Project acronym: CS3MESH4EOSC

Deliverable D4.2
Demonstration of application functionality (all workflows) in a user group

Contractual delivery date	31-12-2022 (M36)
Actual delivery date	12-10-2023
Grant Agreement no.	863353
Work Package	WP4
Nature of Deliverable	D (Demo)
Dissemination Level	PU (Public)
Lead Partner	PSNC
Document ID	CS3MESH4EOSC-21-015
Authors	Maciej Brzeźniak (PSNC), Armin Burger (JRC), Diogo Castro (CERN), Marcin Sieprawski (Ailleron); Piotr Wichliński (Ailleron), Guido Aben (AARNet), Giuseppe Lo Presti (CERN), Juri Hößelbarth (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)

Disclaimer:

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

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

Versioning and Contributions History

Version	*Date*	*Authors*	*Notes*
0.1	18.09.2023	Maciej Brzeźniak (PSNC)	Initial draft
0.2	15.09.2023	Maciej Brzeźniak (PSNC), Marcin Sieprawski (Ailleron)	Early draft, with Data Science environments added
0.3	27.09.2023	Marcin Sieprawski (Ailleron)	Data Science environments - CS3API4LAB JupyterLab extension product vision update and advanced pilots after M18
0.4	27.09.2023	Maciej Brzeźniak (PSNC)	Early draft for partners' review
0.5	29.09.2023	Maciej Brzeźniak (PSNC)	Complete draft for partners' review
0.6	03.10.2023	Maciej Brzeźniak (PSNC)	Improved draft for partners and STC review
0.7	04.10.2023	Juri Hößelbarth (University of Münster)	Additions to "Open Data Systems"
0.8	06.10.2023	Maciej Brzeźniak (PSNC)	Addressed partners' comments
0.9	9.10.2023	Marcin Sieprawski (Ailleron)	Review of the deliverable
1.0	9.10.2023	Maciej Brzeźniak (PSNC)	Comments applied
1.1	11.10.2023	Sonia Mentrida (CERN)	Formatting

Table of Contents

Table of Figures

Table of tables

1 Introduction

This document presents the state of work in the WP4 of the CS3MESH4EOSC project related to preparing demonstrators of the application-level functionality provided on top of EFSS. These application areas include Collaborative Data Science environments, Collaborative Documents Editing platforms and solutions, Open Data Systems and FAIR and Data Transfers.

This deliverable provides a summary of the work on integrating and validating application-level workflows in the context of user communities and groups. It also summarizes the work on preparing the demonstrators of functionality and features developed within the mesh, with the focus of the project's flagship user's group needs and their feedback.

The work has been performed in the overall context of the efforts undertaken by the CS3MESH4EOSC consortium related to building the federated mesh of sync & share systems.

These efforts are organised in two main dimensions: (1) spanning multiple technologies of EFSS (ownCloud, NextCloud and others) and multiple sites (including project partners sites and early adopters' sites) – the horizontal integration; (2) providing application-level functionality focused on four identified flagships use-cases from various science and research domains, in collaboration with relevant user communities – the vertical integration.

The full context of the work is presented in deliverable D4.1: "Demonstration of application functionality across federated share" which provides the overview of the use-cases, presents the basic technical components adopted for integrating application-level functionality into the mesh, details the technical design and the integration work and discusses the first, early versions of the demonstrators, prepared until month 18.

Based on the initial prototypes worked out until M18 (documented in D4.1), a demonstration of the application functionality has been performed with pilot user communities identified. Selected application components have been deployed and tested, by the end-user groups on a larger scale. Further developments and integrations have been conducted based on this more extensive evaluation and feedback coming up from these tests.

The document reflects the state of work as of M36 of the project, as foreseen by Project Amendment #2 of April 2022.

2 ScienceMesh Applications – state of work in M36

2.1 Overall progress

Based on the proof of concept solutions (PoCs) and small-scale demonstrators worked out until M18 (documented in D4.1 in M22 of the project) as well as the small-scale evaluation of the application-level and mesh functionality further integration, development and more extensive validation work has been conducted in all key application areas of WP4.

The development, integration, improvements end evaluation work in M18-M36 included:

(1) designing and implementing further extensions of application-level components that provided comprehensive functionality and features that better reflect and respond to the user requirements vs the application workflows that were collected over the project timeline from piloting, PoCs and evaluations;
(2) improving the quality of the application-level components and integration solutions focused on performance scalability, responsiveness (to user requests in the GUI and to API calls, e.g. to Reva) as well as correctness, stability, reliability and security;
(3) extending the integration of applications services with EFSS systems and their storage back-ends, in particular by improving the level of support for the EFSS federation features based on the deeper integration with IOP and CS3APIs;
(4) integrating and demonstrating application functionalities over the federated mesh nodes spanning various technologies of EFSS (ownCloud, NextCloud and others) and multiple sites (including project partners sites and early adopters' sites;
(5) evaluating the project results: application components, mesh components, their functionality, interfaces (GUIs, APIs, CLIs) as well procedures, policies and documentation in collaboration with larger user groups, which provided the added value and extra source of information compared to M1-M18 evaluation, that was mostly focused and performed by the developers, early users and site operators.

2.2 Progress of work in the application areas (M18-M36)

This section details the improvements and extensions in the application stacks conducted in after M18 of the project. The presentation of the work includes a short summary of the state of the components until M18 and the analysis of the work done until M36 of the project.

M36 (December 2022) has been set as the checkpoint, as (1) it was originally planned to be the last month of the project (extended until June 2023); (2) it directly preceded the CS32023 conference in Barcelona, where advanced prototypes were to be presented to the audience of CS3 community including important stakeholders, EFSS system operators, early adopters and active and potential users of the project outcomes; WP4 applications were presented during project partners-delivered presentations in the main programme of the conference and within the dedicated ScienceMesh workshop at the conference.

2.2.1 Data Science Environments

The initial pilots (<M18)

Data science solutions' pilots have been initially built for CERN, JRC and for PSNC based on the original software stacks used in particular sites extended with federation features, visualisation capabilities and back-end storage integrations.

CERN's Data Science software and services stack included SWAN[1] (Service for Web based Analysis) that is a Jupyter Notebooks environment (based on JupyterLab v3), CERNBox (based on ownCloud Infinite Scale – OCIS) and EOS (CERN-developed scalable filesystem) as well as containerized Spark, Hadoop and/or HTCondor clusters for data analysis and computations

JRC used CERN-developed SWAN, ported to JRC infrastructure and adopted to characteristics of the JRC management policies including specific security measures.

PSNC used stand-alone 'vanilla' JupyterLab with NextCloud as the EFSS, orchestrated by Openshift (application provisioning) and OpenStack (compute and storage provisioning).

Until M18 extensions of the SWAN environment, in particular JupyterLab has been implemented and deployed in order to ensure basic support for the ScienceMesh federation.

This included the CS3APILAB plugin[2] implementation and integration that provided federation features including notebooks sharing and made them available via JupyterLab user interface.

Project-delivered extensions included also advanced visualisation capabilities enabled at JRC, based on Voila[3], which created the possibility to compile the data analysis workflows created with Jupyter Notebooks to visual dashboards for broader usage by less experienced users.

The advanced pilots (M18+)

Collaborative notebooks editing capability, advanced visualisation features and backend storage view integration have been implemented and demonstrated already before M18. Based on this work, larger scale evaluation, testing and feedback, extensions and improvements have been performed organised along the following two streams of work:

(1) improving and extending the integration of the Jupyter environment with the CS3APIs based federation layer as well as the storage back-ends of Jupyter/EFSS environment has been improved; the work focused on stabilizing the integration components such as the JupyterLab plugin, as well as ensuring a timely synchronised view on data from the Jupyter environment, EFSS and computing kernel codes;
(2) improvement of advanced visualisation features, by developing the VOIS library that extends the visualisation capabilities, delivered within in JRC Big Data Analytics

[1] https://swan.web.cern.ch/swan/
[2] https://github.com/sciencemesh/cs3api4lab
[3] https://github.com/voila-dashboards/voila

Platform (BDAP).

Improving the Jupyter-CS3 APIs integration

After M18, extensive quality and performance improvements and stabilisation work have been conducted as a common effort undertaken by Ailleron and CERN.

Based on the demonstrations, feedback has been collected, which influenced further work on corrections of the low-level mechanics implementation (mechanics) and improvements of the sharing and collaborative editing functionality interface. They included:

1. **Identification and removal of performance bottlenecks:**
    a. reworking several frequently used operations that impact the user-observed performance including file existence checking file stat, examining the possibility of locking the files, checking the privileges and permissions of users vs the files and shares etc.
    b. refactor the interface of the library – in order to limit the request count for processing basic operations such as checking file attributes, metadata etc. e.g. by filtering the objects queried and checked with stat during preforming file opening operations as well as caching passing the meta-data information among the layers of the software stack (to avoid duplicated stats);
2. **Optimising interfaces to APIs**: (gRPC CS3APIs, OCM) by applying filters and avoiding excess I/O wherever possible.
3. **Fixing bugs** and **implementing improvements**:
    a. fixing bugs, e.g. related to stat operating on different paths to the same file;
    b. improved dealing with conflicting edits (e.g. by creating a conflict file)
    c. Implemented a share generator
4. **Improving monitoring and operational analysis –** A repository with the integration codes and instruction/manual for ELK stack deployment and integration has been prepared; these included among others creating logstash logger and configuring the relevant Filebeat module for collecting extensive logs from the Jupyter plugin

The work related to improvements of the UI components included adaptation targeted to fit the UI to the refactored notebooks sharing, collaborative editing and locking interface, for instance by including notifications on the conflicts or shared files. The improved Jupyter GUI integrations contributed to the overall better usability of the collaboration features.

In addition, substantial work has been put into ensuring the long-term sustainability of the JupyterLab plugin. Various actions have been taken and planned including (1) Regular maintenance and updates that kept the plugin compatible with the latest versions of Jupyter; (2) Setting up the CI/CD environment; (3) Collecting feedback from users at CERN (hands-on experience) and through the public events and wide exposure of the solution the end-users, and (4) advertising the plugin to the community (e.g. at JupyterCon).

Final set of functionalities of the extended prototype, available through JupyterLab's file browser include: (1) Sharing buttons and entries in the context menus – enabling to share a

notebook or directory; (2) “Shares” window – displaying sharing status, enables searching users (by name, username, email) to add and manage shares – with “viewer” and “editor” access; (3) Share tabs – enable to see resources “shared by me” and the ones “shared with me”; (4) Collaborative workflows, secured with locking mechanism for overwrite protection: “File locked” popup enables two options: read-only access to locked files (Preview mode) or working on a copy. The collaborative workflows are prepared for future implementation of merge functionality; (5) Read-only access: when the notebook is shared with read-only access, the grantees cannot modify the code, but code cells are visible.

An exemplary look at the redesigned JupyterLab GUI, with the “shares” tab and panel highlighted is presented in Figure 1 below.

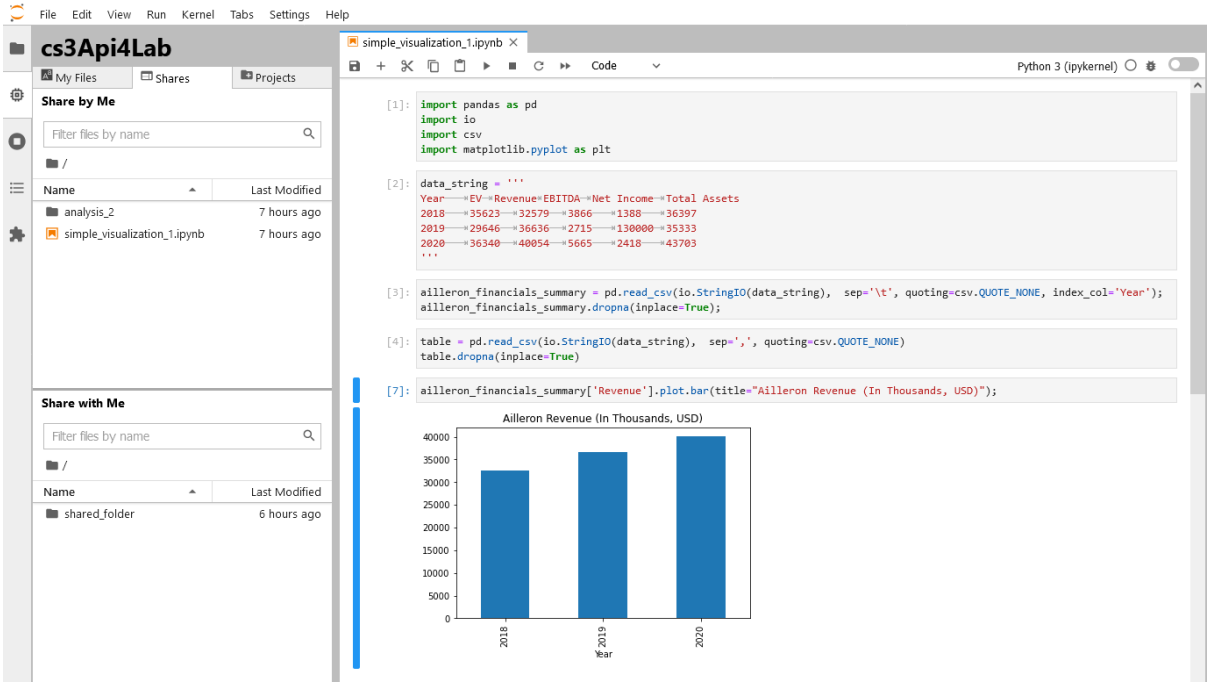

Figure 1. View of the redesigned JupyterLab GUI (“shared” tab and panel)

Public demonstrators

Public demonstrations based on the extended prototypes have been conducted in order to increase the exposure of the solutions and open new feedback channels. They have been performed during the project related events such as the dedicated ScienceMesh workshop at CS3 conference and within the main programme of this conference as well as during other events, that provided access to key stakeholders in the domain (e.g. JupterCon that is a forum for data science end-users, research communities and software developers). Table 1 below summarizes these interactions.

Table 1 Demonstrations of the Juptyer-based data science environments

Conference/Event	Paper/talk	Location	Date
Data Science Summit	Marcin Sieprawski: „ScienceMesh + JupyterLab: How collaborative Data Science helps CERN analyse data from Large Hadron Collider (case study)"	Warsaw (Poland)	16 Dec 2022
ScienceMesh workshop @CS3 Conference	Marcin Sieprawski Data Science Environments: JupyterLab sharing and collaborative editing https://indico.cern.ch/event/1210538/contributions/5317088/	Barcelona (Spain)	7 Mar 2023
CS3 Conference	Marcin Sieprawski **Data Science environments in ScienceMesh** https://indico.cern.ch/event/1210538/contributions/5208015/	Barcelona (Spain)	8 Mar 2023
JupyterCon	Diogo Castro, Marcin Sieprawski **Federated collaborative workflows for Jupyter** https://cfp.jupytercon.com/2023/talk/Y3ZDC7/	Paris (France)	10-12 May 2023
Conference „Artificial Intelligence"	Marcin Sieprawski: participation in the discussion panel **"KEY ROLE OF DATA IN ARTIFICIAL INTELLIGENCE APPLICATIONS"** https://www.youtube.com/watch?v=rscZGvoNzSk	Warsaw (Poland)	17 May 2023
EGI Conference / Science Mesh Workshop	Marcin Sieprawski: Science Mesh Interoperable Research Workflows and Research Infrastructures" for the **"Earth Observation: Pushing Data Science to The Limits with Voila"**.	Poznan (Poland)	19-23 June 2023

Improving the Jupyter-storage back-end integration

Fully-fledged Collaborative Data Science solution should ensure that: (1) notebooks can be shared and edited collaboratively, (2) data sets can be consistently accessed both at the level of JupyterLab and the level of computing kernels.

The first feature is addressed by the extension of JupyterLab including the CS3API plug-in discussed above. The second feature i.e. consistent view of data sets across JupyterLab GUI interface, its editors, terminal interface and computing kernels required integration work. In case of CERNBox this functionality is available by design - due to the fact that all the components share EOS filesystem-based backend. An approach has been developed in order to enable such functionality in other environments including vanilla Jupyter setups.

Two implementations have been prepared to ensure the synchronised view of data across Jupyter environment, EFSS and the kernels: (1) fuse-based mount of the WebDAV exposed EFSS data space; (2) access via EFSS sync client running within the JupyterLab instance.

The relevant demonstrator has been prepared, already before M18, firstly for the vanilla JupyterLab and NextCloud EFSS running at PSNC. It has been extensively tested, including

using it in pre-production Jupyter instances and improved, based on the feedback. This enabled them to identify and complete the following developments in M18-M36 period:

1. **Streamlined authorization to improve user experience** – Thanks to the user feedback, a difficulty in the initial setup of Jupyter -> EFSS authorization has been identified. The initial approach required a user to manually grant the Jupyter service access to the EFSS, which turned out to be non-intuitive for less-experienced users. After M18, a new approach to the authorization has been developed, and it requires no manual actions from users. It is based on an extension of EFSS API and a new JupyterHub module, which securely exchanges user credentials behind the scenes.
2. **Validated vendor neutrality** – In the initial approach, the file synchronisation was based on a vendor-specific sync client. After M18, it was replaced with Rclone that introduced the feature of bidirectional sync at that time, which allowed its adoption for this work. The resulting solution has been tested with different EFSS vendors, i.e. CERNBox (OCIS), OwnCloud and NextCloud.
3. **Validated portability across different platforms** – The deployment of this solution has been tested across different container platforms. Its compatibility with container engines – Docker and CRI-O – has been verified. Container orchestration has been validated both in Openshift/OKD and vanilla Kubernetes.
4. **Improved monitoring and alerting** – Monitoring of the JupyterHub and EFSS deployments has been improved by introducing a dedicated Zabbix dashboard, Prometheus extracting metrics from JupyterHub, as well as a custom alerting on Jupyter spawning problems based on aggregation and analysis of logs.
5. **Performance tests** – Performance tests were conducted using the testing tool Locust to simulate concurrent users of the Jupyter-EFSS setup. In those tests, various metrics from both Jupyter and EFSS services were analysed to look for possible bottlenecks. Tests resulted in adding a few new configurable settings to the setup, that can be adjusted depending on particular use cases and environments.
6. **Porting to new versions of JupyterHub** – The initial solution was built on top of the well established JupyterHub deployment, 'Zero to JupyterHub', at that time available in version 0.9. In the meantime, versions 1.x have been released that introduced significant changes to configuration and upgrades of several key dependencies. The developed solution has been ported to and validated with those new versions.

The final architecture of the solution worked is presented on the Figure 2 below.

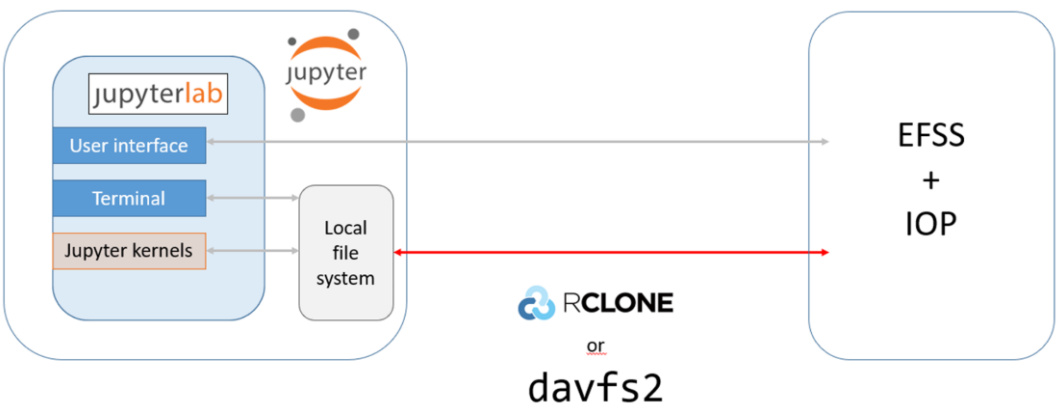

Figure 2. The high-level architecture of the data view unification mechanics for vanilla JupyterLab

Public demonstrations

The public demonstration of solution that enables the consistent view of data sets across JupyterLab GUI interface, its editors, terminal interface and computing kernels required integration work has been performed among other on the ScienceMesh workshop at CS3 2023 Conference in Barcelona (7 mar 2023)[4].

Improving the visualisation features (VOIS)

During the second period of the project (M18-M36) advanced visualisation features of Data Science Environments have been improved by developing the VOIS library. Initially, until M18, Voilà was used as a visualisation layer for Data Science at JRC. The demonstrators and production environments have been created and operated already before M18. Following user feedback, collected over the course of the CS3MESH4EOSC project and the testing and validation within JRC a need for a more advanced user interface framework was identified. Therefore, to provide advanced visualisation capabilities for Jupyter Notebooks, the VOIS library[5] was developed by JRC and integrated into JRC Big Data Analytics Platform (BDAP).

VOIS library aims to simplify the complex tasks involved in the creation of impactful and modern data visualisation dashboards and to provide easy-to-use and reusable components such as compound widgets, custom interactive charts and multipage applications. It was subsequently used to create impactful dashboards for Earth Observation Data (SHERPA-CLOUD and LUCAS 2022). The VOIS library was released under an Open-Source licence, and it may be used to create functionally and visually rich Data Science solutions, thus adding to the portfolio of the application-level services of the ScienceMesh. VOIS gained a lot of interest in creating rich and engaging user experiences: ipyvuetify[6]. It is a widget library based on the Vuetify/Vue JavaScript frameworks[7] for making modern looking GUIs in Jupyter notebooks

[4] https://indico.cern.ch/event/1210538/sessions/463166/#20230307
[5] https://vois.readthedocs.io
[6] https://ipyvuetify.readthedocs.io/en/latest/
[7] https://vuetifyjs.com/en/

and dashboards. It implements the Google material design philosophy[8], best known for the Android user interface, and provides a large set of widgets with multiple variants, all highly customizable. ipyvuetify presented a relatively steep learning curve, which is why JRC developed, with partial funding provided by the CS3MESH4EOSC Project, a library with the aim to simplify the complex tasks involved in the creation of modern dashboards and to provide easy-to-use and reusable components: the VOIS library[9]. This pure Python package provides many ready-to-use widgets and exposes an "app" class that can serve as the base for creating the dashboard structure. With few lines of code, the "app" can be customised using styles, colours, fonts, images, and all the graphic elements that provide it with its uniqueness. VOIS library has many functions for the easy creation of complex geospatial visualisations. It contains several custom-made SVG interactive charts that allow for modern user interaction and widgets for the display of hierarchical and tabular data.

The VOIS library has been integrated and configured for several use-cases at JRC, among which SHERPA (Screening for High Emission Reduction Potential on Air) cloud dashboard and LUCAS 2022 survey (Land Use and Coverage Area frame Survey"[10]) are work mentioning. Successful integration of these use-cases, that deal with multi-terabyte source dataset, into the JRC Data Science platform has proved the overall scalability and modularity of the Data Science platform worked out in the project.

Public demonstrations

The public demonstration of the VOIS-based solution that enables interactive, multi-faceted presentation and visualisation of the massive dataset based on the integrated software stack have been performed within the project-related events, including the ScienceMesh workshop at CS3 2023 Conference in Barcelona (7 Mar 2023)[11,12].

2.2.2 Open Data Systems

The Open Data solution has been available since M18 of the project. It is based on ScieboRDS[13], which provides services and tools for research data management and scientific data analysis. It has been developed and maintained at the University of Münster (formerly known as WWU), and extended partially based on the CS3MESH4EOSC project funding and the University of Münster's own funding. It supports meta-data handling via digital objects, data packaging and repository deposits from different EFSS systems.

During the initial phase of the project, extensive studies on user expectations were conducted. The aim was to get a clear view of the expectations from the research community. Mainly, interviews with more than a dozen researchers from different research communities

[8] https://m2.material.io/

[9] https://vois.readthedocs.io/en/latest/1_intro.html

[10] https://ec.europa.eu/eurostat/web/lucas

[11] https://indico.cern.ch/event/1210538/sessions/463166/#20230307

[12] https://www.daigio.it/JRC/APOD_VoilaDemo.mp4

[13] https://github.com/Sciebo-RDS/Sciebo-RDS

were carried out. The focus was on their view on FAIR research data management. Mockups of the user interface were created and presented to the users. The outcome of these interviews was considered in the design of the first Sciebo RDS prototype, which was then deployed at the University of Münster.

In the course of the development, the need for a standardised way to store the metadata came up. After a careful evaluation, the decision to use so-called RO crates[14] was made. These can be managed by the reference implementation tool „Describo“, which was subsequently integrated into the Sciebo RDS prototype.

The advanced pilots (M18+)

Based on the integrated pilot and proof-of-concept implementation available at M18, development and testing on ScieboRDS continued. In the context of the CS3MESH4EOSC project and based on the University of Münster's own funding, further development of Sciebo RDS and its transformation to a more general solution was conducted. This work included:

1. **Creating new connectors to the repositories.** Initially, the University of Münster created the connector to ingest data from ownCloud into Sciebo RDS. Over the course of the project, the University of Münster created connectors for the Open Science Framework (OSF), OpenAIRE-based Zenodo and Datasafe (custom data archive service at the University of Münster). SURF, the CS3MESH4EOSC project partner, adopted the Open Data solution and developed additional connectors for SciebeRDS to share data to other research data repositories including Figshare[15], iRODS[16] and Dataverse[17]. SND (Swedish National Data Service) has begun work on a connector for their own data repository, DORIS.
2. **Deploying and integrating ScieboRDS with other EFSS systems.** SURF adopted the Open Data service based on ScieboRDS by integrating it with SURFdrive, an ownCloud 10-based EFSS system for SURF's research and education audience. In addition, SUNET (an early adopter of the project results) deployed the ScieboRDS solution to their EFSS system, SUNET drive, which is based on the NextCloud EFSS. The development work necessary to adapt the EFSS integration from ownCloud 10 to NextCloud was carried out by SUNET.
3. **Working on the overall interoperability.** During the later part of the project (after M18) a collaboration was established with developers of the InvenioRDM[18] data management system with the goal to ensure compatibility between ScienceMesh and InvenioRDM. InvenioRDM implemented new transport mechanisms (e.g. added RO-Crate support in the InvenioRDM API) to ensure more interoperability between

[14] https://www.researchobject.org/ro-crate/
[15] https://figshare.com/
[16] https://irods.org/
[17] https://dataverse.harvard.edu/
[18] https://inveniosoftware.org/products/rdm/

ScienceMesh and InvenioRDM as well as a load balancing capability through pull mechanisms dynamically scheduled on the repository side.

4. **Bug fixes and improved user interface.** A ScieboRDS associated research group formerly at the University of Duisburg-Essen, now at the University of Potsdam, conducted user studies based on the initial user interface prototype developed for M18. The University of Münster subsequently took the results of that study to further redesign and improve the user interface and fix bugs that had been found in the study and by forementioned partners, SURF and SUNET.

Figure 3 below shows the ScieboRDS „projects" panel that allows planning and executing the data package description and repository deposits.

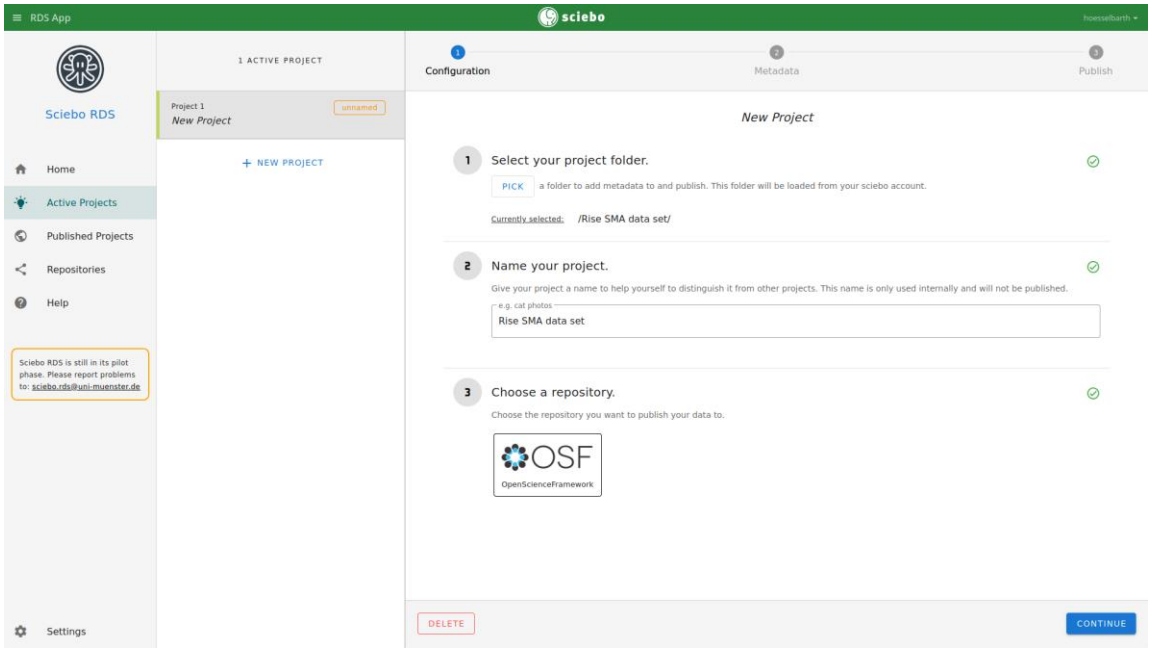

Figure 3: The Sciebo RDS interface - the advanced pilot version (as of M36)

Public demonstrations

ScieboRDS has been run in pre-production mode since M18 of the project. It has been exposed to several user communities including members of University of Duisburg-Essen and University of Potsdam project RISE_SMA related to Social Media Analytics for Society and Crisis Communication.

SURF, another CS3MESH4EOSC project member, integrated the ScieboRDS solution with their EFSS service to address the needs of the Dutch R&D community. Similarly, the external partner SUNET deployed a prototype and is working on a full deployment to serve ScieboRDS to the Swedish R&D community.

The ScieboRDS-based Open Data solution has been demonstrated at multiple conferences and workshops, as shown in Table 1 below.

Conference/Event	Description	Location	Date
ScieboRDS workshop @SURF	Workshop conducted by the University of Münster for SURF	Amsterdam (Netherlands)	25 - 28 Jul 2022
ScieboRDS workshop @SUNET	Workshop conducted by the University of Münster for SUNET and SND	Stockholm (Sweden)	27 - 29 Sep 2022
1st International Conference on FAIR Digital Objects	Juri Hößelbarth, Guido Aben: „The use of FDOs for a democratised approach to FAIRness“ (Poster)[19]	Leiden (Netherlands)	26 - 28 Oct 2022
CS3 Conference 2023	Juri Hößelbarth, Richard Freitag „Sciebo RDS - reducing friction of FAIR data handling for researchers“ (15 min talk)[20]	Barcelona (Spain)	07 Mar 2023
TNC23 (Conference)	Juri Hößelbarth, Richard Freitag „Sciebo RDS - reducing friction of FAIR data handling for researchers“ (25 min talk)[21]	Tirana (Albania)	06 Jun 2023
EGI Conference / Science Mesh Workshop	Lennart Hofeditz „Use of Science Mesh Open Data Systems for RiseSMA Social Media Analytics“[22]	Poznań (Poland)	22 Jun 2023

Table 1: Demonstrations of the ScieboRDS system

2.2.3 Collaborative Documents

The work on the 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. The key enabler of this capability is a WOPI connector[23] which was developed as a plugin for the IOP. WOPI[24] is the Web Application Open Platform Interface Protocol, supported by numerous authoring tools and services that allow to integrate components in private and hybrid cloud scenarios.

The initial pilots (<M18)

The initial pilots, made available before M18 of the project, included a set of collaborative editors including CollaboraOnline, CodiMD and Etherpad.

[19] https://zenodo.org/record/7310739
[20] https://indico.cern.ch/event/1210538/contributions/5207925/
[21] https://tnc23.geant.org/sessions/#s254
[22] https://cs3mesh4eosc.eu/news-events/events/cs3mesh4eosc-final-event-science-mesh-unlocking-open-science-and-collaborative
[23] https://github.com/cs3org/wopiserver
[24] https://docs.microsoft.com/en-us/microsoft-365/cloud-storage-partner-program/online/

The basic integration work included developing at WOPI bridge, initially applied to CERNBox, the CERN's production EFSS system. Extensions included triggering updates of the documents in the EFSS in response to notifications on changes received from the online editing application. CodiMD has been integrated using webhooks that catch up the file modifications. Also, an early implementation of the Etherpad plugin has been prepared and contributed upstream.

These solutions have been extensively used in pre-production (beta) mode, tested and validated by the users of production EFSS installation at CERN.

The advanced pilots (M18+)

Over the course of the project the scope of integration has been broadened. This included:

1. **Analysing and prototyping integrations with additional authoring tools**
    a. Investigating the possibility of integrating OnlyOffice. The initial approach was to exploit OnlyOffice's WOPI support, which hadn't materialised during the first 18 months of the project. In the second part of the project efforts have been put into developing the integration components for OnlyOffice. At the end, in mid 2021, OnlyOffice themselve has integrated WOPI Support which facilitated including this application into the portfolio of collaborative editing systems of ScienceMesh.
    b. Performing the integration of OnlyOffice and Diagrams.net analysed before and providing the proof-of-concepts for these applications.
    c. Considering the integration of two other authoring applications – Excalidraw (whiteboard editing) and Overleaf (LaTeX editing). In both cases, fully-fledged integration of the solutions would require involving their authors (through cooperation in the open source model or through outsourcing), however, these efforts have been ceased as the outsourcing funding has been assigned to other urgent project activities (including broadening federation features support in EFSSes by backporting CS3APIs to OC10 and NextCloud)
2. **Testing the existing integrations on a large scale.** Based on the preproduction pilots at CERN, ported to WWU, large-scale testing of the collaborative editing products has been performed. CERNBox servers around 37k users, Sciebo serves 227k users. According to statistics, around 10% of these users run online editing applications. This volume of authoring applications users provided a solid ground for large scale testing. The feedback from these tests has been included in the integration work.

Implementation details

Overall, most of the integration mechanics have been built on top of WOPI. WOPI is a standard for online collaboration, developed by Microsoft for its Office applications - it has been widely adopted by the industry, and document editing solutions are compatible with it. It is a high-level protocol which enables applications to communicate with the storage server to perform operations such as locking, reading or saving files.

Integration with Collabora Online was straightforward thanks to already existing vendor support for WOPI. OnlyOffice integration was also achieved after the vendor decided to natively support the WOPI protocol in August 2021. This decision followed from the interactions with the ScienceMesh sites which provide OnlyOffice to their users' community.

With applications such as CodiMD and EtherPad, which do not support the WOPI protocol natively, but have a similar workflow, a different route was taken. The *WOPI Bridge*[25] module was developed by CERN for its CERNBox service and adopted and extended for the purposes of CS3MESH4EOSC. The WOPI Bridge was integrated into the main WOPI server package. Thanks to this integration, file updates can be triggered in the storage system (e.g. EFSS), in response to notifications received from the online editing application.

As a result of this work both Collabora and OnlyOffice integrations became part of ownCloud's OCIS platform. Furthermore, CodiMD integration is used in production at CERN.

Overleaf integration was demonstrated in a hybrid scenario in the CERNBox service. The setup allows local EFSS (OCIS) to connect to public Overleaf services (https://overleaf.com). The details are provided in the report "Integration Overleaf with CERNBox"[26]. The Overleaf integration with the Sciebo EFSS at WWU, where the CERN's approach to integration has been ported, has been available for beta-testing and it has the potential to become a widely adopted service; reaching the production stage is planned in the coming months.

The task partners were also looking into possibilities of integrating online graphic editing applications such as Diagrams.net (previously Draw.io) and Excalidraw. Diagrams.net has been the object of a proof-of-concept integration at CERN, but work is still required to create a more robust implementation that would also support collaborative editing. The consortium also considered Excalidraw as a possibility, but it is not currently offered as an on-premise solution and is not an option unless such a possibility is offered in the future.

Public demonstrations

Collaborative editing solutions' integrations into EFSS systems have been exposed to users of CERNBox at CERN and users of WWU-operated Sciebo EFSS. They have also been demonstrated during the project events, including ScienceMesh webinars and workshops.

2.2.4 On-demand data transfers

On-demand data transfers solutions address moving the large datasets between nodes in ScienceMesh as well as to and from external storage services including data infrastructures, data repositories etc. Two major use-cases and workflows have been identified:

(1) on demand, ad-hoc data transfer between collaborating users at federated sites, dealing with small and large datasets;

[25] https://github.com/cs3org/wopiserver/blob/master/src/bridge/readme.md
[26] https://cds.cern.ch/record/2868534/files/Overleaf-Integration-Adriana.pdf

(2) managed and scheduled transfers of massive datasets from and to large data repositories, driven and operated by research and scientific communities;

For the first use-case, general purpose tools have been analysed, and RClone[27] has been chosen as the candidate for the integration. For the second case, more specialised tools needed to be applied; Rucio[28] and FTS[29] have been selected for further integration.

The initial pilots (<M18)

The integrations until M18 resulted in the prototype for ad-hoc, on-demand data transfers based on RClone. The details of the RClone-based solution have been presented in D4.1.

Managed transfer solution based on FTS and Rucio integration has also been designed however implementation has been postponed to the later part of the project.

FileSender[30] integration has been considered as the user-oriented data transfer mechanics to be used with the EFSS, however, the conclusion was to postpone its integration for the future.

The advanced pilots (M18+)

In the second part of the project, extensive testing of the prototypes has been conducted, in collaboration with early adopter user communities.

The M18 pilot for ad-hoc data transfers based on RClone has been first tested mostly in the development and testing environment at SURF and at CERN. Then the solution has been made available to a wider audience. Among others HPC.NRW project[31] utilised the Rclone integration for data transfers between HPC clusters. A pre-production setup has been tested for data movements among the systems of the universities of Münster and Paderborn.

In the second part of the project, the development work continued and resulted in the integration of the Rucio/FTS-based managed data transfers into the EFSS software stack.

The integration work has been conducted in several parallel activities:

1. **Developing the CS3APIs implementation for Rucio.** As a part of the 2021 Google Summer of Code project "Rucio and CS3API to enable data management for the ScienceMesh cloud"[32], an HTTP third-party copy mechanism was implemented for IOP/Reva-enabled sites. This is the key technical capability that enabled further integration with the FTS service and Rucio. The solution has been tested in the contributing partners's sites (e.g. SURF), and then made available for public pilots.

[27] https://rclone.org/
[28] https://rucio.cern.ch
[29] https://cern.ch/fts
[30] https://filesender.org/
[31] https://hpc.dh.nrw/en/
[32] https://hepsoftwarefoundation.org/gsoc/2021/proposal_rucio_cernbox.html

2. **Collaboration with the ESCAPE project.** Extensive work on developing, evaluating and improving the proof-of-concept implementation of the managed data transfers has been conducted involving external data infrastructure. Within the CS3MESH4EOSC collaboration with ESCAPE a dedicated "ScienceMesh" group (Virtual Organisation) was created in the ESCAPE Data Lake testbed infrastructure. The ESCAPE project was an EU-funded project which addressed the common needs of a number of ESFRIs and other research infrastructures like SKAO, HiLumi, VIRGO, CTA, and LSST). In the ESCAPE Data Lake, the distributed data management is controlled by Rucio and actual data transfers between the storage nodes in the lake are orchestrated by FTS.

Future work

Future development options include integration of all existing technical components and integration with the EFSS Web Interface as well as deployment of the integrated solutions across the mesh. The functionalities are already available on the CLI and API level and have been extensively used during the integration work as well as for testing and validation purposes.

As a possible extension of the current workflow, fully-fledged interactions between users and the data handling services could be provided. For instance, the user may interact with Rucio through the JupyterLab interface. This capability, based on SWAN, was developed and demonstrated as a 2020 Google Summer of Code project "Making Exabytes of LHC data seamlessly accessible on Jupyter Notebooks"[33]. It can help to extend the usage of large, federated data infrastructures to cater to modest data management use cases.

Adoption of the managed massive data transfer solutions has been also discussed with Astron, the coordinator of the LOFAR project, and participants in the Square Kilometre Array (SKA). These results were also presented to and attracted the interest of the "Tangerine Team" of the SKA Regional Centres (SKA Data Science and Simulations Program). The team deals with how SKA users interact with data, so it has a direct influence on the future architecture of the data platform within SKA.

Public demonstrations

The on-demand data transfers" solution based on Rclone has been demonstrated in collaboration of CS3MESH4EOSC and the HPC.NRW project[34] that involves all universities with computing centers in North-Rhine Westfalia (Germany). The pilot utilised the RClone integration for data transfers between HPC clusters. This pre-production setup has been tested between the systems of the universities of Münster and Paderborn.

The managed data transfer solution based on FTS and Rucio has been firstly tested in the contributing partners sites (including SURF), and then made available for public pilots. One of the demonstrators has been deployed to implement the massive, managed data transfers between SURF and Laboratoire d'Annecy de Physique des Particules (LAPP). As already

[33] https://hepsoftwarefoundation.org/gsoc/2020/proposal_SWAN_RUCIO_integration.html

[34] https://hpc.dh.nrw/en/

mentioned, this work was part of CS3MESH4EOSC-ESCAPE collaboration.

The overall idea was to integrate the ESCAPE data lake and the ScienceMesh and thereby connect the two e-infrastructures and enable massive data transfers among them. This possibility has been validated and demonstrated: a person with a data manager role in ESCAPE was able to transfer data using FTS from a storage system at LAPP to a ScienceMesh node at SURF. Then the data at SURF could be shared with a primary investigator who was a local user at SURF and after that this PI shared the data with another local user at CESNET using the ScienceMesh invitation workflow and sharing mechanism. Figure 4 below overviews the demonstrated case.

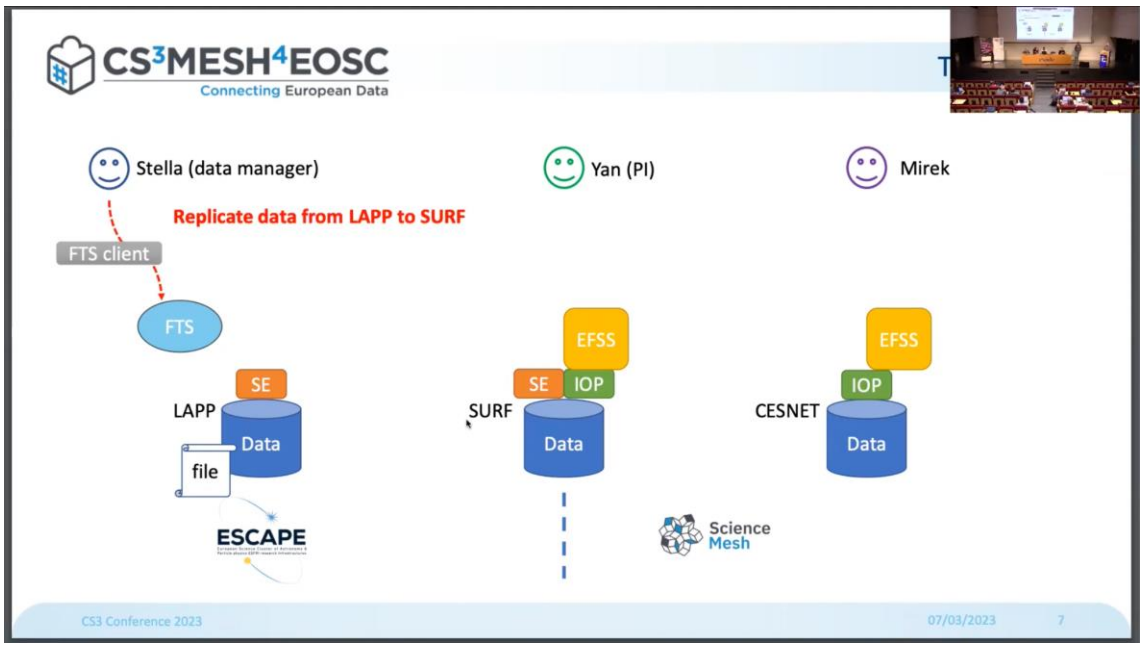

Figure 4: Demonstration of managed data transfer at CS3 2023

The video recording of this workflow is included the proceedings of the ScienceMesh workshop[35] organised during the CS3 conference in Barcelona, 6-8 March 2023.

2.2.5 Integration of Event Management Systems

Over the course of the project, after M18, the new application area, Event Management Systems, has been included into the WP4 application-level functionality integration work.

FilerPicker integration between CERNBox and Indico[36], the CERN's developed and widely adopted event management system, has been designed and prototyped as well as integrated in production services at CERN.

[35] https://indico.cern.ch/event/1210538/contributions/5317101/attachments/2611351/4513815/go
[36] https://getindico.io/

It allows the Indico users to pick and upload their files directly from their home directories kept in the EFSS system to the institutional Indico system instance. The CERN's instance of Indico has this feature enable for public use and tested on the large scale.

Figure 5 below shows the basic steps performed by the user while choosing the CERNBox as the source of the data to be attached to the conference contribution.

File picker uses the CS3APIs to query the user's EFSS content, that is then displayed in a panel within the file upload popup windows. Selected files are then transferred to Indico, again using the means of the EFSS and CS3APIs.

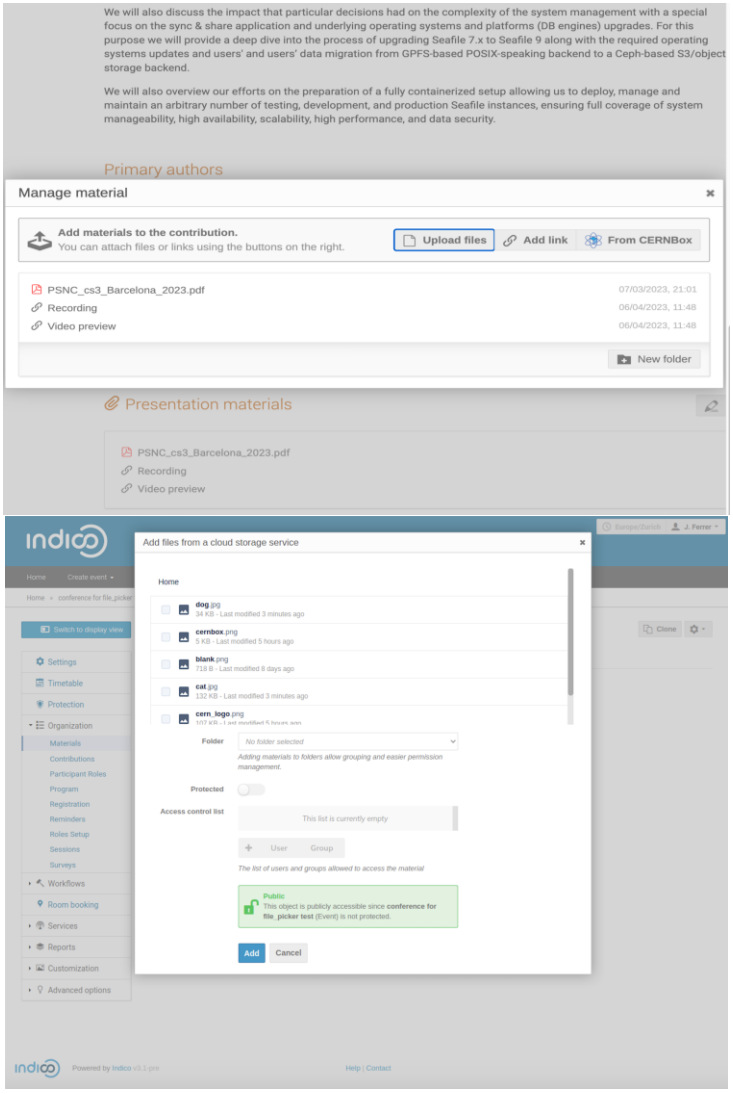

Figure 5: Cloud file picker interface for Indico

3 The demonstrators

The integration, design, implementation and evaluation work from M18 until M36 resulted in preparing new integrations of the application-level functionality as well as extending and advancing the M18 pilot and prototypes. They have been tested and evaluated both within the closed environment of the developers of particular solution, within the sites that participate in the CS3MESH4EOSC project and in the open environment through the public pilots, demonstrations and evaluations and pre-production and production use.

These developments are described in details in Section 2 of this document. In this section a list of the demonstrators is included, along with the status information as of M36 and links to the codes, documentation and related presentation and demonstration materials.

3.1 Application-level functionality demonstrators

Public pilots and demonstrators of the application-level functionality integration in the mesh have been prepared and are extensively tested with the early adopters users groups. The following points summarize the demonstrators available for particular application areas.

3.1.1 Data Science environments

Public pilots and demonstrators of data science environments are available at CERN, JRC and PSNC. CERN Data Science environment is part of the larger and comprehensive data management and computing system, based on CERNBox and SWAN. JRC environment is being used in production by various units of EC for performing data analysis and research. PSNC's educational instances of Jupyter Noteboolks environments are available to students.

Table 2 below summarizes Data **Science** demonstrators available at project sites:

Table 2: Data Science environments demonstrators by CS3MESH4EOSC

Environment name	Status / short info	Availability	Location / URL:
Data Science environment @CERN	Production grade. Based on SWAN and CERNbox.	available to registered CERN users and project participants	https://swan.web.cern.ch/
Data Science environment @JRC	Production grade. Based on SWAN and Voila/VOIS.	available to registered users within EC and EC-driven data science projects participants	https://jeodpp.jrc.ec.europa.eu/bdap/
Data Science environment @PSNC	Production grade. Based on vanilla Jupyter Notebooks with EFSS integration	available to students in high-schools participating in PSNC educational project on Data Science	https://jupyter.up2digischool.eu/ https://notebook.classroom.pionier.net.pl/

3.1.2 Open Data Systems

Public pilots and demonstrators of data science environments are available at WWU, SURF and SUNET. Sciebo EFSS system at WWU runs production grade ScieboRDS service, the remaining pilots are in their early pilot and/or pre-production phase.

Table 3 below summarizes the Open Data Systems demonstrators available:

Table 3: Open Data Systems demonstrators by CS3MESH4EOSC

The name	Status / short info	Availability	Location / URL:
ScieboRDS @WWU	Production grade. Based on original ScieboRDS and OC10.	available to Sciebo users across North-Rhine Westfalia (Germany)	https://www.research-data-services.org/
SURFdrive @SURF	Pre-production grade. Based on ported ScieboRDS and OC10	available to SURFdrive users: from research and education sector	https://wiki.surfnet.nl/pages/viewpage.action?pageId=89708109
SUNET drive @SUNET	Pilot based on ported ScieboRDS and Nextcloud	Internal testing in operations department at SUNET	https://snd.gu.se/en/describe-and-share-data (current research repository, ScieboRDS integration in future)

3.1.3 Collaborative Documents

Public pilots and demonstrators of the federated collaborative documents platforms are available across many sites within ScienceMesh. Table 4 shows only selected examples of the solution deployments at CERN and WWU:

Table 4: Collaborative Editing platforms demonstrators by CS3MESH4EOSC

The name	Status / short info	Availability	Location / URL:
CERNBox @CERN	Production grade. Based on CERNbox (OCIS), Collabora, OnlyOffice, CodiMD, Etherpad.	available to registered CERN users and project participants	https://cernbox.cern.ch/
Sciebo @WWU	Production grade. Based Sciebo EFSS on OC10 and Collabora	available to Sciebo users across North-Rhine Westfalia (Germany)	https://hochschulcloud.nrw/

3.1.4 On-demand data transfers

On-demand data transfer solutions have been extensively tested in the closed environments in the sites participating in the development and testing. There is no publicly available demonstrators at the moment. The live demonstration performed to the open audience is listed in section 2.1.4.

3.1.5 Event management systems

The work on enabling Indico, the CERN-provided event management system in the reporting period focused on making this functionality available at CERN, therefore there is only one publicly available demonstrator, that can be used by registered event participants and owners of CERNBox account - it is shown in Table 5 below.

Table 5: Event management system integration demonstrators by CS3MESH4EOSC

The name	Status / short info	Availability	Location / URL:
CERNBox @CERN	Production grade. Based on Indico instance @CERN and CERNbox + FilePicker	Available to Indico event management system users. CERNBox access available to registered CERN users and collaborators.	https://indico.cern.ch/

4 Summary and conclusions

In this report we summarized the design, development and integration work across four application areas of CS3MESH4EOSC project as well as one new, extra use-case. We have also summarized the course and the results of the evaluation with the users groups.

The WP4 activities on integrating the application-level functionality concerned working out the demonstrators and performing the evaluation of the advanced prototypes as well as opening to the wide public and real-life testing the pre-production and production versions of the services.

The purpose of the evaluation was to validate and confront the services functionality, features, quality, performance, reliability, scalability and usability. i.e. experience related to the interfaces design, reactivity etc. with the design assumptions and the requirements and expectations of the end users groups.

The overall conclusion from this stage of work is that application functionality integrations worked out reached the maturity level that enables presenting and exposing them to the larger, external users groups. This aim has been achieved for all application areas. In several cases, the public demonstrators have been made available to the wide public. In other cases, extensive testing of the application components and their integrations with the mesh have been conducted in the early adopters groups.

The feedback coming from preparing and running the public demonstrators as well as performing application components and their integration validations has been taken into account while adopting the design, implementation and integration over the second part of the project.

In order to complete the full validation of the application-level solutions, the application components, as well as integrated application-EFSS and application-federation software stacks, have to be validated. This aspect has been included in the M18-M36 work already, however, it deserves a special attention and dedicated efforts.

Portability is one of the key aspects that may influence the practical usability of the project results beyond the environment of sites that participated in the project and the sustainability of the project results. The portability of the software components can also impact the level of adoption of the CS3MESH4EOSC application-level products as well as the adoption of the federation concept and its implementation in general.

The dedicated deliverable of the project - the D4.4 is devoted to cover these aspects.