Sentinels Rolling Archive User Access, Operations, Maintenance and Evolutions

DHuS Administration Manual

Role/Title	Name	Signature	Date
Authors	DHuS Team		20/09/2017
Verified/Approved	Andrea Tesseri		20/09/2017

Change register

Version /Rev.	Date	Description
1.0	05-02-2015	First issue of the document
1.1	17-03-2016	The following deletion has been made to be in line with the Software distribution of versions after 0.9.0-2: • GWT GUI deletion • Statistic panel description deletion User manual remanded to the online versions
1.2	27-07-2016	The following information has been added in the document to be in line with the Software distribution of versions after 0.10.3-4: • configuration parameters description • removed example of software update from 0.4.3-1 version • synchronization best practice in case of proxy rules added in the synch without copy instructions. • synch with copy description and instructions • timeout parameter instruction • the data hub software functions description and image changed • add missing info on the synchronization instructions new dhus.xml updated
1.3	09/02/2017	Review and Update of the whole document
1.4	21/03/2017	Shapefile configuration description
1.5	20/09/2017	Update according to DHuS 0.13.4-14 Software Release Note

Table of Contents

Table of Figures

Table Index

1. Introduction

1.1 Scope

This document applies to the Data Hub Software ("DHuS") and is maintained within Sentinels Rolling Archive User Access, Operations, Maintenance and Evolutions" contract (n.: 4000116830/16/I-BG) by Serco.

1.2 Document applicability

Here below we provide a cross check between the SW version and the version of this document. In addition, the new features and performance improvements of each release can be found in the corresponding Software Release Note.

DHuS version	DHuS Administration Manual version	SRN reference
0.12.5-6	v. 1.2	Release Notes v.0.12.5-6-osf
0.12.5-11 0.12.5-12 0.12.5-13	v. 1.4	COPE-SERCO-TN-17-0079
0.13.2 to 0.13.4-14	V1.5 (this version, or "reference version")	COPE-SERCO-TN-17-0106

Table 1 DHuS version & Documents reference Mapping Matrix

1.3 Purpose

The purpose of this document is to describe the intended uses of the DHuS software and to provide a comprehensive step-by-step guide to the targeted user typologies.

In particular, it describes (and dedicates a specific section to):

- How to install the DHuS software;
- How to administer, manage and operate the installed DHuS instance;

The target audience of this document therefore are System administrators that will install the DHuS software and manage the DHuS. The User-oriented manual of the Data Hub is available here https://scihub.copernicus.eu/userguide/.

1.4 Reference documents

Reference	Document name, Reference, issue number, revision number
RD 1	GitHub open source framework https://github.com/SentinelDataHub/DataHubSystem
RD 2	Odata System Design Document, GAEL-P286-SDD-003-01-00, v1.2, 2016-04-25
RD 3	Open Search Description Technical Note, GAEL-P286-TCN-013, v1.0, 2017-06-12
RD 4	OWC System Design Document, SPA-COPE-DHUS-ADD-001, v1.0
RD 5	DHuS Architectural Design Document, GAEL-P286-ADD-001, v2.1 2014-06-03
RD 6	DHuS OData and Open Search Interface Control Document, COPE-SERCO-IF-17-0018, v2.0
RD 7	Cache management GAEL-P286-TCN-014, v1.0
RD 8	OWC Developer guide: http://sentineldatahub.github.io/DataHubSystem/owc.html
RD 9	OWC User Manual: http://sentineldatahub.github.io/DataHubSystem/OWC/user_manual.html

Table 2 Reference Documents

1.5 System overview and functionalities

The Data Hub Software (DHuS) is an open source software developed by a Serco/Gael consortium to the purpose of supporting the ESA Copernicus data access.

The DHuS provides a web interface (UI) to allow interactive data discovery and download, and a powerful Application Programming Interface (API) that allows users to access the data via computer programs/scripts thereby automating/integrating the download within their workflow.

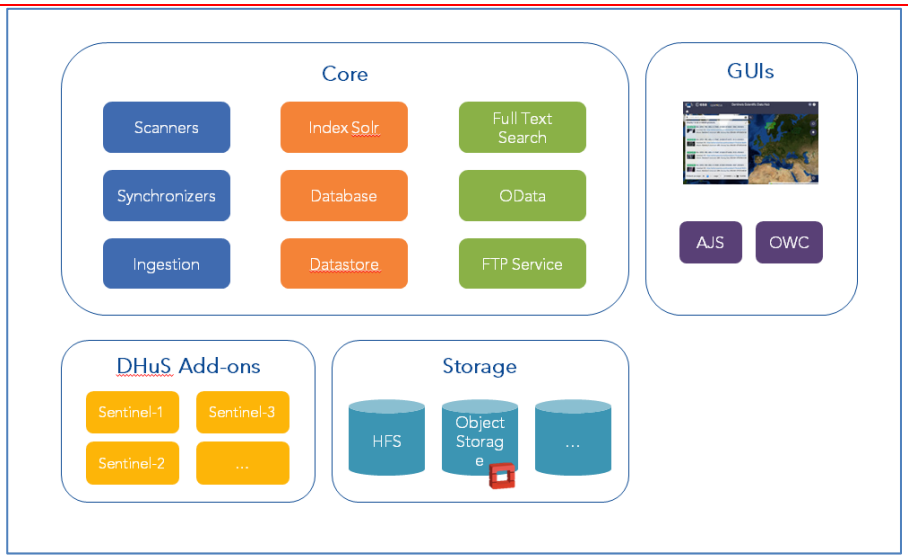


Figure 1 DHuS High Level Architecture

The major functionalities of the Data Hub Software are schematically represented in figure below.

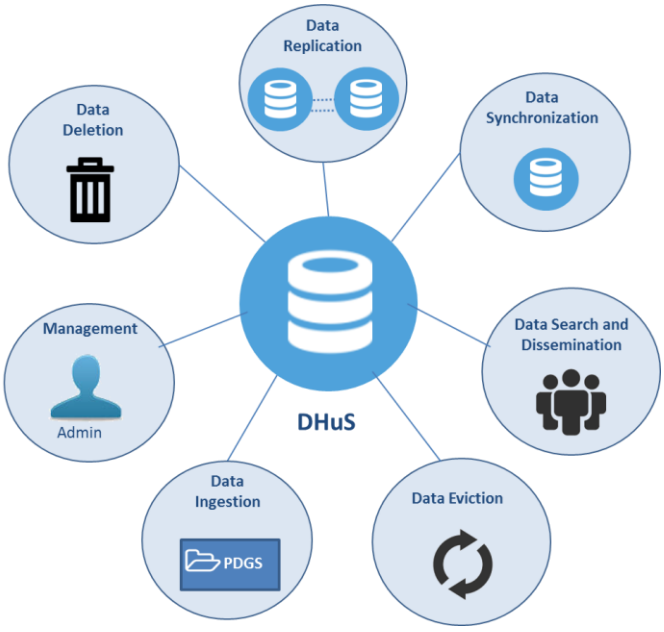


Table 3 DHuS functionalities

Data Ingestion (DHuS→external data provider, e.g. PDGS): this function is activated by DHuS, which retrieves Sentinels' products from an external data source.

Data Search (Users->DHuS): this function is activated by external users who perform searches via four possible interfaces:

- Web-based Graphical User Interface

- Open Web Component (see [RD 4])
- Open Data Protocol (OData) interface (see [RD 6]).
- OpenSearch interface (see [RD 3])

Data Dissemination (DHuS→Users): The dissemination function manages the product dissemination, retrieval and access by the user. It also allows disseminating/retrieving sub-components of products. This is considered of particular relevance for the voluminous data products managed in the Sentinels mission.

Data Synchronization (DHuS 1→ DHus 2): this interaction, based on the OData protocol, allows a DHuS Instance (DHuS front end) to retrieve products and their metadata from another DHuS instance (DHuS back end).

Data Replication (DHuS 1→ DHuS 2→….. →DHuS N): this interaction is activated by DHuS when two or more instances are deployed in scalable mode. The scalable Data Hub System is based on a set of multiples instances working together as a cluster of nodes, with a front-end proxy acting as the load balancer to distribute users' requests among the cluster. The replication functionality is the process allowing the data spread among the cluster (from a DHuS instance called "Master" to the other instances called "Replicas").

- one filesystem is needed for storing the DB where the products are indexed, along with the logs and application binaries. This filesystem needs to reside on the local disks.
- one or more filesystems are used for archiving the products. It can be the same filesystem as the one hosting the application or a dedicated storage.

Data deletion (Administators->DHuS): this functionality permits to the administrators to delete products from the archive.

Management (Administrators·DHuS and Users): this function is in charge of managing the user accounts for access to the DHuS. It enables the definition of the roles and permissions of the users and prevents uncontrolled accesses. Moreover, it controls the data eviction and the system configuration.

Data Eviction(Administators->DHuS): The DHuS rolling archive is managed through control of the overall disk space available for EO products, the management of a general eviction policy and the creation of specific eviction rules governing the period during which products and collections remain available for dissemination prior to eviction.

2. Installation pre-requisites

2.1 DHuS deployment modes

DHuS can be installed and configured to run in three different modes. The purpose of having different modes is to cope with increasing user load and amount of data to be managed. The modes, in order of performances (and requirements) are:

- **Single instance mode**
- **Front-end/Back-end Mode**
- **Scalability mode**

Single Instance Mode

The technical specifications of the DHuS in Single Instance Mode are provided in the following table:

	MINIMUM	MEDIUM	HIGH
CPU Core Number	4	24	32
RAM	8 GB	32 GB	48 GB
LOCAL DISK	> 1 Gb		

Table 4 Single Instance System Requirements

Front-end/Back-end Mode

The deploy in front-end/back-end mode foresees the set-up of two (or more) Data Hub instances connected via OData synchronizers (sections 4.1.5 and 4.2.4). Depending on the kind of OData synchronizer active among the Data Hub instances, this deployment can be set up in two different ways:

- the data harvesting functionality (ingestion) is managed by one or more DHuS instances, called "*back-ends*"
- the management of user requests and products publication to end users is managed by **one** DHuS instance called "*Front-end*"

Resources allocation for the FE and BEs instances depends on the user load and the ingestion rate. Values provided for single instance apply.

Scalability Mode

The objective of the configuration in scalability mode is to have several DHuS instances acting as one to share the user load and the products information: the deployment in scalable mode is completely transparent to the user.

The deployment of DHuS in scalable mode suitable foresees three main actors:

- **Master:** The DHuS master is the one and only product data source and is in charge of the ingestion/synchronization of products.
- **Replicas**: The DHuS replicas are master's doppelgangers. Products and user information stored in the DHuS master are replicated to all the replicas so that users can access product metadata. Replicas are accessed by the users (through a proxy). Consequently, the user information (e.g. profile changes) is spread from the replicas to the master. It is mandatory that master and replicas share the data store to allow access to ingested products. The product deletion and eviction shall be executed on the replicas. User registration shall be executed only on one of the replicas (to avoid database conflicts).
- **Proxy**: A proxy is needed for load balancing among the replicas. It must be configured to redirect incoming traffic to the DHuS replicas based on a load balancing algorithm with sticky sessions. Please refer to the proxy documentation for instructions on how to implement this.

Here a representation of the deployment in scalability mode:

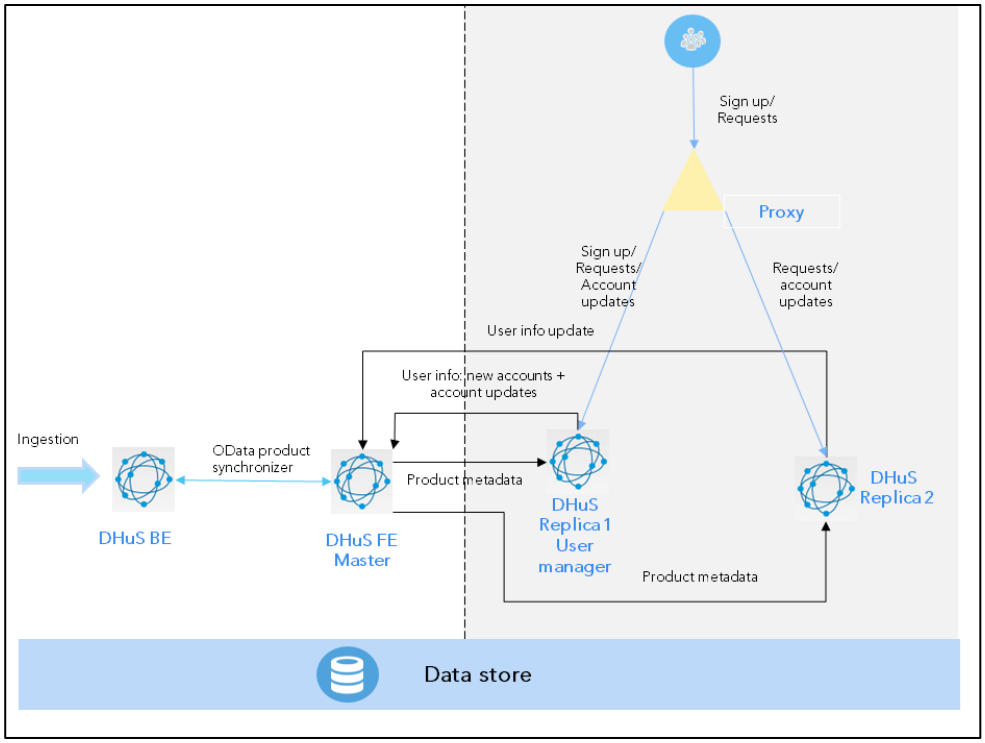


Figure 2 Deploy of DHuS in scalability mode

Resources allocation for the Master and Replicas instances depends on the user load and the ingestion/synchronization rate. Values provided for single instance apply.

2.2 System datastores

Generally speaking, three stores are available in DHuS:

- RelationalStore: HSQL database
- SolrStore: Solr Index

- DataStore: Where products are stored. All the three installation modes described in section 2 can be configured to use a block storage or an object storage. The supported Object storage is OpenStack Swift. Swift is a highly available, distributed, eventually consistent object/blob store (https://docs.openstack.org/developer/swift/).

Warning: multiple dataStores can coexist in DHuS; each dataStore is independent from the others. During the ingestion process, products are stored in all dataStores with write access. Please refer to the `[Group]/Parameter` description concerning "`[System]/archive and [System]/database`". Please refer to `[Datastore]/datastore` in [A.1] for details about how to configure Datastores.

2.3 Network Requirements

DHuS is accessed primarily via HTTP and FTP interface. The Installation procedure of the DHuS Software must be performed using a non-privileged user (not root); application installed in this way cannot start services listening on ports numbers smaller than 1024. By default the HTTP interface is reachable on 8081 port that must be opened for inbound requests. The DHuS FTP service is reachable, by default, on 2121.The DHuS requires for some of his functions a mailing service based on an external SMTP server. Following table describes the default DHuS network ports configuration:

SERVICES	INBOUND	OUTBOUND
HTTP	8081	External map server port External Nominatim service port
HTTPS	443	-
FTP	2121	-
SMTP	-	25

Table 5 Network Requirements

Please note that, since the UI Map is retrieved from an external service (and not internally from DHuS), the machine where DHuS is installed shall be able to access a map server (e.g., OpenStreetMap (https://www.openstreetmap.org). Moreover, DHuS provides also an interface with Nominatim (http://nominatim.openstreetmap.org/), consequently the machine where DHuS is installed shall be able to reach the service.

2.4 Software Requirements

DHuS software is fully written in java and can be considered portable to any platform providing the JRE (Java Runtime Environment). The DHuS supports:

- Java RE version 7 and 8
- Open RE version 7+

For what regards the operating system, the distribution is specifically tailored for Linux based systems and has been tested on the following distributions:

- Debian >= 7.7

- Red Hat >=6.7
- Centos >= 6.6
- Mac OS > =El Capitan (run but not tested)

It is recommended to use a Linux Operating System working on a multithread environment running in 64bit.

3. Installation and Configuration

3.1 Single Instance installation

- **Step 1**: Create a user named `dhus`. Every step in the installation procedure, if not explicitly mentioned, shall be performed as dhus user.
- **Step 2**: Create the installation directory: `mkdir -p [install-dir]`
- **Step 3**: Download the DHuS package (shar package) and save it into the installation directory
- **Step 4**: Change the permissions on the file: `chmod +x dhus-XX.XX.XX.shar`
- **Step 5**: Launch `./dhus-XX.XX.XX.shar` (the package will autoinstall). Once executed, the system setting configuration file can be accessed and updated.
- **Step 6**: Edit the `etc/dhus.xml` configuration file and modify the varFolder variable to an absolute path of your choice. This directory will contain the local archive, the incoming products, the database, etc. Eg: `<!ENTITY varFolder " /home/dhus/local_dhus">`
- **Step 7**: Start the DHuS entering the following command in the installation directory:

  ```
  nohup /bin/bash ./start.sh &
  ```

The log files will be created in the etc directory after the installation.

3.2 Scalability Mode installation

3.2.1 Installation and configuration procedure with an empty database

- **Step 1**: Download the installation package and install on all machines following
- **Step 2**: Master Configuration

  In `start.sh` of the master add the following parameters (some may already be present) to the java executable command line:

```
-Dhttp.proxyHost=[proxy external IP] \
-Dhttp.proxyPort=[proxy port] \
-Dhttp.nonProxyHosts="[proxy internal IP without last block]."\ (e.g., 192.168.1.) --
-Ddhus.scalability.active=true \
-Ddhus.scalability.local.protocol=http \
-Ddhus.scalability.local.ip=[DHuS master internal IP]
```

```
-Ddhus.scalability.local.port=[DHuS master port, as in server.xml] \
-Ddhus.scalability.local.path=/ \
```

In the dhus.xml set the "external" parameter with the proxy hostname (or IP) its port and the master's dhus path. Please check Appendix A for further details about the configuration parameters.

- **Step 3**: Replica Configuration

  Configure the start.sh of the replicas as follows:

```
-Dhttp.proxyHost=[external proxy IP] \
-Dhttp.proxyPort=[proxy port] \
-Dhttp.nonProxyHosts="[proxy internal IP without last block].*"\ (e.g., 192.168.1.*)
-Ddhus.scalability.active=true\
-Ddhus.scalability.local.protocol=http \
-Ddhus.scalability.local.ip=[DHuS replica internal IP] \
-Ddhus.scalability.local.port=[DHuS replica port] \
-Ddhus.scalability.local.path=/\
-Ddhus.scalability.replicaId=[id of the replica, integer] \
-Ddhus.scalability.dbsync.master=http://[DHuS master internal IP ]:[DHuS port]/ \
```

  In the `dhus.xml` set the "external" parameter with:

  - the proxy hostname (or IP),
  - its port,
  - the replica's dhus path.

  Please check Appendix A for further details about the configuration parameters.

- **Step 4**: Start DHuS master and wait until the start-up process is complete.
- **Step 5**: Start DHuS replicas.

3.2.2 Installation and configuration procedure with an already existing database

- **Step 1**: Follow Step 1 in previous procedure
- **Step 2**: Follow Step 2 in previous procedure, configuring the scalability option as follows.

```
-Ddhus.scalability.active=false \
```

- **Step 3**: On the master, copy the database and Solr index (in {varfolder}/database and {varfolder}/solr, respectively), overriding the existing directories

- **Step 4**: Remove this file : {varfolder}/solr/dhus/conf/managed-schema, if present.
- **Step 5**: Start the DHuS Master to perform a database migration (automatically performed).
- **Step 6**: At the end of the migration process (which can take a while, depending on the number of products and users in the database), stop the DHuS Master.
- **Step 7**: Backup the database and Solr index and copy them on each replica overriding the existing directories in {varfolder}.
- **Step 8**: Add the option to the start.sh of the master:

  ```
  -Dauto.reload=false
  ```

  to prevent the master from sending its database to the replicas.
- **Step 9**: Start DHuS master and wait until the start up process is complete.

Follow Steps 3, 4 and 5 of previous procedure; taking care of starting replicas one at a time.

Please note that at each new replica addition to the cluster it is necessary to shut down the master, restart it and start the replicas from the newest to the oldest.

3.3 Software configuration

DHuS configuration files are:

- **start.sh**; located in the installation directory
- **restart.sh**; located in the installation directory
- **stop.sh**; located in the installation directory
- **dhus.xml**; located in the etc directory
- **server.xml**; located in the etc directory
- **log4j2.xml**;located in the etc directory
- **suggester.dic**;located in the etc directory
- **opensearch-description-file.xml**;located in the etc directory
- **AJS GUI**
  - **appconfig.json**; located in[DHUSDIR]/var/tomcat/webapps/new/conf
  - **styles.json**; located in[DHUSDIR]/var/tomcat/webapps/ROOT/conf
- **OWC**
  - **appconfig.json**; located in[DHUSDIR]/ var/tomcat/webapps/new/conf
  - **rules/***.xml; located in the etc directory
  - **task_tables**/*.xml; located in the etc directory
  - **styles.json;** located in [DHUSDIR]/var/tomcat/webapps/new/conf

- **menu.json;** located in [DHUSDIR]/var/tomcat/webapps/new/conf
- **theme.json;** located in [DHUSDIR]/var/tomcat/webapps/new/conf

The following subsection describes in details how to configure the major DHuS functionalities; for the complete list of configuration parameters, please check Appendix A.

3.3.1 Openstack datastore configuration

In order to configure DHuS to use the object storage "openStack swift" as its own datastore, it is necessary to have an Openstack account with *identity credentials* (username, password) and *its project/tenant name, region and authentication URL.*

Here follows details about how to configure DHuS, please note that the following steps assume that DHuS installation package has been already download and installed:

- **Step 1**: connect to the machine where DHuS is installed and check that the application is not running (`ps -eaf | grep dhus`)
- **Step 2**: Edit the `etc/dhus.xml` configuration file and modify the workingDir variable to an absolute path of your choice. This directory will contain DHuS temporary files created during the ingestion process. This folder is automatically deleted when the DHuS is shutdown.
- **Step 3:** Create a container in Openstack (please check Appendix [A.1] for further details about how to create a container in OpenStack).
- **Step 4**: Edit the `etc/dhus.xml` configuration file and modify the `<dataStores></dataStores>` tag section adding a dataStore parameter in order to use the Openstack implementation. The configuration of dataStore parameter is the following:

```
<ds:dataStore xsi:type="ds:openStackDataStoreConf"
     name="datastore_name"
     readOnly="false">
<ds:provider>openstack-swift</ds:provider>
<ds:identity>tenantName:username</ds:identity>
<ds:credential>password</ds:credential>
<ds:url>https://keystone9915.openstack.ovh.net:35358/v2.0</ds:url>
<ds:container>container_name</ds:container>
<ds:region>RegionOne</ds:region>
```

  Please refer to `dhus.xml` section in Appendix A to a complete explanation of attributes stated in the previous example.

- **Step 5**: Start the DHuS entering the following command in the installation directory:

```
nohup /bin/bash ./start.sh &
```

3.3.2 Auxiliary Data Files configuration

Auxiliary Data File Search is a new feature provided by the DHuS and available in OWC Application. Please refer to the OWC User Manual [RD 9] for further details.

Pre-requisites

The following instances are needed to set up the Auxiliary Data Files retrieval chain:

- DHuS instance containing Sentinel-3 Products. Users will access to ADF Search feature from OWC GUI installed on this DHuS instance.
- DHuS instance containing Sentinel-3 Auxiliary Data Files. It is recommended to use the same distribution version of DHuS instance containing Sentinel-3 Products. This DHuS instance has to be accessible to DHuS instance containing Sentinel-3 Products via HTTP connection.

Procedure

- **Step 1:** Create an account with search and download rights (e.g. username: adfsearch, password adfsearch) on the DHuS instance containing Sentinel-3 Auxiliary Data Files.
- **Step 2:** Connect to the machine where DHuS instance containing Sentinel-3 Products is installed and check that the application is not running (`ps –eaf | grep dhus`)
- **Step 3:** Edit file `etc/conf/ExternalDHuS.properties` adding the following parameters needed to connect with DHuS instance containing ADF:
  - **external_dhus_host**: hostname or IP address of DHuS instance containing Sentinel-3 auxiliary files, ending with slash ('/') character (e.g. http://[DHuS_address]/)
  - **external_dhus_username**: username of the account on DHuS instance containing Sentinel-3 auxiliary files (e.g. adfsearch)
  - **external_dhus_password**: password of the account on DHuS instance containing Sentinel-3 Auxiliary files (e.g. adfsearch)

  An example of configuration is the following:

```
external_dhus_host=http://192.168.10.79:8091/

external_dhus_username= adfsearch

external_dhus_password= adfsearch
```

- **Step 4:** Start the DHuS as single instance entering the following command in the installation directory:

```
nohup /bin/bash ./start.sh &
```

3.3.3 OWC configuration

DHuS implements also a User Interface based on Polymer library: the Open Web Components (OWC) is a client side software following the Single Page Application approach, for space data dissemination.

The default configuration of OWC Application is oriented to DHuS operations, so it foresees that the DHuS UI and APIs are accessible from URL including "/dhus" substring (via proxy).

If the DHuS UI and APIs are accessible from different kind of URL, here follows details about how to visualize properly OWC UI after DHuS installation and start up:

- **Step 1**: Change directory to the following folder

```
<dhus_installation_folder>/var/tomcat/webapps/new/elements
```

- **Step 2**: Open the `elements.html` file in editing mode
- **Step 3:** Look for the string "window.location.href.split"
- **Step 4**: In this code block

```
var config = {
            baseUrl: window.location.href.split("dhus")[0] + "dhus",
            clientUrl: window.location.href.split("dhus")[0] + "dhus/new"
             //baseUrl: "../", // dev config
            //clientUrl: "./" // dev config
        };
```

comment the default baseUrl and clientUrl variables and remove comment from baseUrl and clientUrl for development configuration, as shown below:

```
      var config = {
            //baseUrl: window.location.href.split("dhus")[0] + "dhus",
            //clientUrl: window.location.href.split("dhus")[0] + "dhus/new"
            baseUrl: "../", // dev config
            clientUrl: "./" // dev config
        };
```

Where:

- `baseUrl` is the URL composed by the IP address of the host and the port at which the DHuS is exposed,
- `clientUrl` is the URL used to access the local resources.

Step 5: Reload the web page `http://<dhus_host>/new/` to play with OWC Application

Warning: The `elements.html` file is overwritten at each restart of the DHUS, so it is recommended to create a backup copy of the file

4. Administration Guide

4.1 Administration panels

The DHuS UIs provide the Administrator a series of panels to fulfil every service. Such panels are in the AJS UI and in the Open Web Component.

Warning: please note that not all the administrative features (and panels) are currently available in OWC. For the available panels here we report the relative screenshots

The "default" DHuS graphical interface is the **AJS UI**: it is in charge of providing the user with an interface for the discovery, and downloading of products and for the visualization of the relevant metadata. AJS is accessible by the default at: http://DHuS_address/dhus (please check Annex A.7 for details).

The Open Web Component is accessible by default at http://DHuS_address/dhus/new/. Exclusively in OWC, the configuration can be manged by operators directly via the UI clicking on the "gearwheel" icon:

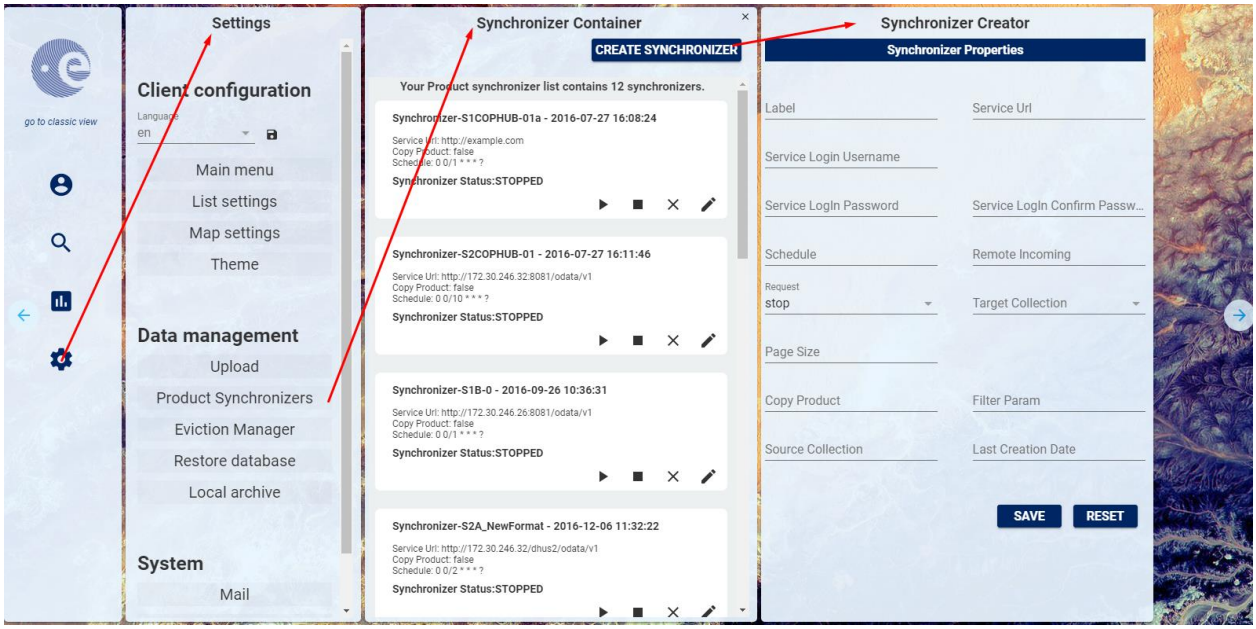


Figure 3 OWC configuration panel

Additional documentation about how to manage and configure the Open Web Component will be added to this manual in the next issues. For developers guidelines please refer to [RD 8].

4.1.1 Login as administrator

Once the installation package has been successfully installed, the DHuS server can be accessed online (https://dhus.xxx.zz) or on local URL (https://localhost/). To access the administrator panels, it is first necessary to login as root, using the default settings. The "Login" button is displayed in the upper right side of the DHuS Home page.

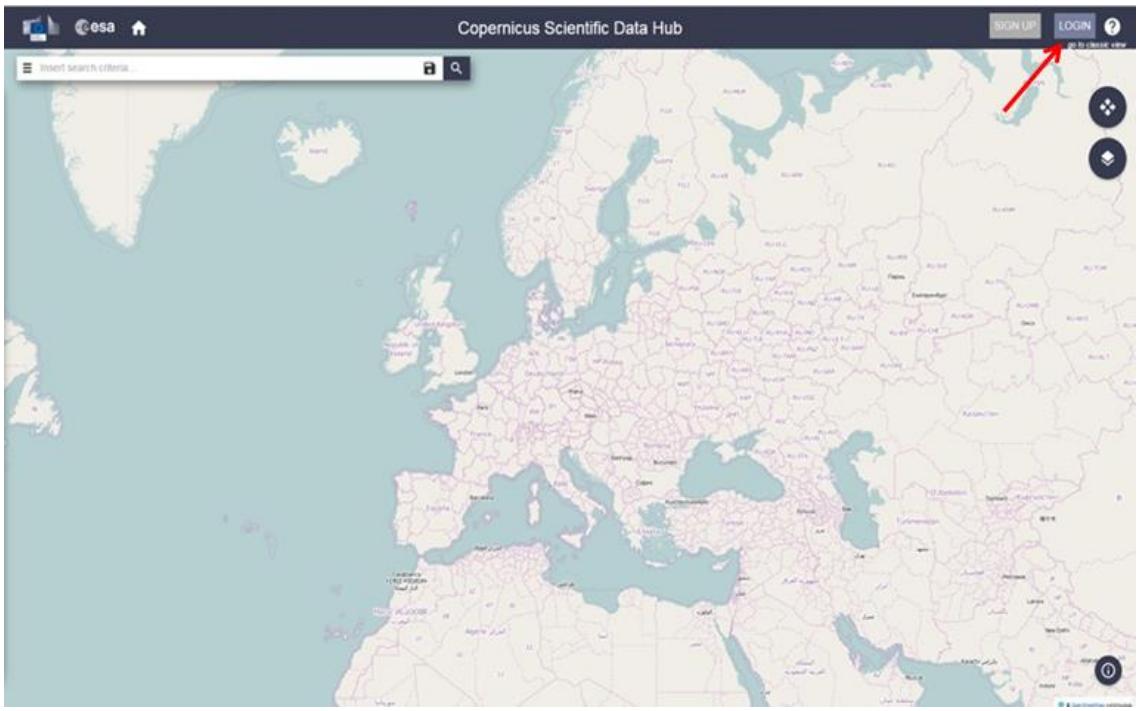


Figure 4 DHuS Login (AJS)

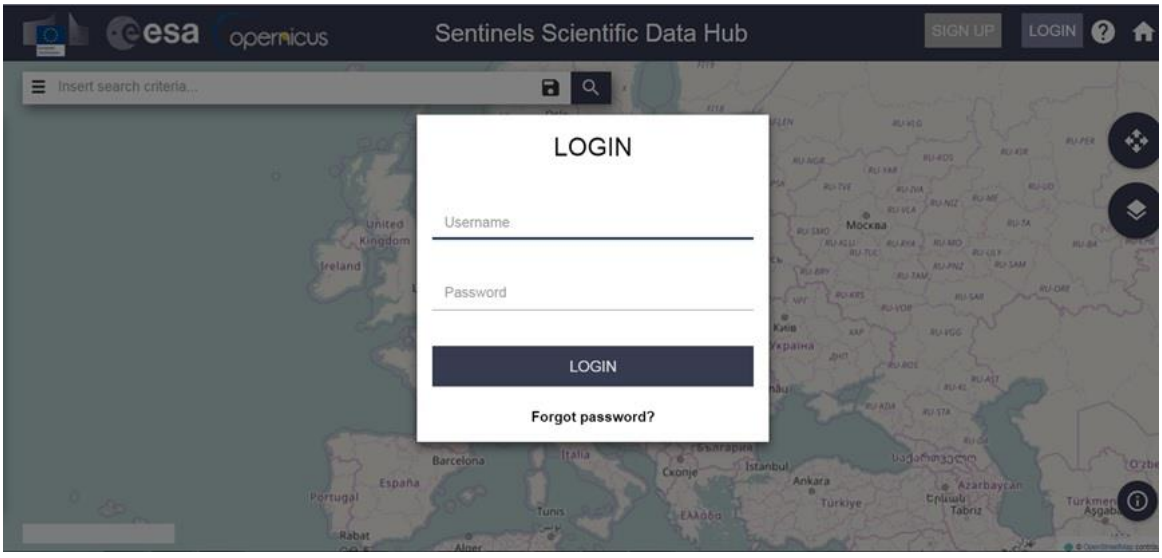


Figure 5 DHuS Login Panel -user view (AJS)

Once the administrator has logged in, the panels are accessible clicking on the "user icon" on the upper right side of the page.

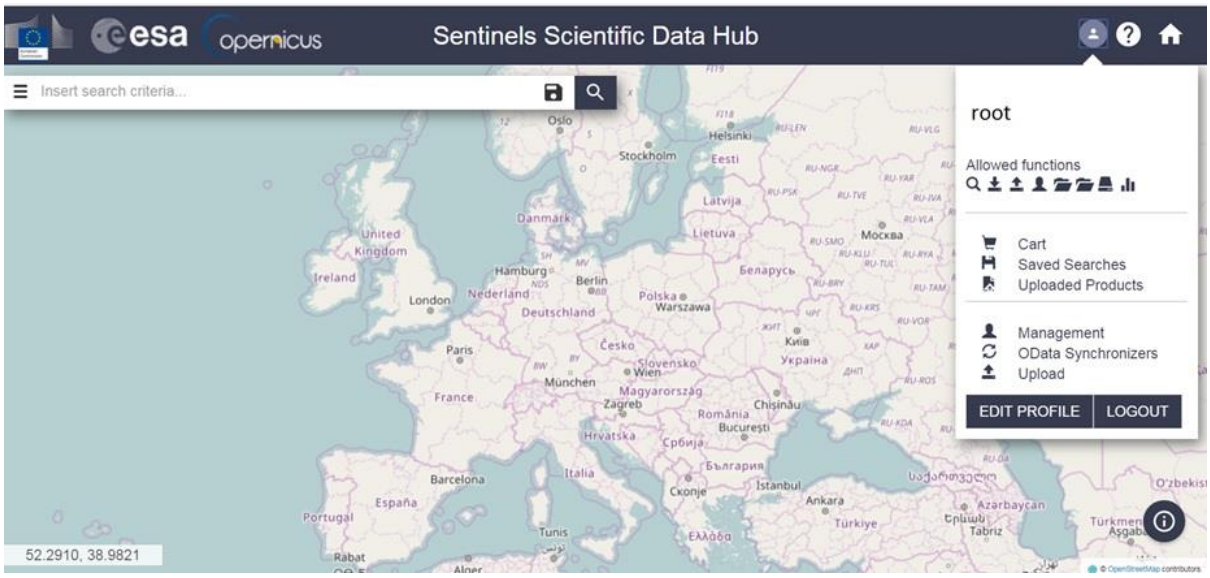


Figure 6 AJS Administration Panels

In OWC, the login button is displayed in the left side of the Home web page.

Figure 7 OWC login panel

4.1.2 Product deletion (from search panel)

Product deletion can be triggered via UI or directly via OData API. This section explains how to trigger the deletion process from the UI, section 4.2.6 will deal with the OData part.

Here follows how to delete one or more products (bulk deletion):

Step 1: Select the product/s to be deleted from the UI list/search panel. It is possible to select a set of products also using the "select all" button;

Step 2: Delete products clicking on the trashbin icon in the lower side of the panel itself

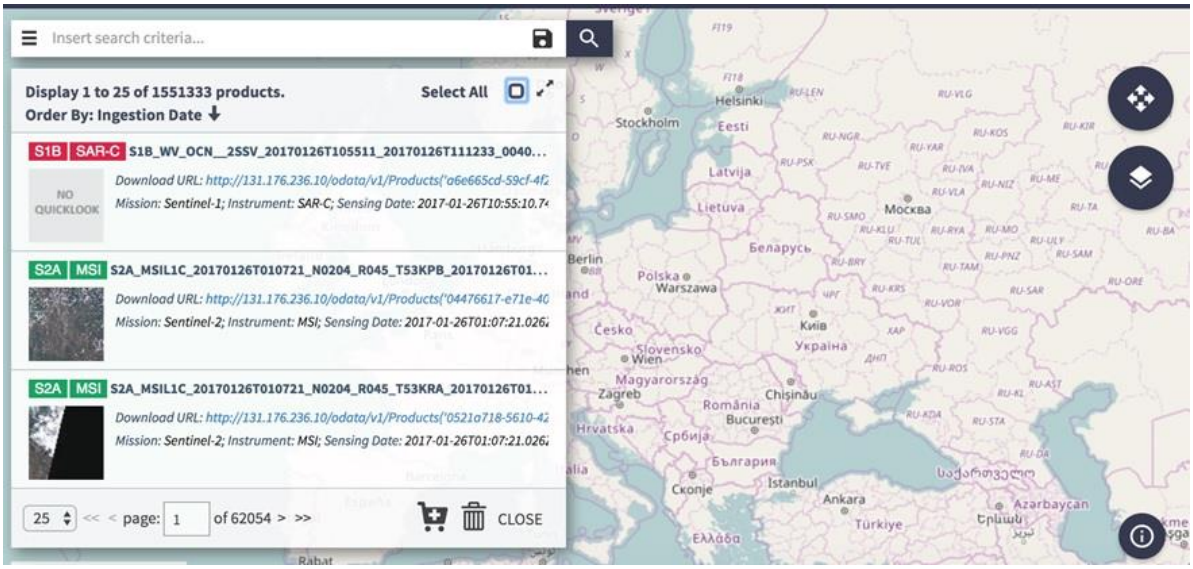


Figure 8 Deletion via OData API

Please note that in a the "Front-end/Back-end" deploy, once activated, the deletion acts on two or more of the DHuS datastores according to the instance on which it is triggered:

- If the deletion is launched on a DHuS FE instance connected to a BE via OData metadata synchronizer, the process will only erase data from the Relational datastore and the SolR datastore of the FE. The BE will not be affected by any change, in fact the product will not be deleted from any of the BE datastores.
- If the deletion is triggered on a BE instance, the process will erase data from all the three datastores. Consequently, if the BE is connected to a FE with an OData metadata synchronizer, to avoid inconsistencies in the FE, it will be necessary to delete the same data also from this instance (otherwise product downloads and product inspection will not work).

A prompt message is showed up to allow inserting a deletion cause:

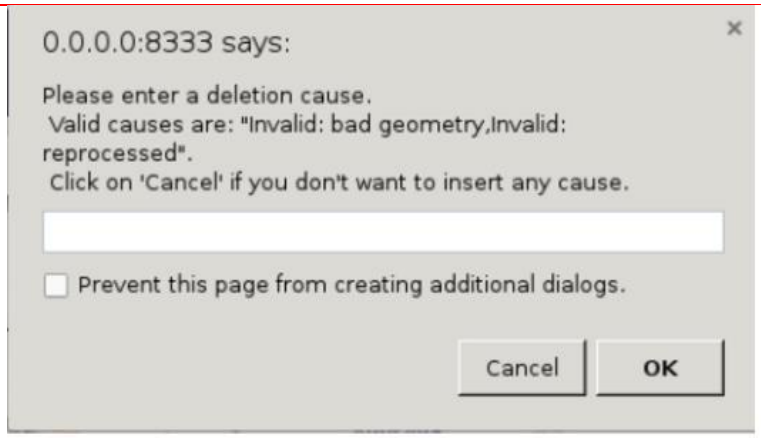


Figure 9 Deletion cause panel

The set of deletion causes shown in the panel can be configured as explained in section Appendix A (AJS GUI `appconfig.json`)

4.1.3 Products Ingestion

The DHuS allows the ingestion of Sentinels products using these methods:

- Ad hoc upload;
- Creating a file scanner;
- Synchronizing remote archive.

Ad hoc upload	File scanner	Synchronizing remote archive
⊕ HTTPS product transfer ⊕ Allowed collection targeting ⊖ No cumulative uploads ⊖ Only zip format compatible	⊕ Cumulative uploads ⊕ Parallel uploads ⊕ Optional FTP transfer protocol ⊕ SAFE & zip format compatible ⊕ Allowed collection targeting	⊕ Cumulative FAST upload ⊕ Parallel upload ⊕ HTTP/HTTPS transfer protocol ⊕ Product transfer not necessary ⊕ Allowed collection targeting ⊖ Other independent DHuS instance needed

Table 6 Ingestion methods

4.1.3.1 Product upload

The Upload feature is available only to the administrator. DHuS system makes available an incoming space to let the user upload a product. Once uploaded, data is processed to be referenced by DHuS clients. This panel gathers all the information necessary to perform the upload (at least the path to the product).

Optional: Assignation of a product to a collection is manually set by the uploader. A product can be included in any collection.

The product upload is accessible by the "upload panel":

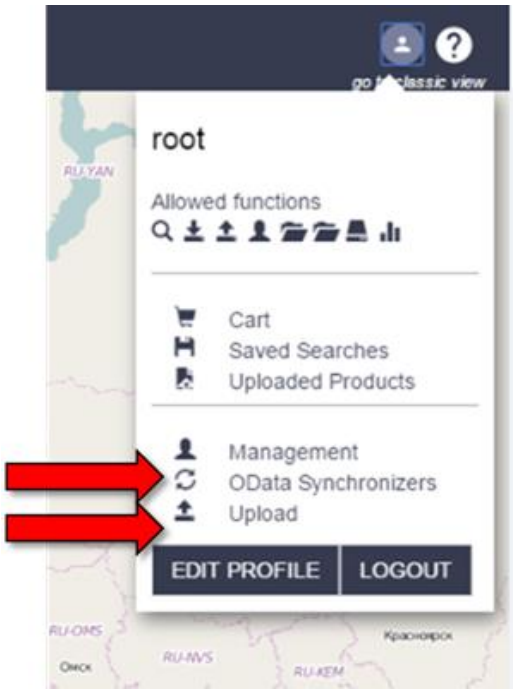


Figure 10 Products Upload (AJS)

Once in the "Upload panel", it is possible to perform the upload of a product: select the input products, then (optionally) select a collection in the list of collections and click on the "Upload" button. The upload will start and at the end of it, a pop up will notify that the upload is over.

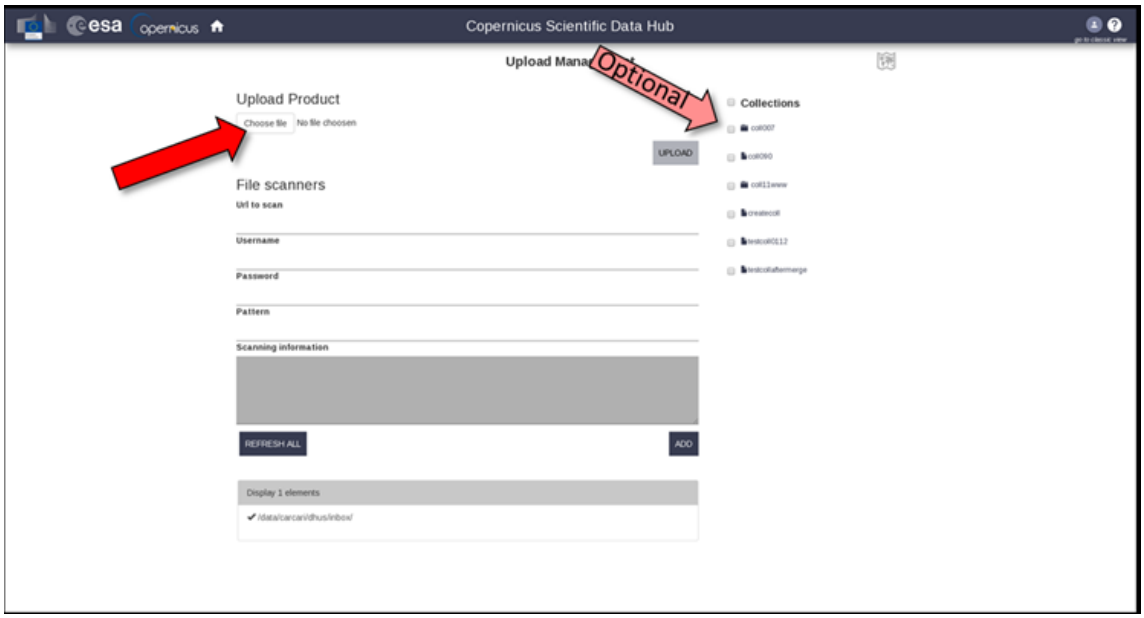


Figure 11 Upload Panel (AJS)

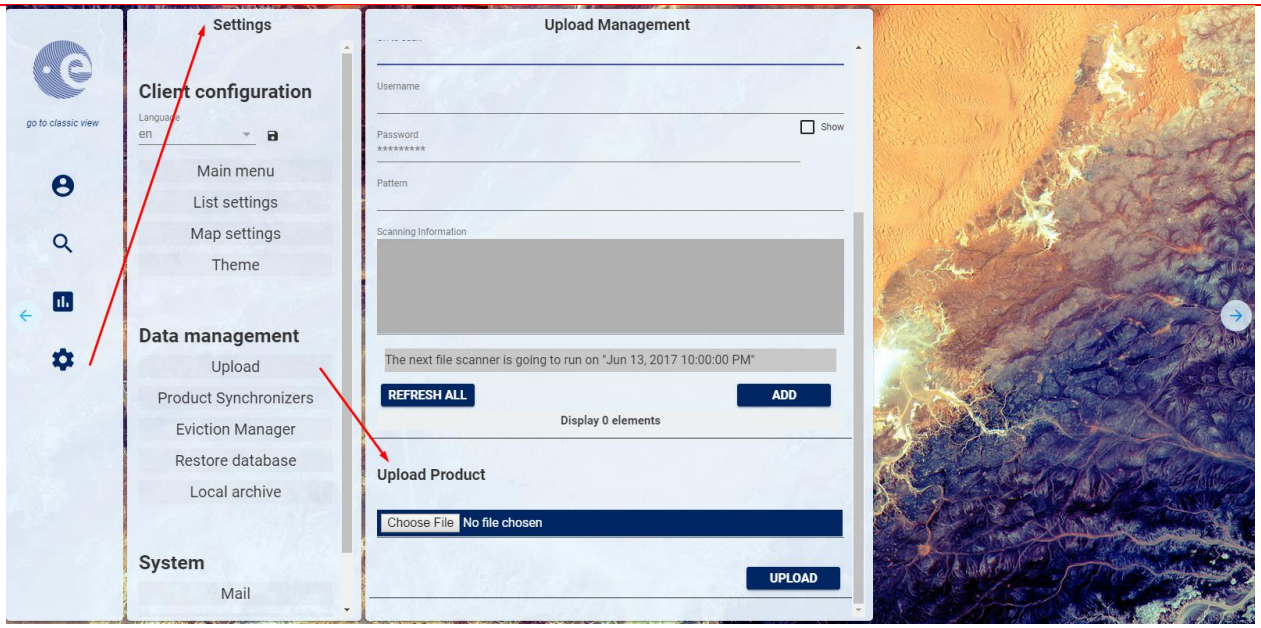


Figure 12 Upload Panel (OWC)

4.1.3.2 Upload via file scanner

If the upload has to be periodic, a scanner can be configured with the panel highlighted by the red arrow in figure below (still accessible from the upload panel).

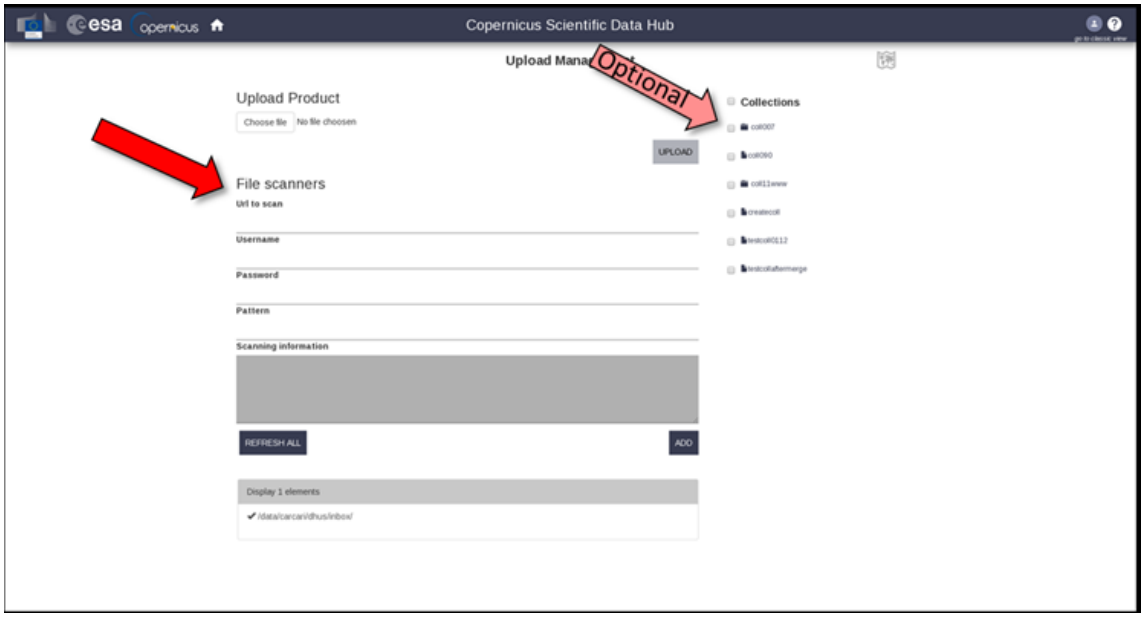


Figure 13 File Scanner Panel (GUI)

To create a file scanner:

- **STEP 1**: Access the upload panel;

- **STEP 2**: Fill the URL to scan field with the path of the folder containing the products (if the products are in the same machine where DHuS is installed, the field shall be filled as '`path/of/the/folder`').
- **STEP 3**: If the products are located on an external data provider (accessible via ftp), configure the username and password to access the machine; otherwise the username and passwords will not be necessary.
- **STEP 4**: To upload just specific types of product, configure the 'Pattern' field according to the regular expression roles explained in http://docs.oracle.com/javase/7/docs/api/java/util/regex/Pattern.html

  (e.g. "S1[AB]_\p{Upper}{2}_(SLC|GRDM).*" to upload only the SLC and the GRDM products)
- **STEP 5**: Click on the 'add' button. In the lower part of the page it will be written 'when' the file scanner will be activated again.

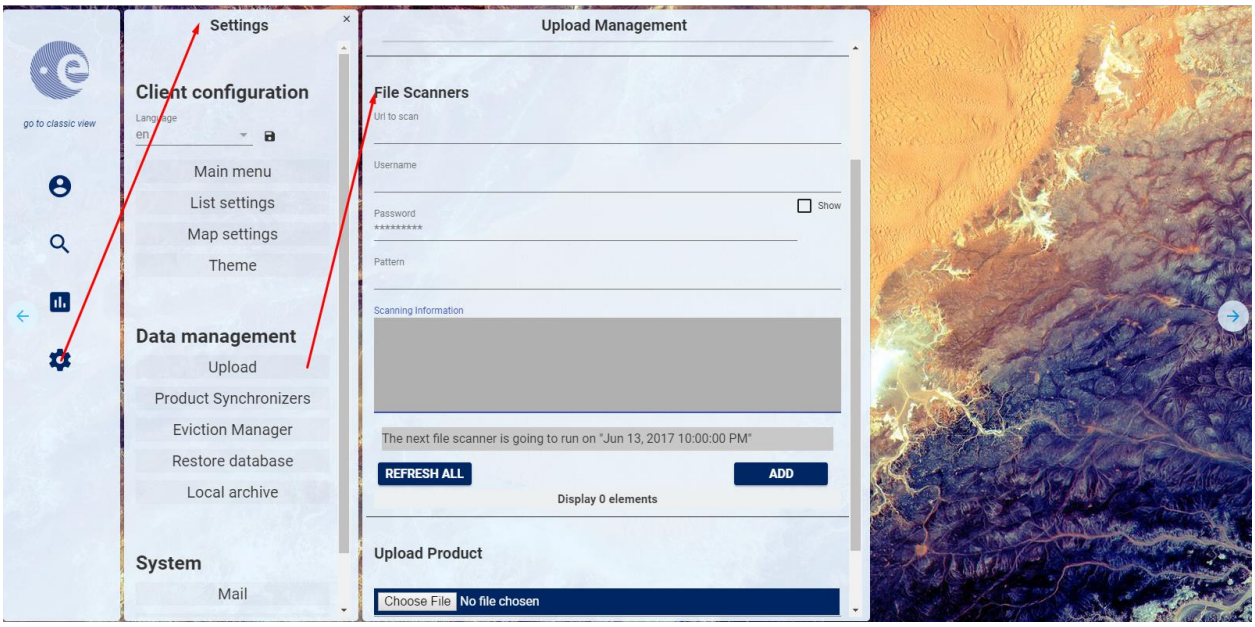


Figure 14 File Scanner Panel (OWC)

4.1.4 Management

The section and its sub-sections describe the management panels implemented in DHuS and accessible via AJS UI and OWC UI.

For the AJS UI, the first step is to access the "Management" area by means of the button at the right side of the Home web page.

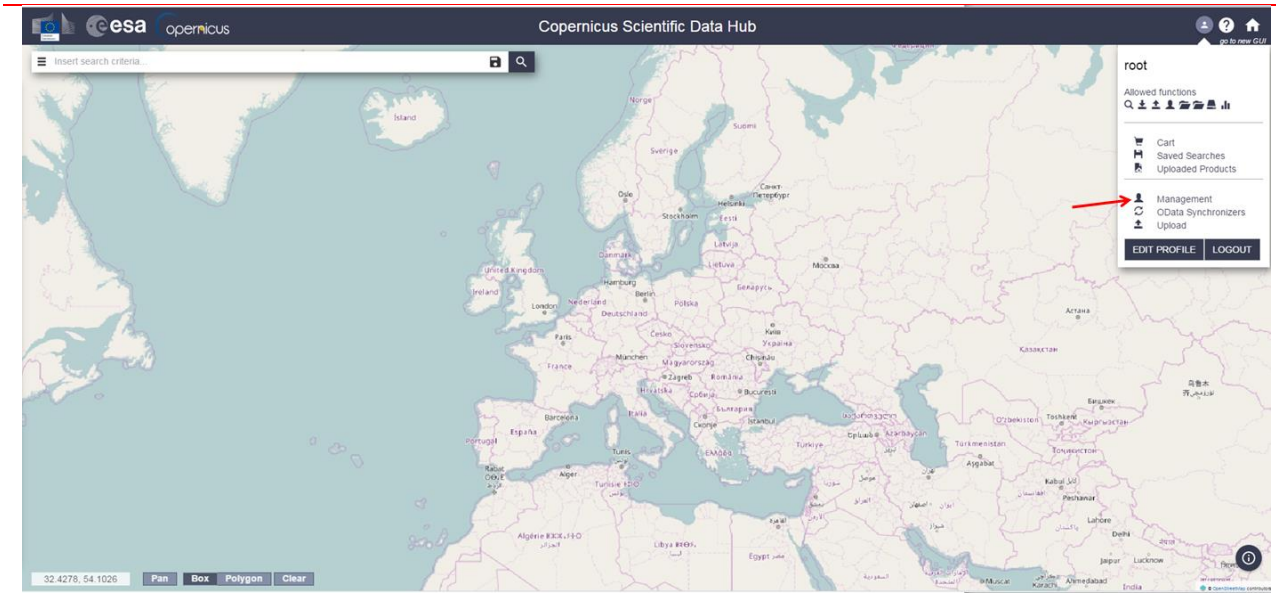


Figure 15 Management area (AJS)

The following four subpanels are present:

- Users
- Collections (not available in OWC)
- System
- Eviction

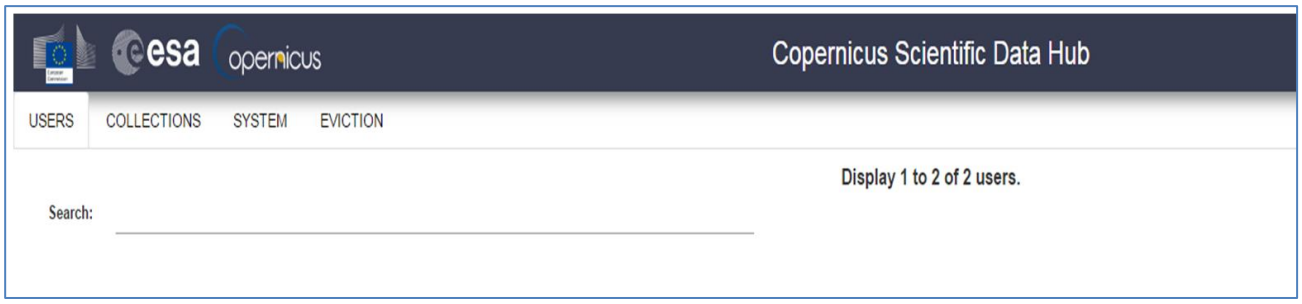


Figure 16 Management Subpanel (AJS)

In OWC, the subpanels are located in the "Settings" Tab:

- Users
- System
- Eviction
- Main Menu settings

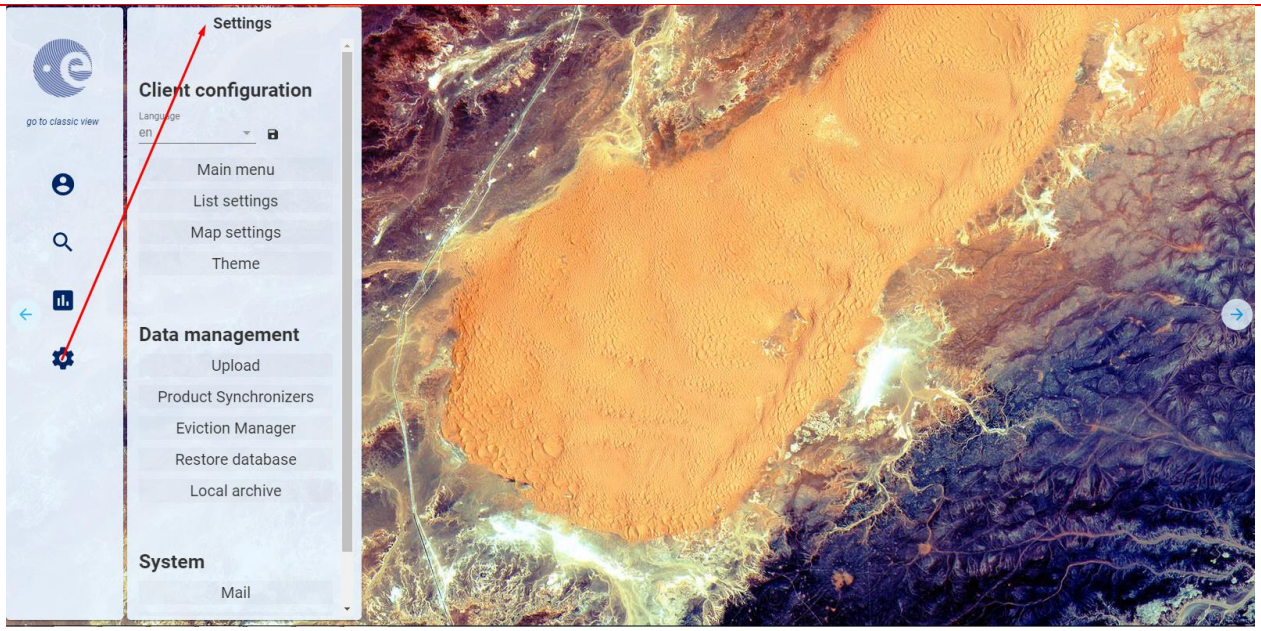


Figure 17 Management Subpanel (OWC)

4.1.4.1 User Management

The administrator management panel allows managing users. This means that the administrator can create, edit and delete any user.

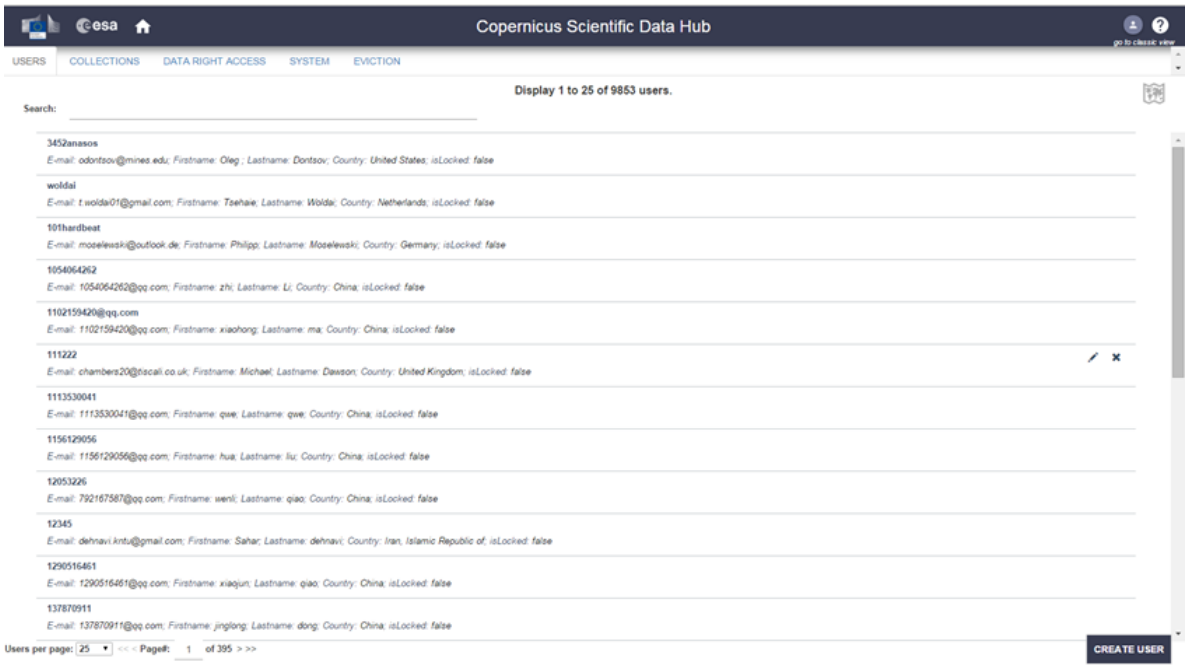


Figure 18 DHuS User Management Panel (AJS)

DHuS implements a user management system that prevents uncontrolled accesses and manipulations from unauthorized users. DHuS proposes a user authentication and authorization strategy defined in its internal Database. Users are able to register or sign-in and the administrator are able to configure the user/group permissions from the Web user interface. The user management activities are:

- **Step 1**: to create or delete a user;
- **Step 2**: to authorize the user to access a list of services;
- **Step 3**: to update a user profile

How to register a new user?

The Administrator shall:

- **Step 1**: Access the DHuS page;
- **Step 2**: Perform the login;
- **Step 3**: Select the Management Panel and then select the Users management panel;
- **Step 4**: Click on the "Create" button in the lower part of the User management page, which will enable the form here below.

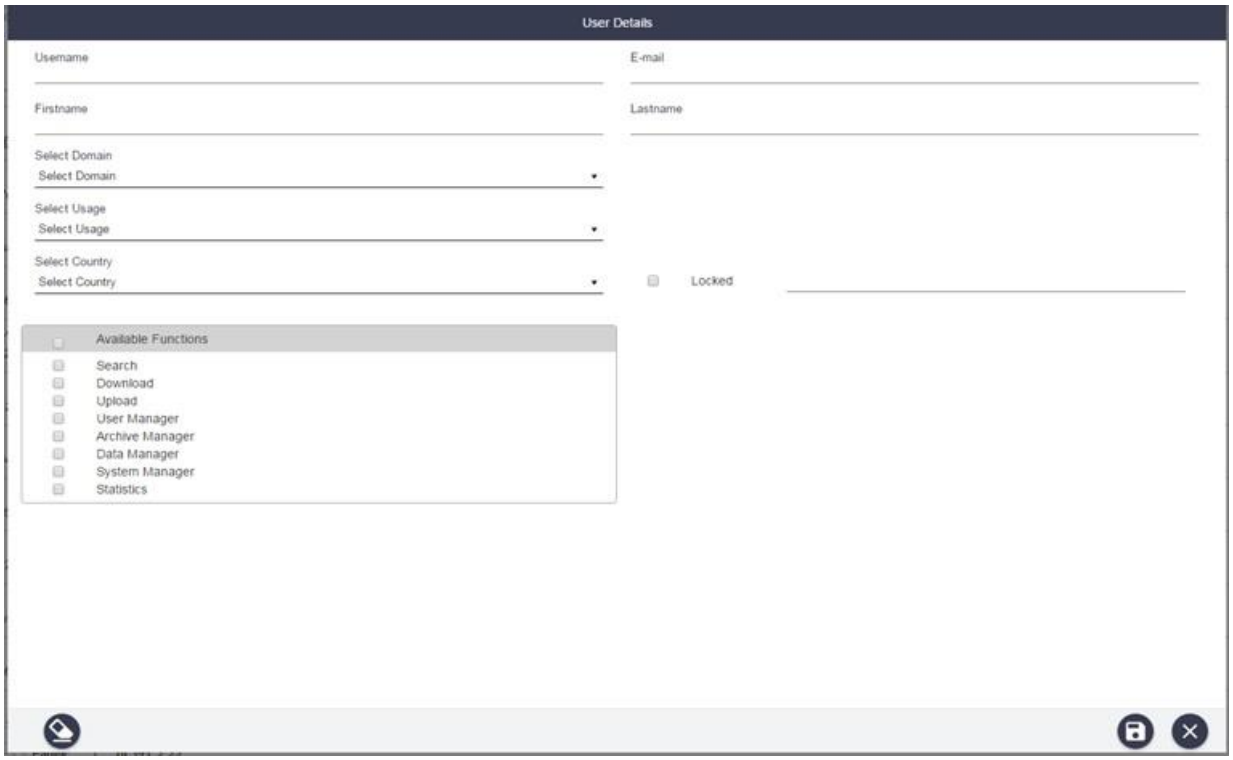


Figure 19 User Creation Form (AJS)

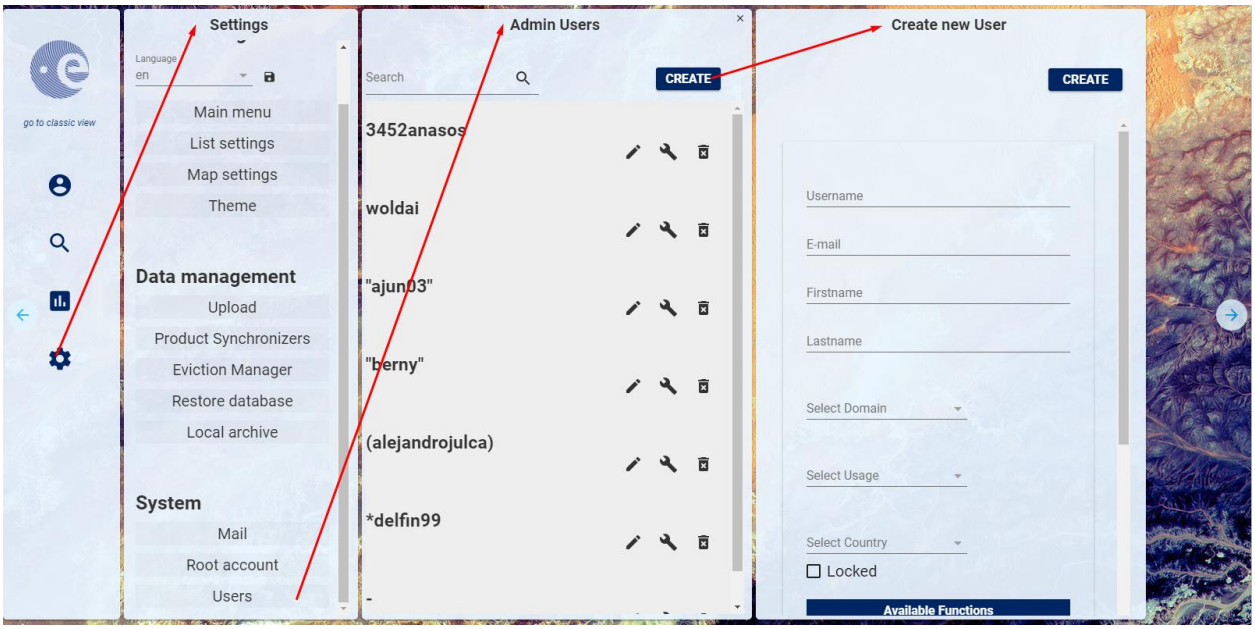


Figure 20 User Creation form (OWC)

- **Step 1**: Fill in the new user creation form (note that the fields marked with an asterisk are mandatory) and click on the functions that the user shall be able to use
- **Step 2**: Click on the "save" button to complete;
- **Step 3**: Then the email notification service will send an e-mail to the user with his profile information (login, first name, last name, available services...) including a generated password.

  The administrator has the possibility to modify user's authorization settings and information. To modify whatever authorization setting or user information, the Administrator, before executing the following "how to" procedures, has to:

- **Step 1**: Access the DHuS page
- **Step 2**: Perform the login
- **Step 3**: Select the Management Panel
- **Step 4**: Select the Users Management Panel
- **Step 5**: Select the name of the user in the users list on the left side of the user management panel

How to lock the selected user?

Click on the "locked" checkbox under the Registration form in the right side of the panel. The administrator shall also indicate the reason of this locking process in the box on the right

- **Step 1**: Click on the "save" button to complete;

- **Step 2**: Then the email notification service will send an e-mail to the user with his profile information (login, first name, last name, available services...) including locking notification and its relative reason, if it has been indicated.

Warning: the locking restriction and its relative reason are registered in the OData "Restriction" EntityType, under the corresponding "User". Please refer to [RD 6 RD 6] for more details.

How to delete the selected user?

Click on the "Delete" button to delete

Figure 21 Update and delete users AJS

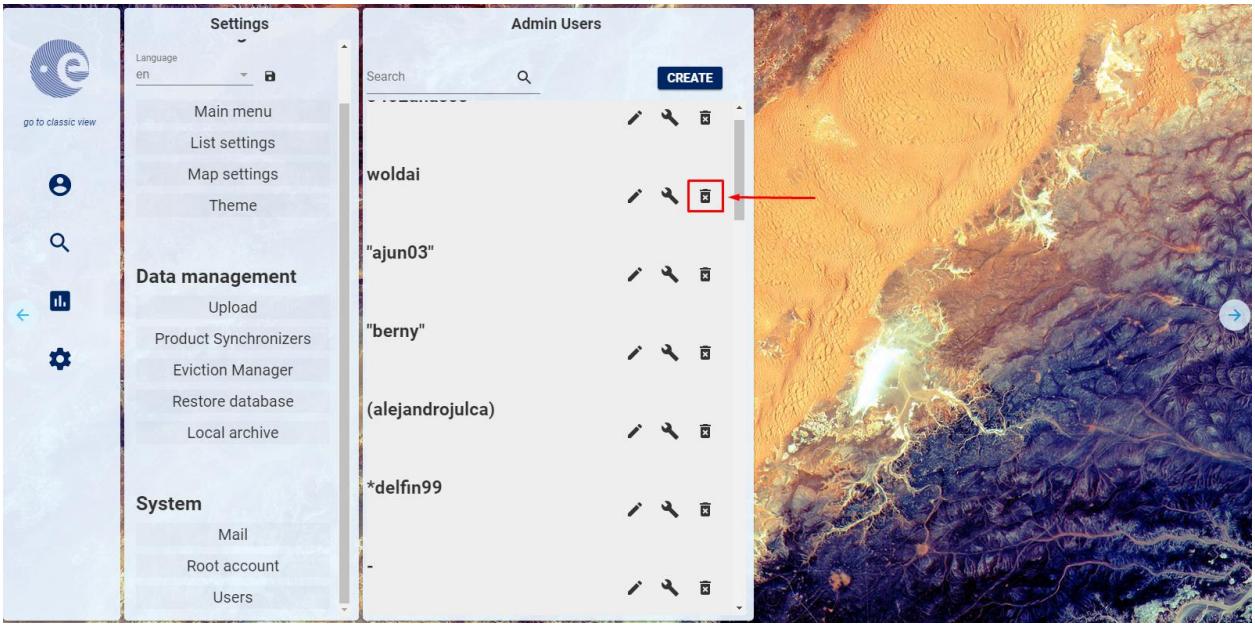


Figure 22 Delete users (OWC)

The email notification service will send an e-mail to the deleted user with the communication of the deletion process.

How to change user password?

Click on the "edit user password" checkbox and the following panel will appear:

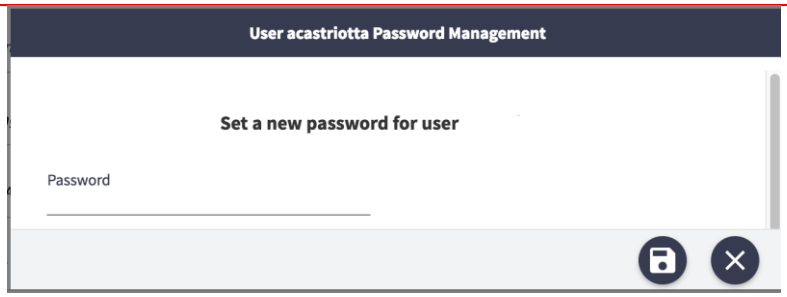


Figure 23 Update, edit user password and delete users AJS

Including the new password and, then, clicking on the save button, the user password will be updated.

4.1.4.2 Collection Management Panel

Products can be gathered into collections. Collections management consists of creating or deleting collections.

The Collection management panel also lists a set of products to be attached to the collection. The selection of collections is possible by browsing the collection hierarchy on the left. To access the collection management panel, the Administrator has to click on the "collections" link, sited in the upper left side of the management panel. The collection management panel here below will open. It contains the list of collections on the left and the list of archived products on the right.

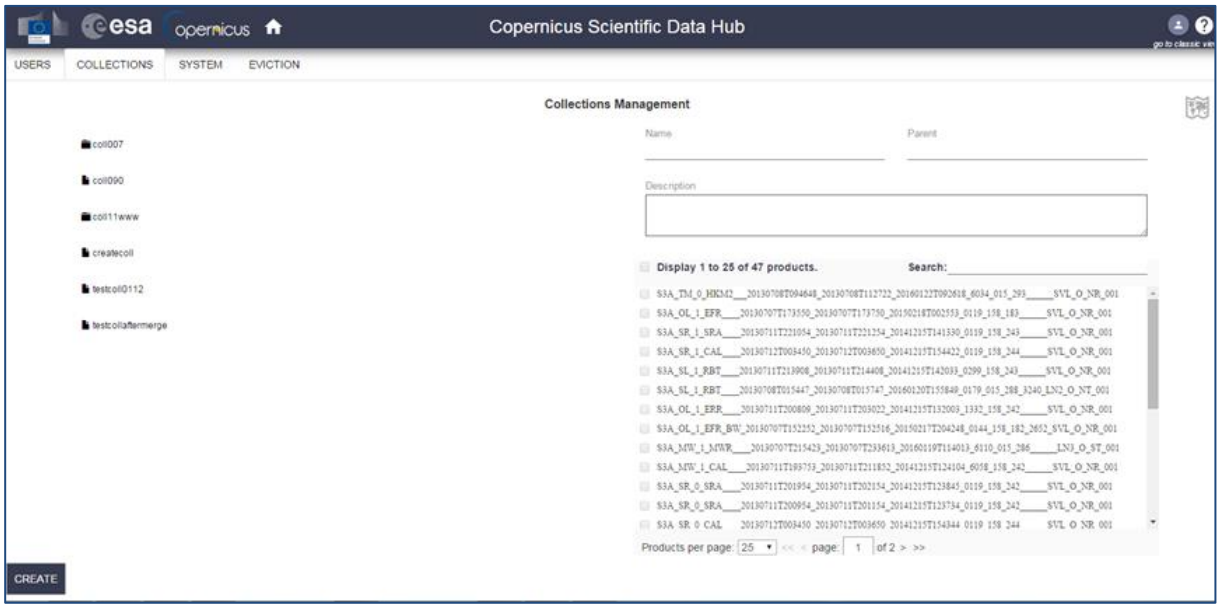


Figure 24 Collection Management Panel (AJS)

The administrator can manage the collection: he can create new collection and delete an existing collection. In the following subsections some "How to" tutorials are presented; the steps described in any of these tutorials can only be performed after the following preliminary actions:

- **Step 1**: Access the DHuS page;
- **Step 2**: Perform the login;

- **Step 3**: Go to the collection management panel.

How to create a new collection?

- **Step 1**: Click on the "create" button in the collection management panel;
- **Step 2**: Create collection;
- **Step 3**: Insert the collection information in the upper right side of the panel (the name of collection is mandatory), (optional) select (by clicking on the associated check box) the products to be added to the collection;
- **Step 4**: Click on the "save" button to register the new collection or click on the "cancel" button to abort the 'creation of collection' procedure.

How to delete a collection?

- **Step 1**: Click on the collection to delete;
- **Step 3**: Click on the "delete" button.

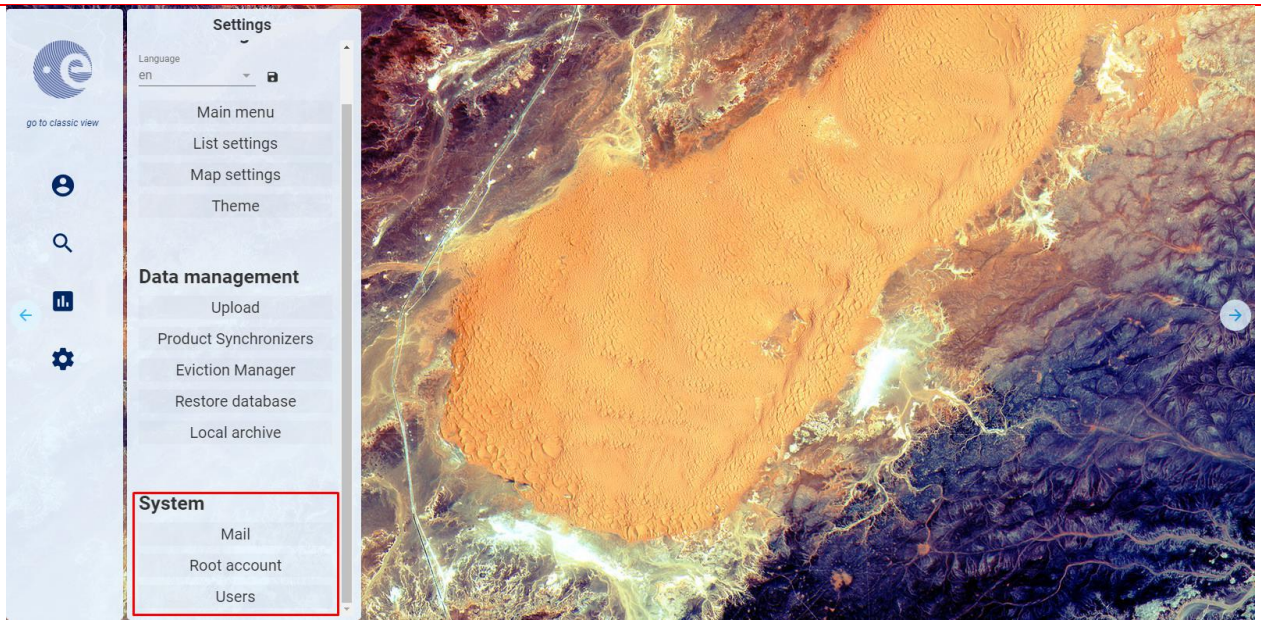
Note that the collection management page includes a searching box. It is useful to know if a product is collected somewhere.

4.1.4.3 System Management Panel

The System management is used to configure basic information in the system.

Figure 25 System management sections (AJS)

Figure 26 System management sections (OWC)



The Administrator from here can access the following sections:

1. **Mail configuration:** In this form it is possible to configure the SMTP server address, the username, password and e-mail account details to send communications to the users.

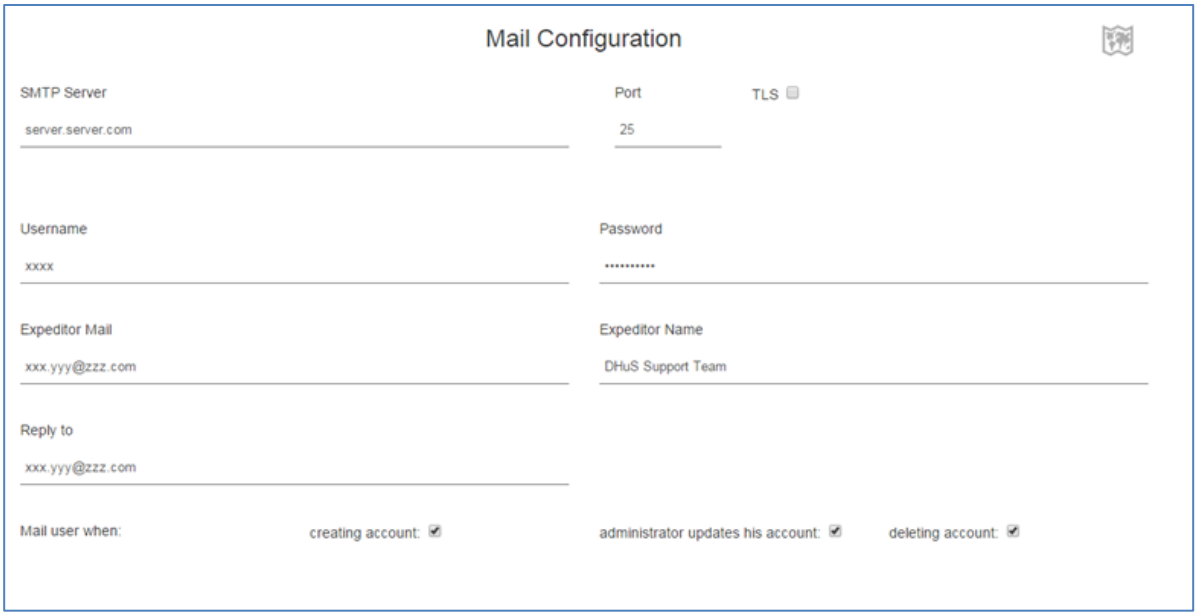


Figure 27 Mail configuration management Panel (AJS)

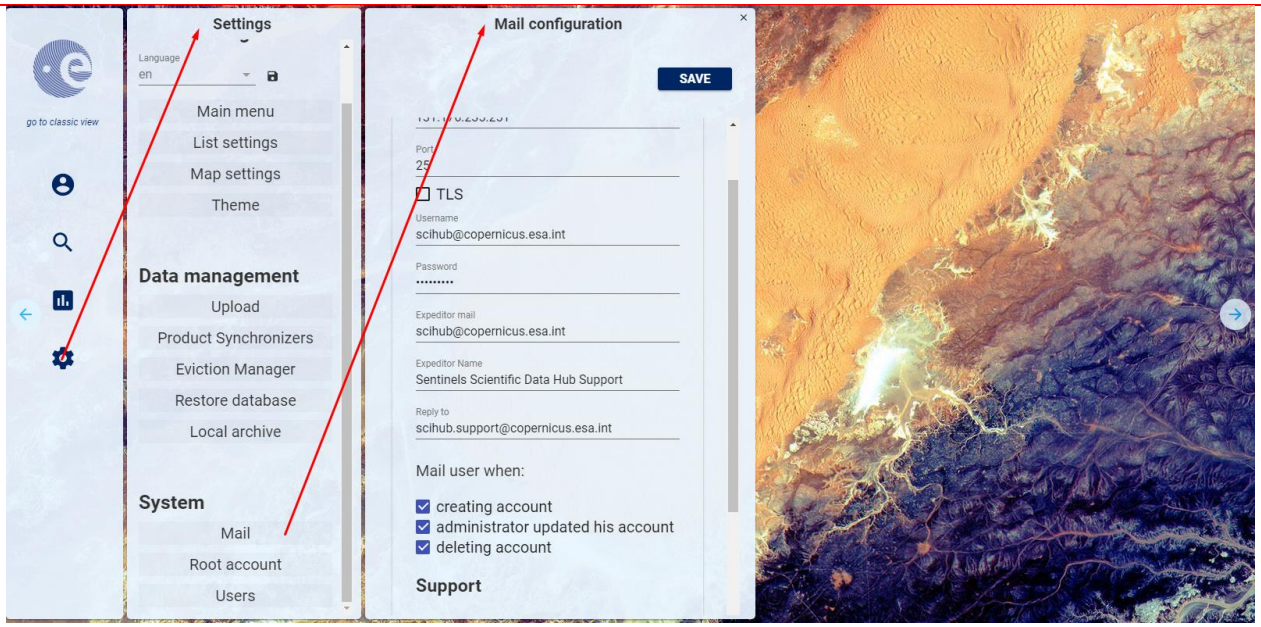


Figure 28 Mail configuration management Panel (OWC)

2. **Support information**: For any support information it is possible to contact the DHuS Support Team sending an e-mail to eosupport@copernicus.esa.int . This is not present in OWC.

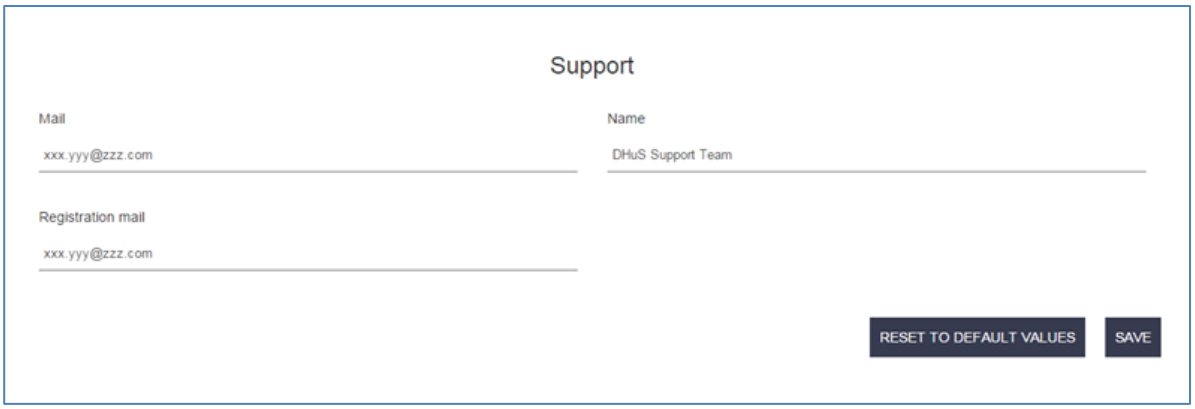


Figure 29 Support configuration management panel (AJS)

3. **Root configuration**: from this panel it is possible to change the administrator password. To do so, insert the old password, the new one and then confirm the new password.

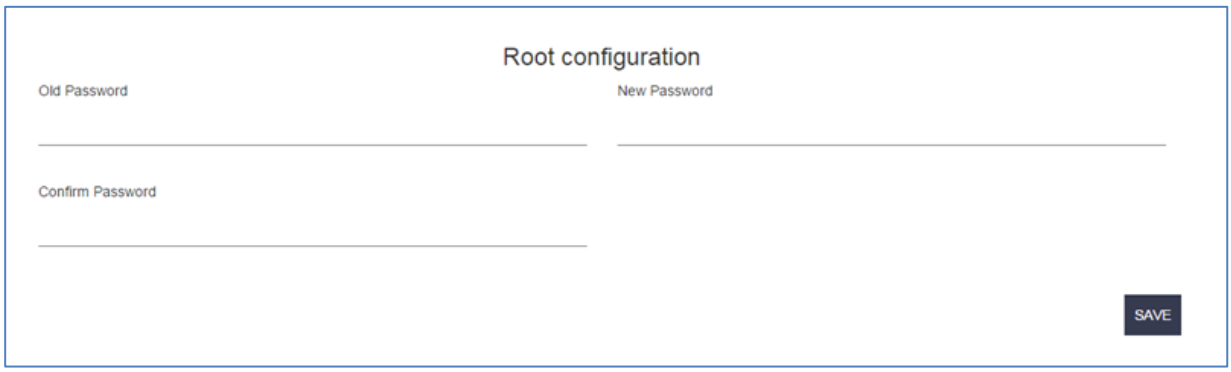


Figure 30 Root configuration management panel (AJS)

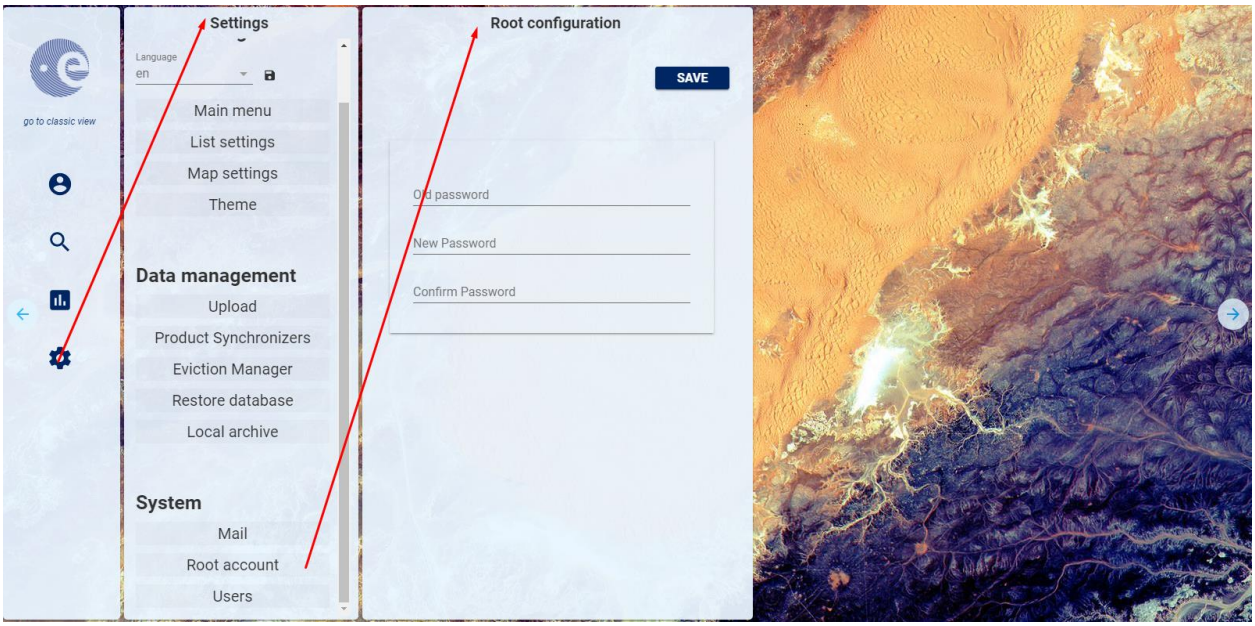


Figure 31 Root configuration management panel (OWC)

4. **Restore database**: in the `dhus.xml` file it is possible to configure DHuS so that it performs a periodical dump of the database. From this panel it is possible to restore the database dump.

Figure 32 Restore Database panel (AJS)

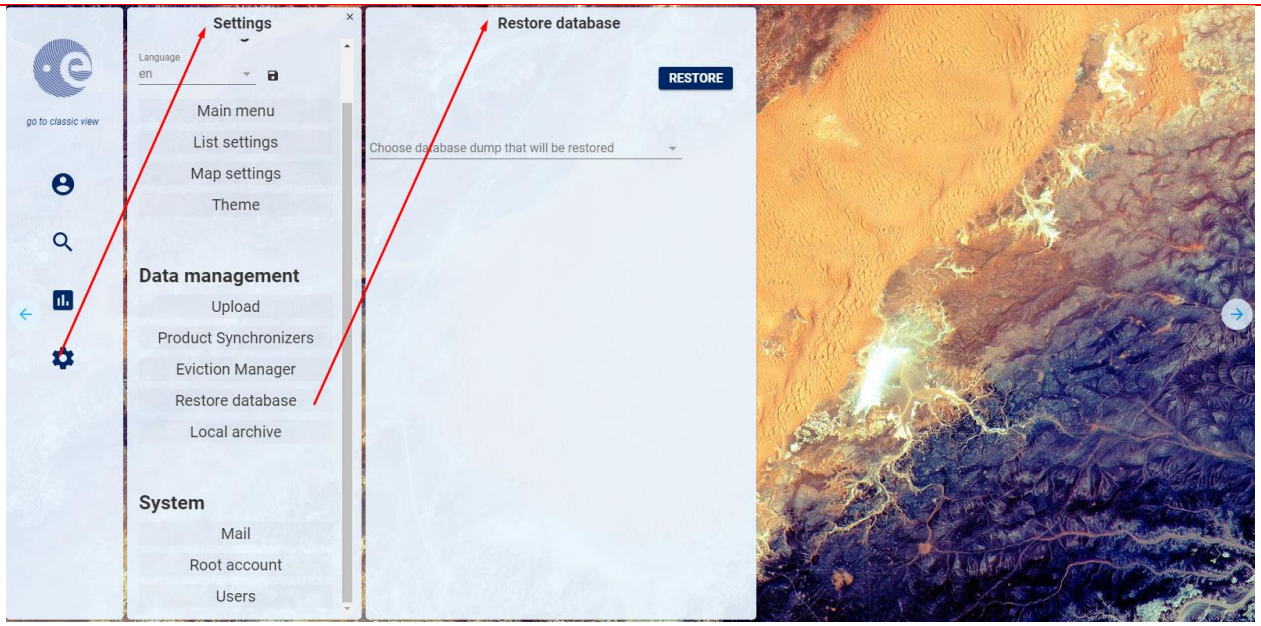


Figure 33 Restore Database panel (OWC)

To do so, perform the following steps:

- **Step 1**: Click on the drop-down menu in the "restore-database" section: the list of available dumps will be displayed through a list of dates (date during which the dumps have been performed).
- **Step 2**: Select the desired date and then click on "restore". DHuS will automatically stop and restart. Once DHuS will be up again, it will contain just the data inserted before the selected dump date.

4.1.4.4 Eviction Management Panel

The Data Eviction Service is responsible for removing data to keep to the Data Store compliant to sizing or data-offer constraints. The administrator can handle the eviction of products through the Eviction panel here below.

The eviction service of the Data Hub can be configured to roll out a certain amount of products per time unit. The rolling process follows a data-offer criteria in the shape of a *minimal* product keeping period and a system safety check in the shape of available disk space. This last criteria is the primary trigger for the eviction, this means that the eviction service is activated if the configured availability of the disk space is exceeded. Consequently, if the configured disk space is not exceeded, products will be kept for more than the configured minimal keeping period.

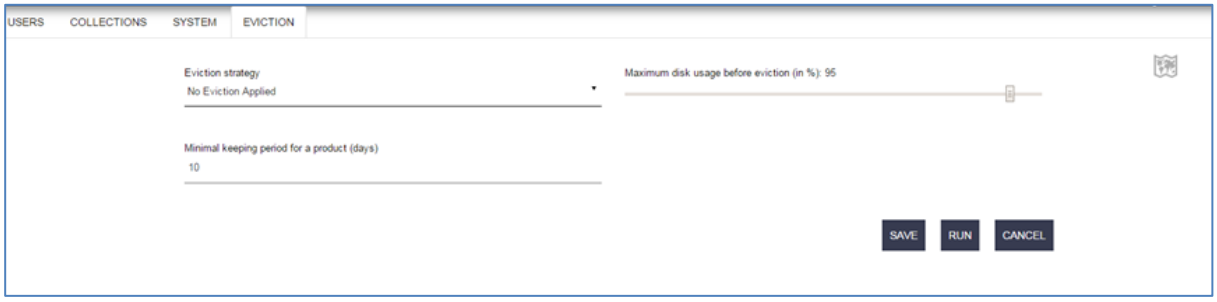


Figure 34 Eviction Management panel (GUI)

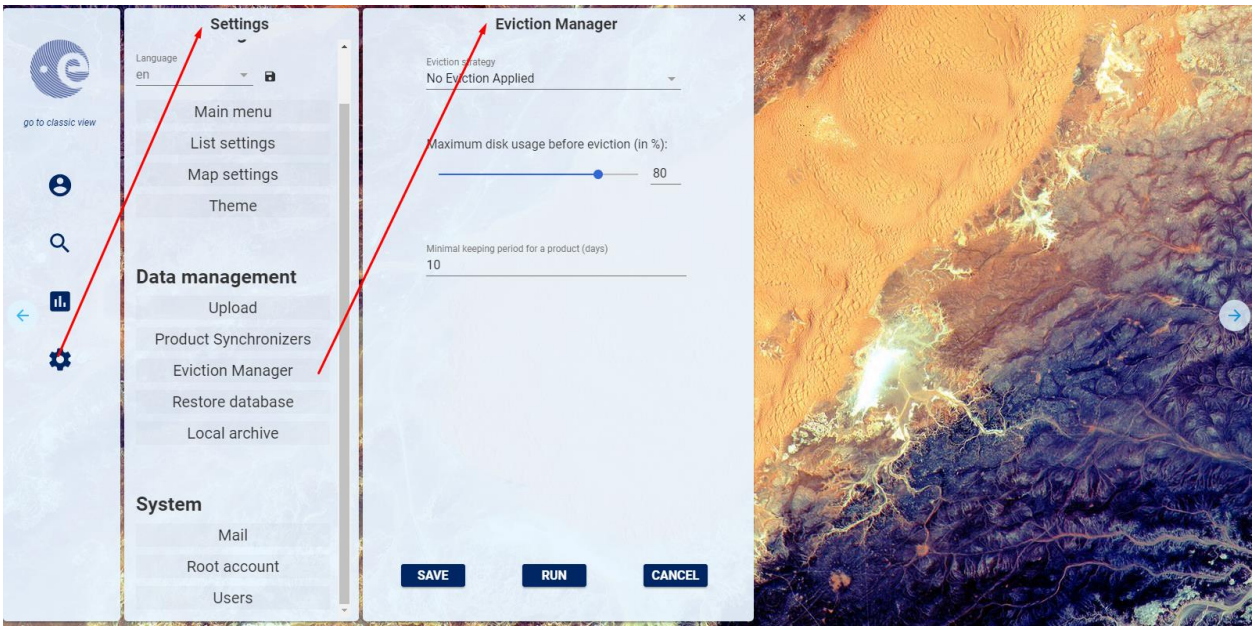


Figure 35 Eviction Management panel (OWC)

The eviction manners are:

- First In First Out (FIFO);
- Least Recently Used (LRU) ←currently not available

They can be chosen through the drop-down menu named "Eviction strategy".

Such rules are applied based on the product creationdate metadata, so for example, selecting the FIFO manner, the product with oldest creationdate will be the first to be deleted.

How to activate the archive rolling policy?

In order to activate the eviction, perform the following steps:

- **Step 1**: Access the DHuS page:
- **Step 2**: Perform the login;

- **Step 3**: Select the Management Panel and then select the Eviction management panel;
- **Step 4**: Select the Eviction strategy using the drop-down menu;
- **Step 5**: Configure the "Maximum disk usage before eviction" depending on how much of the machine space can be occupied by data before triggering the eviction (e.g. if the parameter is set to 80, when the disk will be full at 80%, the eviction will be automatically activated)
- **Step 6**: Configure the "Minimal keeping period for a product" parameter. This parameter represents the number of days each product will be kept in the DHuS archive before being evicted (e.g if the parameter is set to 3, the eviction will delete all the products present in the archive for more than three days.)

Please note that the following two eviction parameters are not configurable via UI, so they shall be customized in the dhus.xml:

- schedule for the eviction run (see A.1 `[crons]/eviction` tag)
- number of products to be evicted at each run (see A.1 `[System]/archive` tag)

Please note that a Data Hub has an eviction policy set, the metadata "EvictionDate" available under [DHuS_address]/dhus/odata/v1/Products('UUID') reports the date calculated as CreationDate+keepingperiod. Please refer to RD 6 for further details.

4.1.4.5 Main Menu Settings Panel

In OWC, with the Administrator privileges, it is possible to manage the accessibility to all the functionalities/items in the left side of the Home web page (green arrows), by means of the Main Menu Panel in the Settings section.

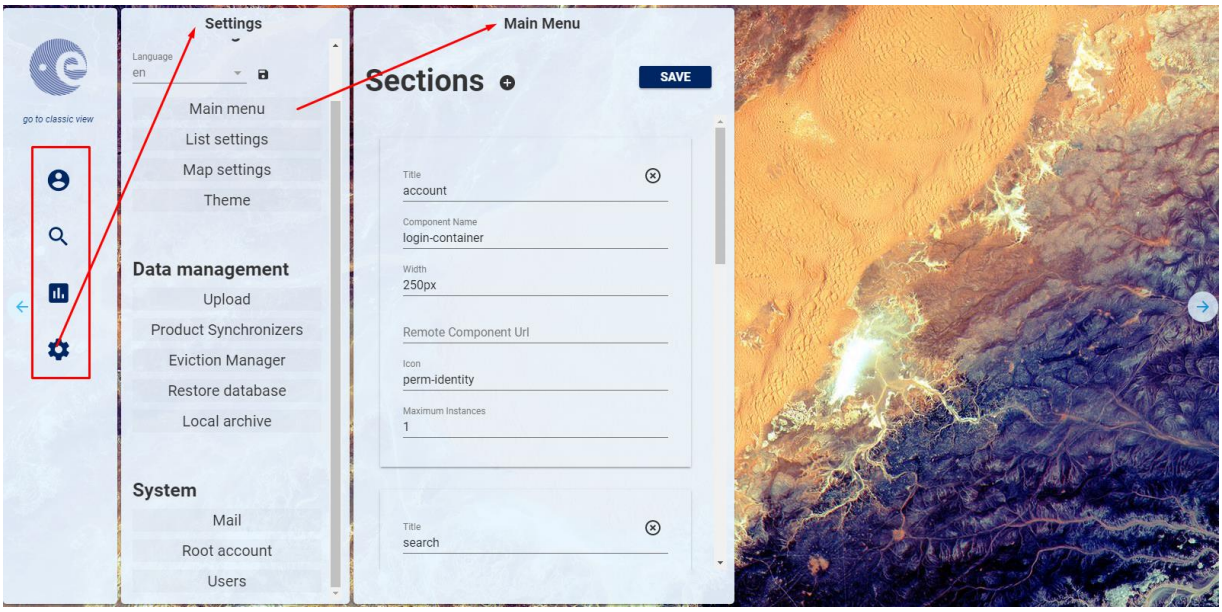


Figure 36 Main Menu Settings panel (OWC)

4.1.5 OData Synchronizer Panel

The DHuS provides end users an OData synchronizer service able to populate a DHuS instance with the data stored on the rolling archive of another DHuS instance. The DHuS instance containing the data to be synchronized is called “back end” instance, while the one that shall receive the data is called ""front end"" instance.

Before delving into the configuration details, it is worth clarifying the following terminology used for operational functions:

- **Metadata synchronization**: Copy of product metadata from a site to another. The copy is performed according to predefined synchronization selection criteria

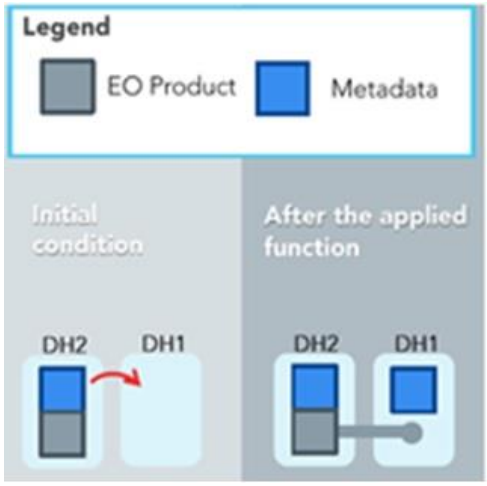


Figure 37 Metadata Synchronization

- **Product/Remote Synchronization**: Copy of product and metadata from a site to another. This is different from a product re-ingestion since during the product synchronization the UUID of the product is not re-generated but it is synchronized from the BE.

Figure 38 Products Synchronization

In case the rolling archive of the Back End contains some products not present in the Front End, once the synchronization runs, the synchronization mirrors products present in the Back End instance respecting the synchronization specification (according to the chosen configuration) and that are not in the database of the Front End instance (in case of metadata synchronization, only the metadata will be mirrored).

The OData Synchronizers panel is the User interface allowing the creation and update of synchronizers among two or more DHuS instances.

Preconditions: The FE/BE instances shall be configured as follows:

- BE: DHuS instance with no quota limitation and having a user with the 'archive management' function enabled.
- FE: having the synchronization functionality enabled, meaning that the dhus.xml of the FE shall contain the following setting:

```
<executor enabled="true" batchModeEnabled="false">
```

In case a proxy has been set up between BE and FE, please change the network properties in order to include the BE in the list of IP that can access to BE without going through the proxy.

Warning: Please note that it is not possible to do metadata synchronization between two Front-end instances; in fact, the metadata synchronizer needs to access the localpath property of products to complete the synchronization of metadata with success. In a DHuS instance, the localpath property is not visible for products synchronized, but only for products ingested.

How to create a new Synchronizer via UI?

The Administrator shall:

- **Step 1**: Log in as Root in the front end DHuS instance and select the tab "user profile"
- **Step 2**: Select the panel "OData synchronizers"

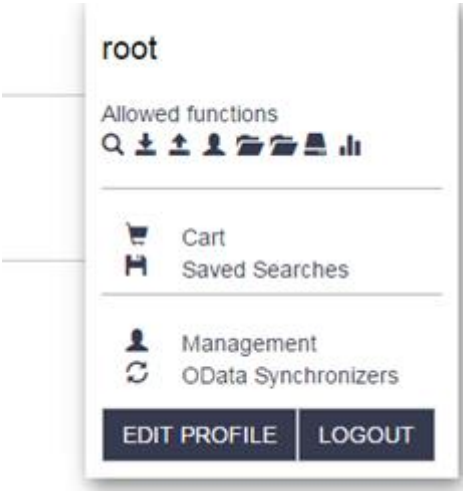


Figure 39 OData Synchronizer Access

- **Step 3**: Click on "Create synchronizer"

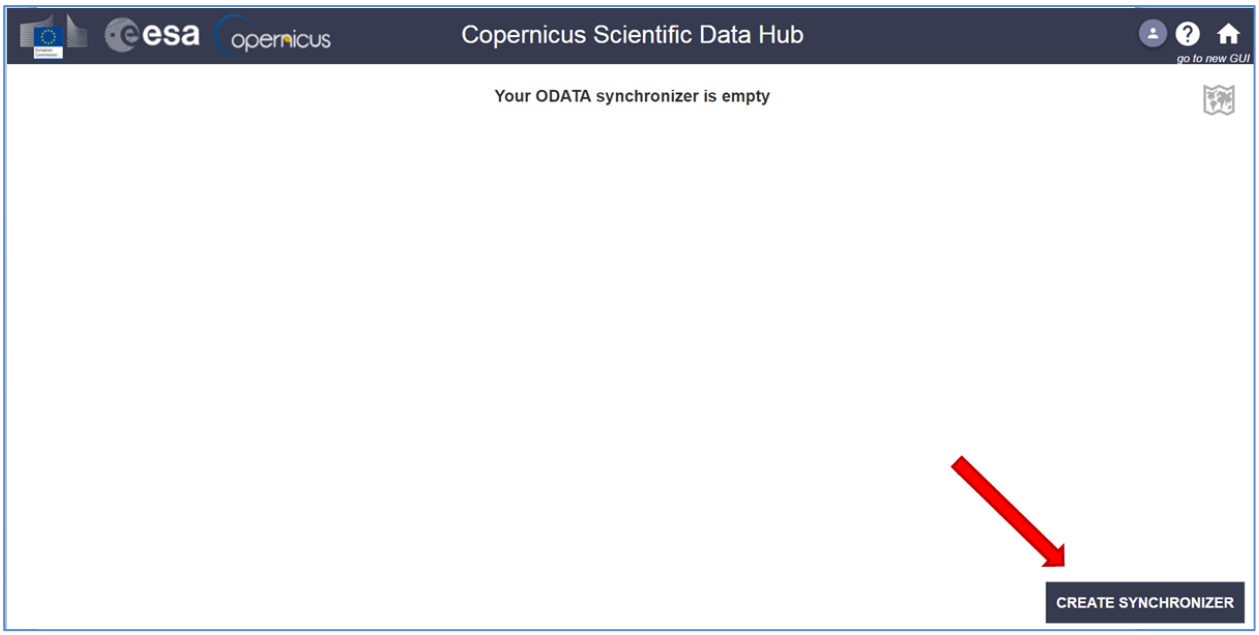


Figure 40 OData Synchronizer Panel (AJS)

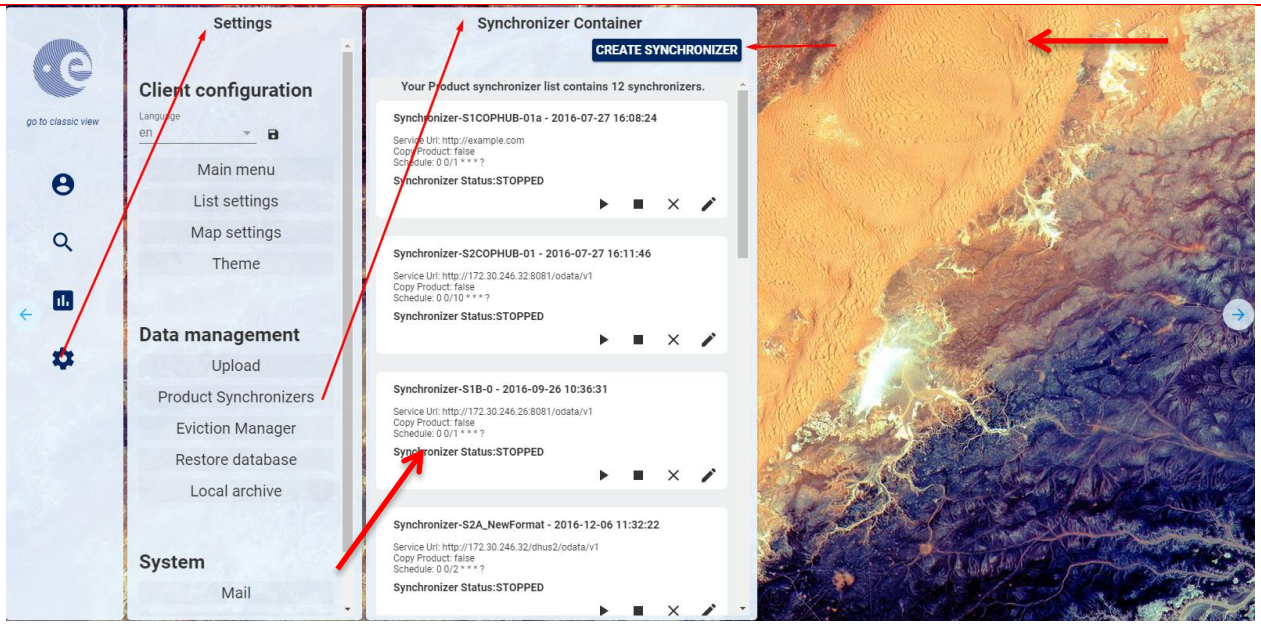


Figure 41 OData Synchronizer Panel (OWC)

- **Step 4**: Fill the records as follows:

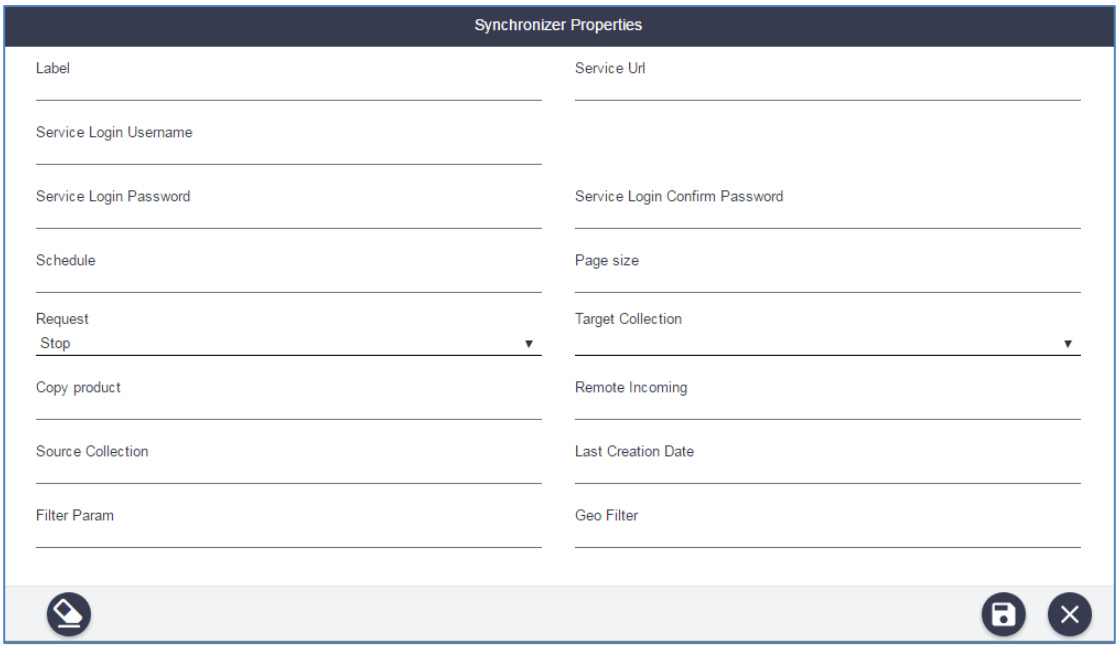


Figure 42 Create Synchronizer Panel (AJS)

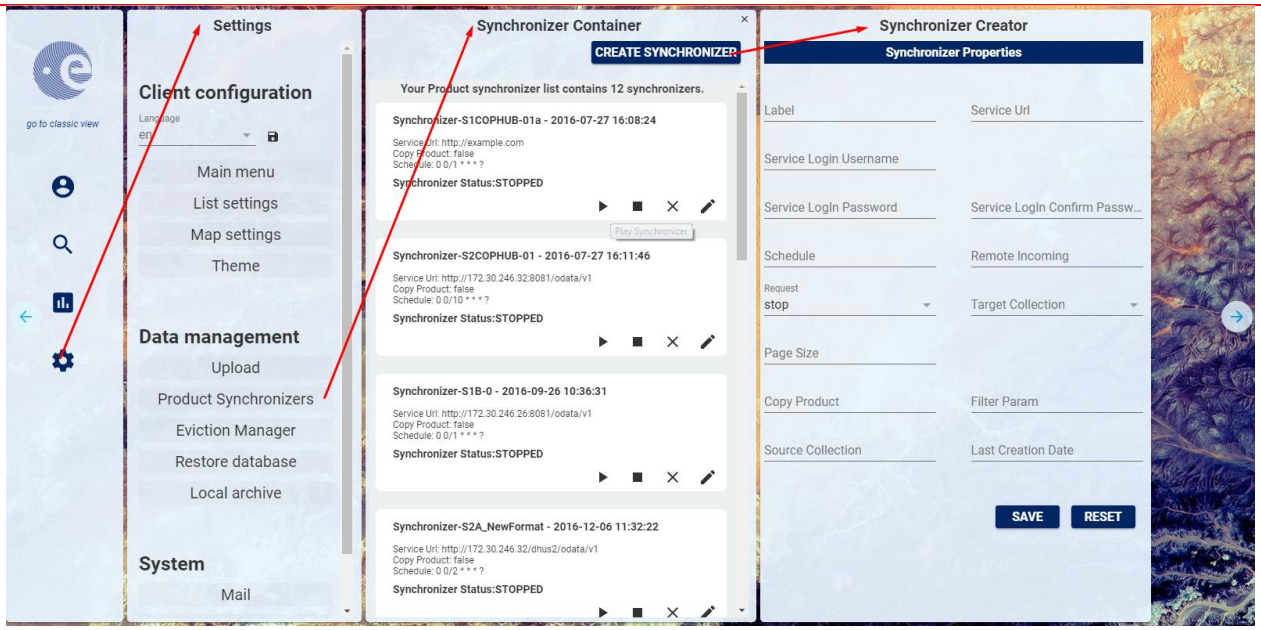


Figure 43 Create Synchronizer Panel (OWC)

- **Label**= Name of the synchronizer
- **Service URL**= `https://[Back-End_DHuS_address]/odata/v1`
- **Service Login Username**= User name of a user registered in the back end. In case of metadata synchronization, the archive manager rights enabled are requested for this user, otherwise the synchronization between Back End and Front End would be incomplete. In case of product synchronization, a normal user (meaning with search and download rights) can be used.
- **Service Login Password**= password of the user in the previous step.
- **Schedule**= how often the synchronizer shall be running. This shall be configured according to the crontab syntax.
- **Remote incoming**= path of the incoming folder configured for the DHuS installed as the Back End instance. The role of the remote incoming is linked to the kind of synchronizer: metadata synchronizer ("Remote Incoming" field shall be set) or product synchronizer (empty "Remote Incoming" field).

  Moreover, the configuration of such depends also on the datastore used by the BE instance, please find below a summary of the different possible configurations:

	Synchronizer type	Remote incoming	Notes
BE incoming in HFS	Product	N	
	Metadata	Y	Absolute path of the incoming folder configured for the DHuS installed as the Back End instance. The FE will have access to products, quicklook and thumbnails that remain in the BE incoming folder.
BE incoming in	Product	N	

openStack	Metadata	N	The dhus.xml <datastore> section of the FE instance shall report the description of the openstack datastore of the BE (where the products are located) in readOnly=true (see Appendix A dhus.xml datastore section). The FE will have access to products, quicklook and thumbnails that remain in the BE openstack container.

Table 7 Synchronizer type vs Remote incoming configuration

One last possible configuration for the remote incoming is the following: "Copy product" field is set as false (metadata synchronizer) and the remote incoming is not set. In this case, the FE will basically be a catalogue of metadata. This means that it will not have access to products, quicklook and thumbnails.

- **Request**= "start" or "stop"
- **Target Collection**= is a nullable and modifiable property with no default value. If specified, the synchronized products will be filled in the referenced FE Collection. Example: `collection_name_FE`.
- **Page size**= number of products synchronized at each synchronizer run. The PageSize parameter for the OData product sync with copy is the number of downloads happening in parallel, however to avoid idling download threads, the download task queue is overfed, thus you may have between 1 and PageSize * 2 download tasks submitted. It is suggested to set this parameter lower than the total number of threads allowed in the system.
- **Copy product**= this parameter determinates if the synchronization is a metadata synchronization or is a products synchronization:
    - **False**: the synchronizer will synchronize only the metadata of the products (the products are stored only in the Back End incoming folder).
    - **True**: the synchronizer will make a copy of the products in the Front End incoming folder during the synchronization.

Note that if this parameter is set as true, the "Remote Incoming" field shall be set as empty.

- **Filter Parameter**= following the OData filters syntax (see [RD 6]), it filters the products and synchronizes only the ones respecting the filter. Example: `substringof('S1A_',Name)`. For reference see : https://scihub.copernicus.eu/userguide/5APIsAndBatchScripting
- **Source Collection**= Back End Collection filter: only the products on the Back End instances, which are belonging to the configured collection, will be synchronized to the Front End instance. Example: `Collections('collection_name')`. If this field is left empty, the synchronizer will synchronize every products without any check on their Back End collection.
- **Geofilter** = Geographical filter to synchronize only products belonging to a specific area from the BE. This filter is a post-filter, which means that it is executed client-side, not server-side.

This Property is a String with a specific syntax: <OPERATOR> <Well-Known-Text Shape>.

<OPERATOR> can be:

- *within* (a product's footprint in inside the given shape)
- *contains* (a product's footprint contains the given shape)
- *disjoint* (a product's footprint and the shape don't overlap)
- *intersects* (a product's footprint and the shape overlap)

<Well-Known-Text Shape> is a topologically valid shape written in WKT (See: https://en.wikipedia.org/wiki/Well-known_text)

Example

To synchronise products whose footprint overlap with Iceland, it could be possible using the following GeoFilter value:

```
intersects       POLYGON((-24.61903123097289       63.345943833554685,       -
13.125342536439836          63.345943833554685,          -13.125342536439836
66.61007811487349, -24.61903123097289 66.61007811487349, -24.6190))
```

- **LastCreationdate** = Filter on the Creationdate of the products stored in the Back End instances. Products with creation date >= of the one here indicated will be synchronized. This not-nullable property is used by the OData product synchronizer to fetch pages of products in the right order. Once the synchronizer ends its run, this field is updated with the creation date of the last synchronized product.

  Please, note that the field Creationdate is not synchronized from BE to FE, so:

  - the "creation date" of a product in the BE represents the availability date on the BE instance of the DHuS during the ingestion process (its value is set to the current Date when the row is inserted in the Database slightly antecedent to the ingestion date)
  - The "creation date" of a product in the FE represents the moment in which the product has been published in the FE instance.

Note: Until 0.12.5-6, the OData product synchronizer used the LastIngestionDate instead of LastCreationdate (using the same approach described above).

- **Step 5**: Click on the button with the "floppy disk" shape.

How to update a Synchronizer via UI?

The Administrator shall:

- **Step 1**: Log in as Root in the front end DHuS instance and select the tab "user profile";
- **Step 2**: Select the panel "OData synchronizers", stop the synchronizer clicking on the "squared" icon, then click on the "pencil" next to the synchronizer to be updated

Figure 44 Updating a synchronizer

- **Step 1**: Edit the records to be updated
- **Step 2**: Click on the button with the "floppy disk" shape and re-start the synchronizer

How to delete a Synchronizer via UI?

The Administrator shall:

- **Step 1**: Log in as Root in the front end DHuS instance and select the tab "user profile";
- **Step 2**: Select the panel "OData synchronizers" and then click on the "X" shaped button next to the synchronizer to be updated.

Next to an existing synchronizer tab, there are also buttons for starting and stopping the item. The "play" button is to start the synchronizer; the "square" button is to stop it.

4.2 Administration API

This section reports the interfaces available for DHuS administrators:

1) Information concerning user details, connections and networks can be visualized via OData by means of the entities "Users", "Connections" and "Network" described in the following paragraphs and in RD 6).
2) Product synchronization, user synchronization, product deletion and ingestion can be managed by means of the OData service operations described in the following paragraphs and in RD 6 .

4.2.1 User details information

It is possible to visualize user details via OData by accessing to the entity "Users" at https://DHuS_URL/odata/v1/Users

Users with the user administration right are allowed to details of all the users registered in the Hub, whereas normal users are allowed to visualize only their own.

Please refer to [RD 6] for further details about the "Users" OData entities and its information such as system roles, connections, restrictions and user cart.

4.2.2 Connections information

Administrators can visualize all the connections managed by DHuS via OData if server.xml file has been configured properly (seeAppendix A). In fact, every user request is available in the "Connections" entity.

Moreover it is possible to monitor the activity of a specific user visualizing the requests sent by him/her at https://DHuS_URL/odata/v1/Users('Username')/Connections

Please refer to [RD 6] for the list of information exposed for each connection stored in the DHuS cache.

4.2.3 Network information

OData exposes to administrators some information about the network used by the Hub. To access such infos it is necessary to connect to the "Networks" at https://DHuS_URL/odata/v1/Networks

Please refer to [RD 6] for the information exposed for each Network stored in the DHuS cache and its Statistics.

4.2.4 Product Synchronization

How to create a new Product synchronizer?

- **Step 1:** Log in the machine where the Front End DHuS (definition of Front end in section 4.1.5) is installed.
- **Step 2:** On the DHuS acting as FE, set the environment variables (export <variable>=value) that configure the DHuS:
  - DHOST DHuS Host address eg: localhost:8080, scihub.copernicus.eu/dhus/
  - DLOGIN DHuS login username to connect to the $DHOST DHuS
  - DPASS DHuS login password to connect to the $DHOST DHuS
- **Step 3:** Create a working folder containing the scripts available at https://github.com/SentinelDataHub/Scripts/tree/master/dhus_sync_script
- **Step 4:** To create the synchronizer use the command:

```
./createSynchronizer <-D_SCHEDULE=cron_expression> <-D_SERVICEURL=URL_to_remote_DHuS_to_sync> <-D_SERVICELOGIN=account> <-D_SERVICEPASSWORD=password> [-D_LABEL=my_sync] [-D_PAGESIZE=X] [-D_REQUEST=start|stop] [-D_COPYPRODUCT=true|false] [-D_FILTERPARAM=filter_expression] [-D_SOURCECOLLECTION=resource/path] [-D_LASTCREATIONDATE=date] [-D_GEOFILTER=geofilter]
```

Where parameters between angle brackets (chevrons) are mandatory, parameters between brackets ([]) are optional. For details about the parameters meaning, please refer to section 4.1.5.

Please note that there are options to nullify nullable properties: `-D_LABEL_NULL`, `-D_FILTERPARAM_NULL`, `-D_SOURCECOLLECTION_NULL`, `-D_GEOFILTER_NULL`

4.2.5 User Synchronization

The DHuS provides an OData User synchronizer service able to populate a DHuS instance with the users stored on the rolling archive of another DHuS instance. The user synchronizer shall be created in the instance where

the users are going to be imported (Front-end) and it can be activated by users with "user manager" rights (see section 4.1.5 for the definition of FE and BE).

The user synchronizer retrieves users (from a back-end) following the user creation date criteria, meaning that it retrieves users from the oldest to the most recent. Here follows an example of the query performed by the User synchronizer:

http://DHuS_IP:8081/odata/v1/Users?$top=100&$orderby=Created

How to create a User synchronizer?

- **Step 1:** Log in the machine where the FE DHuS is installed. This DHuS instance shall be filled with users registered in the BE.
- **Step 2:** On the DHuS acting as FE, set the environment variables (export <variable>=value) that configure the DHuS:
  - DHOST DHuS Host address eg: localhost:8080, scihub.copernicus.eu/dhus/
  - DLOGIN DHuS login username to connect to the $DHOST DHuS
  - DPASS DHuS login password to connect to the $DHOST DHuS
- **Step 3:** Install M4 Macro Language package running
  - `apt-get install m4` (on Debian platforms)
  - `yum install m4` (on RedHat/CentOS platforms)
- **Step 4:** Create a working folder containing the scripts available here https://github.com/SentinelDataHub/Scripts/tree/master/dhus_sync_script
- **Step 5:** To create the user synchronizer use the command:

```
./createSynchronizer <-D_SCHEDULE=...> <-D_REQUEST=start|stop> <-
D_SERVICEURL=...> [-D_LABEL=...] [-D_SERVICELOGIN=...] [-D_SERVICEPASSWORD=...]
```

Example:

```
. /createSynchronizer -D_SCHEDULE="0 */3 * * * ?" -D_REQUEST=stop \
-D_SERVICEURL="http://192.168.0.105:8080/odata/v1" -D_LABEL=my_user_syncer \
-D_SERVICELOGIN=root -D_SERVICEPASSWORD=a
```

Where:

- **Schedule**= how often the synchronizer shall be running. This shall be configured according to the crontab syntax
- **Request**= "start" or "stop"
- **Label**= Name of the synchronizer
- **Service URL**= https://[Back-End_DHuS_address]/odata/v1
- **Service Login Username**= User name of a user registered in the back end with user manager role.

- **Service Login Password**= password of the user in the previous step
- **Page size**= number of users synchronized at each synchronizer run
- **Cursor=** if the cursors is set as "i" (where i is such that 0<i<total amount of users) the synchronizer will import in the FE all the users of the BE from the i-th.
- **Force=**true/false. This parameter defines if this UserSynchronizer should overwrite namesakes. If Forced is set to true, namesakes will be overwritten thus after a complete synchronisation of all users from the data source, there will be no namesakes on the synchronising DHuS.

How to update a user synchronizer?

To update an existing synchronizer execute the following command:

```
./updateSynchronizer <id> [-D_SCHEDULE=...] [-D_REQUEST=start|stop] \
[-D_SERVICEURL=...] [-D_LABEL=...] [-D_SERVICELOGIN=...] [-D_SERVICEPASSWORD=...]
```

Example:

```
./updateSynchronizer 0 -D_SCHEDULE='0 24 * * * ?'
```

How to delete a user synchronizer?

To delete an existing user synchronizer launch the following command:

```
./deleteSynchronizer <id>
```

4.2.6 Products Deletion

The Data Hub provides to administrators the possibility to delete products via OData.

All deleted (and evicted) products are accessible to all kind of users at http://localhost:8080/odata/v1/DeletedProducts

like normal Products (at http://localhost:8080/odata/v1/Products).

How to delete a product via Odata?

- With no cause:

```
curl -u user:password "http://localhost:8080/odata/v1/Products('91890b5e-049a-47c9-97c1-cb9542a694b5')" -X DELETE
```

In this case, the DeletionCause is « null »

- With a generic cause:

```
curl -u user:password "http://localhost:8080/odata/v1/Products('91890b5e-049a-47c9-97c1-cb9542a694b5')?cause=Size%20Needed" -X DELETE
```

- Determining it as "Invalid" :

```
curl -u user:password "http://localhost:8080/odata/v1/Products('91890b5e-049a-47c9-97c1-cb9542a694b5')?cause=Invalid" -X DELETE
```

The 'cause' field is planned to be used (in future implementations) by OData Synchronizers to detect invalid products, with a regular expression for example. So, it is strongly recommended to identify invalid products by starting their deletionCause with "Invalid". Such practice will make easier to set up a regular expression in future (for example : "Invalid, bad geometry" or "Invalid, reprocessed")

Please note that evicted products have their DeletionCause automatically set to "Automatic Eviction"

For further details about how the deletion works in a "Front-end/Back-end" configuration, please refer to 4.1.2.

4.2.7 Products Ingestion

The ingestion functionality can be triggered via OData API, the implementation of such process is currently not supporting the multi-thread management: one product at the time can be ingested.

How to ingest via OData?

- **Step 1:** Log in the machine where DHuS is installed
- **Step 2**: Download scripts to trigger the ingestion and put them in a working directory https://github.com/SentinelDataHub/Scripts/tree/master/dhus_ingest_scripts
- **Step 3:** Set the environment variables to connect to the syncing instance of the DHuS eg:
  - export DHOST=localhost:8080
  - export DLOGIN=root
  - export DPASS=a
- **Step 4:** To trigger the ingestion process use the following command:

```
./ingest <path/to/data.zip>
```

Products will be visible under http://DHuS_address/odata/v1/Ingests

How to delete an ingestion record via OData?

To delete an ingestion record via OData launch the following command:

```
curl -X DELETE --basic -u ${DLOGIN}:${DPASS}\
"http://${DHOST}/odata/v1/Ingests(ID)"
```

or launch the command:

```
./deleteIngest <ID>
```

For deleting products ingested with methods different from the OData API please refer to 4.2.6.

Appendix A. Configuration file parameters description

A.1 dhus.xml

This section explains in details the parameter contained in the `dhus.xml` configuration file. DHuS configuration is contained inside the `<configuration></configuration>` tag.

Here follows details about general attributes needed for the configuration definition:

Parameter	Description	Synthax
`xmlns`	Namespace associated to the DHuS root element	`xmlns="fr.gael.dhus.database.object.config"`
`xmlns:xsi`	Reference to the schema's location	`xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"`
`xmlns:ds`	Namespace needed for the dataStores definition.	`xmlns:ds="org.dhus.store.datastore.config"`
`workingDir`	Folder where DHuS puts its temporary files. This folder is automatically deleted when the DHuS is switched off. Currently, DHuS uses this folder during ingestion.	`workingDir="working/directory/path"`

Groups of configurable options are present inside the <configuration></configuration> tag, the following table summarize them.

[Groups]/ Parameter	Description	Usage
`[crons]/archiveSynchronization`	Cron used to synchronize local archive (whose path is the one specified in the parameter "archive" in the "system" group). In case of eviction, please schedule it after the eviction.	Activate it and specify the schedule.
`[crons]/cleanDatabase`	Cron used to clean database, like removing old statistics or old not confirmed users.	Activate it and specify the schedule. The parameter statistics and temUsers define, respectively, for how long (in days) the system have to keep statistics and how many time (in days) the users have to confirm their registration request
`[crons]/dumpDatabase`	Cron used to dump database	Activate it and specify the schedule.

[crons]/cleanDatabaseDump	Cron used to clean database dumps.	Activate it and specify the schedule. The parameter keep defines how many dumps are stored.
[crons]/eviction	Cron used to evict products when it is required	Activate it and specify the schedule.
[crons]/fileScanners	Cron used to execute user saved filescanners.	Activate it and specify the schedule. The parameter sourceRemove defines if found products shall be removed from source
[crons]/searches	Cron used to execute user saved searches and send results to users	Activate it and specify the schedule.
[crons]/sendLogs	Cron used to send system logs.	Activate it and specify the schedule. The parameter addresses logs recipients addresses. They shall be coma-separated.
[crons]/systemCheck	Cron used to check all system coherence, including database optimization	Activate it and specify the schedule.
[Messaging]/mail	Mail configuration. - onUserCreate: defines if system send mail when creating user - onUserUpdate: defines if system send mail when updating user - onUserDelete: defines if system send mail when deleting user These values are used only at first launch of the system. They shall be modified in Management Panel if needed.	Every notification type needs to be activated. In case of activation of one or more notification service, a sender mail server, its name ('from') and the email address in case the user wants to reply ('reply to') are required: -server: -smtp: server address -port: server port - tls: defines if server is using TLS protocol - username and password: connection information from: - name: displayed name of "from" part of the notification email - address: displayed address of "from" part of the notification email - reply to: defines the "reply to" address of sent mails
[Network]/outbound	outbound bandwidth configuration. This parameters are recharged	Parameters to be configured: channel name and its weight, including the user email pattern identifying the oubound channel, set defauld user

	every time the dhus is restarted.	quota including maxconcurrent, max count, max size, max cumulative size and max bandwidth: - `maxConcurrent` defines the maximum simultaneous accepted transfers. - `maxCount` defines the maximum number of accepted transfers on a customizable period. "periodUnit" attribute defines the unit of the period. Possible units are "DAYS", "HOURS", "MICROSECONDS", "MILLISECONDS", "MINUTES", "NANOSECONDS", "SECONDS"." period" attribute is a sliding time window used to count number of transfered product from now to the past period delay. - `maxSize` defines the maximum accepted size of transfer file. - `maxCumulativeSize` defines the maximum cumulated accepted transfers size on a customizable period. - `maxBandwidth` defines the maximum bandwidth authorized for the referred channel.
`[Network]/inbound`	inbound bandwidth configuration	see above
`[Products]/download`	Download configuration. `compressionLevel`: is a value to build the file compression, from 1 to 9. The compression level defines how rigorously the compressor looks the longest string possible. As a general rule of thumb: Compressing at the maximum level (9) requires around twice as much processor time as compressing at the minimum level (1) • For typical input, compressing at the maximum as opposed to the minimum level adds around 5% to the compression ratio. 0 value means no compression. `checksumAlgorithms`: algorithms to be used for checksum	Specify the compression level. The scale is 1 to 9. Specify checksum algorithms and configure them in comma-separated format

	computations **Warning**: `publicData`="false" is a deprecated parameter, so every products are published even if this is set to false	
`[Products]/quicklook`	Quicklook calculation parameters: • height: height of generated quicklooks • width: width of generated quicklooks • cutting: allow system to cut image when processing quicklooks	Specify dimension in pixel unit To activate the cutting process set the parameter to "true"
`[Products]/thumbnail`	Thumbnail calculation parameters: • height: height of generated thumbnails • width: width of generated thumbnails • cutting: allow system to cut image when processing thumbnails	Specify dimension in pixel unit To activate the cutting process set the parameter to "true"
`[Search]/geocoder`	Geocoder (nominatim) configuration	Set up the Nominatim geocoder and the Geoname configurations: - `boundingBox`: defines if the geocoder is querying only the bounding box of the matching place from the Nominatim Web Service i.e. the four corners encompassing the place. Otherwise, it will query the complete polygon boundaries, that may have lower performance according the the number of vertices composing the place's boundaries. - `maxPointNumber`: maximum number of points that can be returned for a polygon - `username`: username used to connect to Geoname
`[Search]/odata`	Odata configuration.	

	• maxRows: maximum rows returned by Odata Service **Warning**: this parameter is no longer available. Please refer to A.3 to limit the OData response	
[Search]/solr	Solr configuration. • path: solr path • core: solr core name • schemaPath: solr schema path. Shall be empty. • synonymPath: path of solr synonyms file	Specify the parameters. Example: <solr path="&varFolder;/solr" core="dhus" schemaPath="" synonymPath="" />
[Server]/external	External url viewed by users. Used in case of an apache proxy redirection for example. Empty values mean that server values are those which are viewed by users.	Example: protocol://host:port/path
[Server]/ftp	FTP server configuration. • port : ftp port • ftps : using ftps protocol or not • passive port: port to be used for the data channel	Example: <ftp port="2121" ftps="false" passivePort="30200-30220" />
[System]/administrator	Definition of principal administrator user. If User exists, DHuS will give him all rights at launch, but will not his password. This shall be done in Management panel of GUI. If User is not existing, DHuS will create it with defined password.	Specify username and password
[System]/archive	Definition of local archive path (incoming) and the eviction settings **Warning**: please note that the	This configuration includes also the following parameters: 'Eviction' which is the configuration of the rolling

	configuration of "incoming" folder on an HFS datastore can be either configured in this block or in the block "Datastore". This double option has been kept to guarantee backward compatibility with DHuS releases oldest than 0.13.4.	policy, in details: • `maxDiskUsage`: the maximum disk usage that can be allowed for evictable products • `maxEvictedProducts`: the maximum evicted products when running an eviction `keepPeriod`: the minimal time in days • `trashPath`: path where to move deleted products 'Incoming' which is the folder path and the number of maximum subfolders of each stage. • `path`: the path of the incoming • `maxFileNo`: maximum number of sub-folder for each stage • `errorPath`: path where to move products not ingested
`[System]/database`	Definition of database path, where dumps are stored and, optionally, the encryption type and encryption key. Warning: please note that, from the 0.10.3-4 version, the default behaviour is no encryption.	The default configuration of the dhus.xml dows not contain the encryption Type and the encryption key parameters. The shall be inserted manually by the operator as in the following example: `<database path="&varFolder;/database/hsqldb" dumpPath="&varFolder;/database_dump" cryptType="*TO_BE_SPECIFIED*" cryptKey="*ENCRYPTION_KEY*"/>`
`[System]/name`	Definition of system long name and short name.	Example: <name long="IVV Data Hub Service" short="IVV DHuS" />
`[System]/processing`	Processing configuration. • corePoolSize: defines maximal number of active threads for the ingestion process. Default is 1	Configure the corepoolsize as an integer number (>0)
`[System]/support`	Definition of support name and mail. • registrationMail: used to send the administrative registration	

	information. If this field is not set, DHuS is using support mail. These values are used only at first launch of the system. They shall be modified in Management Panel if needed.	
`[System]/tomcat`	Definition of tomcat path	Example: <tomcat path="&varFolder;/tomcat" />
`[System]/executor`	`Executor` (background service that executes synchronizers) It must be enabled if you want to use the synchronisation feature. `batchMode`: the executor will run the synchronizers until there is no more to synchronize.	Activate/Deactivate the two parameters (values: true or false)

`[Datastore]/datastore`	It is possible to configure multiple dataStores in DHuS; two kinds of dataStores are currently available: File system (HFS) and Openstack. Here follows details about available attributes:
	• `type`: define the dataStore implementation to use ("*hfsDataStore*" and "*openStackDataStore*")
	• `name`: dataStore name
	• `readOnly`: true/false. If it is set to true the adding, moving and deleting actions are not supported for the dataStore
	• `path` (only for HSF dataStore): local storage path
	• `maxFileNo` (only for HSF dataStore): maximum number of sub-folder for each stage
	• `provider` (only for Openstack dataStore): provider service to use, only "*openstack-swift*" is currently available
	• `identity` (only for Openstack dataStore): identifier for the authentication service
	• `credential` (only for
		Example of configuration for HFS dataStore:
		`<ds:dataStore xsi:type="ds:hfsDataStoreConf" name="datastore1_name" readOnly="true"> <ds:path>/data/dhus-incoming</ds:path> <ds:maxFileNo>10</ds:maxFileNo></ds:dataStore>`
		Example of configuration for Openstack dataStore:
		`<ds:dataStore xsi:type="ds:openStackDataStoreConf" name="datastore2_name" readOnly="false"> <ds:provider>openstack-swift</ds:provider> <ds:identity> tenantName:username</ds:identity> <ds:credential>password</ds:credential> <ds:url>https://keystone9915.openstack.ovh.net:35358/v2.0</ds:url> <ds:container>container_name</ds:container> <ds:region>RegionOne</ds:region> </ds:dataStore>`

	Openstack dataStore): *project:password* of an openstack account • `url` (only for Openstack dataStore): URL of Openstack authentication service • `container` (only for Openstack dataStore): container to use. Please note that the container shall be created manually before configuring this parameter. • `region` (only for Openstack dataStore): Each Region has its own full Openstack deployment, including its own API endpoints and networks . The region is linked to the openstack account. The default value is *regionOne* . **Warning**: it is necessary to configure a DataStore for each of the incoming folders where the products are located.	

A.2 server.xml

The following table summarize the configurable parameters of the server.xml file.

[Groups]/ Parameter	Description	Synthax

`[Service]/Connector`	It is possible to configure several connectors in DHuS, two of them are configured by default. Such connectors define ports on which the application is in listening mode.
	
	The first default connector is the one for port 8081 which is the port towards which external requests are directed.
	
	The port 30333, configured in the second DHuS connector, is used for internal requests (when OpenSearch requests are sent to DHuS, the application redirect such request internally towards 30333).
		`<Connector port="8081"`
		
		`protocol="org.apache.coyote.http11.Http11NioProtocol"`
		`maxConnections="1000"`
		`maxThreads="400"`
		`keepAliveTimeout="2000"`
		`URIEncoding="ISO-8859-1"`
		`compression="on"`
		`compressionMinSize="1024"`
		
		`compressableMimeType="application/json,application/javascript,application/xhtml+xml,application/xml,text/html,text/xml,text/plain,text/javascript,te`

		xt/css" /> <Connector port="30333" protocol="org.apache.coyote.http11.Http11NioProtocol" maxConnections="1000" maxThreads="400" keepAliveTimeout="2000" URIEncoding="ISO-8859-1" compression="on" compressionMinSize="1024" compressableMimeType="application/json,application/javascript,application/xhtml+xml,application/xml,text/html,text/xml,text/plain,text/javascript,text/css" />
[Service]/Engine/Valve	DHuS keep track of the requests it manages in a Rolling cache. From this section of the server.xml is possible to define which kind of request/connections should be recorded in the cache. The default configuration (reported in the usage here on the right) is such that DHuS record in its rolling cache every request it manages (OData, OpenSearch and UI request). Cache is reset at dhus stop.	Access Filter Settings are pattern: the regular expression to filter user request i.e. "^.*(/odata/v1/).*$" only manages odata request "^((?!/(home\|new)/).)*$" consider all request but the UI. useLogger="true|false" show or hide the user access in logger output. This setting does impact keeping internal track of the request. enable="true|false" activate/deactivate the valve. Here an example matching all the requests managed by DHuS. <Valve

		`className="fr.gael.dhus.server.http.valve.AccessValve"`
		`pattern=".*"`
		`useLogger="true"`
		`enable="true" />`

A.3 start.sh

DHuS command line follows Java VM standard command lines.

Parameter	Description & Usage	Optional/ Mandatory
`country.synonyms`	/path/to/file path to the definition of users countries synonyms	O
`webapp.excluded`	webapp the name of the webapp to not start at system startup (i.e. "fr.gael.dhus.gwt.GWTWebapp")	O
`Archive.check`	=true|false (default=false) force system check at dhus startup	O
`Archive.forceReindex`	true|false (default=false) force all the products indexes being re-indexed.	O
`Archive.incoming.relocate`	true|false (default=false) force the relocation of all the products of incoming	O
`Archive.incoming.relocate.path`	/path/to/relocation (default="") give the new location path to relocate incoming directory. If no pas is provided, incoming will be relocated in its current directory	O
`Archive.processings.clean`	true|false (default=false) clean all the interrupted processing instead of recover them.	O
`force.public`	true|false (default=false) force all the product contained into DHuS become public. This parameter is deprecated.	O

Archive.synchronizeLocal	true|false (default=false) force re-synchronization of local archive path at system startup.	O
users.search.notification.force.inactive	true|false (default=false) deactivates all the user search notifications.	O
checkUserConfiguration	true|false (default=false) activates schema aware validation of input xml configuration file	O
dhus.solr.reindex	true|false (default=false) recreate the Solr index from the database	O
action.record.inactive	true|false (default=false) full deactivates read/write statistics	O
dhus.scalability.active	true|false (default=false) activates scalability in DHuS	O
dhus.scalability.local.protocol	(default=http) local protocol to access this DHuS	O
dhus.scalability.local.ip	local ip to access this DHuS	O
dhus.scalability.local.port	(default=8080) local port to access this DHuS	O
dhus.scalability.local.path	local path to access this DHuS	O
dhus.scalability.replicaId	(default=-1) replica's id; if not used, this node is considered as master	O
dhus.scalability.dbsync.master	URL of dhus master (poitning on root path)	O
dhus.scalability.dbsync.clear	(default=false) clear every system record about db synchronization stored in db (not replicated data)	O

dhus.sync.download_attempts	(default=10) During product synchronization. if a download is interrupted and the remote DHuS supports HTTP ranges, the parallelised download manager will attempt to resume the download. number of download attempts (-1 for infinite, must be at least 1)	O
-Ddhus.search.innerTimeout	Inner timeout for SolR requests	O
-Dhttp.timeout.socket -Dhttp.timeout.connection -Dhttp.timeout.connection_request	Timeout for the OData synchronizers http.timeout.socket is the socket timeout, it occurs when no data has been received within the given time span, it is also known as SO_TIMEOUT. http.timeout.connection is the connection timeout, it occurs when the remote server does not answers our connection requests. http.timeout.connection_request is a timeout on a request to a connection manager, this is a purely programmatic timeout, and you should probably not set this timeout. The socket timeout is longer because it can occur in the middle of a transfer, we don't want to throw all the data that has already been downloaded to the rubbish if the remote server is not responsive only a few minutes.	O
-Dmax.product.page.size	Default=-1 limits the number of products that can be displayed at once via OData Products entity set (-1 for no limits). Please note that processing quotas shall be deactivated to let this parameter be considered by DHuS.	O
-Dmax.rows.search.value	Default=100 Parameter defining the maximum number of rows retrievable via OpenSearch	O
-Dcom.sun.media.jai.disableMediaLib	=true to be removed if media jai native library is provided. DHuS does not requires these libraries for optimization.	M

-Duser.timezone	=UTC Mandatory parameter to force the DHuS timezone to a standard, not depending on the operating system settings.	M
-Dsun.zip.disableMemoryMapping	=true Currently mandatory to avoid a crash in zip library usage.	M

A.4 log4j2.xml

Log4j2 file defines the log level of processes run by the Data Hub, in fact in each logger it is possible to raise/lower the level. The RollingFile section of the `log4j2.xml` defines the name of the application log, its location and its rolling policy.

Log4j2 is a a Java-based logging utility now project of the Apache Software Foundation. Here in below follows an overview of the configuration parameters used in the default log4j2 file. Please refer to https://logging.apache.org/log4j/2.0/manual/ for configuration details.

[Groups]/ Parameter	Description	Usage
[PROPERTIES]	Configuration and definition of the pattern used as template of the DHuS log. The pattern available in the default distribution of the log4j2.xml is such that the log has the following shape `[DHuS_version][timestamp in UTC][level][log message]`	Example: `<Property name="pattern" >[$${sys:fr.gael.dhus.version}][%d{DEFAULT}{UTC}][%-5p] %m (%file:%line - %t)%n%throwable </Property>`

[APPENDERS]/CONSOLE	Configuration of the messages logged for standard output (output of SW run) and standard error (SW errors). The following parameters are available for configuring stdout and stderr:
	• Name=name of the console
	• Patternlayout= pattern/template used in the standard output log (see above)
	• Filters: define which log
		Examples:
		`<Console name="stdout" target="SYSTEM OUT">`
		`<PatternLayout pattern="${pattern}" />`
		`<Filters>`
		`<ThresholdFilter level="DEBUG"/>`
		`<ThresholdFilter level="WARN" onMatch="DENY" onMismatch="NEUTRAL"/>`
		`</Filters>`
		`</Console>`

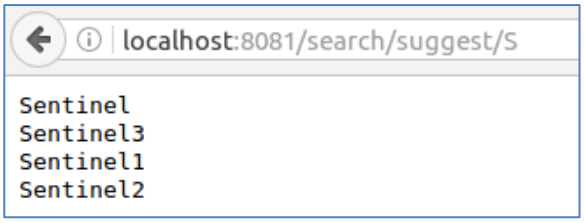
	messages shall not be logged	<Console name="stderr" target="SYSTEM ERR"> <PatternLayout pattern="${pattern}" /> <Filters> <ThresholdFilter level="WARN"/> </Filters> </Console>
[APPENDERS]/ROLLINGFILE	Configuration of the rolling policy for DHuS log file: • Name=name of the log • Filename=location and name of the real time DHuS log • Filepattern=location and name of the rolled log files • Patternlayout=see above • Policies(TimeBasedTriggeringPolicy, interval, modulate)= rule defining that all files matching a given file name pattern (filepattern) and older than a certain number of days are rolled.	Example: <RollingFile name="RollingFile" fileName="/data/dhus/ivvdhus/logs/dhus.log" filePattern="/data/dhus/ivvdhus/logs/dhus-%d{yyyy-MM-dd}.log"> <PatternLayout> <Pattern>${pattern}</Pattern> </PatternLayout> <Policies> <TimeBasedTriggeringPolicy interval="1" modulate="true" /> </Policies> <Filters> <ThresholdFilter level="DEBUG"/> </Filters> </RollingFile>
[LOGGERS]	Loggers allow using specific log levels for classes. The parameters are the following: • Name=logger name • Level= the level may be configured with one of TRACE, DEBUG, INFO, WARN, ERROR, ALL or OFF. If no level is specified it will default to ERROR.	Beyond the loggers provided by the default log4j2.xml, the following loggers are available for synchronizers, datastores and eviction services: <logger name="fr.gael.dhus.sync" level="debug"/> <logger name="org.dhus.store.datastore.hfs" level="debug"/> <logger name="fr.gael.dhus.service.ProductService" level="debug"/>

	• `Root`= root logger. The root logger does not have a name attribute). The appenders attached are by default the stdout stderr and the RollingFile	<logger name="fr.gael.dhus.service.EvictionService" level="debug"/>

A.5 suggester.dic

The file `suggest.dic` can be customized with a list of strings to suggest via /search. Once restarted the DHuS suggester can be request via http://dhusurl/search/suggest/{query} as shown her below:

Figure 45 Suggester configuration



A.6 opensearch-description-file.xml

This file describes DHuS OpenSearch interface of Data Hub service. The OpenSearch description document describes the web interface of the DHuS search engine. Please refer to RD 3 for further details about the opens-search description document.

A.7 AJS GUI

The DHuS is equipped with an AJS GUI. The files in charge of the GUI configuration management are located in:

[DHUSDIR?]/ var/tomcat/webapps/ROOT/config

- **appconfig.json**

and in [DHUSDIR?]/var/tomcat/webapps/ROOT/config

- **styles.json**

appconfig.json

The configuration file appconfig.json manages mission specific filters. A "missions" section is present in the file, containing an array with the following structure:

```
"name": , "indexname": , "indexvalue":, "filters":[filter_array]
```

where [filter_array] is an array of mission-specific filters with the following structure:

```
"indexname": "indexlabel": "regex": [OPTIONAL] "indexvalues"
```

If a set of value is configured, it appears a combo box containing the list of all specified values.

Once a value in the file has been changed, a simple refresh of the browser is necessary to see the change immediately applied. There is no need to restart the DHuS.

The configurability of the AJS GUI allows a wide set of configuration actions which do not need a restart of DHuS to be applied. Due to the growth of the different centres and related installations, a configuration management module has been added into the AJS web app. It allows configuring various aspects of the GUI; mainly it is related to style, texts and layout:

- **Title** (shown in the header bar)
- **Sections visibility** (Cart, Profile, Sign In)
- **URL and texts of the link logos** (shown in the header panel)
- **Version text** (shown in the info panel)
- **Data Hub Logo**(shown in the info panel)
- **Mission Tag**(shown in the Product List panel)
- **Mission footprint style and color** (shown in the Map panel)
- **Advanced Search Mission specific fields** (shown in Advanced Search Panel)
- **Map Layer** (shown in the Map View)

Please note that all the settings are included in the client side, thus it is possible to change a parameter without restarting the DHuS, but just doing a refresh via browser.

Here follows instructions for specific features configuration:

- **Shapefile:** The configuration parameter "`enable_shapefile`" of the file "appconfig.json" enable/disable the shapefile usage for area selection.
- **Terms and conditions link:** The T&C link is configurable through the "`t_and_c`" parameter in the "settings" section
- **List pagination:** It is possible to configure the pagination limit via the parameter "pagination_limit". The default configuration is [25,50,75,100,125,150].

  Please note that, since AJS uses the SolR engine, the pagination limit cannot exceed the row limitations configured for SolR (please note that the parameter defining the maximum number of rows retrievable via OpenSearch is -DMAX.ROWS.SEARCH.VALUE of the start.sh).
- **Deletion cause:** AJS GUI configuration file (appconfig.json) allows the configuration of deletion causes with the following parameter (in blue):

```
"settings": {

...,
```

```
"deleted_product": { "show_deletion_cause": true, "causes": ["Invalid: bad geometry","Invalid: reprocessed"], "default_cause": "Invalid" },
...
}
```

Here follows the explanation about the mentioned line:

- deleted_product.show_deletion_cause (true|false): if true the prompt message to insert deletion cause is shown, if false the prompt message is not shown
- deleted_product.causes: is the array containing available deletion causes.
- deleted_product.default_cause: contains the default cause used when end user inserts a deletion cause not matching with one of the available deletion causes

- **Map toolbar:** A toolbar for geographical filters is visible in the lower-left side of the AJS UI. The toolbar contains:
  - a pan button, to enable pan
  - a box button, to enable bounding box area selection (selected as default, as in the current UI)
  - a polygon button, to enable polygon drawing for area selection
  - a clear button, to clear a geographical selection

  It's possible to hide the toolbar changing the following value of the appconfig.json: `"show_map_toolbar": false`

A.8 OWC

The files in charge of the OWC configuration management are located in:

[DHUSDIR?]/var/tomcat/webapps/new/conf

These files are:

- **appconfig.json**

In the following paragraphs the configurable parameters of these files are described.

appconfig.json

The following table summarize the configurable parameters of the appconfig.json file.

Parameter & Description	Synthax
Debug mode	`"debug": false,`
Show Auxiliary data file feature	`"adf": false,`
Calendar Setting	`"calendar_refresh_time": 360,`

	`"show_calendar": true,`
Max number of instances per panel pushed in the the navigation manager (main layout).	`"max_instances_per_panel": 3,`
Show signup page	`"show_signup": true,`
Term and conditions page link	`"t_and_c_link": "https://scihub.copernicus.eu/twiki/do/view/SciHubWebPortal/TermsConditions",`

Show the link to open the legacy user interface (AJS client)	`"show_oldgui_link": true,`
	`"oldgui_link": "../",`
	`"oldgui_label": "go to classic view",`

Default map layer
(See elow)
	`"default_layer": {`
	`"Road": {`
	`"sources": [{`
	`"class": "OSM",`
	`"params": {`
	`"url" : "http://tile2.opencyclemap.org/transport/{z}/{x}/{y}.png",`
	`"wrapX": false`
	`}`
	`}],`
	`"title": "Road",`
	`"type": "base",`
	`"visible": ":visible"`
	`}`
	`},`

Logos to show in the help page	`"dashboard logos": ["./images/serco.png", "./images/gael.png", "./images/copernicuslogo.png", "./images/logo esa.png", "./images/european_union_emblem.png"],` `"timeline": {`

	<pre>"styles": {
	"S1": "#dc143c",
	"S2": "#0f9d58",
	"S3": "#0000ff"
	}
	},</pre>

Size of list page	<pre>"page_size":25</pre>

Here below we describe additional features:

- **Flex layout templates (prototype):**

```
"component_layout": [{
        "name": "empty_layout",
        "icon": "./images/grey-layout/01_layout.png",
        "preview": "./images/black-layout/01_layout.png",
        "component": "empty-template"
    },{
        "name": "twocolumns_layout",
        "icon": "./images/grey-layout/02_layout.png",
        "preview": "./images/black-layout/02_layout.png",
        "component": "twocolumns-template"
    },{
        "name": "tworows_layout",
        "icon": "./images/grey-layout/03_layout.png",
        "preview": "./images/black-layout/03_layout.png",
        "component": "tworows-template"
    },{
        "name": "square_layout",
        "icon": "./images/grey-layout/04_layout.png",
        "preview": "./images/black-layout/04_layout.png",
        "component": "square-template"
    },{
```

```
            "name": "trielements_layout",
            "icon": "./images/grey-layout/05_layout.png",
            "preview": "./images/black-layout/05_layout.png",
            "component": "trielements-template"
        },{
            "name": "trielements2_layout",
            "icon": "./images/grey-layout/06_layout.png",
            "preview": "./images/black-layout/06_layout.png",
            "component": "trielements2-template"
        }],
```

- **WMS - Map servers configuration in the Client:** the configuration of the map servers has been added in the new section "wms_server_config", that contains some parameters about the map server URL/credentials and the mapping rule between products and mapservers.

  Currently the mapping rule is based on regular expression in order to find any possible pattern based on the product name.

  An example of configuration could be:

```
        "wms_server_config": [{
            "name": "Serco WMS",
            "url": "http://149.202.246.167/instance00/ows",
            "username": "",
            "password": "",
            "rule":"S3",
            "priority":"1"
        }],
```

  This component allows the configuration of a list of WMS server, specifying the following parameters:

  - server name, i.e. server identifier
  - url, i.e. wms request url
  - username, i.e. username of the account enabled to perform WMS request (if needed)
  - password, i.e. password of the account enabled to perform WMS request (if needed)
  - rule, i.e. a regular expression on product identifier used to perform WMS request only for products matching the rule. This parameter is configurable by means of free text regular expression or simplified regular expression.

NOTE: the parameter "priority", so far, is not implemented (on 0.13.4)

Warning: the current implementation is a proof of concept, with the purpose of testing feature feasibility, so there is not yet a full configurability of this feature, since only a sample WMS with S3 OLCI layers is supported.

To deactivate the WMS interface delete the WMS block mentioned above.

- **Metadata plugin definition (prototype):**

```
    "processor_plugins": [{
        "name": "Products size average",
        "visualizer": "average-visualizer",
        "webworkers": ["processor-plugins/001.js"]
    }],
```

Where:
- name: name of the plugin to show in the plugin selector
- visualizer: the web component to show the outcomes of the plugin
- webworkers: array with the url where are located the web workers to call for the specific plugin.

- **ADF feature (optional):** Auxiliary Data File (ADF) Search is a new feature provided by the DHuS and available in OWC Application. Auxiliary Data File Search fills the need of using the DHuS not only to disseminate products but also to inform the user on the availability of new information, such as new Auxiliary Data Files (ADFs), which can be used to re-process products with a better quality.

  ADF feature is disabled by default. In order to enable it, setting `adf` property to true, as reported in the example below.

```
{

    "debug": false,

    "adf": true,

...

...

}
```

The configuration files for ADF component are present in two subfolders (rules, task_tables) contained in the etc folder of DHuS installation directory:

- ./rules/OL_1_EO_SelectionRules.properties
- ./rules/OL_1_RAC_SelectionRules.properties
- ./rules/OL_1_SPC_SelectionRules.properties
- ./rules/OL_2_SelectionRules.properties
- ./rules/SL_1_SelectionRules.properties
- ./rules/SL_2_SelectionRules.properties

- ./rules/SR_1_CAL_SelectionRules.properties
- ./rules/SR_1_MEAS_SelectionRules.properties
- ./rules/SR_2_SelectionRules.properties
- ./rules/SY_1_2_SelectionRules.properties
- ./rules/SY_2_VG_SelectionRules.properties
- ./task_tables/OL_1_EO_TaskTable.xml
- ./task_tables/OL_1_RAC_TaskTable.xml
- ./task_tables/OL_1_SPC_TaskTable.xml
- ./task_tables/OL_2_TaskTable.xml
- ./task_tables/SL_1_TaskTable.xml
- ./task_tables/SL_2_TaskTable.xml
- ./task_tables/SR_1_CAL_TaskTable.xml
- ./task_tables/SR_1_MEAS_TaskTable.xml
- ./task_tables/SR_2_TaskTable.xml
- ./task_tables/SY_1_2_TaskTable.xml
- ./task_tables/SY_2_VG_TaskTable.xml

styles.json

This configuration file allow configuration of product footprint colour according to the product instrument.

[Groups]/ Parameter	Description	Synthax

matches	Product Instrument	"matches": [
		{
		"name": "Instrument",
		"value": "SAR-C"
		}
		],

Styles/fill	RGB colour for footprint fill	"styles": {
		"fill": {
		"color": [
		179,

		27, 27, 0.1] },

Styles/stroke	RGB colour and width for footprint borders	"stroke": {
		"color": [
		179,
		27,
		27,
		1
		],
		"width": 1.5
		}

menu.json

The file menu.json contains the main menu sections definition.

Parameter & Description	Synthax
Array containing the sections of the main menu.	"sections":[…]

Every section is an object with the following parameters:

Parameter & Description	Synthax
The title of the section	"title":"section title"
The web-component to visualize when the user clicks on the button associated with this section in the main menu	"component":"login-component"
The icon to show for the section	"icon":"search"

Width of the panel that contains this component	"width":"150px"

theme.json

The file theme.json contains the global appearance customization for the owc application.

Parameter & Description	Synthax
Title of the application	"title":"title"
Url of the logo image	"logourl":"http://www.example.it/image.png"
Application panels background color	"bgcolor":"#e9f4ff"
Background image url	"bgimageurl":"image.png"

Appendix B. OpenStack storage command line samples

In this section, we report some OpenStack storage command line samples, using curl & jq commands:

Get token and swift url for specific user / project (tenant)

```
curl -s -d '{"auth": {"tenantName": "<tenantName>",
"passwordCredentials":{"username": "<username>", "password": "<password>"}}}' -H
"Content-type: application/json"
https://keystone9915.openstack.ovh.net:35358/v2.0/tokens | jq '{url:
.access.serviceCatalog[] | select(.name=="swift") | .endpoints[].publicURL, token:
.access.token.id}'
```

- List containers in the account

```
curl -i -H "X-Auth-Token: <token>" <url> -X GET
```

- Create container

```
curl -i -H "X-Auth-Token: <token>" <url>/<container_name> -X PUT
```

- Info about container content

```
curl -i -H "X-Auth-Token: <token>" <url>/<container_name> -X HEAD
```

- Delete container

```
curl -i -H "X-Auth-Token: <token>" <url>/<container_name> -X DELETE
```

Please note that a container can be deleted only if it is empty.

- Info about object in a container

```
curl -H "X-Auth-Token: <token>" <url>/<container_name>/<object_name> --head
```

- Delete object

```
curl -i -H "X-Auth-Token: <token>" <url>/<container_name>/<object_name> -X DELETE
```

<END OD DOCUMENT>