Research Lifecycle Management technologies for Earth Science Communities and Copernicus users in EOSC

Deliverable D6.4
Copernicus Data Pipelines Report

Grant agreement number	101017501
Start date of the project	Reliance
Duration of the project	24 months
Type of Action	Research and Innovation Action
Coordinator	PSNC

Due date of delivery	31/12/2021
Actual date of delivery	18/01/2022
Work package	WP6
Type of deliverable	Report
Dissemination level	Public
Responsible	Terradue
Reviewers	Simone Mantovani (MEEO) Oscar Corcho (UPM)
Version	1.0

This project has received funding from the European research infrastructures (including e-Infrastructures) *under the European Union's Horizon 2020 research and innovation programme* under grant agreement No 101017501

List of authors, contributors and reviewers

Name	Role	Organisation
Pedro Goncalves	Author	Terradue
Herve Caumont	Author	Terradue
Manuela Sagona	Contributor	Terradue
Simone Vacari	Contributor	Terradue
Simone Mantovani	Reviewer	MEEO
Oscar Corcho	Reviewer	UPM

History of changes

Version	Date	Change	Authors	Organisation
0.1	26/03/21	ToC	Pedro Gonçalves	Terradue
0.2	03/05/21	Introduction and user stories	Pedro Gonçalves	Terradue
0.3	06/09/21	Application Package	Herve Caumont	Terradue
0.4	16/11/21	Data Pipelines	Manuela Sagona	Terradue
0.5	01/12/21	Use Cases	Pedro Gonçalves	Terradue
0.9	14/01/22	Revision and Typos	Fabrice Brito	Terradue
	16/01/22	Review	Simone Mantovani	MEEO
	16/01/22	Review	Oscar Corcho	UPM
1.0	18/01/22	Review	Pedro Gonçalves	Terradue

Glossary

Acronym	Explanation
API	Application Programming Interface
B2B	Business-to-business
CEOS	Committee on Earth Observation Satellites
CLI	Command-Line Interface
COG	Cloud Optimized GeoTIFF
CWL	Common Workflow Language
DIAS	Data and Information Access Services
DInSAR	Differential Synthetic Aperture Radar Interferometry
EO	Earth Observation
EOEPCA	EO Exploitation Platform Common Architecture
EOSC	European Open Science Cloud
ESA	European Space Agency
GDAL	Geospatial Data Abstraction Library
GSNL	GEO Geohazard Supersites and Natural Laboratory
HR	High resolution;
InSAR	Interferometric synthetic aperture radar
JSON	JavaScript Object Notation
OGC	Open Geospatial Consortium
PaaS	Platform-as-a-Service
RO	Research objects
SAR	Synthetic Aperture Radar
SLC	Single Look Complex
SSO	Sun-synchronous orbit
STAC	SpatioTemporal Asset Catalog
VHR	Very-High Resolution
VM	Virtual Machine
WPS	Web Processing Service

Table of Contents

1 Executive Summary

RELIANCE, short for Research Lifecycle Management for Earth Science Communities and Copernicus users in EOSC, aims to realise the vision of FAIR research in EOSC by adopting a holistic research management approach based on three key and complementary technologies: Research objects (RO), Data cubes and Text mining. The services that RELIANCE will integrate into EOSC include:

- ROHub platform providing Research Object management functionalities
- ADAM platform providing Data Cubes management functionalities
- Text Mining services providing functionalities for information extraction and semantic annotation, content-based retrieval and recommendation, as well as extended analytics services in support of the scientific enterprises
- Copernicus Data Pipelines supporting the research community accessing and processing Earth Observation (EO) data to create new information layers

In this document we will present the Copernicus Earth Observation Data Pipeline as an integrated intermediate service layer for the systematic execution of Earth Observation Applications continuously delivering data and information to users. A data pipeline is the solution provided to a specific data challenge defined by a researcher as a tailored data processing workflow responsible for information extraction from a wide range of large volume Earth observation data sources that are executed within defined spatial and temporal intervals.

The Data Pipeline uses shared data processing infrastructures from EOSC where the Earth Observation applications are deployed, providing a hybrid and reliable Cloud infrastructure for the applications needs. The service deployment follows the best practice to package and deploy applications as reproducible, deployable and executable in different platforms. Terradue Ellip Platform manages the underlying Cloud resources provided from EOSC, orchestrates the execution of the triggered Data Pipelines and publishes the results.

This report documents the design of Copernicus Data Pipelines and the implementation and deployment activities performed responding to RELIANCE users challenges. During this 1st year of the project, the activities focused on INGV's *Grand Challenge 4* preparing the response to *Scenario 1* (analysing the consequences of large volcanic eruptions) and *Scenario 2* (developing models and creating a new specific workflow to analyse SAR data).

Through the activities performed we aim to demonstrate that EOSC can offer an alternative integrated virtual environment to deposit, share and reuse EO data and could provide the storage and computing resources necessary to manipulate and manage large amounts of Copernicus data. EOSC could make it easier to retrieve and process EO data, and thus stimulate demand for EO data research and services.

2 Introduction

2.1 Scope

Earth Observation (EO) refers to the use of remote sensing technologies to monitor land, marine (seas, rivers, lakes) and atmosphere. Satellite-based EO gathers imaging data through satellite-mounted payloads that are then processed and analysed in order to extract different types of information that can be obtained from optical, thermal or radar sensors. Copernicus is the European Union's EO programme designed to meet the needs for space-derived, timely and accurate geospatial information.

To integrate Copernicus data and services into the service catalogue of the European Open Science Cloud (EOSC) the RELIANCE project will deploy Copernicus Earth Observation Data Pipelines as an integrated intermediate service layer for the systematic execution of Earth Observation Applications continuously delivering data and information to different research users.

This report documents the implementation and deployment of Copernicus Data Pipelines for RELIANCE users provided by Terradue's Ellip[1] Platform. The deployment onto Infrastructure as a Service (IaaS) providers from EOSC and data production is managed by Ellip that orchestrates the deployment and execution of the triggered Data Pipelines and publishes the results.

2.2 Audience

This deliverable is intended for internal use by the RELIANCE Consortium, although it may be valuable to external stakeholders, including other EOSC related projects who are also dealing with the integration and deployment of their Earth observation applications services into the EOSC.

2.3 Structure

The rest of the document is structured as follows:

- Section 3 provides a background overview of the Earth Observation and the Copernicus Programme and how the Data Pipelines provides an integrated EO intermediate service layer for Data Access and Processing.
- Section 4 provides an overview of Earth Observation Exploitation Platforms use cases and architecture and how Ellip, a Platform-as-a-Service (PaaS) work environment, is made available to RELIANCE users to setup Copernicus data pipelines together guaranteeing portability, scalability and interoperability in EOSC-powered Cloud Computing environments.
- Section 5 provides an overview of Earth Observation Applications design rules and how the applications are packaged and provided as container images available for deployment.
- Section 6 describes the implementation actions taken in the 1st year of the project. It describes the implemented use cases, data sourcing and Earth observation applications selected.

[1] https://ellip.terradue.com/

3 Background

3.1 Earth Observation

Earth Observation (EO) refers to the use of remote sensing technologies to monitor land, marine (seas, rivers, lakes) and atmosphere. Satellite-based EO gathers imaging data through satellite-mounted payloads that are then processed and analysed in order to extract different types of information. These payloads can contain optical, thermal and radar sensors.

The optical and thermal sensors monitor the energy received from the Earth due to the reflection and re-emission of the Sun's energy by the Earth's surface or atmosphere. They operate between the visible and infrared wavelengths of the electromagnetic spectrum. The spectral resolution of optical and thermal sensors defines the ability of a sensor to define fine wavelength intervals. The finer the spectral resolution, the narrower the wavelength range for a particular channel or band enabling some applications that require the ability to analyse specific wavelengths.

The radar sensors are payloads operating in the lower part of the spectrum (longer wavelengths) and most of them send energy to Earth and monitor the energy received back from the Earth's surface or atmosphere, enabling day and night monitoring during all weather conditions. Radar sensors are generically of three main types: synthetic aperture radar (SAR) for high-resolution imaging, a radar altimeter, to measure the ocean topography and a wind scatterometers to measure wind speed and direction

Another important differentiator of EO sensors are the spatial and temporal resolutions. The spatial resolution defines the size of the pixel captured by the sensor and can be divided in three broad categories: low and medium (more than 10 metres) , high (between 1 and 10 metres) and very high (less than 1 metre)[2]. The temporal resolution defines the frequency of data acquisition for the same area and varies substantially ranging from once an hour for geostationary orbit satellites to once every few days for sun-synchronous orbit satellites.

3.2 Copernicus Programme

Copernicus is the European Union's EO programme designed to meet the needs for space-derived, timely and accurate geospatial information in support of policy making and laying the foundations for various private-sector services and applications.

Since its inception in 1998 (then called 'GMES'), the Copernicus programme provides a long-term commitment to "the development of space-based environmental monitoring services, making use of, and further developing, European skills, and technologies'' and is now a highly successful example of European cooperation. There are currently seven missions under the Sentinel programme (Sentinel 1, 2, 3, 4, 5P, 5, 6) with the following objectives:

- Sentinel-1 provides all-weather, day and night radar imaging for land and ocean services in a sun synchronous, near-polar orbit.

[2] The use of Very High Resolution (VHR) imagery with less than 0.3 metres is subject to restrictions that limit it to government applications.

- Sentinel-2 provides high-resolution optical imaging for land services (e.g. imagery of vegetation, soil and water cover, inland waterways and coastal areas) in a sun synchronous orbit.
- Sentinel-3 provides ocean and global land monitoring services in a sun synchronous orbit .
- Sentinel-4 provides data for atmospheric composition monitoring in a geostationary orbit,
- Sentinel-5 (together with Sentinel-5 Precursor) provides data for atmospheric composition monitoring in a sun-synchronous orbit
- Sentinel-6: high precision altimetry sea level measurements with a radar altimeter and a microwave radiometer.

Each Sentinel mission is based on a constellation of two satellites to fulfil and revisit the coverage requirements for each mission, providing robust datasets for a wide range of applications, covering land, marine and atmosphere monitoring.

Copernicus also has six thematic services (Land, Marine, Atmosphere, Climate, Emergency and Security) supporting the development of many applications. The Copernicus services process and analyse the data, integrate it with other sources, offer Geo-Information Systems (GIS) products to their users, and serve public authorities and commercial businesses.

3.3 Data Access and Processing

Researchers using EO data are present in many domains, each with very specific needs, and the raw data coming from Copernicus missions cannot be exploited directly. Specific EO applications derive information contained in the images and act as the interface between the satellite technical features and the end users' specific needs. Researchers need the access to the data but also the storage and computing resources to enable their data analysis.

In 2017, the European Commission launched an initiative to develop the Copernicus Data and Information Access Services (DIAS) that facilitate access to the data of the Sentinel missions and information from the Copernicus services. Four consortia (CREODIAS, ONDA, Mundi, Sobloo) were chosen to set up DIAS computing environments under ESA management, and a fifth consortium (WekEO) is managed by EUMETSAT, ECMWF and Mercator Ocean International (MOI).

The DIAS represented an opportunity to federate the access to the Copernicus data and information close to processing facilities allowing further value extraction from the data, and respond through a dedicated service approach that is complementary to traditional data downloading. The DIAS targeted access to Copernicus data and information close to processing facilities and, through this, created the possibility to easily build applications and offer added-value services.

Introduced during the Baveno+20 conference in June 2018, the five DIAS have entered their operational phases, acquiring customers and developing their business in a commercial environment. Nevertheless, after the end of the procurement contracts, signed with the European Space Agency by the consortia (in July 2021), the operational continuity of the platforms and established services, customer relations and innovative solutions might be at risk[3].

[3] *Industry view of the future of the Copernicus programme: key issues to address*. Position Paper from the European Earth Observation Services Industry, EARSC, November 2019

Furthermore, since the second half of 2020, Sentinel data have been gradually moved from on-line rolling cache to long-term storage. For the analysis of large time-series, like the Differential Synthetic Aperture Radar Interferometry (DInSAR), this involved the implementation of a two-steps approach where users need first to order the data for the cold archive and wait for the data availability to initiate the processing. As no data retention policy is guaranteed, and with stringent quotas applied (e.g. on ONDA DIAS max 20 products per hour), it has become problematic for processing large time series for applications that need the full stack of products to be online and available to initiate the processing .

Integrating Copernicus data and services into the service catalogue of the European Open Science Cloud (EOSC) provides researchers with an alternative path[4]. By federating existing research infrastructures and scientific clouds the EOSC offers an open pan-European virtual environment to access, store, analyse and re-use data and can support the development of cloud-based services for open science and provide the computing resources necessary to manipulate and manage large amounts of Copernicus data.

In the following sections, we will show how RELIANCE is integrating the Copernicus Data Pipelines in EOSC and demonstrate how the availability of the free and open data Copernicus Sentinel products, together with the availability of EOSC computing resources, creates an opportunity for the wide adoption and use of EO data in a growing number of research fields.

3.4 Data Pipelines

Copernicus EO Data Pipelines is an integrated EO intermediate service layer for the systematic execution of EO Applications to continuously deliver data and information to different users, complying with their specifications for information content and format.

In its essence, a data pipeline is the solution provided to a specific data challenge defined by a researcher. It is a tailored data processing workflow responsible for information extraction from a wide range of large volume data sources that are executed within defined spatial and temporal intervals and publish the generated results. An example of a data pipeline is, for instance, the continuous production of orthorectified and atmospherically corrected Sentinel 2 vegetation indexes or providing Sentinel-1 based land deformation or change detection maps for a given area.

The EO Applications executed in the Data Pipeline are specific data processing functions defined by the researcher that perform data operations like processing / reprocessing, projection, visualisation or analysis. The applications can be written in a variety of coding languages (e.g. Python, R, Java, C++, C#) and make use of specific software libraries (e.g. SNAP, GDAL, Orfeo Toolbox). Each data processing function is an application (e.g. a command line tool) that is a non-interactive executable program that reads some input (or a set of inputs organised in an atomic unit), performs a computation, and terminates after producing some output. Each application is released, packaged and built to be deployed in a IaaS provider ready to process inputs of the data pipeline.

[4] Open science and Copernicus: a story of mutual support. Augusto Burgueño Arjona, June 2018 https://ec.europa.eu/digital-single-market/en/blogposts/open-science-and-copernicus-story-mutual-support

The Data Pipeline uses shared data processing infrastructures where the applications are deployed, providing a hybrid and reliable Cloud infrastructure for the applications needs. The service manages the IaaS and Cloud resources, orchestrates the execution of the triggered Data Pipelines and publishes the results. Working with Cloud-hosted data, the data pipeline provides fast access to the data sources and processing resources. Together with the availability of interfaces, APIs and tools for accessing and storing the data, Cloud computing provides the on-demand delivery of computing power, servers, databases, networking, software, analytics and other resources that support the data pipeline operations.

The Data Pipelines are basically triggered in two modes: Event Driven and Data Driven. The data driven approach queries an associated EO catalogue and retrieves the list of data products to be processed. In the Event Driven approach, there is an external service that announces an event (e.g. earthquakes, vulcanos) and generates a spatial & temporal domain of execution. For the data stage-in the input products are retrieved and made available as inputs for the local processing of the EO Application. For the data stage-out, the outputs generated by the processing are retrieved and automatically published onto an external persistent storage together with the generated metadata.

According to the need identified by the researchers, the stream of outputs of a data pipeline are delivered in an agreed format (e.g. CoG, zarray) and ready to be consumed through a Jupyter Notebook or to be ingested in a dedicated data cube.

4 Earth Observation Exploitation Platform

Earth Observation (EO) Exploitation Platforms are collaborative, virtual work environments providing access to EO and value-added products together with the tools, processors, and ICT resources required to work with them, through one coherent set of interfaces. The fundamental principle of these platforms' operations concept is to move the users to the data and tools. Users access a platform work environment providing the data, tools, and resources required, as opposed to downloading, replicating, and exploiting data 'at home'.

With the support of the European Space Agency (ESA) EO Exploitation Platforms' initiative, a platform-based ecosystem providing infrastructure, data, compute and software as a service started to emerge and defined an EO Exploitation Platform Common Architecture (EOEPCA). The resulting Exploitation Platforms are a collaborative environment where scientific and value adding activities are conducted, to generate targeted outputs for end-users.

As part of that ecosystem, Ellip, a Platform-as-a-Service (PaaS) work environment operated by Terradue, supports the Earth Science and Services Communities to exploit Earth observations from satellites directly on the Cloud. The Ellip Studio service offers an environment for developing Cloud-ready Earth Observation Applications tailored for reproducible and portable Earth Observation (EO) application packages accessing large EO data collections respecting the FAIR principles.

In RELIANCE, Ellip is made available to users with the goal to help them develop and deploy Copernicus data pipelines together with the necessary EO data processing services guaranteeing portability, scalability and interoperability in EOSC Cloud Computing environments.

4.1 Use Cases

In RELIANCE, users have different expectations and goals when addressing an EO Exploitation Platform that can be addressed through the definition of scientific researchers with two main roles: developers and consumers.

On the one hand, the consumers want to access existing and available services within the platform to acquire new knowledge. This knowledge may be consumed immediately or delivered to external downstream services that may be scientific, research oriented or commercial in nature. For example, a user in RELIANCE in Grand Challenge 4 analysing the consequences of large volcanic eruptions (Scenario 1) will want to execute specific existing services to analyse the surface deformations at active volcanoes that will process SAR (Synthetic Aperture Radar) images with interferometry to obtain multi-temporal velocity maps of the surface deformation for large areas with millimetric precision.

On the other hand, service developers aim at integrating an existing algorithm or application, improving a workflow or developing a new data processing workflow. For example, when the user is developing models of a volcanic plumbing system (Scenario 2) it creates a new specific workflow to analyse SAR data and produce a new information layer.

Both actors may be experts in the domain, or require specific filters and visualisation tools in order to acquire and consume the information obtained. Most importantly a service developer may be also a consumer of a service and vice-versa.

In this section we illustrate several of the use cases targeted by the EO Exploitation Platform taking in consideration those two main actors.

4.1.1 Developer builds new processing service

As a developer, I want to integrate, into the platform, my own software, written in a specific programming language as a containerized application (or a set of containerized applications), to be exposed as a new processing service. To achieve this I will follow the deploy, test, validate and publish steps described as :

- Prepare one or more container images containing the execution dependencies of the software
- Prepare the application package manifest orchestrating the container(s)
- Test the application package by deploying and executing it in a hosted test environment (with access to platform data for testing).
- Publish the application package as a new processing service, supported by ancillary information including metadata and processor user manual.
- The new processing service should be discoverable in the platform and made available to a community of Consumers according to specific access rights.

4.1.2 Developer builds new processing service chaining applications

As a developer, I want to combine and/or chain multiple processing services, potentially offered by different developers/organisations, in parallel or sequentially. To achieve this I will follow the workflow design, test, validate and publish steps described as:

- Prepare a self-contained application package including the workflow execution model and sequence by defining relations of the application packages, and the input/output parameterisation of each step in the chain.
- Test the application package by deploying and executing it in a hosted test environment (with access to platform data for testing).
- Publish the application package as a new processing service, supported by ancillary information including metadata and processor user manual.
- The new processing service should be discoverable in the platform and made available to a community of Consumers according to specific access rights.

4.1.3 Consumer discovers and visualises EO and value-added products

As a Consumer I want to search for EO and value-added products of a specific type by specification of textual and faceted search criteria and visualise the search results. To achieve this I will follow the discovery, listing and bookmarking steps described as:

- List the EO and value-added products collections available on the Platform.
- Select one or more collections to search.
- Define search parameters for time and geospatial domains.
- Refine the search results with additional metadata parameters restrictions.

- Select individual products and obtain detailed metadata for the product including, if available, an image preview.
- Bookmark or store the references of the selected products.

4.1.4 Consumer discovers and executes Processing Service

As a Consumer I want to search and select an on-demand Processing Service to execute with specific input parameters. To achieve this I will follow the discovery, submission, monitor and result visualisation/publication steps described as :

- Search the on-demand Processing Services available on the Platform
- Retrieve Processing Service complementary information (e.g. description, processing requirements and costs, terms and conditions, licence).
- Define the Processing Service input parameters with access to data catalogues to discover compatible data.
- Optionally obtain an estimation of the execution costs and/or time based on the input parameters,
- Monitor the Processing Service status upon submission.
- Access and visualise results obtained.
- Share the results publicly or within a given community (link to the previous use case).

4.1.5 Consumer orders systematic processing

As a Consumer, I want to select a Processing Service to systematically execute as a data pipeline that follows specific metadata parameters (e.g. geographic, sensor) supported by schedulers for time-driven, data-driven flows or triggered by external events (e.g. earthquake). At the end, or during the processing time frame, I want to access data already processed formatted to allow immediate analysis, with minimal additional effort, and major interoperability with other datasets. The systematic processing can be automatically ordered or follow an ad-hoc ordering process. Optionally, the processing data results may be saved, visualised on the platform or published in an external system.

For example, a Consumer scenario might be to select earthquakes of a certain nature (magnitude, depth or tensor characteristics) to systematically trigger several processing services.

4.2 Architecture

The EO Exploitation Platform responds to the identified use cases by focusing on a scenario where an Application is directly packaged as an Application Package, registered in a Platform and made available as an OGC API — Processes Web Service.

The Web Service allows end-user portals and B2B client applications to pass input products, processing parameters, trigger on-demand or systematic data processing requests and establish the data pipeline to retrieve the information produced as described in the Figure 1.

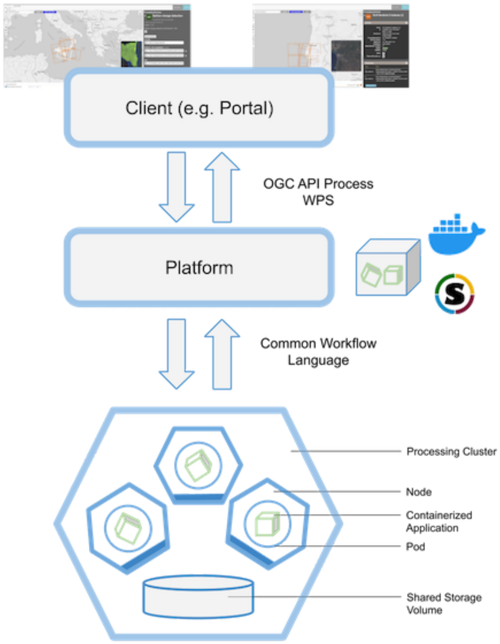


Figure 1- Architecture overview of the Application execution in a Platform

The main responsibilities of the Platform are to:

- Provide the discovery of EO products (OpenSearch and SpatioTemporal Asset Catalog)
- Validate and accept an application deployment request (OGC API Processes Transaction Extension)
- Validate and accept an execution request (OGC API Processes)
- Submit the process execution to the processing cluster (application package encoded as a CWL description document)
- Monitor the process execution (OGC API Processes)
- Retrieve the processing results (OGC API Processes and OpenSearch)

The Exploitation Platform architecture follows the best practice to package and deploy Earth Observation applications as defined in OGC 20-089. The following sections describe in high level the main interfaces that the platforms provide for the data discovery, processing service deployment and execution.

4.2.1 Data Discovery Interfaces

Main Specifications : OpenSearch plus Geo, Time and EO Extensions, SpatioTemporal Asset Catalog API

The Exploitation Platform catalogues both Data and Processing Services metadata and allows complex queries using OpenSearch and/or the SpatioTemporal Asset Catalog (STAC) API to expose product discovery capability with several parameters to filter the resources based on textual, spatial, temporal, or other characteristics.

The catalogue provides a search engine capable of dealing with different types of queries (Geographic and non-Geographic) and a distributed interface with diverse metadata search capabilities together with online access points with multiple access protocols. It provides a framework to ease the discovery of EO data (remote-sensing and in situ) using best practises for search services as defined by the Committee on Earth Observation Satellites (CEOS).

Both Opensearch and STAC specifications standardise the way geospatial assets are exposed online and queried in a interchangeable fashion but the STAC specification adds the possibility to drill down to products assets level (e.g. JPG2000 band file, auxiliary data, browse).

4.2.2 Service Integration

Main Specifications : Common Workflow Language, SpatioTemporal Asset Catalog

In a Exploitation Platform the user (e.g. a Developer) is able to integrate their own processing services and make them available for exploitation by other users also individually or in their own workflows. To support their development activities, users are provided with a development environment where they can integrate, build, test & debug and deploy.

The application integration environment considers several levels of data processing integration:

- Importing: The service/application is directly packaged as a black-box. It relies upon the stage-in/out of data to the applications existing data access expectations by the Processing Framework.
- Adapting: The service/application is adapted to use the data access interfaces for data input and output.
- Porting: The service/application is ported to use the platform services by, for example, using the Client Library defined by the common architecture.

The overall integration supports developers that want to adapt and package their existing algorithms written in a specific language to be reproducible, deployed and executable in different platforms. The developers create Application Packages encoded in Common Workflow Language (CWL) documents that describe the data processing applications, providing information about the parameters, software item, executable, dependencies and metadata.

Developers build one or more container images with their Application and command line tool(s) and respective runtime environments, publish the container image on a repository and write the Application Package document with a workflow that invokes the command line tool(s) included in the image.

For the staging of input and output EO products during the service execution the data flow management can use a local catalogue encoded using the SpatioTemporal Asset Catalog (STAC) specification as a data manifest for application inputs and outputs. The local catalogue provides knowledge about the input and output files data contents like spatial footprint, sub-items (e.g. masks, bands) and additional metadata.

Aside from the role of the CWL in packaging existing applications, the Application Package in itself can orchestrate the execution of a workflow of several pre-packaged applications. In this scenario the Application Package combines and/or chains multiple processing services, potentially offered by different developers/organisations, in parallel or sequentially.

4.2.3 Service Execution

Main Specification : OGC API Processes or OGC Web Processing Service (WPS)

The OGC API - Processes specification is the ultimely endpoint for the user (e.g. Consumer) application client (e.g. portal, command line, Jupyter Notebook) and defines how the client application can request the execution of a process, how the inputs to that process can be provided, and how the output from the process is handled. The specification allows for the wrapping of computational tasks into an executable process that can be invoked by a client application.

When invoking the service, the request includes the reference to the input data for processing and the Platform handles the data stage-in and stage-out across the workflow steps.

The data can be listed by a direct URL reference to the data products (e.g. File, WCS, WFS) or by resolving the search parameters (i.e. OpenSearch query parameters) to a specific catalogue representing a data package of a given collection within a given area and temporal of interest and filtered metadata. All processing steps are monitored and assessed by the OGC API - Processes operations for status and results.

The service execution has two main operating modes: on-demand, in which users specify settings manually, or load previously saved settings, and scheduled automatic execution of the processing chains based on saved settings (including monitoring and failure-notification) for creating a data pipeline. Both modes must provide efficient access to processing outputs and options for repatriating or disseminating results and the Platform needs the technical capability to distribute information through the prioritised distribution channels.

5 Earth Observation Applications

A Copernicus Data Pipeline is fed by an Application executed as a command-line interface (CLI) tool that runs as a non-interactive executable program: it receives input arguments, performs a computation, and terminates after producing some output.

Earth Observation Applications typically offer functions that perform data operations like processing / reprocessing, projection, visualisation or analysis. The applications can be written in a variety of coding languages (e.g. Python, R, Java, C++, C#, shell scripts) and make use of specific software libraries (e.g. SNAP, GDAL, Orfeo Toolbox). In overall, in the context of the Copernicus Data Pipeline, the EO application needs to be:

- Executable as a command-line tool.
- Delivered in a container image with all the necessary software, libraries and configuration files.

The application is treated as a black-box that, according to its application design pattern, must comply with data stage-in and data stage-out mechanisms defined by the platform. Two main design patterns are identified: fan-in and fan-out. An application can combine these patterns in the nodes of a Directed Acyclic Graph (DAG).

5.1 Data-driven application with a fan-in application pattern

The data driven application fan-in pattern refers to the execution of a data processing function that aggregates several input products. The application accesses a list of input products, stages the input products making them available to the application execution block.

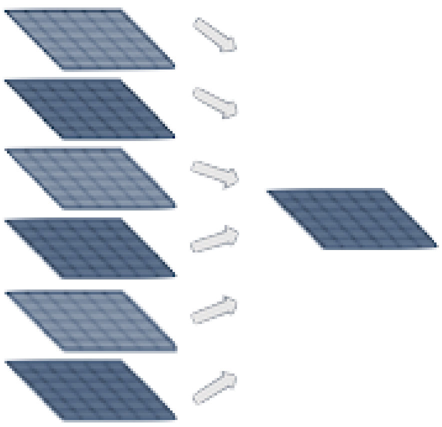


Figure 2 — Data-driven application with fan-in input references where an application processes the aggregates of n-input EO products

An application following this pattern must take in consideration that it will be invoked once with all the input products and is expected to create one output (but not necessarily a single file).

5.2 Data-driven application with a fan-out application pattern

The data driven application fan-out pattern refers to the execution of a data processing function that processes concurrently several products generating independent output for each input.

The application loops from a list of input products, stages each of the individual products making it available to the application execution block. The platform can apply different strategies to parallelize the execution of each individual product.

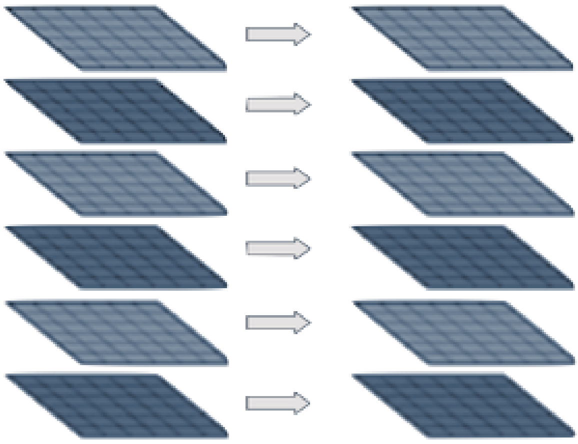


Figure 3 — Data-driven application with fan-out input references where an application processes several input EO products independently.

5.3 Staging Input and Output EO Products

EO product files come in different formats (e.g. GeoTIFF, HDF5, SAFE) and might have sub-items (e.g. metadata, bands, masks) that can be encoded in the same file or follow a given folder structure.

For example, SENTINEL-2 products are made available to users in the SENTINEL-SAFE format, including image data in JPEG2000 format, quality indicators (e.g. defective pixels mask), auxiliary data and metadata. The SAFE format wraps a folder containing image data in a binary data format and product metadata in XML. A SENTINEL-2 product refers to a directory folder that contains a collection of information that can include several files like seen in the next figure.

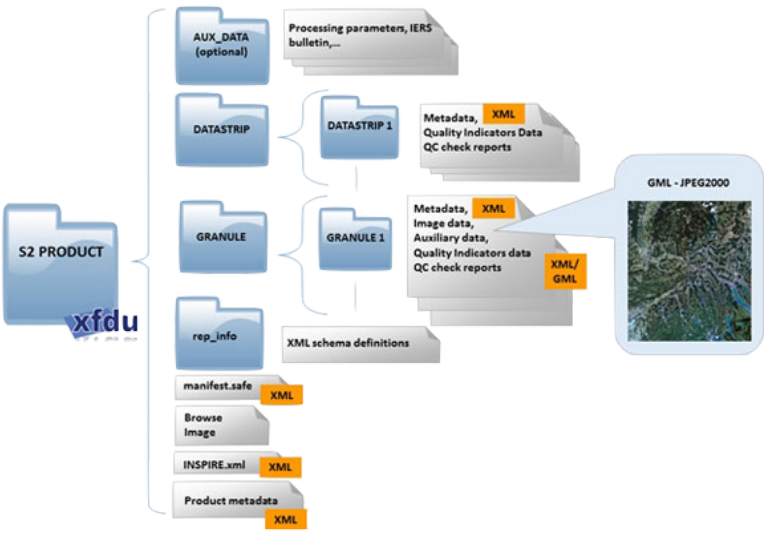


Figure 4 — SENTINEL-2 product physical format

A main concern application developers face is the different approaches through which the products are made available (i.e. stage-in) to the applications. For example, applications might find the same exact folder structure and return the folder root or the main XML manifest file or have the folder structure compressed in a single archive file.

In general, the onus of navigating the input folder directory and programmatically reacting to how the file was staged-in by the platform is on application and the application developer needs to consider all possible cases when developing their read routines.

Conversely, the outputs of the application are fully managed by the developer that places the resulting files in an output directory. The only information the platform might receive about the output files is the file media type (formerly known as "MIME-type") and is often missing critical information like spatial footprint, sub-items (e.g. masks, bands) and additional metadata (e.g. ground sample distance, orbit direction).

The solution chosen to represent the data manifest for input and output products is brought by the SpatioTemporal Asset Catalog (STAC). The STAC specification standardised the way geospatial assets are exposed online and queried. A 'spatiotemporal asset' is any file that represents information about the earth captured in a certain space and time (e.g. satellites, planes, drones, balloons).

The STAC specification defines several objects:

- STAC *Catalog*: STAC *Catalog* object is a collection of STAC *Item* objects or other STAC *Catalog* objects (sub-catalogues). The division of sub-catalogues is transparently managed by links to ease online browsing.
- STAC *Collection*: extends the STAC *Catalog* object with additional fields to describe a whole set of STAC *Item* objects that share properties and metadata. STAC *Collection* objects are meant to be compatible with OGC API — Features Collections (OGC 17-069r3).
- STAC *Item*: a GeoJSON Feature with additional fields (e.g. time, geo), links to related entities and STAC *Asset* objects.
- STAC *Asset*: is an object that contains a link to data associated with the STAC *Item* that can be downloaded or streamed (e.g. data, metadata, thumbnails) and can contain additional metadata. Similar to atom:link it has properties like href, title, description, type and roles; but, most significantly, it allows relative paths.

Most importantly the STAC specification can be implemented in a completely 'static' manner as flat local files located near the data enabling the application to access products assets (e.g. JPEG 2000 band file, auxiliary data, browse) with a relative path (something that was not possible using OpenSearch as defined by OGC 13-026r8, OGC 13-032r8).

Using a STAC *Catalog* object with STAC *Item* objects as the data manifests format for applications that require staging input data and/or output results allows the development of applications that are portable among different platforms.

5.4 Application Package

The Application Package is a document that describes the data processing application by providing information about the parameters, software item, executable, dependencies and metadata. This file document ensures that the application is fully portable among all supporting processing scenarios and supports automatic deployment in a Machine — To — Machine (M2M) scenario. Most importantly, the Application Package information model can also allow the deployment of the application as an OGC API — Processes [OGC 18-062] compliant web service.

The Application Package includes the following information:

- Reference to the executable block that implements the Application functionality
- Description of its input/output interface

The Application Package uses the Common Workflow Language (CWL) Workflow Description specification as an encoding to describe the Application, its parameters, command-line tools, their runtime environments, their arguments and their invocation within containers. The CWL is a set of open standards for describing analysis workflows and tools in a way that makes them portable and scalable across a variety of software and hardware environments, from workstations to cluster, cloud, and high-performance computing (HPC) environments.

The CWL targets data-intensive processing scenarios and makes these portable and scalable across platforms capable of interpreting and execute the processes by describing:

- A runtime environment
- A Workflow (Directed Acyclic Graph or “DAG”)
- Command line tool(s)
- Parameter of the process
- Inputs/outputs

The CWL contains two main specifications. The Command Line Tool Description Specification that specifies the document schema and execution semantics for wrapping and executing command line tools and the Workflow Description Specification that specifies the document schema and execution semantics for composing workflows from components such as command line tools and other workflows. The CWL file is able to reference the application container images and also allow the definitions of the Application parameters, input/output interface and the overall process offering parameters.

Each input to a command line tool has a name and a type (e.g., File, string) and developers are encouraged to include documentation and labels for all components. Metadata about the command line tool descriptions can contain well-defined hints or mandatory requirements such as which software container to use or how much compute resources are required (memory, number of CPU cores, disk space, and/or the maximum time or deadline to complete the step or entire workflow.)

The execution block (i.e. Application Artefact) describes the ‘software’ component that represents the execution unit in a specific container image to be executed or specific workflow script that can be invoked on the processor directly. Based on the context information provided with the processor, the execution block maps how the container image can be parameterized or tailored.

A container image is an immutable, static file containing the dependencies for the creation of a container. These dependencies may include a single executable binary file, system libraries, system tools, environment variables, and other required platform settings.

Overall, a container image describes a container environment whereas a container is an instance of that environment, run by a container engine (e.g. Docker Engine). It is possible to run multiple containers from the same image, and all of them will contain the same software and configuration, as specified in the image.

With the use of CWL Workflow Description Standard as encoding, the Application can also possibly yield several Application Packages that expose parameters and inputs in different flavours and execution patterns.

Several examples of the usage of CWL to package Earth Observation applications can be found in the Section 8 of the Best Practice for Earth Observation Application Package [OGC 20-089] published[5] by the OGC.

[5] https://docs.ogc.org/bp/20-089r1.html#toc24

6 Implementation

6.1 Use cases

During this 1st year of the projectm, the activities focused on INGV's *Grand Challenge 4* preparing the environment and capacity for analysing the consequences of large volcanic eruptions response (covering *Scenario 1*) and developing models and creating a new specific workflow to analyse SAR data (covering *Scenario 2*). To achieve these objectives, it was identified the need to provide data pipelines accessing long time series of Copernicus Data for the GEO Geohazard Supersites (represented in RELIANCE by the INGV).

6.1.1 Geohazard Supersites

The GEO Geohazard Supersites and Natural Laboratory initiative[6] (GSNL) is a voluntary international partnership aiming to improve, through an Open Science approach, geophysical scientific research and geohazard assessment in support of Disaster Risk Reduction. It is focused in different regions as shown in Figure 5.

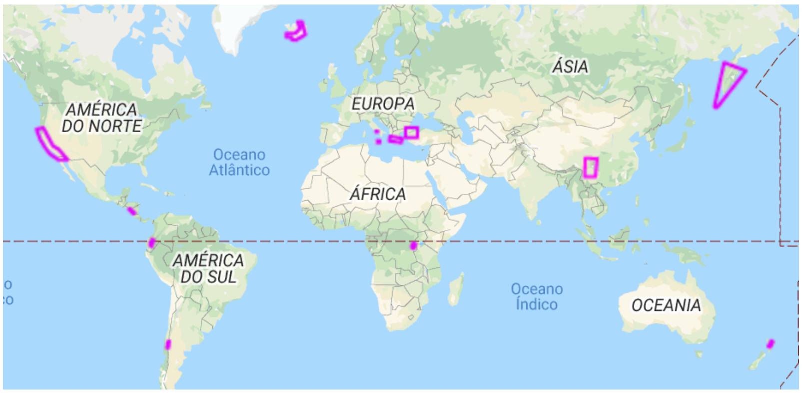


Figure 5 - GEO Geohazard Supersites and Natural Laboratory Areas of Interest

The data requirements are focused on the Copernicus Sentinel-1 SLC Products[7] (S1A + S1B Full mission) crossing the Supersites locations and we calculated the storage size needed at 358TB (with a subsequent 78TB yearly volume update).

This section will be further updated in the next version of this document.

6.2 Data Sourcing

As we saw, the EO Data Pipelines are basically the scheduled execution of EO Applications, where the systematic processing is fed by EO data products, be it in a data time driven or data driven, backward or forward looking. While Open Data policies are becoming more commonplace for the Earth observation domain, some technical challenges still remain to efficiently store, curate and serve such datasets to be deployed as Data Pipelines. Also, user applications might have to compose with multiple data sources having different types of dissemination and exploitation policies, and user support must be provided in this perspective.

[6] https://geo-gsnl.org/
[7] https://sentinel.esa.int/web/sentinel/technical-guides/sentinel-1-sar/products-algorithms/level-1-algorithms/single-look-complex

Nevertheless, as the Sentinel data moves from online to ordering mechanism it is necessary to deploy Virtual Archives where the data access is provided directly from the infrastructure (mirrored), or it can be fetched and cached in the infrastructure with adjustable time window and caching policy, or it can be routed directly to a remote data provider facility. The approach also includes a data preparation feature that may increase the internal data throughput, avoiding bottlenecks by staging the data for the tasked processes.

To respond to use cases we started the ingestion of Sentinel-1 data (potentially extended to Sentinel-2, Sentinel-3 and Landsat-8 in the future). We conceived it to support two basic scenarios:

- opportunistic - where data ingestion is (manually) triggered by the user in response to a specific need / event
- systematic - where data ingestion is performed automatically for a given community / stakeholder on top of a data hosting framework.

To respond to the immediate data storage needs of the use cases we established a collaboration with the DICE-EOSC[8] project established cache for specific areas identified by the GEO Geohazard Supersites. This will allow us to access the full time series of Sentinel-1 Data (SLC) products for those locations (initially ~ 320 TB with a subsequent ~70 TB yearly volume update) through object storage accessible via HTTPS and compatible with S3.

The following table enumerates the different Supersites, their geographic area and approximative number of Sentinel-1 SLC products that are currently being cached at CESNET.

Table 1 - Number of Sentinel-1 SLC products for the GSNL Supersites

GSNL Name	Number of Products	Yearly Update	Area of Interest (WKT)
China	~6700	~1360	POLYGON ((99.63 25.5, 104.02 25.44, 104.76 31.91, 99.92 32.03, 99.63 25.5))
Ecuadorian Volcanoes	~2200	~420	POLYGON ((-78.21 0.52, -78.37 0.41, -78.66 -0.08, -78.9 -0.63, -78.96 -0.88, -78.99 -1.4, -78.94 -1.78, -78.43 -2.09, -78.22 -2.1, -78.32 -1.81, -78.28 -1.29, -78.09 -0.82, -77.7 -0.28, -77.6 -0.07, -77.54 0.36, -77.49 0.54, -77.68 0.65, -77.65 0.68, -77.96 0.85, -78.21 0.52))
EnCeladus Hellenic	~5800	~1080	POLYGON ((19.28 37.82, 24.18 37.23, 24.58 38.64, 19.6 39.32, 19.28 37.82))
Hawaiian Volcanoes	~1800	~410	POLYGON ((-156.14 19.77, -155.98 19.06, -155.66 18.83, -154.67 19.51, -155.2 20.08, -155.94 20.37, -156.14 19.77))
Icelandic Volcanoes	~5400	~1060	POLYGON ((-22.99 63.7, -20.82 63.7, -20.78 63.2, -19.1 63.19, -15.2 64.2, -16.58 66.4, -18.22 66.35, -17.45 65.19, -19.35 64.4, -22.99 64.4, -22.99 63.7))
Kamchatka	~5900	~1280	POLYGON ((152.84 48.03, 154.06 47.74, 165.86 56.96, 156.9 58.7, 152.84 48.03))
Marmara Region	~8300	~1490	POLYGON ((26 39, 31 39, 31 42, 26 41.99, 26 39))
Mt Etna	~2200	~430	POLYGON ((14.5 37.3, 15.5 37.3, 15.5 38, 14.5 38, 14.5 37.3))

[8] https://www.dice-eosc.eu/

New Zealand Volcanoes	~2000	~460	POLYGON ((177.55 -37.32, 176.56 -36.98, 174.97 -39.36, 175.89 -39.77, 177.55 -37.32))
Nicaragua Volcanoes	~1300	~270	POLYGON ((-87.38 13.28, -87.76 12.94, -85.71 11.2, -85.32 11.49, -87.38 13.28))
San Andreas Fault Natural Laboratory	~11800	~2900	POLYGON ((-125.73 41.49, -124.11 38.33, -122.2 35.73, -120.16 33.55, -116.51 31.52, -112.87 31.48, -116.6 35.24, -118.81 36.21, -122.84 41.99, -125.73 41.49))
Southern Andes	~1700	~350	POLYGON ((-72.16 -39.75, -71.44 -39.74, -70.97 -37.76, -71.61 -37.74, -72.16 -39.75))
Vesuvius - Campi Flegrei	~1700	~310	POLYGON ((13.96 40.76, 14.55 40.75, 14.55 40.94, 13.96 40.94, 13.96 40.76))
Virunga	~1000	~210	POLYGON ((28.49 -2.53, 29.47 -2.79, 30 -0.99, 29.06 -0.76, 28.49 -2.53))

The availability of that storage will allow us to perform:

- ingestion and cataloguing of data (metadata extraction and publication)
- organisation and transformation of data (STAC, COG)
- data calibration and creation of data visualisations

This section will be further updated in the next version of this document with data sourcing activities for additional use cases.

6.3 Data Discovery and Access

With the data ingestion and cataloguing activity it is possible for the users to discover the data through the STAC API, the dynamic version of a STAC JSON file. This API returns a STAC *Catalog*, *Collection*, *Item*, or a STAC API *ItemCollection*, depending on the endpoint and enables complex queries to be performed by the users. We deployed a STAC API instance[9] where the data described in Table 1 is being registered and made available for discovery.

As one of the main objectives in RELIANCE is the use of Jupyter Notebooks as the default working and computing environment for researchers, we also developed several documentation and examples on how to discover, access and analyse EO products directly from a Notebook.

In fact, Jupyter Notebook (and its JupyterLab) is one of the most popular web-based interactive computing platforms for scientists. The JupyterLab provides a flexible interface that allows users to configure and arrange workflows for EO data processing with several existing libraries directly supporting STAC and EO processing. Ellip Solutions from Terradue manages Jupyter Notebooks already tailored for reproducible and portable EO application packages accessing large EO data collections respecting the FAIR principles.

Client libraries like *pystac_client*, *hvplot.pandas* and *ipyleaflet* allow the user to seamlessly and interactively navigate through the EO catalogue while documenting the overall process directly in the Notebook. As seen in Figure 6 , it is possible to programmatically access the catalogue and interactive maps (Figure 7) defining the geospatial domain of the study area.

[9] https://reliance-stac.terradue.com/

```
[4]: # STAC API root URL
     URL = 'https://reliance-stac.terradue.com'

     # custom headers
     headers = []

     cat = Client.open(URL, headers=headers, ignore_conformance=True)
     print(cat.title)
     print(cat)

     stac-fastapi
     <Client id=stac-fastapi>
```

Figure 6 - Python code example in a notebook creating a connection with the STAC API instance using the *pystac_client* library.

```
m = Map(center=center, zoom=6)

bbox = box(aoi_bbox[0],aoi_bbox[1],aoi_bbox[2],aoi_bbox[3])

# Now plot the AOI on the map
aoi_etna = Polygon(
    locations=np.asarray([t[::-1] for t in list(loads(bbox.wkt).exterior.coords
    color="green",
    fill_color="green"
)

m.add_layer(aoi_etna)

m
```

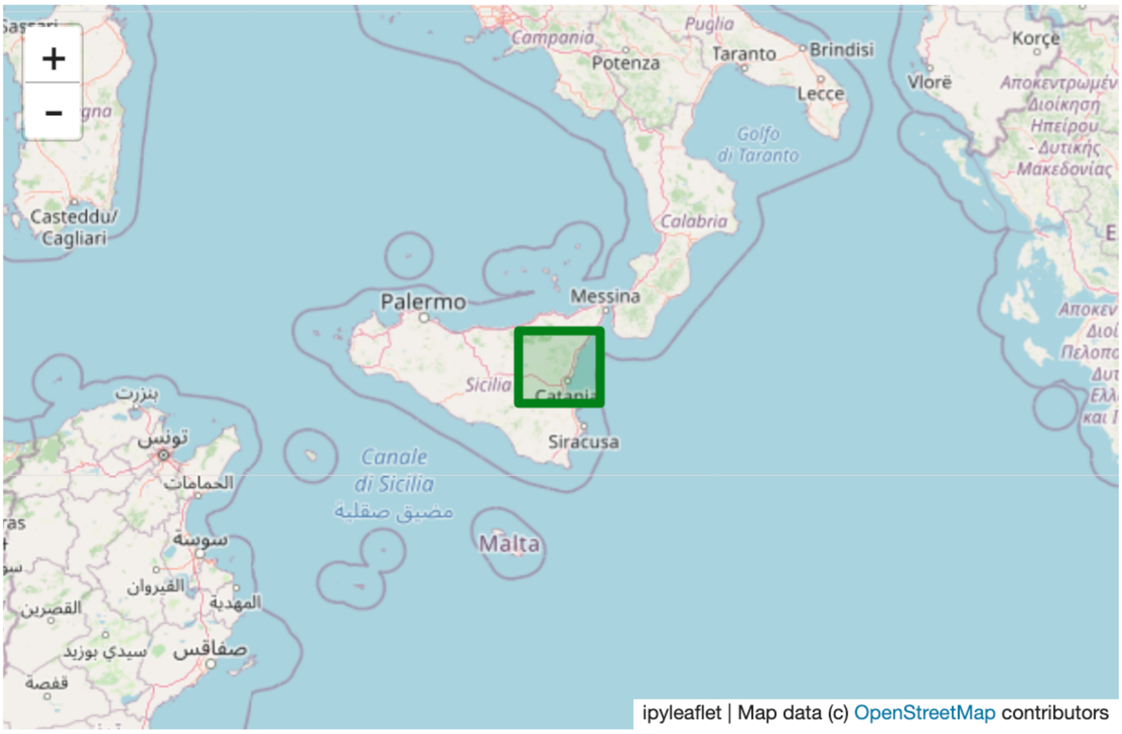


Figure 7 - Python code example in a notebook creating an area of interest for the query using the *ipyleaflet* library.

Apart from the collection we are searching, the normal default query parameters focus on the geospatial domain and data time range. However, for the needs of the INGV's *Grand Challenge 4* , in our case it was necessary to extend the query capabilities to specific metadata fields that are essential for interferometry processing. The following metadata was extracted and made available for querying:

- Sensor Type
- Instrument
- Processing Level
- Product Type
- Orbit State
- Relative Orbit
- Polarisation
- Frequency Bands

The query result is a JSON object that returns several STAC *Item* objects with the information for each discovered product (Figure 8).

```
[25]: # Show interactive content of STAC Item
      print('Show interactive content of STAC Item')
      IPython.display.JSON(items_json)
```

```
      Show interactive content of STAC Item
[25]: ▾ root:                                                          Filter...
        type: "FeatureCollection"
      ▾ features: [] 107 items
        ▾ 0:
            id: "S1A_IW_SLC__1SDV_20171115T050436_20171115T050503_019271_020A5A_6563"
          ▸ bbox: [] 4 items
            type: "Feature"
          ▸ links: [] 5 items
          ▸ assets:
          ▸ geometry:
            collection: "sentinel-1"
          ▸ properties:
            stac_version: "1.0.0"
          ▸ stac_extensions: [] 4 items
        ▸ 1:
        ▸ 2:
        ▸ 3:
```

Figure 8 - Contents of the STAC Item object.

By adding the resulting JSON objects to a data frame it is possible to create a tabular view of specific metadata fields (Figure 9) or to add the product's footprint directly in the interactive map (Figure 10) and area of interest.

[28]:
```
result = []
locations = []

for index, elem in enumerate(items_json['features']):

    bbox = box(elem['bbox'][0],
               elem['bbox'][1],
               elem['bbox'][2],
               elem['bbox'][3])

    locations.append([t[::-1] for t in list(loads(bbox.wkt).exterior.coords)])

    slave_wkt = loads(bbox.wkt)

    result.append({'id' : elem['id'],
                   'datetime' : elem['properties']['datetime'],
                   'pol' : elem['properties']['sar:polarizations'],
                   'rel_orb' : elem['properties']['sat:relative_orbit'],
                   'wkt' : loads(bbox.wkt)
                  })

slaves = gpd.GeoDataFrame(result)
slaves
```

[28]:

	id	datetime	pol	rel_orb	wkt
0	S1A_IW_SLC__1SDV_20171115T050436_20171115T0505...	2017-11-15T05:04:36.625761Z	[VV, VH]	124	POLYGON ((16.288456 36.604851, 16.288456 38.62...
1	S1A_IW_SLC__1SDV_20171103T050436_20171103T0505...	2017-11-03T05:04:36.994565Z	[VV, VH]	124	POLYGON ((16.287598 36.604584, 16.287598 38.62...
2	S1A_IW_SLC__1SDV_20171022T050436_20171022T0505...	2017-10-22T05:04:36.996392Z	[VV, VH]	124	POLYGON ((16.28792 36.604164, 16.28792 38.6290...
3	S1A_IW_SLC__1SDV_20171010T050436_20171010T0505...	2017-10-10T05:04:36.891211Z	[VV, VH]	124	POLYGON ((16.287987 36.603928, 16.287987 38.62...
4	S1A_IW_SLC__1SDV_20170928T050436_20170928T0505...	2017-09-28T05:04:36.635018Z	[VV, VH]	124	POLYGON ((16.288151 36.603657, 16.288151 38.62...

Figure 9 - Loading specific fields of the STAC API query result to a data frame in a Jupyter Notebook

[16]:
```
for index, elem in enumerate(items_json['features']):

    coords = elem['geometry']
    coords_poly = Polygon(
        locations=np.asarray([t[::-1] for t in list(coords['coordinates'][0])]).tolist(),
        color="red",
        fill_opacity=0,
        weight=1
    )

    m.add_layer(coords_poly)

m
```

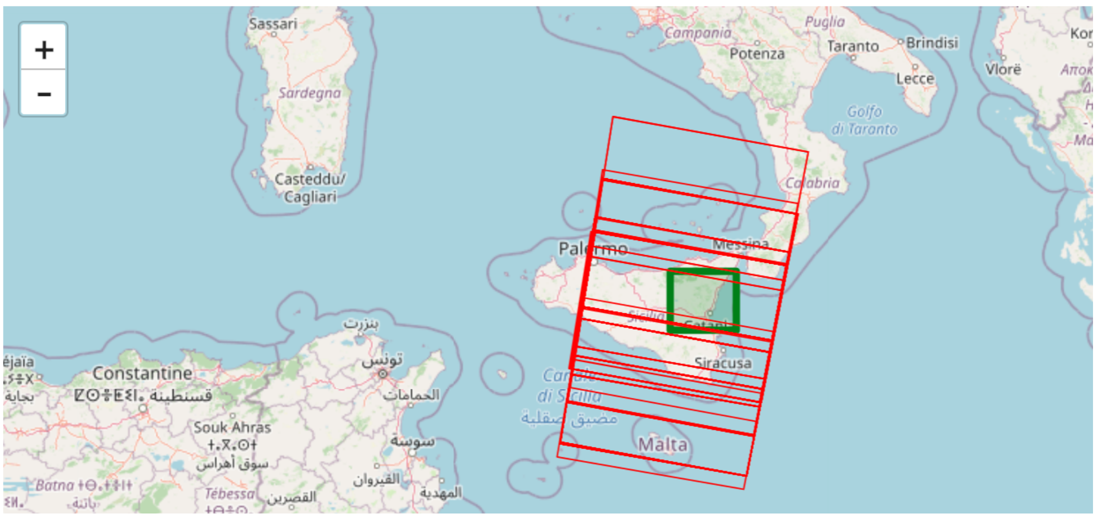


Figure 10 - Displaying the STAC API query result on an interactive map directly in a Jupyter Notebook

The next release of this document will further document the procedure for accessing the Sentinel-1 products with examples for data transformation, calibration and visualisation.

6.4 Processing Resources

As discussed previously, the Copernicus Data Pipelines use shared data processing infrastructures where the applications are deployed. The service manages the IaaS and Cloud resources, orchestrates the execution of the triggered Data Pipelines and publishes the results.

Working with Cloud-hosted data requires the availability of interfaces, APIs and tools that will alleviate the burden for operators when it comes to deploy and execute the Data Pipeline. With Kubernetes, the platform is able to dynamically respond to such changes by distributing the processing and analysis on a variable number of nodes (i.e. scaling out) and/or adding more storage space, to ensure the response time stays within a reasonable limit. Kubernetes is a Container orchestration system for automating application deployment, scaling and management fully compatible with the EO Application Package Best Practice described in the previous chapter.

Every software unit is packaged as containers and deployed on a Kubernetes infrastructure managed by Helm charts and supervised by a Continuous Deployment tool. All system components are able to scale according to the processing load at every stage: download, harvesting, pre-processing and synthetic/systematic product.

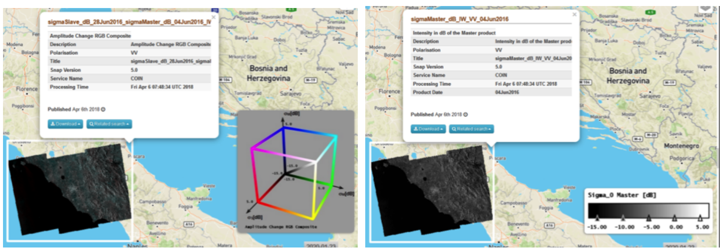
PSNC is providing the necessary resources for Kubernetes infrastructure requirements with the initial definition of the nodes with 8 VCPUs, 32 GB of memory RAM and Storage class to ensure data persistence of at least 100GB SSD.

This section will be further updated in the next version of this document.

6.5 Processing Services

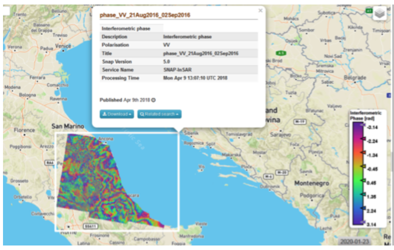
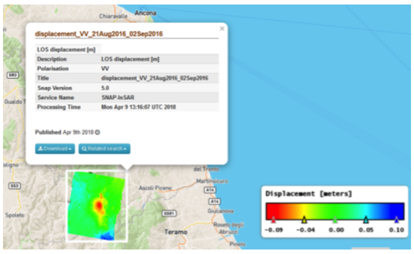
In response to *Scenario 1* and *Scenario 2* of *Grand Challenge 4*, three initial services feeding the Copernicus Data Pipelines were selected and are ready to be deployed as soon as the resources at PSNC are available.

6.5.1 COIN – Coherence and Intensity change for Sentinel-1

Description	The COIN (Coherence and Intensity change for Sentinel-1) service provides geocoded composites of coherence and amplitude images from a pair of Sentinel-1 TOPSAR IW data pairs.
Input dataset	Sentinel-1 SLC
Geographical Coverage	Global (land)
Output	Coherence and amplitude products, composite images
	

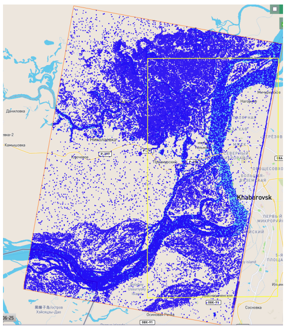
6.5.2 SNAP-InSAR – SNAP Sentinel-1 IW SLC Interferogram and Displacements

Description	The SNAP-InSAR provides the interferometric processor for the Sentinel-1 TOPSAR IW SLC data. Interferometric synthetic aperture radar (InSAR) exploits the phase difference between two observations taken from slightly different sensor positions and extracts information about the earth's surface.
Input dataset	Sentinel-1 SLC
Geographical Coverage	Global (land)
Output	Coherence, phase and displacement product

6.5.3 COMBI-Plus - Advanced Multi-Sensor Band Composite

Description	The Advanced Multi-Sensor Band Composite (COMBI-Plus) provides an advanced processor to create RGB composites at full resolution from a fully customizable combination of multiple assets from single or multiple EO data products (optical or radar) using s-expressions.
Input dataset	Pleiades 1A/1B, SPOT- 6/-7, Sentinel-2, Landsat 8, UK-DMC 2, Kompsat-2/-3, Kanopus-V, Resurs-P, Terrasar-X, Sentinel-1, ALOS, ALOS-2, Radarsat-2, Rapideye, Deimos-1, VRSS1, GF-2
Geographical Coverage	Global
Output	EO data product in GeoTIFF at full resolution Output can represent different physical measurements, such as: reflectances, backscatter in db, spectral indexes, interferometric coherence, etc.



```
{
   "input_reference": [
       ""
       ],
   "red_expression": "(interp 1.s0_db_c_hh (asarray -23 5) (asarray 0 0))",
   "green_expression": "(interp 1.s0_db_c_hh (asarray -23 5) (asarray 0 0))",
   "blue_expression": "(where (<= 1.s0_db_c_hh -23) 1 0)",
   "alpha_expression": "(where (<= 1.s0_db_c_hh -23) 1 0)"
}
```

This section will be further updated in the next version of this document.

7 References

- [OGC 18-062] OGC API — Processes — Part 1: Core Standard, 2021. https://docs.ogc.org/is/18-062r2/18-062r2.html
- [OGC 20-089] OGC Best Practice for Earth Observation Application Package, 2021 https://docs.ogc.org/bp/20-089r1.html
- [CWL] Commonwl.org: Common Workflow Language Specifications, 2021 https://w3id.org/cwl/
- [STAC] Radiant Earth Foundation: SpatioTemporal Asset Catalog specification, 2021 https://stacspec.org