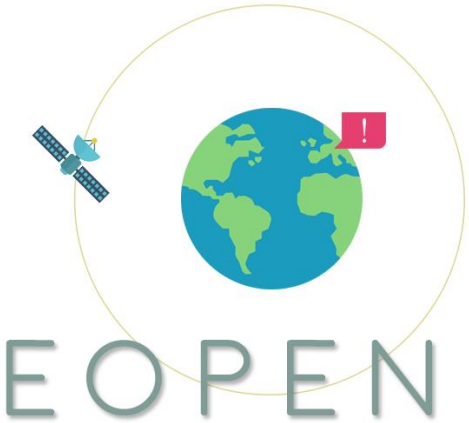

EOPEN

opEn interOperable Platform for unified access and analysis of Earth observatioN data

H2020-776019

D3.1

EO data acquisition from the Collaborative Ground Segment and quality control

Dissemination level:	Public
Contractual date of delivery:	Month 18, 30/04/2019
Actual date of delivery:	Month 18, 30/04/2019
Date of resubmission:	03/07/2020
Work package:	WP3 Knowledge discovery and content extraction

Task:	T3.1 EO data acquisition from the Collaborative Ground Segment and quality control
Type:	Report
Approval Status:	Approved
Version:	1.0
Number of pages:	59
Filename:	D3.1-EO data acquisition from the Collaborative Ground Segment and quality control_v1.0.docx

Abstract

This deliverable reports on the development of an Application Programming Interface (API) that functions as a single point of access for Sentinel data, connecting to multiple Sentinel Data Access Points. Specifically, it reports on the issue of the fragmented access to Sentinel data through the different available access points, which differ in performance, Sentinel mission availability, geographic coverage and rolling archive policy. This deliverable then discusses how the newly implemented single Sentinel Data access point, which is called Umbrella Hub, resolves the aforementioned issues by providing a uniform access to Sentinel data for all missions and by downloading from the best performing hub. Finally, the current version addresses all comments received in the 2nd review and explains the differences between this application and the Sentinel Linker Service, which has been developed during the NextGEOSS project.

co-funded by the European Union

History

Version	Date	Reason	Revised by	Approved By
0.1	10/04/2019	Initial Draft	Athanasios Drivas	Vasileios Sitokonstantinou
0.2	13/04/2019	Contributions	Vasileios Sitokonstantinou	Ioannis Papoutsis
0.3	20/04/2019	Contributions	Athanasios Drivas	Vasileios Sitokonstantinou
0.4	24/04/2019	Contributions	Vasileios Sitokonstantinou	Ioannis Papoutsis
0.5	26/04/2019	Internal Review	Dennis Hoppe	
0.6	28/04/2019	Updated document after review	Athanasios Drivas	Vasileios Sitokonstantinou
0.6	29/04/2019			G. Vingione
1.0	03/07/2020	Resubmission after the 2nd Periodic Review	Athanasios Drivas Vasileios Sitokonstantinou M. Gabriella Scarpino	Ioannis Papoutsis

Author list

Organization	Name	Contact Information
NOA	Athanasios Drivas	tdrivas@noa.gr
NOA	Vasileios Sitokonstantinou	vsito@noa.gr
NOA	Ioannis Papoutsis	ipapoutsis@noa.gr
NOA	Dimitris Filippas	dfilippas@noa.gr
NOA	Charalampos Mageiridis	cmageiridis@protonmail.com
NOA	Christos Rousakis	chroussakis@noa.gr
NOA	Fotis Tsamis	ftsamis@noa.gr
FMI	Petteri Karsisto	petteri.karsisto@fmi.fi
CERTH	Stelios Andreadis	andreadisst@iti.gr
CERTH	Ilias Gialampoukidis	heliasgj@iti.gr

Executive Summary

This deliverable presents the work done under WP3 in *EO and non-EO data acquisition* as part of the Task 3.1 on *Earth Observation (EO) data acquisition from the Collaborative Ground Segment and quality control*. It also provides brief descriptions on the work done or to be done in deliverables D3.2 and D3.3 on Meteorological and Climatological data acquisition and EOPEN Social Media crawlers respectively [Note that the most recent update about Meteorological data and Climatological data acquisition, at the time of submitting this document, is presented in D3.3]. The document has been updated so to provide a detailed analysis of the most important Sentinel data sources regarding their characteristics. This analysis is presented in Chapter 2. In addition, APPENDIX C – NEXTGEOSS "Sentinel Linker Service" VS EOPEN "Umbrella Hub Application"has been added in order to provide all the necessary answers to the 2nd review comments regarding the Umbrella Hub Application and the NextGEOSS alternative application, the Sentinel Linker Service.

It should be mentioned that the title of the deliverable, as introduced in the proposal phase, might be misleading, as the scope of the task was extended. The acquisition of EO data does not strictly come from the Hellenic National Sentinel Data Mirror Site, but rather from multiple Sentinel Data Hub Access points, also including the Copernicus Open Access Hub, the Finnish Mirror Site, and Sentinel-5P Pre-Operations Data Hub. The aforementioned hubs were selected for the demonstration of the developed EOPEN Umbrella Hub that functions as single point of access to data from all available Sentinel missions: Sentinel 1, Sentinel 2, Sentinel 3 and Sentinel 5p, as shown in Figure 1 below.

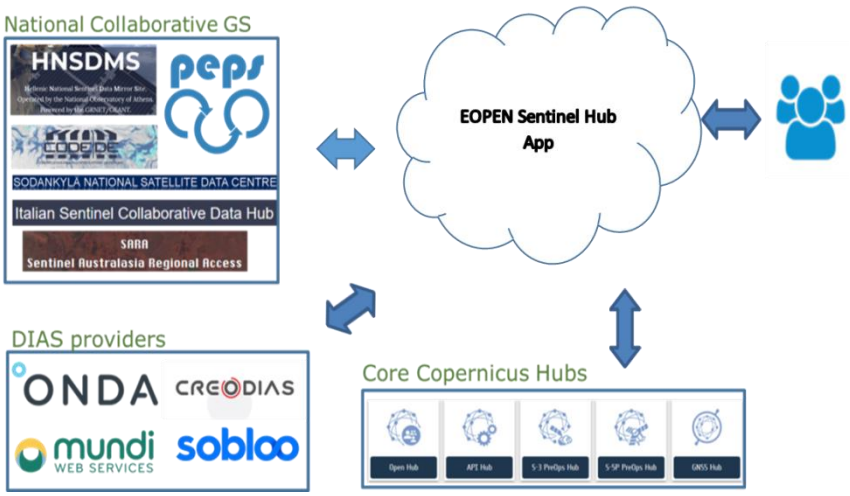

Figure 1. Umbrella application of Sentinels Data Hub Access points.

There are several Copernicus Hubs out there to access Sentinel data

- Core Hubs

- ✓ Open Access Hub (formerly SciHub)
- ✓ 4 DIAS Hubs
- ✓ ApiHub
- ✓ S3 PreOps Hub
- ✓ S5P PreOps Hub

- **23** National Collaborative Ground Segments. Indicatively:
  - ✓ HNSDMS (Greece, https://sentinels.space.noa.gr/)
  - ✓ CODE-DE (Germany, https://code-de.org/)
  - ✓ FinHub (Finland, http://nsdc.fmi.fi/services/service_finhub_overview)
  - ✓ Peps (France,https://peps.cnes.fr/rocket/)

Ten hubs of different characteristics are currently chosen to be exhibited as a proof of concept for this 2nd delivery. Nonetheless, the overall design of the proposed application was made to be linearly scalable with an increasing number of connected hubs. This will be thoroughly demonstrated in the document.

Abbreviations and Acronyms

API	Application Programmable Interface
CERTH	The Centre for Research & Technology, Hellas
DBMS	Database Management System
DHUS	Data Hub System
DIAS	Copernicus Data and Information Access Services
EO	Earth Observation
FINHUB	Finnish Data Hub
FMI	Finnish Meteorological Institute
GDAL	Geospatial Data Abstraction Library
HNSDMS	Hellenic National Sentinel Data Mirror Site
HTTPS	Hyper Text Transfer Protocol Secure
JSON	JavaScript Object Notation
MVC	Model View Controller
NOA	National Observatory of Athens
NUMPY	Numerical Python
ORM	Object-relational Mapping
PRE-OPS	Pre-Operations
REST	Representational State Transfer
SCIHUB	Scientific Data Hub
SQL	Structured Query Language
URL	Uniform Resource Locator
XML	Extensible Markup Language
YAML	Yet Another Markup Language

Table of Contents

1. MOTIVATION AND RELEVANCE

Searching for Sentinel data is often a complicated process due to the different missions available and the different hubs that host data, but also the different performances of the hubs in terms of download speed and latency (at both the inter and intra level). Thus, there is the need for an Umbrella data hub that brings them all together. We have developed such a single data access point, which is already successfully deployed and accessible on the EOPEN platform. This way, we offer to the users of EOPEN platform uniform access to Sentinel 1, Sentinel 2, Sentinel 3, and Sentinel 5p metadata via connecting in the back end to a number of the available Sentinel hubs and serving the results via an Application Programming Interface (API).

Most of these data hubs are using data hub system (DHuS[1]) that allows users to access the data via their own computer programs, scripts or client applications. API Query is based on OpenSearch protocol, which is a collection of technologies that allow publishing of search results in a format suitable for syndication and aggregation. Ultimately, this application gives the potential for

- linking federated Copernicus Sentinels Hubs to a single data hub, instead of searching for the appropriate one for the user's needs;
- accessing to all Sentinel mission data and better performance on downloading products, as products are chosen from the most appropriate data hub based on integrity, speed and availability tests.

Users occasionally visit more than one hub to discover and select the required products due to several reasons. First of all, there is no availability of all **Sentinel missions** in every hub. For example, there is only one hub that offers Sentinel-5p products. Furthermore, there is no **global coverage** for most of the hubs; for instance, the Hellenic National Sentinel Data Mirror Site (HNSDMS) provides data for southeastern Europe, Middle East and North Africa. The next problem to tackle is the **data rolling policy**. Sentinel Products are kept in each hub's repository for a specific amount of time and are deleted afterwards. The hubs keep only the products ingested on the last N days, where N is a number of days determined by the hub's policy. For example, HNSDMS has set the retention period of the products to 30 days. In addition, occasionally users come up against certain performance issues such as slow response times and download speeds. Figure 2 reveals these differences by download time not among the different hubs (inter level), but also concerning the average, the maximum and minimum download speed within the same hub (intra level). For example, Figure 2 shows a great variability in Copernicus Open Access Hub, which is represented by abrupt inner changes from high to low download speed and vice versa . We can see

[1] https://sentineldatahub.github.io/DataHubSystem/

that in the month of January there is an average of high speeds above 200 Mbps, while in the month of February we see an average of high speeds below 100 Mbps. The product will be downloaded from currently most efficient hub. This is to show that the aforementioned issues have been documented and their adverse impact has been quantified. This is based on a key measurement evaluation on multiple Sentinel hubs, as performed by the DevOps team of NOA that operates HNSDMS, S5P PreOps Hub, Sentinels International Access Hub, Collaborative Hub Node 3, and DIAS Hub Node 3.

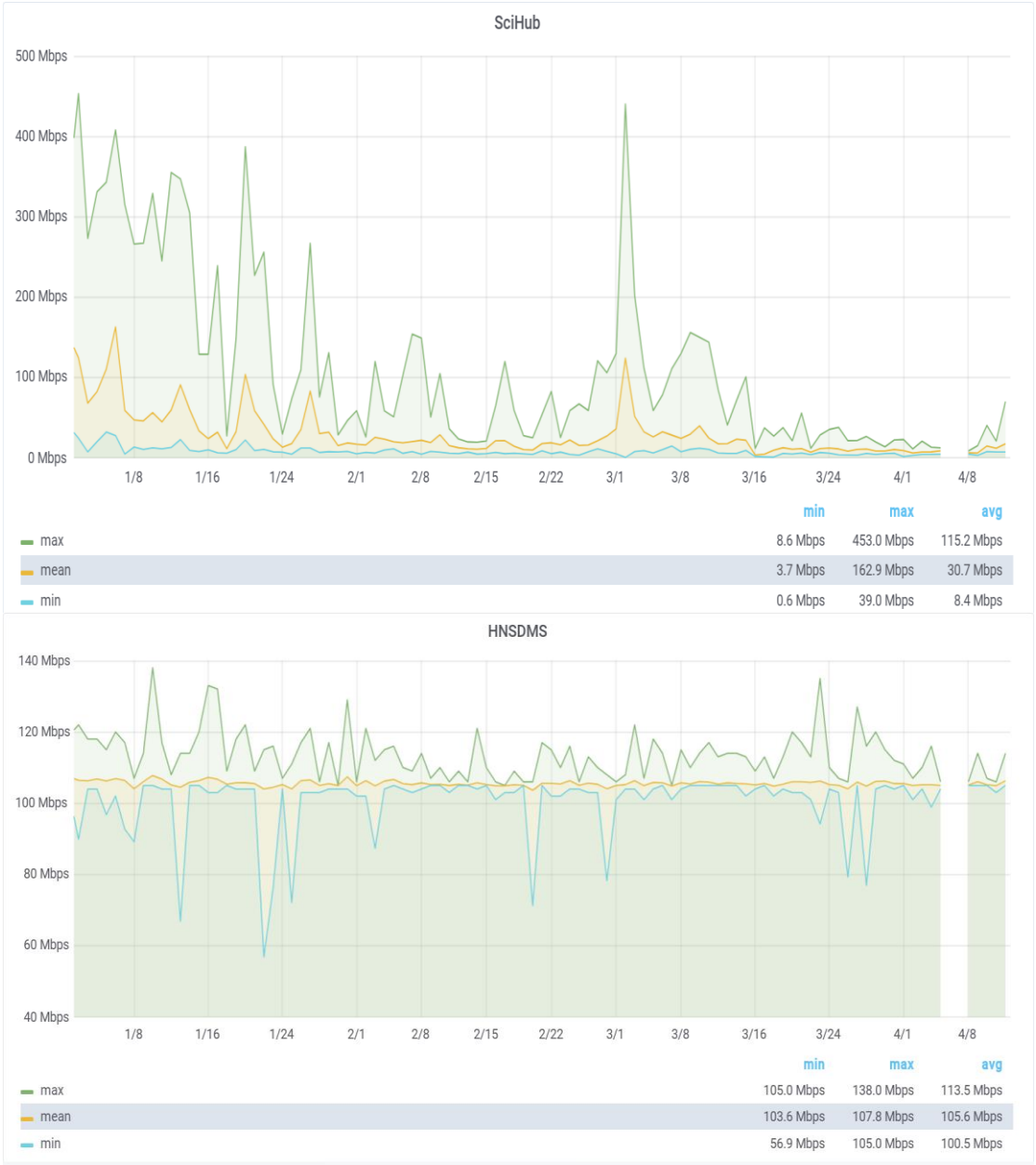

Figure 2. Maximum, average, and minimum download speeds for SciHub and HNSDMS.

Another limitation is the **data availability** in the different hubs; firstly because not all Sentinel missions and their data are available in each hub and secondly data must be put under maintenance mode on specific time intervals. Thus, products coming from these hubs are not available for several hours or days. Even worse, there might be a timeout or a server error causing access problems.

Taking into consideration all the above, the need a single point of access to Sentinel data is manifold. The application proposed in this document connects to the APIs of the available hubs, searches and stores new metadata and chooses the most appropriate source from which a requested product will be downloaded in the most efficient way. The advantages of linking these sources together are recorded below:

- Access to a single hub instead of looking across several Sentinel Hubs to find the appropriate products.
- Access to all Sentinel mission data
- No geographic restrictions
- Better performance/download variability by exploiting Hub diversity
- No delay due to maintenance of a hub

As a proof of concept, metadata from **ten hubs** are ingested; showcasing the innovations of the Umbrella hub, including its modular architecture that allows for the connection of hubs with diverse architectures.

2. ANALYSIS AND SELECTION OF DATA SOURCES

The growing demand of Sentinel data results in the creation of several data hubs. These dedicated data hubs are based on different architectures and address different user needs. For example, there are data hubs that provide access to both online and archived data , whereas other hubs do keep a catalogue of the most recent data. Access to data is free and users can self-register to the data hubs in order to gain access. Data hubs provide either a web Graphical User Interface (GUI) to allow interactive data discovery and download or open APIs allowing automated access to the catalogues. An analysis has been performed to several data hubs so to identify information regarding the offered Sentinel missions, the geographic coverage, the rolling policy and statistics. The analysed hubs are presented below.

2.1. Open Access Hub

The Copernicus Open Access Hub[2] (known as Sentinels Scientific Data Hub) provides complete, free and open access to Sentinel-1, Sentinel-2, Sentinel-3 and Sentinel-

[2] https://scihub.copernicus.eu/

5P data (via Sentinel-5P Pre-Operations Data Hub) . Open Access Hub manages data via DHUS and makes it available via an interactive graphical user interface as shown in Figure 3 or two dedicated APIs, ODATA and OpenSearch, for browsing and accessing the EO data. An automatic and immediate self-registration process is required in order to access and download the data (maximum of 2 concurrent downloads per user). Data are kept online according to a rolling policy based on sensing date. Specifically, Non-Time-Critical (NTC) products of all missions have a retention period of 12 months, except from Sentinel-2 L2A (18 months). Additionally, NTC products are moved from the online storage to the long term archive after a certain time period determined by each hub's policy and marked as offline. This kind of data can be requested from users and be restored for a limited amount of time. At the time of writing, 338,150 users are registered, 251.57 PB of data is downloaded, and 28,058,073 products are published with a daily publication rate of over 30,500 products/day since the start of operations. Finally, Open Access Hub had 98.3% availability for May 2020.

Figure 3. Open Access Hub Interface

2.2. Sentinels Collaborative Ground Segments

The Sentinel Collaborative Ground Segments, offered by EU Member States and International partners enhances the access to Sentinel data. In the framework of the Sentinels Collaborative Ground Segment, National Mirror sites are set up by partner countries. As mentioned before, most of these data hubs are using Data Hub Open Source software and specifically the new version 2.0.0, released in March 2019. This latest version offers an extreme reduction in query times along with improved relay performance.

2.2.1. Austrian Collaborative Ground Segment

The Austrian Collaborative Ground Segment[3] includes the Sentinel National Mirror Austria (Figure 4) and a Data Hub Relay (DHR) operated by ZAMG. The National Mirror

[3] https://sentinel.zamg.ac.at/

provides access to Sentinel-1, 2 and 3 missions. Products are offered globally with a sliding window of 35 days. During the last years, this hub managed to duplicate its computational capacity with each Sentinel mission run on a dedicated virtual machine, while it has been upgraded to the new version of DHuS. In addition, Austrian National Mirror served, in 2019, 300 TB of products to 1600 users.

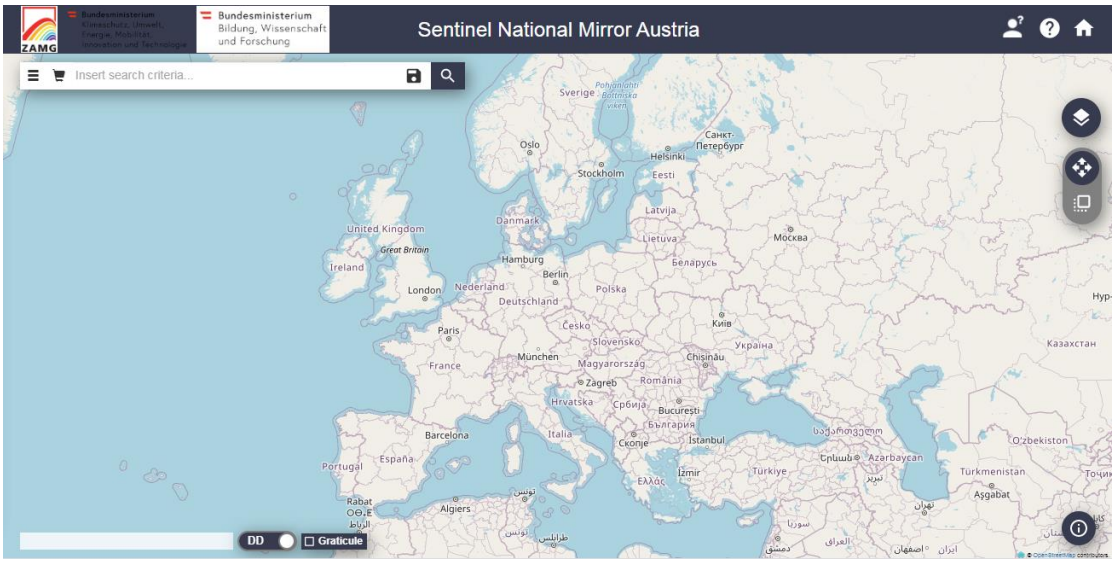

Figure 4. Austrian National Mirror Interface

2.2.2. Czech Collaborative Ground Segment

The Czech CollGS[4] is managed by the Ministry of Transport and developed and operated by CESNET. A National Mirror (Figure 5) has been available since late 2016 and a Data Hub Relay since spring 2018. During 2019, there have been 163 registered users, 67 of them active, mostly Czech users and a few notable international. ODATA queries reached 1.3 million, whereas 201,850 products have been served, mostly used in Land and Agriculture services. At the same time, DHR feeds the National Mirror using a DHUS back-end instance separately for each satellite. At the moment, and for the foreseeable future, there is not any rolling policy on the dataset, which covers the Czech Republic and surrounding regions, trying to maintain a full archive of Sentinel products for the national area.

[4] https://www.cesnet.cz/

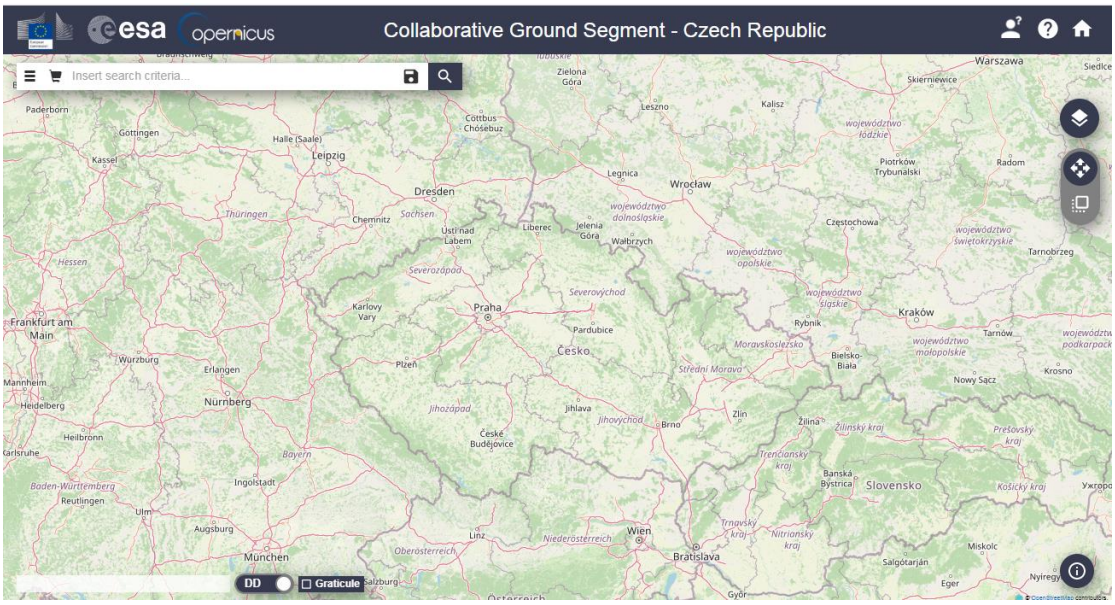

Figure 5. Czech National Mirror Interface

2.2.3. French Collaborative Ground Segment

The Sentinel Product Exploitation Platform (PEPS)[5] is developed and operated by CNES. It was launched in September 2015 and counts already 5500 registered users (56% French users). PEPS comes with a different architecture as it uses the rocket interface (Figure **6**) and resto as the metadata catalogue for geospatial data. The main functionalities of PEPS are the mirror site of Sentinel-1 and Sentinel-2 products (Sentinel-3 download functionality terminated in 2019 due to limited interest), processing tools and access to high performance resources with user support. PEPS provides access to global coverage and full archive. The amount of daily acquisition is approximately 10 TB, whereas the total number of products at the time of writing is 21,074,087. Among other metrics, the number of registered users is stably increasing, while the interest for Sentinel-1 and Sentinel-2 is balanced. Finally, PEPS has a rolling policy of 30 days, as the most recent products go on disk and after this period the storage mode turns to tape.

[5] https://peps.cnes.fr/rocket/

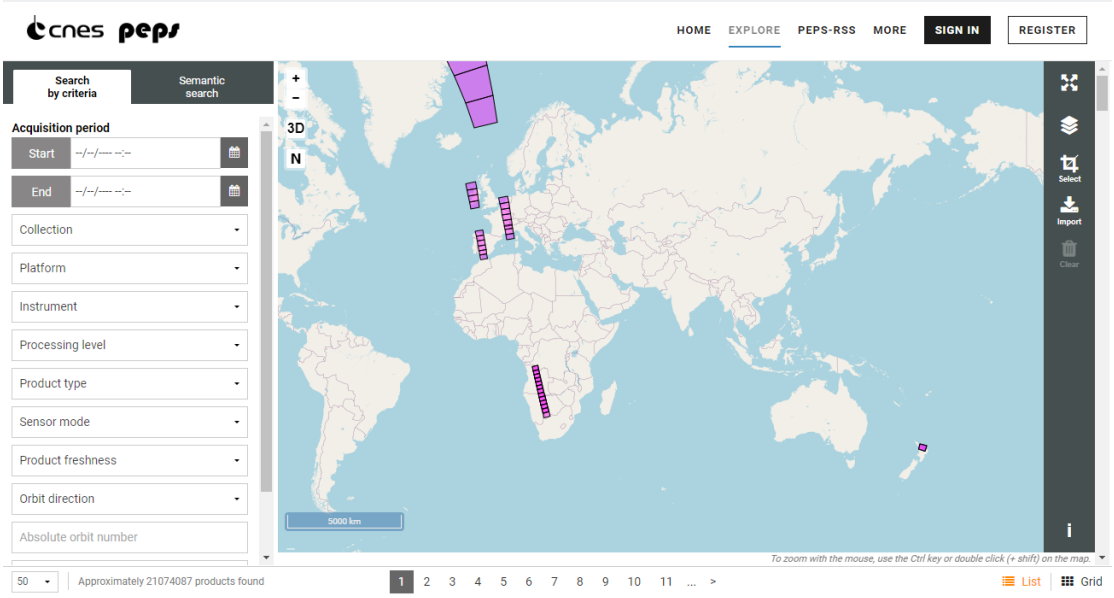

Figure 6. PEPS Interface

2.2.4. CODE-DE

The CODE-DE data and exploitation platform[6] has been in operational mode since March 2017, supporting data access and on-demand processing. CODE-DE provides Sentinel-1, Sentinel-2, Sentinel-3 and Sentinel-5p products. Sentinel-1 and 2 products are stored on a global-scale for ten days, on a European-scale for 20 days and over Germany for the entire period. On the contrary, Sentinel-3 and 5P products are stored globally for 1 month and 1 year for Europe. EO Finder tool (Figure **7**) provides visualized access to this data, along with automated search, processing and downloading via an API.

[6] https://code-de.org/

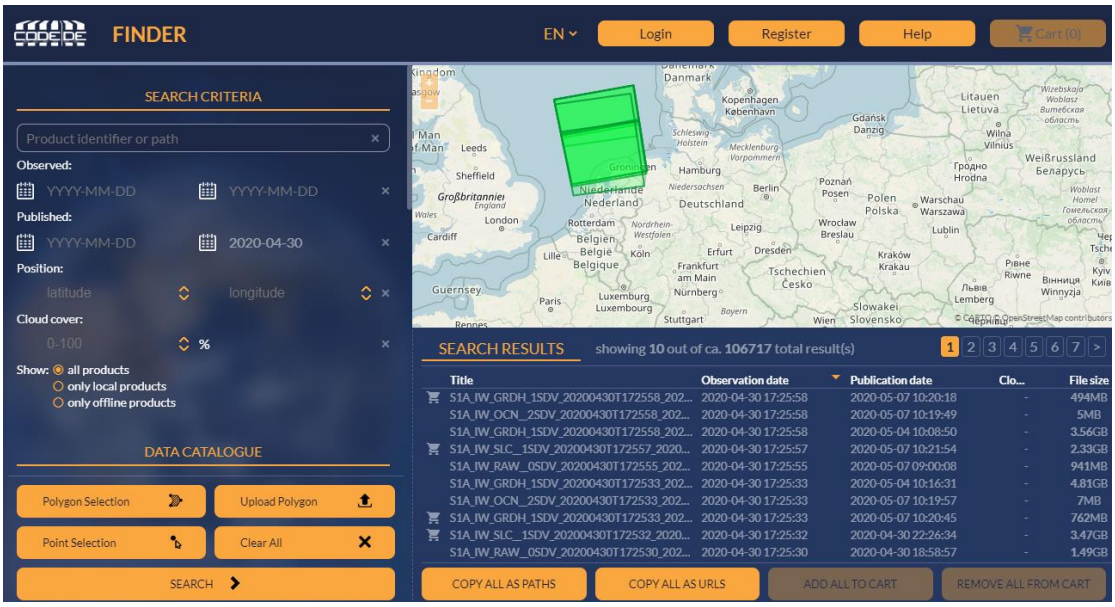

Figure 7. EO Finder Interface

2.2.5. Hellenic Sentinel Data Hub

The Hellenic Sentinel data mirror site[7] (Figure **8**) synchronizes products from the ColHub Node 3 for a specific area of interest, covering the Mediterranean, Black Sea and surrounding lands. Sentinel-1, Sentinel-2 and Sentinel-3 missions are available with a 25-day rolling archive using a 44TiB NAS storage. No deletion list of products is provided. In addition, most of the approximately 700 registered users are from Greece but also a number of mostly European countries. The main usage of the downloaded products is atmospheric and land applications.

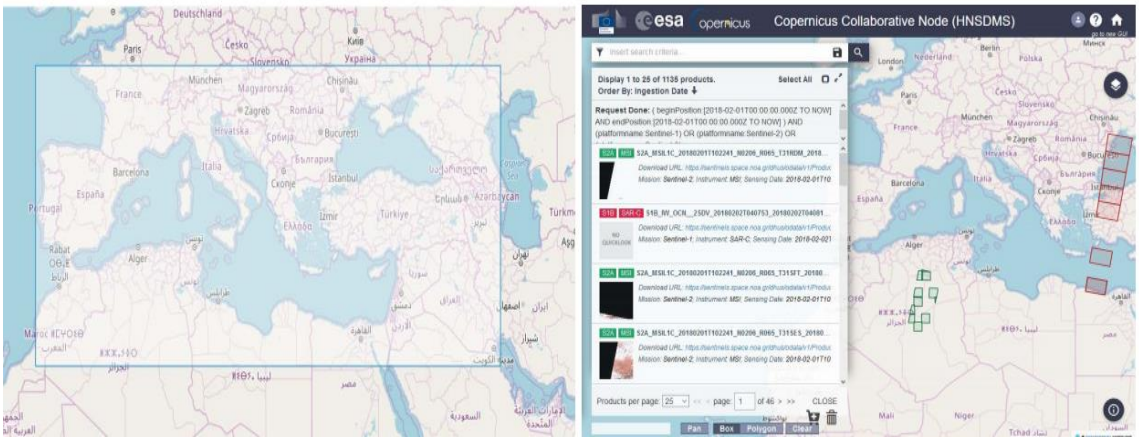

Figure 8. Hellenic Mirror Site Interface

[7] https://sentinels.space.noa.gr/

2.2.6. Finnish Mirror Site

The Finnish Collaborative Ground Segment[8] comprises of the National Mirror site (Figure **9**), a local reception station for Sentinel-1 Direct Broadcast and data processing services. It has been open to the public since May 2015, by providing access to Sentinel-1, Sentinel-2, Sentinel-3 and Sentinel-5P in the area that covers mainly Baltic Sea Drainage basin utilizing DHUS. There is no rolling policy adapted, so it includes products generated back in 2017. Moreover, it does not provide any deletion catalogue. The total of the registered users is approximately 316 with an annual 60% increase. The majority belongs to Finnish research institutes, but there are also European and overseas users.

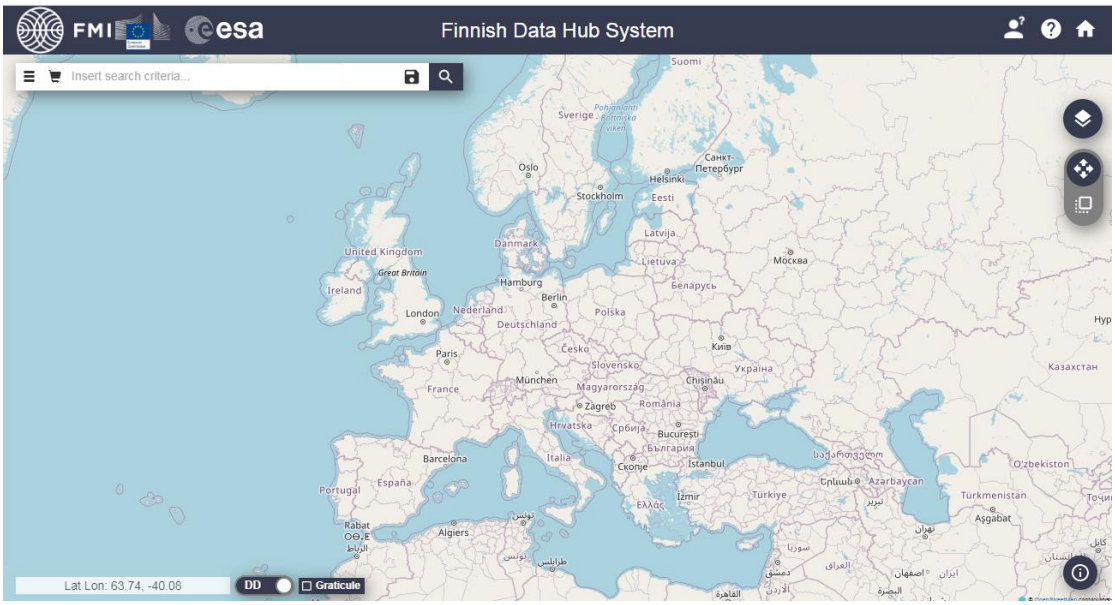

Figure 9. Finnish National Mirror Interface

2.2.7. IPSentinel

Portuguese infrastructure for storage and availability of images from Sentinel satellites (IPSentinel)[9] allows the access to open and free data for the Portuguese area including the search and rescue responsibility area in the Atlantic using DHUS (Figure **10**). This data comes from Sentinel-1, 2 and 3 missions. Especially, Santa Maria collaborative station collects Sentinel-1 data quickly, upgrading this station to the first one from the national territory that acquires data which will only be accessible afterwards from the ESA repository. Finally, IPSentinel keeps the last 60 days of archive available in the aforementioned missions.

[8] https://nsdc.fmi.fi/

[9] http://ipsentinel.ipma.pt/

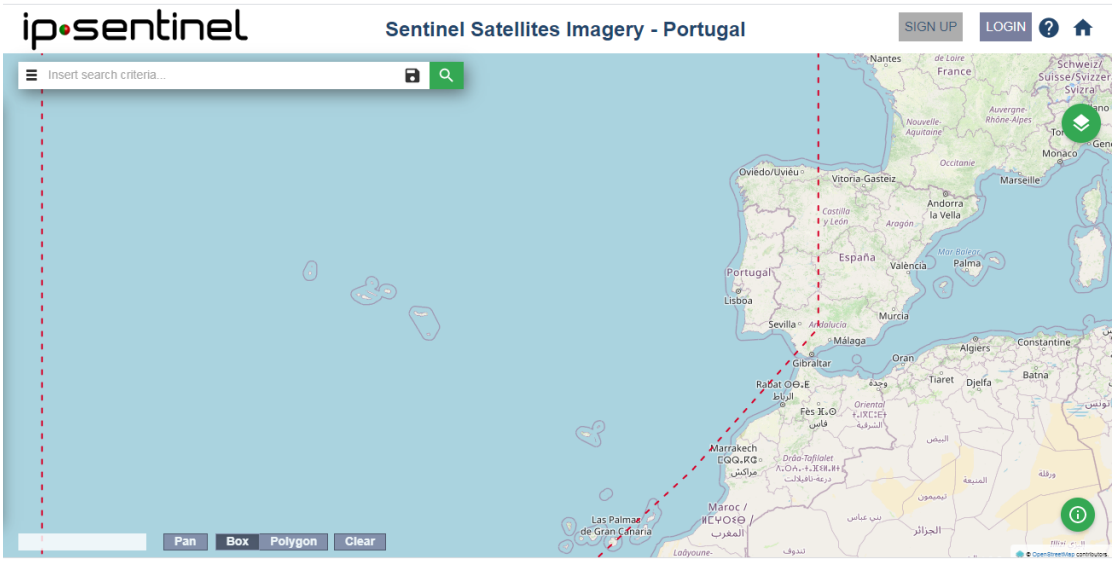

Figure 10. IPSentinel Interface

2.2.8. Norway Collaborative Ground Segment

The Norwegian Collaborative Ground Segment[10] offers two portals for the data distribution: colhub.met.no (Figure **11**) and satellittdata.no.DHUS Approximately 430 registered users have been granted access to Sentinel-1, Sentinel-2 and Sentinel-3 data, totally 11.5 million products. Sentinel-5p products are available only in back end and will be available when system will be upgraded to DHUS v2. Note here that the Norwegian hub is not and will not mirror the open access hub. Thus, there are products in it that do not exist in the open access hub and vice versa. Currently, all data in Norwegian hub are online.

[10] https://satellittdata.no/

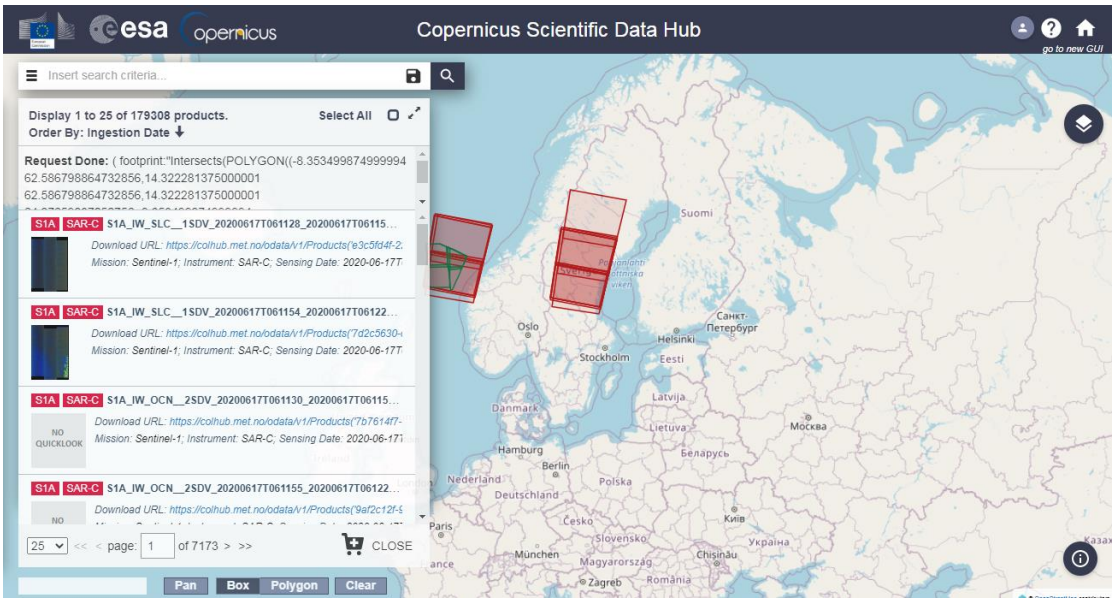

Figure 11. Norwegian hub Interface

2.2.9. Romanian Collaborative Ground Segment

The Romanian mirror site[11] was created and is maintained by the Romanian Space Agency, known as ROSA (Figure **12**) , provides Sentinel data from all missions for one month. The current area of interest is Romania and eventually will be extended to the entire Danube territory and Black Sea basin. The national mirror of Romania considered to be a central point of access for several organizations, such as Meteo Romania. DHUS is also used by this mirror for the dissemination of Sentinel data.

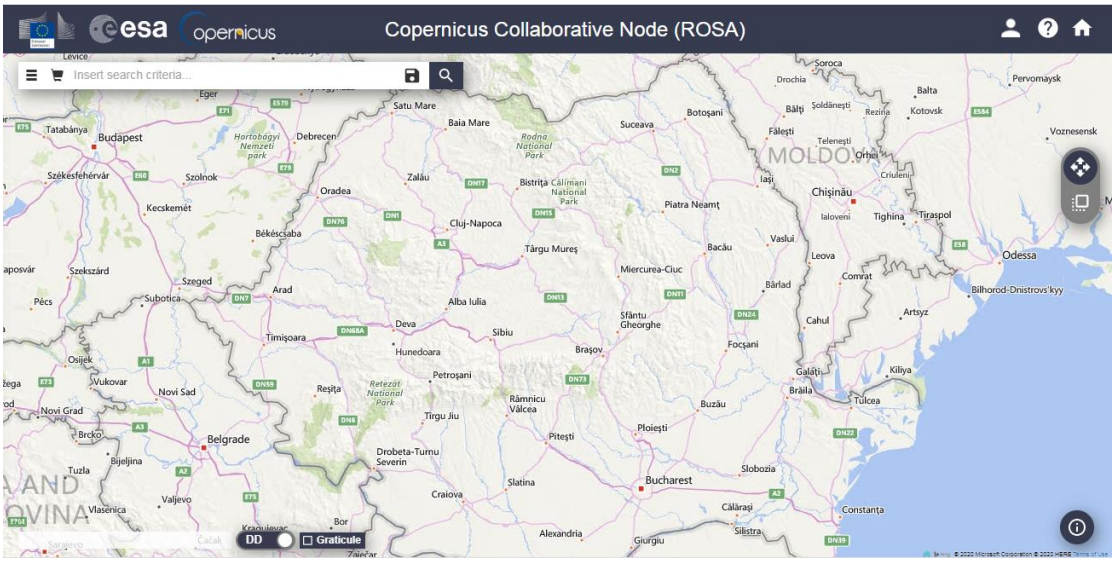

Figure 12. ROSA Interface

[11] http://www2.rosa.ro/index.php/en/

2.2.10. UK Collaborative Ground Segment

The current UK CollGS comprises two parts: the first one is SeDAS[12] (Figure **13**), operated by Catapult and serving commercial users and the other one is JASMIN, operated by STFC-RAL and providing academic data access. Specifically, SeDAS stores Sentinel-1 and Sentinel-2 globally data. Data are maintained for a minimum of one year rolling archive. The latest official statistics revealed 670 registered users from more than 65 countries. A dedicated API has been built to allow automatic data search and download. Moving to JASMIN super data cluster, it stores approximately 5 PB of data from all Sentinel missions, with more than 590 registered users.

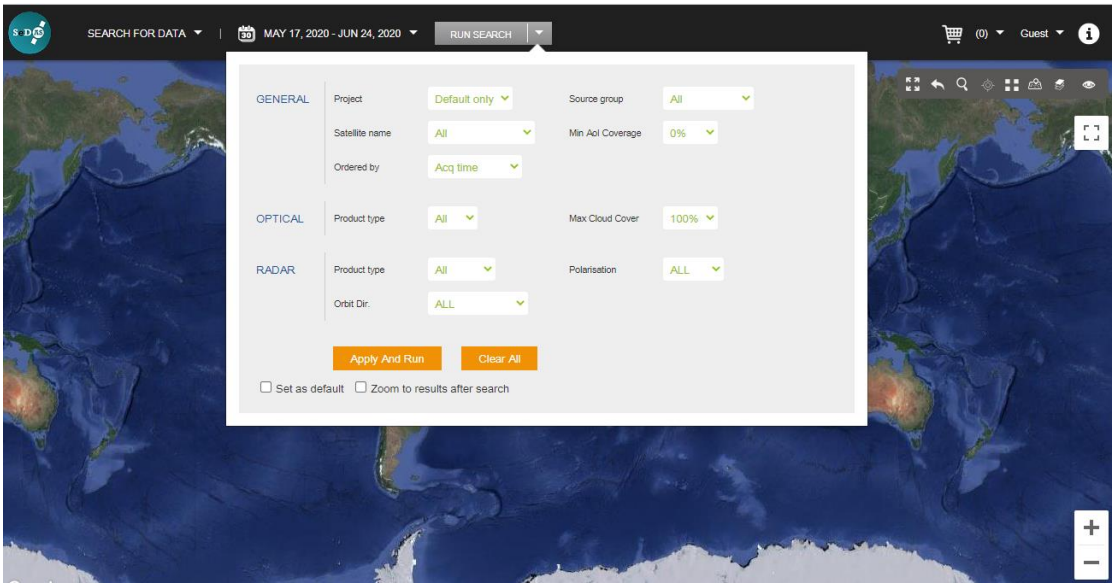

Figure 13. SeDAS Interface

2.3. Data and Information Access Services (DIAS) Projects

The Copernicus Big Data approach consists in Data and Information Access Services (DIAS). DIASs are five cloud-based platforms that provide a single access point to Sentinel data, along with pertinent processing tools. DIASs, additionally, provide access to the information products generated by Copernicus' six operational services. It is noteworthy that there is no reason for the Umbrella hub to be set up on one of the DIASs. The Umbrella hub can be deployed in any infrastructure bringing together live metadata from a number of different hubs and to provide the most efficient hub to download a certain product at a particular instance. EOPEN's scope is to serve these metadata to any user globally and not only to the DIAS users. However, the Umbrella hub is able to enhance the DIAS hubs by connecting to one or more of them and get metrics in order to provide metadata also from there.

[12] https://sedas.satapps.org/

2.3.1. ONDA-DIAS

ONDA DIAS[13] is the Serco's cloud-based platform for accessing geospatial data. OVH, GAEL and Sinergise are the partners of this project. ONDA stores more than 34 million products. It provides free and open access to all Sentinel missions with more than 2104registered users at the time of writing according to public dashboard of ONDA. Access to data is achieved either via a Catalogue Graphical User Interface as depicted in Figure **14** that allows filtered queries or via two APIs: OData Protocol and Advanced, Linux-based, API (Elastic Node Server). Sentinel-1 (except for SLC products over Europe), Sentinel-2 and Sentinel-3 with sensing date older than three months are archived. Beyond data access, ONDA brings in open source software tools for data processing (e.g. SNAP) in preconfigured virtual environments and the possibility for addition of custom features and dedicated services to the virtual servers.

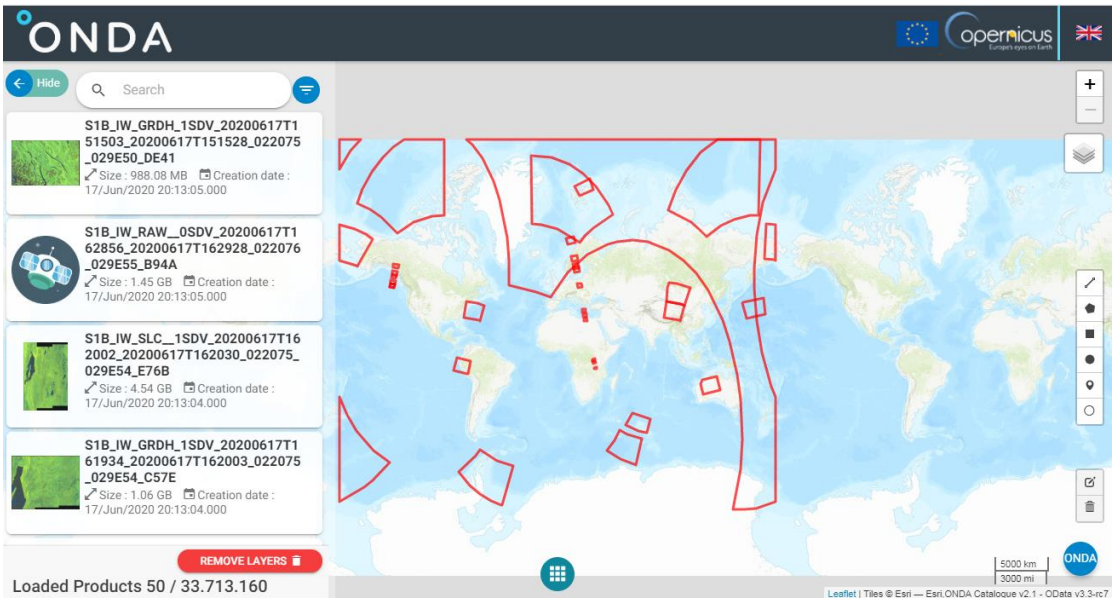

Figure 14. ONDA Dias Catalogue Interface

2.3.2. Creodias

Creodias[14] cloud platform combines big EO Data storage with EO processing tools. The platform contains partial repository of Sentinel-1 and full repositories of Sentinel-2, 3 and 5-P, Envisat and Landsat-5, 7, 8 data. Approximately 15 PB of data are stored. CREODIAS provides the EO Browser (Figure **15**), a browser that enhances visualisation of chosen Earth Observation images. It also provides Finder Tool which acts as an advanced search engine allowing queries based on certain parameters. EO Finder uses resto but requests are limited to 60 per minute per source IP address.

[13] https://www.onda-dias.eu/cms/

[14] https://creodias.eu/

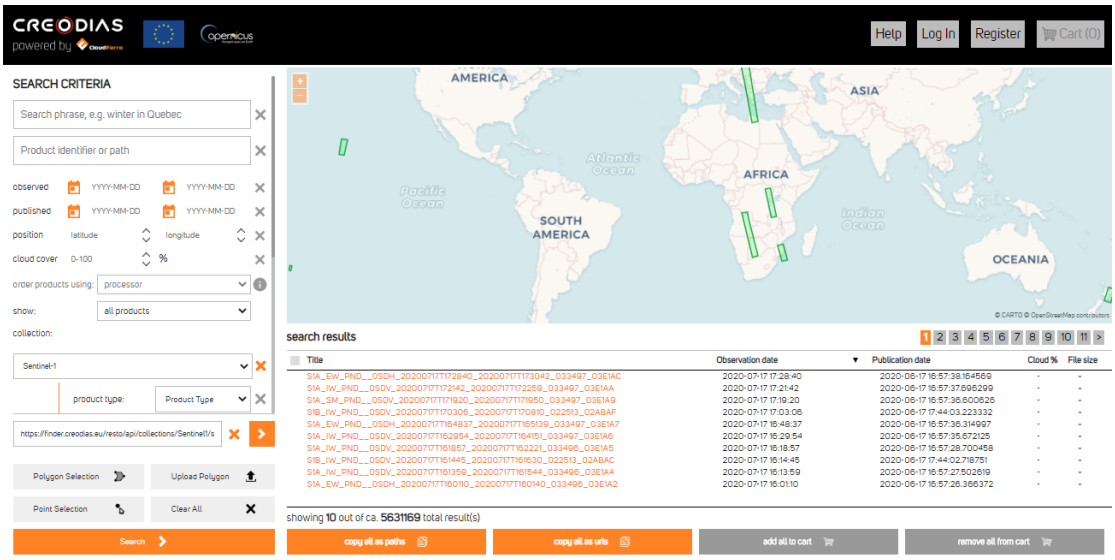

Figure 15. Creodias EOBrowser Interface

2.3.3. Mundi

Mundi[15] has been developed by a fresh consortium, composed of DLR, e-Geos, EOX, GAF, Sinergise, Spacemetric, Thales Alenia Space and T-Systems, with the leadership of Atos. Mundi provides access to a large range of satellite data and information provided by the Copernicus services and in the same time it offers tools and utilities for advanced processing. Regarding Sentinel data, a rolling policy of 12 months for World is applied to Sentinel-1 GRD and SLC products and 24/48 months for Europe respectively. Moving to Sentinel-2, L1C fresh data with global coverage are online for the last 12 months. L2A data are online from July 2015 with Europe coverage only. Sentinel-3 OLCI global data is provided from the beginning of 2018. Finally, Sentinel-5P L1B data can be discovered online from January 2019 with global coverage.

[15] https://mundiwebservices.com/

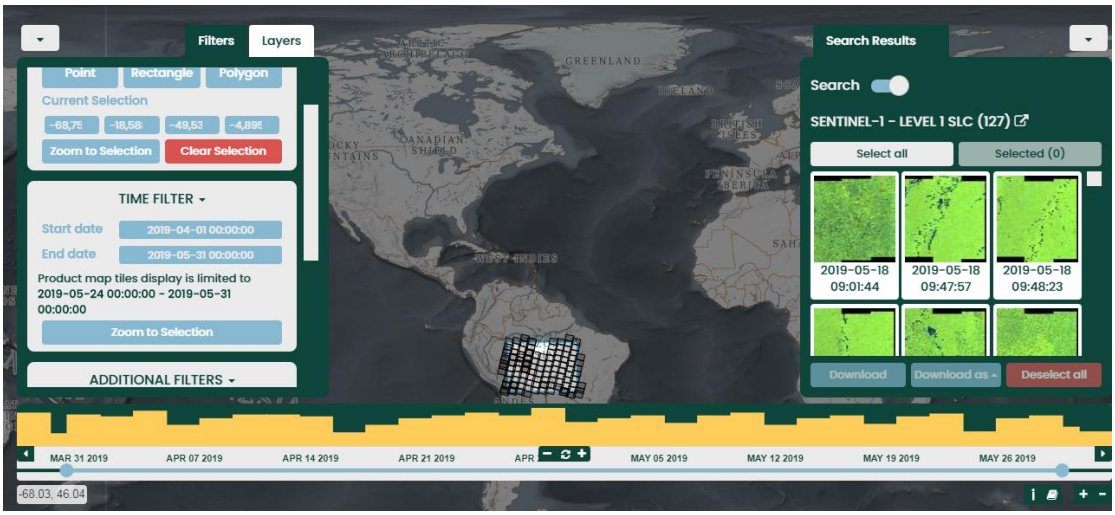

Figure 16. Mundi Catalogue Interface

2.3.4. Sobloo

Sobloo[16] was developed and is run by Airbus, Capgemini, Vito, and Orange. This cloud-based platform provides access to different types of data such as Copernicus Data, Mobile data, commercial imagery etc. In addition, a stack of features, including processing EO data tools, high computational power, generic cloud managed services and APIs is offered. Public dashboards serve statistics regarding number of items per collection, IaaS performance and products' volume, download speed and response time of APIs. Last but not least, Sobloo offers data from all Sentinel missions globally, storing currently more than 5 million of Sentinel-1 products, more than 19 million of Sentinel-2 products, more than 4 million Sentinel-3 products and more than 0.3 million of Sentinel-5p products.

[16] https://sobloo.eu/

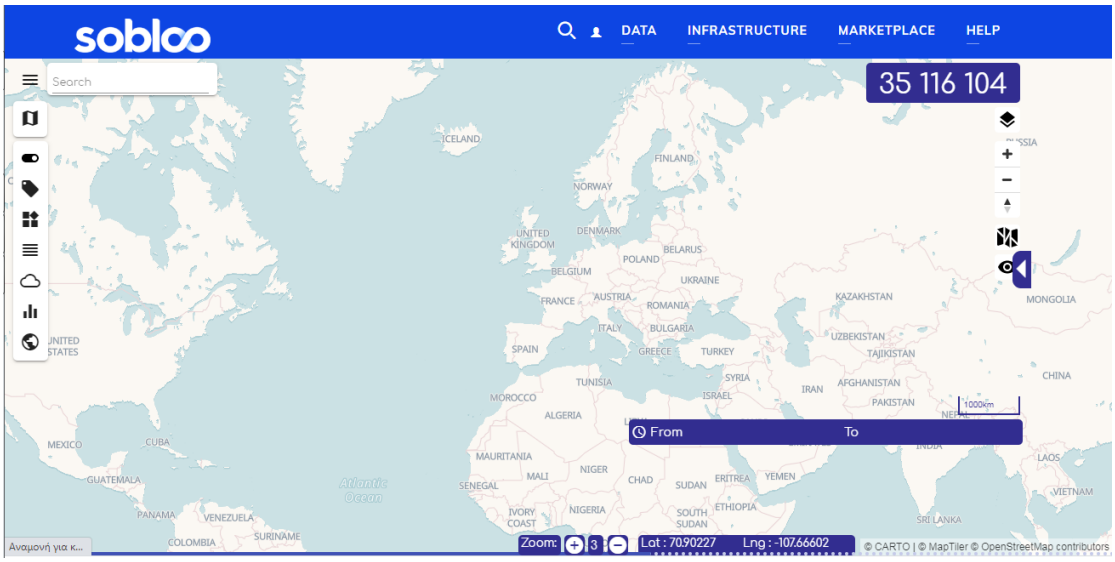

Figure 17. Sobloo Catalogue Interface

2.3.5. Wekeo

Wekeo[17] is the result of the EUMETSAT, ECMWF and MERCATOR OCEAN collaboration. The three data and infrastructure centres are linked together to a distributed platform. Wekeo stores data from multiple Sentinel missions (Sentinel-1,2,3 and 5-p), as wells as Copernicus services. A web portal (Figure **18**) is offered for accessing the catalogues along with a dedicated Rest-based API, the Harmonised Data Access API.

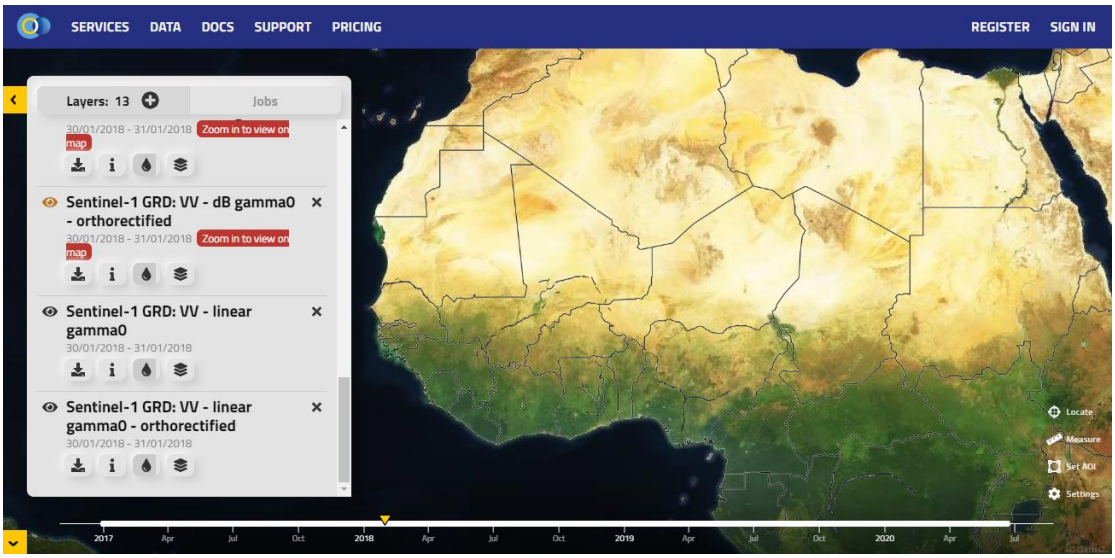

Figure 18. Wekeo Catalogue Interface

[17] http://wekeo.eu/

2.4. Data hub selection

The Umbrella hub has the potential to connect to hubs' and mirror sites' APIs with any architecture, DHUS or not, as long they satisfy a number of criteria, as presented below:

- Rolling policy knowledge. There is the need to know the rolling policy in order to delete any product that is no longer available. Alternatively, a Deletion catalogue must be provided so that Umbrella can look into it and identify the products that are no longer online.
- Specific search parameters in API. Candidate hub's API must allow queries based on date. The reason lies in the architecture of Umbrella that requires the harvest of products to start from the previous search date time to the current timestamp. Another useful parameter is a parameter that separates online and offline products (usually called online or offline).
- Download capability. A critical module of the Umbrella hub is the scoring process. This process is mainly based on downloading a part of the product and measure the download speed. Thus, the candidate API has to provide free download of the products. By analyzing the characteristics of the hubs, eleven hubs have been gradually selected to be connected to the Umbrella hub. These hubs are either DHUS based or based on another architecture. One of them, IPSentinel, is set to non-active as there is a problem on certification verification during the metadata request, so, finally, ten of them are connected and presented in Table 1.

Table 1. The characteristics of connected hubs

Data Source	Archive Policy	Deletion Policy	Missions	Performance	Geographic Coverage	Selection Reasons
Copernicus Open Access Hub	Products from January 2018 (online archive of at least the latest year of products)	Corrupted and duplicate products are deleted every 24 hours	Sentinel-1 Sentinel-2 Sentinel-3	Slow response and variant download speed	Global	Global coverage, three missions, long time availability of online products, many registered users
Hellenic National Sentinel Data Mirror Site.	Products online from last 25 days	No deletion list	Sentinel-1 Sentinel-2 Sentinel-3	Very fast response and high download speed	South & South-eastern Europe, Middle East & North Africa	Fast API Response, three missions. simple to connect
Finnish Mirror Site	Products online from February 2017	No deletion list	Sentinel-1 Sentinel-2 Sentinel-3	Fast response and high download speed	Sentinel-1,2: Scandinavia and Baltic areas, Shaksgam valley, Kyagar glacier lake, Kirgisia, Tazdikistan, Iceland strait,	Products online from 2017, three missions, simple to connect

					Bolshevik island, Tiksi Seninel-3: SLSTR Northern hemisphere	
Sentinel 5P Pre-Ops Hub	Products online from April 2018		Sentinel-5	Fast response and high download speed	Global	Primary source of S5p
Austrian Mirror Site	Products online from last 35 days	Deleted Products collection	Sentinel-1 Sentinel-2 Sentinel-3	Slow response and high download speed	Global	Updated to DHUS v2, global coverage, many registered users, simple to connect
Czech Mirror Site	No rolling policy	Deleted Products collection	Sentinel-1 Sentinel-2 Sentinel-3	Very fast response	Czech Republic and surrounding regions	Updated to DHUS v2 with an external database, no rolling policy, very quick API response, simple to connect
PEPS	Products online from last 30 days		Sentinel-1 Sentinel-2	Fast response and high download speed	Global	Non-DHUS, resto API, global coverage

Norway Collaborative Ground Segment	Currently, no rolling policy	Deleted Products collection	Sentinel-1 Sentinel-2 Sentinel-3	Fast response and high download speed	Global	Does not mirror Open Access Hub, therefore provide access to products that cannot be found in Open Access Hub, no rolling policy, global coverage, three missions
Romanian Mirror Site	Products online from last 30 days	Deleted Products collection	Sentinel-1 Sentinel-2 Sentinel-3	Fast response and high download speed	Romania	Fast API Response, three missions. simple to connect
Ondadias	Products online from last 90 days, except from Sentinel-1 SLC products over Europe		Sentinel-1 Sentinel-2 Sentinel-3	Very fast response and very high download speed	Global	Non-DHUS, global coverage, three Sentinel missions

The aforementioned connected hubs can be categorized into two groups: the hubs that use DHUS combined with a dedicated ODATA and OpenSearch API and the hubs that use another API format (such as resto). OData (Open Data Protocol) is a standard, allowing the definition of rules, conventions and formats for handling data online via Hypertext Transfer Protocol (HTTP/HTTPS) requests, the constructing of URIs for data identification and the usage of reserved URI query string operators. In order to gain access to the API, full authentication is required. ODATA uses an URI based on three parts (Figure 19):

- the Service Root URI identifies the root of the OData service;
- the Resource Path defines the collection entity to look into (e.g. DeletedProducts);
- the Query Options used to filter the results.

Figure 19. An example of an ODATA request

However, hubs using ODATA do not share the same format, parameters, entities or query options. For example, the selection of products created from 1 January 2020 has different approaches in Open Access Hub and ONDA Dias as presented below:

- https://scihub.copernicus.eu/DHuS/odata/v1/Products?$filter=CreationDate gt datetime '2020-01-01T00:00:00.000'
- https://catalogue.onda-dias.eu/dias-catalogue/Products?$search="creationDate:[2020-01-01T00:00:00.000Z%20TO%20NOW]"

Regarding Rest, the other most used API, it is defined as a catalogue that aims at handling not only Earth Observation satellite imagery, but also a kind of metadata. Many projects have used resto, such as CREODIAS, PEPS, Remote Sensor Technology Center of Japan (EPIC project) , Sentinel Australia Regional Access, Sinergise sentinel-hub OpenSearch API etc. An example of resto request is the following: https://peps.cnes.fr/resto/api/collections/S1/search.json?collection=S1&page=1

3. ARCHITECTURE

3.1. System Architecture

The goal of the system architecture is the design of a single point of access for Sentinel metadata by searching and collecting them from all the available hubs, DHuS based or not. The system overview depicts the various ways in which the developed Umbrella hub functions and is accessible by the user. As shown in Figure 20 there are three main processes that take place in order to eliminate the aforementioned limitations: i) searching hubs for new metadata, ii) scoring hubs based on performance, and ii) deleting unavailable metadata.

The searching process takes place multiple times per day, in particular every fifteen minutes in order to harvest new metadata ingested in the hubs. Sentinel missions have a revision time from one to five days, which implies that the different hubs are populated with newly ingested products daily. Sentinel data are made available to the hubs usually two to twelve hours post sensing. Thus, the update frequency of fifteen minutes was chosen as an optimal so as to capture all these newly ingested metadata. This frequent update of metadata is crucial for near real time applications that are dependent on the immediate acquisition of the required data, one such application could be the flood extent mapping (PUC1 Product). Moreover, the more frequent the searching process is the shorter the execution time; with the process being dependent

on the number of products to be harvested. The process is thoroughly described in section 5.2.1.

In addition, metadata are deleted due to several reasons, including the respective hub's archive rolling policy. Thus, the system will clear all the records that do not longer exist at the hubs.

Moreover, the scoring process identifies the most efficient hub at a particular instance. The process is executed every ten minutes and it checks the availability of each hub and measures the download speed for the same type of products in order to rank the hubs. As there is a time variability concerning the download speed of certain hubs, such as the Copernicus Open Access Hub, the scoring frequency of ten minutes allows for the frequent status of each hub. This leads to the optimal selection for the most efficient hub. Having a short time scoring frequency ensures that the last measured download speed is indeed representative of the current download speed. Consequently, when a user asks for a product, they get metadata from the most appropriate - at this time - hub.

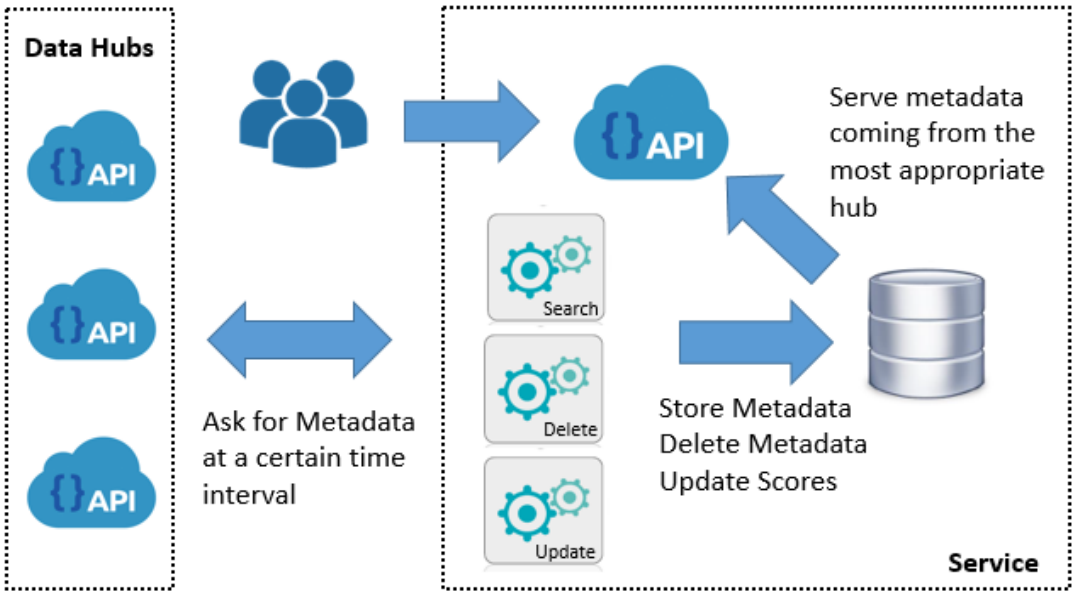

Figure 20. System Architecture.

Finally, as shown in Figure 20, the system provides via an API the collected Sentinel metadata by allowing users to perform requests on the database resources and get the download link for the required products.

4. TECHNOLOGIES

The proposed design consists of two layers: the database layer and the application layer. In order to implement these layers, a thorough evaluation was performed by reviewing best practices and by taking into accounted potential architectures. Thus,

the choice of the employed technologies was based on the best combinations and trade-offs among efficiency, reusability, and suitability.

4.1. Application Layer

As Figure 21 reveals, the application layer interacts not only with the hubs by sending requests and handling responses easily, but also with the users accepting requests and serving metadata. At the same time, the source code of the application layer had to be written in a language that is able to handle multi-dimensional containers of generic data.

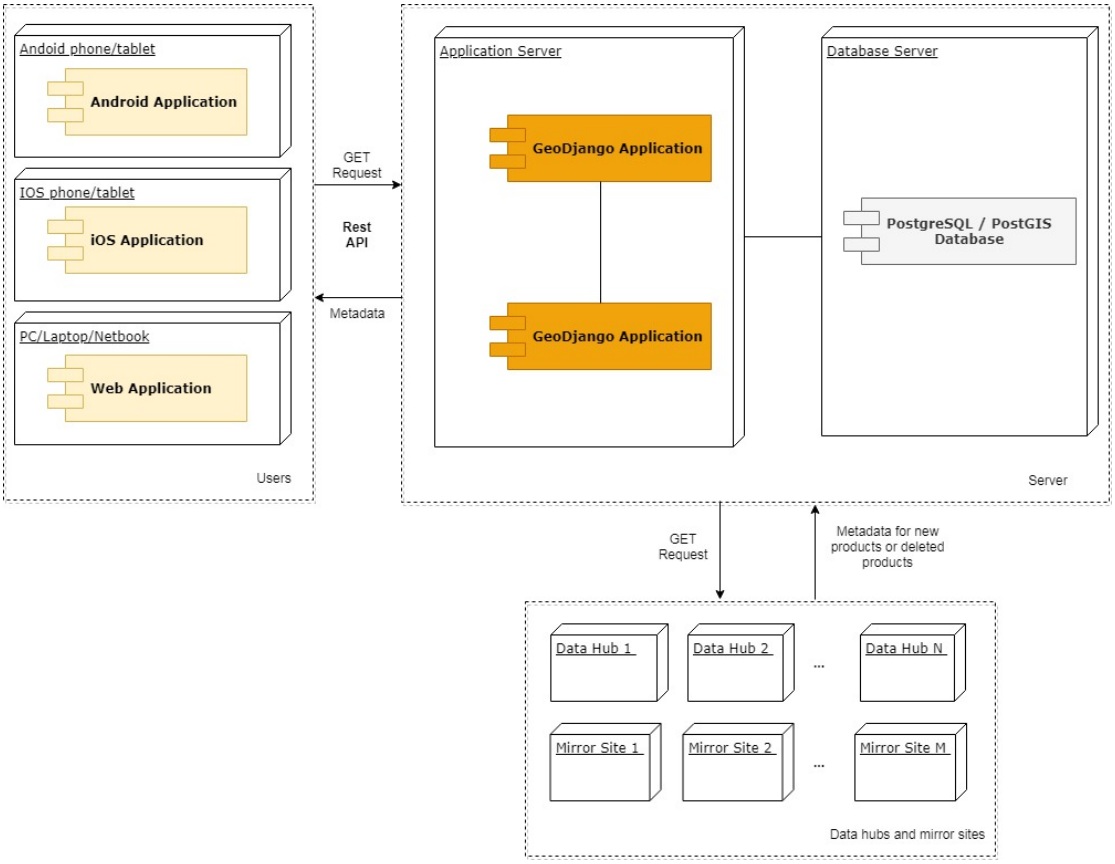

Figure 21. Overall system architecture.

Taking into consideration all the above, **Django**[18] was selected. Django is an open source web framework for Python. It provides a high-level abstraction of common web development patterns (Holovaty and Kaplan-Moss, 2009). Django comes fully loaded with libraries related to user authentication, site maps, et cetera. These functionalities help to automate any process without specifically writing new code.

[18] https://www.djangoproject.com/

Moreover, Django provides the **GeoDjango**[19] module, which is essential in EOPEN, as it provides geospatial functionalities. For instance, our metadata include a field related to their footprint. This field must be added to the database as a geometry, which can easily done by using the GeoDjango tools.

Finally, **Django REST framework**[20] was selected for the Umbrella API. Representational State Transfer (REST) is an architectural model that is used to design distributed software architectures based on network communication. Responses of the selected API type have to be in eXtensible Markup Language (XML), JavaScript Object Notation (JSON), Yet Another Markup Language (YAML), or any other format depending on what the client requests. Moreover, it has to be stateless, meaning that requests can be made independently of one another, and each request contains all of the required data to complete itself successfully. The Django REST framework is powerful and flexible for building Web APIs, allowing filtering and easy data serialization.

4.2. Database Layer

Figure 21 illustrates the database layer that is used for storing the harvested metadata. This layer demands a database management system (DBMS) that performs well in complex queries on a great number of records. In addition, database management systems must support geographic objects so it can be used as a geospatial data store for location-based services and geographic information systems.

PostgreSQL[21] is an open source relational database management system, which has proven very reliable in use (Obe and Hsu, 2011, Stones and Matthew, 2006). As PostgreSQL is free to use with no licensing costs, it allows for the scalability of the database size. In addition, GeoDjango is fully compatible with PostgreSQL, also supporting JSON type.

PostGIS[22] is a spatial database extender for PostgreSQL object-relational database. Geographic objects are added allowing location queries to be run in SQL. PostGIS adds extra types (geometry, geography, raster and others) to the PostgreSQL database. Functions, operators, and index enhancements that apply to these spatial types can also be added (Obe and Hsu, 2011). These extra functions, operators, index bindings and types empower the core PostgreSQL DBMS, upgrading it into a fast and robust spatial database management system.

[19] https://docs.djangoproject.com/en/2.2/ref/contrib/gis/

[20] https://www.django-rest-framework.org/

[21] https://www.postgresql.org/

[22] https://postgis.net/

Concerning the interaction among the databases, **Django Models API** allows a simple way of creating and interacting with them. Most common databases are programmed with some form of Structured Query Language (SQL), however each database implements SQL in its own way. Django Models API provides an Object-relational Mapping (ORM) to the underlying database. ORM is a powerful programming technique that simplifies tasks related to data and relational databases. In Django, the model is the object that is mapped to the database. At the time a model is created, Django executes SQL for the creation of the corresponding table in the database while there is no need for the user to write any SQL code. Django prefixes the table name with the name of the Django application. In addition, the model links related information in the database.

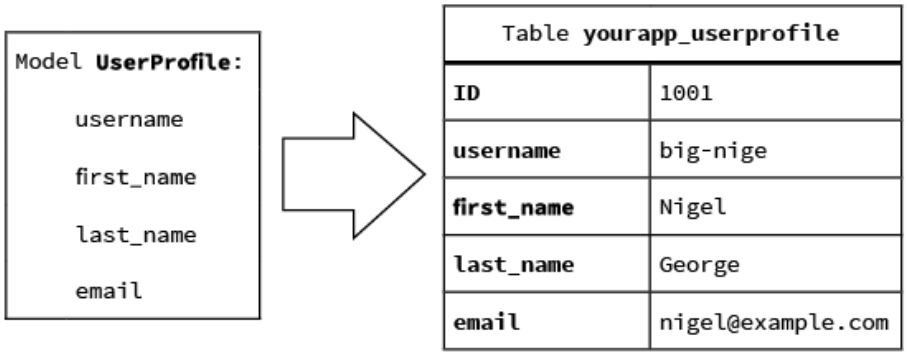

Figure 22. Django Model to Database table.

5. IMPLEMENTATION

The implementation of the Umbrella hub can be split into two different modules: the database module (IDLE mode) and the application module (user request mode).

As this Umbrella hub saves metadata from the aforementioned hubs, there is the need for a database system. The database stores not only information about the metadata, but also information about the hubs. Metadata information is the same with the one on the source hubs, e.g. polarization for Sentinel-1 or cloud coverage for Sentinel-2. Moreover, the database is populated with new data so that users have access to newly ingested metadata as fast as possible. Thus, a searching application is executed every fifteen minutes. At the same time, hubs delete their data due to either their rolling archive policy or due to several other reasons, such as duplicate, corrupted products etc.. The Umbrella hub has to update the metadata in the database by deleting them when they are deleted from the hubs. Therefore, the delete process takes place once per day to prevent users from attempting to download a product that is no longer available.

The greatest advantage of this Umbrella hub is that it provides users the download link from the most appropriate hub for the requested product. This is achieved by a scoring process which runs every ten minutes and checks the availability of the hubs.

If the hub is available, it also measures its download speed of online products. Offline products can be requested, but cannot be checked immediately for performance as they become available at a later stage. Hubs get points based on their availability status and download speed. The total score of each hub is stored in the database. Thus, when users make a request to the Umbrella hub API, they get as a response the metadata that stem from the hub with the highest score and in which the products are available. In this way, users take advantage of the best combination of the product-hub pair. At the next sections there is further elaboration on these two modules.

An alternative approach would be the scores of the hubs to be computed every time a user makes a request. But this would add additional time to respond to the user request, as the scoring process needs several seconds to execute. In addition, as multiple users may make requests at the same time, there is the need for storing multiple users' credentials in the database. This requirement is related to the fact that only one authenticated request is allowed per session.

5.1. Database Design

Once the hubs and metadata were analysed, the next step was to design the database. A smartly designed database structure is able to improve the efficiency of data storing, to ensure the data integrity, and consistency. As Figure 23 illustrates, metadata are divided into four different tables corresponding to the four Sentinel missions, for which the searching process identifies metadata. Each mission table lists the different attributes associated with the respective sensors. Hence, the metadata are organized in a logical manner, similar to the way in which the real-world objects that the data represent are organized. In addition, the API becomes more user friendly as users can access each mission as a separate catalogue.

In the scenario that more hubs connect to the Umbrella API, there is no need for time consuming parameterization of the source code, as new information related to these hubs are easily inserted to the database via an administrator form allowing simple and fast integration.

The great amount of metadata that is ingested daily by adding more hubs has been taken into consideration during this design. As most hubs keep products that have been sensed within a specific period of time (i.e. HNSDMS keeps records of products sensed in the past month) and delete older ones, we have developed a delete process to make sure that our database is up to date. In the case of Copernicus Open Access hub, the archive of available products is very long, however it keeps a list of deleted

products, allowing the Umbrella hub to easily delete the appropriate products, without having to go through its entire archive. See more details on section 5.2.1.

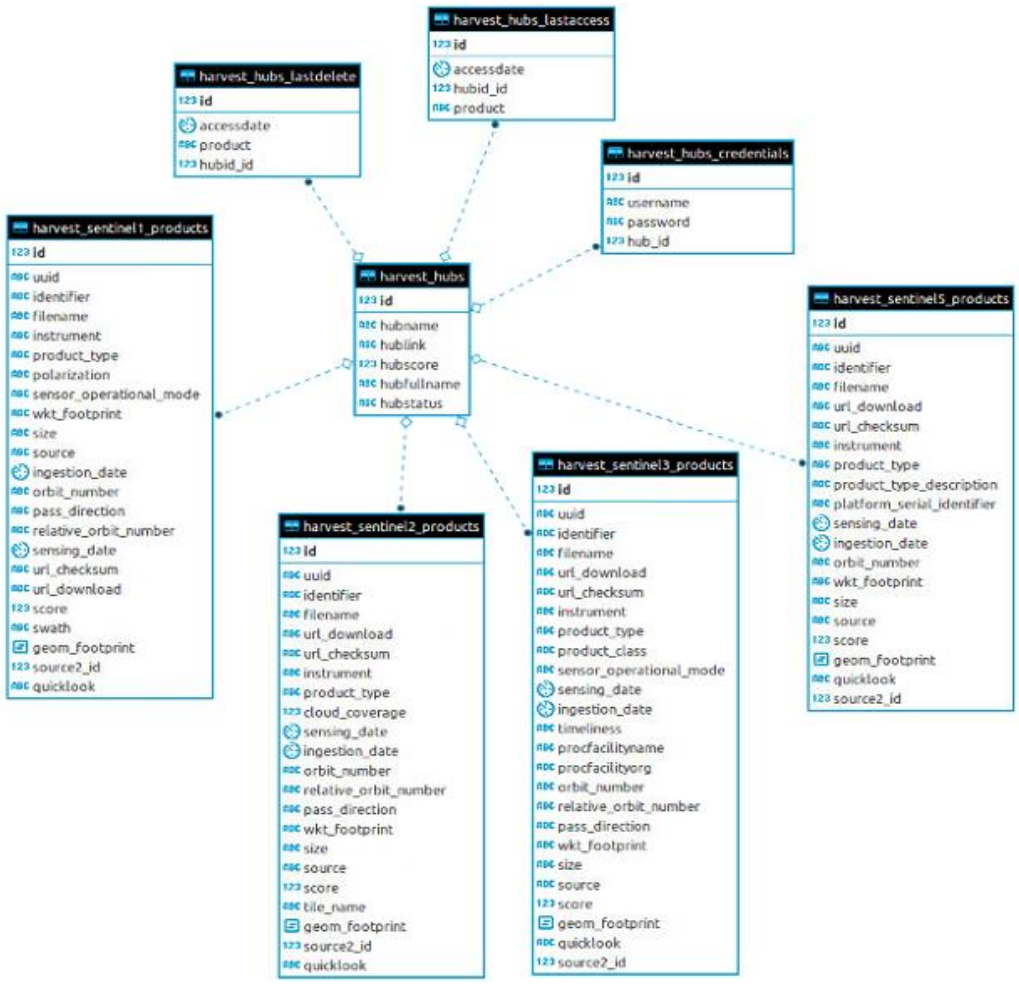

Figure 23. Database Diagram.

5.2. Django Applications

Currently users can access one of the many hubs where the requested product does not exist, either because it has been deleted or because it has never been ingested due to geographic restrictions. In addition, users are facing problems related to download speed as they are hubs, such as Copernicus Open Access Hub, from which the products are being downloaded with low rate.

5.2.1. IDLE Mode Application

The **search process** gets metadata from all aforementioned APIs and gathers them together in a single database. The search process executes parallel tasks and each of them searches for the newly ingested metadata in each connected hub, starting from the previous searched date time up to the current date time. Taking advantage of this parallelization the search process is straightforwardly scalable, assuming the existence

of the required sources. In order to access the different hubs, authentication is required. Thus, credentials are stored to the database and are retrieved when needed. These credentials are set up by the developers in order to have access to the hubs. Users must set their credentials only for downloading the required products. For example, if Umbrella hub identifies that Onda DIASis the most efficient hub for a requested product in a particular instance, a download link to this product is provided to the user. This link can be accessed if only user has been authorized to access the aforementioned hub and all the hubs the Umbrella application is connected to. Afterwards, a dedicated URL – query is constructed for requesting the required metadata, coming from the selected Sentinel satellite, at a specific time range, in JSON format.

```
url = sci_url
url += '/dhus/search?q='
url += 'ingestiondate:[' + starting_date + ' TO NOW] '
url += 'platformname:Sentinel-1 '
url += '&format=json'
```

Figure 24. Example of building a query String used in a request to Copernicus Open Access Hub API

Triggering scripts send two requests - one to the whole product catalogue and one to the dedicated URL (subset of the product catalogue). If there is a successful response status from both of them, metadata are stored in a dictionary. Multiple requests to the dedicated URL must be made due to pagination. The results do not appear in a single page, but only a subset of them. For example, Copernicus Open Access Hub has a pagination of 100 results per page. If the process is completed successfully, metadata are stored into the database and along with the searching date time in order to use this as the start date timestamp for the next search. It is noteworthy that requests to non DHUS-based hubs' APIs have a different format than the one that depicted above. Therefore, an analysis of the structure, the searching and the filtering parameters has been conducted on each of these hubs (Chapter 2).

The **deletion process** keeps the database updated on the currently available products. The different hubs have a different deletion mechanism, and thus we had to identify the products that have been deleted either due to the rolling archive policy or to other reasons such as corrupted products etc. However, the deletion policy is different for each hub. For instance the Copernicus Open Access hub has a catalogue of deleted products that we use to update the database so as not to go through the entire catalogue and hence save time. On the other hand there is no such catalogue for the Greek mirror site, which archives products only for one month. Therefore in this case we can use these products to compare them with our database and see if any of those have been deleted. The final option is to delete products stored in the Umbrella hub's database after the retention period for each hub. The last process demands the exact knowledge of the rolling policy. Taking into consideration the above, deletion process is divided into three sub processes.

The first one searches on every hub that provides a catalogue of deleted products. The script makes a request to the API's catalogue and gets all the deleted products. Afterwards, a dataset is generated from the intersection between the stored products in our database against the harvested deleted metadata. The resulted intersection set determines the metadata that are going to be deleted from the database. It should be noted that PEPS does not accept queries based on online status of the products despite the fact that metadata does have this kind of attribute. Therefore, the discrimination between online and offline products takes places after the retrieval of the API response. Moreover, the second process looks into the hub's product catalogue and saves all metadata. Then, the set of harvested metadata is compared to the set of the stored metadata in the database coming from the database. The intersection of the two sets are kept in the database, whereas the non-common metadata is deleted. . The third sub process checks whether metadata related to a certain hub must be deleted from the local database based on the sensing date, checking the slide window period of the hub.

This process is also straightforwardly scalable due to the fact that hubs are checked for unavailable products in parallel.

The **scoring process** is crucial, as the download speed of the hubs is often low and significantly fluctuating due to a series of reasons. Currently, users are not able to find the appropriate hub that not only has the products they need but is also available and with high download speed. The Umbrella hub gives the solution to such issues. Having all the metadata in the local database gives the potential to provide products coming from the most appropriate hub based on the availability and measured download speed at the time of request. The scoring process is executed every ten minutes, checking not only the status of the data hub but also the download speed. The chosen time interval is based on the high variability of Copernicus Open Access hub download speed (significant fluctuations from hour to hour), while the HNSDMS and the Finnish Hub had nearly constant download speed during our tests.

To check the hub's availability, a request is sent to the products catalogue of the API for relevant products and size. If the response status code is related to an unsuccessful request, then the hub's score is zero. In case the response status code is listed as successful, then a download test is queued by randomly selecting metadata of a certain type of product from the postgresql database. The idea is to start the download of the product from each source and count the downloaded bytes during a fixed time interval of five seconds to measure the available bandwidth. Finally, each of the N hubs is sorted based on its download speed; the hub with the highest download speed scores N points, the one with the second highest download speed scores N-1 points etc. The updated scores are stored in the postgresql database. Having these scores the application easily provides users the best hub to download from at a given time.

Scalability can be achieved here in the following way: On each test, requests in hubs are sent in parallel and hubs' download speeds are stored in the database. Then, these speeds are sorted and the most efficient hub can be selected.

5.2.2. API Application

Anyone who needs to search and download Sentinel data uses either the user interface that each hub provides (DHuS), or an API in which he can send query requests. These requests consist of parameters that identify the requested products, such as the footprint, the mission etc. These APIs respond with lists containing metadata of each requested product. At the next step, users can download the products by getting the appropriate download links. The Umbrella API follows the same process.

This application makes use of the results the IDLE mode application generates, retrieving them from the database. The API application creates a REST API via Django views module and allows users to make GET requests to it. In this request, the users are able to define their parameters based on their needs. The Umbrella hub gives back a result set that contains metadata from the most efficient source to download from.

As the user makes a request query, the application finds on the fly all the available tuples {products, hub site} and serves the metadata from the highest scored source. This application also gives the potential for spatial queries by using `django.contrib.gis.geos` and `rest_framework_gis` libraries. Figure 25 depicts the process generated upon a user's request.

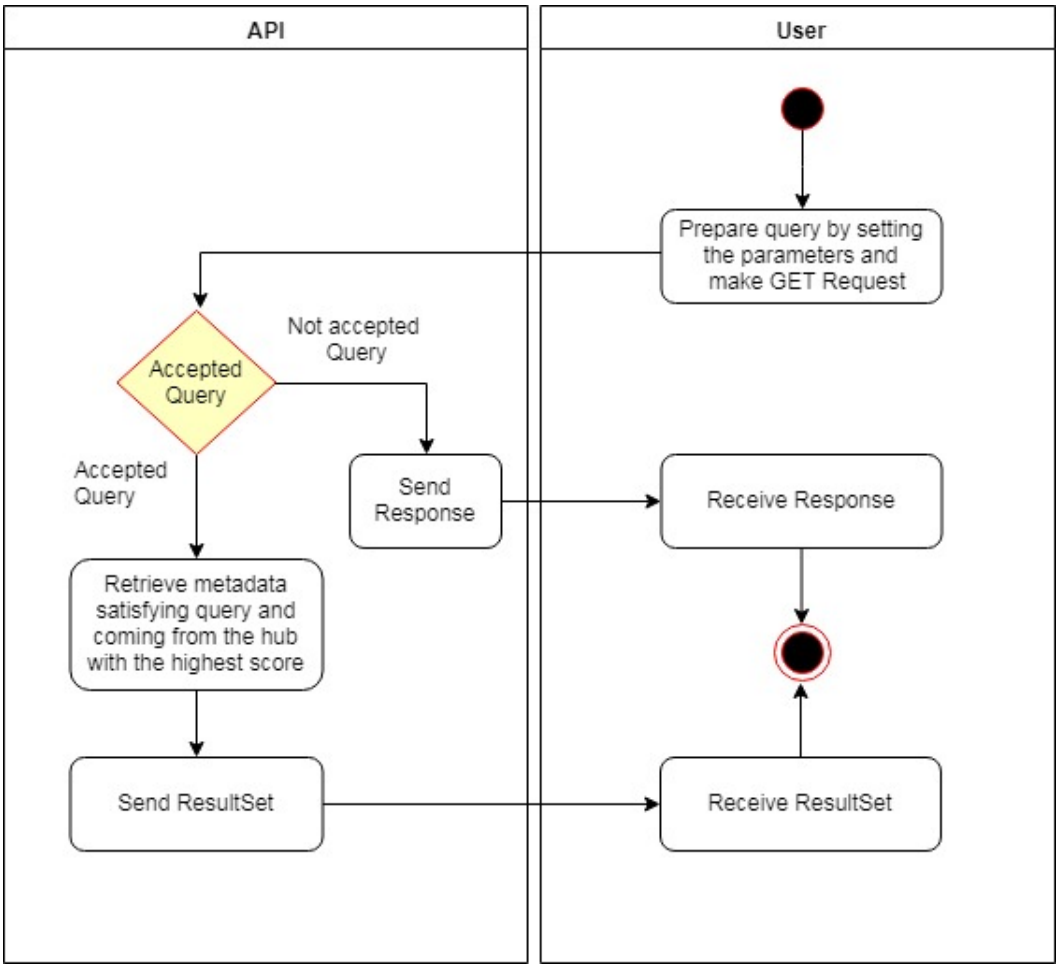

Figure 25. Activity Diagram for User Request.

Scenario: A user wants to get all Sentinel-2 products for the summer of 2019 over a specific area of interest (bounding box). The request will be the following:

```
https://proto2.eopen.spaceapplications.com/eocatalogue/products/sentinel2?in_bbox=20.8,38.41,23.82,40&sensing_date__gte=2019-06-01&sensing_date__lte=2019-08-31&format=json
```

6. INTEGRATION WITH THE PLATFORM

After the application was locally tested, the next step was to integrate it within the EOPEN platform. The platform gives the potential for other deployed applications to exploit the Umbrella data hub. These applications can be either applications inside the platform or any other application which provides metadata by requests to the API. Concerning the integration, any application can be deployed either as a service or as a process. Services run constantly without the need for a triggering mechanism, so as to be accessible anytime from the users. On the contrary, processes contain one or multiple scripts, which demand a triggering event to be executed.

6.1. Services

Firstly, the database was set up with all ingested data coming from 2018 (for all Sentinel missions) – it will be populated with all available data at a later stage. For the import of the database into the platform, a backup file was required, containing the structure of the database along with the relations among the database objects and the records already stored in the tables. This file was then imported into the postgresql server.

The next step was to push the Django project as a Docker image into a platform's repository so that it could be placed into a Docker container. A Docker container can be seen as a computer inside another computer in order to avoid having problems related to the transfer of projects from local to production. Among other assets, Docker allows users to wrangle dependencies starting from the operating system up to details such as Python package versions and ensures that user's analyses are reproducible.

```
1  FROM python:2.7
2  ENV PYTHONUNBUFFERED 1
3  RUN mkdir /code
4  WORKDIR /code
5  ADD requirements.txt /code/
6  RUN pip install -r requirements.txt
7  ADD . /code/
8
9  CMD ["python", "/code/manage.py", "runserver", "0.0.0.0:8000"]
```

Figure 26. Dockerfile used for creating the Docker image.

Finally, since the service has been initiated, any platform process has access to the database and the API.

6.2. Processes

As mentioned, the Django project consists of three principal modules: searching, deleting and scoring. These applications must stand as independent processes within the platform, so that they run in different time intervals. In order to convert these applications into processes, EOPEN Platform provides a process wrapper template generator. This generator allows users to enter basic information about the process related to name, description etc. The most important part is that the input and output parameters are defined along with the required libraries.

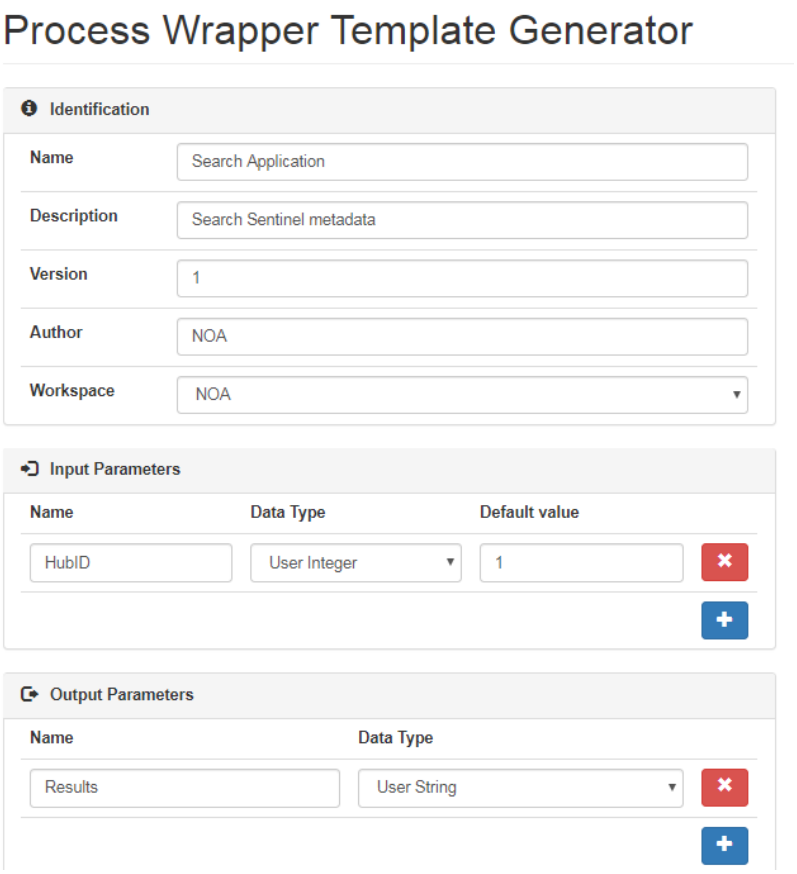

Figure 27. Process Wrapper Template Generator.

The Process Algorithm Importer of the platform automatically containerizes the uploaded file and creates a process. All created processes are available to be added into a processing chain, which is called processor. For example, as we need to search at the same time tenhubs, our processor consists of the same process in tenreplicas but with different input parameters, namely the hub IDs.

As the score, search and delete processes must be triggered in different time intervals, ten minutes, fifteen minutes and twenty hours respectively, a scheduler must be set up. A scheduler module within the platform gives the potential to set the time interval in which each process is executed.

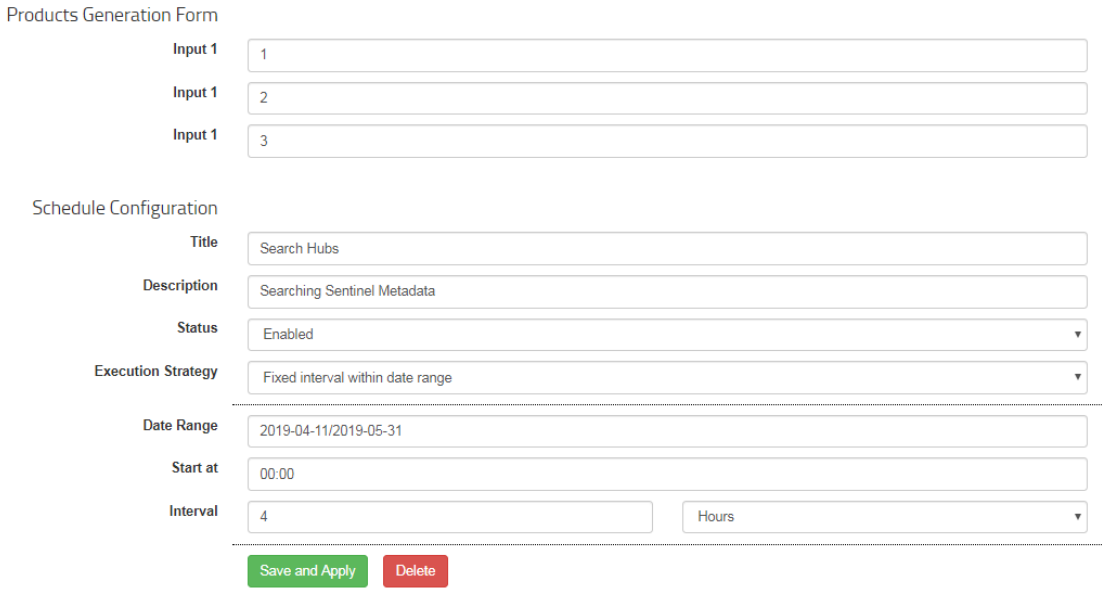

Figure 28. Process Scheduler.

Recent executions section on the platform's interface allows users to inspect the status of each execution. Moreover, there are log files which show the errors, inputs and outputs of each process.

Requesting user	Workspace	Processor	Version	Request time ▲	Parameters	Status	Execution Report
NOA	noa	Umbrella_App_Scores	1	Sat, 20 Apr 2019 09:30:07 GMT	Show	✔ Generated	Execution Report
NOA	noa	Umbrella_Hubs_App	1	Sat, 20 Apr 2019 09:16:18 GMT	Show	✔ Generated	Execution Report
NOA	noa	Umbrella_App_Scores	1	Sat, 20 Apr 2019 08:30:07 GMT	Show	✔ Generated	Execution Report

Figure 29. Recent Executions.

7. USER GUIDE

This section guides users on how to interact with the Umbrella hub API. Users are able to use this API in order to get metadata without any geographic restrictions based on their needs and having the best possible performance for downloading the products. The EOPEN Web API is based on REST principles (Massé, 2012). Data resources are accessed via standard HTTPS requests in UTF-8 format to an API endpoint. The API returns all response data as a JSON object.

The requests consist of the mission catalogue (Sentinel-1, Sentinel-2, Sentinel-3 and Sentinel-5 catalogues) and the parameters used to query it.

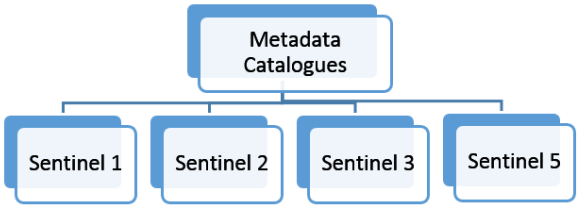

Figure 30. Catalogues of Umbrella hub API.

Users encounter these parameters in order to get the most relevant results related to their needs. Each satellite comes with different parameters. However, certain parameters are common. The general form of a request is https://proto2.eopen.spaceapplications.com/eocatalogue/products/<mission>?&format=json. For example, if the mission is Sentinel-1, the request has the form: https://proto2.eopen.spaceapplications.com/eocatalogue/products/sentinel1?&format=json

The Table 2 shows every parameter that can be used to build the query along with the catalogue that can be applied to.

Table 2: Umbrella hub API request parameters.

	Sentinel-1	Sentinel-2	Sentinel-3	Sentinel-5
relative_orbit_number	Yes	Yes	Yes	Yes
sensing_date	Yes	Yes	Yes	Yes
product_type	Yes	Yes	Yes	Yes
in_bbox (xmin,ymin,xmax,ymax)	Yes	Yes	Yes	Yes
filename	Yes	Yes	Yes	Yes
cloud_coverage	No	Yes	No	No
polarization	Yes	No	No	No

The distinct values some of these parameters can take along with the operators that can be applied to them and examples of them are revealed on APPENDIX A – Umbrella Hub API query parameters.

Finally, the endpoint supports a way of paging the dataset, taking an offset and limit as query parameters:

https://proto2.eopen.spaceapplications.com/eocatalogue/products/sentinel1?sensing_date__gte=2020-06-

01&in_bbox=20.8,38.41,23.82,40&limit=100&offset=200&format=json

In the example above offset means that starting from the 200th result and users retrieve the next 100 (`limit=100`) results.

8. CONCLUSIONS

Open satellite data coming from Sentinel missions have opened the gates to newer applications related to many scientific areas. These data can be found free on several hubs with differences related to the available satellite missions and the geographic coverage. In this deliverable, we have presented the first version of an Umbrella data hub, which takes into consideration users' demands for searching and downloading satellite data. This application provides a more efficient service to gain access to Sentinel data. The current situation related to searching Sentinel data is rather complicated having many limitations set by the individual characteristics of the different hubs, such as geographic coverage, deletion policy, mission availability, download speed, hub availability etc. In this context we have presented the need and the implementation of a single point of access. In addition, this version addressed the comments of the 2nd review by presenting a detailed analysis of a significant number of hubs and the selection criteria for the connected one. It is also made apparent that Umbrella hub application brings a new era to the Sentinel access points as it provides a more flexible architecture than the alternative Sentinel Linker Service by allowing the connection of both DHuS based and non-DHuS based hubs.

9. REFERENCES

Aalto, J., Pirinen, P., & Jylhä, K., 2016. *"New gridded daily climatology of Finland: Permutation-based uncertainty estimates and temporal trends in climate."* Journal of Geophysical Research: Atmospheres, 121, 3807-3823.Copernicus Climate Change Service (C3S), 2017. *"ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate"*. Copernicus Climate Change Service Climate Data Store (CDS), Accessed 17 Apr 2019. https://cds.climate.copernicus.eu/cdsapp#!/home

Holovaty, A, and Kaplan-Moss, J., 2009. *"The definitive guide to Django: Web development done right",* Apress, pp. 3-74.

Luojus, K., J. Pulliainen, M. Takala, M. Kangwa, T. Smolander, A. Wiesmann, C. Derksen, S. Metsämäki, M. Salminen, R. Solberg, T. Nagler, G. Bippus, S. Wunderle, F. Hüsler, 2013. *"GlobSnow-2 Product User Guide Version 1.0*", Available online at http://www.globsnow.info/swe/GlobSnow2_SE_SWE_Product_User_Guide_v1_r1.pdf

Massé, M. 2012. "*REST API Design Rulebook",* O'REILLY, pp. 85-91

Obe, R. O. and Hsu, L. S., 2011. "*PostGIS in action*", Manning Publications Co., pp. 13-14

Rautiainen, K., Parkkinen, T., Lemmetyinen, J., Schwank, M., Wiesmann, A., Ikonen, J., Derksen, C., Davydov, S., Davydova, A., Boike, J., Langer, M., Drusch, M., and Pulliainen, J. 2016. *"SMOS prototype algorithm for detecting autumn soil freezing*". Remote Sensing of Environment, 180, 346-360. DOI: 10.1016/j.rse.2016.01.012

Stones, R and Matthew, N. 2006. *"Beginning databases with PostgreSQL: from novice to professional",* Apress, pp. 11-16.

World Meteorological Organization, 1992. *"International Meteorological Vocabulary"*, WMO – No. 182. pp. 784. ISBN 978-92-63-02182-3

10. APPENDIX A – Umbrella Hub API query parameters

Product Type Parameter (product_type)	
Distinct Values	RAW, OCN, SLC, GRD (Sentinel 1) / S2MSI2Ap,S2MSI2A, S2MSI1C (Sentinel 2) / SY_2_SYN___,SR_1_SRA_A_,SR_2_LAN___,SL_2_LST___,SY_2_VG1___, OL_2_LRR___,OL_2_LFR___,OL_1_EFR___,SY_2_VGP___,SL_1_RBT___, SR_1_SRA_BS,SR_1_SRA___,OL_1_ERR___(Sentinel-)
Operators	= (equals) / icontains= (containing case insensitive text) / icontains != (does not contain case insensitive text)
Example	/products/sentinel1?product_type=GRD&format=json /products/sentinel2?identifier__icontains=S2A&format=json
Polarization Parameter (polarization)	
Distinct Values	HH, VV,HV,VH,VH VV,HH HV,VV VH
Operators	= (equals) / __in= (value in list of values)
Example	/products/sentinel1?polarization=HH& product_type=GRD &format=json /products/sentinel1?polarization__in=VH,HH&format=json
Sensing Date Parameter (sensing_date)	
Distinct Values	
Operators	= (equals), __lte=(less than or equal), __lt=(less than), __gt=(greater than or equal), __gt=(greater than), __range=
Example	/api/products/sentinel1?sensing_date__range=2019-02-15,2019-02-22 &format=json
Relative Orbit Number Parameter (relative_orbit_number)	
Distinct Values	1-175 (Sentinel 1) / 1-143 (Sentinel-2) / 1-442 (Sentinel-3)
Operators	= (equals), lte(less than or equal), lt(less than), gte(greater than or equal), gt(greater than)
Example	/api/products/sentinel1?relative_orbit_number__gt=100&format=json
Area of Interest Parameter (in_bbox)	
Distinct Values	
Operators	= (equals)
Example	/api/products/sentinel1?in_bbox=-90,29,-89,35&format=json
Cloud Coverage (cloud_coverage)	
Distinct Values	0-100
Operators	= (equals), __lte=(less than or equal), __lt=(less than), __gt=(greater than or equal), __gt=(greater than), __range=
Example	/api/products/sentinel1?cloud_coverage__lte=20&format=json

Table 3 - Query parameters, operators and examples.

11. APPENDIX B – Work done in other tasks

11.1. Meteorological and climatological data acquisition

In WP3, the task 3.3 *Meteorological and climatological acquisition* aims to gather the necessary meteorological and climatological data and provide the data to EOPEN users. In this section, we provide a short overview of the task; deliverable D3.2 (due Month 26) will include a more detailed description of the task and the acquired data.

Here, meteorological and climatological data refer to *information about weather and climate*. According to World Meteorological Organization (1992), **weather** is defined as ”state of the atmosphere at a particular time, as defined by the various meteorological elements”, while **climate** is defined as ”synthesis of weather conditions in a given area, characterized by long-term statistics (mean values, variances, probabilities of extreme values, etc.) of the meteorological elements in that area”.

Below is a list of the main categories of data, which we are considering to use in this task:

- **Weather observations:** This data includes instantaneous weather observations from automatic weather stations (AWS), and statistics for daily, monthly, seasonal, and yearly values. We also consider derived products, such as grid-interpolated timeseries (e.g. FMI ClimGrid; see Aalto et al., 2016), a part of weather observation data.
- **Weather model output:** This data includes both numerical weather prediction (NWP) model forecasts and model reanalyses, for example ERA5 (Copernicus Climate Change Service, 2017).
- **Climatological values:** Long-term statistics of meteorological elements. Typically the statistics are calculated over a 30-year period, for example 1981-2010.
- **Climate model output:** Estimated changes in climatological values over decades, as calculated by specialised numerical weather models.
- **EO-based snow and ground frost related products:** For example, GlobSnow SWE (Luojus et al., 2013) and SMOS Level 3 Freeze/Thaw (Rautiainen et al., 2016)

Current EOPEN Framework in meteorological and climatological data acquisition

Currently we’re mapping and defining the weather and climate data needs for the use cases and selecting the potential data sources for the data. The aim is to have all datasets from machine-readable APIs, which follow the industry standards (e.g. RESTful). Machine-readability is desired so that the data can be automatically retrieved from sources to EOPEN platform. In the case where an identified data need cannot be fulfilled via any API, we will consider whether the data can be transferred to EOPEN platform manually. The manual transfer should be done only for datasets, which are critical for any use case, and have infrequent refresh cycles. Manual transfers are considered case-by-case, should a need for it arise.

Table 4 lists the current data sources we've identified and are investigating. We expect the list is not exhaustive, as the meteorological and climatological data requirement mapping is not yet finished. We will consider unlisted data sources, for example NOAA's services, if these sources offer data which is requested by use cases and is unavailable from listed sources. Open data is preferred whenever possible, so that data is easily available for EOPEN users and without usage restrictions.

Table 4: Data sources and their data sets, which are currently under consideration.

Source	Potential datasets	Status	Openess
FMI Open Data	Finnish weather observations; NWP forecasts; Finnish climatology;	Work in progress	Machine-readable API; no registration needed;
Paituli spatial data download service	FMI ClimGrid	API under evaluation	Machine-readable API; Some datasets are not available via API; some datasets are restricted; no registration needed;
KMA OpenAPI	Korean weather observations	API under evaluation	Machine-readable API; registration needed;
C3S Climate Data Store	ERA5 reanalysis	API under evaluation	Machine-readable API; registration needed;
Sodankylä National Satellite Data Centre	GlobSnow SWE; SMOS Level 3 Freeze/Thaw	FTP under evaluation	FTP access; credentials needed for FTP; data unlicensed;

Notably, weather observations from Italy are missing from the data source table. While PUC1 utilises meteorological and climatological data, AAWA obtains Italian weather observations directly from Italian authorities, and thus Italian weather observations are not considered in this task.

11.2. Social Media Crawling

One of the main objectives of EOPEN is to gather information from multiple heterogeneous sources and thus collect both EO and non-EO data. Regarding non-EO data, the focus is on online content from social media and particularly the popular platform Twitter. With the assist of the end users, an appropriate search is performed to crawl social media posts that are interesting to the use cases examined within the project. Collected data are then displayed and filtered, but they can also serve as input to other tasks, including the event detection, the clustering of non-EO content, the similarity fusion and the community detection.

The current implementation of the social media crawling procedure can be seen as a flow in **Error! Reference source not found.**In order to gain access to the global stream o f Twitter data, Twitter API[23] has been selected. This platform offers the free option to stream real-time tweets exploiting filtering capabilities. Filters can be keywords to be found inside the post, identifiers of user accounts and location boxes. Following the end users' suggestions, such filters have been defined for the use cases of flooding (in English and Italian), snow coverage (in English and Finnish) and food security (in English and Korean). These retrieval options are stored in a MongoDB collection and are used as input to the Twitter API, along with unique credentials that are required. In a real-time manner the API constantly retrieves newly created tweets, which satisfy the filtering criteria, in a JSON format. Since all filter options necessarily form a single query, a reverse process is needed to find which use case and which language the new tweet has been matched to.

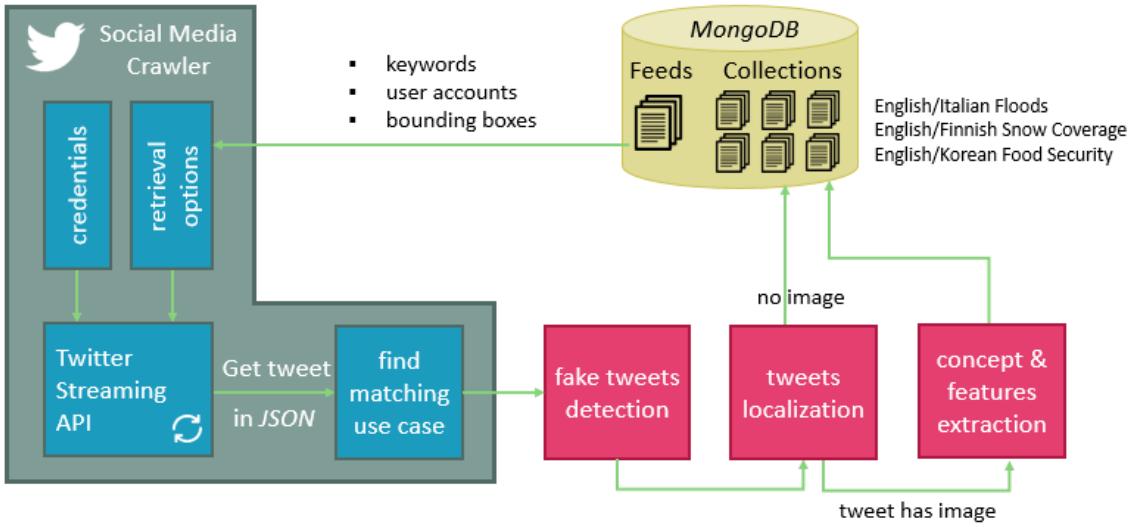

Figure 31. Complete flow of the Social Media Crawling.

Before storing the crawled tweets, an additional analysis is performed to enhance the data or check their quality. Firstly, tweets are automatically classified as fake or real

[23] https://developer.twitter.com/en/docs/tweets/filter-realtime/overview

by a ready solution developed in CERTH [] in order to overcome the current trend of hoax news in social media. Then, a localization methodology tries to detect the locations mentioned inside the posted text and assign corresponding coordinates, so as to be able to appear on a map. More details on the localization technique will be provided in the upcoming deliverable D5.1 (M19). If the tweet contains an image, visual concepts are extracted from the image, as well as its feature vector to be later used in the similarity fusion. Concept and feature vector extraction has been introduced in deliverable D4.1 (M16) and will be further described in deliverables D4.3 (M31) and D4.4 (M33). The initial JSON received by Twitter API is updated with all the outcomes of the afore-mentioned analyses and, finally, it is stored to a respective collection in the MongoDB (one collection per use case and language).

At the point of writing, more than 6.5 million tweets have already been collected. The size of each collection and the exact time period of crawling can be seen in Table 5. The database is password-protected and IP-protected and EOPEN takes all possible measures to be compliant with the guidelines and restrictions declared in Twitter's "Development Agreement and Policy"[24]. Sharing the data to third parties is prohibited and regular checks are performed to the collections in order to find tweets that are no longer available on Twitter and thus have to be removed from the EOPEN database too. More details are reported in deliverable D9.1 (resubmitted in M17).

Table 5: Current number of crawled tweets per collection.

Dataset	# English	# Other Language	
Flood events (Mar 2017 – Apr 2019)	6,071,617	Italian	76,768
Snow cover (Dec 2017 – Apr 2019)	44,876	Finnish	35,225
Food security (Dec 2017 – Apr 2019)	454,163	Korean	1,900

The stored tweets can be displayed on a dedicate user interface (Figure 32). Text, image (if existing), and publication date are shown per tweet, while user accounts are pseudonymised. Additionally to this information, users of the interface are presented with the classification of the tweet as real or fake, the detected locations and the extracted image concepts. Available filter capabilities include use case selection, time period of publication, keywords, and the option to filter out retweets, fake tweets, and tweets without images.

At this stage, the EOPEN Dashboard (the main Web platform of the project) contains a page to display and filter tweets, similar to the dedicated user interface, using a copy of the original MongoDB. The next step will be to have the two databases synchronized, while the final goal is to deploy the crawling procedure and all the

[24] https://developer.twitter.com/en/developer-terms/agreement-and-policy.html

analysis modules in the EOPEN platform. Other future works concern the integration of a service to exclude posts with pornographic material and the implementation of two classification techniques -one based on visual and one on textual features- to estimate whether a collected tweet is in fact relevant to the examined use cases.

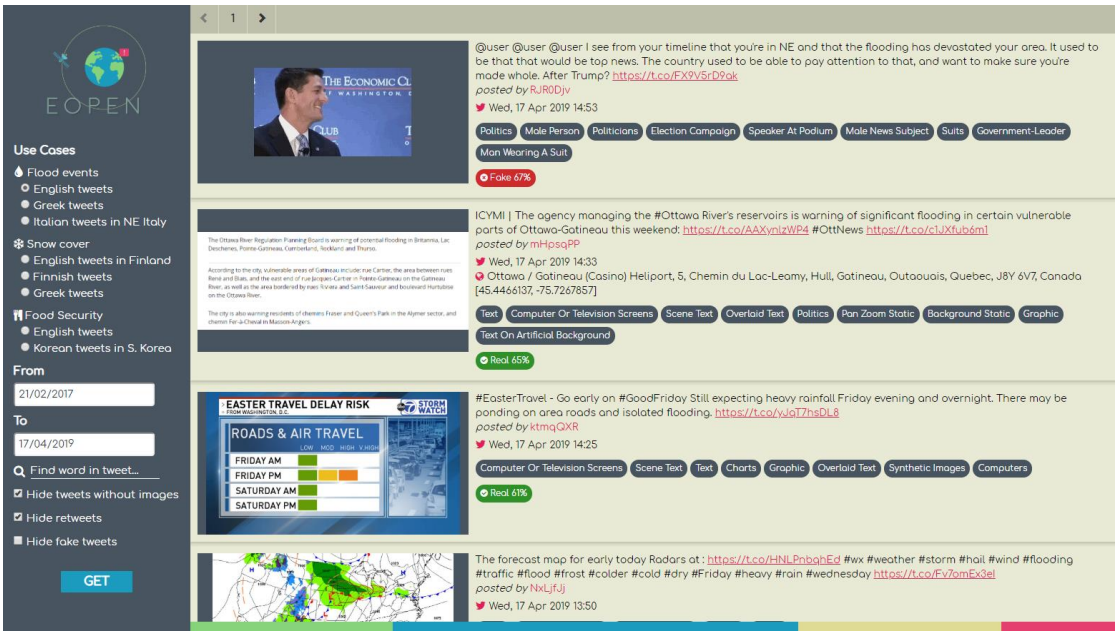

Figure 32. Screenshot of the dedicate user interface to display and filter collected social media.

12. APPENDIX C – NEXTGEOSS "Sentinel Linker Service" VS EOPEN "Umbrella Hub Application"

12.1. Sentinel Linker Service Solution

To fulfill task 3.1 of NextGEOSS Project, it was agreed with the project Coordinator (DEIMOS), and WP/Task leader (DLR), and the platform responsible (TDUE), that NOA will develop a "Sentinel Linker application" to federate Sentinel data available in the rolling archives of Hellenic Mirror Site, CODE-DE, and the CollHub3 (Access is restricted only for specific users) and make them available to the NextGEOSS pilots through the CKAN instance developed in WP2. The application was denominated as "Sentinel Data Linker Service" and was built based on the Data Hub Software (DHuS, https://sentineldatahub.github.io/DataHubSystem) as at the time it has been the main solution adopted by ESA for the operation of Copernicus Hubs and still stands the main architecture for all Copernicus Hubs and several of the Mirror Sites. To be noted that DHuS provides a GUI, which is well known, the users are familiar to using it, a system that is incorporating OpenData and OpenSearch (machine-to-machine interfaces) APIs. For this reason, NOA has been mandated to proceed with DHuS solution for developing the relevant metadata harvester solution. This development in the framework of the NextGEOSS project was completed and delivered to WP2 on 06/2019.

12.2. Umbrella Hub Application.

As time evolved and with the continuous development of Sentinel Data Access Points and Copernicus Hubs and Mirror Sites, and with the advent in the meantime, of the implementation of DIAS platforms, the needs for seamless and timely access to on-line Sentinel Data through additional gateways has been significantly important. Therefore, and as imposed by the EOPEN concept, a need emerged for the development of an interface that is DHuS independent. This new endeavor required development of new APIs from scratch in different time frames than NextGEOSS with the allocation of diverse development skills. In turn this allowed the access to other Hubs than the planned to support the NextGEOSS pilots. More details about systems' functional characteristics and corresponding time schedule for their development are given below (please see Table 6). This latter need led to the development of the EOPEN advanced umbrella solution, which took place in different time periods, with the NextGEOSS solution starting on 1/2019 and ending 06/2019, while the EOPEN has started on 03/2019 and is still ongoing. The main difference between the two solutions is that the NextGEOSS harvester is using the DHuS architecture while the EOPEN solution is based on a more flexible and open source architecture. It can connect to any API that meets certain requirements, ensuring interoperability with different architectures, while the Sentinel Linker application can only connect to DHuS-based Hubs and use the internal synchronizer modules to populate its metadata database. Finally, the overall design of the EOPEN solution was made to be linearly scalable with an increasing number of connected hubs in terms of data storage and execution time.

12.3. Differences between Sentinel Linker Service and Umbrella Hub Application

The two different architectures and applications, the Sentinel Linker Application (NextGEOSS) and the Umbrella Hub (EOPEN), have concluded with the development and delivery of two completely different software with completely different architectures, modules and technologies: the 1st one is a modified DHuS where we have made several changes to its internal architecture, and the second one is an entirely new set-up. Both of them are available to download and experiment with, and we invite you to do so (see apps links in Table 6). This will help appreciate how different the two applications are.

More specifically, what was developed by NOA in NextGEOSS can be found herein below:

1. Containerize DHuS application;
2. Setup and test remote debugging profile
3. New database schema design and deployment by adding new tables in the DHuS DB structure
4. An interface that checks continuously the products and products availability, along with information related to Hellenic Mirror Site, CollHub 3 , CODE-DE, and/or other DHuS based portal performance, and the management of the relevant log files that indicating response time, error messages and the timestamp.
5. Override existing Synchronization and Eviction implementations between the three DataHubs.

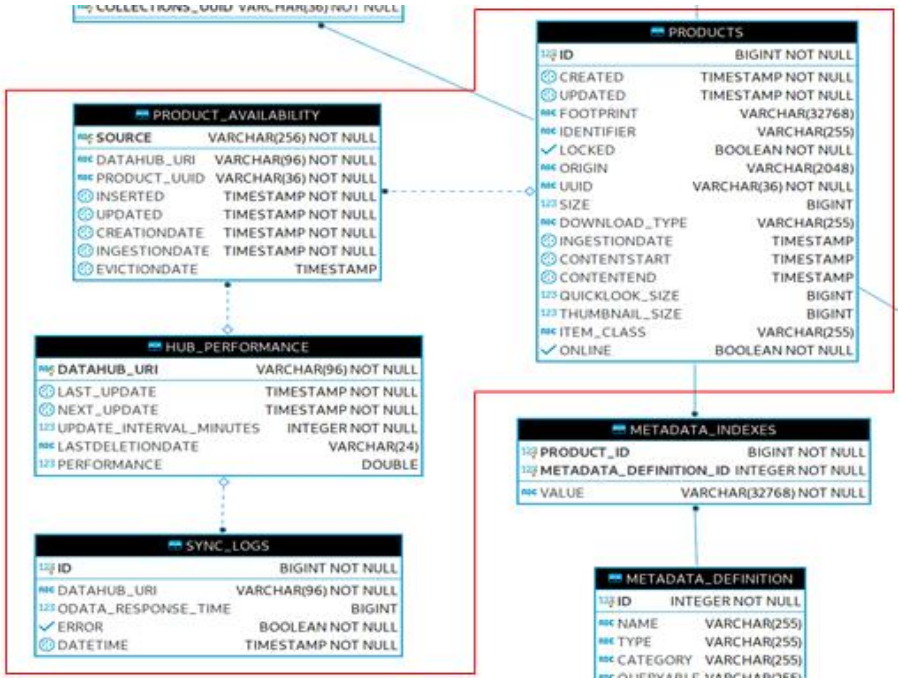

Figure 33. NextGEOSS - Updated Database schema with additional tables

Moreover, the responses received by users is depicted in Figure **34** and provided both in JSON and XML responses.

JSON OpenSearch response:


```
"S1B_IW_RAW__0SDV_20190429T164139_20190429T164211_016024_01E213_B3B9",
"id": "9fc39525-a6ef-44bf-b1bc-ffce0a3ffda0",
"summary": "Date: 2019-04-29T16:41:39.324Z, Instrument: SAR-C SAR, Mode: VH VV,
Satellite: Sentinel-1, Size: 1.5 GB",
"sources": {
"link": [
{
"href": "https://colhub3.copernicus.eu/dhus/odata/v1/Products('9fc39525-a6ef-44bfb1bc-
ffce0a3ffda0')/$value"
},
{
"href": "https://sentinels.space.noa.gr/dhus/odata/v1/Products('9fc39525-a6ef-44bfb1bc-
ffce0a3ffda0')/$value"
}
]
},
```


XML OpenSearch Response


```
<entry>
    <title>S1B_IW_RAW__0SDV_20190429T164139_20190429T164211_016024_01E213_B3B9
    </
title>
    <id>9fc39525-a6ef-44bf-b1bc-ffce0a3ffda0</id>
    <summary>Date: 2019-04-29T16:41:39.324Z, Instrument: SAR-C SAR, Mode: VH VV,
Satellite: Sentinel-1, Size: 1.5 GB</summary>
    <sources>
        <link href="https://colhub3.copernicus.eu/dhus/odata/v1/Products('9fc39525-a6ef-44bfb1bc-
ffce0a3ffda0')/$value"/>
        <link href="https://sentinels.space.noa.gr/dhus/odata/v1/Products('9fc39525-a6ef-44bfb1bc-
ffce0a3ffda0')/$value"/>
    </sources>
    <date name="ingestiondate">2019-04-29T19:24:32.087Z</date>
    <date name="beginposition">2019-04-29T16:41:39.324Z</date>
```


Figure 34. NextGEOSS – Sentinel Linker Service OpenSearch responses

Moving to Umbrella Hub Application, it covers the specific needs, as envisaged in the relevant GA, to address the pilots through a dedicated open platform set by partner: Space Applications. In this regard, NOA has developed and integrated a prototype single data access point deployable on the EOPEN platform. The application provides to the EOPEN platform users, uniform access to Sentinel 1, Sentinel 2, Sentinel 3 and Sentinel 5p metadata via connecting in the back end to a larger (than NextGEOSS) number of the available Sentinel hubs and serving the results via an application programming interface (API).

Specifically, the goal is achieved by the system architecture design allowing Sentinel metadata search and collection from all the available hubs, DHuS-based or not. The system overview depicts the various ways in which the developed application functions and is accessible by the user. There are three main processes that take place in order to eliminate the aforementioned weaknesses: i) searching hubs for new metadata, ii) scoring hubs based on performance, and ii) deleting unavailable metadata.

Finally the system provides via an API the collected Sentinel metadata by allowing users to perform requests on the database resources and get the download link for the required products.

Below you can see the activity diagram and a scenario based on which of a user wants to get all Sentinel-2 products for the summer of 2019 over a specific area of interest (bounding box). The request will be the following: https://proto2.eopen.spaceapplications.com/eocatalogue/products/sentinel2?in_bbox=20.8,38.41,23.82,40&sensing_date__gte=2019-08-01&sensing_date__lte=2019-08-31&format=json

As a summary, the Table **6** presents the main differences of the two applications, developed by NOA.

Table 6. The main differences of the Sentinel Linker Service and the Umbrella Hub Application

	Characteristic of Architecture and concept	DataHubs Harvested	Start of Development	End of Development	APIs used	Deployment
NextGEOSS	The implementation of the Linker mechanism within ESA's DataHub Software works by overriding the standard product synchronization and eviction routines. Every couple minutes, it synchronizes the latest products' metadata, as well as the newly evicted products from the enlisted DataHub Services and updates each products' availability. Every 8 hours (interval that can be changed through settings) a new performance indicator is produced for each DataHub using the performance metrics (response time, availability) which are stored during each synchronization round. The products catalog, as well as their availability, can be accessed using the well-known OpenSearch API that DHuS already uses. Last but not least the products availability list appears sorted using the performance metrics, we mentioned above, within the products metadata.	**Staging:** environment collhub3 & Hellenic Mirror Site **Production**: Collaborative DataHubs using DHuS software (Finnish, Portugal, ApiHub, Schihub, Hellenic, Austrian) & Sentinel-5P Pre-Operations Data Hub) CODE-DE connectors are not working as hub is in an transition period	01/2019	6/2019 (waiting feedback from WP2 to move to the production environment, and start the synchronization with different hubs)	OpenSearch API (interface on top of SOLR) Sentinel Linker Service (staging environment): http://83.212.169.170:8081/search?q=*&format=json **Username**: root **Pass**: password	GRNET(Staging) GRNET (Scalability-mode using external SOLR and PostgreSQL) **Pending waiting feedback from WP2 actions to move to the production environment**

EOPEN	The implementation of the Umbrella hub can be split into two different modules: the database module and the application module As this Umbrella hub saves metadata from several hubs, there is the need for a database system. The database stores not only information about the metadata, but also information about the hubs. Metadata information is the same with the one on the source hubs, e.g. polarization for Sentinel-1 or cloud coverage for Sentinel-2. Moreover, the database is populated with new data so that users have access to newly ingested metadata as fast as possible. Thus, a searching application is executed every fifteen minutes. At the same time, hubs delete their data due to either their rolling archive policy or due to several other reasons, such as duplicate, corrupted products et cetera. The Umbrella hub has to update the metadata in the database by deleting them when they are deleted from the hubs. Therefore, the delete process takes place once per day to prevent users from attempting to download a product that is no longer available. The greatest advantage of this Umbrella hub is that it provides users the download link from the most appropriate hub for the requested product. This is achieved by a scoring process which runs every ten minutes and checks the availability of the hubs. If the hub is available, it also measures its download speed. Hubs get points based on their availability status and download speed. The total score of each hub is stored in the database. Thus, when users make a request to the Umbrella hub API, they get as a response the metadata that stem from the hub with the highest score and in which the products are available. This way, users take advantage of the best combination of the product-hub pair.	Copernicus Open Access Hub, Hellenic Mirror Site, Sentinel-5P Pre-Operations Data Hub, Finnish Data Hub System, Austrian Data Hub System, Czech Data Hub System, PEPS, Onda-DIAS, Norway Collaborative Hub, Rosa Hub	3/2019	Still ongoing 06/2019 – onwards 1st prototype of the application was used from EOPEN's users, receiving feedback and fine tuning activities from NOA Undergoing activities to integrate more non-DHuS hubs	Django REST framework has been used for implementing the Umbrella API., which offers four endpoints, one for each Sentinel mission. The general form of a request is https://proto2.eopen.spaceapplications.com/eocatalogue/products/<mission>?&format=json. In addition to the aforementioned lookup parameters, Django framework allows to use more sophisticated lookups; several parameters can be added to the request query such as the relative_orbit_number, the sensing_date, a bbox et cetera.	SpaceApps platform

12.4. Use case description – Introducing the aim of applications

A use case is presented below that introduces the aim of the different applications developed from NOA within the scope of NextGEOSS and EOPEN.

A research organization (RO) at northern Italy is tasked by the regional Civil Defense authority (e.g. local DPC), to rapidly provide flood delineation maps following flash floods that occur frequently in the region. The requirement is the short timeliness for the delivery of the product, as officers need to make rapid assessments and take decisions. Time is of paramount importance for this application.

The RO has access to a novel algorithm that is fed with Sentinel-1 Synthetic Aperture Radar Data. The algorithm performs fast in its execution, so the time bottleneck is the availability of satellite data. The developer team from the RO have three options for accessing data in an automated way:

1. Create a script that regularly queries Copernicus Open Hub or their National Sentinel Mirror Site.

2. RO deploys the application within one of the available DIAS, and rely on the data that is available in the particular DIAS.

3. Create a script that regularly queries either of our applications (NextGEOSS of EOPEN).

For the three options above these are the pros and cons respectively:

1. (i) The service has a single point of failure, the Open Hub, in terms of availability, (ii) Open Hub experiences the lowest download rate so far, which makes sense since it has the largest user base. (iii) The timeliness of the satellite data (i.e. when they become available to be downloaded by users) relies on the data synchronization setup, from the backend repository where all data lay.

On the plus side, only one account is needed.

2. (i) The service has again a single point of failure to what concerns data access: the selected DIAS, (ii) Copernicus data have now two layers of synchronization to pass through, before becoming available: From the repository back-end —> to the Copernicus DIAS Hub Centers 1,2 and 3 (operated on behalf of ESA by three separate contractors) —> to one of the four available DIAS environments. Therefore, we would anticipate that for this criterion. Open Hub would outperform DIAS Hubs.

On the plus side, download rates are extremely high, since data transfer happens in the internal DIAS network. Again, only one (DIAS) account is needed.

3. (i) The service does not rely to a single Hub for accessing data. It connects to multiple Hubs, and through Hub diversity will have the highest availability (in terms of

data access) for both 1. and 2. above, (ii) The umbrella Hub for example does not sync data, only metadata. As soon as a requested Sentinel product becomes available, our application will become aware. The timeliness of the umbrella Hub is as good as the timeliness of the best performing Copernicus or DIAS Hub for this criterion, (iii) The product will be downloaded each time by the best performing Hub in terms of capacity. However, it will never match the download rates of any of the DIAS hubs that will store the product locally.

On the cons side, our applications will have to use as many accounts, as the number of Hubs it connects to. If we connect to 10 different Hubs then we will need 10 different accounts to be created once following the ESA policy on data download and user tracking.

On the plus side the applications we have developed will be a solution for RO that will be the cheapest, has the best availability in terms of data access, matches the timeliness performance of the best performing Hub, and also secures downloading from the Hub with the highest, at the time, download capacity if the same product exists in more than one Hub.

12.5. Why not use a DIAS platform

The use case provided above is evident of application's aim. Therefore, users should be capable to access any of the existing data access points, and DIAS is only one possible platform, providing access to raw Sentinel data. The applications we have developed do not aim to create services for downloading a lot of Sentinel data. We aim at:

1. Minimizing collective data access downtime, which create bottlenecks in production systems/applications.
2. Achieving the best performance in terms product timeliness. In principle, a product will appear sooner in our Sentinel Linker or umbrella application than it would on a DIAS.
3. Providing a cost-efficient application for accessing data.

12.6. Conclusions

Sentinel Linker Service (NextGEOSS) development came as a follow up to be usable by any platform, fully interoperable, and based on the need that in the meantime new sentinel access points had been available which are only using DHuS. Since then many more platforms became available, but the users still facing the problem of having a fragmented ecosystem of access points, thus users would need guidance to accessing the best hub.

The EOPEN solution is developed so as to access to any hub that is using open standards and is performing many tests on the fly including data integrity, connection speed. These tests contain a scoring process that is executed every ten minutes for checking the availability of each hub. In addition, it measures the download speed for

the same type of products in order to rank the hubs, leading to the optimal selection for the most efficient hub related to a product at a particular instance. Nowadays the developments in EOPEN have established the link with PEPS and Onda-DIAS.

If a user is more comfortable to use the standard UI of DHuS and its functionality, then she will use the Sentinel Linker Application. If a user requires more flexibility and better performance then she will exploit the EOPEN Umbrella Hub. From our side, we will keep maintaining and supporting both applications. As far as ESA keeps having DHuS as its main data access software and most of the Mirror Sites use this as well, the Sentinel Linker Application is useful. When more customized solutions are required, then Umbrella Hub is the way to go.

The developments and allocation of funding in NextGEOSS for this action have been done in different time frames than the newer version of EOPEN (which started in a later time and is still on-going). No funds have been dedicated for this action in NextGEOSS since June 2019 that the DHuS based architecture was finalized. Since then NextGEOSS team (WP2) has started to test the developed DHuS application at staging environment at the second quarter of 2020 before moving it to the production environment.