

D5.9 Service deployment and cloud computing on distributed Nordic cloud resources

Author(s)	Helmut Neukirchen (Ulce), Ernir Erlingsson (Ulce), Lorand Janos Szentannai (Sigma2), Claudio Pica (SDU), Dan Sebastian Thrane (SDU), Matthias Obst (GU), Tewodros Deneke (CSC), Abdulrahman Azab (UiO)
Status	Final
Version	2.0
Date	10. October 2022

Document identifier:	
Deliverable lead	SIGMA2
Related work package	WP5
Author(s)	Helmut Neukirchen (Ulce), Ernir Erlingsson (Ulce), Lorand Janos Szentannai (Sigma2), Claudio Pica (SDU), Dan Sebastian Thrane (SDU), Matthias Obst (GU), Tewodros Deneke (CSC), Abdulrahman Azab (UiO)
Contributor(s)	
Due date	1. November 2022
Actual submission date	
Reviewed by	Ilja Livenson (ETAIS), Anders Sjöström (LU), Adil Hasan (Sigma2)
Approved by	
Dissemination level	Public
Website	https://www.eosc-nordic.eu/
Call	H2020-INFRAEOSC-2018-3
Project Number	857652
Start date of Project	01/09/2019
Duration	36 months
License	Creative Commons CC-BY 4.0
Keywords	Service deployment, Cloud computing

I

Abstract:

This deliverable aims at showcasing service deployment and cloud computing on distributed Nordic cloud resources. The demonstrator was applied to leverage computing workflows involving artificial intelligence algorithm for language analysis, and federated machine learning for marine biodiversity exploration using distributed cloud resources, as well as climate modeling using machine learning.



Copyright notice: This work is licensed under the Creative Commons CC-BY 4.0 licence. To view a copy of this licence, visit <https://creativecommons.org/licenses/by/4.0>.

Disclaimer: The content of the document herein is the sole responsibility of the publishers and it does not necessarily represent the views expressed by the European Commission or its services.

While the information contained in the document is believed to be accurate, the author(s) or any other participant in the EOSC-Nordic Consortium make no warranty of any kind with regard to this material including, but not limited to the implied warranties of merchantability and fitness for a particular purpose.

Neither the EOSC-Nordic Consortium nor any of its members, their officers, employees or agents shall be responsible or liable in negligence or otherwise howsoever in respect of any inaccuracy or omission herein.

Without derogating from the generality of the foregoing neither the EOSC-Nordic Consortium nor any of its members, their officers, employees or agents shall be liable for any direct or indirect or consequential loss or damage caused by or arising from any information advice or inaccuracy or omission herein.

Table of Contents

Table of Contents	2
Table of Abbreviations	4
Glossary	5
1. Introduction	6
2. Nordic cloud infrastructures	7
2.1 cPouta	7
2.2 NIRD Service Platform and NIRD Toolkit	7
2.3 UCloud	7
2.4 SNIC	8
3. Use cases	8
3.1 Machine Learning for Climate Modeling	8
3.1.1 Background and scientific value	8
3.1.2 Resources and services	10
3.1.3 Machine learning approach	10
3.1.4 Challenges, blockers	11
3.1.5 Status and outlook	12
3.2 Marine biodiversity exploration using cross-border resources and federated machine learning	12
3.2.1 Background and scientific value	12
3.2.2 Objective	14
3.2.3 Workflows	15
3.2.4 Technical Solution	16
3.2.5 Possibilities and options	21
3.2.6 Challenges, possible blockers	21
3.2.7 Status and outlook	22
3.3 Nordic Digital Humanities Laboratory data analytics on UCloud	23
3.3.1 Background	23
3.3.2 Objective	24
3.3.3 Workflows	24
3.3.4 Technical Overview	25
3.3.5 Technical Solution	29
3.3.6 Possibilities	31
3.3.7 Challenges, possible blockers	32
4. Benefits and lessons learned	33
5. Conclusions	34

Table of Abbreviations

Table 1. Abbreviations appearing in the document

Abbreviation	Explanation
AI	Artificial Intelligence
API	Application Programming Interface
AUV	Autonomous Underwater Vehicle
CPU	Central Processing Unit
CSC	IT Center for Science
EOSC	European Open Science Cloud
FEDn	Model agnostic framework for hierarchical federated machine learning
FedML	Federated machine learning
GB	Gigabyte
GPU	Graphical Processing Unit
HPC	High Performance Computing
IaaS	Infrastructure as a Service
KSO	Koster Seafloor Observatory
LoA	Levels of Assurance
ML	Machine Learning
MPI	Message Passing Interface
NDHL	Nordic Digital Humanities Laboratory
NIRD	National Infrastructure for Research Data
NIRD SP	NIRD Service Platform
NLP	Natural Language Processing
OS	Operating System
PaaS	Platform as a Service
PI	Principal Investigator
RI	Research infrastructure
ROV	Remotely Operated Vehicle
SaaS	Software as a Service
SDU	University of Southern Denmark
SGD	Stochastic Gradient Descent
SP	Service provider
STACKn	Scaleout's machine learning platform
TLS	Transport Layer Security
vCPU	Virtual CPU, subpart or share of a physical CPU core
VM	Virtual Machine

Glossary

Table 2. A brief dictionary of terminologies appearing in the document

Term	Definition
Cloud computing	On-demand availability of computer resources (storage and computing), without direct active management by the user
Container	Packages of software containing all of the necessary elements to run in any environment
Docker	A set of platform as a service products that use OS-level virtualization to deliver software in packages called containers
Docker Compose	A tool to help define and share multi-container applications
Helm chart	A collection of files that describe a related set of Kubernetes resources
Kubernetes	An open-source container orchestration system for automating software deployment, scaling, and management
OpenStack	Cloud computing platform, mostly deployed as IaaS
Pod	A pod is the smallest execution unit in Kubernetes

1. Introduction

The goal of the T5.2.3 subtask is to investigate and showcase innovative platforms for making scientific tools able to discover and consume cloud computing resources, such as IaaS and Kubernetes solutions running either on premises and/or on public clouds.

The subtask investigated solutions provided by Sigma2's NIRD Toolkit¹, STACKn² and FEDn³ from Scaleout⁴, cPouta⁵ from CSC, as well as UCloud⁶ from SDU. These provider solutions were selected based on the user experience and eligibility to host and analyze the associated research data. The goal is to provide a platform for researchers to deploy and run a variety of tools on user-selected data and cloud computing resources, giving researchers the possibility to focus mainly on research and less on tools and irrelevant tasks. The solutions we have investigated as part of the subtask are to enable AI/ML workflows for natural language analysis, climate modeling and biodiversity use cases and to provide cloud infrastructure resources as well as to enable sharing of workflows across borders.

Among tasks and activities, STACKn (formerly known as LeanAI), was brought into EOSC-Nordic. STACKn was deployed on Kubernetes resources at Sigma2 on the NIRD Service Platform (NIRD SP), and at Scaleout. This gave opportunities for demonstrating cross-border cloud computing capabilities and facilitating dialog for collaboration with the NLP community. The NLP community has channeled their focus and efforts on consuming more traditional HPC resources. Since slightly different needs arose from the marine biodiversity community, FEDn was chosen as the technical platform solution.

The task investigated possibilities for enhancing the interoperability between the cloud container-based computing and the traditional HPC to support compute-intensive AI/ML workflows from the EOSC-Nordic research community. A proof concept based on UCloud has been developed, deployed and thoroughly tested on the NIRD Toolkit.

Furthermore, this subtask discussed and evaluated possibilities for setting up a Nordic cloud toolbox, where other service providers would have one single entry point for accessing all the tools and cloud infrastructure enablers in one place.

Finally, effort has been invested into prototyping accounting for better cloud resource distribution, optimal utilization, and improved resource planning and prediction.

After a close dialogue with communities, three main use cases have been investigated and worked on to develop solutions which can support research, facilitate collaboration and scale out of silos offering scalable cloud computing solutions.

¹ <https://apps.sigma2.no>

² <https://github.com/scaleoutsystems/stackn>

³ <https://github.com/scaleoutsystems/fedn>

⁴ <https://www.scaleoutsystems.com>

⁵ <https://research.csc.fi/-/cpouta>

⁶ <https://cloud.sdu.dk/app/login>

2. Nordic cloud infrastructures

Cloud infrastructure efforts participating in the WP5 used cases of EOSC-Nordic are listed and shortly described below to help the reader understand the different solutions, concepts and services enabling the use cases described further below.

2.1 cPouta

cPouta at CSC, Finland, is based on the open source cloud software called OpenStack⁷ and is a generic service which can be used for many tasks. One can use cPouta to build their own service, do a quick test, have a development platform or build a data processing pipeline.

cPouta provides: i) high performance computing with superior flexibility and user experience via IaaS; ii) self-service model for accessing, using and managing virtualized infrastructure; iii) deployment of resources such as VMs, storage and networks; iv) variety of resources like block devices, virtual networks, HPC and GPUs.

2.2 NIRD Service Platform and NIRD Toolkit

The NIRD Service Platform⁸ (NIRD SP) at Sigma2, Norway, is a Kubernetes-based cloud infrastructure, enabling several types of services and software. NIRD SP allows researchers to run cloud services, such as web services, domain- and community-specific portals, tools for data visualization, pre-/post-processing, data discovery and data sharing.

The services run in containers to ensure high portability of the tools and reproducibility of the results. NIRD SP is part of NIRD - the National Infrastructure for Research Data - an ecosystem of storage and cloud services designed to support scientific research in every step of the research data life cycle. The services can be used to consume data in place, without moving and staging. NIRD SP can host any service and services can have a permanent web address or be launched on-demand on the NIRD Toolkit.

NIRD Toolkit provides on-demand access to a variety of predefined applications, such as Apache Spark, Deep Learning Tools, Jupyter Notebook, Jupyter Hub, MinIO and RStudio. The NIRD Toolkit can be enriched with even more tools.

2.3 UCloud

UCloud at SDU, Denmark, is designed to be a user-friendly solution for research with an intuitive graphical user interface. The solution is flexible and extensible to account for the multi-scale and multi-disciplinary research challenges, and the high data intensity and heterogeneity. The focus of UCloud is to make complex digital technology accessible to all users. The platform runs on premises hosted at the University of Southern Denmark, but it can also be used to federate remote infrastructure operated by another SP.

UCloud can provisioning system is flexible, including bare-bone⁹ virtual machines available via terminal to complex software solutions in just a few seconds. The cloud infrastructure natively supports multi-tenancy, with a separation of data and computing resources. It comes equipped with advanced data analytics tools

⁷ <https://www.openstack.org>

⁸ <https://www.sigma2.no/nird-service-platform>

⁹ Each VM is mainly consuming the resources of a full bare-metal node

for data processing and visualization, many popular frameworks being pre-installed and optimized for the user experience.

2.4 SNIC

SNIC Science Cloud (SSC)¹⁰ is a large-scale, geographically distributed OpenStack cloud Infrastructure as a Service (IaaS), intended for Swedish academic research. SNIC Science Cloud is funded by the Swedish Research Council (Vetenskapsrådet) through SNIC, and is available free of charge to researchers at Swedish higher education institutions through open application procedures. Other research infrastructures are also welcome to join SSC with a co-funding model with dedicated capacity. Platforms may be added to SSC in order to support the Swedish research community as seen fit by SNIC (PaaS).

Some resources from SSC have been used by the marine biodiversity use case. This has been described in Deliverable D5.2 Cross-borders computing through portals¹¹.

3. Use cases

The following pilots serve as uses cases to develop and study service deployment and cloud computing on distributed Nordic cloud resources:

1. Machine Learning for Climate Modeling
2. Marine biodiversity exploration using cross-border cloud resources and federated machine learning
3. Nordic Digital Humanities Laboratory data analytics on UCloud

3.1 Machine Learning for Climate Modeling

3.1.1 Background and scientific value

Climate change modeling involves many computations, e.g., concerning the Earth's energy budget that accounts for the balance between the energy that the Earth receives from the Sun, and the energy the Earth radiates back into outer space (see Figure 1).

¹⁰ <https://cloud.snic.se/>

¹¹ Abarenkov, K.; Fouilloux, A.: D5.2 Cross-borders computing through portals, Deliverable, EOSC-Nordic 2021, DOI: [10.5281/zenodo.4607199](https://doi.org/10.5281/zenodo.4607199)

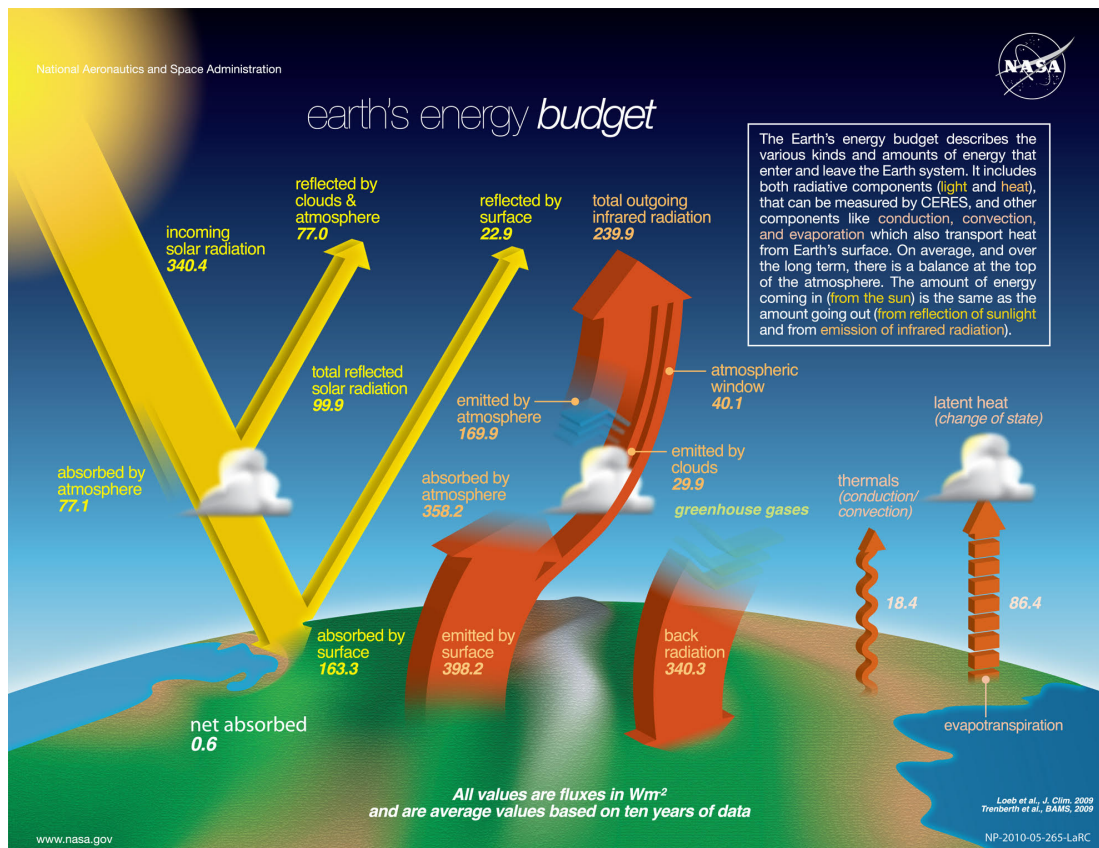


Figure 1. Schema of the Earth's energy budget (image source: nasa.gov).

In this case study, we focus on computations related to modeling the radiative transfer of aerosol. The reflectivity properties of different aerosol compositions in the atmosphere are an important part of calculating the Earth's energy budget. The number of aerosol combinations, however, are vast and complex (see Figure 2 to get a rough idea). Therefore, it is extremely time-consuming to compute a good approximation. In NorESM, this is done using a software module called Aerotab^{12, 13} which is implemented using Fortran code that is hard to scale and to port to new platforms. To address this, this use case investigates using deep learning to train a surrogate model, i.e., a neural network that has been trained using the computational outcomes for many input parameters so that it can infer these outcomes without needing to do the time-consuming calculations of the complex climate model. The input and output values were obtained by running the legacy code. The learned surrogate model can then be used as a replacement for the legacy code. Using standard deep learning software frameworks leads to better portability and performance, being in fact even faster than the complex original computations (while training a neural network is computationally intensive, inferring, i.e., using the trained neural network, is relatively fast).

¹² Kirkevåg, A., Grini, A., Oliví, D., Seland, Ø., Alterskjær, K., Hummel, M., Karset, I. H. H., Lewinschal, A., Liu, X., Makkonen, R., Bethke, I., Griesfeller, J., Schulz, M., Iversen, T.: A production-tagged aerosol module for Earth system models, OsloAero5.3 – extensions and updates for CAM5.3-Oslo, Geosci. Model Dev., 11, 3945–3982, 2018. DOI: [10.5194/gmd-11-3945-2018](https://doi.org/10.5194/gmd-11-3945-2018)

¹³ https://noresm-docs.readthedocs.io/en/latest/faq/aero_faq.html

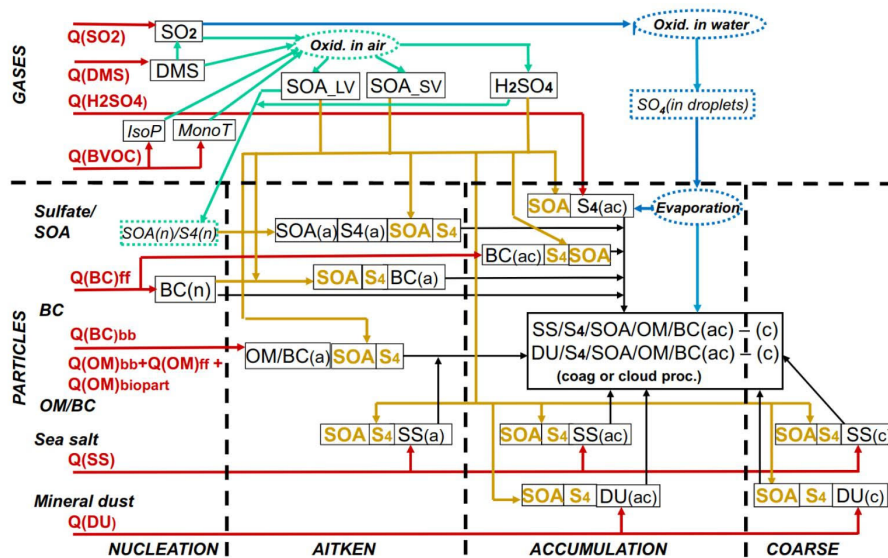


Figure 2. Schema of aerosols influencing the Earth’s energy budget (image source: [CC BY 4.0 © 2018 Kirkevåg et al. https://doi.org/10.5194/gmd-11-3945-2018](https://doi.org/10.5194/gmd-11-3945-2018))

3.1.2 Resources and services

To make running the original code portable, it was packaged as a container that includes everything needed to build (e.g., the matching Fortran compiler) and run the code on any container-enabled system, ranging from a local system to distributed cloud resources.

The usage of container-based cloud and HPC resources by the climate community via the Galaxy portal has in detail been described in Section 2.2 of Deliverable D5.2 Cross-borders computing through portals¹⁴ and will be covered in a peer-reviewed publication¹⁵ and is therefore not repeated here again.

3.1.3 Machine learning approach

At first, the machine learning approach was to devise regressive deep learning models; i.e., a feed-forward neural network using Tensorflow¹⁶ and Keras¹⁷ to eliminate the need for extensive auxiliary lookup tables employed by Aerotab. However, this presented a problem as the lookup tables could not supply enough training data to yield a good regression model with deep learning. Therefore, the approach was fundamentally changed for the better by taking the model a step further and have it replace the whole Aerotab application instead of only replacing internal property lookup tables. By targeting the whole application a near infinite supply of training samples could be guaranteed and the usability potential of the neural network significantly increased.

¹⁴Abarenkov, K.; Fouilloux, A.: D5.2 Cross-borders computing through portals, Deliverable, EOSC-Nordic 2021, DOI: [10.5281/zenodo.4607199](https://doi.org/10.5281/zenodo.4607199)

¹⁵Abarenkov, K.; Fouilloux, A.; Neukirchen, H.; Azab, A.: Reproducible Cross-border High Performance Computing for Scientific Portals, Workshop on Reproducible Workflows, Data Management, and Security (REWORDS), IEEE, 2022, to appear. DOI: <https://doi.org/10.48550/arXiv.2209.00596>

¹⁶<https://www.tensorflow.org>

¹⁷<https://keras.io>

Unfortunately, however, the undertaking proved to be too much for the limited time this part of the project was assigned. At first the machine learner had to gain a deeper domain knowledge of the Aerotab application and its usage to correctly build it in the Docker container, which contained only source code that needed to be built by the container itself. When correctly applied, which was not always the case, the internal build process took around 100 core hours and could only run on a single core on a single node as Aerotab is made up of legacy Fortran code that is difficult to parallelize. The duration of this process is due to the application's necessity in generating the aforementioned internal lookup tables before its result can be considered accurate. Furthermore, this proved to be a painstaking process as Aerotab does not notify its user explicitly if it has been incorrectly set up, either reporting Fortran errors that were cryptic to the ML expert, or inaccurate results which lead to troves of generated training data being discarded.

The study was successful in devising a realistic approach of building regressive deep learning models to replace Aerotab, and estimate the radiative transfer of aerosol directly. However, the study could not provide a conclusive proof of the approach or its efficacy due the difficult learning process of Aerotab, its domain, and the case study's time duration. A follow up of this activity could be part of a future NeIC coordinated EOSC-Nordic Project Affiliate.

3.1.4 Challenges, blockers

In fact, problems well known from classical software engineering research occurred in this use case combined with new problems from machine learning.

The required efforts have been underestimated. Despite recent developments, such as automated hyperparameter tuning, machine learning (ML) is still a task involving trial-and-error experiments which require both significant human and computational resources. Doing a human resource intensive ML sub-task in a more infrastructure-centric research project that has not many human resources allocated to each sub-task was not appropriate.

Communication between experts from different domains (in this case: climate and ML) is a known problem – the Interaction Room method could help to avoid this in future: it facilitates communication between experts and stakeholders from different domains and thus achieves a common understanding: coming from business applications, it has been adapted¹⁸ to pure simulation science HPC projects¹⁸, and is currently adapted for AI projects in the European Centre of Excellence (CoE) Research on AI- and Simulation-Based Engineering at Exascale (RAISE)¹⁹.

A known problem from data science and ML is availability (and quality) of data sets. In this sub-task, the data needed as input for ML first needed to be generated by running the legacy code. This did require a significant amount of time in this sub-task. In fact, the Cross-Industry Standard Process for Data Mining

¹⁸ Book, M., Riedel, M., Neukirchen, H., Götz, M.: Facilitating Collaboration in High Performance Computing Projects with an Interaction Room. The 4th ACM SIGPLAN International Workshop on Software Engineering for Parallel Systems (SEPS 2017). DOI: [10.1145/3141865.3142467](https://doi.org/10.1145/3141865.3142467), ACM Digital Library 2017.

¹⁹ <https://www.coe-raise.eu/> and <https://www.coe-raise.eu/news-2021-04>

(CRISP-DM)²⁰ describes a process to be used for data mining projects, which notably includes the phases Data Understanding and Data Preparation, thus making clear that this must not be underestimated.

Also, while machine learning experts are fluent in new technologies (e.g. Python-based), they are typically not used to complex environments such as huge Fortran-based earth system model implementations that have grown over decades. Getting familiar with these huge software packages takes time and therefore, introducing machine learning in legacy projects is not trivial.

3.1.5 Status and outlook

Due to lack of human resources combined with the challenges described above, this case study had to be discontinued before being able to prove the efficacy of its approach. Nevertheless, replacing the whole computation itself by a surrogate ML model is an emerging topic²¹ that promises to be worthwhile to investigate further in the future in order to improve scalability and portability. However, the lessons learned from the above challenges need to be taken into account, when doing so.

3.2 Marine biodiversity exploration using cross-border resources and federated machine learning

3.2.1 Background and scientific value

The increasing access to autonomously-operated technologies offer vast opportunities to sample large volumes of biological data²². This is especially the case for high-definition optical imagery coming from remotely operated vehicles (ROVs), autonomous underwater vehicles (AUVs), drones, drop-cameras, camera-traps, and video plankton recorders. The scientific potential of these technologies will change the modus operandi of the ecological research community in the near future. The access to data from camera-based systems allows scientists to extract biological information from remote ecosystems with unprecedented quantity and quality. However, these technologies also impose novel demands on ecologists who need to apply tools for data management and processing that are efficient, publicly available and easy to use. Such tools are starting to be developed in the community, but they are rarely connected to e-infrastructures for data management, archiving, and computation.

In this pilot use case, we explored the integration of one image-analysis system with EOSC Nordic resources. We used a system that was developed under the NeIC program DeepDive²³ called Koster Seafloor Observatory (KSO). KSO is an open-source platform to analyze large amounts of subsea movie data for marine ecological research. The KSO system architecture is shown in Figure 3. The system incorporates three distinct modules to: manage and archive the subsea movies, involve citizen scientists to accurately

²⁰ Shearer, C.: The CRISP-DM model: the new blueprint for data mining, *J Data Warehousing* (2000); 5(4):13–22.

²¹ Weber, T., Corotan, A., Hutchinson, B., Kravitz, B., Link, R.: Technical note: Deep learning for creating surrogate models of precipitation in Earth system models. *Atmos. Chem. Phys.*, 20, 2303–2317, 2020. DOI: [10.5194/acp-20-2303-2020](https://doi.org/10.5194/acp-20-2303-2020)

²² Guidi, L., Fernandez Guerra, A., Canchaya, C., Curry, E., Foglini, F., Irisson, J.-O., Malde, K., Marshall, C. T., Obst, M., Ribeiro, R. P., Tjiputra, J., Bakker, D. C. E. (2020) *Big Data in Marine Science*. In: Alexander, B., Heymans, J. J., Muñiz Piniella, A., Kellett, P., Coopman, J. [Eds.] Future Science Brief 6 of the European Marine Board, Ostend, Belgium. ISSN: 2593-5232. ISBN: 9789492043931. DOI: [10.5281/zenodo.3755793](https://doi.org/10.5281/zenodo.3755793)

²³ <https://neic.no/affiliate-deepdive/>

classify the footage and, finally, train and test machine learning algorithms for detection of biological objects²⁴ (see Figure 4). This modular approach allows researchers to customize and further develop key functionalities to fit various types of data and research questions related to analysis of marine imagery. This adds more usability to the user experience.

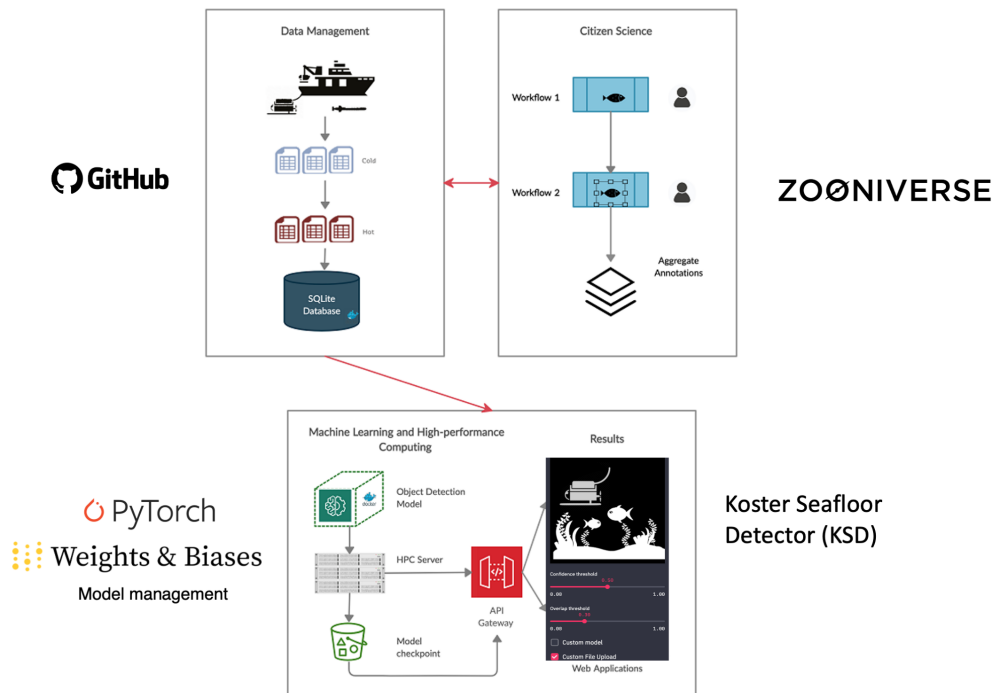


Figure 3. KSO system architecture. Detail are described by Anton et al 2021 and updated on <https://www.zooniverse.org/projects/victorav/the-koster-seafloor-observatory/about/results>

KSO is currently used in numerous scientific studies of marine biodiversity. These include

- Habitat maps for mussel beds in the Baltic Sea.
- Monitoring of Cold water corals in Marine Protected Areas.
- Impact of trawling on soft corals (sea pens) in Kattegat and Skagerrak.
- Monitoring and trophic interactions of invasive fish (round goby) in Nordic waters.
- Application of Generative Adversarial Networks (GANs) for rare species monitoring.
- Deep Learning methods to predict engagement of citizen scientists.

²⁴ Anton, V., Germishuys, J., Bergström, P., Lindegarth, M., Obst M. An open-source, citizen science and machine learning approach to analyse subsea movies. Biodiversity Data Journal 9: e60548. 2021 DOI: [10.3897/BDJ.9.e60548](https://doi.org/10.3897/BDJ.9.e60548)

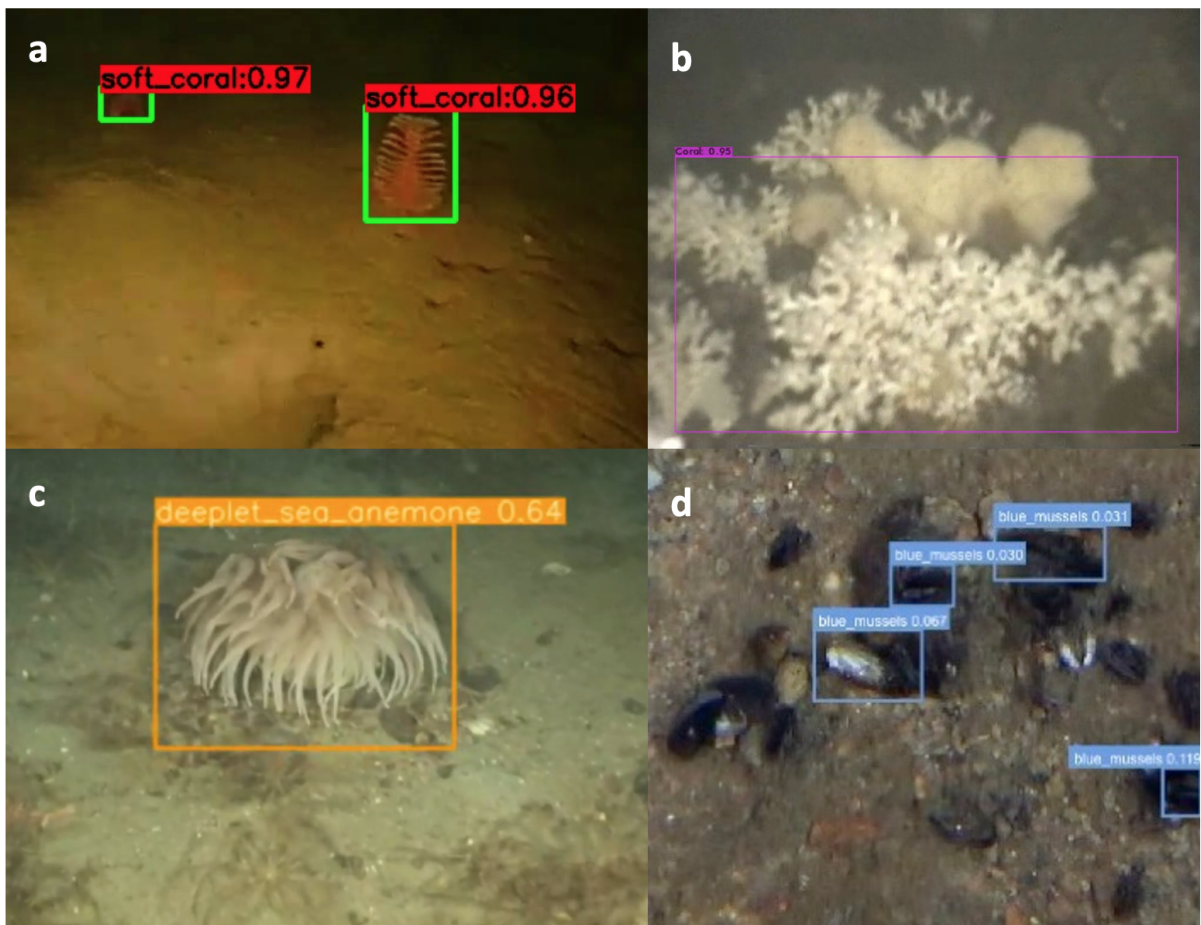


Figure 4. Examples of object detection ML models developed in KSO for ecological key species and habitat builders in Nordic marine ecosystems, showing (a) the soft coral *Pennatulula phosphorea* a key species of Nordic soft bottom ecosystems which is affected by trawling; (b) the cold-water coral *Desmophylum pertusum* which occurs especially in Norwegian and Swedish waters and is declining due to trawling and climate change; (c) the soft coral *Bolocera tuedie* provides shelter for many other species in Nordic waters, and (d) the blue mussel *Mytilus edulis/trossulus* which as a dominant habitat builder in all Nordic waters – from the eastern Baltic to the arctic waters of the Barents Sea. The ML models are used for creating high resolution habitat maps of key species and habitats, and for analyzing changes of marine biodiversity across large spatio-temporal scales.

3.2.2 Objective

Although image analysis tools and systems are starting to be developed by the scientific community, they are rarely connected to e-infrastructures for data management, archiving, and computing facilities. The lack of connectivity between data and analysis systems is a major problem – both for the data research e-infrastructures and for the community built systems. In this pilot we attempt to explore connecting cloud infrastructures and develop the connections with research e-infrastructures as much as possible, including linkage to HPC systems which are part of EOSC (e.g., SNIC resources for storage and computation).

Many of these studies deal with large data sets which are difficult to copy/move. However, as underwater data are also sensitive if linked to geographic and depth information, this project explored the possibilities of using cloud resources in providing infrastructure and computing capabilities for analyzing datasets in place where the data resides.

3.2.3 Workflows

KSO developed a series of modules for data management, image annotation (including citizen science), training and testing models, as well as data submission. These modules are made available through Jupyter notebooks and can be combined in analytical workflows. Each notebook is a self-standing workflow including a guided tutorial and allows users to perform a specific task of the system (e.g. upload footage to the citizen science platform or analyze the classified data). The notebooks rely on the Koster utility functions²⁵ and are detailed below (Figure 5 and Table 3). The published notebooks have been deployed with slight adjustments to the underlying infrastructure, e.g., cPouta and NIRD Service Platform.

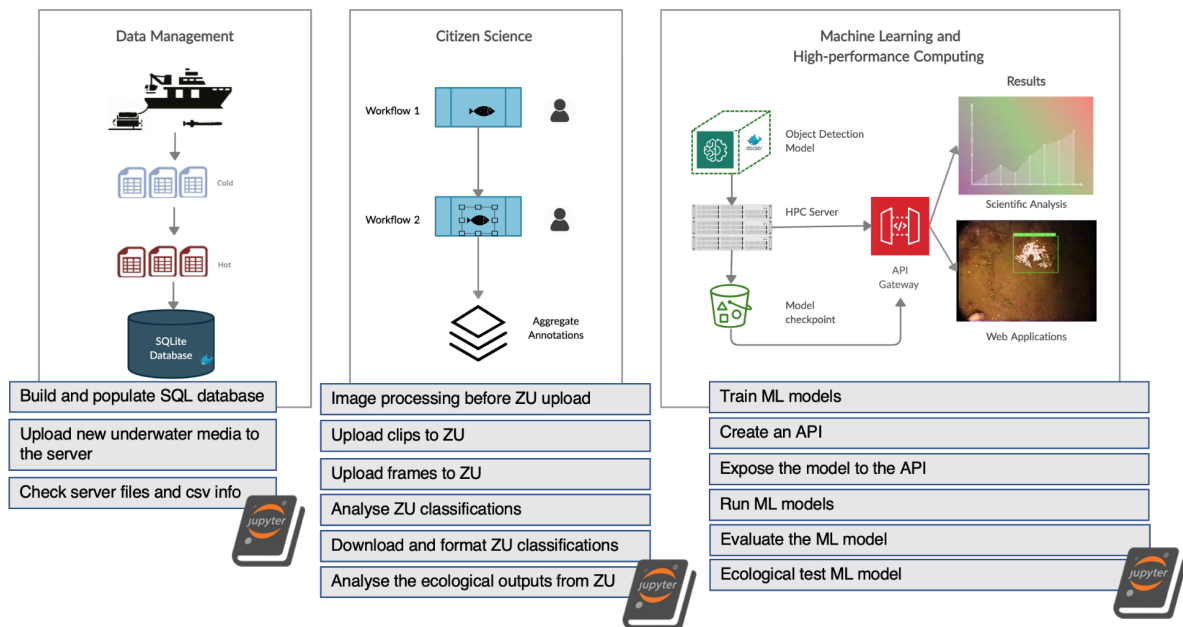


Figure 5. KSO workflows available on Ocean Data Factory's Github site²⁶

²⁵ https://github.com/ocean-data-factory-sweden/kso_utils

²⁶ <https://github.com/ocean-data-factory-sweden/>
















Name	Description	Try it!
1. Check and update csv files	Check and update initial information about sites, media and species of interest	Coming soon
2. Upload new footage	Upload new underwater media to the cloud/server and update the csv files	 Open in Colab  launch  binder
3. Upload clips to Zooniverse	Prepare original footage and upload short clips to Zooniverse	 Open in Colab  launch  binder
4. Upload frames to Zooniverse	Extract frames with animals of interest from original footage and upload them to Zooniverse	Coming soon
5. Train ML models	Prepare the training and test data, set model parameters and train models	 Open in Colab  launch  binder
6. Evaluate ML models	Use ecologically-relevant metrics to test the models	 Open in Colab  launch  binder
7. Expose ML models	Expose model to the API	Coming soon
8. Analyse Zooniverse classifications	pull up-to-date classifications from Zooniverse and report summary stats/graphs	 Open in Colab  launch  binder
9. Download and format Zooniverse classifications	pull up-to-date classifications from Zooniverse and format them for further analysis	Coming soon
10. Publish classifications	Publish classifications to OBIS	Coming soon

Table 3. Overview of current KSO notebooks with short description of function and state of development.

3.2.4 Technical Solution

Current machine learning practices often require centralizing data to a machine or a datacenter. This means a secure, robust, and often custom-made cloud infrastructure (e.g., NIRD Service Platform or ePouta) is required in advance before any data movement could happen, especially for training a model on non-person sensitive data. In addition, the legal procedures related to data movement across borders has to be discussed and agreed upon before any model training can happen.

Federated machine learning (FedML) can be used as a viable alternative to centralized analysis. FedML is often employed when data can not be moved to a centralized place either due to privacy and security requirements or size considerations. In such cases FedML is employed to train models on client facilities or devices (e.g., to learn a model on client interaction patterns or train a species classification model on local clusters). In FedML, multiple parties, or clients, jointly train a model while keeping data local and private which means one moves computation instead of data. Incremental model updates are computed locally and combined into a global model. Currently, FedML work is mainly focused on deep learning which is based on

Stochastic Gradient Descent (SGD)²⁷. Some support for data analysis (e.g., correlation analysis) and federated learning approaches for tree-based algorithms such as random forest are also coming to light. Challenges in this area include system scalability due to uneven distribution of data, preserving client privacy and system fault tolerance. Also, iterative optimization (e.g., SGD) needs low latency communication which becomes a challenge when clients are distributed across different network domains. One needs to remember also that no direct access might be feasible to the labeling of raw data.

Lot of effort is currently going towards innovative averaging algorithms that enhance system scalability, compression and quantization schemes for partial model upload and update efficiency, and secure aggregation protocols (e.g., protocols that allow no individual client updates can be directly inspected or decrypted by the reducer) to enhance privacy of the system. Other algorithmic improvements include efficient learning on sparse data.

There are several open source frameworks for facilitating federated machine learning. In this project we used FEDn²⁸ which is a scalable, modular, and model agnostic framework for hierarchical federated machine learning. Stateless model combiners work independent of others on a single partial model update at a time, making the system robust and scalable. The security of a FEDn network is protected through a secret token exchange among its members. As FEDn scales massively horizontally, showing good performance cross-device and cross-silos, it is suitable for being used in cross-border cloud computing, supporting the use case where data cannot leave premises and computing is following data.

Other federated learning frameworks include TFF²⁹, which provides quantization techniques to reduce communication overheads during distributed model training. PySyft and FATE which provide homomorphic encryption features to enhance security. PaddleFL provides a federated averaging scheme rather than a hierarchical or centralized one.

3.2.4.1 Deploying FEDn

Before we can train a model in a federated way using FEDn, we need to first deploy a reducer-combiner network. A FEDn network consists of three types of components, a *reducer*, several *combiners* and *clients*. Figure 6 shows the overall architecture of a FEDn network.

²⁷ https://en.wikipedia.org/wiki/Stochastic_gradient_descent

²⁸ <https://github.com/scaleoutsystems/fedn>

²⁹ https://www.tensorflow.org/federated/api_docs/python/tff/framework

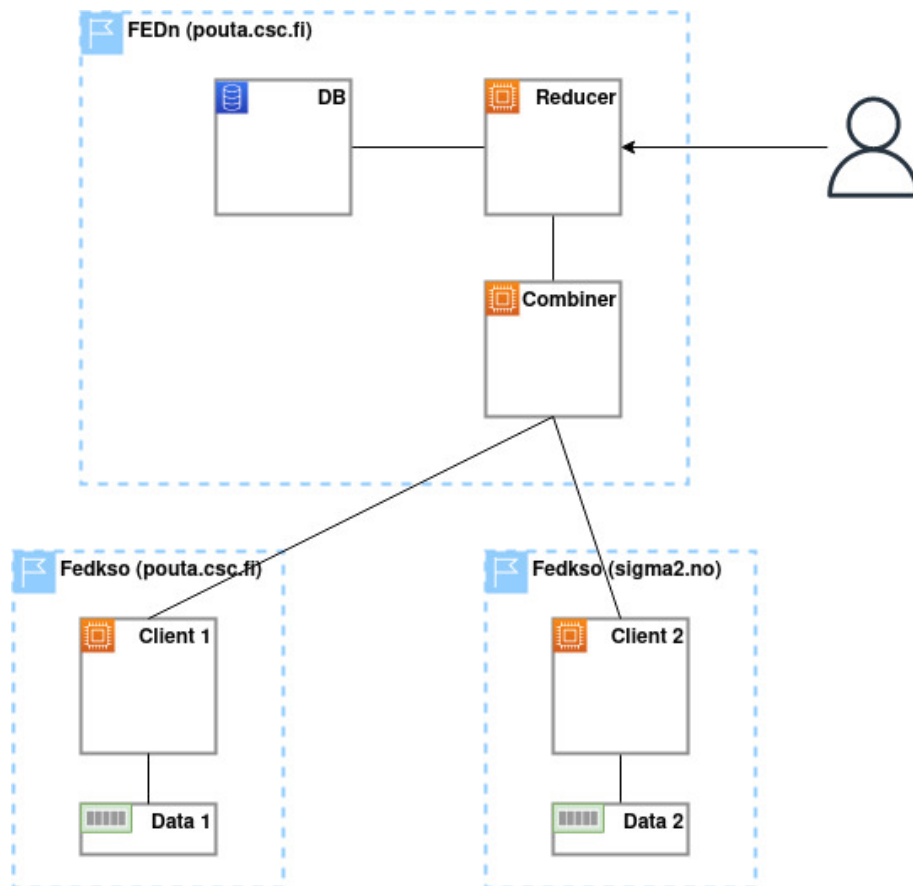


Figure 6. FEDn architecture

The role of the *reducer* in FEDn is to create a global training strategy and coordinate the *combiners* as well as dictate how to average model updates from individual *combiners* as well as dictate how to average model updates from individual *combiners*. The *reducer* also keeps the global state of our model and facilitates the discovery and connection of new *clients* and *combiners*. The role of the *combiner* is to aggregate model updates coming from the *clients* according to the strategy laid out by the *reducer*. A FEDn *client* is a component that holds private local data on which the partial model training will be done. When a *client* connects to a FEDn network for the first time, it will be provided by the *reducer* with code to be used to train and validate its model and get assigned to a *combiner*.

The deployment of FEDn as shown in Figure 7 supporting KSO consists of the following instances and base services:

- 1) a *reducer* and a *combiner* deployed at CSC;
- 2) a FEDn *client* deployed at CSC;
- 3) and a FEDn *client* deployed at Sigma2 on the NIRD Service Platform.

The deployment of the *reducer* and the *combiner* is done through Docker Compose on a virtual machine with 3 cores and 4 GB main memory running on cPouta³⁰ at CSC. cPouta is an OpenStack based cloud solution which provides virtual machines, storage, network and other compute-related resources.

³⁰ <https://pouta.csc.fi/>

The FEDn client, Fedkso³¹ is deployed on the NIRD SP and on cPouta in a dedicated VM.

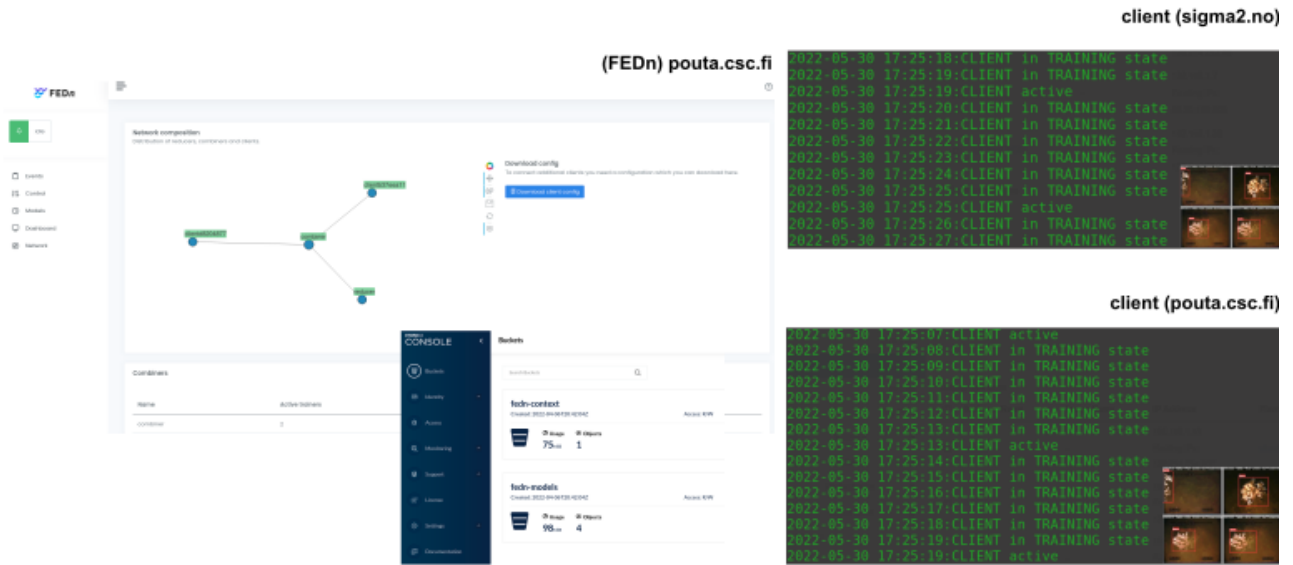


Figure 7. FEDn (including FEDn client deployments) training a KSO model at various nordic infrastructures

3.2.4.2 Packaging KSO For Federated Learning

A KSO *compute package* is a compressed (tar.gz) bundle of code that will be executed by a FEDn client to train a local model on its local data. The package is first uploaded to the *reducer* along with an initial model (i.e., *yolov5m*) during initialization of the FEDn network. Every client will be provided with this package when it joins the FEDn network, and later it will unpack and use it to train and validate its local model. Figure 8 illustrates how the package looks and is used in the current implementation.

³¹ <https://github.com/tdeneke/fedkso>

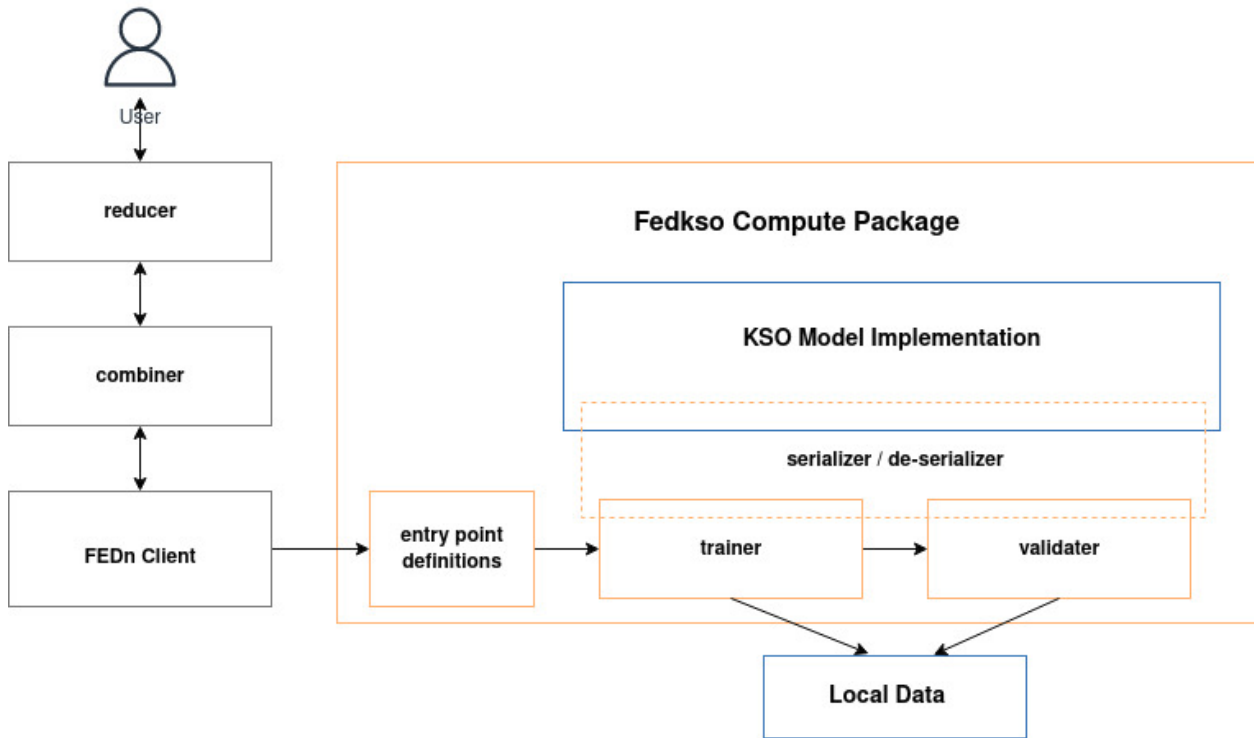


Figure 8. Visualization of FEDn - KSO pilot implementation

First, a FEDn *client* receives a model update request from the *combiner*. The *client's* dispatcher module then consults an endpoint definition file found in the package and obtains and runs the endpoint (command to execute) to produce the next model update. There are two endpoints that are required by FEDn, the train and the validate endpoint. The training endpoint expects a single input and produces a single output. It takes a global model which was aggregated in the previous epoch or an initial seed model as an input and produces an updated model based on its local data. This new updated model is then sent back to the combiner for aggregation. Custom model serialization and deserialization functions are implemented with the help of FEDn SDK to handle the integration with the KSO model training libraries. The validation endpoint is a model input and produces the validation results which can be sent back to the reducer via the combiner for monitoring and reporting of the model training process.

3.2.4.3 Deploying Fedkso Clients

Two Fedkso clients were deployed, connecting to the FEDn network supporting the KSO use case. One Fedkso client is deployed at CSC and another on the NIRD SP container cloud.

The deployment of the Fedkso client at CSC is done through Docker Compose on a virtual machine (separate from the one running the *combiner* and *reducer*), with 4 cores and 8 GB main memory running on cPouta OpenStack infrastructure.

The deployment of the Fedkso client at NIRD SP is done using Helm-charts on the Kubernetes-based cloud infrastructure. The client pod has access to 4 vCPUs and 8 GB memory.

In supporting the FEDn KSO pilot, we utilized two cloud computing platforms, the OpenStack-based Pouta Cloud and Kubernetes-based NIRD SP cloud infrastructure.

OpenStack is an open-source open standard cloud computing platform that provides virtual machines, storage, network, and such compute-related resources on demand. Instantiation of cloud resources such as virtual machines is simple and can either be done through a web UI or the REST API provided by the platform. Our FEDn pilot framework components and a FEDn client are deployed on the virtual machines through Docker, an engine for managing containerized applications. This fact makes the deployment process automated, portable and reproducible.

Another FEDn client running the KSO use case is also deployed on a container running on a Kubernetes cluster. Kubernetes is an open-source container orchestration platform which makes it easy to deploy, scale and manage containerized applications. Our Kubernetes deployments are done through the use of an automated deployment management system called Helm which takes a set of declarative deployment YAML manifestes and translates it to running application components on Kubernetes containers. This shows the adaptability of the system to various cloud platforms enabling better collaboration.

3.2.5 Possibilities and options

Federated machine learning, as well as cloud infrastructures, be it homogeneous or heterogeneous like in the showcased setup, can be an additional alternative tool facilitating open research. This approach allows research institutions to collaborate on global models together, breaking down silos and bridging across borders. Another possibility is including the components in application catalogs so that they can be used on-demand by research collaborators. This is mainly a collaborative modeling platform where most of the tools are containerised, which allows a significant level of reproducibility.

3.2.6 Challenges, possible blockers

During the work with the use case, legal blockers were identified, which hindered us in deploying the solution on a larger scale and constrained us to use test data. For example, as in our case, knowing when is it legally and technically feasible to share a model trained on private local underwater data without compromising a leakage (e.g., input data reverse engineering attack) of any sensitive geographic and depth information was a challenge.

FEDn-KSO, in addition to automated and scalable cloud deployment, provided a suitable solution for facilitating input data privacy by making it visible to local clients only. Its composable architecture also accommodates future enhancements (e.g., output/model privacy mechanisms such as the addition of controlled noise during model training to prevent input data reverse engineering). However, even though FEDn analyses data were in place, the lack of legal and technical assessment for data/information leakage became a showstopper and hindered us taking the use case further and preparing production deployment. Furthermore, it is unclear whether there are any legal blockers, or policies regulating sharing the machine learning models across borders and institutions. In the future, based on this work and observations made, we hope for a legal framework that would set a more concrete privacy threshold that needs to be met for output model (i.e., weights, parameters) sharing.

Deploying FEDn on multiple cloud infrastructures, as a consequence of heterogeneity, revealed both possibilities and challenges. It has clearly been demonstrated that the approach of deploying and providing computing capabilities for researchers on cloud infrastructure is future-oriented and a wise decision. At the

same time, adaptations for different cloud solutions had to be carried out, in the first instance requiring manual adaptations and adjustments.

At this stage it is important to disseminate the results of the FEDn pilot to domain specific infrastructures who may want to make use of such services in the future. Here important links to national, nordic, and European infrastructures dealing with biodiversity and ecology need to be consolidated. This includes, e.g., Swedish Biodiversity Data Infrastructure³² as well as European ERICs, e.g., European Marine Biological Resource Center³³ (EMBRC). Our pilots can be used to present tangible suggestions how seamless service interaction can be created between such domain specific infrastructures and more foundational e-infrastructures, such as e.g., Swedish National Data Service³⁴ (SND), Swedish National Infrastructure for Computation³⁵ (SNIC), and European Open Science Cloud³⁶ (EOSC).

The FEDn pilot encountered several blockers that turned into opportunities. During the KSO service development we developed an open API to run object detection models over new footage. This API, exposed through a website (called Koster Seafloor Detector), reached its limit very quickly with the original design because of lack of computational capacity. This blocker is currently addressed by outsourcing the computation to the SNIC Science cloud.

3.2.7 Status and outlook

Currently, FEDn-KSO provides an automated approach for distributed model training related to marine biodiversity exploration and analysis. More specifically, it provides a viable technological solution for ecological research communities in the nordic and further, for collaborating on building predictive models across borders while keeping their data samples locally and in place. In the past, such collaboration would require additional ways to facilitate data transfer and model aggregation.

The FEDn-KSO pilot was deployed with example datasets, using non-sensitive data as input for setting up, testing and training the distributed data model. However, the pilot could not move forward to a production deployment as a consequence of policy- and legal limitations.

It is important that technical, scientific, and even legal activities are synchronized and progress together. Hence, KSO will wait with further implementation of the FEDn system until a scientific use case requests this functionality, and is investing effort in sorting out the legal aspects hindering use of the demonstrated capabilities of cross-border cloud computing. Meanwhile, KSO is focusing on the technical development of the workflows.

KSO will also put emphasis on balancing the scientific use cases and the technical development more equally among the Nordic countries. To this end, KSO is preparing a user Nordic workshop (funded by NeIC) in autumn 2022. The goal of the workshop is to bring Nordic groups together, understand the architecture of various image analysis systems in biodiversity research, establish contacts between researchers and developers, and explore the potential for co-development. Discussions will focus especially on infrastructure service development, functionality for image archiving, data management, image annotation, model testing & training, as well as model execution. Figure 9 shows a roadmap of the prospective development of KSO.

³² <https://biodiversitydata.se/>

³³ <http://embrc.eu/>

³⁴ <https://snd.gu.se/sv>

³⁵ <https://www.snic.se/>

³⁶ <https://eosc-portal.eu/>

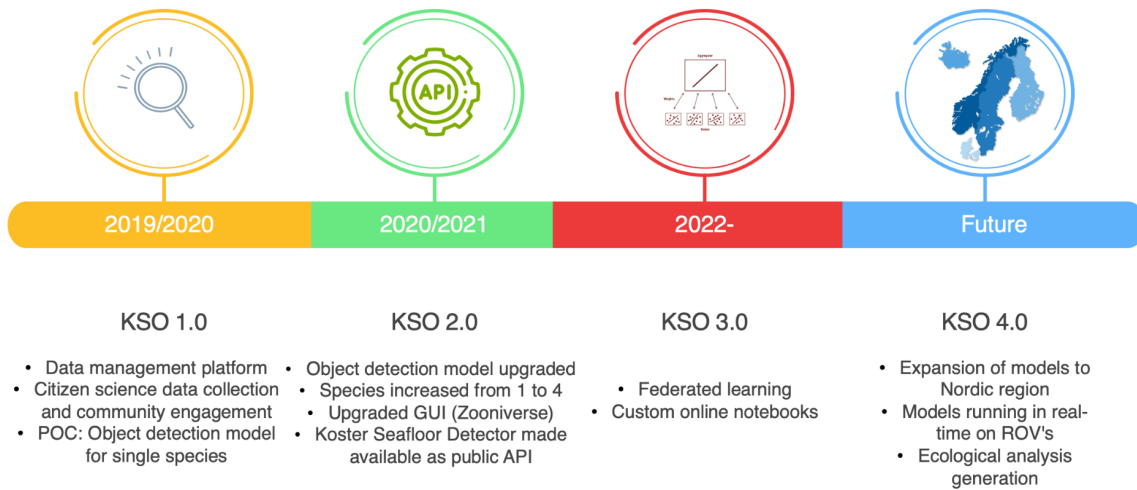


Figure 9. Roadmap for developing KSO as a Nordic resource for underwater image analysis

3.3 Nordic Digital Humanities Laboratory data analytics on UCloud

3.3.1 Background

With this use case we wanted to demonstrate how researchers from social sciences and humanities at Nordic universities can collaborate across borders in a secure and interactive virtual research environment with the best possible resources to run a job. Specifically, representatives from Nordic Digital Humanities Laboratory (NDHL) collaborated on monitoring news and social media during the first phase of COVID-19 (Jan 1 - Jun 30 2020), see Nielbo et al 2021³⁷ and Figure 10. Importantly, all data came with restrictions on storage location and access due to copyrighted and sensitive content. Finally, the demonstrator provides a context for showcasing the UCloud platform and its development.

³⁷ Nielbo, K.L., Hastrup, F. and Enevoldsen, K.C. and Vahlstrup, P.B. and Baglini, R.B. and Roepstorff, A. When no news is bad news - Detection of negative events from news media content, arXiv:2102.06505 [cs.CY] 2021 DOI: [10.48550/arXiv.2102.06505](https://doi.org/10.48550/arXiv.2102.06505)

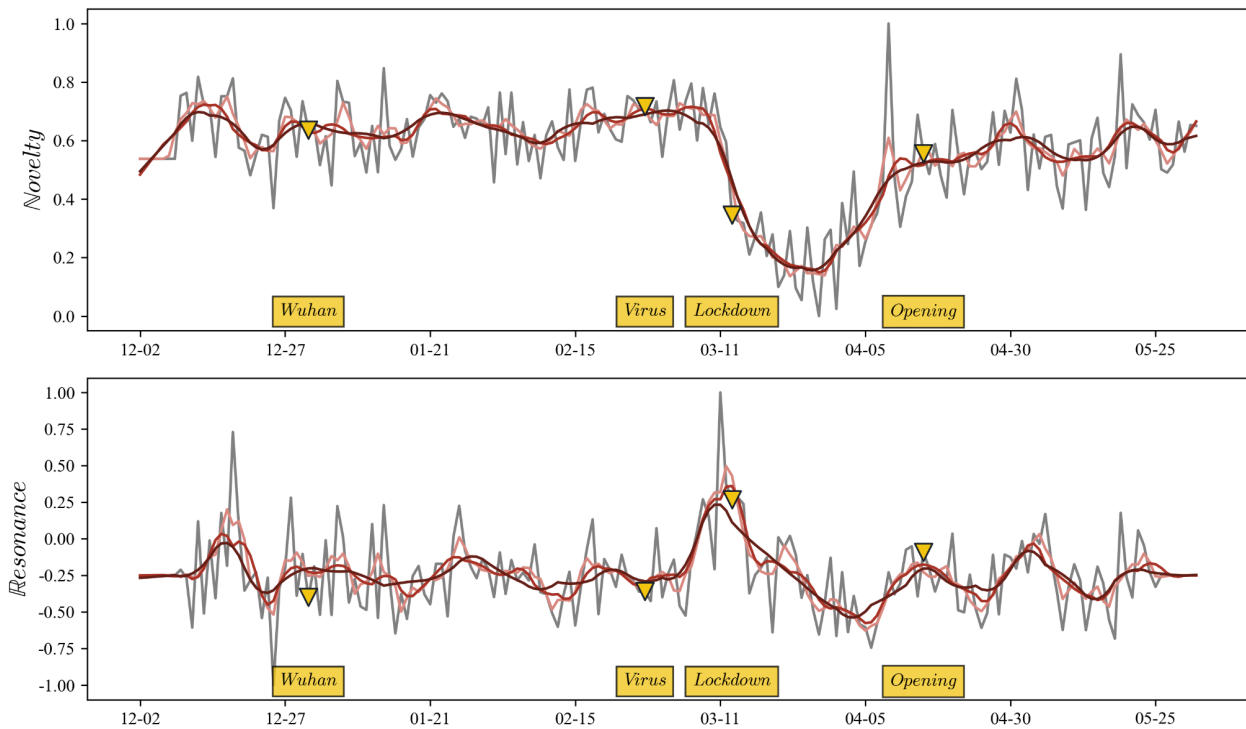


Figure 10. News monitoring (broadsheet newspapers) during the first phase of COVID-19 in Denmark. The monitoring system uses a multilingual Scandinavian language model to derive structured representation of front pages, then compute surprise (upper) and persistence (lower) between representations, and run a change detection algorithm over the signals.

3.3.2 Objective

The use case has three objectives: 1) sharing of access to compute and restricted data resources (copyrighted Danish newspapers) in order to monitor media response to COVID-19 in the Nordic countries; 2) research collaboration in a low barrier to entry interactive computing environment; 3) showcase the UCloud platform for research collaboration in the social sciences and humanities.

3.3.3 Workflows

NDHL ingests media source data from news websites and the Informedia database and aggregates the data in the HOPE database, see Figure 11. Data are then normalized using a state of the art language model DaCy in order to extract nouns and adjectives (lemmatized forms). Data extraction and normalization are only available to the host (in this case DK) due to restrictions on the source data. In the final step, representations of front pages are inferred using a domain-specific model for news media (trained on 10 years of news media data), surprise and persistence signals are extracted, and a change detection model applied to find robust structural changes, see Figure 10. The implementation and deployment of the use case has been based on UCloud instances.

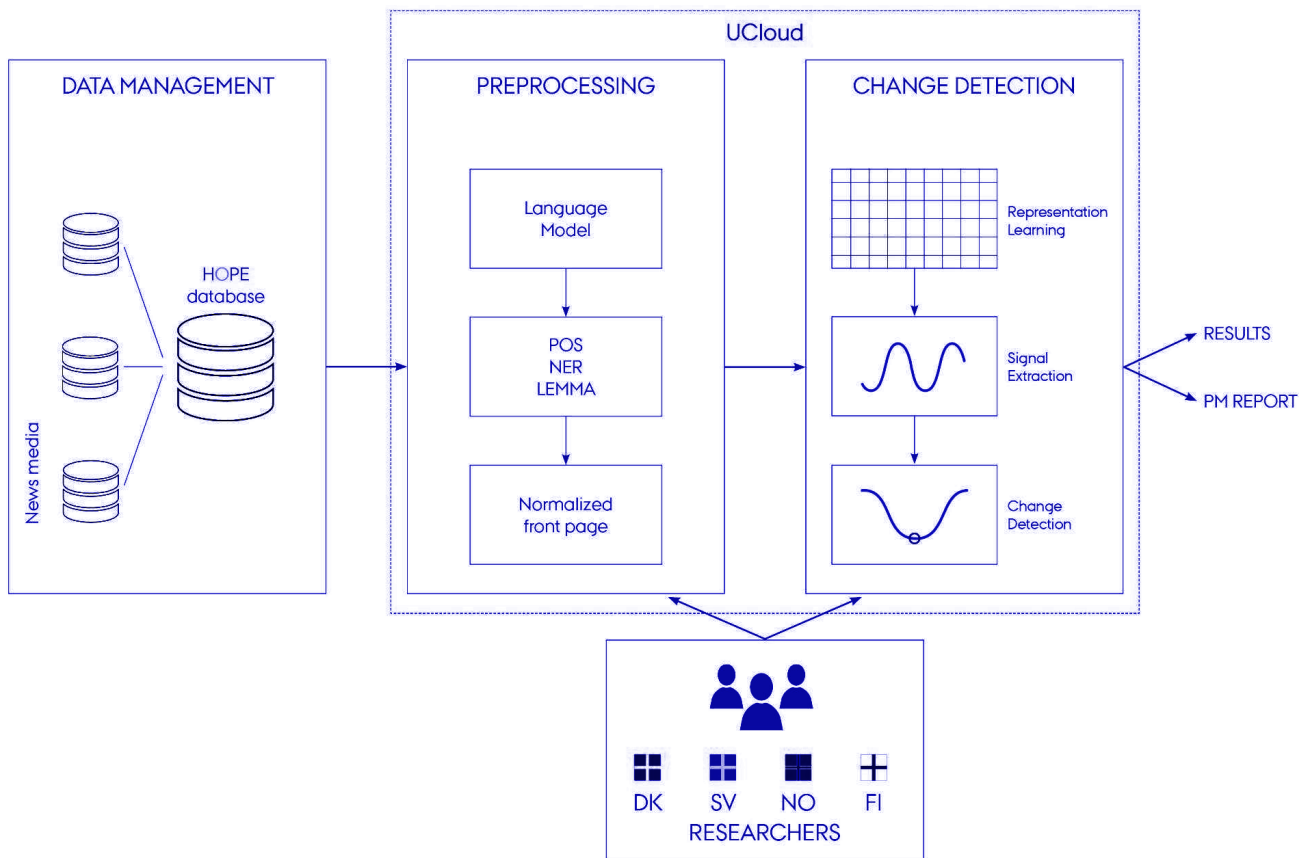


Figure 11. Demonstrator workflow for NDHL’s media monitoring during the first phase of COVID-19.

3.3.4 Technical Overview

UCloud is designed to be user-friendly with an intuitive graphical user interface, flexible and extensible to account for the multi-scale and multi-disciplinary research challenges, and the high data intensity and heterogeneity. The focus of UCloud is to make complex digital technology accessible to all users.

In a sense, UCloud acts as an orchestrator of resources. Allowing users to consume resources, such as compute and storage, from multiple different providers using the same interface. This allows for seamless experience when consuming resources from different providers, allowing researchers to focus on their work as opposed to the specifics of any given provider.

UCloud ships with a powerful system for managing projects and applying for resources. Requesting resources for use through UCloud is an integrated process and all occurs inside UCloud. A potential principal investigator (PI) can initiate the application process through the interface. The application process is illustrated in Figure 12.

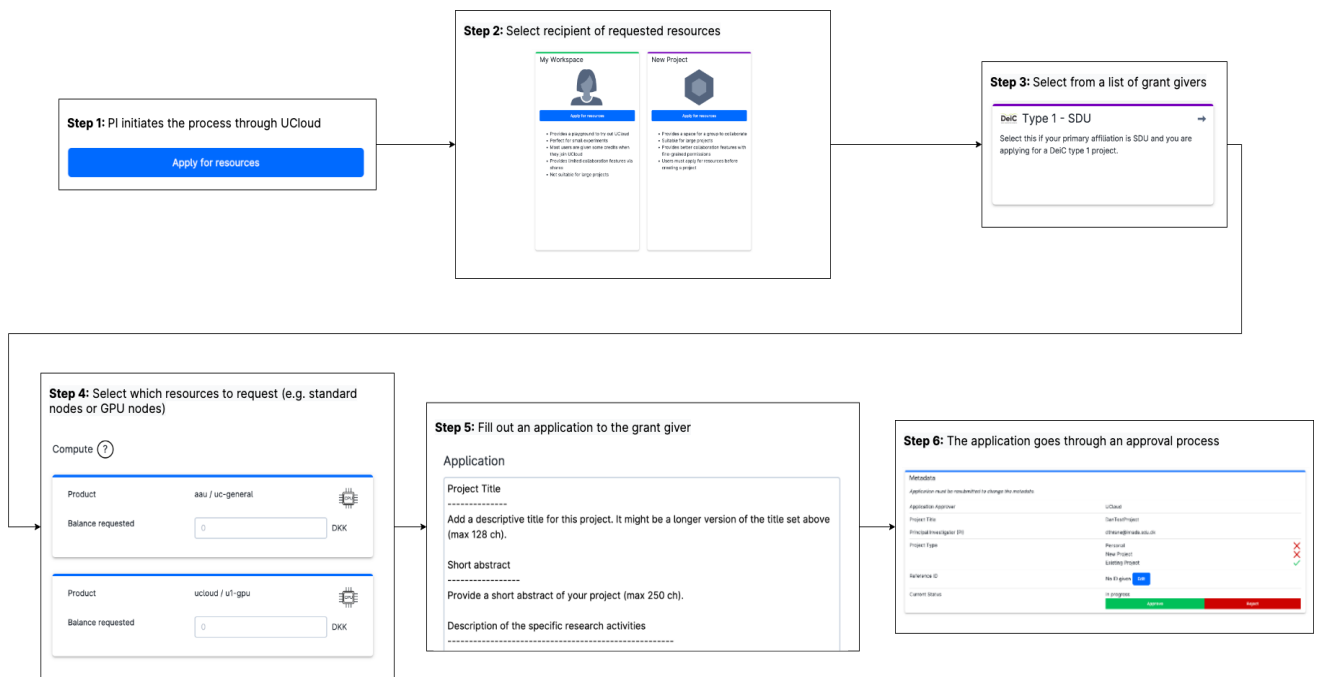


Figure 12. A graphical illustration of the allocation process in UCloud.

The allocation process goes through the following steps:

1. You first choose to create an allocation
2. Then you select the intended recipient of the resources
3. You can then select from a list of grant givers
4. And fill out which resources are required
5. You must then fill out an application, describing why you need the resources
6. The approval process begins and the grant giver ultimately decides whether to approve or reject the application

A project is created as a result of a successful grant application. Projects allow for collaboration between different users across the entire UCloud platform. A project has one or more members and contains zero or more groups. Every member of a project has a role (see Table 4), this role is used to dictate which actions they can perform within the project.

Role	Notes
PI	The primary point of contact for projects. All projects have exactly one PI.
Admin	Administrators have the same privileges as a PI, but they are not considered the primary point-of-contact. A project can have zero or more admins.
User	Has no special privileges

Table 4. The possible roles of a project member of UCloud. All members have exactly one role in the project.

Groups are used to subdivide members into smaller sections (see Figure 13). This subdivision is used for permission management of the resources that are consumed within UCloud. For example, a PI can use this feature to grant read-only access to some data for some members of their team, while granting read-write access to others.

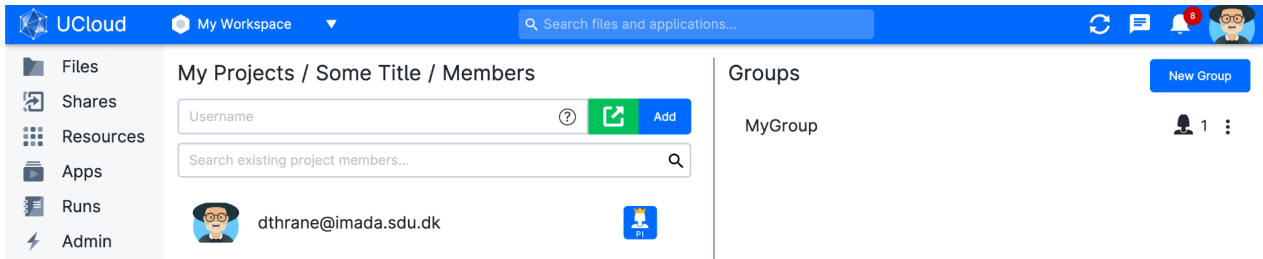


Figure 13. Project management interface of UCloud. In this interface an admin or PI of a project can invite new members into the project. The members that have accepted the invite can then further be arranged into groups.

UCloud’s core job is to act as an orchestrator of resources. The resources supported fall roughly into either storage or compute. It is the job of UCloud to specify a set of core rules for accessing and manipulating these resources. The implementation of the resources is backed by well-tested established components, such as Kubernetes and POSIX file systems.

The file-system abstraction of UCloud provides researchers with a way of storing large data-sets efficiently and securely (see Figure 14).

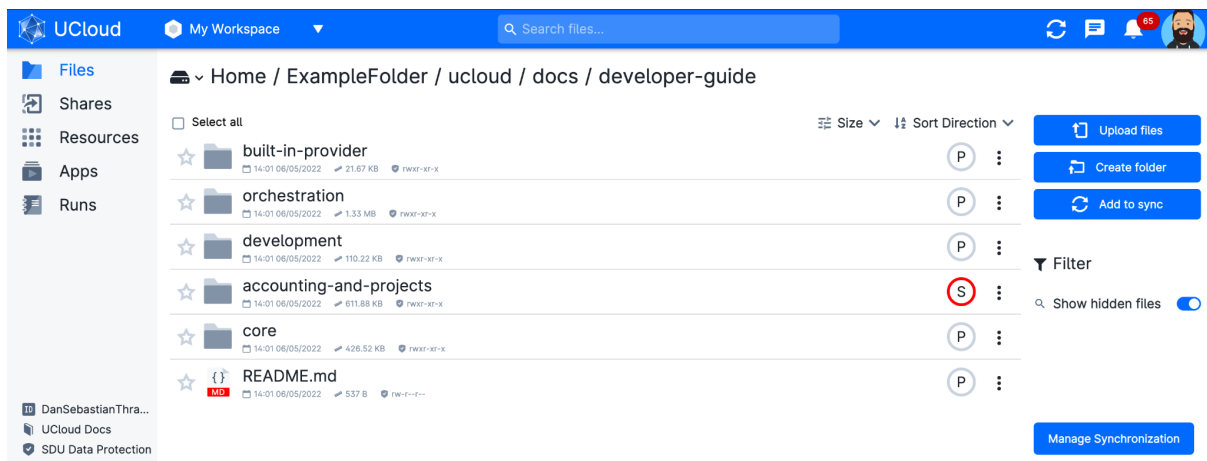


Figure 14. The file management interface of UCloud. It highlights some of the features included in UCloud. For example, we see that each file has a sensitivity classification, indicated by an “S”, which means that the folder contains sensitive information.

The file-system of UCloud interfaces with project management, and accounting to provide a fully featured experience. For example, UCloud supports:

- All interactions with the file-system are automatically audited
- Share files with collaborators with authorization based on projects and groups
- Powerful file metadata system for data management

The computational system of UCloud provides researchers a way of performing computations on their datasets. Researchers begin by selecting an application from a large catalog of scientific applications. A small selection of applications can be seen in Figure 15.

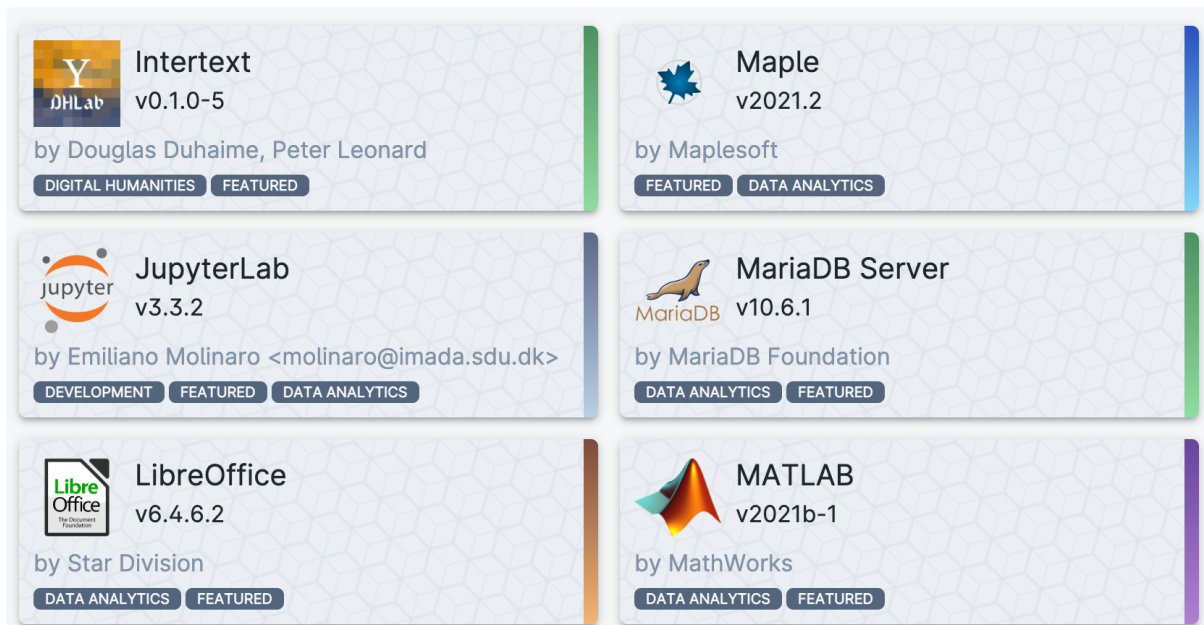


Figure 15. A small subset of applications available on UCloud.

UCloud hosts a variety of different applications designed for different styles of scientific computing:

- Batch applications provide support for long running computation workloads. This type of workload is typically highly parallelized using technologies such as MPI.
- Interactive web applications provide support for graphical and interactive applications. This type of application is well-suited for developing new code. For example, JupyterLab is exposed via its web interface.
- Virtual desktop environments provide many of the same benefits of a web application, but with a full desktop attached.
- Virtual machines support advanced workloads which have special requirements not covered by the other types of applications. This allows users to, for example, create their own Kubernetes cluster.

UCloud empowers cross-border and cross-organization collaboration. It does so by providing a technical framework for integrating compute and storage to a central platform. These service providers (SPs) are considered independent, and UCloud exposes few limitations on how they are run. As a result, the

framework enables both cross-border and cross-organization collaboration. The technical framework is exposed to SPs through a provider API, which is covered in Section 2.3.5.

Throughout the UCloud project’s life, several providers have integrated with the UCloud platform. This is illustrated in Figure 15. The integrations cover platforms of many different types. This includes traditional cloud technologies, such as Kubernetes and OpenStack. Yet, it also covers more traditional HPC technologies such as Slurm. The SPs all integrate through a provider API, but the API itself is implemented using software designed for different types of SPs. The software itself is generic and can be deployed at multiple sites, as long as the underlying technology stack is similar.

For the NDHL demonstrator, a proof-of-concept SP was created and integrated with the central UCloud service platform. This cluster was installed and physically located at CSC in Finland and it integrated with the central UCloud service platform which is physically located at the University of Southern Denmark. The CSC provider ran a small Kubernetes cluster and integrated with UCloud using the Kubernetes integration. This is the same Kubernetes integration which runs at the YouGene Cluster, see Figure 16.

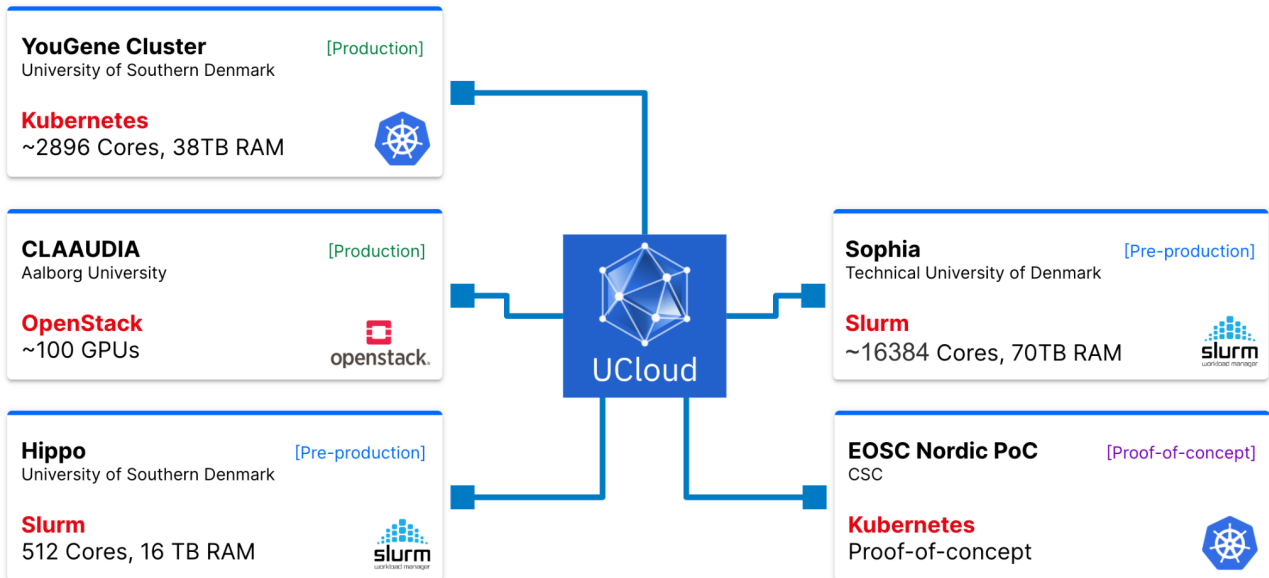


Figure 16. An illustration showing a range of service providers integrating with the UCloud service platform. The providers are located and hosted by different organizations spanning multiple countries. The service providers integrate using different technology stacks, such as Kubernetes. The [Production], [Pre-production] and [Proof-of-concept] labels all refer to the integration with UCloud and not the operating status of the cluster itself.

3.3.5 Technical Solution

UCloud is an orchestrator of resources. It is primarily responsible for orchestrating different SPs to do the work as requested by the end-user. To facilitate this work, UCloud has a provider API. This API specifies both the low-level technical of how the SP and orchestrator must communicate but it also covers the high-level rules and concepts which are used when communicating to the end-user. Figure 17 illustrates some of the work going into an ordinary request from the end-user. There are a few pieces of information worth highlighting.

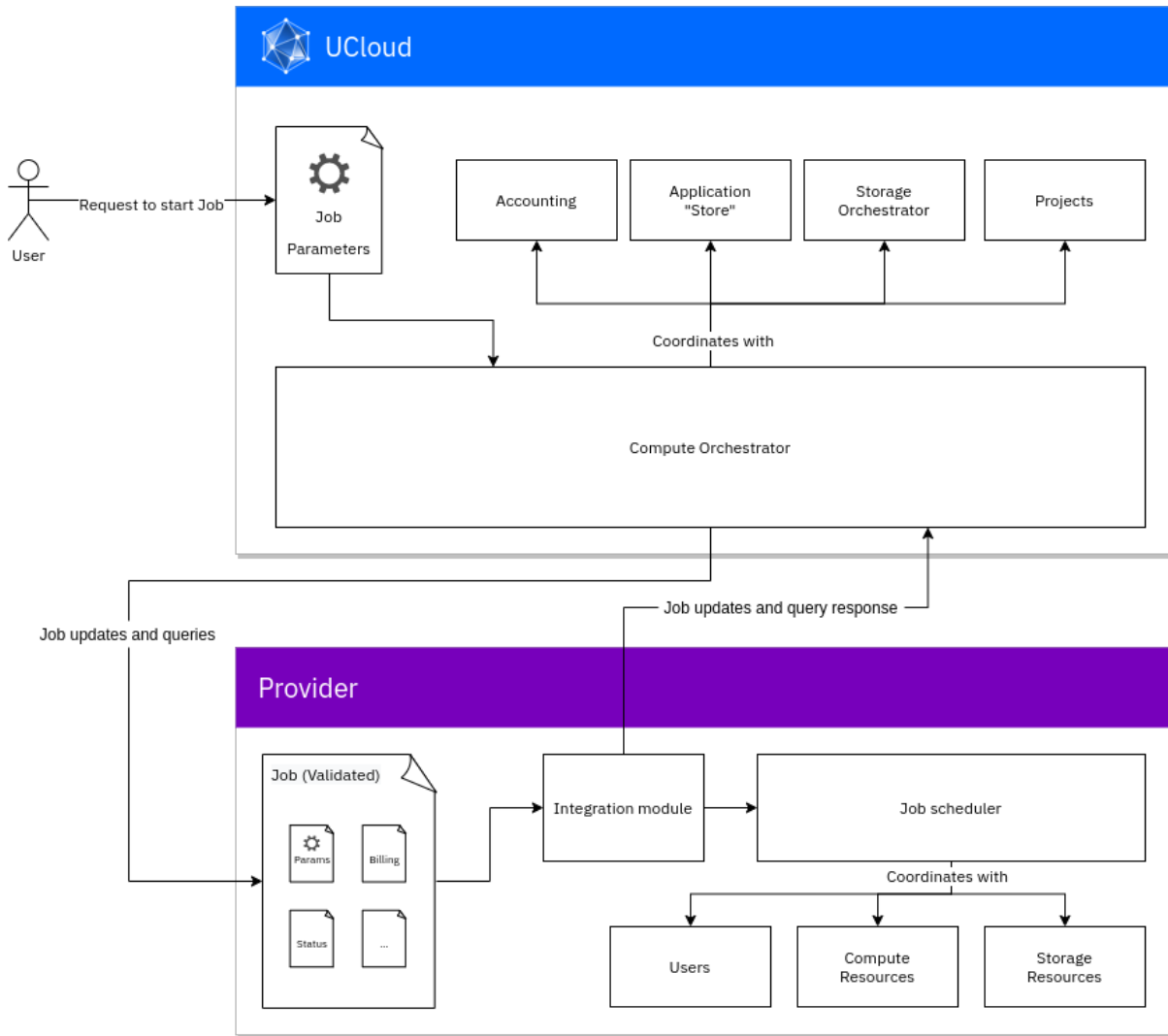


Figure 17. UCloud is an orchestrator of resources. This diagram illustrates how the core of UCloud communicates with a provider to facilitate a computational workflow. The key part of this is the “integration module” which is responsible for communicating with UCloud and coordinating with existing infrastructure. The integration module communicates with UCloud through a well-defined provider API.

The figure exemplifies this, but it is generally the case that the end-user communicates directly with the UCloud service platform and not the SP. This allows for UCloud to perform secondary tasks, such as: authentication, authorization, auditing and accounting. These tasks are of high importance but are typically complicated to implement and usually do not provide a lot of business value to the end-user. This does not remove all responsibility from the SPs but, generally speaking, this makes the task easier. Effectively, allowing the SPs to better focus their energy on delivering a good service to the end-user.

Once UCloud has performed these secondary tasks, the SPs responsible for a given resource (in this case a computational job) are contacted regarding the request. The SP internally acts on this request and then proceeds to update UCloud about any changes with the resource. For example, the SP might notify UCloud about changes in job state or accounting usage.

The provider API is illustrated in Figure 17 as the arrows moving bidirectionally between UCloud and the provider. In practice, this API is implemented by a piece of software which we simply refer to as an “integration”. Figure 16 illustrates these different integrations installed in various SPs.

One such integration is the Kubernetes integration. This integration currently runs in production on the YouGene cluster and was tested as part of the NDHL demonstrator on a cluster installed at CSC. This integration delivers a service which is deeply integrated with UCloud and tailored primarily towards interactive high performance computing. It focuses on workloads which are easy to containerize, such as JupyterLab. It is designed to be coupled with a sibling integration delivering storage based on any distributed POSIX filesystem. This allows users to easily spin up an analysis workflow, web service or batch processing on demand. Being based on containers, end-users don’t have to focus on any management or configuration of their workloads. They simply spin up a workload when they are needed and shut it down when they do not need it. Applications are periodically updated on UCloud, thus an end-user can always pick an up-to-date version.

The OpenStack integration delivers persistent virtual machines through UCloud. Virtual machines are particularly useful when a workload is long running, requires persistent data or is otherwise unsuitable for containerization. This integration gives end-users a much higher degree of freedom when it comes to managing the workload’s environment. This freedom also comes with an increased responsibility for the end-user, as they now need to manage software updates and configuration manually.

The integration module and along with it the Slurm integration, delivers an integration tailored for HPC providers. Both end-users and SPs gain a lot from such an integration between HPC systems and UCloud. All end-users get to take advantage of the many years of development which has gone into HPC software and hardware. This allows users to squeeze out more performance for their workloads when it matters. Unfortunately, most HPC systems can be quite daunting to use for beginners. UCloud can help in this respect, by providing a user experience more similar to the ones they would get from their own laptop’s operating system, while still achieving the performance of an HPC system. Finally, SPs and power users gain benefits from UCloud’s handling of tasks which are secondary to the actual compute and storage. Amongst other things, the UCloud service platform improves the grant application process, project management and accounting. This makes system management easier for SP owners. For researchers, this means spending less time on tasks which are secondary to their research.

3.3.6 Possibilities

UCloud today delivers a solution which enables cross-organization and cross-border collaboration for compute and storage. It already has in-production and pre-productions services which span multiple organizations and uses dramatically different technology stacks. All of this is delivered in a platform which helps both SPs and end-users deal with the tasks that are not directly related to compute and storage. For example, this includes tasks such as: authentication, authorization, auditing, accounting and project management.

In the future, we believe it is important to focus on a number of areas. UCloud must continue to expand its suite of integrations to better cover the needs of SPs. For example, this can include support for different compute schedulers and storage technologies. It must also continue to integrate more SPs into the service

platform and make the required adjustments to facilitate this. Finally, with more SPs integrated into UCloud, it becomes crucial to focus on functionality which crosses multiple SPs. One important example of this is facilitating the movement of data between SPs.

3.3.7 Challenges, possible blockers

With projects, such as UCloud, come a lot of challenges, for example related to trust and security. Cyber-security is a complex and moving target. But, we will attempt to cover some risks and the principles behind mitigating these risks.

As already mentioned in Section 3.3.5, UCloud communicates with SPs using a (web) API. Due to the distributed nature of UCloud, this ends up having to use public networks for communication. Because of this, UCloud employs best practices for communication across such networks, in particular all traffic is encrypted via TLS. UCloud was designed based on principles from Zero Trust architecture³⁸. As part of the message protocol, UCloud authenticates and authorizes all messages. There is no implicit trust between an SP and UCloud based on, for example, network addresses. Furthermore, the API avoids proxying information, when not strictly needed. We mention in Section 3.3.5 that the end-user almost always communicates with UCloud directly. This is true in most cases, but not always. In some cases, such as upload of data, UCloud only facilitates a session between the end-user and the SP. This way, UCloud never sees any of the files stored on any of the providers. This is true even when a user uploads files from within the UCloud interface.

UCloud always authenticates itself with an SP. Yet, this only proves to an SP that UCloud is able to authenticate itself. It does not prove, for example, that any backing user requests are actually legitimate. Assuming that UCloud is acting in good faith, this is not a problem. But it does introduce some trust in UCloud that is not acting malicious. To mitigate this, UCloud provides a mechanism for SPs to authenticate that a message is from an end-user. This mechanism uses public key cryptography with keys exchanged between end-user and SP directly. This works by having the end-user digitally sign all messages they send to UCloud. UCloud, having never seen any of the keys, has no way of forging or validating a signature. Thus, UCloud will forward the signature as it receives it. This gives the SP a tool to validate messages from UCloud which target a specific user. The SP, using the public key, must verify that the message received from UCloud matches the end-user's intention.

In a similar fashion, the SP always authenticates itself with UCloud. But, UCloud cannot be certain that the SP is not acting malicious. Because of this, SPs are only allowed to query and update information relative to their own SP. This enforces the principle of least privilege. Furthermore, UCloud keeps audit logs for all API requests and backups of the backend data, which can aid in recovering in case an SP is compromised.

4. Benefits and lessons learned

Through the course of the subtask, we have had the chance to have dialogues with several research communities, close collaboration between sites, test and push the capabilities of existing technologies and

³⁸ <https://www.nist.gov/publications/zero-trust-architecture>

cloud infrastructure implementations from which scientists did benefit by getting an easy access to e-Science resources. The lessons learned and recommendations for the future that can be drawn from this experience are as follows:

1. Trial-and-error experiments are still necessary for development of machine learning models and thus still require significant human effort and computational resources.
2. Communication between experts from different domains is a known problem and needs to be better facilitated in the future in order to achieve a common understanding. Good dialog between experts and stakeholders must be at the core of activities to ensure a successful outcome.
3. It has become clear that despite cloud infrastructures providing an easy way for end-users to work and focus their effort on the research projects, not all research communities can benefit equally from it. It is important to consider having a technical solution which not only aims for providing web-based solutions for computing capabilities, but also combines or is complementary to more traditional ways of processing and consuming data, such as command line interfaces.
4. Accounting of cloud resource utilization is a fairly difficult topic, and it proved to be a challenging topic in relation to federating resources across different SPs and RIs, since they all have different ways of handling accounts, projects and resource allocations. A full common solution would require further work in order to harmonize or at least find common grounds on a standard for cloud resource allocations. Work in this direction exists, which addresses, at least in part, the problem. In the Nordic region, a noteworthy effort is the NeIC Puhuri project³⁹. Puhuri aims to provide a common scalable digital authorization platform for accessing HPC resources. It is currently used by the ten LUMI consortium countries to offer access to the LUMI supercomputer. Puhuri integrates with MyAccessID, a service from Geant, as a common platform for identity and access management. Much work in the Puhuri project has been dedicated to harmonizing resource and project definitions among the participating members in the Nordic region. Puhuri now provides a common platform where national allocators can manage projects and resources at SPs and where they can obtain information on utilization of resources. Although the initial goal for Puhuri was HPC and, in particular, the LUMI supercomputer, the platform is capable of handling resources specific for cloud computing as well.
5. The cloud solutions were designed to fulfill requirements shaped by various stakeholders. The different code bases, architecture designs and technical implementations are based on different sets of use cases identified by each project. There are certainly common use cases and functionalities provided by each solution, however, with different technical approaches and solutions. Intrinsically, the knowledge base and the competence pool at each SP and RI have also influenced the respective technical implementations. Adopted licensing models for each of the solutions are meant to reflect requirements and technical choices. Harmonizing these might be a difficult exercise.
6. Security is in general another difficult topic. Challenges exist due to lack of trust and agreements between different service providers and institutions. This is even more difficult when the providers are in different countries with perhaps different regulations and requirements for security and data protection. Legal challenges for the use of sensitive data across borders are a long standing problem. Even if we just restrict the problem to the Nordic region, the challenges remain very acute. This task did not focus on these legal challenges, but we note that other tasks of EOSC-Nordic cover that part. At the more technical level, two frameworks have been used in this task. In regards

³⁹ See <https://neic.no/puhuri/>

to identity management and authentication, AAI solutions like the previously mentioned MyAccessID and Puhuri, offer different Levels of Assurance (LoA) to SPs. While the implementation of strict guarantees on LoA is not without challenges, once in place, LoA will help SPs to have a strict control on user identities. The second technical approach showcased in this WP is the use of Zero Trust approach (see UCloud demonstrator). Zero Trust frameworks by definition remove or greatly reduce the trust required between the different actors and, while not completely new, are still not broadly used, as they require complex solutions to be implemented.

7. While certainly there are still technical challenges to be addressed and resolved, there are clear stoppers and hindrances introduced by regulations, laws and policies. As earlier mentioned, it is important that technical and legal activities are synchronized and progress together. WP2 is in fact coordinating efforts between national initiatives and contributing to harmonizing policies and investing effort in removing legal issues in Nordics in the context of EOSC. WP2 is amongst others, looking into policies and resource provision, as well as cross-border collaboration models, which are addressed in Deliverable D2.5 Open Science policies and resource provisioning in the Nordic and Baltic countries (second report).⁴⁰
8. Finally, while work and collaboration, as a consequence of the COVID-19 pandemic, was very much limited for the most of the time to remote video meetings, webinars and workshops, it has proven to be efficient and optimal for cross-border collaboration. However, the physical meetings, face to face dialogues cannot be substituted for but rather complemented, video meetings being a perfect addition to periodic physical meetings.

5. Conclusions

Important initiatives and solutions have emerged within EOSC-Nordic and have proven to be groundbreaking and to the benefit of research and collaboration.

The cross-border computing used by the climate modeling community is based on the user-friendly Galaxy portal giving easy access to cloud computing resources and bare-metal HPC clusters. Part of this has been developed earlier in this subtask and the details are therefore already described in Deliverable D5.2 Cross-borders computing through portals⁴¹.

The marine biodiversity exploration using cross-border resources and federated machine learning, i.e. the FEDn-KSO use case, showcased the capabilities of cross-border cloud computing, exemplifying possibilities for deploying the same software across multiple cloud architectures. It also demonstrates how different technical challenges related to transfer and movement of large amounts of datasets can be addressed.

We can also conclude that a better way for providing ML frameworks, like FEDn, is to provide applications in the catalog of the cloud solution provider, e.g., by registering it as an application in NIRD Toolkit instead of deploying it as a service on the NIRD SP. For this, one should use Helm charts extensively, and offer parametrized applications, i.e., one application which can instantiate the FEDn components. This approach would fit better and give more flexibility to the end-users and offer self-service capabilities.

⁴⁰ Hammargren, P.-O.; Arvola, M.; Rauste, P.: D2.5 Open Science policies and resource provisioning in the Nordic and Baltic countries (second report), Deliverable, EOSC-Nordic 2021, DOI: [10.5281/zenodo.5537068](https://doi.org/10.5281/zenodo.5537068)

⁴¹ Abarenkov, K.; Fouilloux, A.: D5.2 Cross-borders computing through portals, Deliverable, EOSC-Nordic 2021, DOI: [10.5281/zenodo.4607199](https://doi.org/10.5281/zenodo.4607199)

The Nordic Digital Humanities Laboratory data analytics on UCloud use case demonstrates the ability of UCloud to deliver a solution which enables cross-organization and cross-border collaboration for compute and storage. With UCloud one can accomplish connecting different cloud infrastructures, allowing researchers to set up projects and workflows and deploy applications within the same namespace. While orchestration of jobs and computing resources is at the core of UCloud, moving data where the compute capacity is, requires further prototyping and work.

Similarly to UCloud, great results have been achieved on the NIRD Service Platform and NIRD Toolkit, supporting multiple use cases, like STACKn and FEDn services being deployed on the NIRD SP. Also, predefined applications in the NIRD Toolkit, e.g., Jupyter Notebook, allows Norwegian and Nordic users to deploy applications, start AI/ML workflows with a few clicks, giving them the opportunity to focus on what matters most, research.

While a common Nordic cloud toolbox was not deemed to be practical solution given the different code bases as well as licenses for the different cloud solutions, the collaboration certainly revealed possibilities for collaboration and possibilities for setting up a shared application repository, based on for example Helm charts, making applications, frameworks shareable across projects and borders.

Accounting of cloud resource usage was addressed by each service provider and adjusted for the particular solution in each case. Accounting evolved in parallel with the solution but never reached a satisfactory state as it is a complex field, being very much dependent on the resource allocation model and technical solution for the cloud infrastructure. As a consequence, prototyping an accounting solution which would work across borders was not yet possible to achieve and would require further effort.