Deliverable D3.1

Unified Catalogue Format Description

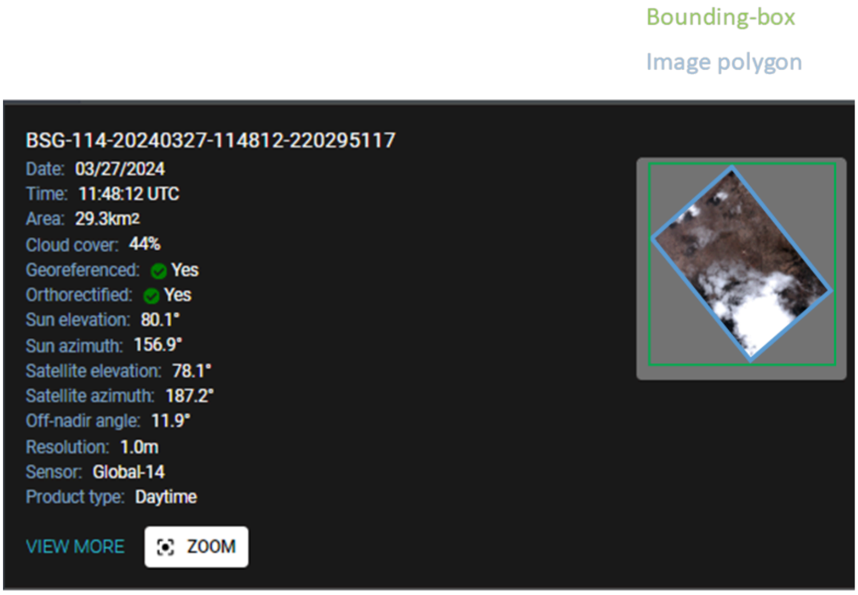
Deliverable Name	Unified Catalogue Format Description
Deliverable Number	D8
Deliverable Related Number	D3.1
Work package	WP3
Deliverable Leader	ECMWF
Date of Delivery	31 July 2024 (M06)
Deliverable Type	Report
Dissemination level	Public

Version	**Author**	**Date**	**Comment**
1.0	Simon Smart *et al.*	17 July 2024	First issue

Table of Contents

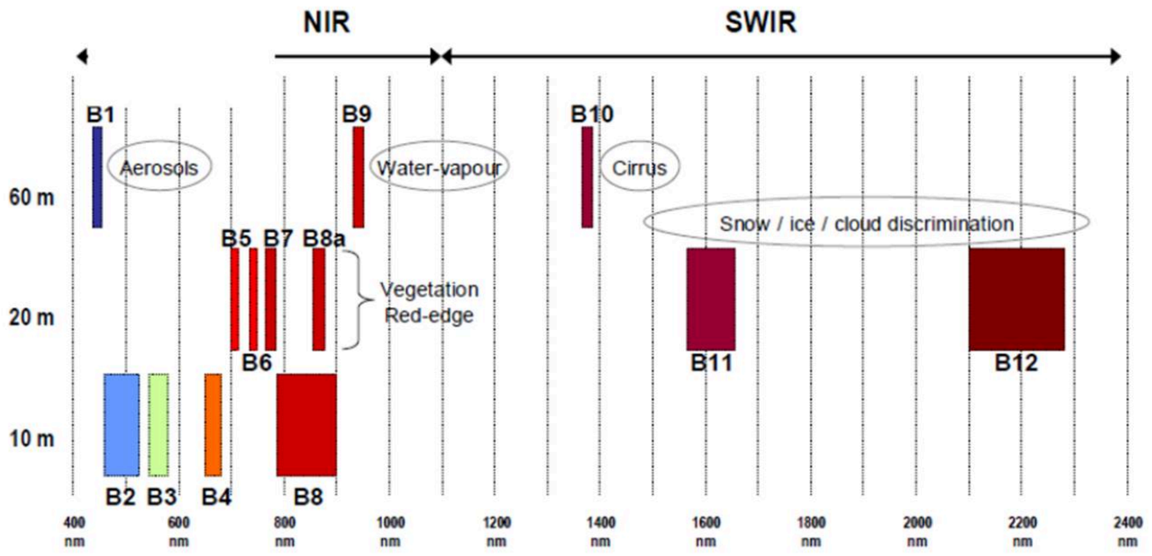
1. Terminology

- **AI**, stands for Artificial Intelligence.
- **ANN** (Artificial neural network), a computational model that mimics the way nerve cells work in the human brain.
- **AoI**, Area-of-Interest
- **Bounding Box:** A square definition of (Longitude1, Latitude1, Longitude2, Latitude2) to define an earth observation image boundaries on a CRS



- **Copernicus**, Copernicus is the Earth Observation component of the European Union's space program, looking at our planet and its environment for the benefit of Europe's citizens.
- **CPU,** central processing unit refers to the general processing unit without constraint on the instruction set (x86, ARM)
- **CRS:** Coordinates Referential System
- **DL** (Deep Learning), the subset of ML methods based on ANNs.
- **EO,** stands for Earth Observation.
- **External metadata,** metadata which refers to the object embedding data. For instance a file size.
- **GPU,** Graphical Processing Unit, data parallel architecture initially used for computer vision and graphical processing, now the workhorse of most of AI workload. Promoted by NVIDIA, AMD and Intel.
- **Hierarchical schema,** a scheme corresponds to the structure of an ontologies, where metadata are organized by categories. For instance, internal metadata, and external metadata.

- **Image segmentation**, a process of partitioning a digital image into multiple image segments, also known as image regions or image objects (sets of pixels).
- **Internal metadata**, or **Semantic metadata,** metadata which refers to the content of the data. For instance, the maximum value of parameters fields which the meteorological data file.
- **IPU,** Intelligence Processing Unit, novel architecture dedicated to AI processing and promoted by Graphcore, a UK based startup**.**
- **Metadata catalog,** a database holding descriptive information about data objects which are themselves stored remotely. An example being a catalog of geospatial coordinates, each associated to a satellite image stored in a remote datahub. Metadata catalogs are tools used for querying and dataset building.
- **ML** (Machine Learning), a field of study in AI concerned with the development and study of statistical algorithms that can learn from data and generalize to unseen data, and thus perform tasks without explicit instructions.
- **MSI:** Multi-Spectral Instrument embedded on Sentinel2 earth observation satellites. MSI spectral-bands versus spatial-resolution is provided in the next figure.



- **Neuromorphic Architecture**, an approach to computing that is inspired by the structure and function of the human brain. There is no per-se Neuromorphic processor generally available on the market but it remains an intense domain of research.
- **Non-Von Neumann Architecture**, any computing architecture, such as dataflow systems, that does not follow the fetch-decode-execute cycle of traditional CPUs.

- **Ontologies,** formal explicit description of concepts in a domain, properties of each concept describing various features and attributes of the concept, and restrictions. An ontology together with a set of individual instances of classes constitutes a knowledge base.
- **OSM,** OpenStreetopen collaboration. Map is a free, open geographic database updated and maintained by a community of volunteers via open collaboration.
- **Semantic Web**, sometimes referred to as Web 3.0, is an extension of the WWW standard. To enable the encoding of semantics with the data, technologies are used to formally represent metadata and can describe ontology, relationship between entities,and categories of things. These embedded semantics offer significant advantages such as reasoning over data and operating with heterogeneous data sources.
- **Sentinel-2**, Copernicus Sentinel-2 mission is based on a constellation of two identical satellites in the same orbit. Each satellite carries an innovative wide swath high-resolution multispectral imager with 13 spectral bands for a new perspective of our land and vegetation.
- **Sentinel2 L0 level:** Compressed raw image data acquired by satellite Multi Spectral Instruments (MSI) and specifically packaged. This product is not provided to users.
- **Sentinel2 L1A level:** Uncompressed raw image data with spectral bands coarsely coregistered and appended auxiliary data. This product is not provided to users.
- **Sentinel2 L1B level:** Radiometrically corrected multi spectral image data with spectral bands coarsely co-registered, without refined geometric model applied.
- **Sentinel2 L1C level:** Ortho-rectified and UTM geocoded top-of-atmosphere reflectance with sub-pixel multispectral and multi-date registration. Covers 100 km x 100 km, accessible to all users.
- **Sentinel2 L2A level:** Ortho-rectified and UTM geocoded bottom-of-atmosphere reflectance. Additional outputs are Aerosol Optical Thickness map, Water Vapour map, Scene Classification map and Quality Indicators data. Covers 100 km x 100 km, accessible to all users.
- **SNN** (Spiking neural networks), an ANN that more closely mimics natural neural networks (human brain).
- **ToI**, Time-of-Interest

2. Executive Summary

The primary goal of WP3 is to develop an advanced metadata management system specifically for Copernicus data, resulting in a functional metadata catalogue. This system will be built by integrating a number of components provided by different project partners. Metadata will serve as the common language of interoperability between these components and the end users. Consequently, the system encompasses elements for generating, storing, and updating metadata, and ensuring it is readily accessible to users.

From the user perspective, at the core of this system is a metadata catalogue which presents a unified and accessible view of a range of metadata, integrating intrinsic metadata describing the available data with external knowledge and metadata which is generated by analysis of features of interest within the data. This is known as the Unified Metadata Catalogue.

The purpose of D3.1 is to present an overall architectural description of the system including the Unified Metadata Catalogue. This is enhanced with a description of the types of metadata available and potentially generated for the Copernicus Earth Observation data within the project, along with an outline of how this metadata will be used to operate the unified system.

3. Introduction

The DaFab project aims to address several challenges associated with the use of the vast amounts of available Earth Observation (EO) data. It is often challenging for a user to identify what data is available, locate the specific data they need, and access and use it efficiently.

The data available, including those from Copernicus, are not just large and numerous but comprise many distinct datasets - each of which is different, differently labelled, and located across many different organisation's systems. This diversity complicates the identification of the best datasets for specific purposes and necessitates a patchwork approach, making data access and use a significant effort.

To address these challenges, DaFab will develop a system to facilitate the use of public EO datasets. DaFab proposes solutions that enable timely analysis of diverse and siloed data within a sustainable architecture composed of mature and reusable components. Metadata serves as the glue that binds these elements together - by abstracting away technical details about data locations and allowing data querying and handling based on metadata, we make EO data accessible to a broader audience. Metadata thus becomes the language of interoperability, bridging the gap between technical system components and the end user.

Advanced AI techniques will be employed to extract semantic features (metadata) that describe the contents of the imagery. These techniques will be integrated with systems designed to optimise and interlink the generated metadata, making it easier to identify EO data of interest. The project will then develop a unified and searchable catalogue which will expose this data and metadata to a range of users.

Metadata access is meaningless without a broader infrastructure to support its use. In this deliverable we present our proposed architecture for the DaFab project. This includes not only the pipeline for obtaining EO data, performing semantic feature extraction, and storing and accessing EO data, but also the metadata catalogue and infrastructure to make these datasets available in various contexts.

This system will be assembled by connecting and aligning components from the different DaFab partners, incorporating both existing systems and new developments. More specifically, SKIM, developed by TAS, will use technologies to enrich EO data with information from other sources, to enable automated decision-making. For that, SKIM will be extended to support a generic interface capable of generating secondary metadata and interlinking it with primary EO metadata, web knowledge, and scientific knowledge. RUCIO, developed by CERN, will provide a distributed scientific data management system, and a publicly queryable metadata catalogue interface. DASI, developed by ECMWF, will provide in-HPC and in-Cloud access to data within workflows driven by domain-specific metadata. For that, DASI will be enhanced with backends that enable access to data exposed by RUCIO alongside locally generated earth-systems data.

To demonstrate the use of this system we will consider two use cases that highlight the downstream scientific application of Copernicus data. The first use case focuses on the identification and feature extraction of anomalous water and flood conditions. The system built will facilitate automatic global monitoring of satellite imagery to identify anomalous water levels, extents, and flood-related conditions. The resultant metadata will be ingested into the system, and help to reduce the time window for downstream use of this knowledge. The second use case involves the analysis of satellite imagery to generate field-level crop maps, and facilitate and accelerate their use in downstream decision making.

Section 4 outlines the proposed architecture, a description of its components, and how they will be adapted and used. Section 5 describes the workflows and corresponding automation required for the EO data ingestion and to support the use cases for evaluation. Section 6 discusses the use of metadata as understood by the various components, and the necessary adaptations to facilitate interoperability between these components.

4. Architecture of the Unified Metadata Catalogue

From Figure **1,** we can see the overall architecture flow and usage pattern proposed by DaFab. The architecture is split into primary sections; firstly the analysis of EO data and extraction of metadata describing relevant features, and secondly interfaces and systems to enable the user to query and access appropriate EO data.

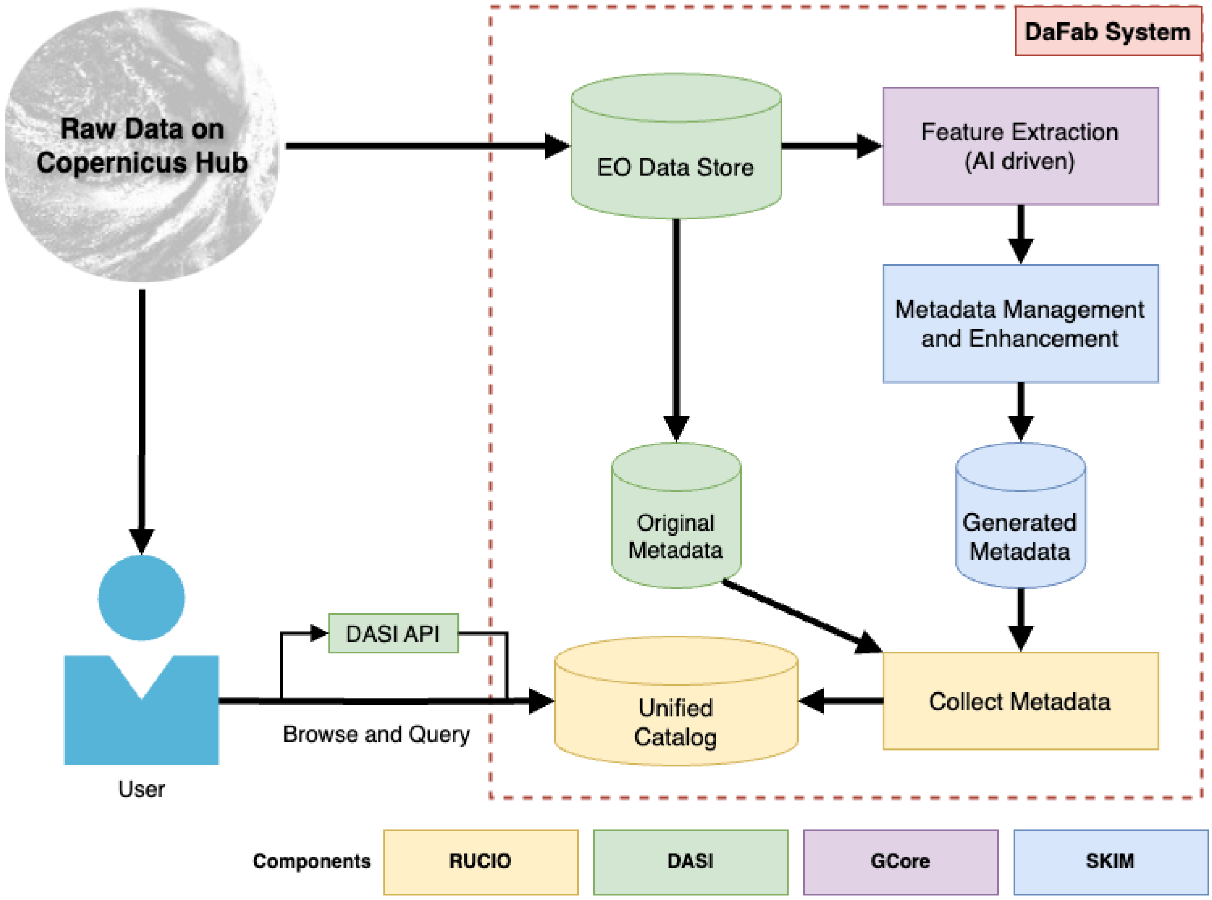


Figure 1: An overview of the DaFab architecture, indicating the data storage systems, the metadata management components, the interconnectivity of the components, and how the system interacts with the user.

Analysing vast quantities of Copernicus EO data takes substantial resources and time. One primary prerequisite is obtaining the EO data itself, and decoupling the computation from the data movements. To achieve this DASI will be deployed on the Lustre file system local to the metadata extraction processes. DASI provides a local high-performance object store to hold EO data objects and their corresponding descriptive metadata, indexed according to their identifying metadata. Tasks will be orchestrated to cache data here, and then to run AI-based feature extraction on this rolling cache as appropriate. The DASI store, and any other temporary storage make use of a Lustre file system, which supports advanced extended attributes (metadata) features. These features will be used to optimise the connection with RUCIO.

SKIM will be used to coordinate the feature extraction, and assemble the resultant generated knowledge along with any additional metadata available. This will then be presented to RUCIO, along with the original metadata held in DASI, to form the contents of the Unified Metadata Catalogue. Users will be able to query features in the EO data from the metadata catalogue held in RUCIO, and the results of this query will identify the underlying EO data of interest.

Once the EO data is ingested in RUCIO, the primary means to access the data from the DaFab catalogue will be through the RUCIO API. DASI will implement a backend able to query RUCIO, such that users/applications/workflows may access the data through the DASI API rather than using RUCIO directly, which facilitates integration of this EO data into workflows using semantic metadata in a DASI-compatible way.

The DaFAB system is always on, as user-facing frontends run at all times. Background components run intermittently as part of marshalled workflows to prepare data. This may be as a result of (periodically) updating the DaFab catalogue for newly created data, or as a larger processing task.

The workflows (coordinated by a workflow controller) comprise many different tasks, including those that retrieve the Copernicus data from its source, and those that process the data. These are decoupled in time, as data retrieval is limited by network capability, and we wish to avoid wasting computational resources. The data retrieval tasks will store data in the rolling cache. In a rolling fashion, once sufficiently large time windows of data are in the cache, the processing tasks will be launched to make use of this data. Once the data in the cache is no longer required (as per the workflows) it can be erased. The removal will be lagged to keep as much data in the rolling archive as possible, to eliminate the need to re-retrieve data if workflows need to be rerun, or the data may be required in another context.

Once the EO data is ingested in RUCIO, the primary means to access the data from the DaFab catalogue will be through the RUCIO API. DASI will implement a backend able to query RUCIO, such that users/applications/workflows may access the data through the DASI API rather than using Rucio directly, which facilitates integration of this EO data into workflows using semantic metadata in a DASI-compatible way.

From the user perspective, we can see two scenarios. First the production of metadata. In this case, the end-user will use SKIM API to submit a request for metadata generation. SKIM propagates this request, potentially with additional external data, to the DaFab orchestration layer based on ARGO. The orchestration layer will arbitrate on the best platform to execute the metadata generation. Metadata generation itself is obtained by the application of a GCORE's trained AI model (see section 4.1) on the dataset. Once data has been successfully tagged, the new set of metadata is propagated by the local RUCIO instance, the general network of RUCIO distributed metadata catalogue.

The second scenario is related to the exploitation of the enriched metadata space. A user is making a request to RUCIO to identify the best possible dataset from the HPC datahub. The query is processed at the level of RUCIO , and the set of data meeting request requirements is identified. For instance, all the images of Luxembourg with fields delineated, and no cloud coverage. The Orchestration layer will process the second phase by applying the user application to the data set identified by RUCIO. Depending on data gravity and resource availability, the DaFab orchestration Layer will dispatch the Kubernetes pods on one or several HPC Data Hub sites.

In addition to the use cases, multiple metrics will be collected to demonstrate the sustainability of the solution at scale. For instance, energy consumption metrics will be taken during the models training. A trade-off should be made between inference accuracy and performance. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025.

4.1. Communication between components

In a general way, regarding the distribution of roles within DaFab, RUCIO is gathering the Copernicus data and the new metadata. DASI is providing the Copernicus ones to RUCIO. SKIM is providing the new ones to RUCIO. All services interact with RUCIO using RUCIO's REST API or its Python client library. This API allows parties to insert, update, or remove metadata associated with data identifiers, such as individual Sentinel image entries. This generation or update of metadata ensures that the unified metadata catalogue remains current.

The different components in the system will communicate using JSON (JavaScript Object Notation) format as a baseline.

Extension of JSON will be involved for some specific 1:1 components communication channels.

Two extensions of JSON are used in the architecture:

- GeoJson (Geographic JSON) that is an extension of JSON specialised on earth observation data. Similarly
- STAC (Spatio Temporal Assets Catalog) is an extension of JSON which aims to organise a batch of JSON files.

Overall, the three formats are compatible but offer specific tools and functionality to provide a higher level of services.

Components communication (services with external interface) are based on REST API (GET, POST).

The communication will be orchestrated by another component than DASI, RUCIO, GCORE or SKIM.

Authentication and authorization for applications and services will be defined in a future phase of the project.

In addition to the inter-component communication, the system has to interact with end-users.

The general metadata format basis is JSON, and for spatial metadata (e.g. water mask), the format is COG (Cloud Optimised GeoTIFF) for raster and GeoJSON for vector.

Spatial indexing is critical for fast retrieval of geodata , the usage of spatial indexing will be defined in a future phase of the project.

4.2. Data Curation and Preparation

The Copernicus data that is to be mainly used as an AI pipeline input in agricultural flood use cases - Sentinel-2 Level-2A product data. AI pipeline consists of 3 major steps: preprocessing, model inference, post-processing.
Preprocessing of Sentinel-2 scene/tile before feeding to the AI model includes splitting the tile of S2 product (100 km by 100 km) into sub-tiles (256 by 256 pixels) for AI to process.
Post-processing is an algorithm of AI model outputs conversion to required metadata formats.

AI pipeline is taking into consideration cloudy scenes and scenes with missing data. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025.

4.3. Metadata Generation and Extraction - GCORE

AI Driven Metadata generation
GCORE accelerates AI training, provides comprehensive cloud services, improves content delivery, and protects servers and applications. One key technological differentiator of GCORE is its ability to master different computing, such as CPU, GPU, IPU and storage technologies.

During the DaFab project, GCORE provides computation resources for efficient EO AI models training and metadata generation/extraction workflows. GCORE also develops use-case-specific AI models for EO metadata extraction (e. g. parcel delineation, flood detection) in DaFab. In the framework of DaFab, two models will be built:

- Delineation of the fields in satellite images,
- Detection and characterization of water bodies

These models act as proof of concept for the overall architecture, demonstrating its applicability to a range of future models.

The metadata generation pipeline is complex, and needs to consider both model training and inference. To train the model, relevant datasets have to be built, largely from open-sourced datasets. The more accurate and the larger the datasets, the more accurate the final model will be in the inference phase. Under the framework of WP2, the training will be conducted on the most efficient architecture (CPU, GPU or IPU).Once the model is built, it will be run against the Copernicus data (inference) to generate the desired metadata. The newly generated metadata will be handled by the components described below.

As an example, for the field delineation use case the model should be able to generate GeoJSON. GeoJSON is a widely-used format for encoding geographic data structures. Ensuring output in this format will facilitate interoperability with various GIS tools and platforms. GeoParquet is columnar data for Geo that has Cloud Data Warehouse Interoperability [see Section 6.3].

Example of GeoJSON:


```
{
  "type": "FeatureCollection",
  "features": [
    {
      "type": "Feature",
      "properties": {},
      "geometry": {
        "coordinates": [
          [
            [
              5.710857266453388,
              50.18521713490796
            ],
            [
              5.710857266453388,
              49.44420720525011
            ],
            [
              6.543239677619653,
              49.44420720525011
            ],
            [
              6.543239677619653,
              50.18521713490796
            ],
            [
              5.710857266453388,
              50.18521713490796
            ]
          ]
        ],
        "type": "Polygon"
      }
    }
  ]
}
```


Figure 2: Example of GeoJSON metadata. This example is a Polygon approximating the geometry of the Luxembourgese borders.

Model Performance and Accuracy

There is a trade-off to be made between the accuracy and processing speed of the metadata generation. For crop classification and yield forecasting, it is important that the model have high-precision distinguishing between separate fields. The accuracy of field boundaries should be within ±1 pixel (10 m for Sentinel-2), recognizing the challenge posed by potential inaccuracies in ground truth data. For our initial release of the model, we assume the delineated fields will contain only one crop type.

As new Copernicus data is produced constantly, the inference stage must be run in an ongoing and automated manner to ensure this new data is labelled. More sporadically, if the model is updated, e.g. to produce more accurate results, it will have to be rerun and the metadata consequently updated. This should be a rare operation.

4.4. Metadata Management and Enhancement - SKIM

SKIM (Semantic Knowledge IMprover) is a backend software solution that specialises in managing and processing heterogeneous data from various sources that provide added-value to earth observation satellite images. The software does not directly store the satellite images or their associated metadata, as these are managed by the rolling cache DASI. Instead, SKIM focuses on integrating and processing diverse types of data that provide additional context and insights related to the satellite images. These data sources may include:

- **Social media events:** SKIM can collect and analyse data from social media platforms to identify events or trends that may be relevant to the satellite images, such as natural disasters, social gatherings, or cultural events.
- **Objects detected through image processing:** SKIM can incorporate the results of image processing techniques that detect and identify objects within satellite images, such as buildings, vehicles, or geographical features.

The data management includes ingestion, discovery, and transactions (add, update, delete) on metadata. The user search is handled by SKIM and can be easy (only image) or complex (object and event). The answer is an image.

SKIM is also able to consider user requests to follow events based on satellite images, in a specific Area-of-Interest (AoI) and Time-of-Interest (ToI). If the selected metadata exceeds user-defined thresholds, an alert can be raised. To enhance the follow-up, a satellite mission-programming request can be sent to another software to plan the mission of the earth observation satellite. These features may not be part of the DaFab project.

The structure of the metadata catalogue is based on STAC (Spatio Temporal Assets Catalog), refer to [6], which is an up-to-date standard for earth observation data. This standard interface (STAC: SpatioTemporal Asset Catalogues, 2024) allows SKIM to interface with many heterogeneous data providers and other software.

4.5. (Meta)data Catalogue - RUCIO

RUCIO is a comprehensive data management system developed by CERN, designed to handle the vast and complex data requirements of scientific experiments (Barisits et al., 2019). Within the DaFab project, RUCIO will serve as the central metadata catalogue, leveraging its robust data management capabilities to handle the vast and complex metadata associated with EO data. RUCIO will also be adjusted to integrate seamlessly with other components like SKIM and DASI, ensuring efficient metadata ingestion, storage, and querying. The following presents its architectural components and their roles.

At its core RUCIO is a scientific data catalogue, with distributed data management features (transfer of data, lifecycle management, identification of corrupt data, etc.) on top of it. This enables scientific experiments or organisations to manage exabytes of data distributed over many different data centres worldwide.

With the vast amounts of data under management by RUCIO, it is fundamental to enable users to efficiently find and access the data they require for their scientific workflows. To do this, RUCIO consists of a metadata component, which enables the data producers to associate metadata with the produced data products and users to write compound queries to find, and access, their needed data products.

To do this, RUCIO has a plugin-based approach to metadata. Associated metadata can either be stored within the RUCIO database itself, via a RUCIO-native metadata plugin based on json, or in external metadata systems, accessed via a custom-written plugin within RUCIO. This enables the user to just store/query metadata against RUCIO transparently, while in the backend the actual metadata queries are constructed and relayed to different backends via plugins.

Within the DaFab project, RUCIO will enhance this metadata component with several features. DaFab requires a quick turnaround between metadata queries and delivering the actual data products. To achieve this, we will investigate and implement indexing capabilities to increase the performance of the metadata lookup of compound metadata queries. Secondly, RUCIO's metadata model will be extended by integrating a semantic layer. This enrichment will enable more sophisticated and meaningful metadata representations, facilitating better search and retrieval capabilities. The semantic layer will use ontologies to represent relationships between metadata elements, allowing for the interlinking of metadata across different datasets. This approach enhances the ability to perform complex queries and obtain more relevant results. RUCIO’s metadata schema, the structured framework that defines the types and structures of metadata elements to ensure consistency and compatibility across the system, will be adapted to align with the unified metadata model defined by DaFab.

4.6. Rolling Data Store, Metadata Consumer and HPC Data Access - DASI

Meteorological and climate workflows are formed of a large number of interacting components, forming a web of producers and consumers. This encompasses not only the forecast generation itself, but also downstream data post-processing, statistical analysis, product generation and user-facing interactive tools.

The original metadata will be stored in DASI's internal database, called catalogue. This will be ingested in RUCIO, and later access will be provided via DASI’s RUCIO backend.

DASI provides a layer of abstraction between these workflow components and the underlying storage systems, implementing a metadata-driven object-store with a well-defined API and clear semantic behaviour. Workflows are configured to use a domain-specific metadata language, according to a schema, and then all of the data movements are described by applications solely using the domain-specific metadata language as depicted in

Figure 3. Data storage locations, data collocation, and data routing through the system are run-time configurable.

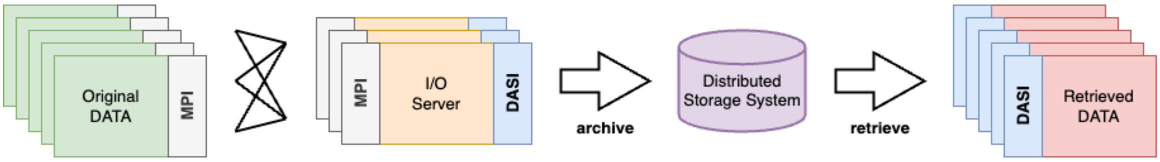


Figure 3: DASI in a simplified workflow from “original” to “retrieved” data.

DASI provides an API with very specific semantics to support data interchange between components of a large-scale HPC or cloud workflow (Sarmany et al., 2024; Smart et al., 2019). Across this API, and internally, DASI handles data using its metadata description via *messages*, which are self-contained, self-describing, and location-independent (can be handled equivalently in memory, on disk, tape or streaming from a network socket). A dataset can be considered as a sequence of messages. In this approach, many of the data handling and processing challenges are reduced to ones of routing messages through pipelines (Manubens et al., 2024).

Among other benefits, this approach allows scientific developers to focus on the domain-science without being distracted by the technical underpinnings of complex software and hardware systems. It unifies the data management in a range of workflow components.

Within the DaFab project, DASI will fill two functional niches. Firstly, the workflow that will generate metadata describing features in the EO data will require processing extremely large volumes of data, a Copernicus corpus of data (to be hosted in LXP datacenter) is in the range of 40PBytes. The I/O (Inputs/Outputs) and data movement requirements of this workflow will be very large. DASI will be used to implement a rolling store of Copernicus EO data which can be used to decouple data transfers from the local processing of the EO data. This work will involve development of a DASI-compatible metadata schema for the data to be stored, deployment on the local storage resources and integration with the data processing workflows.

Secondly, new backends will be implemented for DASI to enable user queries against metadata stored in RUCIO, and retrieving associated data from RUCIO. This will facilitate the enhancing of meteorological workflows, by enabling workflow components to access both near-real-time forecast and meteorological data alongside Copernicus Earth Observation data through the same interface.

5. Multi-Site Workflow Orchestration

In the context of the DaFab project, FORTH will design and implement a workflow orchestration system. The workflow system will enable workflows to run their stages at multiple sites (cloud or HPC) and access data transparently. As a result, it will shift application deployment and data discovery efforts to the platform, accelerating their development.

The workflow system will be responsible for moving data across. Results generated from workflow steps will be stored at the local site, and information about their location will be provided through Rucio. For performance purposes, the workflow system could move or create a copy of the data to a remote site and tell about its existence through Rucio.

The workflow system will run each step on Kubernetes (Cloud) or Slurm (HPC) sites. For the Slurm case, it will use and extend the HPK project[1] by FORTH, which runs Kubernetes jobs in a Slurm environment.

The workflow system will use the RUCIO catalogue to discover and access datasets before each workflow step. It will explore scheduling mechanisms to transfer data or the computation based on a) data locality and b) computing resource availability.

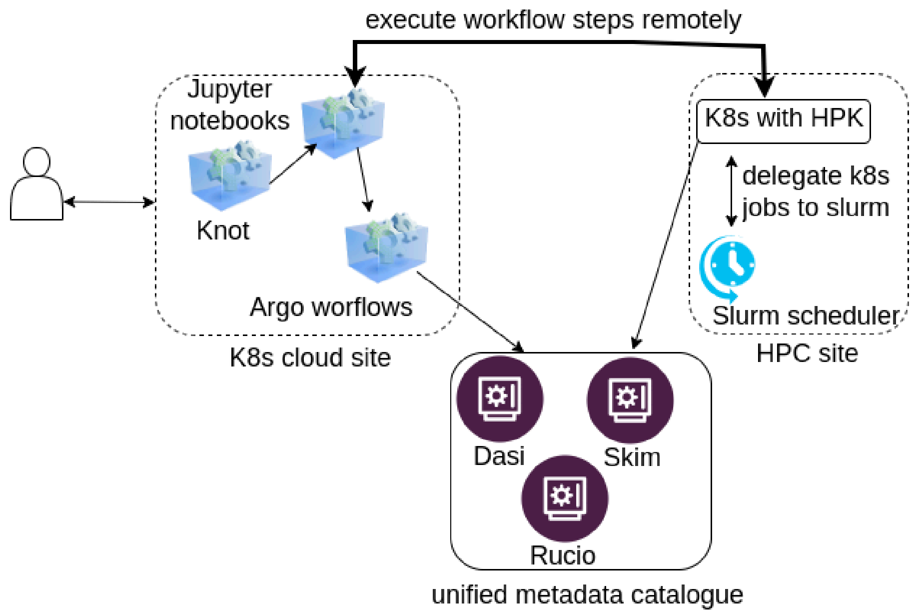


Figure 4: Workflow system architecture and interaction with the unified metadata catalogue.

5.1. High-Performance Kubernetes (HPK)

HPK (High-Performance Kubernetes) offers capabilities for remote site and high-performance computing (HPC) execution by seamlessly integrating Kubernetes with HPC environments that use Slurm. An HPC user can run HPK as a user service similar to all

[1] https://github.com/CARV-ICS-FORTH/HPK

other HPC jobs. HPK manages container workloads through the hpk-kubelet executable. HPK-kubelet acts as a virtual Kubernetes node representing the entire HPC cluster.

HPK translates container lifecycle actions into Slurm scripts and commands, enabling the efficient execution of Kubernetes-native workloads. HPK can automatically translate workloads described in Kubernetes' YAML format into Slurm scripts, which execute on the HPC cluster. This architecture eliminates the need for separate Cloud and HPC setups, thereby reducing hardware and maintenance costs while allowing users to utilize HPC resources through familiar Kubernetes interfaces.

The benefits of HPK are substantial, particularly in terms of flexibility and efficiency for both Cloud and HPC users. For existing HPC users, HPK opens access to a vast array of Cloud-native applications and services, enabling the execution of complex hybrid workflows that combine Cloud and HPC components without significant modifications. As a result, HPC centres no longer need to maintain distinct hardware partitions for different workloads; instead, they can run both Cloud and HPC tasks on the same infrastructure. Additionally, HPK facilitates the attraction of Cloud users to large HPC installations by providing a familiar Kubernetes interface, simplifying the use of HPC resources. This convergence enhances resource utilisation, streamlines workflow management, and broadens the potential for advanced computational research and enterprise applications.

It should be noted that HPK supports K8s auto-scaling. HPK can run all the native K8s pods that the auto scaler needs to operate. However, special permissions may be needed for the containers to access the machine performance counters.

5.2. Specifying and executing workflows

The required steps for applications to run on the workflow system are: a) Containerise each workflow step and b) describe the workflow as a graph in a data serialisation language such as YAML or code (e.g., Python). It will use and extend three basic technologies:

1) Kubernetes orchestration manager for the deployment and lifecycle of each step of the workflow
2) ARGO workflows for the workflow graph execution
3) RUCIO catalogue for the discovery and access to datasets. The workflow system will use the RUCIO catalogue to discover the locations of data. Based on their locations, the workflow system could either a) schedule the workflow step close to the site where the data are if computing resources are available. b) fetch the data to the current size and inform RUCIO that another replica of the data is present.

5.3. Knot

The workflow system will use and extend Knot[2], a web-based environment developed by FORTH where users can perform actual work on Kubernetes.

Knot is a complete environment for simplifying interaction with Kubernetes. It includes a set of web-based tools, as shown in Figure 5, to improve user productivity by avoiding the use of the command line and offering a single-point-of-entry for all tasks a user needs to perform.

[2] https://github.com/CARV-ICS-FORTH/knot

At its core, the Knot dashboard supplies the landing page for users, allowing them to launch notebooks and other services, design workflows, and specify parameters related to execution through a user-friendly interface. The dashboard manages users, wires up relevant storage to the appropriate paths inside running containers, securely provisions multiple services under one externally-accessible HTTPS endpoint, while keeping them isolated in per-user namespaces at the Kubernetes level, and provides an identity service for OAuth 2.0/OIDC-compatible applications.

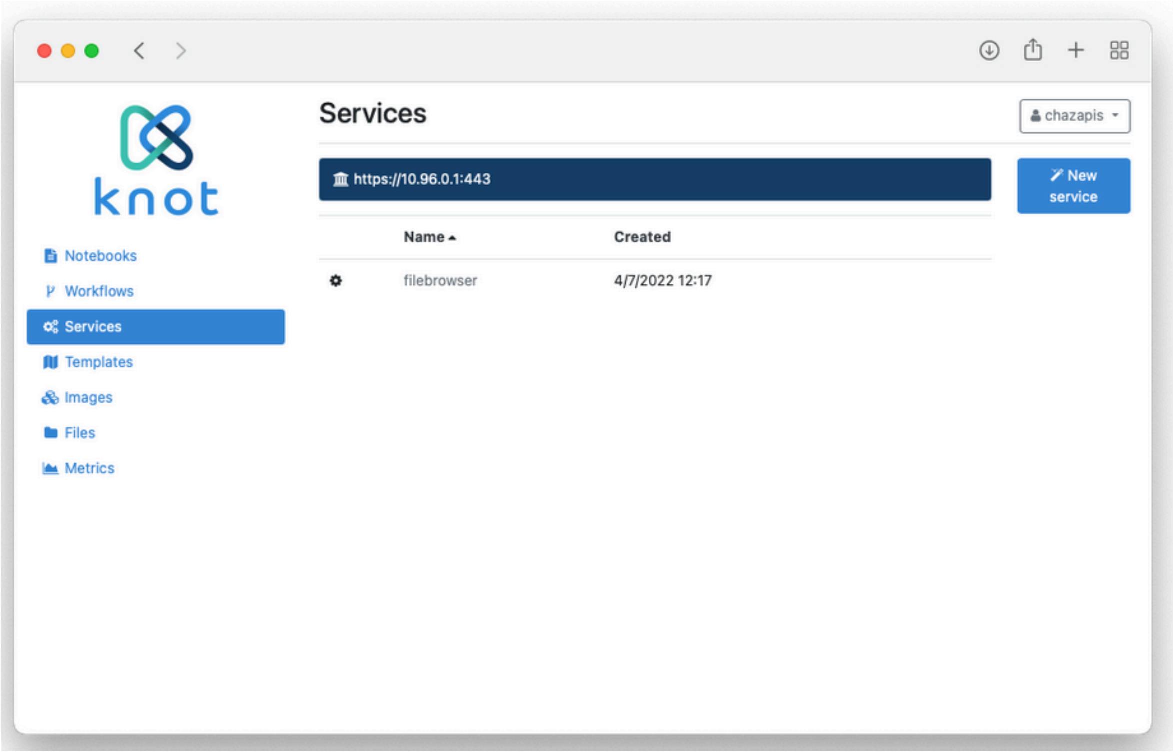


Figure 5: Knot web-based interface.

The KNOT user-facing component allows users to deploy and execute their workflows in the DAFAB platform. KNOT allows users to log in and initiate their workflows. While KNOT already includes user authentication, it will be extended to meet the specific requirements of the DAFAB project, enhancing both its functionality and security.

6. Metadata Ontologies and System Interoperability

Ontologies, schemas which describe the structure of data and metadata, will be used in DASI, SKIM and RUCIO.

6.1. Ontologies and Metadata

Digital data, in its raw form, is merely a sequence of 0s and 1s that lacks inherent meaning. To interpret and understand this data, it must be associated with specific semantics. Domain and application-specific knowledge are essential for ascribing meaning to these binary digits, allowing us to make sense of the information they represent. This applied knowledge is captured and defined through a set of metadata. Metadata provides the necessary context, structure, and descriptions that enable the accurate interpretation of the raw data, and facilitate its use.

An ontology is defined as a scheme, outlining a structure and set of concepts that describe how data (and its associated metadata) are organised. Within the scientific computing domain, ontologies are intrinsically linked to the scientific fields to which the data belongs.

Ontologies allow for effective metadata management by acting as a structured vocabulary that defines concepts, their properties, and the relationships between them. In the context of metadata, ontologies offer several advantages:

- **Standardisation:** Ontologies establish a common understanding of metadata elements and their meaning. This reduces ambiguity and ensures consistency across different datasets.
- **Interoperability:** By using a shared ontology, data from different sources can be easily integrated and compared. This allows for more comprehensive analysis and knowledge discovery.
- **Machine Readability:** Formalised ontologies enable machines to understand the structure and meaning of metadata. This facilitates automated data processing, search, and retrieval tasks.

Defined ontologies facilitate managing data and metadata by providing four (4) key features:

1. **Defining Concepts:** An ontology identifies the key concepts relevant to a specific domain or data type. For example, in the field of an Earth Observation data repository, concepts might include "time", "geolocation", and "instrument used for acquisition"
2. **Specifying Properties:** Ontologies define the properties or attributes associated with each concept. For instance, the "instrument used for acquisition" concept might have properties like "satellite identifier", "lens properties", and "image resolution".
3. **Establishing Relationships:** Ontologies specify the relationships between concepts. These relationships can be hierarchical (e.g., "Sentinel-2" is a type of "satellite") or more complex (e.g., "observed in" relates a "satellite" to a "location").
4. **Vocabularies and Codes:** Ontologies may also incorporate controlled vocabularies or code lists to ensure consistent use of terms for specific properties.

These features lead to the definition and use of various types of metadata:

1. **Identifying metadata:** This is a minimal set of metadata that uniquely identifies a piece of data for storage or retrieval.
2. **Technical Metadata:** This metadata describes how a piece of data has been created, prepared or stored. It is typically required for the use of data and may come bundled with it upon retrieval. It includes details such as grids, encodings and file format information.
3. **Structural Metadata:** This metadata helps identify relationships between multiple pieces of data or helps identify other data related to known data.
4. **Provenance Metadata:** This metadata describes the origin, authorship, processing, and chain of custody of data.
5. **Descriptive Metadata:** This metadata describes the contents or properties of data, especially from the perspective of an end user. For example, it identifies the types of subject in an image rather than how the image should be technically decoded. This is highly useful to end users when identifying data of interest.

The proposed architecture integrates components that utilise different types of metadata, making it easier for users to access and use the data.

6.2. Geospatial image and metadata

Geospatial observation is a mature domain, with a large corpus of tools, frameworks and data formats.

In EO, the ubiquitous image data format is TIFF, which supports a wide range of metadata embedded within the file. Not all metadata are mandatory, and they can be accessed with standard tools including `tiffinfo`.

- **Basic Image Data:** Image width, height, resolution, colour depth, and compression type.
- **Camera Information:** Maker, model, shutter speed, aperture, focal length, and ISO speed used to capture the image.
- **Date and Time:** The date and time the image was taken.
- **Copyright and Licensing:** Information about who owns the rights to the image.
- **Software Used:** Software used to create or edit the image.
- **Custom Tags:** Some programs may add their own custom tags to the TIFF file.

There are two main types of tags used in TIFF metadata:

- **Standard TIFF Tags:** Defined by the TIFF specification and widely supported by most software programs.
- **Private Tags:** Defined by specific software programs and may not be understood by other programs.

Sentinel-2 Metadata:

In addition to standard TIFF metadata, the Sentinel-2 program has developed its own set of metadata to describe Sentinel-2 satellite imagery. These metadata are used for data quality assessment, to ease the interpretation (e.g. cloud coverage), to facilitate interoperability in workflows, and to aid in long-term preservation and archiving of images. In more detail:

1. **Data Quality Assessment:** Provides information about the data's acquisition conditions, processing steps, and quality metrics, allowing users to assess the suitability of the data for their specific application.
2. **Data Interpretation:** Offers context for understanding the data content, such as cloud cover percentage, atmospheric correction status, and sensor calibration details, which are crucial for accurate data interpretation.
3. **Data Integration:** Facilitates the integration of Sentinel-2 data with other datasets, such as land cover maps or topographic data, by providing information about the data's spatial and temporal reference systems.
4. **Data Archiving and Preservation:** Plays a vital role in long-term data archiving and preservation by documenting the data's origin, processing history, and quality characteristics, ensuring its future usability.

The Sentinel-2 metadata can be broadly categorised into two main types:

1. **Product Metadata:** This metadata is associated with the Sentinel-2 data product itself and provides information about the data acquisition, processing, and quality. It includes details like:
   - **Product Identifier:** Unique identifier for the data product
   - **Acquisition Date:** Date and time the data was acquired
   - **Sensor:** Sentinel-2 sensor that acquired the data (e.g., Sentinel-2A or Sentinel-2B)
   - **Processing Level:** Level of processing applied to the data (e.g., Level 1C, Level 2A)
   - **Resolution:** Spatial resolution of the data for each spectral band
   - **Cloud Cover:** Percentage of cloud cover in the image
   - **Processing History:** Steps involved in processing the data from raw sensor data to the final product
2. **Auxiliary Metadata:** This metadata provides additional information that complements the product metadata and enhances the understanding of the data. It includes details like:
   - **Sensor Calibration:** Calibration parameters for the Sentinel-2 sensors
   - **Geometric Information:** Georeferencing information for the image, including projection and transformation details
   - **Atmospheric Information:** Atmospheric correction parameters applied to the data
   - **Validation Data:** Validation data used to assess the accuracy of the data

Parsing and accessing the Sentinel-2 metadata can be made through different methods or channels:

1. **Copernicus Open Access Hub:** The Copernicus Open Access Hub (OA Hub) provides free access to Sentinel-2 metadata in JSON and XML formats. Although the OA Hub was closed in October 2023, it remains a valuable reference for metadata access.
2. **Data Hub Catalogue API:** The Copernicus Data Hub Catalogue API allows programmatic access to Sentinel-2 metadata using various query parameters. This API provides a flexible way to search and retrieve metadata based on specific criteria.

3. **Sentinel-2 Toolbox:** The Sentinel-2 Toolbox is a software suite including tools for processing and analysing Sentinel-2 data. The toolbox provides access to Sentinel-2 metadata within its workflow.
4. **Third-party Tools:** Several third-party tools and libraries can be used to access and process Sentinel-2 metadata, such as GDAL, Rasterio, and ESA's Sentinels Application Platform (SNAP).

Managing Metadata: The FAIR Principles

Any proposed metadata extension must consider the FAIR principles. These principles are general guidelines to help make data more accessible to a range of users. The metadata catalogue being developed in DaFab aims to support FAIR use of data in the following ways:

- **Findable:** The primary purpose of a metadata catalogue is to facilitate the discovery of data by exposing a range of metadata queries to the user.
- **Accessible:** All data will be retrievable by identifying metadata queries, and utilizing the system's APIs.
- **Interoperable:** By bridging and interlinking systems with different views of data access and descriptions, DaFab will enable users to access multiple datasets uniformly, enhancing the overall interoperability of the EO space.
- **Reusable:** Data are encoded according to widely accepted, domain-relevant community standards. The stored metadata and provenance information will further enable the data to be used in a wider range of contexts.

6.3. DaFab metadata list

In addition to the standard metadata provided by Copernicus (product metadata and calibration metadata), DaFab aims to add use case specific metadata. For instance, for the smart agriculture use case, we propose an initial set of metadata presented in Table 1.

Metadata name	Description	Metadata type/format	Goal
Parcel boundaries	The field boundaries for individual parcels, updated once per growing season. These are computed several weeks before the typical harvest time.	Field boundaries are to be provided as closed Polygon or MultiPolygon objects according to the most recent GeoJSON (RFC 7946) or GeoParquet specifications.	Consolidating input data to one vector per agricultural field, rather than per pixel, will dramatically lower processing time and cost for classifying crop types and forecasting yields.
Agricultural coverage	The percentage of agricultural land within a satellite scene as aggregate score. It is computed as the total area of all field polygons, divided by total area covered by the	Percentage value per scene updated each new growing season.	Enable filtering of scenes based on agricultural coverage to reduce the amount of EO data required for downloading and processing.

	image.		
Cloud-free agricultural coverage	The percentage of land within a satellite scene which is used for agriculture and not obscured by clouds in this observation. It is computed as the total area of all field polygons not partially covered by clouds, divided by total area covered by the image.	Percentage value per scene updated every acquisition of satellite in the growing season.	Enable filtering of scenes based on agricultural land with preferred (e.g. minimal) cloud cover.

Table 1: Definition of the metadata to be generated by the DaFab AI model to feed the use case on smart agriculture

The metadata associated with the crop yield forecast use case will vary based on the agricultural seasons, as the crops grown change between summer and winter. The crop yield forecast will be recomputed for each crop season using the appropriate Sentinel-2 data as an input. To streamline data discovery and selection for the crop yield forecast, users would be able to filter and search the Sentinel-2 data catalogue using metadata values assigned to each dataset. These metadata values reflect specific characteristics of the data, such as the presence of crop fields or cloud coverage extent, enabling users to quickly identify and select datasets relevant to their specific interests and requirements.

From the smart agriculture use case definition, the fields boundaries are stored in vector format. The size of this metadata is not assessed yet, and will be detailed in a future deliverable.

At the time of writing of this deliverable, the second use case on Flood detection is less mature. Our initial design for metadata is detailed in the Table 2.

Metadata name	Description	Metadata type/format	Goal
Reference Water mask	Mask of pixels in the scene representing open/calm water bodies (either permanent or seasonal).	Raster data in Cloud Optimised GeoTiff format	Define a baseline to compare water extent anomalies.
Surface of permanent water bodies	The area of permanent water bodies in the scene.	Numeric value in square kilometres	Enable filtering of scenes based on the presence of permanent water bodies.

Observed water bodies	Mask of pixels in the scene representing currently observed water bodies.	Raster data in Cloud Optimised GeoTiff format	Monitor/analyze changes in water bodies over time.
Surface of observed water bodies	The area of observed water bodies in the scene.	Numeric value in square kilometres	Enable filtering of scenes based on the presence of observed water bodies.
Water extent anomaly score	Extent difference between observed water bodies and reference water bodies. Positive anomalies indicate increased water levels, potentially signalling flooding, while negative anomalies suggest reduced levels, possibly indicating drought.	Numeric value in percentage	Enable filtering of scenes based on user-defined thresholds for water extent anomaly.
Anomaly type	Labels the observed scene as normal, or flood/drought based on detected anomalies.	Labels (normal, flood, drought, flood and drought)	Enable filtering of scenes based on the selected type of anomaly.
Delineated Water anomaly	The delineated difference between observed water extent and reference water, if water extent anomaly is present in the scene.	Vector data in GeoJSON	Allows users to focus on affected water extent anomaly areas. Vector data allows users to easier refine the affected area.

Table 2: Draft of the metadata for the second use case on flood prediction and prevention.

The system will calculate metadata for the flood detection use case for every archived and new acquisition of Sentinel-2 and Sentinel-1. Users can then filter the catalogue to find scenes of interest based on the presence of water bodies and the severity of any detected water extent anomalies.

Regarding metadata size, for the flood detection use case, the masks (for example water mask) are stored in raster format. The size of this metadata is not assessed yet. This will be detailed in a future deliverable.

6.4. RUCIO

RUCIO is a central component in DaFab architecture, as all the users' queries are destined to RUCIO, which hosts the unique catalogue. This unique catalogue gathers the Copernicus data and metadata and the new metadata generated by the DaFab system.

Current State of Metadata Handling in RUCIO

In its current state, RUCIO manages metadata through a flexible architecture that includes a plugin system and a powerful filtering engine. The plugin system in RUCIO allows for extensibility by enabling custom plugins to handle metadata operations. This system starts with the BASE plugin, which manages core metadata types, and can include additional CUSTOM plugins for handling specific metadata requirements. For instance, RUCIO supports plugins for both MongoDB and PostgreSQL, allowing for flexible backend storage solutions, but also offers the possibility to write plugins against proprietary metadata stores. The filtering engine in RUCIO enables users to perform complex queries on metadata, supporting a variety of logical operators and conditions to facilitate precise data retrieval. This engine translates user queries into executable commands across different storage backends, ensuring efficient data access.

RUCIO supports bulk operations for getting and setting metadata, metadata inheritance, but also adding both "optimized" and "generic" metadata. Optimized metadata is stored in fixed columns in the data identifier's table and is managed by the base plugin, while generic metadata is stored in JSON blobs, managed by the JSON metadata plugin. Therefore, users can add optimized metadata like ***'panda_id'*** using the base plugin, or generic metadata using the JSON plugin by default. Retrieval of metadata can be done through the ***'RUCIO get-metadata'*** command in the command line interface (CLI) or via corresponding functions in the Python API, with options to specify plugins. The filtering engine enhances metadata functionality by enabling complex query expressions with logical (***";"*** for OR, ***","*** for AND) and comparison operators (***"="***, ***"!="***, ***">"***, ***"<"***, ***">="***, ***"<="***, ***"LIKE"***, ***"NOT LIKE"***). These expressions are translated into executable queries across different storage backends like PostgreSQL, and MongoDB.

Required Adaptations for Interoperability

To facilitate interoperability between the various components of the DaFab system, RUCIO's metadata handling will need several adaptations. Firstly, RUCIO's metadata schema will need to be aligned with the unified metadata model defined by DaFab. This alignment will ensure that all metadata conforms to a consistent structure, making it easier to integrate with other components like SKIM and DASI.

Integrating a semantic layer into RUCIO’s metadata model will be crucial. This semantic layer will leverage domain-specific ontologies (as described in the current section above) to enhance the existing metadata with rich, contextual information. The ontologies will provide a structured vocabulary to define key concepts, properties, and relationships relevant to the Earth Observation domain. By mapping RUCIO's metadata schema to these ontologies, the metadata will gain a standardised, machine-readable representation. This will enable RUCIO to support complex, semantically rich queries that can intelligently navigate the relationships between metadata elements. For example, a user could query for all satellite images covering a specific geographic area, acquired within a certain timeframe, and having cloud cover below a given threshold. Furthermore, ontology alignment techniques will be employed to establish semantic links between related metadata elements across different datasets. This interlinking will allow RUCIO to infer additional insights and enable more comprehensive search and retrieval capabilities across the entire DaFab data ecosystem.

Additionally, leveraging PostGIS for geospatial data management could significantly enhance RUCIO's capabilities. PostGIS extends PostgreSQL to handle geographic objects, making it ideal for managing and querying geospatial metadata (PostGIS: Spatial and Geographic Objects for PostgreSQL, 2024). By integrating PostGIS, RUCIO can efficiently manage spatial data, perform complex spatial queries, and support use cases involving geospatial analysis.

Enhancing the plugin system to better support these semantic and geospatial extensions will be necessary. This includes developing new plugins or extending existing ones to handle the enriched metadata types and ensuring that the filtering engine can process these advanced queries effectively. Overall, these adaptations will ensure that RUCIO can seamlessly and efficiently integrate with the broader DaFab ecosystem, facilitating efficient metadata management and interoperability between system components.

6.5. SKIM

Current state of metadata structure in SKIM

As a background of the DaFab project, SKIM is a software to manage the earth observation valorized (external to images) metadata. The standardisation of the metadata structure and associated interface was mandatory to interoperate image metadata and valorized metadata.

The standard chosen is STAC (Spatio Temporal Asset Catalogue) which is based on JSON (Java Script Object Notation) standard, refer to [6].

This standard allows structuring data with hierarchical links. Therefore SKIM catalogues is using STAC standard, which allow to interoperate with other STAC catalogues in a virtual unique one.

SKIM catalogue is based on MongoDB database solutions.

The User search service was integrated in another TAS component, composed by a division function (satellite images metadata request for this other component, external metadata request for SKIM) and a combination function. This service allows the validation of simple use cases, for example, the search of flood satellite images from a hurricane that occurred in Florida, USA in Autumn 2022.

Challenges for DaFab project

For the DaFab project, the proven interoperability of two different softwares and database solutions can be extended to other softwares of DaFab consortium. The User search service will be integrated in SKIM to simplify the system. The STAC standard, even if allowing hierarchical links can be challenged. It may be compared to semantic-web standards allowing interlinks.

6.6. DASI

Within the scientific workflows in HPC systems, data may be produced by different and unrelated components, and stored in a range of accessible subsystems. These may change with time over the lifetime of an operational system. One of the guiding principles of DASI is that the scientific software should focus on producing and consuming data according to its domain-appropriate metadata description rather than data management through direct specification of any data locations.

The language used within DASI is derived from the MARS language, originating in ECMWF's Meteorological Archival and Retrieval System (MARS). All data operations can be described using this same language within this ecosystem. This not only facilitates integration of different systems but also enables the domain scientists to work with their data using scientifically and semantically meaningful terms without knowing the underlying technical details, such as network and storage technologies.

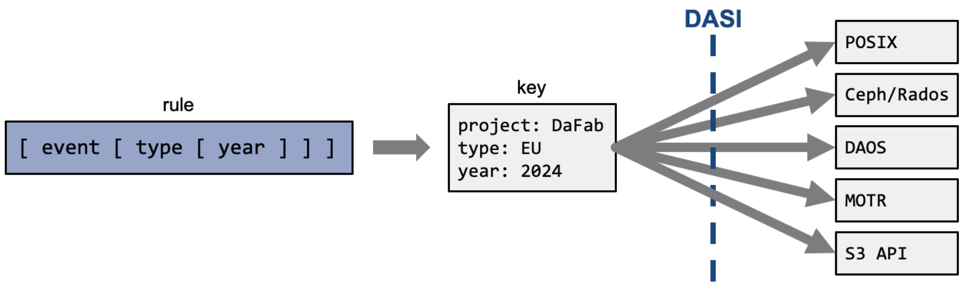


Figure 6: Example data archival using schema rule and key via DASI.

Figure 7 shows simple archival and retrieval operations for data in DASI. A third example shows the input to a post-processing tool, which itself retrieves the data from DASI, demonstrating that the metadata language can be common across entire workflows.

```
archive,
  class    = od,
  expver   = 0001,
  stream   = oper,
  date     = 20240526,
  time     = 1200,
  step     = 1/to/24,
  param    = z/t,
  levtype  = pl,
  levelist = 1000/500,
  source   = in.grib
```

```
retrieve,
  class    = od,
  expver   = 0001,
  stream   = oper,
  date     = 20240526,
  time     = 1200,
  step     = 1/to/24,
  param    = z/t,
  levtype  = pl,
  levelist = 1000/500,
  target   = out.grib
```

```
disseminate,
  class    = od,
  expver   = 0001,
  stream   = oper,
  date     = 20240526,
  time     = 1200,
  step     = 1/to/24,
  param    = z/t,
  levtype  = pl,
  levelist = 1000/500,
  grid     = 0.5/0.5,
  area     = europe,
  target   = XXX:ZZ
```

Figure 7: A range of data operations for the same data

There are two primary goals for DASI in the DaFab project. Firstly, we will explore the use of DASI to store indexed EO data within the data analysis pipeline, to create a rolling, local store of data to be processed, decoupling the processing from any required data transfer. This will involve standardising the identifying metadata ontology for the EO data under consideration and defining and deploying an appropriate schema. This functionality may be deployed on cloud systems using Ceph as the relevant data backend, or HPC systems using the POSIX backend on top of a parallel file system.

Secondly, DASI will make data stored in RUCIO available using DASI as an interface. This will also enable other software that already uses DASI to directly use data from RUCIO, alongside existing datasets, in their workflows. To achieve this, we will extend the metadata model of DASI to support queries against technical and descriptive metadata (stored in RUCIO), in addition to the “pure” queries that uniquely identify metadata as per this example.

6.7. Platform support for metadata acceleration

The storage system on the platform can accelerate the querying mechanism. In DaFab two approaches are investigated:

- Lustre Extended attribute: the Lustre file system, operated at LXP facility, supports Extended Attribute. This mechanism is an optional POSIX generic interface to tag files with additional metadata. In DaFab we want to investigate the possibility of enriching these fields with AI-generated metadata. The ability of the file system to natively handle this extended metadata will greatly streamline and accelerate the interoperability with RUCIO. For instance: massive exports of metadata can be done periodically from the platform to the metadata catalogue; Data movements can be done transparently from one Lustre instance to another.
- KV-STORE: DaFab will design and implement a key-value store capable of querying and ingesting various types of metadata. It will ensure predictable performance regardless of the size or nature of the write (insert, update, or delete) or read operations. Additionally, its design will optimise technologies, such as NVMe storage devices and RDMA networking in HPC and cloud environments. DaFab will investigate the integration of the KV store either within RUCIO or as a metadata backend engine for one of the project's metadata providers, such as DASI or SKIM.

Regarding the Lustre Extended Attribute, in the framework of DaFab a new tool is under development to optimise Lustre metadata interoperability. This tool is based on the original `lipe_find`[3] [7] Lustre library.

To contextualise the development of the tool, some background needs to be provided. Within Lustre the metadata service is an independent service from the data layer, the metadata service runs in dedicated virtual machines and most of the time on dedicated hardware.
To perform access to the metadata service users need to query files (data service) for their metadata attributes. The data service redirects the call to the metadata service which in turn

[3] https://wiki.lustre.org/images/9/92/Thur08-XiLi-Policy_Engine_LUG_LiXI_0.5.pdf

replies to the end-user. This mechanism is efficient and performs well for most of the workloads.

However, DaFab is a slightly different use case, where metadata are generated at scale, and have to be collected to feed a Metadata catalogue in charge of answering queries. In DaFab, our work intends to develop an approach where the metadata service is directly accessed through a specific low-level Lustre API (Lustre `lipe_find`) to handle massive access to metadata. These massive accesses result in bulk ingest of updates for the metadata catalogue. Our tool supports the extended_attribute, thus allowing acceleration of DaFab-defined metadata. The tool will be made open-source once it has reached the relevant maturity level.

Regarding the scalability of DaFab approach, two metrics have to be considered. First the extra requirement for capacity, and the processing cost of generating the metadata.
As an example, regarding our smart agriculture use case, our current model generates a 1-bit mask for every pixel: if the pixel is part of a field border or not. The current size of a geopatch of 256x256 pixels (65 K pixels) is 1MB, thus 16B or 128 bits per pixel. The output of the model is a mask, 1 bit per pixel. Consequently, the total volume of raw metadata is ($^{1}/_{128}$)th of the data size. The total volume of data hosted in the Copernicus database is in the range of 45PB, Thus DaFab will generate an additional 350TB of metadata. This may sound considerable, however the mask can be compressed as a very high ratio, either with standard image compression ratio (RLE) or with more complex transformation such as polygon extraction out of the mask. Therefore we only expect a marginal capacity increase (few TBs at most). The size of the model for mask generation is in the range of 650 MB which is negligible in respect of the data volume we are addressing.
The second aspect of scalability is the computational overhead of generating the mask. This aspect is linked to WP2 T2.1. This task is focused on assessing the cost-performance of different hardware solutions. Our model running on a single core CPU node processes 8 images per second for the training.

7. Summary

The Unified Metadata Catalogue is a core component of the DaFab system. It will make a rich set of metadata ontologies, and the corresponding generated metadata, available to the end user. In doing so it alleviates one of the most problematic aspects of the use of EO data - identifying relevant datasets, and data within these datasets containing features of interest.

Behind this interface lies a larger machinery. A system is required that efficiently processes the available Copernicus EO data to identify features of interest, and generate commensurate secondary metadata. The workflows that carry this out are complex, require significant resource and data movement and the orchestration of these components working together. Various sources of knowledge about this data need to be merged, and the results made available to the user. When the metadata is queried, the system must be able to link the results back to the original EO data to satisfy the users scientific needs.

This deliverable presents how a number of components, prepared by different project partners, will be deployed across a range of virtual, cloud and HPC platforms, and orchestrated to work together to make this Unified Metadata Catalogue a reality.

8. References

[1] Barisits, M., Beermann, T., Berghaus, F., Bockelman, B., Bogado, J., Cameron, D., Christidis, D., Ciangottini, D., Dimitrov, G., Elsing, M., Garonne, V., Di Girolamo, A., Goossens, L., Guan, W., Guenther, J., Javurek, T., Kuhn, D., Lassnig, M., Lopez, F., … Wegner, T. (2019). RUCIO: Scientific Data Management. Computing and Software for Big Science, 3(1), 11. https://doi.org/10.1007/s41781-019-0026-3

[2] Manubens, N., Smart, S. D., Danovaro, E., Quintino, T., & Jackson, A. (2024). Reducing the Impact of I/O Contention in Numerical Weather Prediction Workflows at Scale Using DAOS. Proceedings of the Platform for Advanced Scientific Computing Conference, 1–12. https://doi.org/10.1145/3659914.3659926

[3] PostGIS: Spatial and Geographic Objects for PostgreSQL (3.4.2). (2024). [Computer software]. PostGIS Project Steering Committee & OSGeo. https://postgis.net

[4] Sarmany, D., Valentini, M., Maciel, P., Geier, P., Smart, S., Aguridan, R., Hawkes, J., & Quintino, T. (2024). MultIO: A Framework for Message-Driven Data Routing For Weather and Climate Simulations. Proceedings of the Platform for Advanced Scientific Computing Conference, 1–12. https://doi.org/10.1145/3659914.3659938

[5] Smart, S. D., Quintino, T., & Raoult, B. (2019). A High-Performance Distributed Object-Store for Exascale Numerical Weather Prediction and Climate. Proceedings of the Platform for Advanced Scientific Computing Conference, 1–11. https://doi.org/10.1145/3324989.3325726

[6] STAC: SpatioTemporal Asset Catalogs. (2024). https://stacspec.org

[7] Qian, Y., Cheng, W., Zeng, L., Li, X., Vef, M. A., Dilger, A., ... & Brinkmann, A. (2023, November). Xfast: Extreme File Attribute Stat Acceleration for Lustre. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (pp. 1-12).

DaFab Consortium

ddn

CERN

LUX PROVIDE
NEXT-LEVEL INSIGHTS WITH SUPERCOMPUTING

ECMWF

Jožef Stefan Institute

ThalesAlenia Space
a Thales / Leonardo company

FORTH
FOUNDATION FOR RESEARCH AND TECHNOLOGY - HELLAS

PROJECT:	DaFab: AI Factory for Copernicus Data at Scale	**REVIEWER:** Niko Lukac	
DELIVERABLE REF:	Deliverable D3.1	**VERSION**	1.0
DELIVERABLE TITLE:	Unified Catalogue Format Description	**DATE:**	29.07.2024
AUTHOR:	Simon Smart et al.		
FINAL RECOMMENDATION:	Modification required (N)		

The presented document of the Deliverable 3.1 is well-structured and well-written. Updates and comments are approved.

Additional comments:

Detailed comments

Id.	Chapter / Page	Type (*)	Reviewer's comments and/or proposals	Author's response and agreed action	Approved / More information needed
1	1 / 3	m	Definition of CRS is missing.	CRS is the acronym of Coordinate Reference System. The terminology has been extended.	Approved

Id.	Chapter / Page	Type (*)	Reviewer's comments and/or proposals	Author's response and agreed action	Approved / More information needed
2	4 / 7	m	It is unclear how the different components will communicate, what protocols will be used, and how different data formats will be exchanged. There are different metadata formats mentioned for each component (e.g. STAC, GeoJson, json), how will they be interchanged is unclear. It is unclear about authentication and authorization concerns.	The JSON (JavaScript Object Notation) format applies as a basis. The GeoJson (Geographic JSON) is an extension of JSON for earth observation data. The STAC (Spatio Temporal Assets Catalog) is an extension of JSON which aims to organise a batch of JSON files. Therefore the 3 formats described are compatible. Components communication (services with external interface) are based on REST API (GET, POST, etc…). The communication will be orchestrated by another component than DASI, RUCIO, GCORE or SKIM. Authentication and authorization for applications and services will be defined later. (Patrick)	Approved
3	4 / 8	m	It is unclear what is the final data format to the end user of DaFab architecture. Spatial indexing shall be discussed, as it is vital for fast retrieval of geodata (although it is mentioned briefly to be incorporated in RUCIO updates).	Refer to previous answer, the metadata format basis is JSON. For the spatial metadata (water mask for example), the format is COG (Cloud Optimized GeoTIFF) for raster and GeoJSON for vector. The spatial indexing should be studied, answer will be provided in Q1-2025 within the annual report.	Approved

					information needed
4	4 / 8	m	It is not entirely clear if DASI API has to be used to access unified catalog or if it can be done directly using RUCIO API.	Once the EO data is ingested in RUCIO, the primary means to access the data from the DaFab catalogue will be through the RUCIO API. DASI will implement a backend able to query RUCIO, such that users/applications/workflows may access the data through the DASI API rather than using Rucio directly, which facilitates integration of this EO data into workflows using semantic metadata in a DASI-compatible way. The Figure 1 is unclear, to be updated in the next version of the document. (Simon)	Approved
5	6 / 17	m	Unclear in the overall Fig. 1 how ontologies will be used in what component.	Ontologies, schemas which describe the structure of data and metadata, will be used in DASI, SKIM and RUCIO. (Patrick)	Approved

(*) Possible Types are M / m / U / Q where M=Major, m=minor, U=Non compliance with contractual requirements and Q=Question.

Deliverable Review Sheet (DRS)

PROJECT:	DaFab: AI Factory for Copernicus Data at Scale	**REVIEWER:** Pierre-Marie Brunet	
DELIVERABLE REF:	Deliverable D3.1	**VERSION**	1.0
DELIVERABLE TITLE:	Unified Catalogue Format Description	**DATE:**	31.07.2024
AUTHOR:	Simon Smart et al.		
FINAL RECOMMENDATION:	Modification required (N)		

The Unified Catalog Format Description is well-presented overall. The DaFab framework's reliance on existing software, which will be extended, is an asset that will accelerate the project's progress. However, it is crucial to describe the interactions and interfaces between each component precisely. The document could be expanded in some areas to address this point.

Additional comments:

Detailed comments

(*) Possible Types are M / m / U / Q where M=Major, m=minor, U=Non compliance with contractual requirements and Q=Question.

Id.	Chapter / Page	Type (*)	Reviewer's comments and/or proposals	Author's response and agreed action	Approved / More information needed
1.	4 / 7	m	It is not clear if the DaFab System runs continuously and if so how the rolling cache (EO Data Store) is managed.	The DaFab system has a number of components. The user-facing frontends run at all times. The background components run intermittently as part of marshalled workflows to prepare data. This may be as a result of (periodically) updating the DaFab catalogue for newly created data, or as a larger processing task. The workflows (coordinated by a workflow controller) comprise many different tasks, including those that retrieve the Copernicus data from its source, and those that process the data.	Approved

				These are decoupled in time, as data retrieval is limited by network capability, and we wish to avoid wasting computational resources. The data retrieval tasks will store data in the rolling cache. In a rolling fashion, once sufficiently large time windows of data are in the cache, the processing tasks will be launched to make use of this data. Once the data in the cache is no longer required (as per the workflows) it can be erased. The removal will be lagged to keep as much data in the rolling archive as possible, to eliminate the need to re-retrieve data if workflows need to be rerun, or the data may be required in another context. (Simon)	
2.	4/7	m	It is not apparent from the report how users will get access to the EO Data once selected thanks to the DaFab catalogue. It seems to be through DASI but is it a specific API, is it compliant with existing software?	Once the EO data is ingested in RUCIO, the primary means to access the data from the DaFab catalogue will be through the RUCIO API. DASI will implement a backend able to query RUCIO, such that users/applications/workflows may access the data through the DASI API rather than using Rucio directly, which facilitates integration of this EO data into workflows using semantic metadata in a DASI-compatible way. (Simon)	Approved
3.	4/7	M	Raw Data on Copernicus Hub are not for all of them ready for AI processing. The report lacks details on EO Data prerequisites (potential data conversion, etc.)	The Copernicus data that is to be mainly used as an AI pipeline input in agricultural / flood use cases - Sentinel-2 Level-2A product data. AI pipeline consists of 3 major steps: preprocessing, model inference, post-processing. Preprocessing of Sentinel-2 scene/tile before feeding to the AI model includes splitting the tile of S2 product (100 km by 100 km) into sub-tiles (256 by 256 pixels) for AI to process, etc. Post-processing is an algorithm of AI model outputs conversion to required metadata formats.	Approved

					AI pipeline is taking into consideration cloudy scenes and scenes with missing data. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025. (Alena)	
4.	4.1/10			It might be interesting to take into account the energy consumption together with Performance & Accuracy especially when considering an update of model (a full rerun seems not optimal: it should be interesting to select only the product that will benefit most from the new algorithm).	Energy consumption metrics will be taken during the models training. A trade-off should be made between inference accuracy and performance. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025. (Alena)	Approved
5.	4.2/10			the storage of satellite images “are managed by another software in the system”. At this stage, it is worth mentioning the software’s name.	The “another software” is DASI, this will be corrected in the next issue of the document. (Patrick)	Approved
6.	4.2/10	Q		It is not clear if the STAC interface is used by SKIM to query remote catalogues or if it is the interface provided to users who use SKIM.	The SKIM interface is available to machines only: GCORE component and RUCIO, there is no interface with User. The sentences “The user search is handled by SKIM and can be easy (only image) or complex (object and event). The answer is an image.” are not applicable to DaFab system, will be cancelled in the next version of the document. (Patrick)	Approved
7.	4.3/11	m		It is not sufficiently clear what will be the interaction between SKIM and RUCIO regarding the metadata catalogue. The 4.2 paragraph may lack a sub paragraph “within the DaFab project, …” like the others..	RUCIO is gathering the Copernicus data and the new metadata. DASI is providing the Copernicus ones to RUCIO. SKIM is providing the new ones to RUCIO. All services interact with RUCIO using RUCIO’s REST API or its Python client library. This API allows parties to insert, update, or remove metadata associated with data	Approved

				identifiers, such as individual Sentinel image entries. This generation or update of metadata ensures that the unified metadata catalogue remains current. (Dimitris/Patrick)	
8.	4.4/12	m	It is not clear where DASI domain specific Metadata will be stored (in RUCIO MD catalogue, DASI internal one, etc.) ?	The original metadata will be stored in DASI's internal database, called catalogue. This will be ingested in RUCIO, and later access will be provided via DASI's RUCIO backend. (Metin)	Approved
9.	4.5/13	m	It is not clear what component is in charge to move Data across sites : is it DASI as explained in 4.4 or the orchestrator as explained in 5 ?	The workflow system will be responsible for moving data across. Results generated from workflow steps will be stored at the local site, and information about their location will be provided through Rucio. For performance purposes, the workflow system could move or create a copy of the data to a remote site and tell about its existence through Rucio. (Georgios)	Approved
10.	5.1/15	Q	Is HPK compatible with K8S auto scaling ?	Yes, HPK supports K8s auto-scaling. HPK can run all the native K8s pods that the auto scaler needs to operate. However, special permissions may be needed for the containers to access the machine performance counters (Georgios)	Approved
11.	6.3/20	Q	A sentinel 2 image may contain several thousands of parcels, some of them with complex geometries. Is there any evaluation of the size of this metadata ? Since this MD is updated once per growing season, is it planned to keep previous field boundaries (how many versions if applicable?) ?	From Agricultural system use case definition, the fields boundaries are stored in vector format. The size of this metadata is not assessed yet. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025 (Alena)	Approved

12.	6.3/22	Q	Same question on the size of raster MD (masks) ? Could it be a problem ? Is there a limit ?	From the Hydrologist system use case definition, the masks (for example water mask) are stored in raster format. The size of this metadata is not assessed yet. This topic is covered by WP2 activities, refer to D2.1 deliverable for Q1-2025 (Patrick - Alena)	Approved
13.	6.5/24	m	The combined use of SKIM and RUCIO catalogues is not sufficiently detailed, especially considering complex queries involving MD from both catalogues.	All the users queries are destinated to RUCIO, which hosts the unique catalogue. This unique catalogue gathers the Copernicus data and metadata and the new metadata generated by DaFab system (Patrick)	Approved
14.	general	Q	Considering the volume Copernicus dataset, it may be interesting to add some details on the scalability of the framework.	We are using HPC and Clouds (Jean-Thomas)	Approved
15.	general	Q	DaFab framework is distributed. How user authentication/permission are managed ?	The KNOT user-facing component allows users to deploy and execute their workflows in the DAFAB platform. KNOT allows users to log in and initiate their workflows. While KNOT already includes user authentication, it will be extended to meet the specific requirements of the DAFAB project, enhancing both its functionality and security. (Georgios)	Approved

Scalability of the framework (Jean-Thomas)

Regarding the scalability of DaFab approach, two metrics have to be considered. First the extra requirement for capacity, and the processing cost of generating the metadata.

Our current model generates a 1-bit mask for every pixel: if the pixel is part of a field border or not. The current size of a geopatch of 256x256 pixels (65 K pixels) is 1MB, thus 16B or 128 bits per pixel. The output of the model is a mask, 1 bit per pixel. Consequently, the total volume of raw metadata is ($^1/_{128}$)th of the data size. The total volume of data hosted in the Copernicus database is in the range of 45PB, Thus DaFab will generate an additional 350TB of metadata. This may sound considerable, however the mask can be compressed as a very high ratio, either with standard image compression ratio (RLE) or with more complex transformation such as polygon extraction out of the mask. Therefore we only expect a marginal capacity increase (few TBs at most). The size of the model for mask generation is in the range of 650 MB which is negligible in respect of the data volume we are addressing.
The second aspect of scalability is the computational overhead of generating the mask. This aspect is linked to WP2 T2.1 which is assessing the cost-performance of different hardware solutions. Our model running on a single core CPU node processes 8 images per second for the training.