globe. It has been in use for more than a decade and has been adopted by sciences other than high-energy physics too. FTS is a service, typically running at the infrastructure side, so users and their applications or scheduled data movement and distribution processes interact with the service in order to trigger and control massive data transfers. FTS provides Python-based CLI, Python Cluent as well as WebFTS portal and Web Monitoring solution. It supports set of parallel data transmission and data handling protocols including WebDAV, HTTPs, GridFTP, xrootd and SRM. FTS also support transfer reliability measures such as data checksums and transfer retrial features.

Rucio is a data management tool that has been developed by the ATLAS LHC experiment. It manages the hundreds of petabytes of data that are stored all over the world. Also, Rucio has gathered followers outside of high-energy physics. Rucio is multi-layers set of services and libraries that can be used by users, applications, and services in order to management and perform operations on huge datasets. Supported functionalities include data storage, replication and providing unified interface for heterogeneous data infrastructures, through a set of pluggable interfaces to various storage systems as well as user users authentication and authorisation, access permission control, namespace organisation, meta-data handling, accounting, and quota implementation etc.
Rucio uses - among others - FTS for the replication of datasets. FTS-speaking sites encompassed in the T4.3 work are mostly based on Rucio with FTS used as the data transfer solution.

Overall, RClone was adopted as the starting point for T4.4 activities as is an effective weapon that can be used for the ad-hoc, interactive massive data transfers for 'regular' EFSS users including the long-tail of science. It can be also used for typical, real-life scientific collaboration or academic staff daily work and operation as well as in the typical business use-cases. Deploying and using RClone at EFSS sites does not require an expert knowledge, so it can be widely adopted across the federation.

[20] https://rclone.org/
[21] https://fts.web.cern.ch/fts/
[22] https://rucio.cern.ch/

In the same time FTS and Rucio are mostly used in large scientific collaborations or by the parties that that collaborate with large data repositories and infrastructures. Deploying and maintaining FTS and Rucio service require more substantial efforts and specific knowledge and experience related to this software and services stack and infrastructure ecosystem, so FTS and Rucio are more applicable as large scale transfer data solution in specific, large-science related and oriented use-cases. As such, FTS and Rucio will be taken up and integrated with ScienceMesh in the second half of the project.

2.3 Technical design and the integration work

Based on this analysis of user requirements and use-cased defined (described in Section 2.1) and the understanding of the features, functionality, and limitations of the existing technological components (described in Section 2.2) as well as taking into account the overall architecture of the federated mesh the application integration the development and integration paths have been determined and implemented. This section summarizes the design and integration work performed until M18 that led to preparing the proof-of-concept implementations and demonstrators of the applications integrated with the EFSS systems and the federated mesh. The demonstrators are described in Section 3 of the document.

The applications-related work has been organised in four technical tasks relevant to the application areas. The following points provide the details of the work in particular tasks.

2.3.1 Data Science Environments

The design and development work on Data Science Environments in Task 4.1 was organized in the two main streams:

(1) developing and integrating the JupyterLab extensions towards the federated EFSS;

(2) preparing the deployment approaches for data science environments at new sites based on packages prepared by CERN (SWAN: JupyterLab, CERNBox ownCloud, EOS).

Extensions of the JupterLab

Within the first stream of work, JupyterLab and sync&share systems integration has been achieved. This applies to both direct integrations of JupyterLab (the Data Science platform of choice) with particular EFSS implementations as well as integration of JupyterLab with the IOP, that in turn has been equipped with relevant plugins to sync & share systems.

The high-level view of the integration architecture is provided in Figure 3. It shows that the JupterLab platform uses CS3APIs to communicate with the mesh of sync & share sites that can provide the basic EFSS functionality for data storage and sharing but may also provide the computing resources (e.g., Spark clusters etc.) available through relevant plugins.

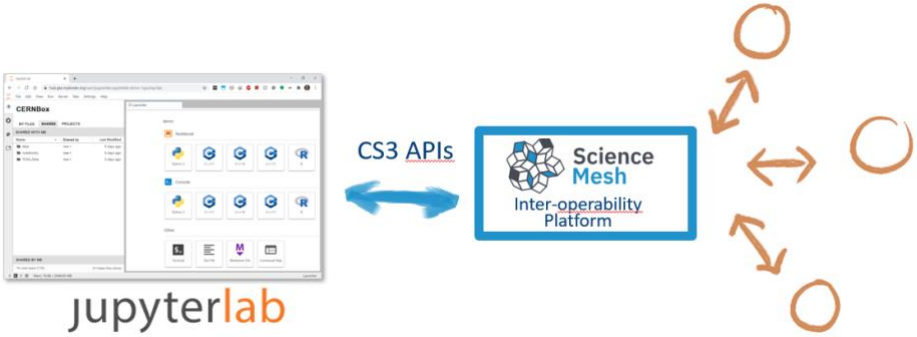

Figure 3. High-level view on JupyterLab integration with federated of EFSS systems

JupyterLab interface to federated sync & share systems has been implemented as a generic component, allowing the connection to EFSS and to the Science Mesh (IOP) using CS3 APIs.

Design and implementation-wise, the extension itself has been designed and implemented as the two layers: front-end and back-end. The front-end extension is implemented as the full client in JupyterLab. File browser has been equipped with the data sharing functionalities visualised as the sharing by/with tab and sharing button. In addition, the context menu presents specific entries related to the fact that objects are being shared. Also, the pop-up windows inform on the file sharing status in addition to usual data.

Example view of the integrated JupyterLab GUI is presented in Figure 4 below. On the picture the tab presenting shared objects has been highlighted and way if presenting the sharing information is visualised.

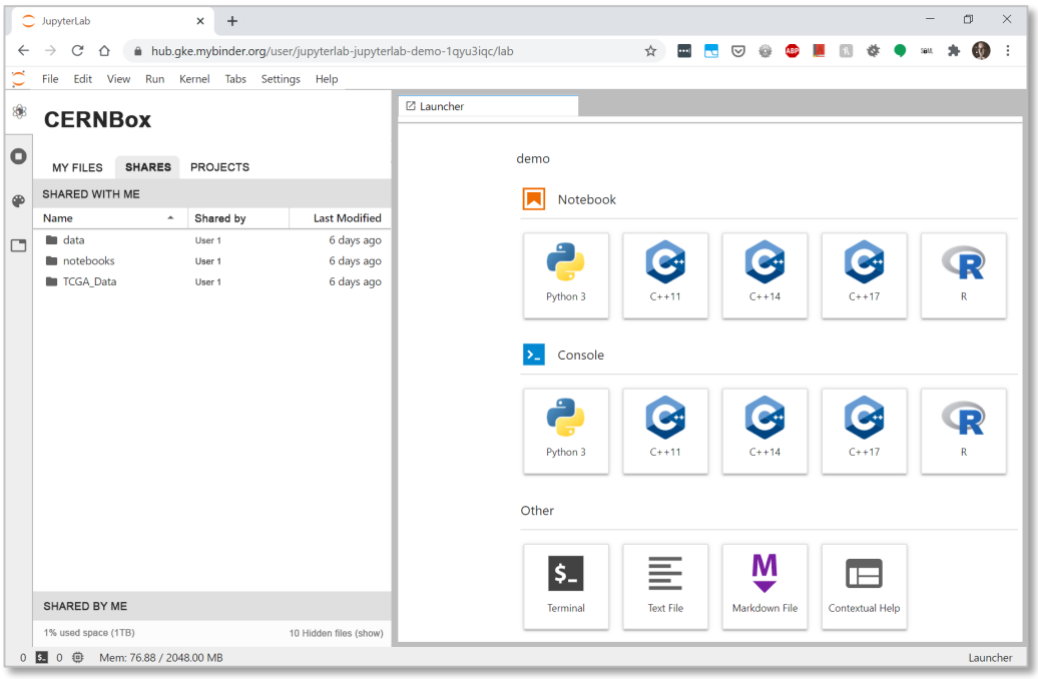

Figure 4. Example view of the JupyterLab GUI with the sharing extensions applied

Back-end part of the JupyterLab extension is implemented by replacing the ContentsManager and Checkpoints components of the JupyterLab as well as providing REST endpoints for integration with the frontend. They expose APIs for content operations, checkpoints operations and share operations. The ScienceMesh's IOPS is interfaced by gRPC (the CS3 APIs). The Figure 5 below depicts the backend architecture.

Figure 5.JupyterLab – CS3 API integration backend component architecture

Overall, on the functional level the JupyterLab extension worked out allows the connection of the JupyerLab to the EFSS and to the Science Mesh (IOP) using CS3 APIs, it makes the collaboration functionalities offered by these platforms available from within the JupyterLab interface, effectively making the extended JupyterLab an integrated one-stop-shop solution for dealing with notebooks preparation, data sets access and data processing. It also enables sharing functionalities across the federated sync & share systems, by leveraging on their OCM support. Parallel access (editing) to notebooks, one of the biggest limitations of the current Jupyter implementation, is on the roadmap (in planning phase). The implementation will need to address potential security issues related to this functionality

The generic JupyterLab extension was designed following the overall Jupyter architecture. It is easy to install and maintain and is portable between Jupyter and various storage systems that can be interfaced using the CS3APIs. The code is maintained in a GitHub repository[23] and can be contributed to upstream code, which is an element of the sustainability plan.

While the Jupyter extension provides access via its user interface, analysis run-times (i.e., Jupyter kernels that run independently) and the terminal interface still require the same consistent access and view of the storage layer (See Figure 6 below). For the latter, two approaches have been pursued by PSNC by using IOP as the target solution, while also attempting direct integrations[24].

[23] https://github.com/sciencemesh/cs3api4lab
[24] https://github.com/sciencemesh/filesystem-for-jupyter

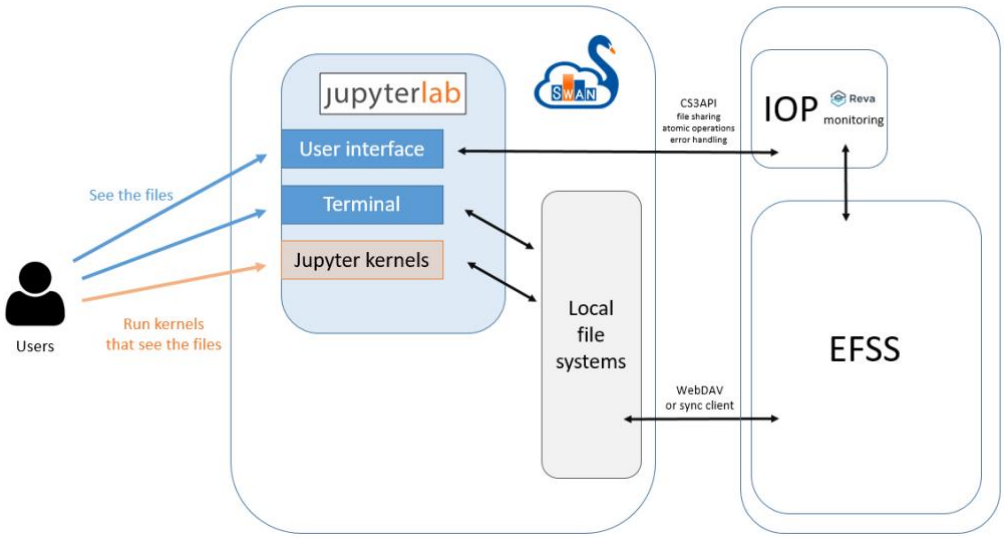

Figure 6 .Integrating JupyterLab with EFSS

Firstly, WebDAV interface of NextCloud sync & share system has been used along with FUSE-based DAVfs project in order to provide file system like access to data residing on it. While this approach provides the up to date and synchronised view of the EFSS's managed data and the namespace through the virtual filesystem interface, there are several limitations of the DAVfs system related to performance and scalability that should be solved in order to provide the consistent view of the datasets updated in real time as well as proper performance while accessing the actual datasets (especially if they're large).

Secondly, a similar functionality has been achieved by synchronizing user files between local user spaces in JupyterLab and NextCloud, using its sync-client solution. In this approach the Jupyter-provided view of the datasets is not always in real-time sync with the actual status of the storage space, however the performance limitations of the fuse-based DAVfs interface faced while performing large data access of write could be overcome in that case.

Overall, these two implementations have proven different performance characteristics, and thus, they fit different use cases and needs.

Deployment and external components integration approaches for data science environments

As for the second stream of work a set of deployment mechanisms for an integrated JupyterLab and EFSS has been prepared, improved, and evaluated.

While the JupyterLab plugin currently targets regular JupyterLab deployments, the plan is to make it also usable from the SWAN Data Science platform, which is developed by CERN. In this regard, an interesting case of technology transfer within the consortium included JRC experimenting with SWAN. CERN-provided SWAN Data Science platform is being ported and adopted to the characteristics of the JRC infrastructure as well as its management policies including specific security measures. Also, the data analysis workflows created with Jupyter notebooks have been compiled to Voilà dashboards[25] (routinely used by JRC) for broader usage by less experienced users. The IOP itself is installed in JRC in

[25] https://github.com/voila-dashboards/voila

a testing phase, it is not yet linked to the Jupyter environments, for the reasons that JRC operational specifics require an extra effort related to the security aspect, which is still under the discussion and analysis.

In addition to this, mechanisms for spawning data analytics and computing jobs in external processing environments, such as Spark, HTCondor, Kubernetes-managed container platforms and public clouds, have been evaluated at CERN, JRC and PSNC. The results of this analysis are being considered in defining the next steps to take in this task.

Concerning the deployment automation approaches, Ailleron and PSNC, worked on the of solutions which integrates “vanilla” versions of JupyterLab with ownCloud and Nextcloud-based sync & share.

The work on Data Science environment’s deployment mechanisms included also applying the containerisation concepts for deploying JupyterLab along with the extension worked out. In particular the extensions are being made available in different mediums in order to facilitate the deployment process (e.g., via containers – already done, pypi – at the planning stage).

Also, the mechanism for spawning data analytics and computing jobs in external processing environments, such as Spark, HTCondor, Kubernetes-managed container platforms and public cloud, has been integrated, deployed, and evaluated in CERN, JRC and PSNC. Results of the analysis has been taken into account in the Data Science application deployment approaches being worked out.

The semi-production instances of the integrated JupyterLab and sync & share systems along with the interfacing components have been developed and deployed at the participating sites (CERN, JRC, Ailleron, PSNC) and made available to collaborating user communities for evaluation and testing, including HEP (High-Energy Physics), Earth Observation (EO), Education-related use-cases and business users in Ailleron. More details on demonstrators are provided in Section 3.

2.3.2 Open Data Systems

The work in the Open Data System related Task 4.2 focused on designing the architecture and providing the reference demonstrable implementation and prototype for the integrated functionality of collaborative data preparation for open access (data curation, annotation).

The functionality developed support the users in data packaging using recognised data and meta-data formatting standard as well as triggering and performing the data publication through the packaged data deposit to the open repository.

Work performed until M18

Within the work performed until M18 the RO-Crate-based data description and packaging approach have been integrated into the selected sync & share system and research data management environment.

The overall integrated solution architecture has been designed including a model of integrating the RO-crate tools as part of online web services with a shared, remote filesystem based on EFSS views. Based on this reference architecture the proof-of-concept integration of the RO-crate-based description and packaging functionality with the EFSS system have been created, using the CS3APIs as the communication interface between the application-level functionality and the EFSS (and federation of EFSS systems in future).

For this proof-of-concept integration, the Sciebo sync & share service developed and operated by University of Munster (WWU), CS3MESH4EOSC partner has been used. The research data management functionality integrated into Sciebo, the Sciebo RDS service, has been adopted and extended. ScieboRDS provides services and tools for research data management and scientific analysis based on the Sciebo EFSS.

The Figure 7 below depicts the basic dataflow in the integrated solutions for WWU's research data management based on the Sciebo EFSS, Sciebo RDS and Describo Online.

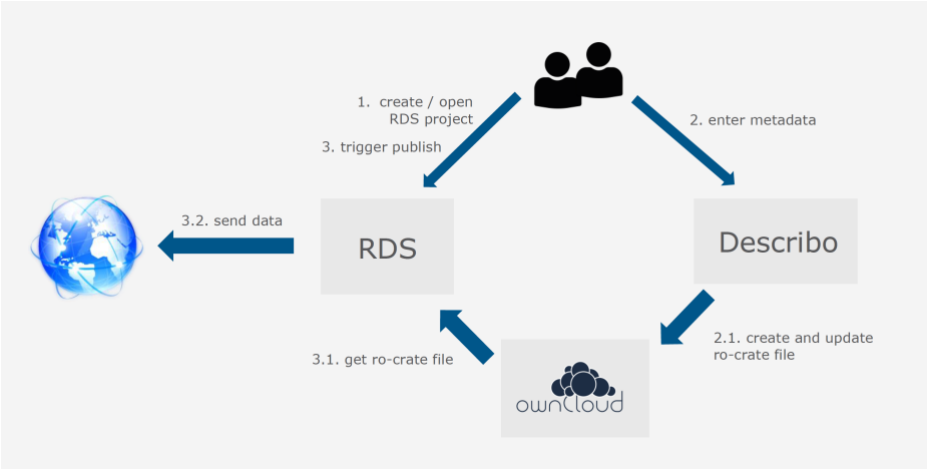

Figure 7. Basic dataflow in the integrated ScieboRDS and Describo Online solution

Development and Integration of the Describo Online tool with IOP and repository deposit functionality provided by Sciebo RDS resulted in providing a running prototype. It is ready for pre-production testing and functional evaluation. The detailed description of the demonstrator is provided in Section 3 of the document.

Broader application of the FAIR-related work results of Task 4.2

The prototype implementation of the integrated data description tool based on the Describo solution, data packaging format – the RO-Crate and the EFSS system constitutes a reference implementation of the FAIR-related integration architecture worked out.

Although until M18 the integrated sync & share and Open Data related functionalities of Describo and Sciebo RDS were mostly evaluated in the context of a research use-cases supported the University of Munster (WWU), however, the proof-of-concept implementation can be used for broader range testing and evaluation of the approach adopted and proposed for integrating FAIR-related functionality with EFSS systems**.**

Moreover, while the current implementation applied mostly to the tools implementing RO-Crate standard for archive packaging and ownCloud-based EFSS, the overall architecture provides a good basis for integrating other tools and services related to Open Data within the ScienceMesh and beyond.

This capability can be demonstrated using ScieboRDS solutions as an example. Structurally, ScieboRDS platform as well as other scientific data management platforms could be integrated with various EFSS systems and various data repositories. Figure 8 below, presents the example range of storage solutions and repository technologies that can be supported by WWU's Research Data Management platform.

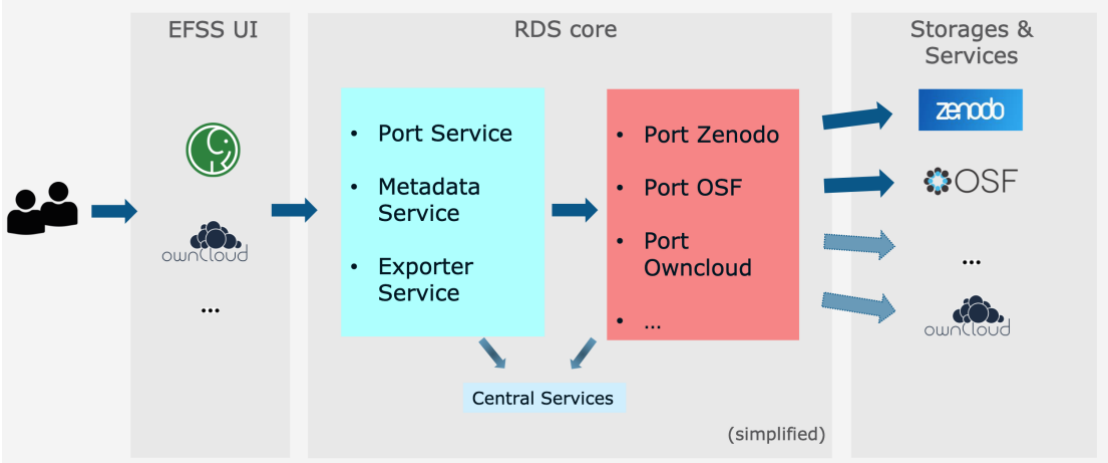

Figure 8. Integration options for ScieboRDS

In the generalised approach various types of EFSS systems could be used to handle data sets, collaboratively curate them as well as prepare and trigger the actual data publication. Similarly, various data repository solutions can be used as targets for the data publication and deposit including popular and promising solutions such as Zenodo can be integrated.

Future work

While in the current implementation of the ScieboRDS, Describo Online and ownCloud integration, it is the direct access to EFSS that is in use the work is being conducted on enabling Sciebo and ScieboRDS to interact with EFSS through CS3 APIs. Potential approach to using CS3 APIs for integrating the EFSS with research data management systems is presented in the Figure 9.

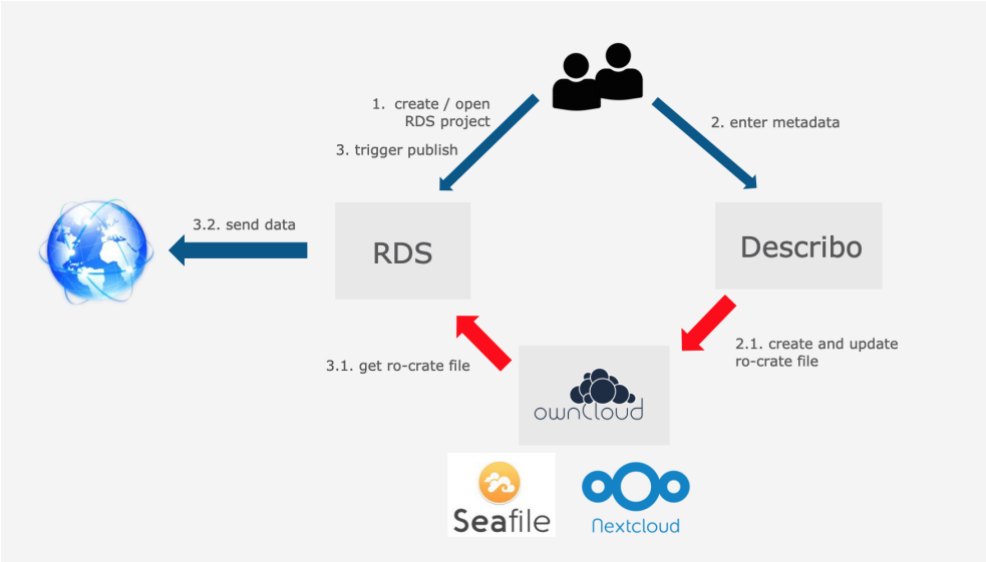

Figure 9. Potential usage of CS3 APIs and the universal interface for RDS integration

While this integrations model is currently under development it can be used for testing Sciebo integration with another storage back-ends.

Moreover, the remaining months of the project can be used to verify if the generalised integration approach worked out can be applied to integrating various types of EFSS systems with other implementation of the research data management functionality, data description, packaging and repository deposit functions. However, the scope of such a work along with the priorities definition has still to be discussed among Task 4.2 participants.

2.3.3 Collaborative Documents Editing Platfor3ms

The Task 4.3 work on collaborative documents editing platforms focused on designing the overall architecture for integrating online editing applications with federated EFSS systems as well as implementing the proof-of-concept integrations for selected products.

On the applications side, the work was targeted at enabling several document editing applications on top of sync and share systems. Candidate platforms identified included:

- office documents editing applications (CollaboraOnline, OnlyOffice),
- text/Markdown editors (CodiMD) abd rich text online editors (Etherpad),
- graphics editors (Diagrams.net); interactive whiteboard (Excalidraw) was considered;
- Latex editors (Overleaf) – were also included (integration hasn't been taken up yet)

Including such a broad spectrum of the applications in the proof-of-concept integrations has been decided taking into account the requirements and practices of key user communities served by the project partners. For instance, the Social Media-related R&D project RISE_SMA[26] that uses the WWU-provided Sciebo sync & share service is handling various types of documents and content including Social Media data (CSVs, graphs), questionnaires (CSV, txt, HTML), survey data (CSV, Excel, SPSS) as well as scripts in R, Json, Python, docx files etc. Notably, research and daily-business activities in many disciplines and organisations require a similarly broad set of content types to be shared and edited collectively.

The priorities have been set up to address in the first stage of work the online editors of office documents (Word format, Open Document Format) as the cornerstone of the online document-based collaboration. This functionality is highly demanded by most of the users.

Integration architecture

On the storage back-end side, most of the integration mechanics developed within the T4.3 task has been built on top of the WOPI protocol[27] (Web-Application Open Platform Interface Protocol). WOPI is the existing standard for online collaboration developed by Microsoft for its Office applications. It has been adopted by the industry - among others Collabora applied it for implementing the Collabora Online application. It is a high-level protocol that implementing operations vs the application server needed to realise the document-based collaboration e.g. file access, file storage, shared locking etc.

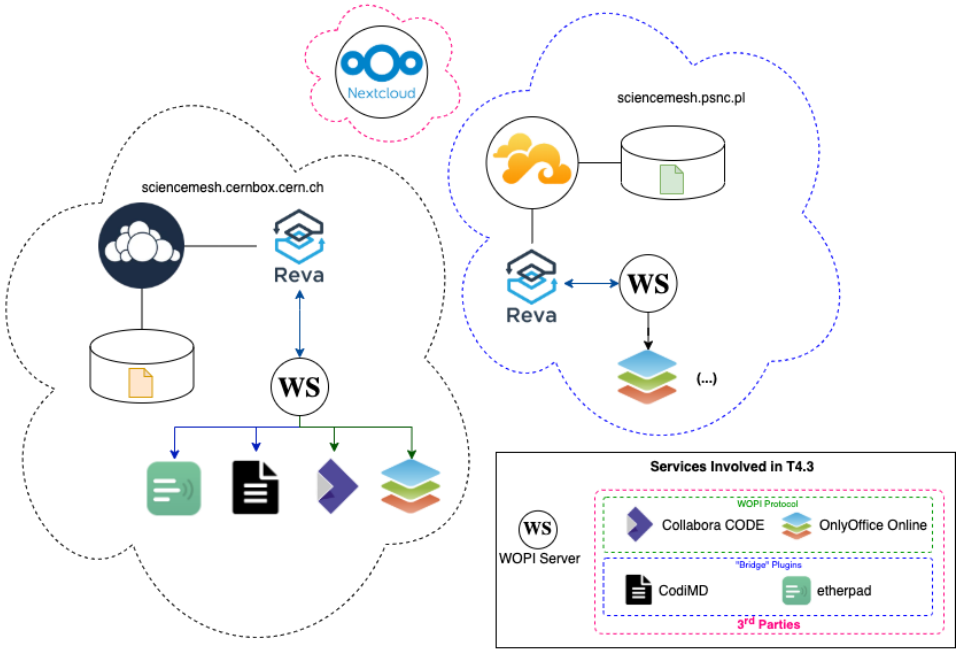

Figure 10. Integration architecture of EFSS systems and collaboration editing applications

The architecture for integrating the sync & share solutions with the collaborative editing applications by using the WOPI protocol extensively as the intermediate communication layer is depicted on the Figure 10. Figure 10 presents several possible approaches for integrating EFSS systems with

[26] https://social-media-analytics.org/
[27] https://docs.microsoft.com/en-us/microsoft-365/cloud-storage-partner-program/online/

collaborative applications based on specifics of the EFSS implementations and collaboration platforms.

While not all integrations options - as depicted - have been taken up by Task 4.3 so far, the overall architecture and integration approach worked out and visualised on the Figure provide a vision for future integrations.

Integration work performed

Among integrations actually performed for office-like applications the Collabora Online case was relatively straightforward as Collabora supports well the WOPI standard. OnlyOffice integration in turn had been suspended due to the fact that the company behind this product did not have plans to enable WOPI support and made it available only in the latest release (August 2021); the integration work will restart right away, as the platform is ready to get further WOPI applications on board. In general, integration of OnlyOffice without the product-side WOPI support would require extra efforts on the project consortium side, which caused this work to be ceased despite the high demand of OnlyOffice integration.

Other collaborative editing applications have been integrated based on the *WOPI Bridge* extensions, now integrated in the main WOPI server. WOPI Bridge has been developed by CERN for the CERNBox and adopted and extended for the purposes of CS3MESH4EOSC. Such extensions enable triggering file updates in the storage system (e.g., EFSS) in response to notifications received from the online editing application.

The editing applications integrated using WOPI Bridge include CodiMD. CodiMD has been instrumented with a webhook that catches up the file modification notifications. The solution that has been deployed in the production CERNBox installation is currently in intense use at CERN. Once the deployment is fully stabilized, it is planned to push the changes upstream to the open source CodiMD repository, in coordination with the company (HackMD) supporting the product.

Similar approach has been adopted for the EtherPad integration; however, this work is still at its early stage. Similarly, the resulting EtherPad plugin will be contributed upstream, where an ecosystem of plugins for several integrations already exists and is actively maintained by the open-source community.

Overall, the proofs of concept performed evidenced that the collaborative applications integration model worked out is lightweight and reusable. The WOPI standard-based mechanism for updating the state of files residing the EFSS system bound with the editing applications through the WOPI bridge, and the application-level hooks proved to be a reliable and stable solution (see the CodiMD-CERNbox integration used in production).

This model and architecture can be reused and applied for enabling integration of new collaborative editing applications to the ScienceMesh

Possible future work

The task partners are looking into possibilities of integrating online graphic editing applications such Diagrams.net (previously Draw.io) and Excalidraw. While there are plans to integrate the former, the latter is not currently offered as an on-premise solution but it might change in future. Therefore, that part of work in Task 4.3 has been suspended but it can be restarted when the on-premise deployment model for Excalidraw will be available.

Overleaf has been identified as an interesting application, however, its sheer complexity has so far

hindered an integration with the ScienceMesh. Outsourcing this work to the company behind Overleaf seems to be the most viable option. However, discussions with this potential partner and internal discussions on the collaboration are still to be conducted.

Several detailed aspects of the integration of the collaborative editing applications with EFSS systems are still to be investigated. They include providing application registries enabling the EFSS interfaces to inform users on the available editors that can handle certain types of files as well as improving the collaboration token management mechanisms.

2.3.4 On-demand data transfers

The Task 4.4 work focused on enabling sharing and accessing large files across the sync & share systems federations. While EFSS systems are capable of handling the sharing and exchange of small files, handling multi-gigabyte or terabyte files and objects typical to large scientific datasets requires improvements, addressing the aspect of transferring the data among geographically dispersed federation sites. The work until M18 focused on designing the architecture for EFSS systems and large data transfer tools and services as well as proof of concept implementation of the design worked out using well-established data transmission solutions.

The overall approach was to equip the sync & share systems into ability to trigger the data transfer on the user's demand from or to other sync & share system or external storage system such as big scientific data archive or data access endpoint in the R&D project infrastructure. This functionality was enabled on CS3API level and in the sync & share application user Web interface.

Sync & share systems taken up for the first round of integrations included ownCloud installations in SURF and CERN. The development and testing work are also being extended to sites of the LOFAR project including PSNC and FZJ, which will enable using large data sets from the Radio Astronomy discipline for testing data transmission efficiency and reliability.

There is a vital need identified by the LOFAR community for an easy to use, flexible and dynamic but at the same time scalable and reliable data sharing platform for research collaboration. Existing LOFAR data management procedures enable acquiring surveys' data from the geographically dispersed instruments and collect them centrally for pre-processing as well as distribute the data into archives - i.e., the Long-Term Archive (LTA) sites in a planned and schedule way.

However, the community still lacks the platform for the ad-hoc data-based collaboration on the data analysis that might require accessing selected large datasets that could be best performed if these datasets are made available locally, in the online storage system as opposed to geographically distributed Long-Term Archive where data typically reside on tapes.

EFSS systems with their easy-to-use interface are the ideal candidate solutions for such an opportunistic data-based collaboration, which has been also concluded by the LOFAR users. Moreover, with extra functionality added such as those integrated within CS3MESH4EOSC's WP4 the EFSS systems may provide a comprehensive platform for data analysis, editing and publication.

However, in order to ensure the data locality and efficient exchange of large datasets, the integration

of the data transfer mechanisms into the EFSS systems is needed.

The transfer mechanisms integration work in the first 18 months of the project applied mostly to RClone[28]. FTS[29] and Rucio[30] integration has also been designed however implementation is not yet performed. Integrating FileSender[31] has also been discussed and plans have been made to tackle the FileSender case in future, after the RClone-based approach gets stabilized and tested.

The overall workflow designed for the RClone based transfers triggered from EFSS is depicted in the Figure 11 below. The transfer process is initiated through the Reva API by the user CLI or GUI application. In response to it the RClone process is spawned or the RClone deamon is contacted in order to trigger and conduct the actual transfer. RClone has the local access to the storage system back-end for optimal transfer performance which is possible due to RClone broad support for various storage types including filesystem-based systems, object stores etc.

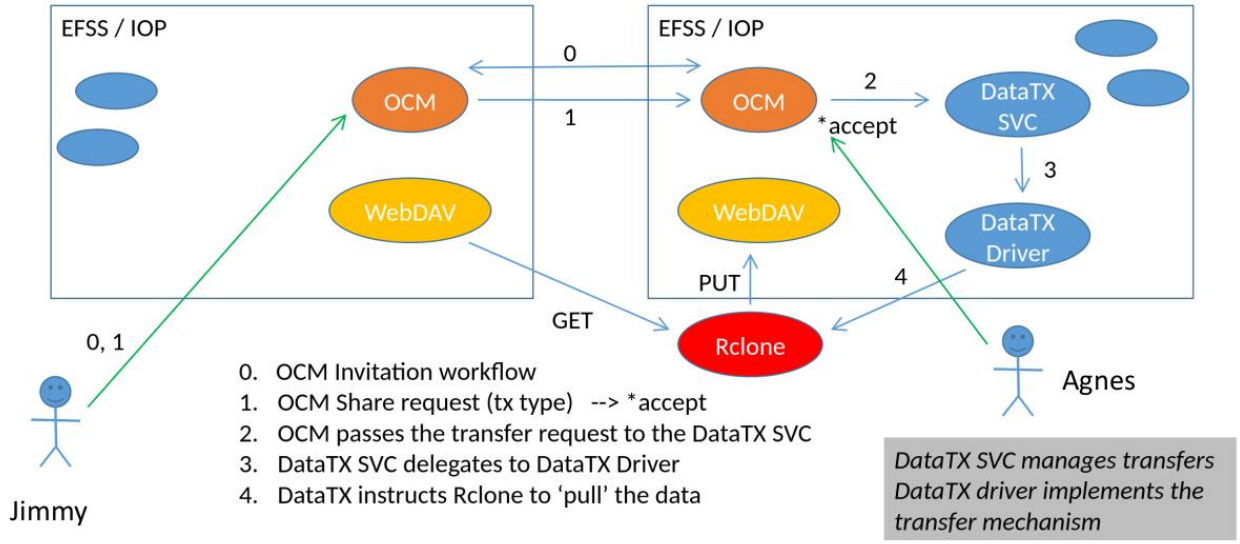

Figure 11. Basic workflow in the RClone-based data transfer among EFSS systems

[28] https://rclone.org/
[29] https://fts.web.cern.ch/fts/
[30] https://rucio.cern.ch/
[31] https://filesender.org/

For the time being the target system is considered to be another EFSS system as the OCM-based mechanics exist that enable establishing mutual EFSS servers and users trust relationship.

The RClone based integration included working out the overall data sharing and transmission workflow that enables creating “transfers” i.e., the shares of the “transfer” type.

For establishing the trust between users and sites while data transferring the sharing invitation workflow based on OCM is used. The basic invitation workflow was extended with steps including accepting the transfer, indicating the transfer destination path, and triggering the RClone service to perform the actual data transfer. Getting the transmission status and cancelling the transfer is also possible in the extended data transmission and sharing model.

The pull model has been adopted, where the receiving site is an active party, determines the transfer details such as target data destination as well as performs the actual RClone transfer. Such approach is compliant with the logic of processing the sharing invitation workflow.

The full interaction diagram depicting the data transfer in the pull model is shown on the Figure 12 below (it is also available at: https://surfdrive.surf.nl/files/index.php/s/1dxkBCzCH40XYhF).

Push model has been analysed however its implementation has not been taken up, as the main focus as far was on enabling data transfers among the EFSS systems. Push model would comply with logic of the other data transmission mechanisms considered in the task such as FTS/Rucio, as these systems implement the logic of managing the data transfers including transfer initiation, status control, retry of failed transfers etc.

Implementing data transfer on a behalf of particular users sharing the data requires passing the user credentials parameters to the instances of RClone fired up. For improved security modification of the RClone has been conducted to avoid storing user credentials in the configuration files of RClone daemons. This work has been outsourced to the RClone lead developer. Results has been contributed to upstream RClone code. The increased the overall data transmission security with RClone in the ScienceMesh use cases and beyond.

Current data transmission mechanics implementation has been mostly tested in the development and testing environment at SURF and at CERN. Extending the evaluations is being prepared for PSNC and FZJ Juelich in the context of the LOFAR use-case. The test plan that assumes transferring large data sets (100s of GBs) among Netherlands, Germany and Poland. It also requires involving IT support group working at Astron (the organisation that coordinates the LOFAR project) in the first run, as well as the actual end-users of the LOFAR infrastructure and data collected in LTA, i.e., the radioastronomy scientists in Netherlands, Germany, UK and Poland.

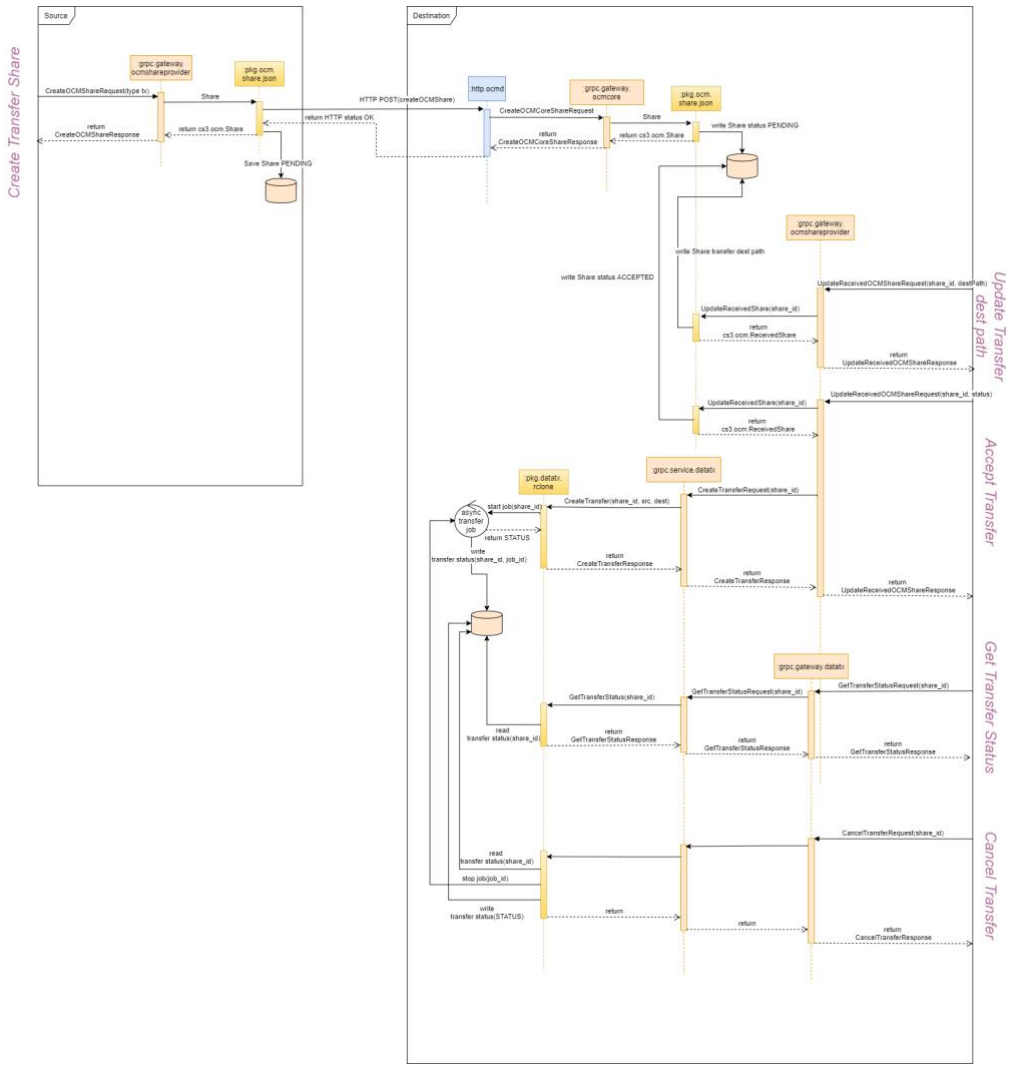

Figure 12. Detailed interaction diagram of the data transfer in the pull model

Future work

Further development work planned includes integrating the code implementing the RClone-based transfers with the IOP/Reva platform. This integration is still at the PoC state. Production readiness of the solution still requires further testing, hardening and scalability analysis and improvements.

The use-cases covered as far in Task 4.4 apply mostly the data transfer among the EFSS systems. This among other enables re-using the general OCM-based data sharing workflow as well as applying the known user and partners identification and authorization information in order to configure and perform the secure data transmission.

Addressing the transmission between the sync & share site and the external storage system is still under discussion and will be a matter of work in the remaining 18 months of the project.

3 ScienceMesh Applications – demonstrators

This section overviews the proof-of-concept solutions (PoCs) and the demonstrators for solutions prepared in particular application areas including Data Science environments, Open Data Systems, Collaborative Documents and On-Demand Data Transfers.

These PoCs and demonstrators has been prepared within the use-cases and requirements analysis as well as integration architecture design and applications-IOP integration work, described in detail in Section 2. While Section 2 provides the technical details of the design, integration work along with its current results and future plans discussion, this section focuses on summarizing the functionality of the PoCs and demonstrators prepared.

These early versions of the solution has been as far (until M18 of the project) used for the purposes of internal work status demonstration, user-targeted presentations aimed at collecting their feedback and involving users in the evaluation and co-development of the solutions and integrations worked out as well during the EC mid-term review of the project.

3.1.1 Data Science Environments

Within T4.1 two demonstrations of the integration work results has been prepared for two use-cases defined in the scope of the task.

First demonstrator is a presentation of the JupyterLab plugin developed and its possibilities has been prepared in a form of the video recording. It is related to JupyteLab integration with IOP performed based on the ownCloud as an EFSS system in the context of HEP and EO use-cases at CERN and JRC.

Note however, that IOP at JRC has only been implemented in a test environment, and it's not yet used for actual collaboration in the production environment - because of the security and data access constrains specific to JRC status and nature of the data (it was explained in previous sections).

In this video, the plugin is presented that enables connection of the JupyteLab to the Reva-based IOP server through GRPC protocol and interact with the remote sync & share storage systems through it. This Jupyter driver handles the remote file system access, remote workspace's content presentation and basic operations on the files in the remote storage space. The demonstration video contains an example of creating a new notebook file from within the JupyterLab environment and sharing it with another user within EFSS systems federation, who gets assigned a permission to edit the shared notebook. The video also shows that the second user can open a shared notebook file and he can run and modify it for both users. The changes are synchronised and the notebooks content view is made consistent across local and remote JupyterLab and storage environments, through IOP.

Second demonstrator has been prepared in the context of the education-related use-cases at PSNC. It has been used both for demonstration and testing purposes as well as the scalability benchmark for the Data Science (JupyterLab) - EFSS (NextCloud) storage back-ends integration and content synchronisation mechanisms worked out by PSNC.

The common datasets access from both JupyterLab and from the level of CLI and kernels running in the computing environment is implemented by using synchronization of the Jupyter storage space back-end and NextCloud-based EFSS storage space.

Authorization of JupyterLab to access files stored in NextCloud is conducted with use of OAuth-like Login Flow[32]. Two instances of the entire stack has been deployed by PSNC within its two existing learning platforms for high-school students, who work with the applications on their IT-related classes and extra-curricular courses, including the PIONIER Research & Classroom platform[33], managed by PSNC, and the Up2U platform[34,35], managed by GÉANT Association.

3.1.2 Open Data Systems

For the components and solutions addressing the Open Data Systems use-cases two levels and approaches to demonstrating the Open Data capabilities integrated within CS3MESH are being conducted: the one related to the Describo Online itself and the other focusing on Sciebo RDS.

In the first dimension, the meta-data annotation and file-set packaging solution based on the Describo Online tool is being constantly demonstrated, tested and co-developed. Describo-Online[36] is a part of a larger suite of tools Arkisto[37] which implements a complete data preservation ecosystem. Arkisto is in live use within PARADISEC community. This community assists the Arkisto team by co-developing the toolkit feature set and providing feedback. Through this live use and co-design, the Describo Online is itself validated as a useful and usable tool in a collections research context.

It the second dimension, the demonstrator of the Describo Online developed in collaboration of Technical University Sydney and PARADISEC community is integrated with the Sciebo RDS service and platform for managing research data developed at WWU, based to WWU's Sciebo EFSS. ScieboRDS itself is developed from both CS3MESH4EOSC and the German Research Foundation funds. Its ScieboRDS started based on the North Rhine-Westphalian research council funding, that too, insisted on co-development as the preferred mode of project implementations. Thus T4.2 work re-uses these established relationships.

As a result, the nesting of DescriboOnline-in-ScieboRDS-on-EFSS that t4.2 is produced as a demonstrator and deployed at WWU as well as validated *in situ*, by the research group around Lennart Hofeditz, focusing on Social Media research in the domain of the Crisis Response.

The demonstrator presented on the EC review explains how the researchers, in particular users of Sciebo EFSS system can use the ScieboRDS invoving Describo Online tool for preparing their data for publication and repository deposit using the means of the service integrated within his EFSS. User's data are sync among the researcher desktop storage and Sciebo EFSS service. For depositing the selected dataset to the Zenodo repository, the ScieboRDS Zenodo deposition features is used. During the preparation step the basic meta-data have to be provided as Zenodo does not accept the deposits without at least the minimum set of meta-data. This is made using the Describo tool. Based on the meta-data provided the extra file is created including all the metadata formatted in the JSON. The meta-data file is kept along with the dataset and can be used for actual data deposit using dedicated tools such as ScieboRDS. The demo shown on the EC review involved creating the new project within Sciebo RDS and takes the meta-data prepared upfront during the deposit to Zenodo.

[32] https://docs.nextcloud.com/server/latest/developer_manual/client_apis/LoginFlow/index.html#login-flow-v2
[33] https://classroom.pionier.net.pl/
[34] https://up2university.eu/,
[35] https://up2u.readthedocs.io/interoperability/jupyter_fss/
[36] https://arkisto-platform.github.io/tools/description/describo-online/
[37] https://arkisto-platform.github.io/

Overall, the demo presents the added value of the integrated solution developed within the T4.2. The set the tools and the workflow worked out enables easy deposit of the dataset from the ownCloud-based Sciebo, throught the Sciebo RDS and Describo tool up to the Zenodo repository. The integrated workflow is in fact a simple procedure not involving any cumbersome manual steps.

The demo also shows the procedure for allowing ScieboRDS application to access datasets stored in ownCloud (Sciebo) as well as granting permissions to and connecting accounts in various repository systems for ScieboRDS application in order to enable automated and integrated acquisition of the source data from the sync & share system and data deposit to the connected repositories.

3.1.3 Collaborative Documents

T4.3 enables the users to collaboratively edit Office-like documents spreadsheets and presentation editors: OnlyOffice, Collabora Online as well as light-weight format documents (Markdown, plain text) through editing applications such as CodiMD, Etherpad etc. On the radar is also integration of the Latex editors (Overleaf), and graphical editors (Diagrams.net, Excalidraw).

Until M18 prototypes and demonstrators of two main integrated documents editing applications has been prepared. Both collaboration applications were presented in the demos during the EC review.

The CodiMD demonstration that has been showed on the EC review, based on the Sciebo EFSS system and the use-case used for the Open Data Systems demonstration. The collaborative editing demonstrations were conducted for scenarios assuming that connection and trust relationship among two users has been already established (e.g., by using an invitation workflow). The demo showed the simultaneous editing of the same Markdown based document using two CodiMD editor applications instances running in parallel in two separate browsers. MarkDown document changes have been tracked and represented at both sides of the editing pair. Also, the rendering of the source document in a graphical form has been updated on-line while one of the users edited the source ‘code’ of the document. This includes the textual fragments as well as charts update shown in the document rendering window of one user’s browser based on edits performed in the source document by the other user. This presentation demonstrated real-time co-editing capabilities of the CodiMD application sharing the MarkDown document over the ScienceMesh federation.

The demo presented at the EC review applied to the CodiMD application. Similar capabilities are to be provided for other applications, however this requires GUI integrations. The GUI integration is - as far - available only for ownCloud Infinity Scale used at CERN and it should still be prepared for the web interfaces of EFSS based on ownCloud 10, NextCloud and Seafile. Another limitation is that the prototype for office documents editors integrated with the mesh communicate via the Reva hosted at CERN.

As soon as the GUI integrations are available for document editors it will be deployed in other sites, starting from - within the scope of further work withing T4.3 - Sciebo services of the WWU Munster. In the latter case the purpose will be to make this functionality available ASAP for the RISE_SMA use case of the researchers around Lennard Hofeditz, for gathering feedback and conducting co-design.

3.1.4 On-demand data transfers

Among the data transfer applications within the T4.4 development roadmap, RClone-based data transfer solution has been deployed in the testing environment of ownCloud EFSSs at SURF and CERN. They mostly use the CLI tools for the sharing initiation and data transfer control.

Similarly, the EC-review demonstration focused on presenting the overall transfer invitation and transfer workflow implementation using two distant sync & share systems and Reva based IOPs steered by the CLI interface commands. The demo explained particular steps of the user-initiated data transfer including creation of the 'transfer-type' shares, transfer acceptance and monitoring.

The prototype for the massive data transfer based on the FTS/Rucio will be demonstrated and tested using the HEP data (mostly at CERN and SURF) and RadioAstronomy data related to LOFAR use-case involving data transfers among EFSSs, the instruments and long-term archives in SURF, FZJ and PSNC.

4 Auxiliary activities

4.1.1 Common application requirements and deployment mechanisms

Task 4.5 is dedicated to harmonize, collect and document the requirements of applications integrated within application tasks of WP4 related to CS3 API extensions and versus the federation sharing protocol (OCM) developments that are conducted within WP3. Task 4.5. is also synchronising and documenting the approaches to applications deployment, management, operational and usage monitoring, accounting in the federated mesh based on the federation design, procedures and management mechanisms worked out in WP2. These aspects are related also to security, authentication, authorisation and access control (among other through support of federated AAI). The T4.5 work takes also into account the WP5 efforts on reporting and communicating to the World on federation usage, integration, and development progress as well as WP5-driven involvement of the early adopters and testers and stimulation of engagement with users through the dissemination activities.

Design and implementation coordination

Task 4.5 was particularly active within the first 9 months of the project, with the aim to feed requirements to the IOP and OCM extension design processes in WP3. The goal was passing the application-specific needs to be implemented in the federation platform identified in WP4 as well as to establish regular communication among federation and application developers.

All application integration discussions, analysis, design, and initial implementation work have been coordinated within WP4 in conjunction with WP3. For WP3 and WP4 synchronisation, a series of direct applications and federation developers' meetings have been organised. Other communication channels have also been used such as Gitter-based chat channels for interactive messaging as well as using common repositories or enabling mutual access to code repositories and documentation related to IOP and application integration codes.

By M9, a consistent set of requirements for the IOP had been collected and passed to WP3. While this was not an official milestone planned in the project description of work, this activity provided the important basis for the design and implementation of application-specific extensions in the IOP. After M9, synchronisation of the requirements, design, and implementation among WP4 and WP3 continued to happen regularly.

In several cases, developers - including e.g., the Ailleron team - participated in the design
and code development work both in WP3 and WP4, focusing on IOP extensions, which helped in synchronization of works on the interoperability platform and the applications.
Thanks to these efforts taken up on both sides: the application side (WP4) and the federation side (WP3), the development and integration of several prototypes and demonstrators for the application workflows to the demonstrable solutions, including the federated use-cases, have been possible. Some of these demonstrators are already running in the testbed and is as of now being tested and evaluated by early adopters.

Application environments deployment and code management

Within Task 4.5, the approaches to management and deployment of the software packages that implement and integrate the application workflows within the EFSS systems have also been worked out. This was necessary in order to support the actual development work, i.e., enable deploying the development and testing environments without too manual hassle, and to ensure the continuity and sustainability of the development environments at sites.

Deployment mechanisms enable also large-scale testing and facilitate adoption of the results of the integration work beyond the project consortium and partners sites. They help create evaluation and testing environments and avoid steep learning curves while trying to port, deploy, and run the federation and application components in user environments.

The WP4 partners managed converge the approaches to software deployment contributed by different sites and tasks, so that a consistent set of deployment tools was put in place.

For instance, within T4.1, various approaches to deployment and virtualisation for Data Science-related JupyterLab environment and its associated components such as notebooks and processing environments interfaces (e.g., jobs spawners to Spark, HTCondor etc.) have been considered and tested. Importantly, while the research partners participating in the task (CERN, JRC and PSNC) have different internal infrastructure management policies and security rules, the overall approach converged to a common vision and practice.

At CERN, the approach to software deployment is based in Kubernetes-orchestrated containerization and is applied both to the JupyterHub engine itself and the containers that implement the actual data processing and run the user sessions. SWAN is deployed using
Zero-to-JupyterHub upstream Helm charts, with additional configuration applied. JRC deploys the CERN-provided SWAN charts and is in the process of migrating the computing infrastructure orchestration towards Kubernetes as well, which proved to offer promising features and flexibility compared to the HTCondor- and the Docker Swarm-based approaches used so far. PSNC uses vanilla JupyterHub, instead of SWAN, but relies also on Zero-to-JupyterHub and the OpenShift platform, which is a Kubernetes-derived container orchestration platform by RedHat, that are automated by Ansible scripts. Ailleron prepared a set of scripts and configurations for Terraform to automate bootstrapping of the infrastructure, which also enables keeping the development and testing environment available and portable in case outages happen.

Overall, by M18, the work on the deployment and software management related solutions
has been completed and the approaches and technical components has been documented.

These were necessary components for supporting the development and internal testing process as well as enabling porting the integrations outside the consortium. The work on application usage and availability monitoring as well as accounting, auditability in local sites and across the federation mesh of applications and sync & share systems will be taken up during the remaining months of the project.

4.1.2 Evaluation of the dynamics of collaborative users communities and governance of the CS3Mesh consortium

Task 4.6 focused on analysing the general user communities-driven and service operators-side factors that may impact the design, implementation, integration and, finally, the adoption of the applications developed under WP4. Moreover, task 4.6 also contributes to planning the long-term sustainability and usage of the Science Mesh in general.

These activities were conducted, taking into consideration the wider lens of the fragmentation of the cloud service ecosystem and the current drive of policymakers and the scientific community towards FAIR data and Open Science. Moreover, to ensure that the project's results are exploited and used to their full use, we need to ensure that key stakeholders are aligned. Figure 13 below shows the view of the main stakeholders for CS3MESH4EOSC.

Stakeholder	Pain point	Value Proposition	Next steps
End-users and research communities	Difficulty in sharing and collaborating on data across researchers with different providers	The mesh enables sharing and data collaborations with ease	Conduct user engagement studies
Institutional operators of services	Vendor lock-in due to lack of interoperability across cloud vendors	Interoperability enables providers to choose flexibly the vendor that serve best their needs	Conduct adoption studies to understand the challenges with joining the mesh
Cloud vendors	Difficulty to compete and attract customers due to natural monopolies	Create a more competitive cloud storage landscape where companies do not lead just due to high switching costs	Conduct studies to understand their hesitations towards interoperability and explore how they can benefit from joining the mesh
Non-commercial software developers	Difficulty in developing software across different software systems	Enable ecosystem for the development of innovative apps for research	Exploratory study on potential research apps that can be developed through the mesh
Policymakers and citizens	Inefficiencies caused by research output not being reused	Foster reuse of data to solve societal challenges	Conduct cost-benefit analysis of the mesh

Figure 13. Main stakeholders for CS3MESH4EOSC

As the primary audience of the Science Mesh are the researchers, we explore the challenges and opportunities in CS3MESH4EOSC considering factors such as the different sync & share solutions in the market, their specific costs and benefits, usage and data management policies and norms across heterogeneous scientific disciplines. To address this, until M18, we have conducted 15 qualitative interviews and case studies with the early adopter community in the context of WP4-provided application integrations with the FAIR-related functionality and processes. In particular,

Integrating this data with results of internal discussions and brainstorming conducted with WP4 and with other project partners, various publications and conference presentations have been presented. The first publication entitled "FAIR Data through a Federated Cloud Infrastructure: Exploring the

Science Mesh" was presented and published at the European Conference on Information Systems[38]. Through our analysis of the Science Mesh, we frame FAIR data as a collective action problem permeating across two levels: researchers downstream and cloud service providers upstream. On one hand, the scientific community would be better off if researchers made their data FAIR, yet misaligned incentives hinder this. On the other hand, cloud providers are also not incentivized to pursue interoperability with other providers, despite its benefits to users. By addressing these two dilemmas towards FAIR data, the Science Mesh promises to unlock new ways of collaborating through frictionless sync and share, remote data analysis and collaborative applications.

Another study preliminarily entitled "Science Mesh: Unlocking novel modes of academia-industry collaborations" is exploring the impact of the infrastructure on industry has been presented at the CACM Europe Workshop[39]. Our review suggests that industry can easily collaborate and leverage data from their academic counterparts. Also, they can develop apps to serve research needs and enable new collaborative functionalities. In this line, we aim to exhibit cases of how organizations can leverage data in different sectors; approaches for data management and analysis to help organizations convert available data into valuable information and knowledge.

Moreover, the opportunities provided by the mesh to scientific collaborations have been presented at the Open Innovation in Science Conference[40]. In this presentation, we provided examples of the potential of FAIR data by enabling novel research workflows in fields including astronomy, high energy physics, earth science and the humanities.

These papers contributed to increasing the awareness of the importance of topics such as the need for ensuring the users' control on the data processing, storage, and preservation policies in the scientific community. Giving the control on the data back to the owners and curators is one of the key factors stimulating adoption on the on-premise data management solutions offered by CS3MESH4EOSC. Clear usage policies are crucial as opposed to public clouds where policies imposed on users are often unclear or might be abusive (i.e., data security and privacy aspects). The results of these studies also led us to understand what researchers expect from the Science Mesh and identify the contradicting perceptions between the project team and the potential users.

ESADE's work on the analysis of the opportunities and challenges show that for the majority of the research community, the biggest challenge is that cloud storage tends to not be a big priority in their workflow. They tend to just store their files depending on what they already use for their personal storage (such as Dropbox) or use the service provided through their institution. When they need to collaborate in document editing, for instance, they tend to send files via email or just use Google Docs due to its ease. Thus, to ensure that the mesh would be used by researchers, the most important is to ensure that it is user-friendly and that it is intuitive so that it can easily be used by researchers. The advantage of the mesh is that it runs in the background of the current cloud storage services, and thus can be used by researchers without necessarily having to know that the mesh exists. As such, getting adoption from the researchers would be straightforward. Nonetheless, establishing good relations with institutional providers would be key to ensuring that it gets properly integrated into their cloud storage service. The results of this work are related to proper addressing of the FAIR-related aspects in and by the CS3MESH4EOSC consortium provided also important feedback to the high-level federation design (WP3 space) as well as to data and services management policies adopted (WP2), and to application integration design and implementation work within WP4. The knowledge and

[38] https://aisel.aisnet.org/ecis2021_rip/14/
[39] https://sites.google.com/view/cacmeurope/schedule
[40] https://ois.lbg.ac.at/en/research/ois-conference-2021

material collected within 4.6 activities have been taken up within dissemination and communication activities of WP5.

In the next stage of work particular application areas related to WP4 tasks and applications will be taken up including Data Science environments, document editing platforms and data transfer solutions enabling access to remote repositories of large volume scientific data.

Having concluded this phase of the study where ESADE interviewed project team members and the general scientific community, there are still many important questions that need to be addressed in order to ensure the success of CS3MESH4EOSC and its long-term impact. Figure 14 summarizes the future working lines.

For one, we still lack insights into the real value of FAIR data in the business context. While the practical integration and evaluation work for scientific data sets and use-case has been conducted within the T4.2 and the FAIR aspects related strategy for scientific and research data is being worked out within T4.7, the data commercialisation aspects have not been taken up to large extent.

Moreover, we lack understanding of the requirements necessary for scientists and research organizations towards adopting open science and FAIR data practices. Undoubtedly, making FAIR data a reality requires major changes in the practice of many research communities, institutions, and funders. It is therefore essential to develop organizational readiness framework that explains the resources and capabilities necessary to leverage FAIR data.

Thus, in the remaining months, ESADE aims to understand data-driven design and management practices and co-design processes of value creation from FAIR data. Furthermore, little is known about how new initiatives such as FAIR data movement influence information-sharing behaviour of industrial and academic scientists. As such, ESADE's goal is to compare grand initiatives by the European Union across academic institutions and industry (e.g., Science Mesh and GAIA-X).

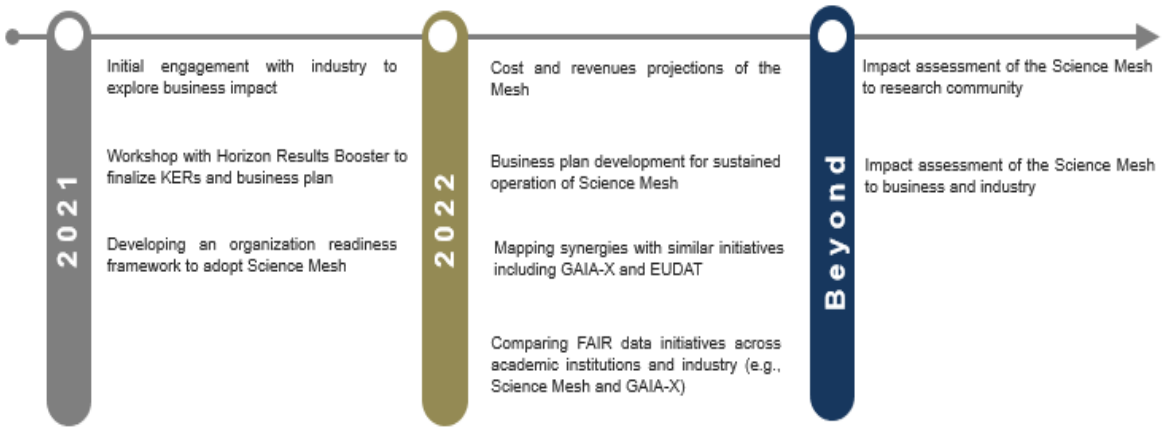

Figure 14. Overview of future working lines

5 Summary

In this report we summarized the design and integration work performed until M18 by the CS3MESH4EOSC Project, within WP4, that resulted in the overall architecture and approach for integrating applications and use-cases through the Interoperability Platform (IOP).

This work included, among others, analysing the flagship use-cases specifics (overview in Section 2.1), capturing the overall functional requirements vis-a-vis the applications, and required characteristics of the integration solutions desired (summarized in Section 2.2), as well as the actual technical design, integration, and implementation work (described in Section 2.3). The latter included, on one hand, extensions to the applications as to improve their usability in the identified use-cases, and on the other one, devising ways to integrate those applications with the IOP, including extensions required for integration on either side.

Details of this work, including design decisions, scope and priorities of implementation, implementation status and future work are covered in Section 2.3 of this report. Those were also some of the objects of the technical report which was prepared for the review of the M1-18 project implementation period.

The work done so far resulted in prototype integrations and demonstrators for the use-cases mentioned above. On the one hand provide proofs of concept which validate the feasibility of each application, and on the other hand can be used to validate the concepts with real users, through close feedback loops, and provide a basis for the first set of user applications for the ScienceMesh.

Overall, the application-IOP integration activities are on track with regards to the workplan of the project, with some limitations and delays mostly related to the fact that user-interface support is still lacking in the ScienceMesh stack and will improve only throughout the coming months

It will be a priority in the M21-24 period to build on that work and provide proper Graphical User Interfaces to users, especially groups of researchers which are not tech-savvy enough to be able to benefit from the prototypes in their current state.

The WP4 work plan is currently being re-prioritized in view of the second half of the Project and with the aim of providing usable “minimal viable product” (MVP) applications to the users as soon as possible. Although limited in functionality, those MVPs can be already used in order to encourage end-users to take-up the on-premise, federated applications on one hand, and on the other one encourage the corresponding system operators and decision makers to join the distributed infrastructure which provides them (ScienceMesh).