ESA Sen4Stat

Sentinels for Agricultural Statistics

D10.1 – SUM

“Software User Manual”

Issue/Rev: 2.0

Date Issued: 13/01/2026

Milestone: 5

Submitted to European Space Agency

Consortium Partners

Participant Organisation Name	Acronym	City, Country
Université catholique de Louvain	UCLouvain	Louvain-la-Neuve, Belgium
CS GROUP - ROMANIA	CS RO	Craiova, Romania
Systèmes d'Information à Référence Spatiale - Filiale CSL	SIRS-CLS	Villeneuve d'Ascq, France
Universidad Polytecnica de Madrid	UPM	Madrid, Spain

Contact

Université catholique de Louvain – Earth and Life Institute

Place de l'Université, 1 – B-1348 Louvain-la-Neuve – Belgium

Email: Sophie.Bontemps@uclouvain.be

Internet: https://uclouvain.be/en/research-institutes/eli/elie

Disclaimer

Document sheet

Authors and Distribution

Authors	Sophie BONTEMPS, Pierre HOUDMONT, Laurentiu NICOLA, Boris NÖRGAARD, Cosmin UDROIU
Distribution	ESA - Zoltan SZANTOI

Document Status

Issue/Rev.	Date	Reason
0.1	18/08/2021	Initial version of the deliverable
1.0	28/08/2021	1st version related to the Acceptance Review
1.1	10/01/2023	Update of the 1st version, describing the Sen4Stat system released to the beta users (Milestone 5)
1.2	20/02/2024	Second update of the 1st version, describing the Sen4Stat system 2.0
1.3	12/12/2024	Third update of the 1st version, describing the Sen4Stat system 3.0
2.0	13/01/2026	Second main update of the deliverable, describing the Sen4Stat system 4.0

Detailed changes between version

From version 1.2 to version 1.3	
Section	**Changes**
2.2.6	Updates for Yield processor
3	Updates to reflect changes for running on Alma and Rocky Linux
4	Updates for the changes in ERA5 CDS configuration and captures from the updated web interface
5	Updates for the processors' configuration
From version 1.3 to version 2.0	
Section	**Changes**
2.2.4	New section to reflect the new processor about in-situ data preparation
2.2.6	Description of the new yield processor included

3.1	Requirements updated
4.2.3	Updates of the “sites” and “seasons” definition and of the external data to be uploaded
4.2.6	Updated
4.4.2	Manual operations for ERA5 Download better explained and updated
4.4.4	New section to reflect the new processor about in-situ data preparation
4.4.6	Important note added
4.4.7	Addition of the CatBoost classifier and update of the associated classification parameterse
4.4.9 & 4.4.10	New sections to reflect the new yield processor included

Table of contents

List of figures

List of tables

1 Introduction

1.1 Purpose and scope

This document is the Software User Manual (SUM) document of the Sentinels for Agriculture Statistics (Sen4Stat) project funded by the European Space Agency (ESA).

The overall objective for the Sen4Stat project is to facilitate the uptake of Earth Observation (EO) information in the National Statistical Offices (NSO) supporting the agricultural statistics. Special attention shall be given to develop and demonstrate EO products and best practices for agriculture monitoring relevant for Sustainable Development Goals (SDG) reporting and monitoring their progress at national scale.

The SUM is one output of the Task 4 (WP 4000) of the Sen4Stat project, named "System Development and documentation". The aim of this document is to detail the operational system implementation, build, installation and maintenance procedure.

1.2 Structure of the document

After this introduction, this document contains **4** main sections and **7 appendices**:

- Section **2** provides a general overview of the system: its main objectives and properties, its processors and its architecture;
- Section **3** describes the deployment procedures of the system, both for the automatic and manual modes;
- Section **4** explains how to use the system, detailing the first steps to allow an automatic use but also the customization of the jobs for the manual operation;
- Section **5** defines the maintenance and troubleshooting procedures.

1.3 References

1.3.1 Applicable documents

Table 1-1. Applicable documents

ID	Title	Reference	Issue/Rev.	Date
AD.1	Statement of Work for ESA Sentinels for Agricultural Statistics	EOEP-EOPS-SW-17-015	1.0	15/03/2017
AD.2	Sen4Stat Technical Proposal - Chapter 3		1.0	12/05/2017
AD.3	Sen4Stat User Requirement Document	Sen4Stat_URD	1.2	26/11/2020
AD.4	Sen4Stat Technical Specification document	Sen4Stat_TS	1.1	26/11/2020
AD.5	Sen4Stat Software Development Plan	Sen4Stat_SDP	1.0	30/11/2020
AD.6	Sen4Stat Design Justification File	Sen4Stat_DJF	1.0	30/11/2020
AD.7	Sen4Stat Design Definition File	Sen4Stat_DDF	1.0	30/11/2020

1.3.2 Acronyms and abbreviations

Table 1-2. List of acronyms and abbreviations

Acronym	Definition
AD	Applicable Document
ATBD	Algorithm Theoretical Basis Document
AWS	Amazon Web Service
CNES	Centre National d'Etudes Spatiales
DDF	Design Definition File
DIAS	Copernicus Data and Information Access Services
DRMAA	Distributed Resource Management Application Api
EO	Earth Observation
ESA	European Space Agency
IPC	Inter-Process Communication
L1, L2, L3, L4	Level 1, Level 2, Level 3, Level 4
L8	Landsat 8
LPIS	Land Parcel Identification System
MACCS	Multi-Mission Atmospheric Correction and Cloud Screening
MAJA	MACCS-ATCOR Joint Algorithm
NSO	National Statistical Office
OGC	Open Geospatial Consortium
PEPS	Plateforme d'Exploitation des Produits Sentinel
REST	REpresentational State Transfer
ROI	Region of Interest
S1, S2	Sentinel-1, Sentinel-2
CDAS	Copernicus Data Space Ecosystem
SDG	Sustainable Development Goal
Sen2-Agri	Sentinel-2 for Agriculture
Sen4Stat	Sentinels for Common Agricultural Policy
Sen4Stat	Sentinels for Agricultural Statistics
SLC	Single-Look Complex
UI	User Interface
USGS	United States Geological Survey

2 System Overview

2.1 General overview

The Sen4Stat system consists of an open source EO processing system linked with (i) a module for in-situ datasets quality control, (ii) a visualization tool and (iii) a set of tools for higher-level statistical analyses. Being open source, it allows any user to generate, at his own premises and in an operational way, products tailored to his needs.

The Sen4Stat EO processing system is a standalone operational processing chain which generates a set of agriculture monitoring products for facilitating the uptake of EO information by the NSO. It relies on Sentinel-2 (S2) L1C and/or L2A, Sentinel-1 (S1) Single Look Complex (SLC) and Landsat 8 (L8) L1T time series to generate agriculture monitoring products and support the agricultural statistics estimation. These agriculture monitoring products are:

- EO-derived pre-processed reflectance / backscatter / coherence time series;
- EO-derived spectral indices and biophysical indicators, e.g. NDVI or LAI;
- Cloud-free colour composites;
- EO-derived crop maps (cropland – non-cropland, annual vs permanent cropland, crop type);
- EO-derived crop growth condition metrics;
- EO-derived crop yield estimates at administrative level.

The logical data flow and the main interfaces of the Sen4Stat EO operational system is provided in Figure 2-1. The system is composed of a set of independent processing modules orchestrated by a data-driven approach. These modules are based on a set of tools which can be re-used outside of the entire Sen4Stat system.

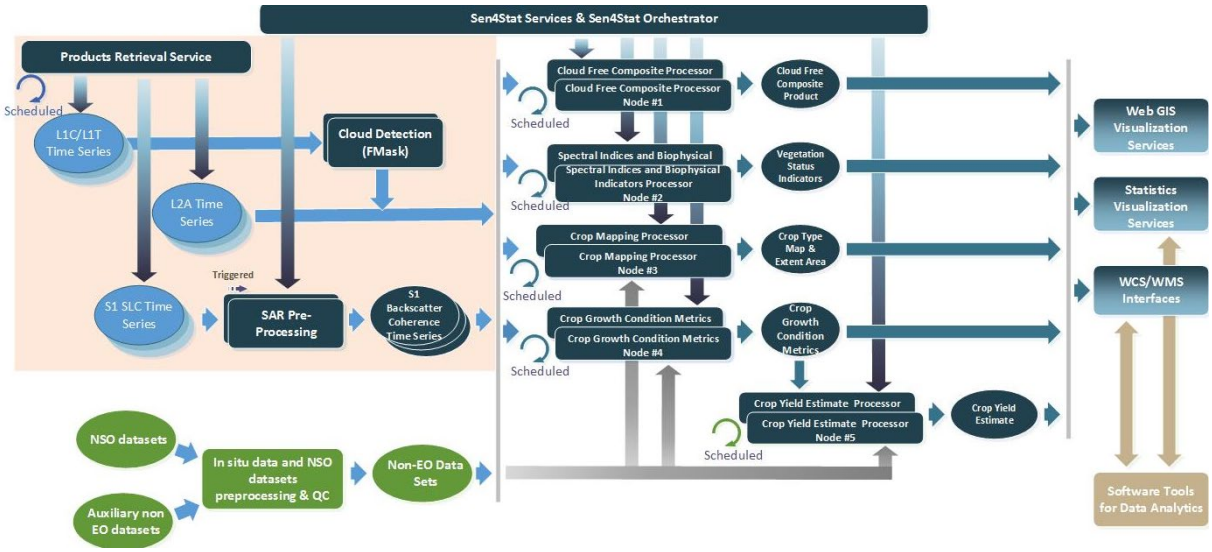


Figure 2-1. Logical data flow of the Sen4Stat EO processing system and its link with the external tools and modules

The main and mandatory inputs of the operational system are:

- A time series of S2 L2A products is automatically downloaded from ESA facility or imported directly into the system if they are available locally as in the case of the Copernicus Data and

Information Access Services (DIAS-es). S1 SLC is also considered as a main data source (and can be automatically downloaded or imported in the system) but is not mandatory;

- Mandatory user parameters: Area of Interest (AOI) defined as a shapefile, definition of the monitoring period, a set of parameters associated with the algorithms (see section 4);
- In-situ statistical and geographical dataset for each country along with the information about the regions, provinces municipalities and segments.

Additional inputs can be provided by the user if available and/or if found relevant. This is the case for the L8 L1T products but also for other products and parameters that will be detailed in section 4.

A Graphical User Interface (GUI) is available to define and handle all the requested parameters. This interface also controls the start of the system, indicates the main directories or files used by the system and allows users visualizing and downloading output products.

The Sen4Stat EO system is designed to be modular and interactive. It includes two main components:

- A set of Sen4Stat Software Components (SC): each Sen4Stat SC is an independent executable that represents an algorithm or a set of algorithms;
- A Sen4Stat Orchestrator: the Orchestrator is the main component used to manage the above Sen4Stat SC on the system. Its role is to monitor the occurrence of new files in the processing area file system, determine the processing chain to be launched, handle user requests and monitor the execution of the current processing chains.

The Sen4Stat system can be run in an ***automated mode***, which means it can be used in operational scenarios with as little as possible operator intervention. It is based on the Orchestrator and it ensures that the system automatically downloads data, processes them until the end of the season and delivers in time the output products. In this functioning mode, the processors can also be executed on user request, by using the public website functionality, which is part of the platform. Some Sen4Stat processors can also be run in a ***manual mode***, meaning that the processors can be invoked manually assuming that the required input data exist.

The Sen4Stat system website can be reached using a web browser and accessing http://[machine_name_or_ip]:8080/ui/login.html or in case SSL is configured https://[machine_name_or_ip]:8443/ui/login.html where [machine_name_or_ip] is the server name or pc. If another port name or port forwarding is used, the port should be updated accordingly.

The modular design and standardization of the interfaces make the Sen4Stat architecture targeted at various user profiles:

- **High-level analysts**, who want to integrate the products into their statistical calculations and are interested in a direct delivery of the products generated by the Sen4Stat system;
- **Operator level users**, who aim at running the system and generating the Sen4Stat products in their own facilities to control and validate the results, then disseminate them to their own high-level users;
- **Research users**, who would rather access and tune the components available, to generate new or adjusted products and possibly operate them as “classical” added-value producers.

The system can either be built from sources, available on a git repository, or installed from precompiled binaries packaged into Redhat Package Manager (RPM) binaries. “Appendix D - Building from source” presents the steps to be followed in order to build the RPM binaries. Section 3.3.1 describes the installation and configuration of the obtained RPM binaries (either from sources or from a distribution package). If any of the component binaries are available or installed, the corresponding step will be skipped.

2.2 Processors

The Sen4Stat system is made of several processors, which take care of the EO data pre-processing and which transform the pre-processed time series into relevant agriculture products. The following sub-sections briefly describe each of them.

2.2.1 EO Data Pre-Processing processors (SAR and optical)

The "EO Data Pre-Processing" processor carries out the pre-processing for all EO data supported by the Sen4Stat system: Sentinel-2, Sentinel-1 and Landsat 8.

For Sentinel-2, users can choose to use directly the Sen2Cor Level 2A (L2A) images automatically produced by the European Space Agency (ESA) and available on the ESA Copernicus Data Space Ecosystem (Copernicus DAS) and on most of the cloud providers. In this case, the pre-processing processor offers the option to generate an additional cloud mask using FMask.

Alternatively, users can decide to work with L1C products. In this case, the processor applies both atmospheric correction and cloud mask algorithms. This is also the case if users want to work with L8 L1T.

For the Synthetic Aperture Radar (SAR) sensors, the processor will transform the Level 1 (L1) products into backscatter and coherence products.

Figure 2-2 presents the general workflow of the EO data pre-processing processor.

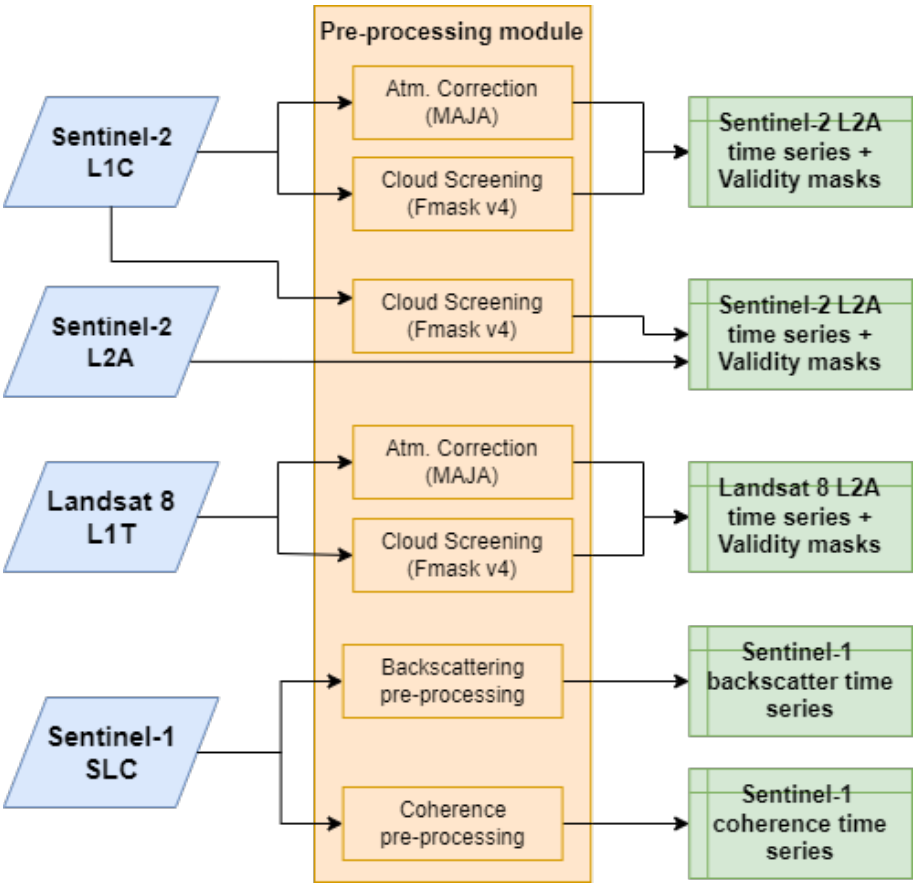


Figure 2-2. Workflow of the EO Data Pre-Processing

2.2.2 Spectral Indices & Biophysical Indicators processor

The Spectral Indices & Biophysical Indicators processor provides three Spectral Indices and three Biophysical Indicators informing about the evolution of the green vegetation:

- The ***Normalized Difference Vegetation Index (NDVI),*** the most popular indicator operationally used for vegetation monitoring, provided to ensure continuity with existing long-term time series and thus, allowing anomalies detection;
- The ***Normalized Difference Water Index (NDWI)***, introduced for the first time in 1996 and reflecting moisture content in plants and soil;
- The ***Brightness***, defined as the Euclidean norm of the surface reflectance values in green, red, NIR and SWIR;
- The ***Leaf Area Index*** (***LAI)***, an intrinsic canopy primary variable that should not depend on observation conditions, which determines the size of leaf interface for exchange of energy and mass between the canopy and the atmosphere;
- The ***fraction of Vegetation Cover (FCOVER)***, corresponding to the fraction of ground covered by green vegetation. It quantifies the spatial extent of the vegetation;
- The ***fraction of Absorbed Photosynthetically Active Radiation (FAPAR)*** by the green and alive elements of the canopy. The FAPAR depends on the canopy structure, vegetation element optical properties, atmospheric conditions and angular configuration.

Complete specifications of all products generated by the spectral indices and biophysical indicators processor are provided in Table 2-1.

Table 2-1. Vegetation status indicators specifications

Properties	Value
Available spectral indices and biophysical indicators	NDVI, NDWI, Brightness, LAI, FAPAR, FCOVER
Spatial extent	Global (not only crop area)
Spatial resolution	10 m
Temporal resolution	For each S2 (L8) acquisition
Geometric accuracy	Same than L2A input data
Format	GeoTIFF raster images
Projection	UTM / WGS84
Metadata	XML file

The products are delivered with several masks that will help appraising its quality:

- the status of the input L2A pixels (no data / cloud / snow / water / land);
- the definition domain of the input spectral bands and of the output variables.

The main and mandatory input of this processor is a S2 L2A time series, optionally completed by an additional L8 time series.

The NDVI is computed using a standard formulation applied to the S2 red (B4) and narrow Near InfraRed (NIR) (B8a) bands. The NDWI also relies on a standard formulation applied to the S2 narrow

NIR (B8a) and Short-Wave InfraRed) (B11) bands. As already mentioned, the Brightness computation makes uses of the S2 bands in the green (B3), red (B4), narrow NIR (B8a) and SWIR (B11).

The LAI retrieval is performed from the bands 3, 4, 5, 6, 7, 8, 9, 12, 13 using machine learning to build a non-linear regression model. For the LAI, FCOVER and FAPAR, the implementation is derived from the one already proposed in the frame of the ESA Sentinel-2 toolbox. The LAI, FCOVER and FAPAR retrieval is performed by applying a global Artificial Neural Network (ANN) on each pixel considering the reflectance values of all the available bands pre-processed at the L2A and some geometric configuration as input. The training of the ANN, which consists in generating the training database, defining the neural network architecture and calibrating the network, is not performed within the Sen4Stat system. Instead, the Sen4Stat system benefits from an already trained ANN, made openly and freely available by the Institut National de la Recherche Agronomique (INRA) which developed the algorithm. From the system implementation point of view, this trained ANN is given as auxiliary data to the processor.

Figure 2-3 presents the general workflow of the Spectral Indices & Biophysical Indicators product algorithm.

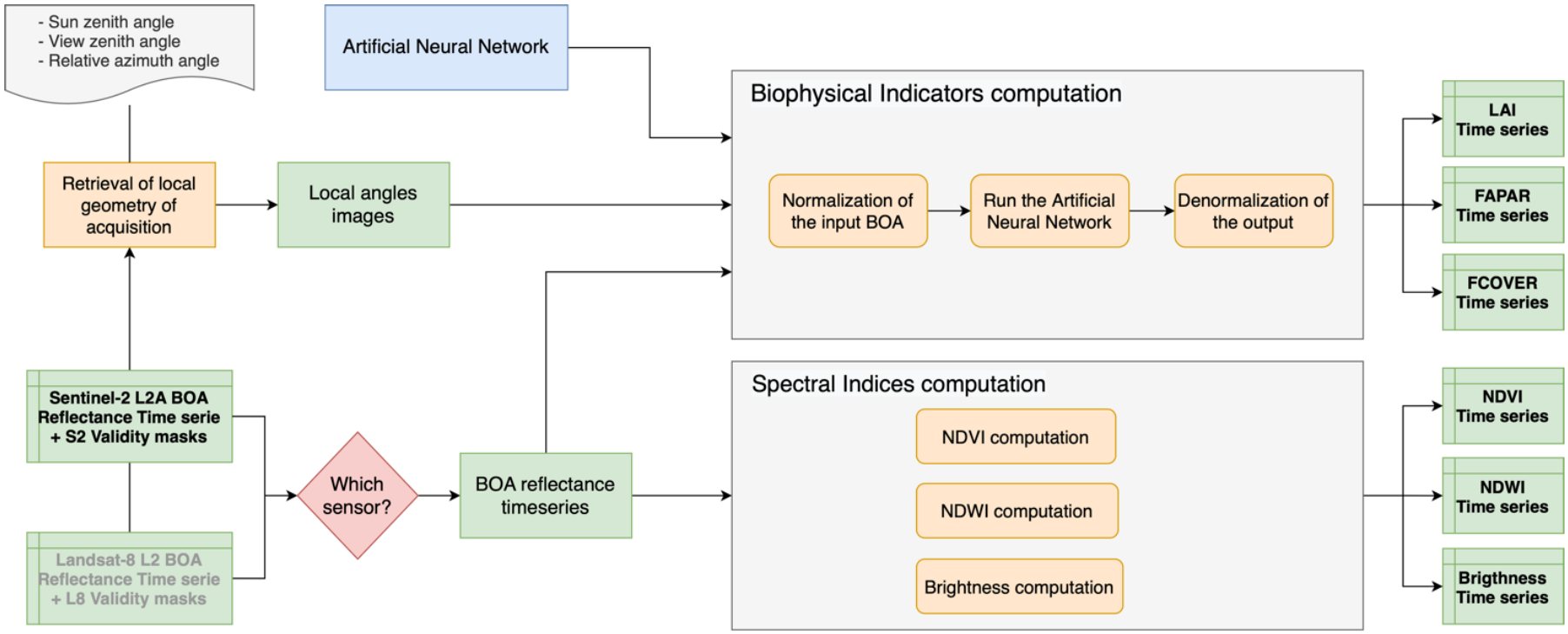


Figure 2-3. General workflow of the L3A Spectral Indices & Biophysical Indicators product algorithm

2.2.3 Cloud-free temporal syntheses processor

The cloud-free temporal syntheses processor provides a cloud-free composite of surface reflectance values in the 10 S2 bands designed for land observation and keeping their native spatial resolution (10 or 20 meters). Complete specifications are provided in Table 2-2.

Table 2-2. Cloud-free composite specifications

Properties	Value
Available bands at 10m spatial resolution	Blue (B2), green (B3), red (B4), near infrared (B8)
Available bands at 20m spatial resolution	Vegetation red-edge (B5, B6, B7, B8a), shortwave infrared (B11, B12)
Temporal resolution	Flexible, depending on the AOI
Geometric accuracy	Same than L2A input data
Format	GeoTIFF raster images
Projection	UTM / WGS84
Metadata	XML file

The product is delivered with several masks that will help appraising its quality:

- the number of valid observations over the period used to generate the composite;
- the status of the pixel over the compositing period: no-data (pixel never observed during the compositing period), cloud (pixel always cloudy during the compositing period), snow (pixel always covered by snow for the available cloud-free observations), water (pixel always covered by water for the available cloud-free observations), land (pixel free from cloud / cloud shadow, snow and water at least once during the compositing period);
- the weighted average of dates used in the synthesis (which informs about the date within the compositing period from which the synthesis is representative).

The main and mandatory input of this processor is a S2 L2A time series including validity masks, optionally completed by an additional L8 time series. When Landsat-8 products are used, the composition is performed using Sentinel-2 as master and using the following association table for the S2/L8 bands (Table 2-3).

Table 2-3. Cloud-free composite S2/L8 bands mapping

S2 Band	L8 band	wavelength (nm)	S2 Resol. (m)	Provided in composite
1	1	450	60	No
2	2	490	10	Yes
3	3	560	10	Yes
4	4	670	10	Yes
5	-	705	20	Yes
6	-	740	20	Yes
7	-	780	20	Yes
8	-	820	10	Yes
8a	5	865	20	Yes

9	-	940	60	No
10	6	1340	60	No
11	7	1650	20	Yes
12	8	2200	20	Yes

The processor is based on the weighted average composite approach, includes the correction of directional effects to consider changes in observation angles and therefore in reflectance values among the different images that are stitched to create the product. It takes as input each L2A tile product available at time *t* but also the temporary product generated at the previous date *t-1*. This *t-1* product is updated iteratively with the new data until the compositing period is completed. The process is illustrated in Figure 2-4.

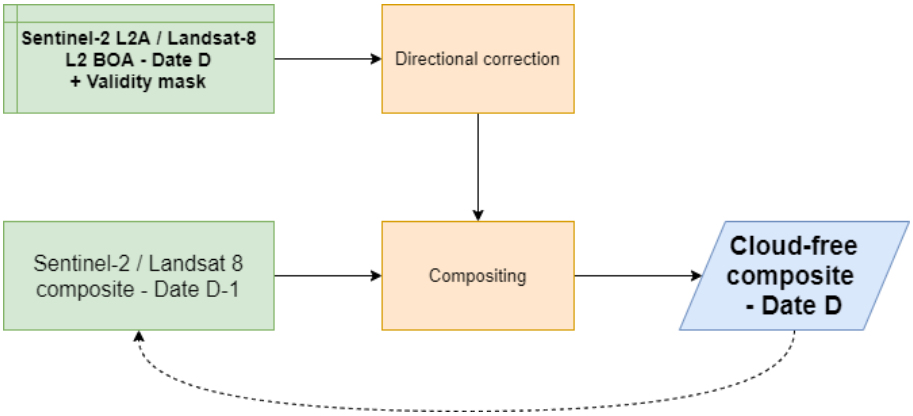


Figure 2-4. Logical data flow of the cloud-free temporal syntheses processor

2.2.4 In-situ Data Preparation processor

In-situ data (about crop type and crop yield) are mandatory to run the crop type mapping and the crop yield estimation processors. As the Sen4Stat system aims at facilitating the uptake of EO information by the NSO, it is assumed that in-situ data will come from agricultural surveys conducted by the NSOs to estimate their crop acreage and yield statistics. Nevertheless, the system can use any other source of in-situ data providing that they are in the good format (explained below).

In-situ data are quality-controlled and formatted before being used by the system to generate the EO products. This is the objective of the "In-situ Data Preparation" processor. This processor assumes that the in-situ data are provided **as polygons with a given set of attributes**. An overview of the processor is provided in the Figure 2-5, and a more detailed description of the in-situ preparation process can be found in the related ATBD.

This processor takes two files as input:

- A shapefile containing the geometries of the parcels and their identifier;
- A CSV file containing the attributes of the parcels (e.g., crop type, yield estimate, crop quality).

The processor aims at qualifying each polygon with a set of indicators or flags related to its geometry, area, quality and stratum. The analysis of the geometries allows to:

- Determine if the geometry is valid (i.e., the geometry is not empty nor overlapping itself);
- Determine if the geometry is unique;

- Determine if the geometry is composed of a multipart polygon;
- Identify polygons overlapping their neighbours.

The in-situ data preparation also includes the rasterization of the polygons, allowing to count the number of underlying pixels for each parcel.

Finally, a negative 10m buffer is applied to the geometries and they are reprojected into the WGS 84 / UTM zone coordinate systems that correspond to the Sentinel tiles underlying the parcels.

For technical reasons, the geometries must be in a projected CRS using meters as the length unit. Geometries in e.g. EPSG:4326 (WGS 84) will still be accepted but might not be imported correctly.

The output of the "*In-situ* Data Preparation" processor is a set of three standardized Sen4Stat PostGIS tables containing:

- The in-situ data corresponding to the parcels;
- The geometries of the parcels and their identifiers;
- The geometry flags and attributes computed by the processor and listed above.

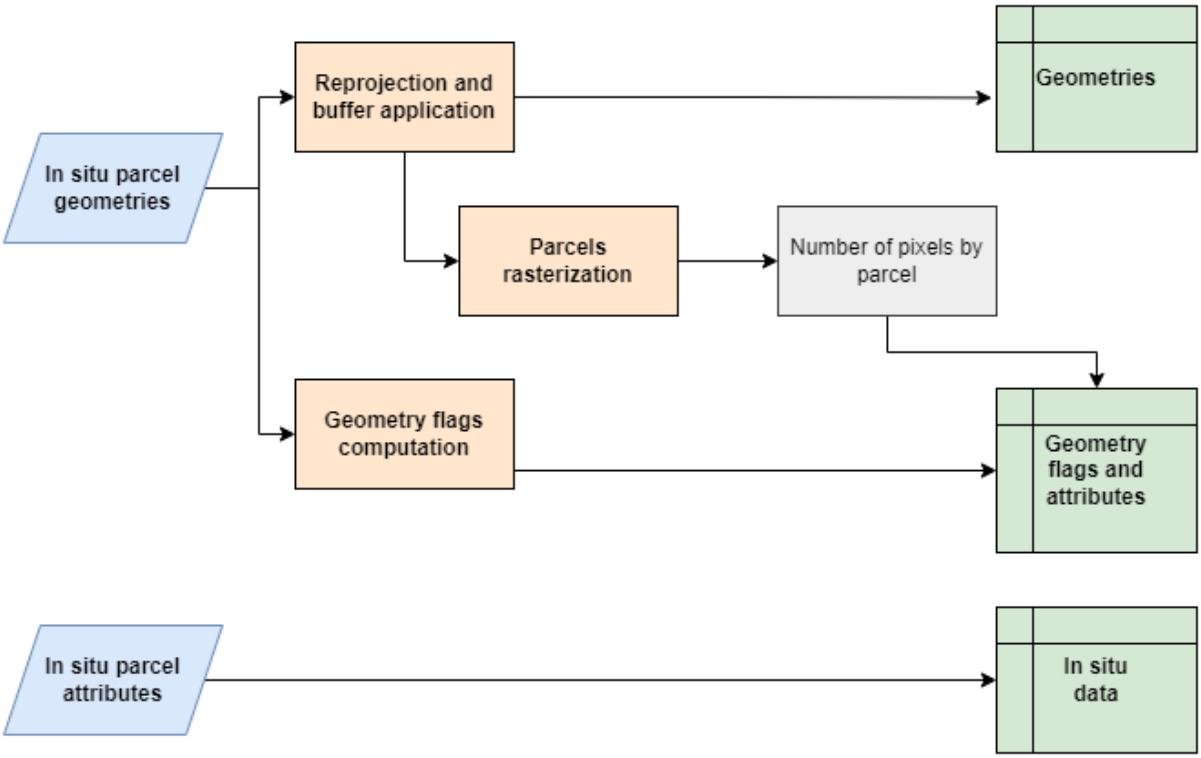


Figure 2-5. Workflow of the in-situ in data preparation processor

2.2.5 Crop Mapping processor

The crop mapping processor relies on per-pixel machine learning and deep learning algorithms to generate various crop maps.

Two types of data are used to feed classifier algorithms: in-situ data (pre-processed by the "In-situ Data Preparation" processor) and EO data.

Figure 2-6 presents the general workflow of the mapping products processing chain, organized into 5 main steps:

1. In-situ data sample selection
2. Optical EO data temporal resampling & gap filling
3. EO features computation & extraction
    a. Optical data
    b. SAR data
    c. Biophysical indicators
    d. Spectral indices
    e. Composite
4. Pixel-based classification
5. Validation

Three classifiers will be included in the "Crop Mapping" processor: Random Forest (OpenCV and Ranger implementations), Catboost and Broceliande.

A detailed crop type legend is used to train the classifiers and at the end, the detailed legend of the crop map can be simplified into different products: binary cropland - non cropland map, binary annual vs permanent crops, main crop type groups, detailed crop type map.

The "Crop Mapping" processor is designed to be efficient at national scale. As the amount of calibration data is limited and probably not distributed uniformly over the country, the processor offers the possibility to stratify the area of interest, i.e. to split it into multiple agro-climatic regions - called strata - which are homogeneous in terms of climate, agro-ecological conditions (relief, soil, etc.), cropping systems and agricultural practices. The use of such stratification allows reducing the natural variability existing when working at national scale by coping with agro-climatic gradients inducing a very diversity of crop calendars and growing conditions. Each stratum is classified independently, i.e. with his own set of calibration pixels and his own classification model.

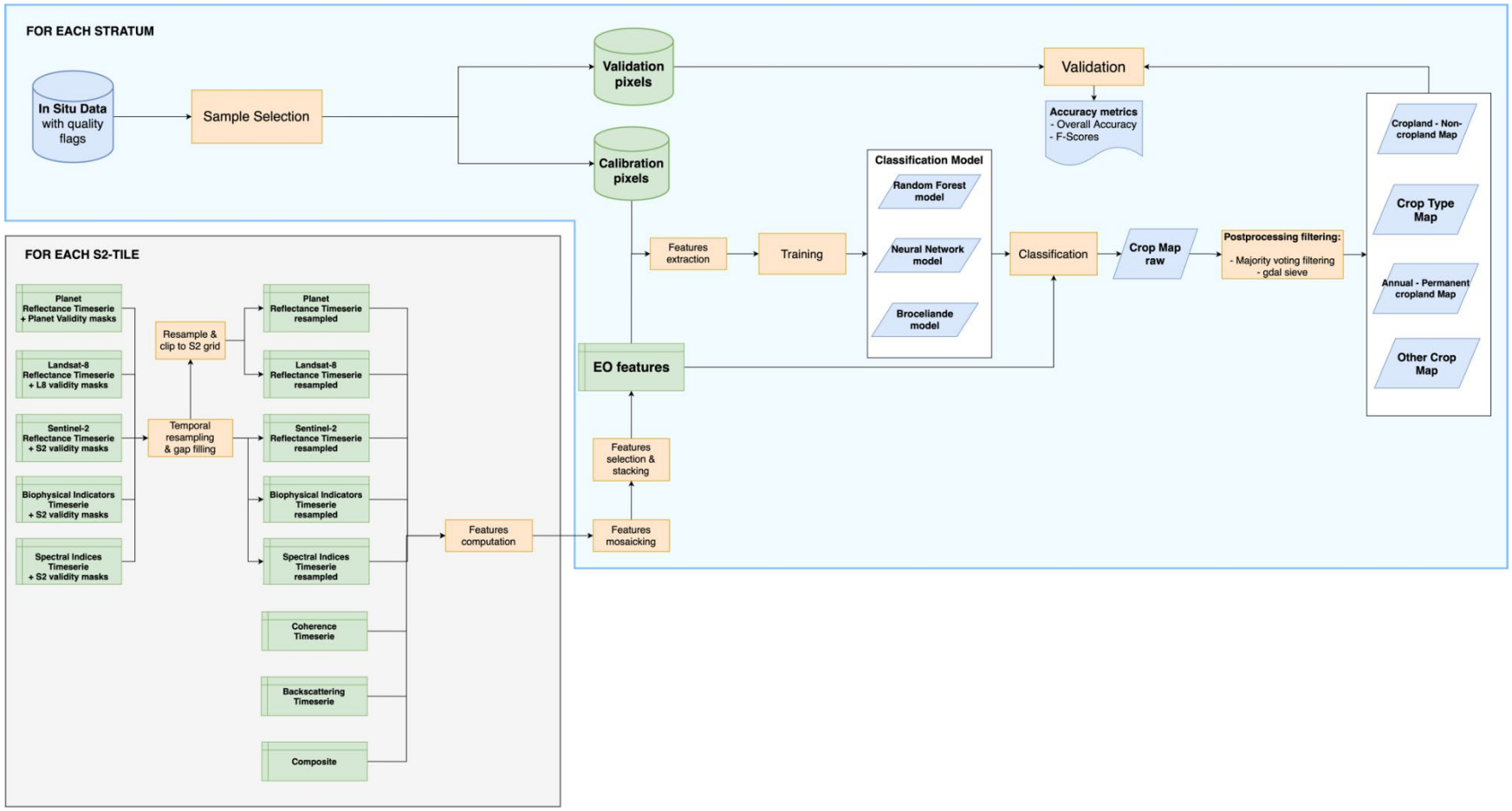


Figure 2-6. Workflow of the crop mapping algorithms

2.2.6 Crop growth condition metrics and yield estimation processor

The yield estimation approach implemented in Sen4Stat relies on two steps, which are implemented in two different processors:

1. Crop growth condition metrics to extract the yield features;
2. Crop yield estimation to design and apply the yield model.

2.2.6.1 Crop growth condition metrics (L4- yield feature)

This processor aims at extracting metrics that are representative of the crop growing and that will be further used as proxy variables of the yield in the training and validation of statistical models in the next processor. The processor can extract features at two different spatial levels:

1. when in-situ yield data are available at very high spatial resolution (e.g. parcel-level), features are extracted at that scale;
2. otherwise, the processor works at the scale of aggregated statistical units, such as provinces, districts, or regions, relying on historical data.

The main EO input is the LAI time series, which directly or indirectly influences most yield-related features. Climate variables extracted from the ERA5-Land database are also incorporated. When in-situ yield data are available, they need to be uploaded with the associated field boundaries. Otherwise, the processor rlies on a statistical unit (SU) shapefile, the corresponding historical yield data, and a crop type map (produced by the crop type processor). The different steps of the crop condition metrics processor are summarized in Figure 2-7.

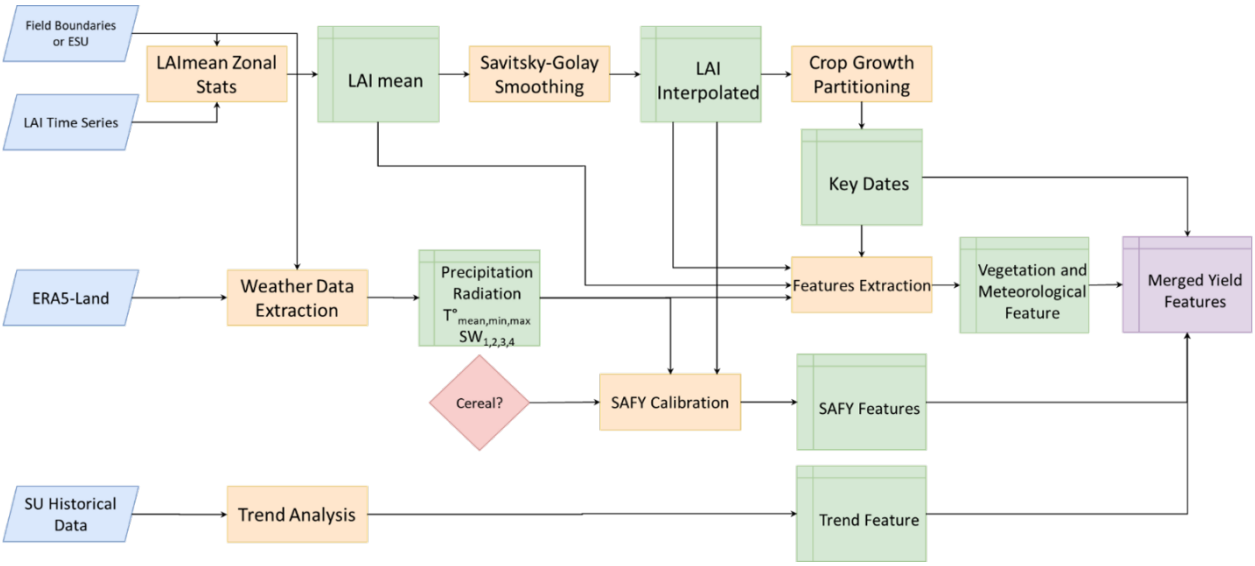


Figure 2-7. Workflow of the crop growth condition metrics extraction

2.2.6.2 Crop yield estimation (L4- yield)

The crop yield estimation processor is designed to train a regressor that links crop growth condition metrics to yield measurements collected by NSOs. The model requires two types of features: those used for training and those used for yield estimation.

When condition metrics are generated at high spatial resolution, an in-situ dataset of field-level measurements or observations associated with the training units is required.

When metrics are produced at the statistical unit (SU) level, historical yield data are required instead. At the SU level, the model should rely on multi-year observations; in other words, generating crop condition metrics over several years is a prerequisite for estimating yields in a given year. This requirement is not mandatory for field-level estimation, although it is recommended.

The regressor and the set of metrics can be selected either based on expert knowledge or automatically (default option). When SU boundaries are available, field-level yield estimates are aggregated to the SU level. The main steps of the *Yield* processor are summarized in Figure 2-8.

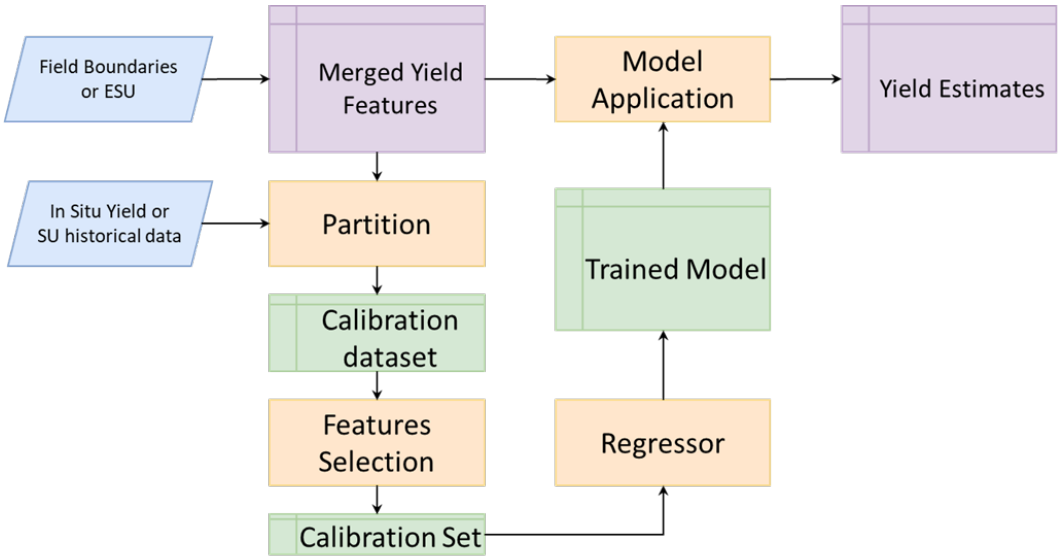


Figure 2-8. Workflow of the yield estimation processor

3 EO processing system deployment

3.1 Prerequisites for automatic usage

Before starting the installation of the Sen4Stat system for an automatic use, a set of preconditions is expected to be met in terms of system hardware, operating system or packages installed.

The **minimum hardware requirements** are:

- disk space for system installation: 120 GB;
- disk space for the resulted products (/mnt/archive - see section 3.3.1): this depends on the number of sites to be defined, the seasons and the number of tiles to be used for each site.
- disk space for the internal directory where the shapefiles of the created sites will be uploaded in order to be used by the system (/mnt/upload - see section 3.3.1): 20 GB are needed;
- RAM: between 64 and 128 GB, depending on the size of the country;
- number of CPUs: between 8 processors and 16 processors (or more), depending on the size of the country.

Rocky Linux or Alma Linux 64-bit are required by the Sen4Stat system.

The Linux user who will perform the installation should have the root rights (i.e. be part of the sudoers list), the installation process will perform a lot of `yum` installation

If a proxy server is required to access the Internet, it needs to be configured in /etc/dnf/dnf.conf, as described in the dnf.conf(5) manual page . For example, some lines with the ones below should be added in /etc/dnf/dnf.conf file:

```
# The proxy server - proxy server:port number

proxy=http://mycache.mydomain.com:3128

# The account details for yum connections

proxy_username=yum-user

proxy_password=qwerty
```

Although the Sen4Stat system is using the already existing Sentinel-2 L2A products from ESA or DIAS repository created with Sen2Cor, **MAJA package** is still needed for processing the Landsat 8 L1T products into L2A products and optionally, if explicitly configured, also Sentinel-2 L1C products (please refer to sections 3.3.1 and 4.1 to know how to configure and use MAJA on your system).

For downloading S1, S2 and L8 acquisitions from the distribution server, the system will need **2 user accounts** to be provided after the installation:

- An account and a password for the ESA Copernicus Data Space Ecosystem (Copernicus DAS), allowing downloading S1 and S2 products. This account can be obtained accessing https://dataspace.copernicus.eu/ and then accessing the “Login” and then “Register” section;

- An account and a password for the United States Geological Survey (USGS) portal, allowing downloading L8 products. The account can be created accessing the link https://ers.cr.usgs.gov/login/ and following the "Create new account" option.

  Nevertheless, in order to activate the downloads for your account the following additional operations need to be performed:

  - Login to your account on USGS
  - Go to https://ers.cr.usgs.gov/profile/access
  - Push the "Request Access" button and fill the information in the form that will be displayed:
    - "Datasets Required" – Landsat Collection 1 Level-1
    - "Data Products Required" - Landsat 8 OLI/TIRS C1 Level -1
    - "Scenes required" – In your case you can fill 2 monthly or 24 yearly (depends on your site dimension). You can also put more if you consider you will have additional sites to monitor
    - "Scripting Capabilities": python, java
    - For the other fields, you will have to fill the requested information

  After that, you should receive after several hours an email from USGS confirming the activation. Once this is done, the L8 should start also being downloaded.

IMPORTANT NOTE: The Crop Type Mapping processor does not support yet the Landsat 8 products and the Yield processor was not yet assessed with these products too.

Sub-section "4.2.1 Configure the data sources" describes how to configure these accounts in the system.

Later on, after the system is completely installed, a **.zip archive containing the shapefile delineating your site of interest** is required. This archive should include a ".shp", a ".prj", a ".dbf" and a ".shx" file. More information about the way this archive is used is given in the sub-section 4.2.2.

3.2 Installation package content

The installation package of the Sen4Stat system consists in the following main folders:

- **install_script** - contains the installation scripts that are used to create the distribution and to install the system;
- **rpm_binaries** - the RPM files for all other system components (SLURM, orchestrator, downloader, processors);
- **sen2agri-services** - the folder containing the Sen4Stat services to be installed.

Additionally, the following archives are provided beside the system installation package:

- **gipp maja**- the gipp files used by the L2A processor (MAJA) during the processing of the S2 L1C / L8 L1T products. Two versions of gipp files are available, corresponding to to the MAJA 3.1 and 3.2.2 versions and to the MAJA 4.5.4 version respectively. Please note that if another more recent MAJA version will be installed, these gipp files should be updated accordingly;

- **srtm.zip and swbd.zip -** these archives contain the Digital Elevation Model (DEM) files needed by the L2A processor (MAJA). Although they are part of the installation package, they are provided separately due to their size and because they will normally not change from a version to another. Before starting the installation of the system, these two files should be copied inside the installation package, in the "Sen4StatDistribution" directory.

3.3 Installation and configuration for automatic usage

3.3.1 Installation procedure

The following procedure allows installing the system for automatic usage. It is worth noting that the installation for automatic usage implies that all functionalities of the system are installed and made available, including those for running the manual mode.

Before of the installation, some default directories have to be created in the system either physically or mounted. The default directories are:

- /mnt/archive – this is the default working directory for the system;
- /mnt/upload – the folder where the files from the user are uploaded (for example shapefiles with in-situ data).

These directories can be changed after installation using the *"Sen4Stat Configurator*" application (see *Appendix E.1 Processing System Configuration Utility*).

The user sen2agri-service (which will be added by the installer, see below) should have full access rights (read/write/execution) over these default directories. In order to do this, launch the following command from a terminal, as a user which is in sudoers list:

```
sudo mkdir /mnt/archive

sudo mkdir /mnt/upload

sudo chmod -R a+wrx /mnt/archive /mnt/upload
```

The directories "/mnt/archive" and "/mnt/upload" can also be mount points to another external storage.

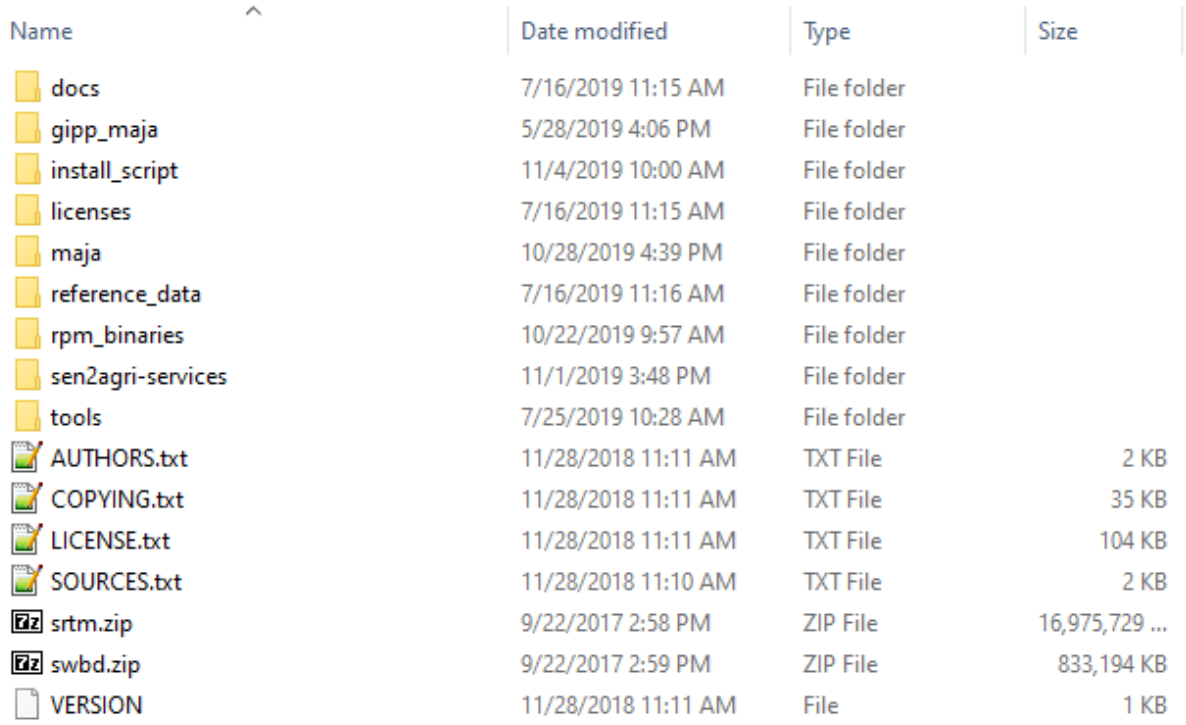
The users on a system and the root user are not recommended to be used as there cannot be known a priori. During the installation, the Sen4Stat installer will create the user *"sen2agri-service"*. This *"sen2agri-service"* user will be the user under which all the Sen4Stat services are executed. The *"sen2agri-service"* user will be created by the installer; he should have write access to these mount points. Since the *"sen2agri-service"* user might not be already created before the installation, these external directories should therefore have initially access rights for all users.

As explained in section 3.1, the SRTM and SWBD files are provided separately, because of their size and also because they will normally not change from one version to another and don't need to be re-downloaded at each update. The following options exist to have these datasets usable by the Sen4Stat system:

- Either, copy the srtm.zip and swbd.zip in the root folder of the installation package (in the "Sen4StatDistribution" directory). In this case, the installer will automatically unzip them and copy them into the right location (/mnt/archive);

- Or, unzip the files from the two archives in folders /mnt/archive/srtm and /mnt/archive/swbd respectively. In this case, the installer will detect their presence in the right location and will use them directly.

An example of a structure of the package, including the SRTMs and MAJA is presented in Figure 3-1.



Name	Date modified	Type	Size
docs	7/16/2019 11:15 AM	File folder	
gipp_maja	5/28/2019 4:06 PM	File folder	
install_script	11/4/2019 10:00 AM	File folder	
licenses	7/16/2019 11:15 AM	File folder	
maja	10/28/2019 4:39 PM	File folder	
reference_data	7/16/2019 11:16 AM	File folder	
rpm_binaries	10/22/2019 9:57 AM	File folder	
sen2agri-services	11/1/2019 3:48 PM	File folder	
tools	7/25/2019 10:28 AM	File folder	
AUTHORS.txt	11/28/2018 11:11 AM	TXT File	2 KB
COPYING.txt	11/28/2018 11:11 AM	TXT File	35 KB
LICENSE.txt	11/28/2018 11:11 AM	TXT File	104 KB
SOURCES.txt	11/28/2018 11:10 AM	TXT File	2 KB
srtm.zip	9/22/2017 2:58 PM	ZIP File	16,975,729 ...
swbd.zip	9/22/2017 2:59 PM	ZIP File	833,194 KB
VERSION	11/28/2018 11:11 AM	File	1 KB

Figure 3-1. Structure of the installation package (including the SRTMs and MAJA)

To install and configure the Sen4Stat system, with all its dependencies, it is enough to run the following script that can be found in the distribution package:

```
## open a terminal -- go into /install_script folder:

cd /path/to/Sen4XDistribution/install_script

## Run the install script

sudo ./install.sh
```

The install.sh script will look for the Linux version and will automatically decide the install script for your Linux. For example, running on Rocky Linux and Alma Linux the “install_rocky.sh” will be automatically invoked for you.

This script will automatically install the system (SLURM, orchestrator, processors, downloader, website, the database and all other dependencies). The installation is completely automatic, requiring minimum interaction from the user.

A PostgreSQL[1] database is used to keep track of the system information. The database is installed and configured by the installation script. The reference tables are already filled in with all the necessary details when the database is installed. For a list of reference tables and access for advanced users, the reader is referred to *Appendix B - Sen4Stat database tables for manual configuration*.

IMPORTANT NOTE: The installation scripts should have execution rights. If by some operations, these rights are lost, the following command could be optionally executed before starting the installation:

```
chmod -R a+x /path/to/Sen4XDistribution/
```

3.3.2 Update procedure

The Sen4Stat system contains also an update processor that will allow easily upgrading to newer versions, once these become available. To upgrade the system, a generic script that will update the sen4stat-services application will be use as below:

```
cd ~/Downloads/Sen4XDistribution/install_script

sudo ./update.sh
```

3.3.3 Additional configurations

Some additional steps might be needed to complete the installation and configuring the system. These steps are optional.

3.3.3.1 Configure your proxy for the Sen4Stat-Services downloaders

This step is optional.

If a proxy server is used for the internet connection, the files have to be manually edited by adding as root user the following configuration keys in the */usr/share/sen2agri/sen2agri-services/config/services.properties* file:

- *proxy.host*=<host of the proxy server>
- *proxy.port*=<port of the proxy server>
- *proxy.user*=<user for the proxy>
- *proxy.password*=<password of the proxy user>
- *proxy.type*=<Type of the proxy> that can be one of the following values:
    - *DIRECT* - Represents a direct connection, or the absence of a proxy;
    - *HTTP* - Represents proxy for high level protocols such as HTTP or FTP;

[1] PostgreSQL is a powerful, open source object-relational database system – see https://en.wikipedia.org/wiki/PostgreSQL

- *SOCKS* - Represents a SOCKS (V4 or V5) proxy.

IMPORTANT NOTE: After modifying the file services.properties, the sen4stat-services should be restarted using the command:

```
# sudo systemctl restart sen2agri-services
```

These values will be used for the first execution of the system, after a fresh installation, when nothing is configured in the system. The next configurations of the system need to be performed only via the IHM of the system that has a dedicated page for this, accessible from the "data sources" item of the main menu (see 4.2.1).

3.3.3.2 Configure your proxy for the web browser

This step is optional.

In order to allow the web browser having the map world interface functional, it can be needed to add your HTTP/HTTPS proxy information in the Sen4Stat terminal or directly into the Sen4Stat web browser.

From the terminal, you can use the following command lines with the right information:

```
export  http_proxy=http://${PROXY_USER}:${PROXY_PASSWORD}@${PROXY_HOSTNAME}:      ${PROXY_PORT}
export https_proxy=https://${PROXY_USER}:${PROXY_PASSWORD}@${PROXY_HOSTNAME}: ${PROXY_PORT}
```

From the Sen4Stat web browser, you should set the proxy information in our settings. For example, for Firefox, you should set in the Advanced section / Network / Settings the correct information.

3.3.3.3 Sen4Stat website users management

This step is optional.

By default, the website interface is provided with the following user having admin rights:

Username: *sen4stat*

Password: *sen4stat*

The website uses a user hierarchy to change the user parameters. There are two types of users: admin users and non-admin users. One might take advantage of this "non-admin users", when several sites are run in parallel on the same Sen4Stat system. In this case, a "non-admin user" can be assigned to each site. In this case, in the website interface, the non-admin user can see only information related to the assigned site.

To add users in the database, a special tab is available in the web interface (see section 3.3.4 to see how to connect). Using this tab, you can check the current information corresponding to a user and add a new one (Figure 3-2). You can also define the role of a specific user and, in the case of a regular user (not admin), limit the access only to certain sites.

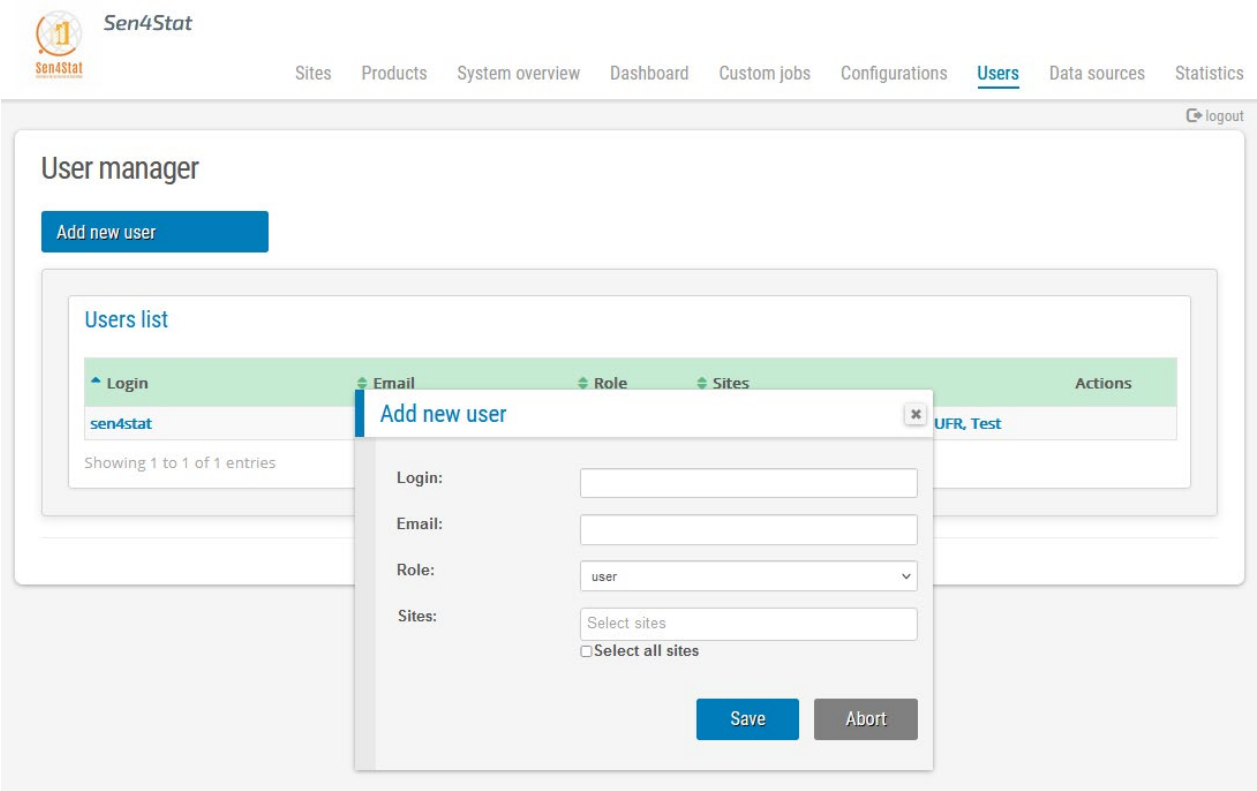


Figure 3-2. In the tab user of the web interface, “Add new user”

After a user is added, the corresponding password needs to be set during the first login, by selecting “Reset password” from the login page (Figure 3-3).

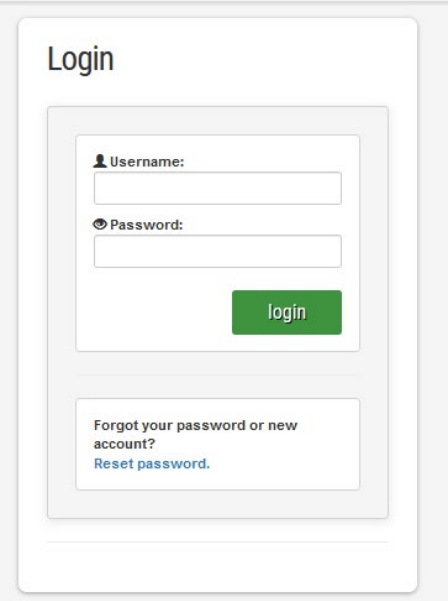


Figure 3-3. In the case of a first login: “Reset password” in the login page

3.3.4 First interaction with the Sen4Stat system

After the installation is finished and all the additional configurations are performed, the web interface of the system can be accessed from a web browser[2] if the user is connected to the same machine as the Sen4Stat system. The following address needs to be inserted: http://localhost:8080/ui/login.html or http://127.0.0.1:8080/ui/login.html. If SSL certificate or another port is used then the link should be updated accordingly for example, accessing https://localhost:8443/ui/login.html.

The following interface (Figure 3-4) will be displayed.

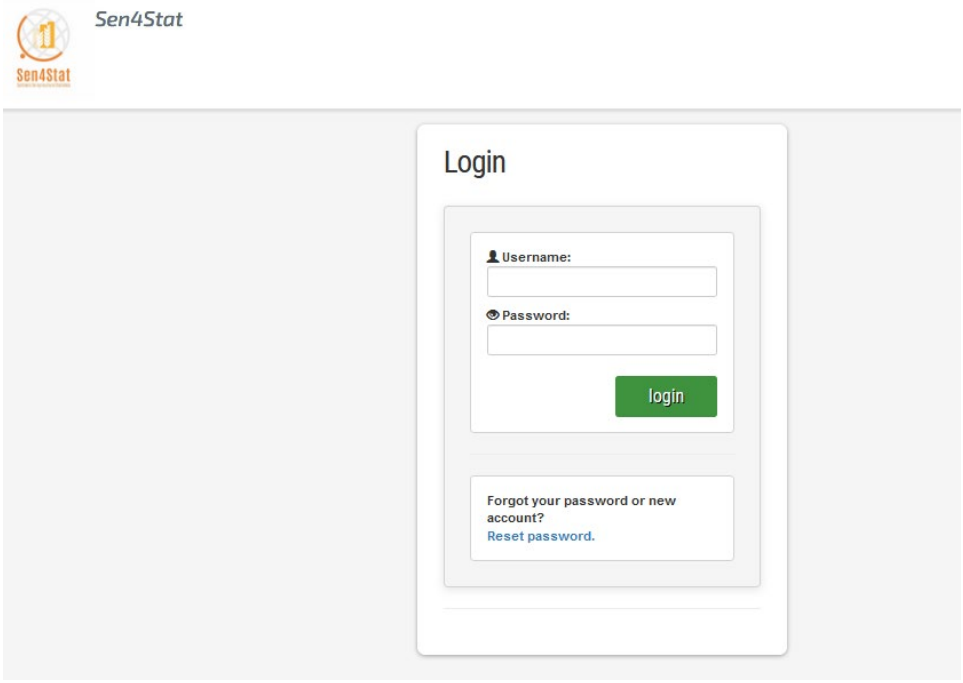


Figure 3-4. Main page of the Sen4Stat web interface

The interface can be also accessed in the same manner from another PC that has access to the port 80 of the Sen4Stat system, by providing in the browser the IP address or the computer name of the Sen4Stat system like for example http://[sen4stat_ip_address]:8080/ui/login.html or http://[sen4stat_machine_name]:8080/ui/login.html.

After the user provided a correct user and password (by default, “sen4stat” and “sen4stat”), the user is automatically redirected to the “Sites” page where he is invited to configure his site to monitor (see section 4.2.2).

This web interface should allow performing all operations of the Sen4Stat system, allowing users that are not-Linux expert to configure and run the system without problem. Thus, in section “Appendix E – Advanced system configuration items" is described also the ***editing the configuration parameters in the database***. To edit these configuration parameters in the database, the Sen4Stat configurator (or manual editing in the database) can be used. More details about these operations are provided in the next section.

[2] Safari is currently not supported to access the interface

4 How to use the Sen4Stat EO processing system

4.1 General

To put it simple, the Sen4Stat EO processing system transforms S1 SLC, S2L1C/ L2A and optionally, L8 L1T products into higher-level products by means of its L2 (pre-processing), L3x (monthly cloud free composite and biophysical indicators) and L4x (crop mapping, crop growth condition metrics and yield estimation) processors.

The normal functioning state of the system is the ***automated mode***, i.e. the automatic download / import and processing of the S1 SLC, S2 L2A and L8 L1T products and production of higher-level products.

The automatic execution of the system is based on the following items:

- At least one site needs to be defined inside the system. Additional sites can be created and edited from the "Sites" page of the Sen4Stat website;
- For each site, the extent and the season to monitor need to be defined. At least one season should be defined, specifying the start and end dates of the season. These parameters are associated to by-default values, which can be modified by the user. The season dates are defined by a month, a day and a year;
- Based on the AOI extent and season definition, the system automatically launches the downloader and the processors to generate the higher-level products. The automatic download and pre-processing are handled by the system with the objective to optimize resource and memory management.

IMPORTANT NOTE: in-situ data will also be needed to run the L4x (crop mapping, crop growth condition metrics and yield estimation) processors. How to format and import this dataset is detailed in section 4.4.4.

As for the download / import, some precisions need to be done:

- The system starts to download / import optical products 2 months before the start of the season date. For example, considering a start of the season on 2018-05-01, products download will thus start on 2018-03-01. These 2 first months of data are needed to initialize the L2A processor but will not be used to produce the higher-level products. If MAJA is not used for the pre-processing of L2A products, the download of images with an acquisition date preceding the beginning of the season can be avoided by modifying the "Season start offset in months" parameter in the downloader configuration panel;
- Similarly, the system also tries to download products during 2 months after the end season (so considering an end of season of 2018-09-31, it means until 2018-10-31), to avoid losing any products inside the season which may be added after the end of season day.

Each processor is associated with main and advanced parameters and each parameter has by-default values. Each processor is automatically run using these by-default values except if the user changes them. More information regarding the processor's parameters can be found in section 4.4. This by-default parameterization results from consistent studies carried out during the project. It is therefore recommended to be cautious when modifying them.

As an alternative to the ***automated*** mode, some of the processors can also be run in the ***manual mode***. In this case, processors can be invoked in various ways:

- from a terminal window, by executing a specific Python script;
- from the Sen4Stat web interface, for L3x and L4x processors. In this case, immediate execution is not performed. Instead, a scheduled job is created and the processor will be executed when the system decides it has enough resources to do it. This is done with the purpose of not interfering with the system production jobs already planned.

The following sections present how to run the system in the automated and manual modes. They also give information on the way to deal with the declaration datasets that are required for the L4x processors.

4.2 First steps for the automated usage of the system

4.2.1 Configure the data sources

Before starting the configuration of the sites to be monitored, it is first needed to configure the data sources to be used to fetch products into the system. The Sen4Stat system offers the possibility to adapt the data sources configuration directly from the web interface (Figure 4-1). For example, you can define (i) which downloading platform to use for the “query” and “download” tasks of the downloading, (ii) the user and password corresponding to your downloading platform, (iii) the destination folder for the downloaded products, etc.

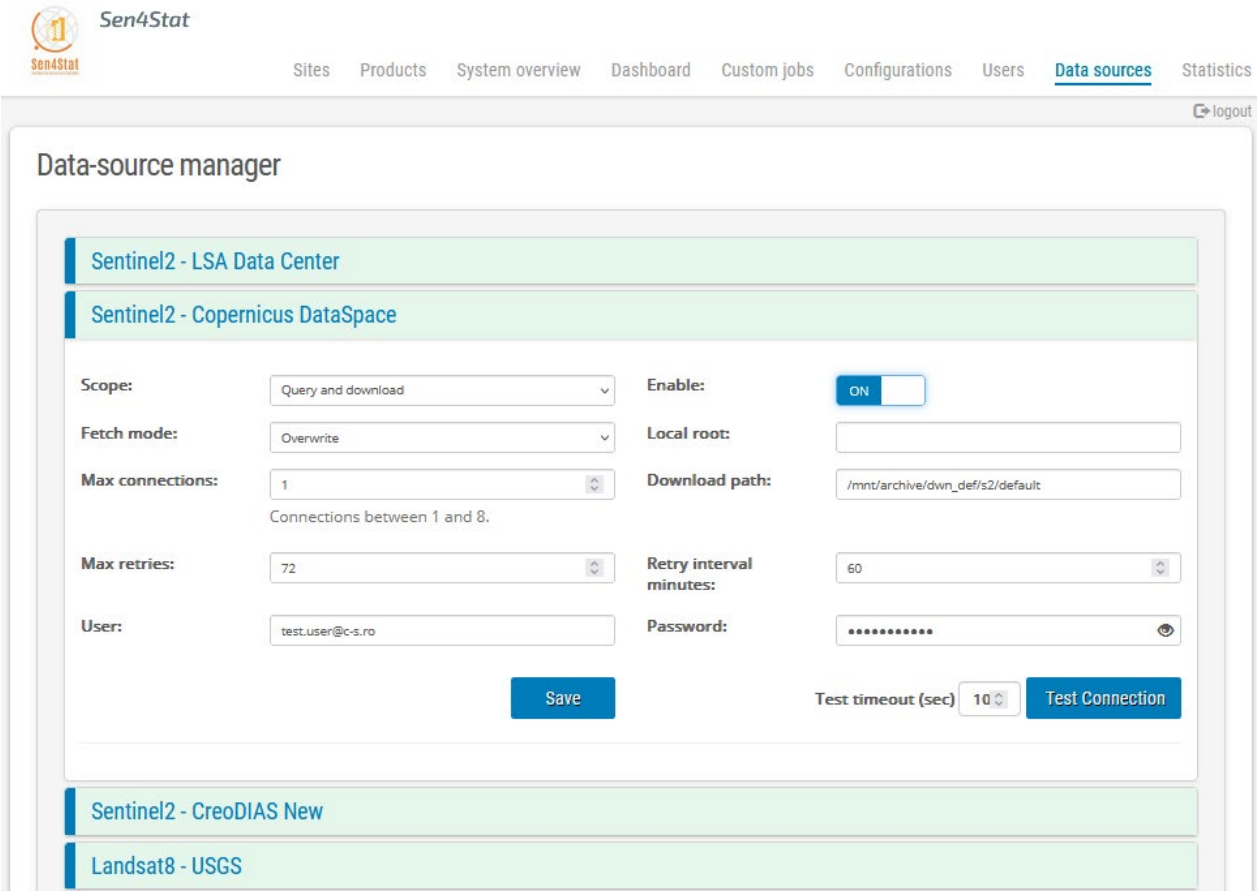


Figure 4-1. The "data sources" tab of the web interface - Configure the data sources parameters

A data source is a module that has the capability of searching and downloading products from remote sources. Every such module is intended for a combination of the remote source and a satellite. Out-of-the-box, the Sen4Stat system comes with three pre-configured data sources: the Copernicus Data Space Ecosystem (Copernicus DAS) for Sentinel-1 and Sentinel-2 and one the US Geological Survey (USGS) for Landsat-8.

The default configuration uses both Copernicus DAS and USGS for search and download. This is enough for an on-premises installation of the system. Alternatives are also included and they are detailed in the following sub-sections.

IMPORTANT NOTE: The sen4stat-services are periodically checking for the changes in the configuration with a timeout of 1 hour. If it is desired to consider the changes immediately, the application can be restarted from a command line using:

```
sudo systemctl restart sen2agri-services
```

4.2.1.1 On-Premises Installation without any Local Product Repository

This is the default scenario and the only necessary configuration of the Sen4Stat data sources consists in providing the credentials (user and password) for both Copernicus DAS and USGS.

4.2.1.2 On-Premises Installation with a Local Product Repository

When the system is installed on an environment which has a product repository available (i.e. products already downloaded in a local folder or on a share on the local network), the following should be applied (the steps are to be applied for all the data sources):

- Change the **[Scope]** of the data source to "*Query*". This way, the remote source will only be used for searching products over a site for the given season, but not to download them.
- Change the **[Fetch mode]** to "*Symbolic link*" or *"Direct link to product"*. When the services will execute the product lookup, they will just create symbolic links to the locally available products or the path to the local directory will be added directly into the database respectively.
- Set the **[Local root]** to the root path of the local product repository. Each data source can have its own local root.

  The default expected organisation of the local products is a folder hierarchy into which products are grouped by year, month and day.

  For example:

  `/mnt/store` -> this is the local root

  `/2019`

  `/01`

  `/20`

  `/<product_1>`

  `/<product_2>`

  `/21`

  `/<product_3>` etc.

  Nevertheless, the structure of the local repository can be described via a couple of settings which are available in the **/usr/share/sen2agri/sen2agri-services/config/services.properties** configuration file. These are:

  - **`<datasource>.local.archive.path.format`**

    A relative path, supporting some placeholders, should be used. The supported placeholders (for all satellites) are:

    - yyyy = the year
    - MM = the month, left-padded with '0'
    - DD = the day, left-padded with '0'

    For example: "S1-SLC/yyyy/MM/DD", where S1-SLC is a real folder

This would allow the grouping of products by acquisition date and for easier existence check.

- **`<datasource>.path.suffix`**

  This specifies if the products, if uncompressed, have their product folder ending in .SAFE, or, if compressed, the extension of the archive file. Allowed values are:

  - *none*
  - *.SAFE*
  - *.zip*
  - *.tar_gz*

- **`<datasource>.product.format`**

  This specifies the format of the products and can have one of three values: '*folder*', '*zip*' or '*tar_gz*'

where <datasource> can be one of:

- *DASDataSource.Sentinel1*
- *DASDataSource.Sentinel2*
- *USGSDataSource.Landsat8*

IMPORTANT NOTE: Due to the recent changes in the external data sources APIs and structure of response, this feature might not be supported by all data sources and was not tested.

4.2.1.3 Installation within a DIAS Platform

The configuration of the data sources, in the case of an installation on a DIAS platform (supported DIAS platforms are Creo DIAS, Mundi, Sobloo and Onda) is similar to that of 4.2.1.2.

However, to cope with the specific paths of the products for each of the DIASes, additional configuration needs to be done.

In the configuration file **/usr/share/sen2agri/sen2agri-services/config/services.properties**, there are 3 sections, commented, for Creodias, Mundi, Sobloo and Onda (Creo DIAS is using the default configuration).

Therefore, depending on the DIAS platform where the system is installed, the respective section **has to be uncommented** (and leave the others commented).

4.2.1.4 Using FMask as masks provider

When FMask is enabled and ESA S2 L2A created with Sen2Cor are downloaded or fetched from a local repository (this is the default mode in Sen4Stat), the system will need to download both S2 L1C and L2A products from the configured data source. Indeed, FMask is using as input L1C and not L2A products.

This mode is configured by having both keys 'downloader.use.esa.l2a' and 'processor.fmask.enabled' set to "enabled" in the "config" table (or in the website configurator)If the 'downloader.use.esa.l2a' key is not set to "true", the system need to be configured to use MAJA for the L1C pre-processing into L2A products.

4.2.2 Configure a new site

By-default, the Sen4Stat platform does not contain any geographical site. This is to the user to create at least one site. This site is then used by the downloader/import processor to download S1 SLC, S2 L1C/L2A and optionally, L8 L1T products. The first step for the automated usage is therefore to create and configure a new site, which means defining the AOI.

In order to create a new site, the Sen4Stat system interface should be used. Select the "**Sites**" tab and press the "**Create new site**" button. At that moment, the dialog box for configuring a new site is displayed (Figure 4-2).

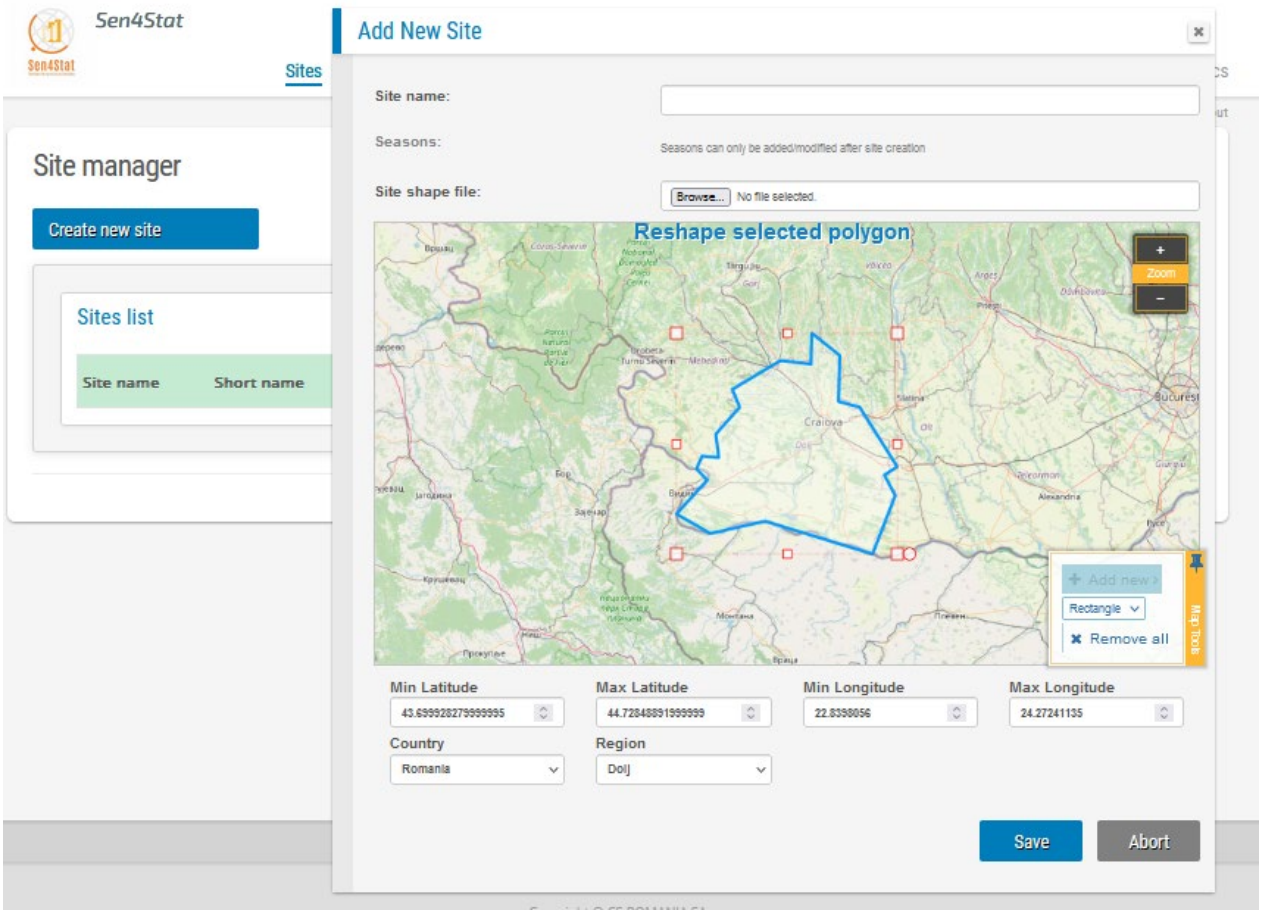


Figure 4-2. Dialog box for site creation through the Sen4Stat system interface

In the "**Add New Site**" dialog box, the following information should be provided:

- Site name: this should be a unique site name defined in the system. The name of the site should follow the following rules: (i) first letter must be uppercase, (ii) letters, digits, spaces and underscores are allowed;
- Define the site extent using the following options:

- "Site shape file": Upload a shapefile with the site extent. The uploaded file will have to be in a ".zip" archive that contains at least ".shp", ".shx" and ".prj" files (".dbf" files can be also present but they are not mandatory). The site can be defined by more than a polygon, i.e. the shapefile can contain a multi-part polygon;

IMPORTANT NOTE: Make sure that your polygon (or each polygon in the case of a multi-part polygon) is **not made of more than 150 points.**

- Draw a rectangle or a multi-point polygon on the map;
- Select a Country and/or region from the available Nomenclature of Territorial Units for statistics (NUTs).

At any time, the chosen area can be reset by selecting "Remove all" from the map.

After providing all information in the dialog box, the user should choose "**Save**" and the site is created.

An error will be given if the information provided is not correct (for example in the case of invalid shapefile archive).

Once at least one season is added (see next session) and the site is "Enabled", the download and processing will automatically start. If the user needs to check and/or change some of the processor's parameters in the system, it is highly indicated to disable the site and re-enable it when all the parameters are set.

4.2.3 Editing the sites to define season and uploading the mandatory auxiliary data

4.2.3.1 Editing a site

As specified in the previous paragraph, by default, a new created site is not associated with season dates, which are to be defined in a second step. To this end, on the Sen4Stat system interface, go to the "**Sites**" tab and press the "**Edit**" button corresponding to your site (Figure 4-3). The dialog shown in Figure 4-4 will then open.

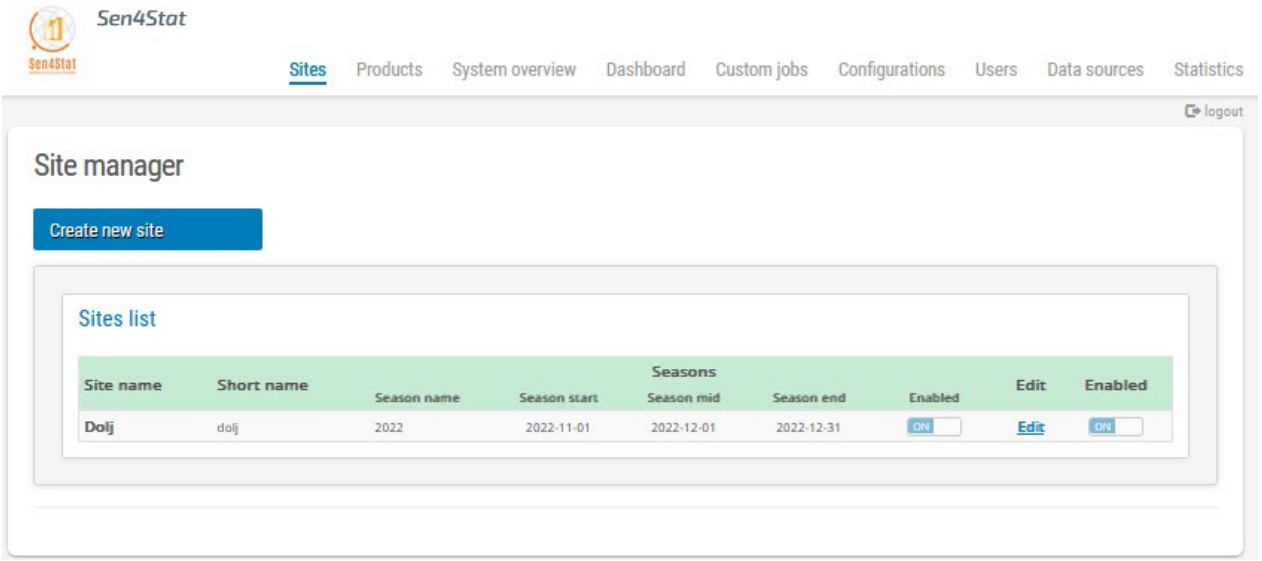


Figure 4-3. Site page of the Sen4Stat interface, to edit site or seasons

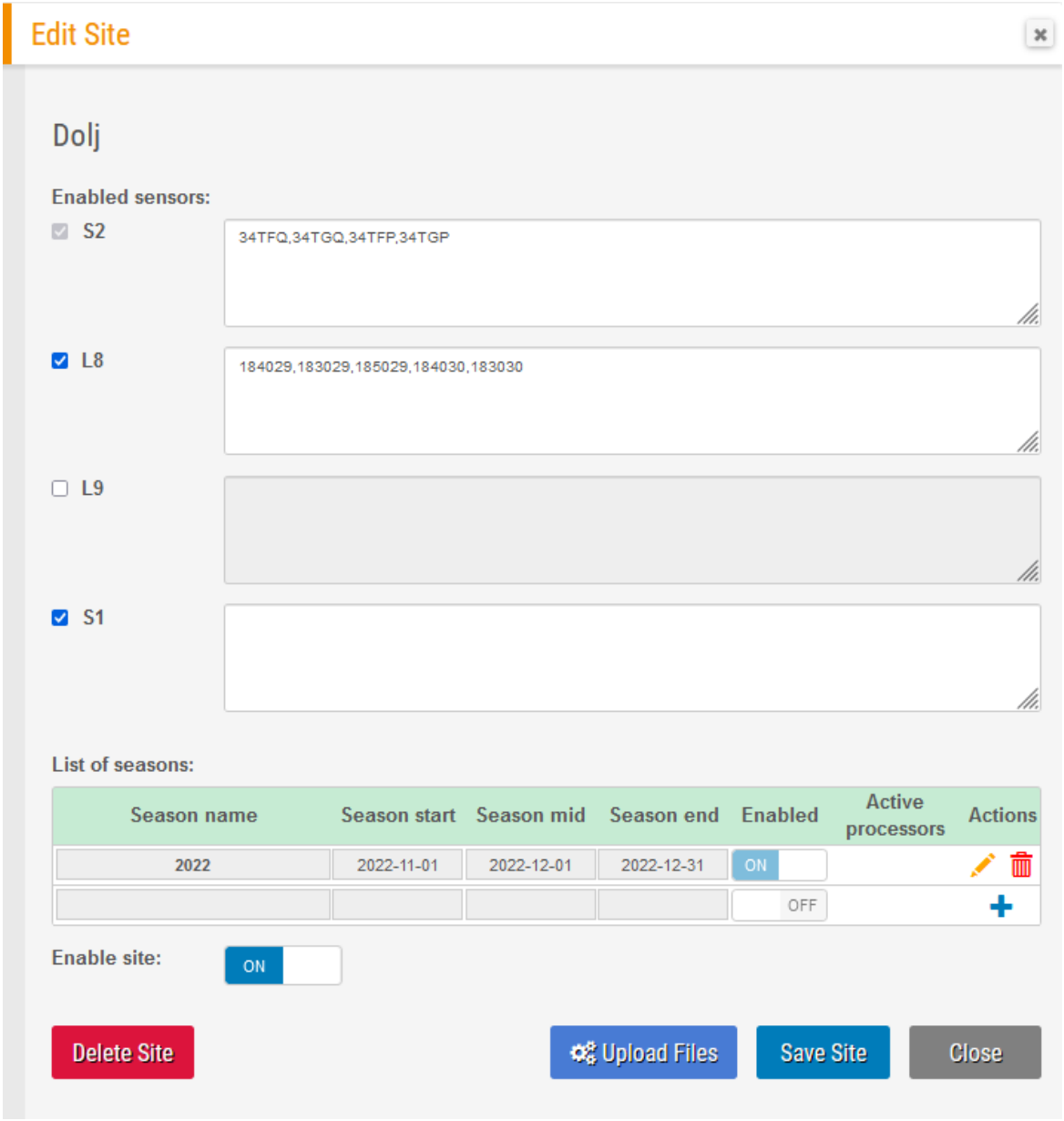


Figure 4-4. Dialog box for sites edition (including season dates) through the Sen4Stat system interface

In order to add a season, press the + icon in the "**List of Seasons**" section. It allows you specifying the following parameters: "Season Name", "Season start", "Season mid" and "Season End". It also gives you the possibility to decide if the season should be enabled or not.

At this moment, the user has the possibility to specify which processor is to be run in the automated mode for the site and season just defined. This is done simply by ticking the boxes associated with the processors you want to activate, as illustrated in Figure 4-5. Note that the pre-processing processors (for S1, S2 and declaration datasets) are always run automatically.

The user can also select what sensor to enable for downloads: by default, L8, S1 and S2 sensors are "Enabled" but the L8 sensor can be disabled. The dialog presents also the list of S2 and L8 tiles intersecting

the site. The user has the possibility to remove some of them, if he is not interested in all of them. The editors for the tiles become enabled once the site is enabled.

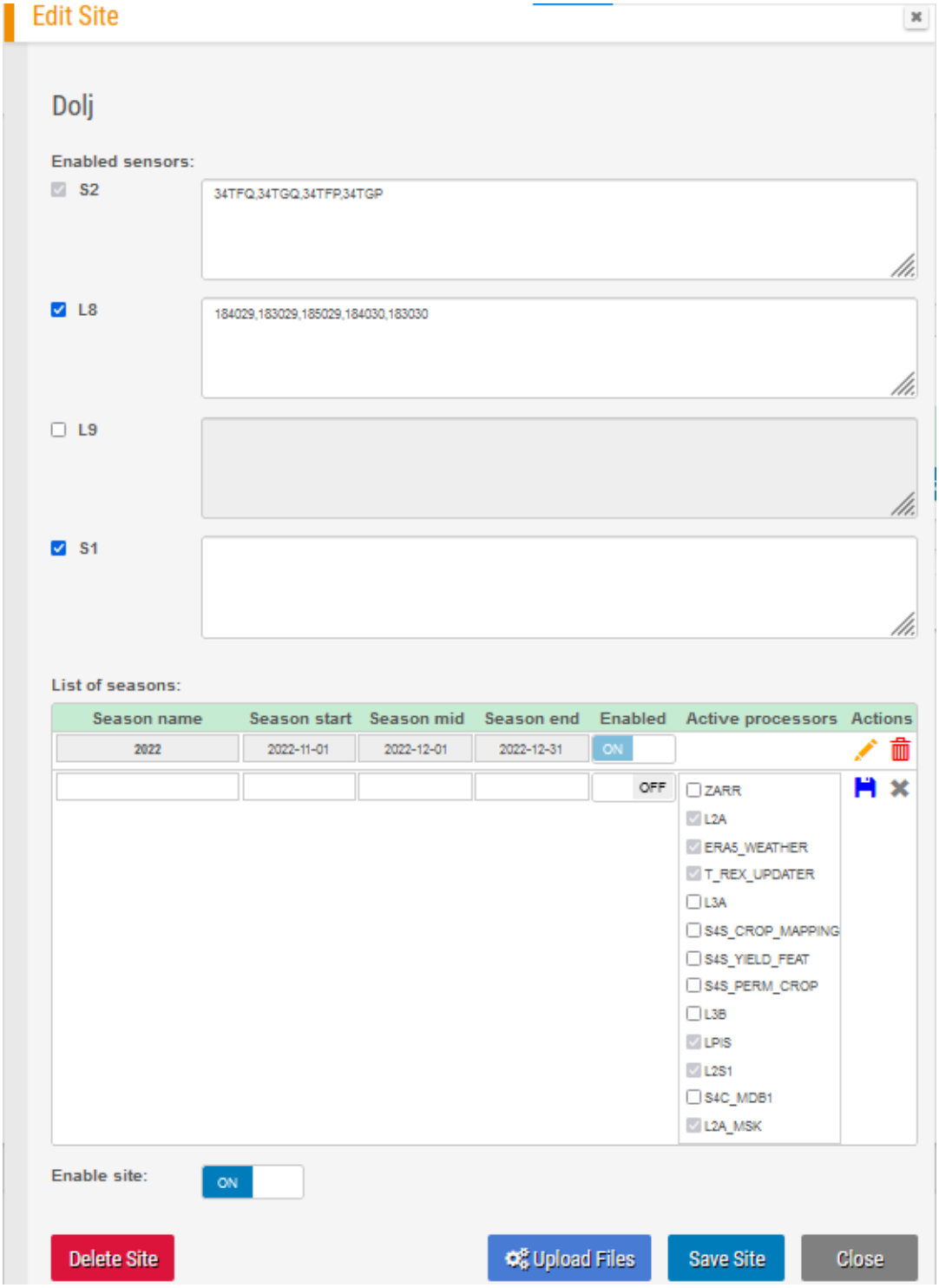


Figure 4-5. Dialog box when a season is defined and the possibility to activate processors is visible

The processors that can be active are:

- L2A and L2-S1: these processors correspond to the pre-processing of both S1 and S2. As already mentioned, they are enabled by default (cannot be modified) as they are the ones that provide input products for all other processors;
- L2 - In-situ data preparation: this processor performs the formatting and quality control of the NSO datasets. It is also enabled by default (cannot be modified) and it is triggered only when a new dataset is uploaded into the system;
- L3 – monthly cloud-free composite: a "Repeat" type schedule will be created that will execute every 1st day of the month in order to create the L3A corresponding to the beginning of the previous month
- L3 - Spectral indices and biophysical indicators (NDVI, NDWI, Brightness, LAI, FCOVER and FAPAR): this processor is associated with a "Cycle" type schedule, i.e. it will execute every day in order to generate one set of L3B products for each L2A product that was acquired and pre-processed during the last day;
- L4 - Yield features: this processor is associated with a "Once" type schedule, i.e. it will be run at the end the season;
- L4 - Crop mapping: this processor is associated with a "Repeat" type schedule, i.e. it will be run at the end of each month;
- L4 - Yield estimation: this processor is associated with a "Once" type schedule, i.e. it will be run at the end the season.

The other processors are activated automatically by the system depending on the needs.

IMPORTANT NOTE: This possibility to specify the active processors is only available when the season is defined. It cannot be edited later on.

Once the season is defined, it can be saved using the icon. The changes can be cancelled using the icon .

By selecting the processors to be active for the season, the corresponding scheduled jobs will be added to the season for automatic processing. Please refer to section 4.3 for additional information about the scheduled jobs. These jobs are available in the "Dashboard" section of the Sen4Stat system interface and will start to execute once the site is Enabled.

At this moment, after the creation of the season, the user has the possibility to enable the site.

4.2.3.2 Uploading and importing in-situ data

In-situ data (coming by default from an agricultural survey conducted by the NSO) is mandatory to run the higher-level products (L4). This dataset shall include both geometries (polygons corresponding to parcels, segments, blocks, etc.) and crop data (crop type, crop area, yield, etc.). The system can start working without this dataset, performing already the download / import, the S1 and S2 pre-processing (creating the L2A time series with validity mask and the coherence and backscatter time series) as well as the L3 composites, vegetation indices and biophysical indicators. As soon as they are available (and in any case, before the creation of the L4 products), the in-situ data needs to be uploaded.

In order to upload this dataset (but also other processors files), the "Upload files" option can be used. A new window then opens, like in Figure 4-6.

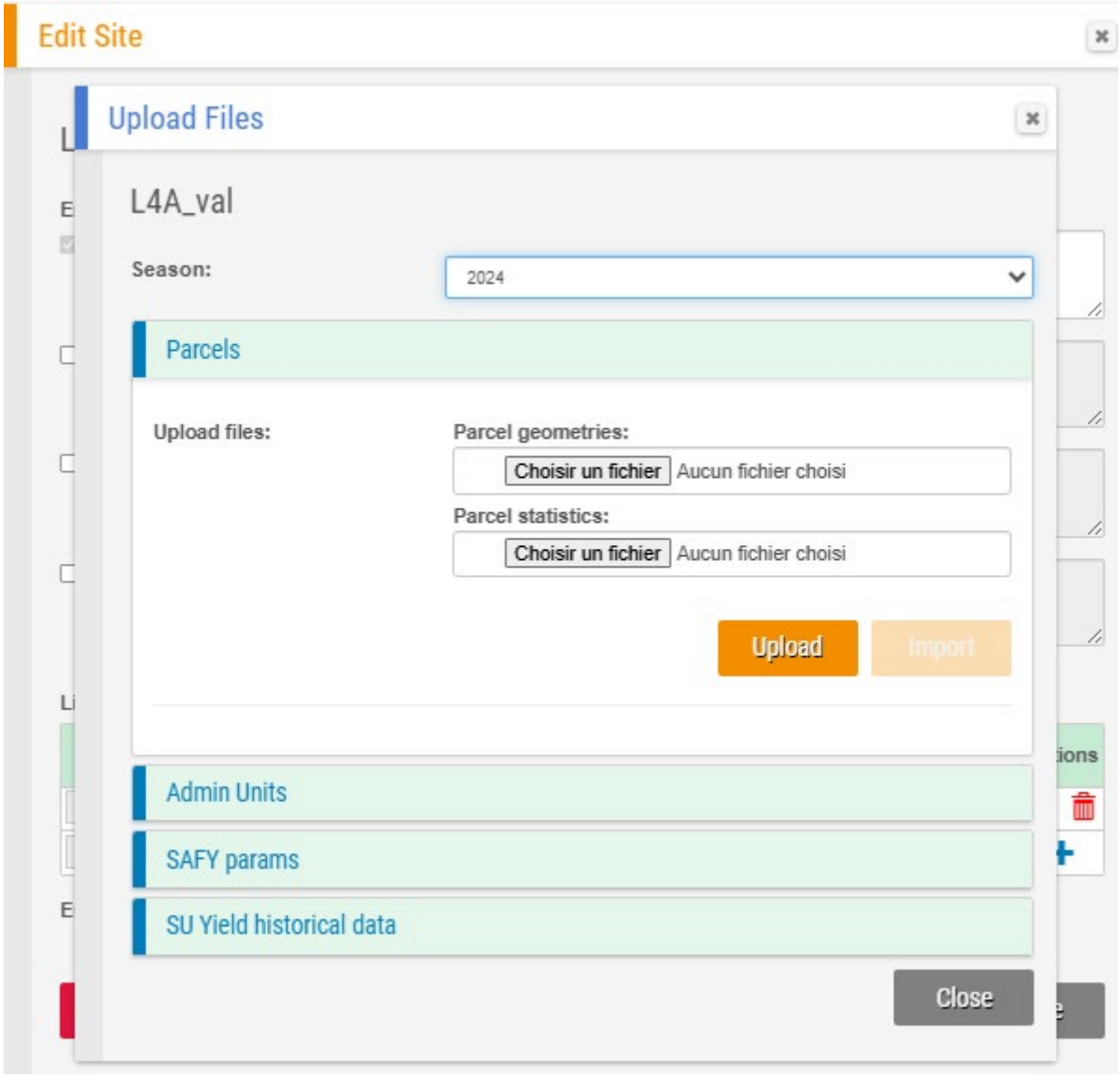


Figure 4-6. Uploading NSO in-situ geographical and statistical datasets

The geometries (polygons delineation) are to be uploaded through the "Parcels" section, in the form of a .zip file containing a shapefile. dataset.

IMPORTANT NOTE: The archive should contain the shapefile files on the top level and no other subdirectories should be present in the archive.

The crop type / yield data collected during the agricultural survey shall be uploaded in the "Parcels statistics" section of the same page. The file should be a CSV file with the structure defined in section 4.4.4.1.

4.2.3.3 Importing Administrative units

In order to import the administrative units' geometries, that will be later used by the Yield processor, the following option should be selected (Figure 4-7).

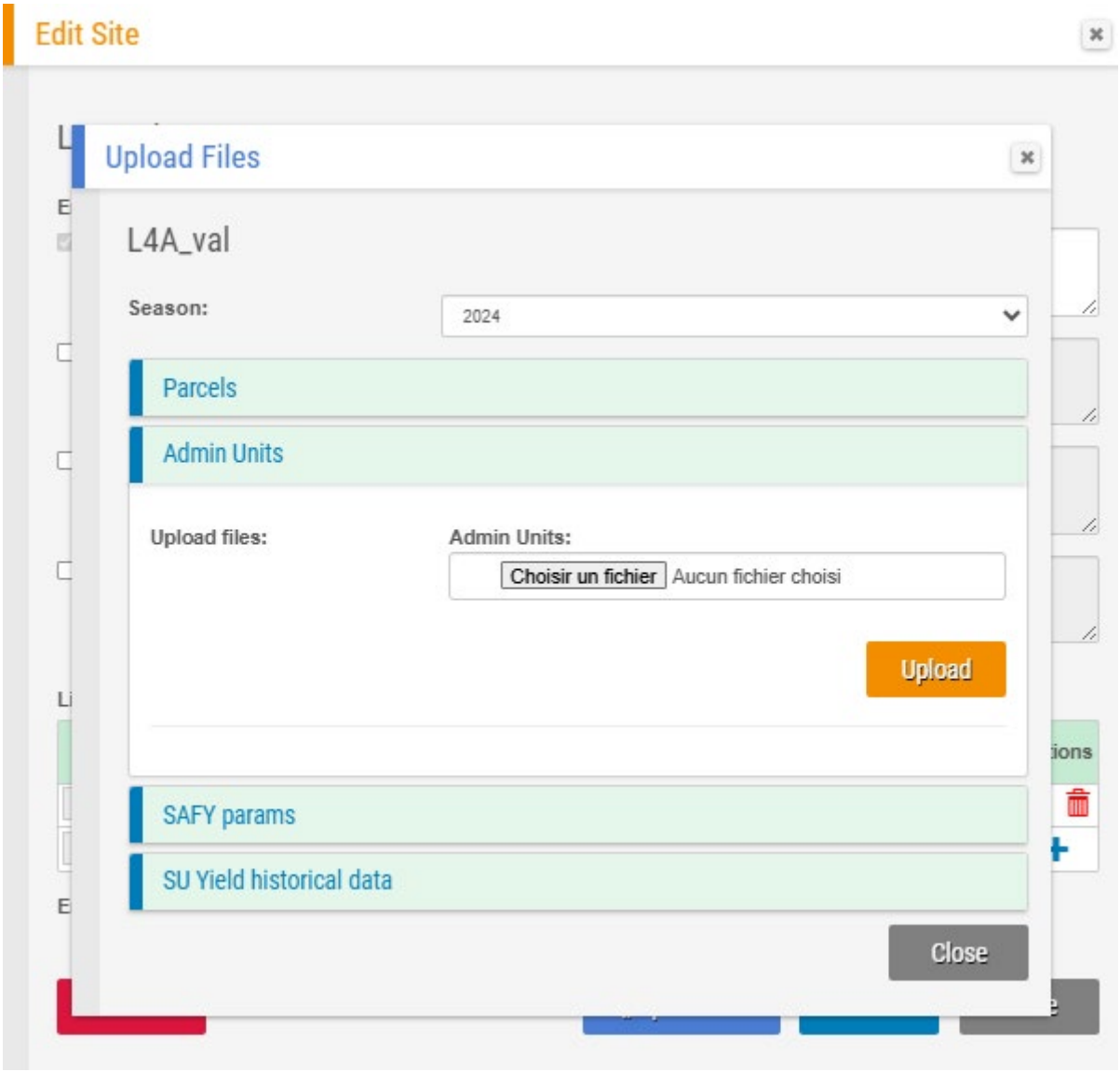


Figure 4-7. Importing Administrative Units shapefile

The imported file needs to be a zip file containing the shapefile with the administrative units.

4.2.3.4 Importing Yield historical data

For the Yield SU computation, when no yield reference is provided at parcel level, the system will require a file with the historical Yield values at SU level for the previous years.

This needs to be provided as a CVS file with the values for each year. In order to upload it into the system, the following option needs to be selected (Figure 4-8).

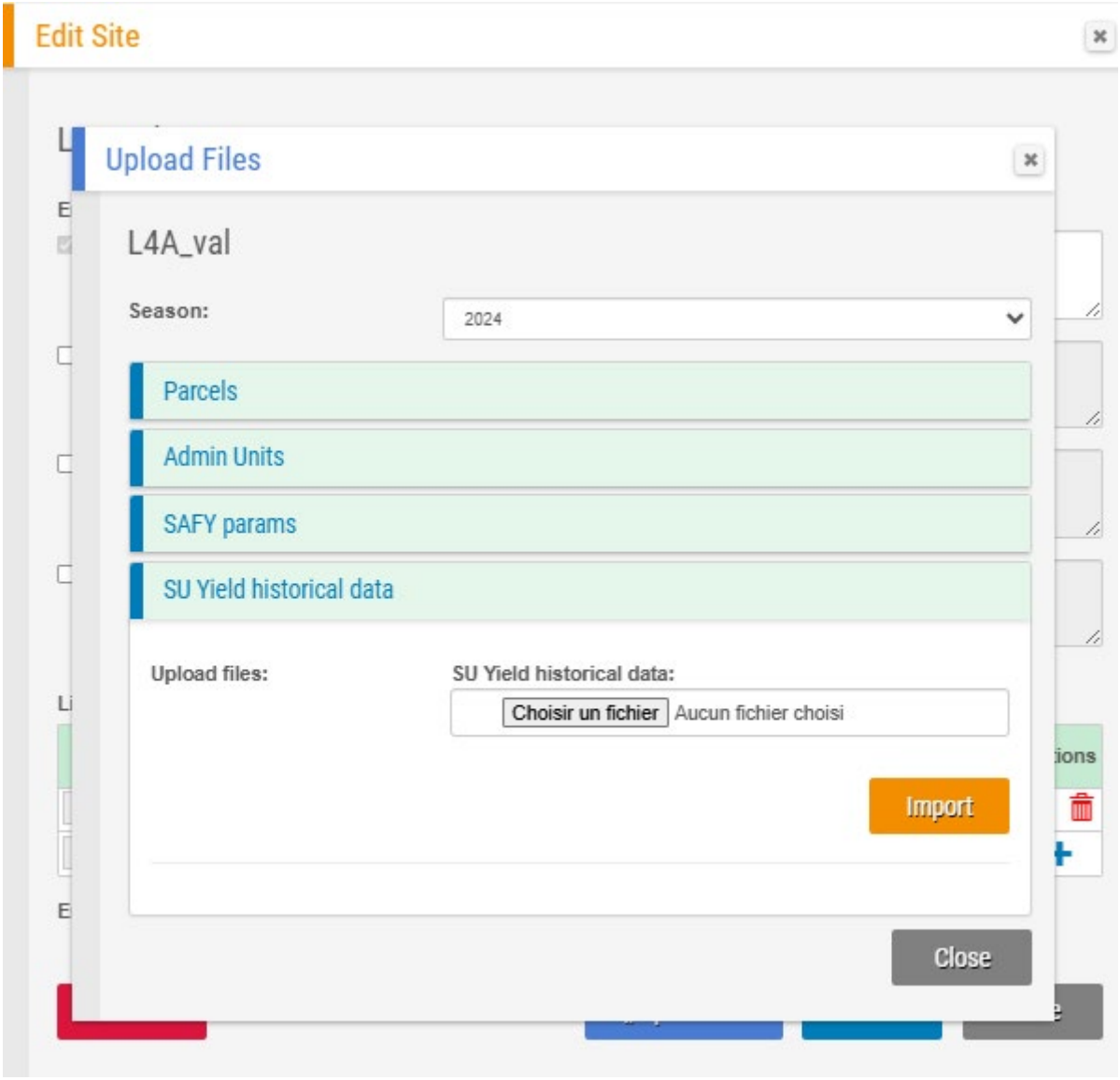


Figure 4-8. Uploading Yield SU historical data

4.2.4 Optionally adapt the main processor parameters

As introduced in sections 2.1 and 2.2, the system is made of several processors, allowing producing a variety of high-level products. Each of these processors is parameterized with by-default values when the system is installed. The automated mode is run using this ***by-default parameterization*** except if the user wants to change them. Changing the by-default parameters can be performed by using the "sen2agri-config" application or by manually changing the parameters values in the database (see *Appendix B - Sen4Stat database tables for manual configuration* and *Appendix E – Advanced system configuration items*). It should be noted that ***this by-default parameterization was found to be the best generic values*** after a consistent benchmarking carried out during the project; consequently, any change needs to be done cautiously.

The parameters of each processor are presented in detail in the section 4.4 about manual operations.

4.2.5 Activate or change the site parameters

If the site was not activated during the season definition, it can be "Enabled" at any time by navigating in the "**Sites**" tab and accessing to the Site Edition dialog by pressing the "**Edit**" button (Figure 4-9).

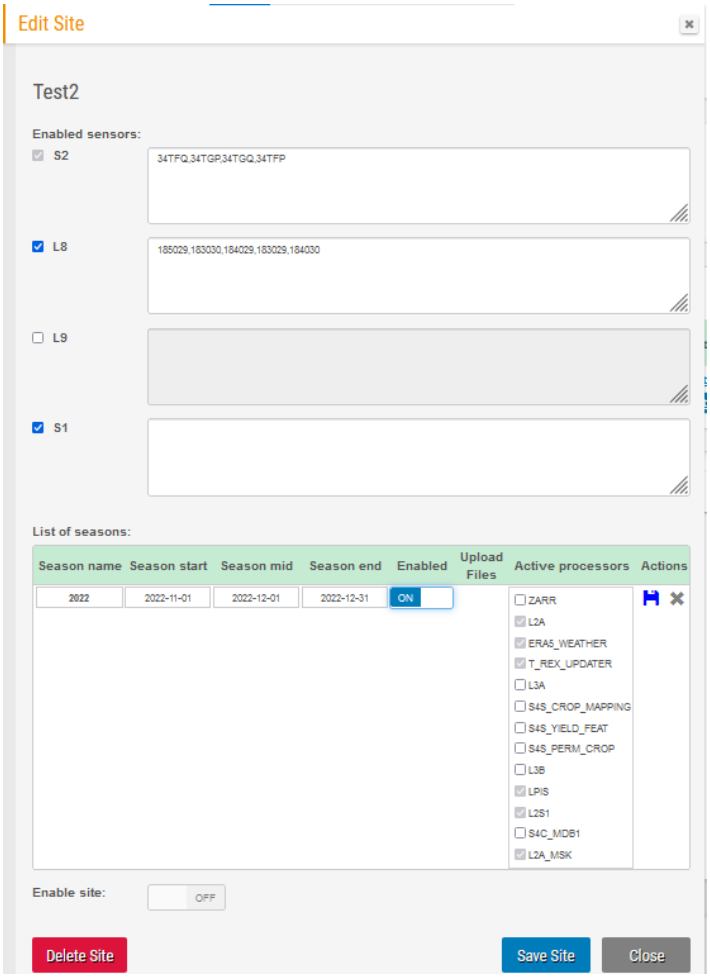


Figure 4-9. Edit site window allowing users modifying site parameters and/or activating it

This dialog box allows activating the site, but also modifying its season (and in the next version, uploading or modifying the subsidy application or other processor configuration files).

Seasons can be edited using the button or deleted using . The list of activated processors for a certain season can be viewed by moving the mouse over the text *"hover to reveal"* but not modified.

The user has the possibility to enable or disable the L8 downloads by unchecking the "L8" button from the "Enable sensor" area.

Please note that there exist two levels of enabling options:

- At season level: the user can enable or disable different seasons of a site;
- At site level: the site, with all seasons (if more than one season has been defined) can be enabled or disabled at once.

4.2.6 Delete a site

The user has the possibility to delete a site at any time by navigating in the "**Sites**" tab and accessing to the Site Edition dialog by pressing the "**Edit**". From this dialog box the user should press the button **"Delete Site"** (also visible in Figure 4-9).

By default, this dialog box allows the user to delete the site with all associated products.

In addition, the user can keep some products of this site by unchecking the boxes associated with the products. Then the user will select the button "**Confirm Delete Site**" (Figure 4-10).

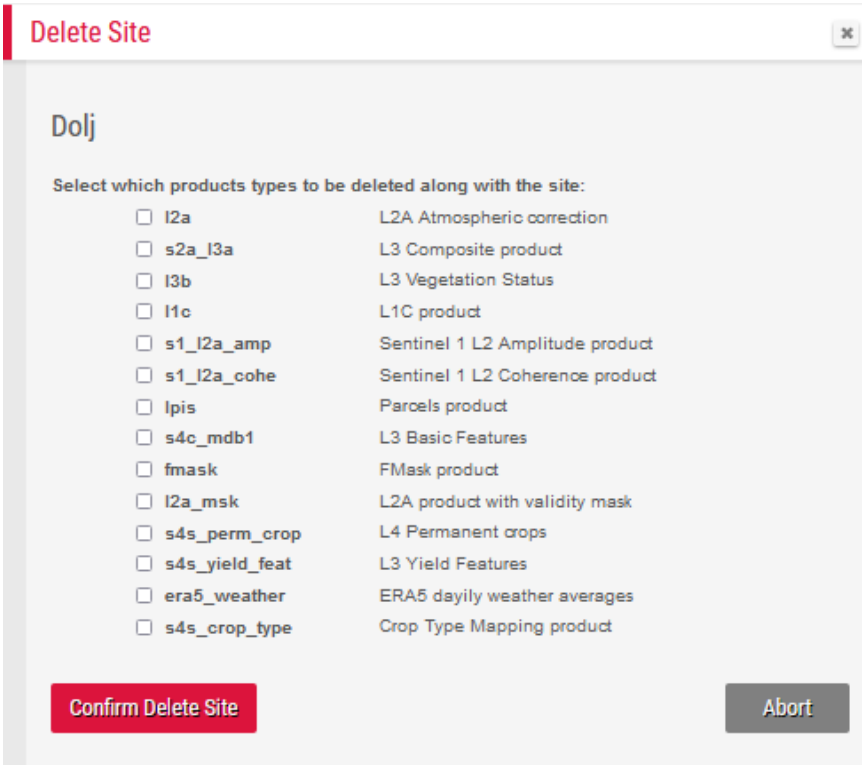


Figure 4-10. Site page of the Sen4Stat interface, to delete a site

4.2.7 Monitor the downloading

The Monitoring tab of the web interface gives various information concerning the downloading of the data (Figure 4-11). In the "Download statistics" bar, you can find:

- The number of images that have been downloaded (and %);
- The number of images that are currently being downloaded (and %);
- The number of images from which the download has failed (and %);
- The number of images from which the download has failed but will be retried (and %).

It also gives you the estimated number of products still to be downloaded.

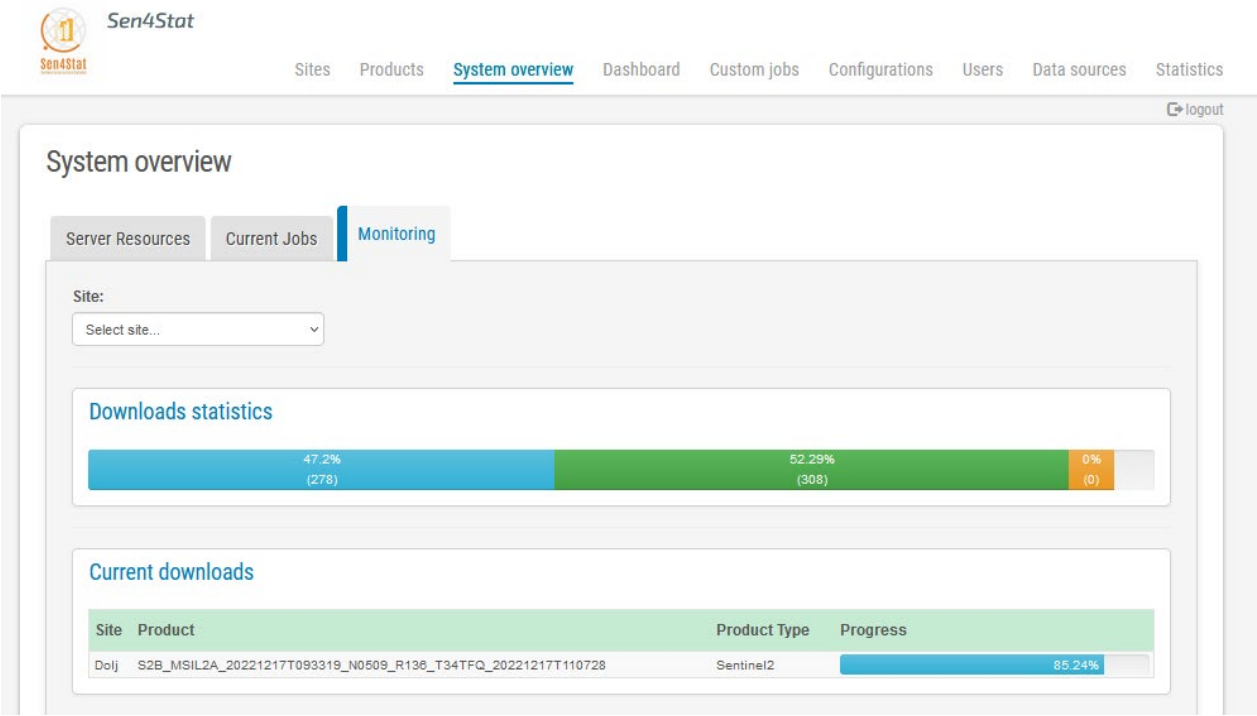


Figure 4-11. The “monitoring” tab of the web interface - A view to the downloading

4.2.8 Monitor the processing

The Execution Dashboard enables the user to monitor the Sen4Stat system. It consists of a web page that dynamically loads data from the Sen4Stat HTTP listener service.

The System Overview tab of this Dashboard (Figure 4-12) presents the jobs that are currently being executed by the system as well as the performance of the server(s).

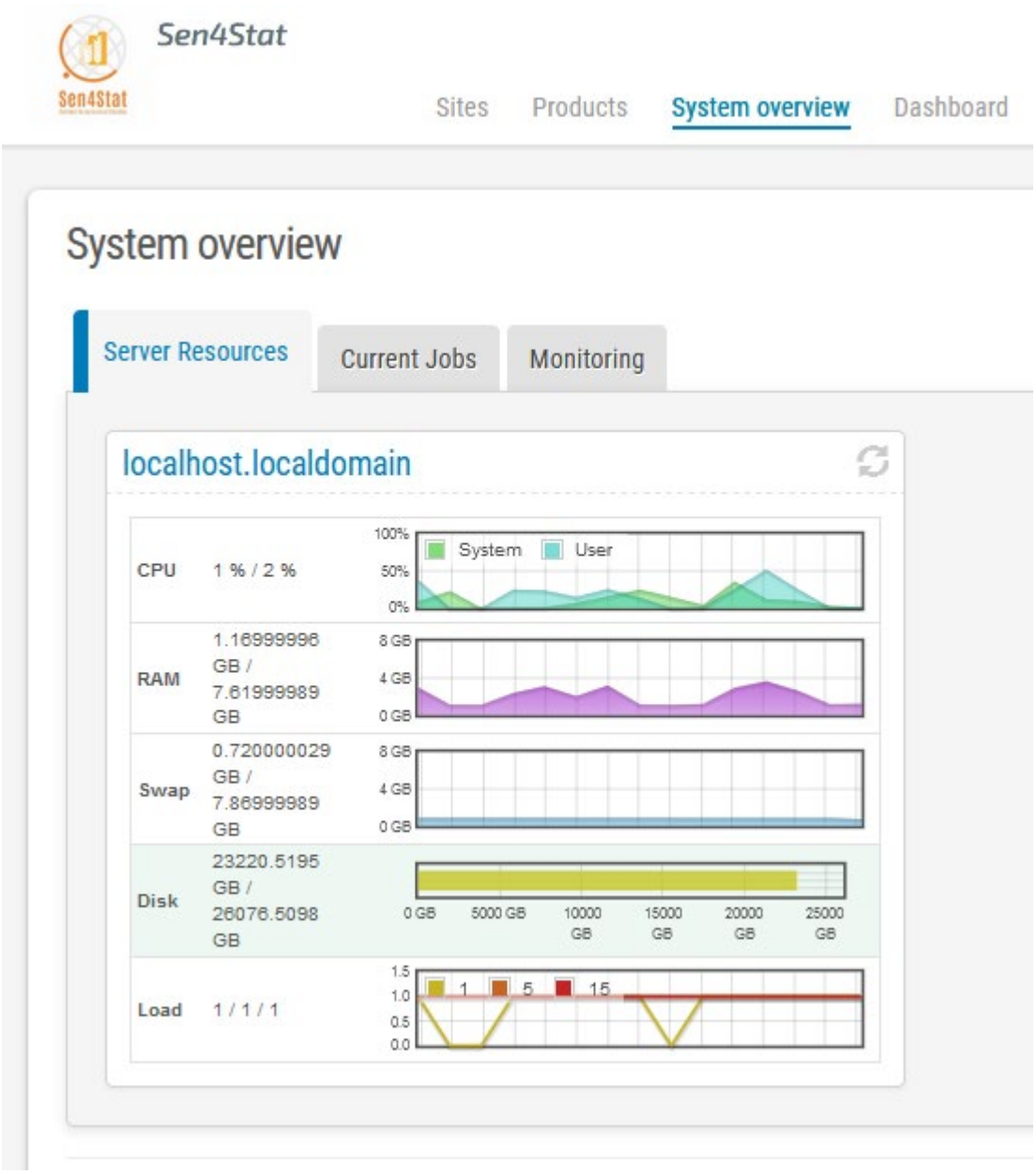


Figure 4-12. Sen4Stat Execution Dashboard – System Overview

For each server (top of the system overview), the following resources are presented:

- CPU (current value and a graph showing the evolution over the last 15 minutes);
- RAM (current value and a graph showing the evolution over the last 15 minutes);
- Swap (current value and a graph showing the evolution over the last 15 minutes);
- Disk – the used disk space from the available disk space;
- Load (current value and a graph showing the evolution over the last 15 minutes) – the load on the server is provided by the operating system in the form of 3 numbers that represent the 1/5/15 minutes averages of the *run-queue length*: the sum of the number of processes currently running plus the number of processes waiting (queued) to be run.

The ongoing execution jobs information can be visualized by accessing the “Current Jobs” submenu, as in Figure 4-13.

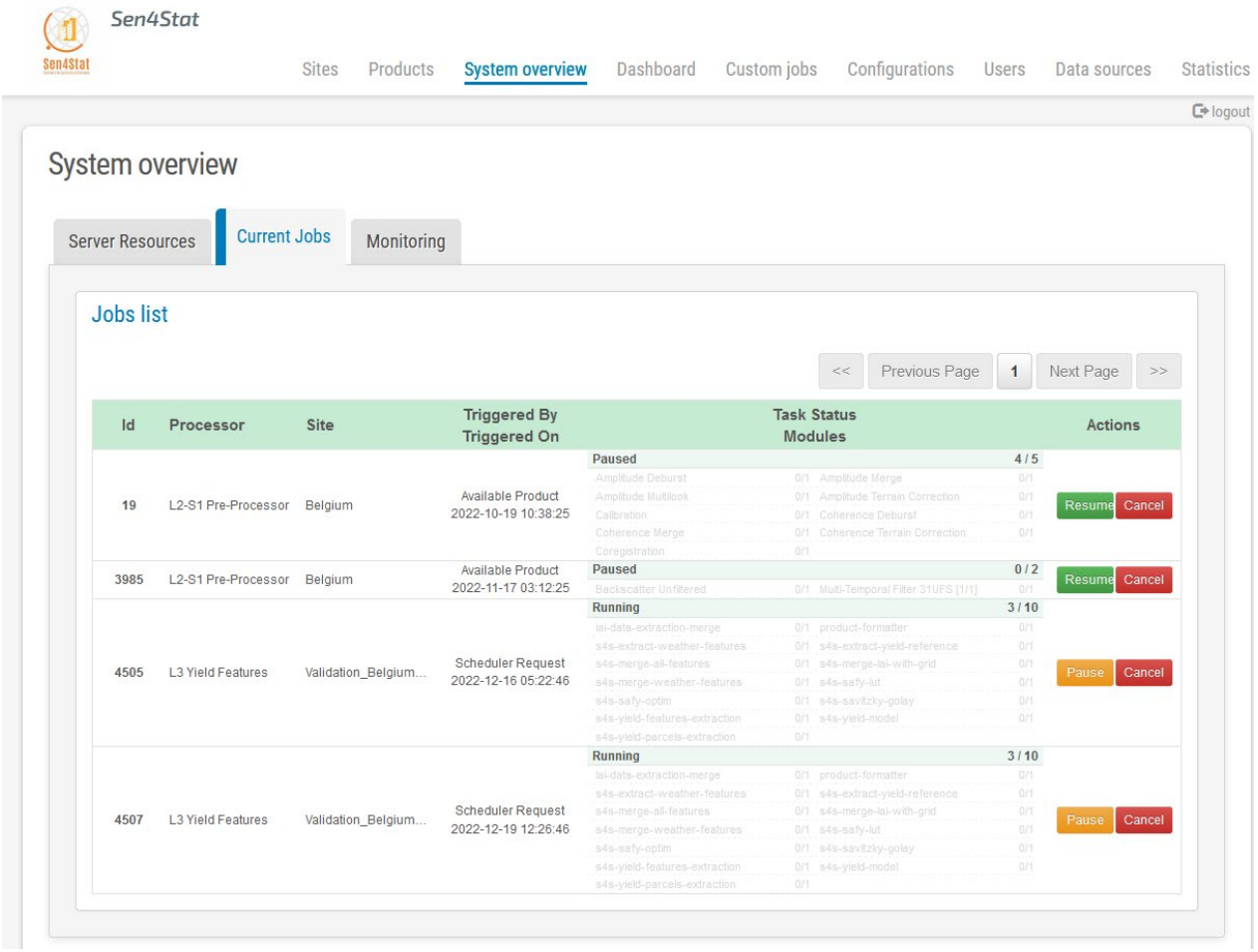


Figure 4-13. Current jobs execution monitoring

For every job, the following information is presented:

- Id;
- Processor;
- Site;
- Triggered By - the possible values are:
  - “Scheduler request” – when the job was created by the automated scheduling mechanism;
  - “User request” when the job was created with a custom job request;
  - “Available product” when the job was created due to the occurrence of a product that the corresponding processor is monitoring;
- Triggered On;
- Status (of the job) - the possible values are:
  - Running;
  - Canceled;
  - Paused;

- Tasks Completed/Runing;
- For the current task:
  - Module;
  - Tiles Completed/Running;
- Actions (depending on the status):
  - As possible actions for a job, we have “Pause”, “Resume” and “Cancel”;
  - “View config” – will show the parameters used when the job was submitted.

Each job has associated two buttons:

- Pause or Resume:
  - The button “Pause” is displayed when the current job is running and allows the user to interrupt the job execution
  - The button “Resume” is displayed when the current job is interrupted and allows the user to resume the job execution
- Cancel : This button allows the user to sto the job execution

Additionally, in the Monitoring tab (Figure 4-14), information can be found about each step executed, executing or to be executed from a job, with details about the command executed, the logs and errors (if any). By pressing on the “[output]” button, details about the job steps can be viewed (Figure 4-15).

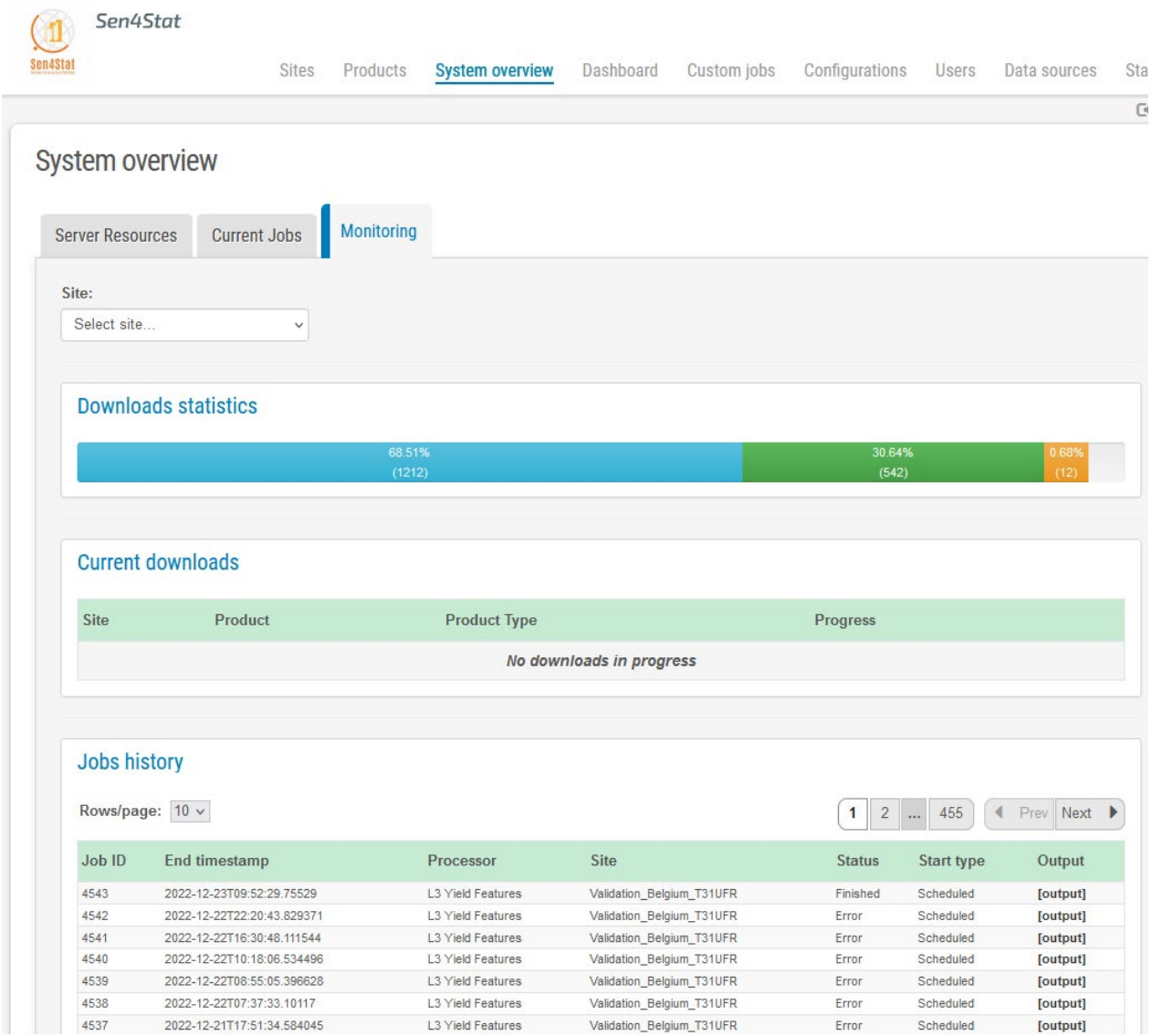


Figure 4-14. Sen4Stat monitoring tab

:

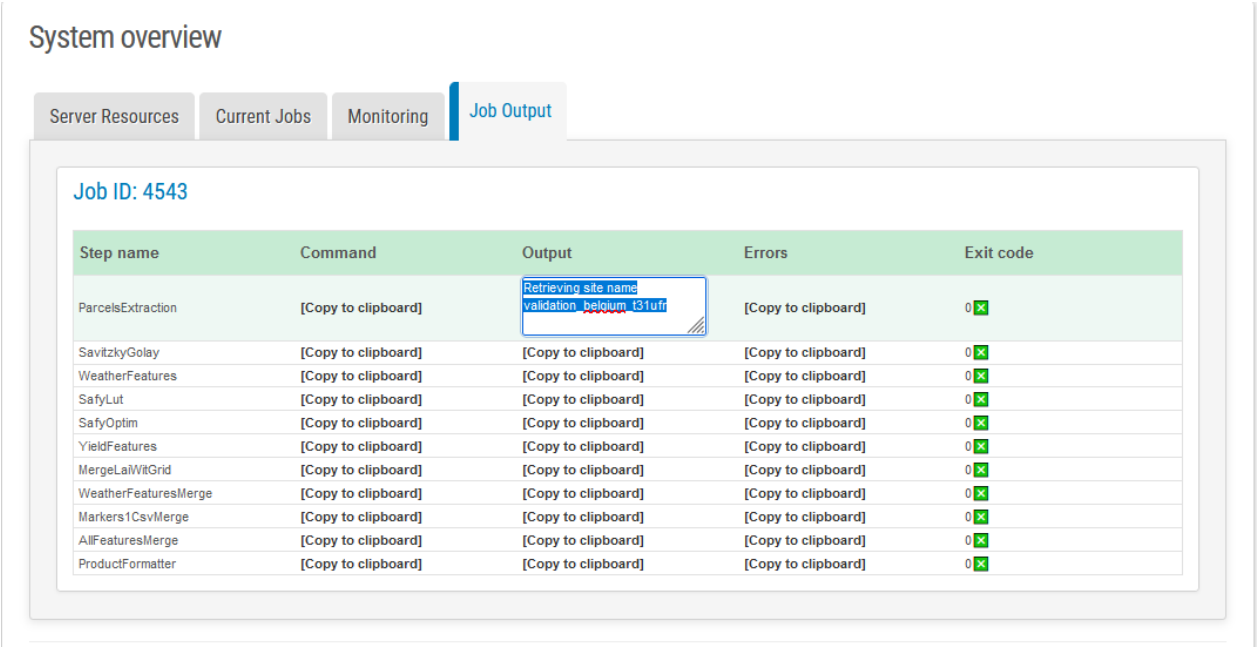


Figure 4-15. Monitoring the steps execution for each product

For each step, by pressing "[Copy to clipboard]" on the corresponding column, the command, output or the error messages can be visualized and automatically copied. Each step has also an icon and the exit status of it (if executed).

4.2.9 Where to find output products

The "Products" tab of the Execution Dashboard (Figure 4-16) shows a tree view with the products that are available. On clicking the site node, all the products available for this site become visible. If the icon of the product is clicked, the full resolution image is visible on the map, with option to zoom in and zoom out. A properties window is also available, showing information about the product with the option to download it locally so that it can be visualized with another application like QGIS or SNAP.

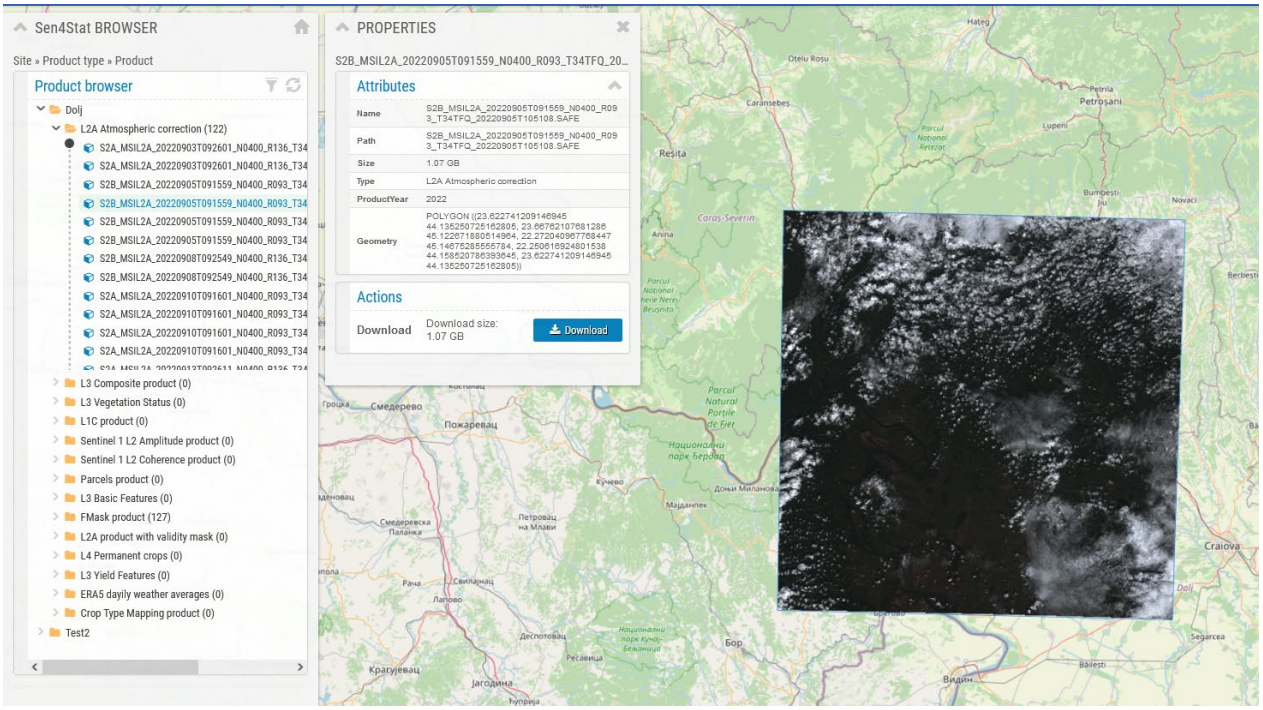


Figure 4-16. Sen4Stat – Product Browser

The paths where the products can be found are the following ones:

- Normally, when working on DIAS, the L2A/L1C/L1T products are used directly from the EODATA repository. Nevertheless, if they are not available here after a certain timeout, they will be downloaded locally. In this case, the L2A/L1C/L1T products found in */mnt/archive/dwn_def/* (see configuration parameters "Write directory for L8" and "Write directory for S2" from *Appendix E.7 Sen4Stat Downloader*)
- If the optical L2A products (Sentinel 2 and Landsat 8 L2A products) are created using MAJA (not the default configuration), they can be found in */mnt/archive/maccs_def*;
- SAR L2A products (Sentinel 1 L2A products) found in */mnt/archive/{site}/l2a-s1,* where *{site}* represents the short name of the site;
- L3A products (corresponding to composites) stored in a folder having the short name of the site*, which can be found in *"/mnt/archive"*. For example, *"/mnt/archive/spain/l3a";*
- L3B products (corresponding to spectral indices and biophysical indicators) stored in a folder having the short name of the site*, which can be found in *"/mnt/archive"*. For example, *"/mnt/archive/spain/l3b"*. Note that from version 4.0, the L3B product are split into distinct products per indicator and for each tile.
- L4A products (corresponding to crop maps) stored in a folder having the short name of the site*, which can be found in *"/mnt/archive"*. For example, *"/mnt/archive/spain/l4a"*.
- L4B products (corresponding to crop yield metrics) stored in a folder having the short name of the site*, which can be found in *"/mnt/archive"*. For example, *"/mnt/archive/spain/l4b"*.

- L4C products (corresponding to crop yield estimation) stored in a folder having the short name of the site*, which can be found in *"/mnt/archive"*. For example, *"/mnt/archive/spain/l4c"*.

4.2.10 Statistics

The "Statistics" tab of the Execution Dashboard shows a list of reports that the user can access in order to analyze processing statistics and daily acquisition numbers (Figure 4-17).

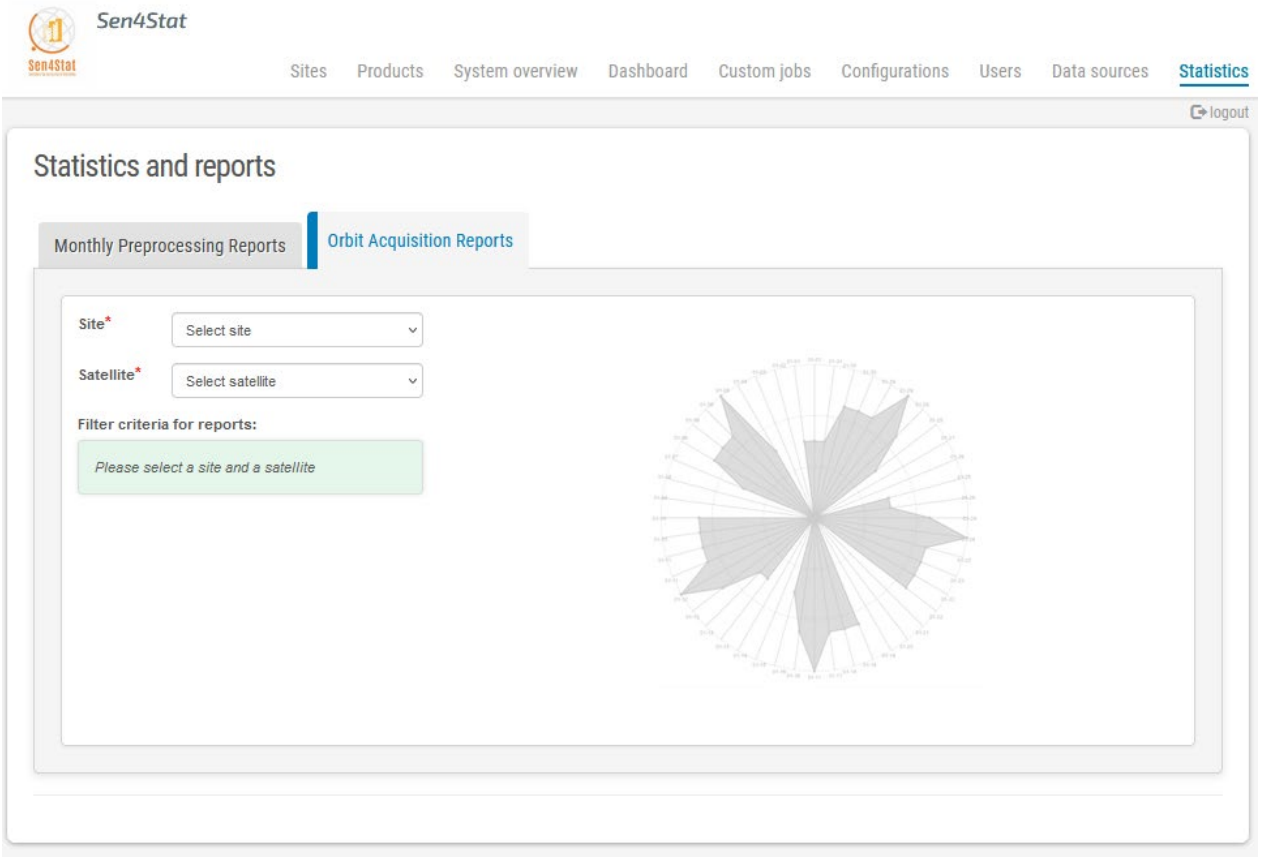


Figure 4-17. Statistics tab of the Execution Dashboard

- **Monthly Processing Reports**

As the name suggests, each report shows operations for each day in a month. These reports can be customized using the left side filters panel where the user can choose a set of months in a given year to produce that many charts, all for a selected site, satellite and orbit/s.

The reports will depict Acquisitions as a continuous line with values shown on the right vertical axis, while all other operations (Processed, Errors, Cloudy, …) will be depicted as discrete vertical bars of different colors with values shown on the left vertical axis (Figure 4-18).

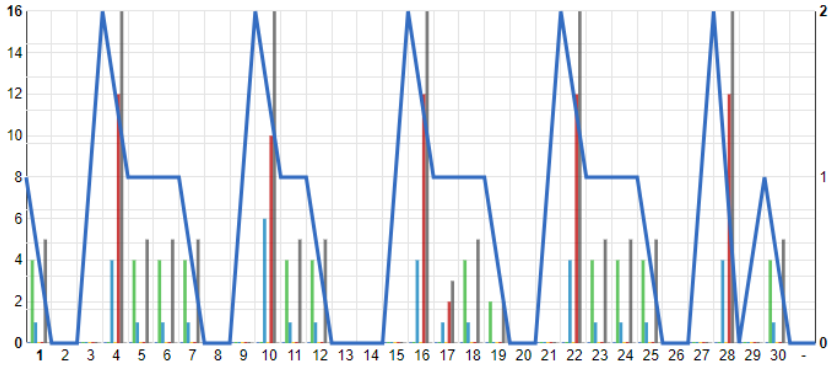


Figure 4-18. Bar charts supporting monthly processing report

For better readability and to facilitate comparison between different months, all reports will be generated as a stack of charts, each with 31 values on the horizontal axis (Figure 4-19). For months with less than 31 days, empty sequences will be added at the end.

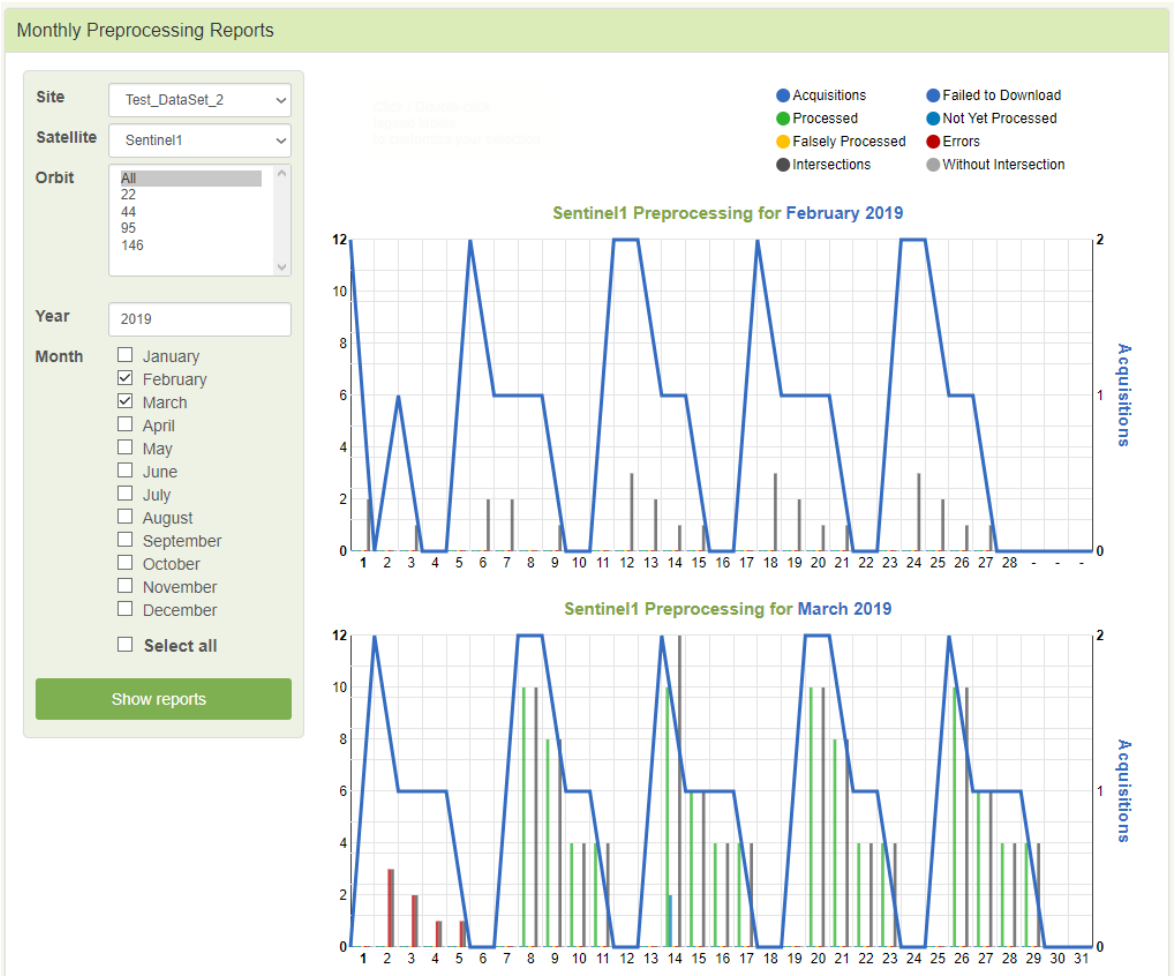


Figure 4-19. Example of monthly processing reports, allowing comparison between months

For all reports, a single legend is generated at the top of the stack (Figure 4-20). The legend is interactive, meaning that the user can customize the selection of values to be displayed, by clicking or double-clicking on its labels. The following image shows an example where the user only selected 3 operations: Acquisitions (blue line), Processed (green bar) and Errors (red bar). Selected operations are depicted in the legend as filled colored circles, while all other as empty circles. A message at the left side of the legend reminds the user that it's possible to customize the report.

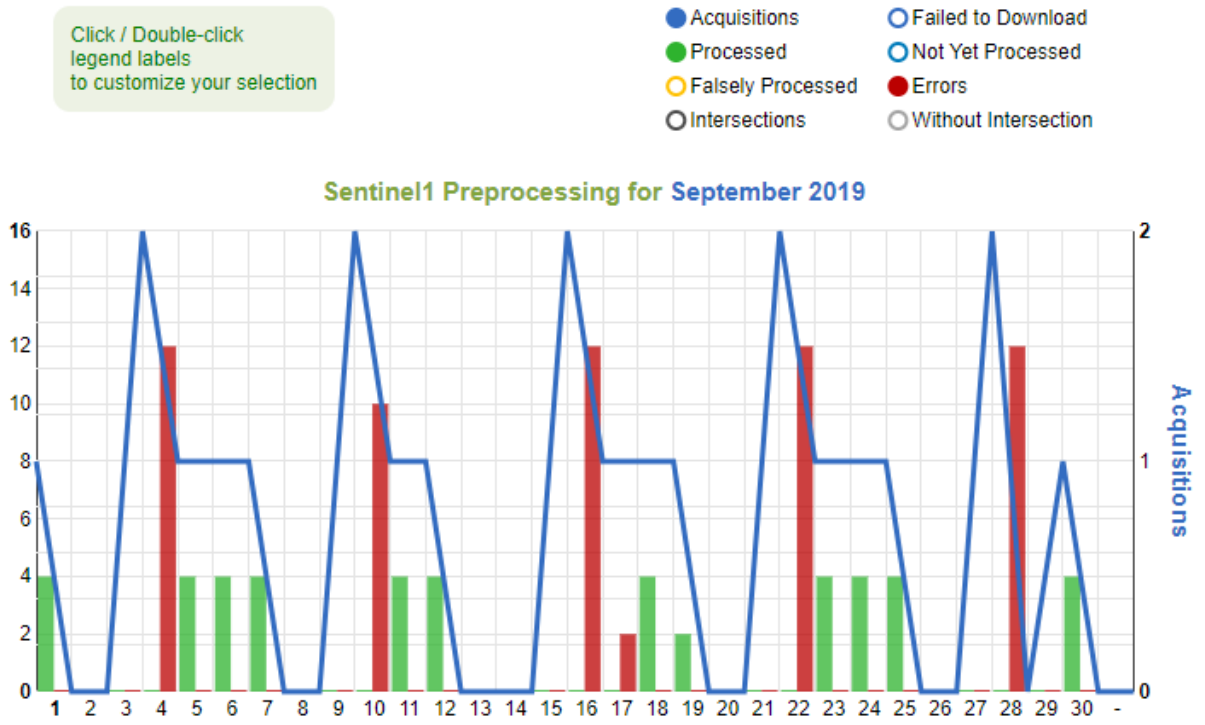


Figure 4-20. Illustration of the legend supporting monthly processing report

While hovering a report chart, a vertical selection bar is displayed which spans over an entire day to include all elements belonging to that day. This allows the user to easily identify the values corresponding to a single day. Alongside this selection bar, a textbox type tooltip is displayed with all the necessary information for the user to read its selection (Figure 4-21).

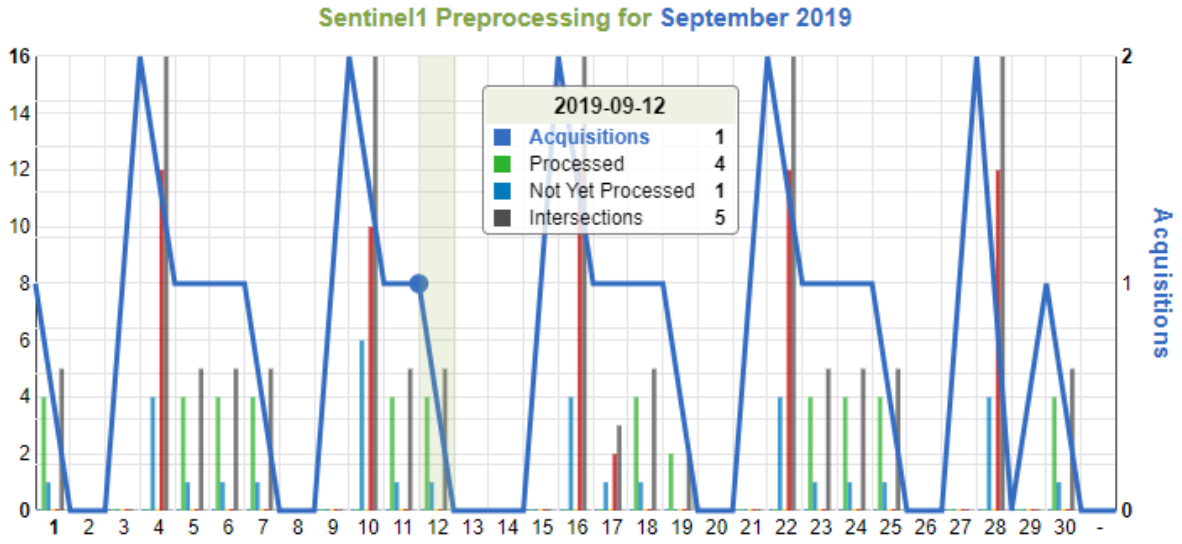


Figure 4-21. Illustration of the tools facilitating the understanding of monthly processing report

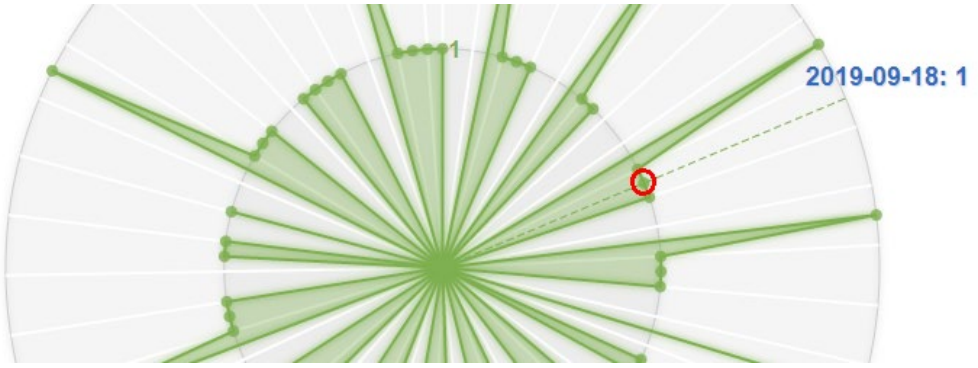
- **Orbit Acquisition Reports**

This report is generated as a "spider web" type chart, where as opposed to the previous type of reports (Figure 4-22). Only one operation is shown: "Acquisitions per day". This report can be customized using the left side filters panel where the user can choose either a single month in a given year, or a custom time window, to produce a single "spider web" chart for a selected site, satellite and orbit/s.

Figure 4-22. Illustration of an orbit acquisition report

The Acquisitions values are shown as concentric circles, as many as the maximum value for the selected time frame. The days are shown as labels on the circumference of the top-level circle. If the number of selected days exceeds 60, the number of labels shown on the circumference will be reduced so they don't overlap. The Acquisition values however will all be shown.

To identify which day corresponds to an Acquisition point, the user can hover that point, then a selection will be displayed with all necessary information (Figure 4-23). All labels on the circle will be hidden to avoid text cluttering. It is recommended to avoid selecting a large time frame, or the chart will become hard to read. The following example shows one-year worth of Acquisition data. It's still usable, but hard to read (Figure 4-23 - bottom).



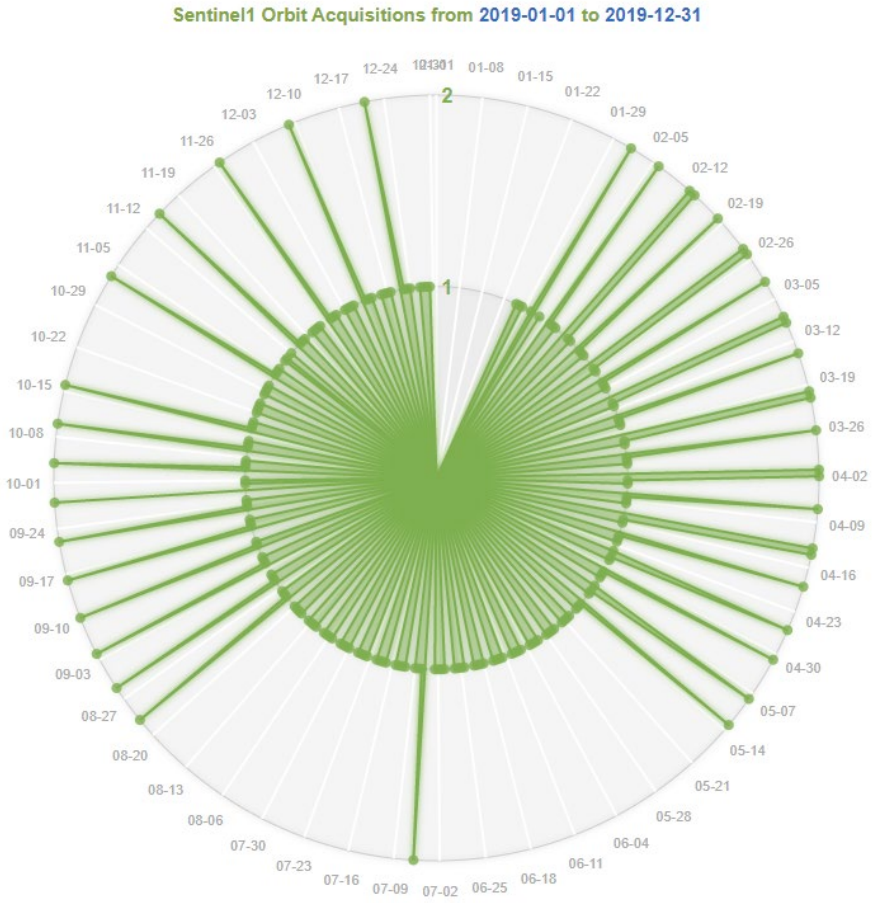


Figure 4-23. How to read the orbit acquisition reports

4.2.11 User management

The default admin user in the Sen4Stat system is the “sen4stat” and with the default password “sen4stat”. Nevertheless, this can be changed before system installation in the install package through the file Sen4XDistribution/install_script/config/install_config.conf and specifically by modifying the following fields:

```
[SEN4STAT]

…

USER_NAME=sen4stat

USER_MAIL=sen4stat@c-s.ro

USER_PASS=sen4stat

…
```

After installation, the default password of the sen4stat user can be changed from the login page, accessing the “Reset password” link (Figure 4-24):

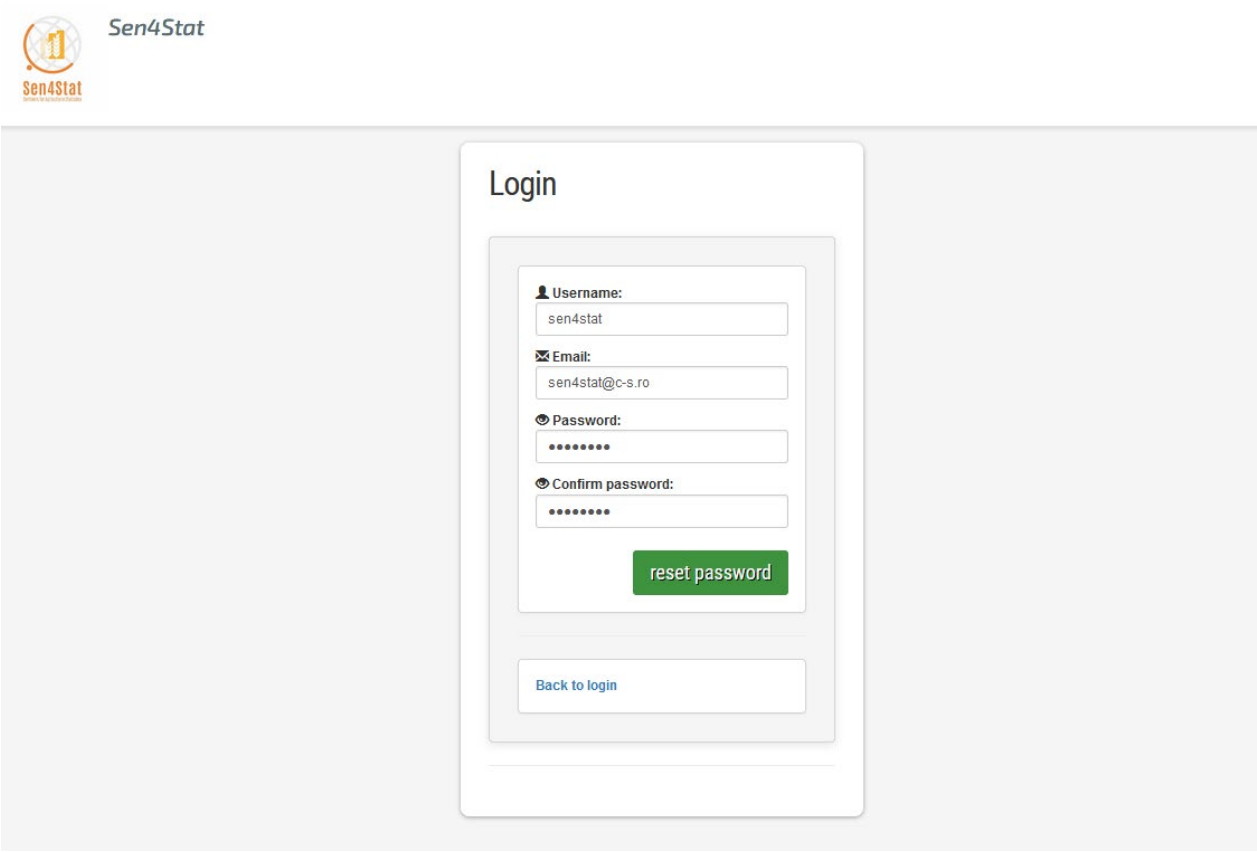


Figure 4-24. Resetting user password

In order to add other users or change the default admin one, the “Users” section from the web interface can be accessed (Figure 4-25). From this web interface, a new user can be added which can be an “admin” or regular “user”. For regular users, a list of sites to which he/she can have access is displayed. A regular user will have access only to those sites allocated during its creation or created by himself, to monitor them, to view products, create jobs but without the option to change the configuration or add other users. The admin users will have access to all sites, products, can add users and change configuration.

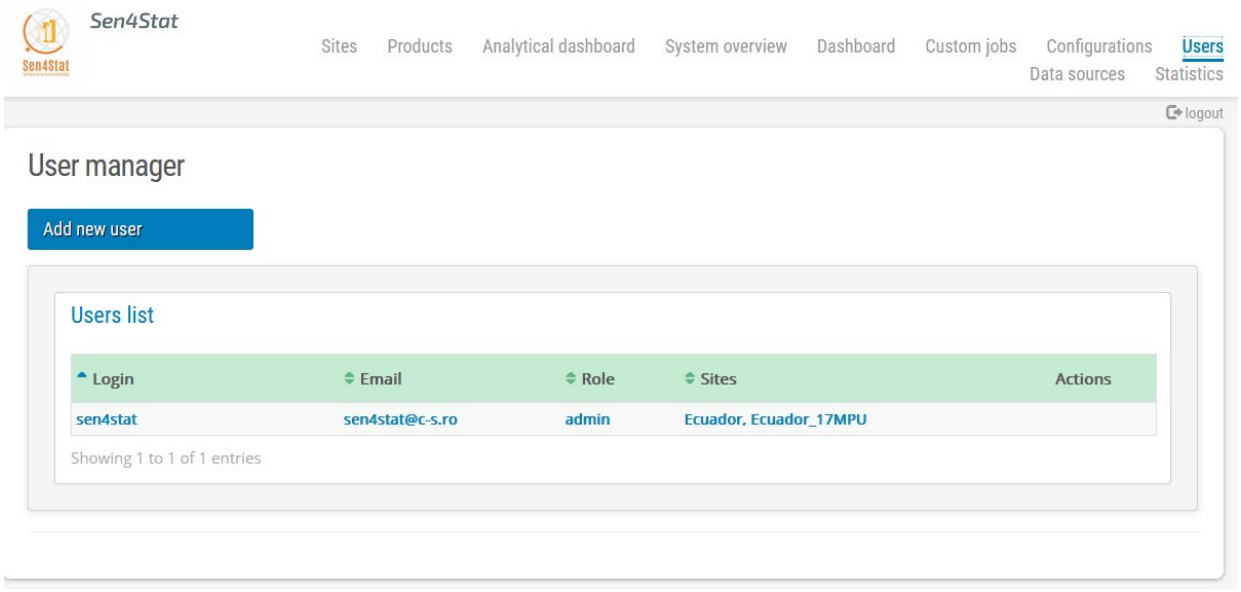


Figure 4-25. Users management

4.3 How to add scheduled jobs

The “dashboard” tab provides the possibility to define scheduled jobs, that is executions that are performed at certain moments in time (one-time executions or periodic executions) and that use the products available at that moment in time, according also to the configuration defined in the database.

The scheduled tasks can be added for all the processors excepting the L2A processor that is a built-in execution module. The scheduled tasks can be added from the “Dashboard” page of the Sen4Stat website.

The scheduled tasks will use the default parameters configured in the database for that processor. The default parameters used can be seen also in each tab of the processor from the “Dashboard” page (Figure 4-26).

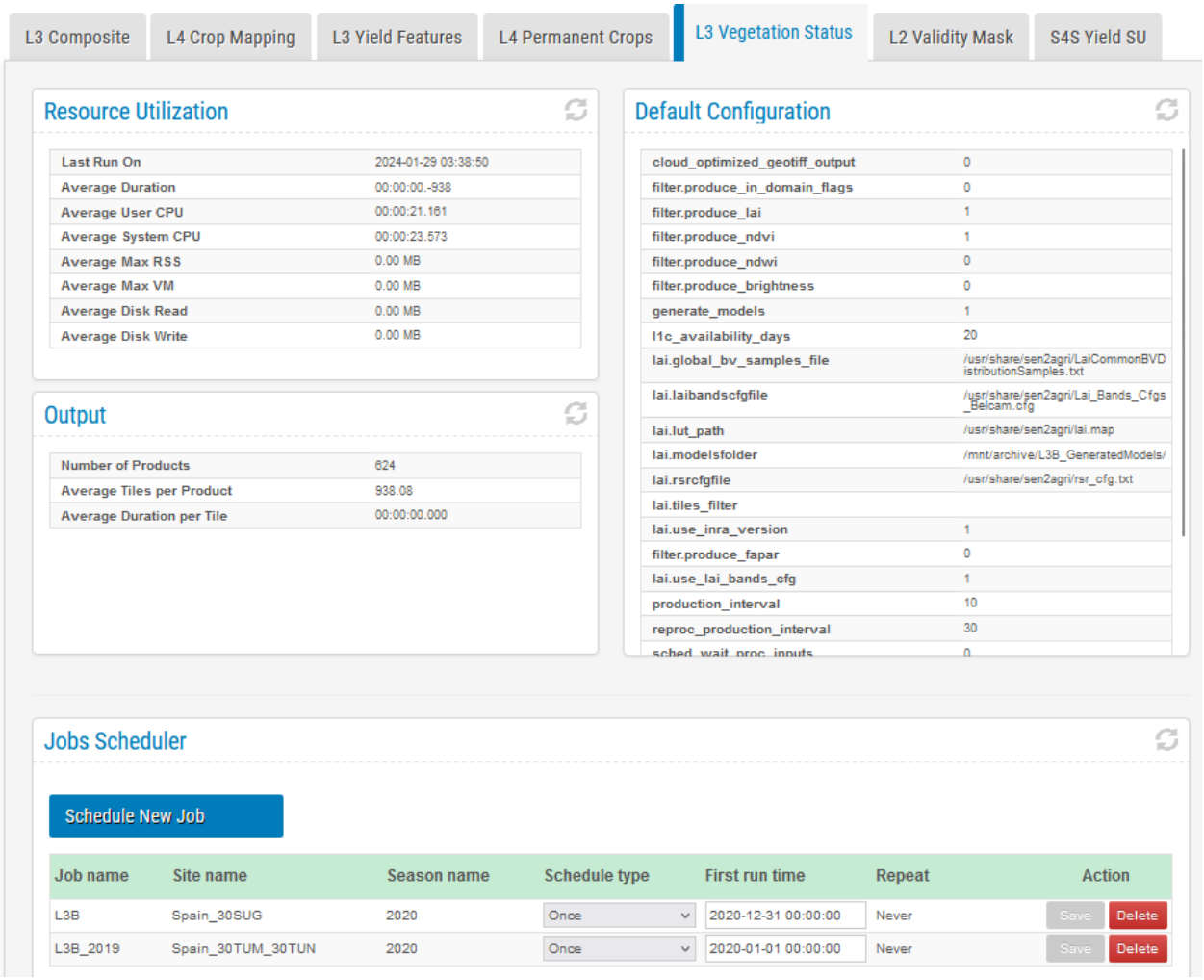


Figure 4-26. Processor parameters

Adding a scheduled task is similar for all processors.

In order to add a scheduled task, the following steps should be followed:

- Press the “Schedule New Job” button from the tab corresponding to the processor, section “Jobs Scheduler”.

  A new dialog will open to add a new scheduled job (Figure 4-27):

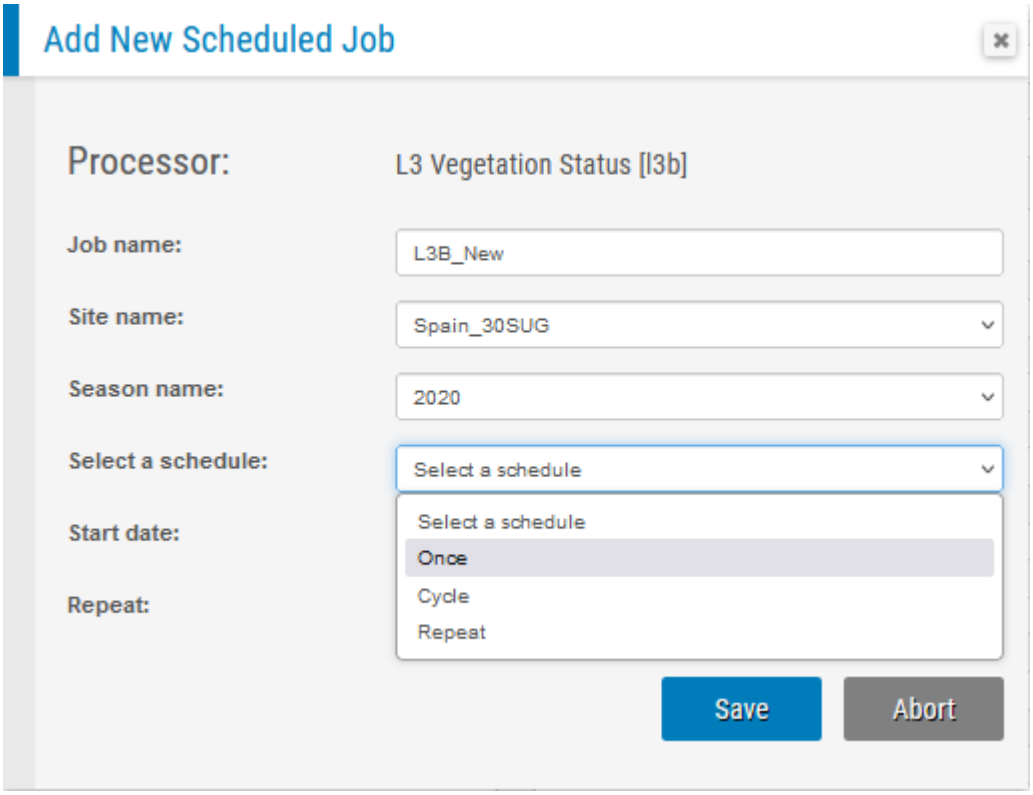


Figure 4-27. Adding a new scheduled job

- In the "Job Name" field add an identifier for the added task (ex. "myjob"). The job name should not contain space and should start with a capital letter.
- In the "Site name" field should be selected one of the existing sites.
- In the "Season name" field should be selected one of the site's season.
- From the "Select a Schedule" field, the following options are available:
    - "Once" – this option specifies to the scheduled task to run only once at the specified date. Selecting this option, a new field "Date" will be added to the entry where the user can select the date when the scheduled task should run (Figure 4-28):

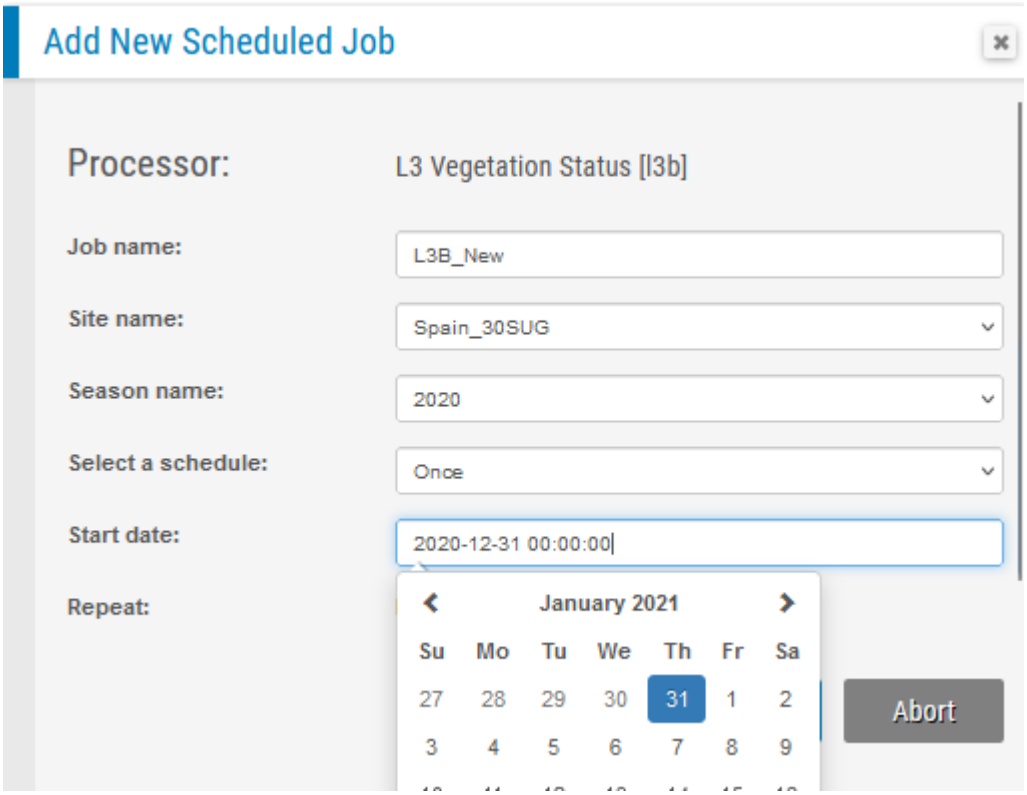


Figure 4-28. Adding "Once" scheduled type

- "Cycle" – this option allows to schedule a task at a certain date and then to reschedule the task after a number of specified days (Figure 4-29). For example, a configuration with the start date of 1st of March and "Repeat after" set to 4 days will execute the scheduler on 1st of March then on 5th of March, then on 10th of March and so on.

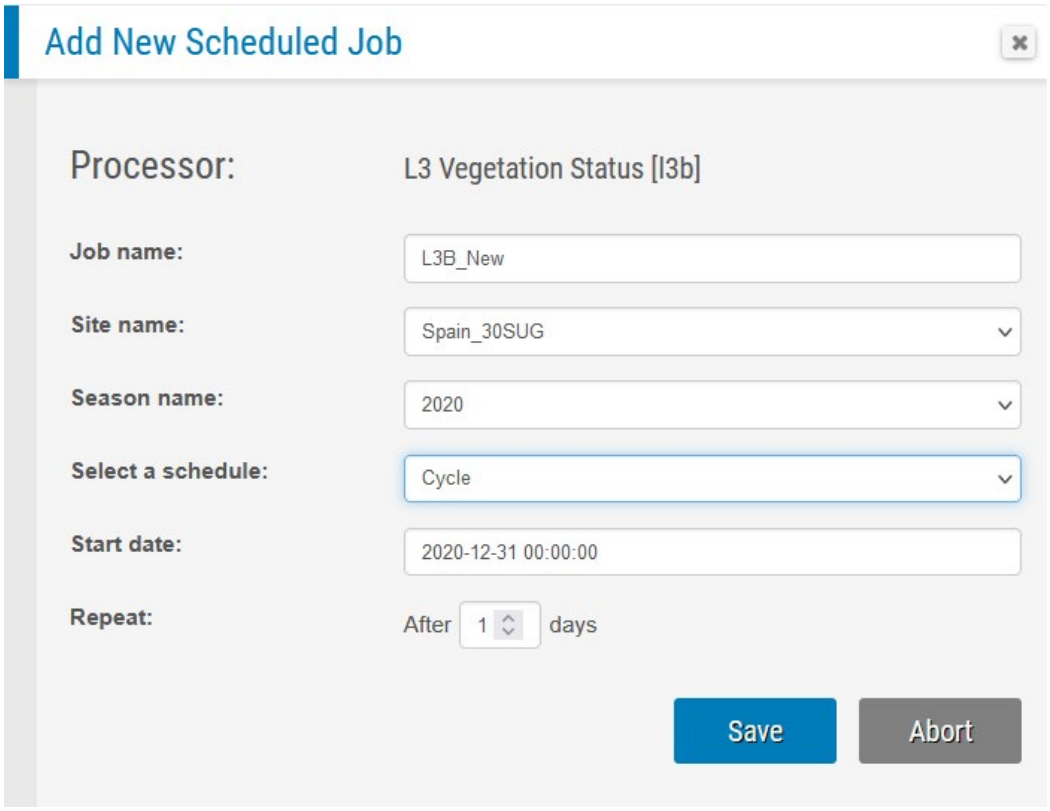


Figure 4-29. Adding "Cycle" scheduled type

- "Repeat" – this option allows to schedule a task at a certain date and then to reschedule the task at a certain day of the month (Figure 4-30). For example, a configuration with the start date of 1st of March and "On every" set to 4 will execute first on 1st of March, then on 4th of March, then on 4th of April, then on 4th of May and so on.

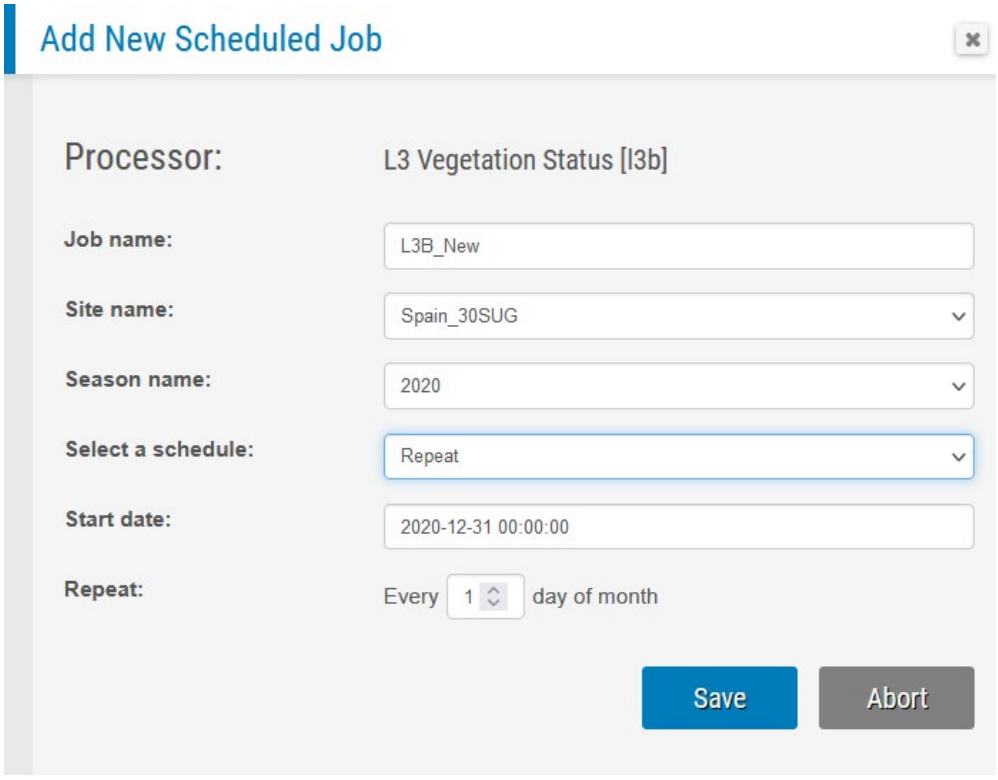


Figure 4-30. Adding "Repeat" scheduled type

In order to delete an added scheduled job, the "Delete" button can be used (Figure 4-31).

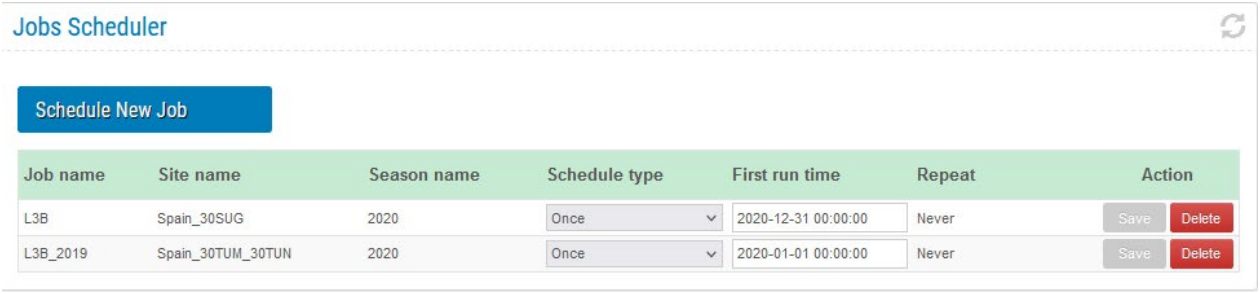


Figure 4-31. Delete scheduled job

4.4 Manual operations in Sen4Stat

The manual operations of the Sen4Stat system can be done by:

- Executing the processors from the command line using a specific Python script for each processor. This execution is available both in automated and manual mode installation of the system. Manually invoking the processors from the command-line does not use the SLURM scheduler. These manual executions from command line are intended for special scenarios like SNAP integration in which SLURM might not be available. When running in manual mode, some of the potential parallelism present in a job is not exploited. If needed, multiple manual jobs can be started at the same time;

- Executing the processors from the web interface. This execution is similar as with the command line invocation except that the execution will be performed using the orchestration mechanism that allows executions of parallel tasks and optionally on different machines and priorities (according the SLURM configuration). Though, this option is available only if the system was installed for an automated usage.

NOTE: When running the system components in the manual mode, make sure that the output location is writable by the current user;

Not all the processors can be run manually. The following sections list all processors and for those that can be run manually, they explain the set of operations to be executed.

4.4.1 Sentinel 1 pre-processing (S1 L2 processor)

The Sentinel 1 pre-processing of the SLC products is done automatically by the sen4stat-services module, by inspecting periodically the database and determining the products to be processed along with the previous products needed for computing the coherences.

The S1 pre-processing module cannot be run manually.

4.4.2 ERA5 Weather Downloader and Pre-Processor (ERA5 processor)

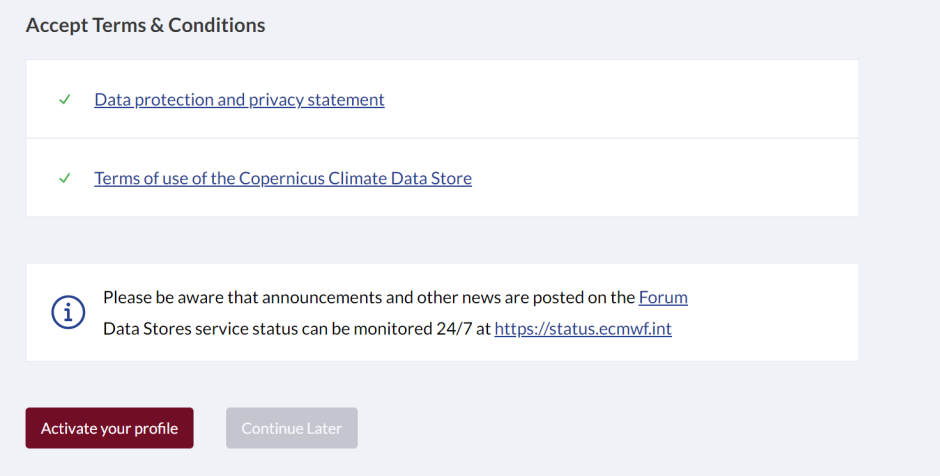
The ERA5 processor is in charge of the download and pre-processing of ERA5 data. The processor determines the area over which downloading the data based on the site defined in the system, downloads the corresponding ERA datasets via the Climate Data Store (CDS) API services and saves the weather data as NetCDF files to be used by the other processors (notably the Yield Features Extraction processor).

The ERA5 processor cannot be run manually; it is automatically run when the user has created an account (see below).

ERA-5 downloads prerequisites

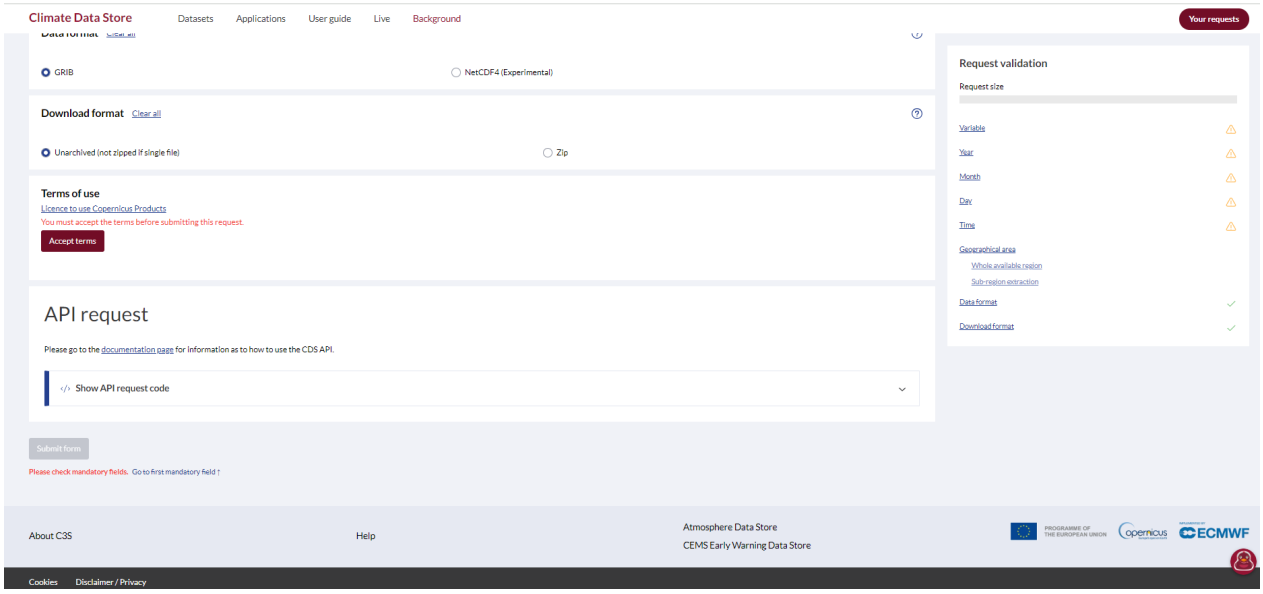
One of the inputs of the Yield Features extraction processor are the ERA-5 products. These products are downloaded from the https://cds.climate.copernicus.eu. As this is an external service, the user will have to perform the following steps in order to be able to automatically extract the ERA-5 products in the Sen4Stat system:

1. Create an account on https://cds.climate.copernicus.eu
2. Provide all the information requested and activate your profile:

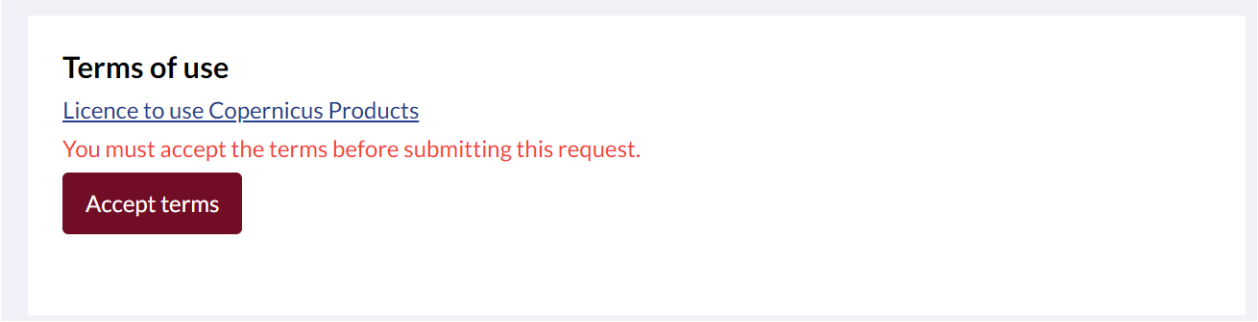


3. After the account is created and the user logs in, the following link should be accessed in order to accept terms and conditions, in the bottom of the page:

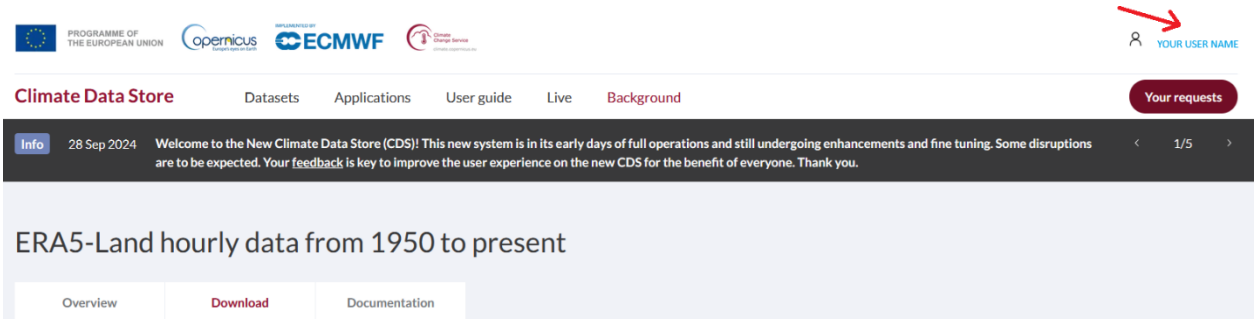
   https://cds.climate.copernicus.eu/datasets/reanalysis-era5-land?tab=download



4. Accept the terms of use:



5. Go to the user profile by clicking on the user name located in the upper right side of the page, near the "Your requests" button and then selecting "Your profile" option.



6. Once you access the user profile page, you can search for the section "Personal Access Token" that contains the access token associated to the user.
7. With the copied Personal Access Token, perform the following operations on the Sen4Stat server:
    a. Create a directory /var/lib/cdsapi/

       sudo mkdir -p /var/lib/cdsapi/

    b. Create a file .cdsapirc

       cd /var/lib/cdsapi/

       sudo touch ./.cdsapirc

    c. Edit the .cdsapirc file and add the following lines:

       url: https://cds.climate.copernicus.eu/api
       key: <Your Personal Access Token>

Once these operations are done, the ERA-5 downloading service should automatically perform the first queries to download the products (within one minute). The current requests can be seen on the https://cds.climate.copernicus.eu/requests?tab=all page, after logging in.

NOTE: The CDS API has a queuing mechanism for downloading the ERA-5 products and the requests are queued and handled according to the current load of https://cds.climate.copernicus.eu. Sometimes, during the day, these requests can be handled even within several hours while during the night they might be handled in less than one minute.

4.4.3 FMask masks extraction

FMask masks extraction is performed using the FMask processor which is used in the automatic mode by the fmask_launcher service. The fmask_extractor.py has the following parameters (Table 4-1):

Table 4-1. FMask extractor parameters

--working-dir	working directory
-- delete-temp	Specifies if temporary files should be deleted
--product-id	The id of the product as it appears in the database
--image-name	The FMask image name
-- cloud-dilation	Number of dilated pixels for cloud
-- cloud-shadow-dilation	Number of dilated pixels for cloud shadow
-- snow-dilation	Number of dilated pixels for snow
-t	FMask threshold
input	Input L1C/L1T product
output	Output FMask directory

4.4.3.1 Executing FMask from a terminal window

The FMask processor can be launched from terminal using a docker image for the fmask_extractor and providing parameters for FMask to be executed. The execution should be preferably made under the user sen2agri-service.

An example for the invocation of FMask can be found also in the next snippet:

```
docker run -v /var/run/docker.sock:/var/run/docker.sock --rm -u 1003:1003 --group-add 993 --name fmask_extractor_8677_OMOX7EUW sen4x/fmask_extractor:0.1.2 /usr/share/fmask/fmask_extractor.py --working-dir /mnt/archive/fmask_tmp --delete-temp --product-id 8677 --image-name sen4x/fmask:4.2 --cloud-dilation 3 --cloud-shadow-dilation 3 --snow-dilation 0 -t 17.5 --log-level info /mnt/archive/dwn_def/l8/default/test19/LC08_L1TP_191026_20190407_20190422_01_T1 /mnt/archive/fmask_def/test19/fmask/output/LC08_FMASK_191026_20190407_20190422_01_T1
```

4.4.4 In-situ dataset preparation

In-situ dataset used in Sen4Stat are crop data (crop type, crop yield, crop area, etc.) associated with polygons. The by-default assumption is that these data are collected by the NSO during their agricultural survey.

4.4.4.1 Input data

- **In-situ statistical dataset**

The dataset must be provided as a table with the fields listed in Table 4-2. If there are associated crops on the same parcel, several records in the in-situ statistical dataset may have the same "parcel_id". All columns must be present in the uploaded file but can be empty if no data is available in a given column, **except** for the "parcel_id", "year" and "crop_code" columns. These three columns **MUST** contain data.

Table 4-2. NSO in-situ statistical dataset (grey lines corresponding to fields that can be empty)

Field name	Role	Default value [format]
parcel_id	ID used to link the in-situ information to a polygon	[integer]
crop_code	Code of the crop	[integer]
year	Year of the survey	[integer]
planting_date	Start date of the crop period	[date]
harvest_date	End date of the crop period	[date]
yield_estimate	Yield estimation in kg/ha	[float]
yield_method	Method used to estimate yield (see Table 4-3)	[integer]
crop_density	Number of crop plants per unit area in nb/m^2	[integer]
crop_quality	Crop quality (see Table 4-4)	[integer]
irrigated	Presence (=1) or absence (=0) of irrigation	[boolean]
associated	Single crop (=0) or associated crop[3] (=1)	[boolean]

Table 4-3. Yield estimation methods

Code	Description
1	Eye estimation
2	Data provided by the farmer
3	Crop-cutting
4	Other methods

[3] Associated crops means more than one crop inside a plot. It could be different crops growing at the same time with superposition of the growth phases (mixed crops) or it could be successive crops. It could be also several turns of the same crops (i.e. rice).

Table 4-4. Crop quality

Code	Description
1	Normal growing conditions
2	Abnormal growing conditions (abandoned crop, young plantation that does not produce, climatic damage, erosion damage, disease, parasite damage)

- **In-situ geometries dataset**

In-situ geometries with a unique "parcel_id" for each entity (polygon for plot boundary) is provided by the NSO as a shapefile (Table 4-5). ***The next version of the system will also accept geometries in the form of points (not only polygons).***

Table 4-5. NSO in-situ shapefile

Field name	Role	Default value [format]
parcel_id	Unique ID for each in-situ polygon	[integer]
year	Year of the survey	[integer]
segment_id	Sampling segment ID	[integer]
stratum_sampling_id	Sampling stratum to which the parcel/segment belongs	[integer]

If the NSO provides in-situ data by segment, the boundaries of these segments must be also uploaded in the system (Table 4-6).

Table 4-6. Segment boundaries

Field name	Role	Default value [format]
segment_id	Segment ID	[integer]
stratum_id	Stratum ID	[integer]

- **Administrative limits**

Sen4Stat aims to provide statistics at national level but also at finer scales. To perform this spatial disaggregation, it is necessary to know the boundaries of each of the different administrative units. User must upload a shapefile with the boundaries containing at least one specific fields, a unique id for each unit (Table 4-7). The upload procedure is detailed in section 4.2.3.3.

Table 4-7. Administrative Boundaries

Field name	Role	Default value [format]
ID_2	Unique ID per administrative division (SU).	[integer]

- **Regional Historical Yield Data**

The regional historical data to upload is a .csv table referencing the region ID, the crop ID and the yield of at least ten years before the year during which the yield is to be estimated, as presented in Table 4-8.

Table 4-8. Yield historical data: the presented years are given as an example and can be changed according to the user. The table must contain at least 10 years before the year of estimation

Field name	Role	Default value [format]
SUid	Uni Unique ID per administrative division (SU).	[integer]
Crop_code	Code of the crop	[integer]
2010	Yield of the corresponding crop and SU in 2010	[float]
2011	Yield of the corresponding crop and SU in 2011	[float]
2012	Yield of the corresponding crop and SU in 2012	[float]
...		
2023	Yield of the corresponding crop and SU in 2023	[float]
2024	Yield of the corresponding crop and SU in 2024	[float]
2025	Yield of the corresponding crop and SU in 2025	[float]

4.4.4.2 Crop remapping sets

The crop mapping processor can optionally merge classes, either before (for the training) or after the classification. This remapping is configured using two database tables, "`crop_remapping_set`" and "`crop_remapping_set_detail`"., that are presented in Table 4-9.

Table 4-9. Crop remapping sets

Table name	Field name	Example
`crop_remapping_set`	`crop_remapping_set_id`	`1`
	`name`	`Default crop remapping set`
`crop_remapping_set_detail`	`crop_remapping_set_id`	`1`
	`original_code`	`1113`
	`remapped_code_pre`	`1113`
	`description_pre`	`Hard wheat`
	`remapped_code_post`	`111`
	`description_post`	`Wheat`

By default, the `original_code` to `remapped_code_pre` remapping which takes place during the in-situ preparation step (described in the following subsection) does not change the input crop code provided in the

in-situ data. This means that the only class remapping happens after classification by remapping classified pixel labels from `original_code` to `remapped_code_post` . To change this behavior and also remap classes prior to classification, the `remapped_code_pre` column must be modified in the `crop_remapping_set_detail` table.

- **Import and preparation step**

The following steps must be carried out for the import of the parcels dataset in the system and its preparation for the subsequent processors.

1. Copy the two input datasets described in the last point (in-situ statistical dataset and geometries) to the server disk drive, e.g. in /mnt/archive/s4s_parcels_upload
2. Make sure the dataset is readable by the sen2agri-service user by running:

```
sudo chown -R sen2agri-service: /mnt/archive/insitu/
```

NOTE: The parent directories also need to be accessible.

3. Find out the "site id":

```
sudo -u sen2agri-service psql sen4stat -c "select id, name from site"
```

4. Run the preparation script:

```
sudo su -l sen2agri-service

/usr/bin/data-preparation-s4s.py -s SITE_ID --year YEAR --parcels-geom
/mnt/archive/s4s_parcels_upload/parcels/sen_2018_polygons.shp --statistical-data
/mnt/archive/s4s_parcels_upload/parcel_statistics/sen_2018_in_situ.csv --working-path
/mnt/archive/s4s_parcels_import_tmp --classification-strata
/mnt/archive/s4s_parcels_upload/strata/classification.shp --yield-strata
/mnt/archive/s4s_parcels_upload/strata/yield.shp
```

Table 4-10. Parameters for the parcels' dataset import and preparation script

Parameter	Description	Type
config-file (-c)	Configuration file location	file
site-id (-s)	Site ID to filter by	int
year	Year (default current year)	int
parcels-geom	NSO in-situ geographical dataset	file
statistical-data	NSO in-situ statistical dataset	file
classification-strata	Strata used for classification	file
yield-strata	Strata used for yield estimation	file

Table 4-11. Strata shapefile columns

Column	Description	Type
stratum_id	Stratum identifier, should be 1, 2, ...	int
geometry	Stratum geometry	geometry

4.4.4.3 Output data

The preparation step generates a series of output files, which are used by the subsequent L4x processors. These outputs are described below.

- **Standardized declaration dataset with quality flags**

The first output is a set of three PostGIS tables. They contain the original parcel data, but also derived attributes and quality flags.

The first table contains the parcel geometries (Table 4-12).

Table 4-12. In-situ polygons geometry table

Field name	Role	Data type
parcel_id	Numeric identifier of the polygon, kept from the input	integer
year	Year of the survey	[integer]
segment_id	Numeric sampling segment identifier, kept from input	integer
stratum_sampling_id	Sampling stratum to which the parcel/segment belongs	[integer]
wkb_geometry	Polygon geometry, kept from input	geometry

The second table contains derived parcel attributes (Table 4-13).

Table 4-13. In-situ polygons attributes table

Field name	Role	Data type
parcel_id	Numeric identifier of the polygon, kept from the input	integer
geom_valid	Geometry validity flag	boolean
duplicate	Duplicate geometry flag	boolean
overlap	Overlapping geometry flag	boolean
area_meters	Polygon area, in meters, in the source projection	real
shape_index	Shape index (measure of the parcel circularity)	real
multipart	Whether the geometry has multiple parts	boolean
municipality_code	Municipality code	text
stratum_crop_id	Stratum identifier for crop classification	integer
stratum_yield_id	Stratum identifier for yield estimation	integer
pix_10m	Number of pixels in geometry when rasterized at 10m in a UTM zone projection	integer

The third table contains a copy of the input statistical data (Table 4-2).

- **Parcels buffer layers**

The second output is the parcels reprojected buffer layers (10m inner buffer). These buffers are reprojected in the WGS 84 / UTM zone {x} projections that correspond to all the UTM zones that cross the country. It only contains one field, the parcel_id of the parcel.

The parcels buffer layers:

- are shapefiles;
- are projected in WGS 84 / UTM zone {x} (as many as the number of UTM zones that cross the country);
- do not have necessarily the same number of lines (parcels) than the original parcels dataset;
- are generated in the folder /mnt/archive/insitu/site-name/{year}/.

- **Parcels raster layers**

The third output is the rasters that are produced for both S2 and S1 data by tile, with the parcel_id as value. Only the pixels that have their centroid located in the corresponding buffer layers (10m) have been assigned the parcel_id value of the parcel.

These layers:

- are .tif files;
- are produced by S2 tile;
- are projected in the WGS 84 / UTM zone {x} corresponding to the UTM zone of the S2 tile;
- value = parcel_id of the parcels;
- are generated in the folder /mnt/archive/insitu/site-name/{year}/.

4.4.5 Creating a composite (L3A) product

An overview of the processor is given in the section 2.2.3 and a more detailed description of the algorithms can be found in the related ATBD. This product will be formatted according to the Product Specification Document (PSD).

4.4.5.1 Processor's parameters

When a new L2A product is available or when a user requests a composite product, the L3A processor can be invoked with the parameters given in Table 4-14.

Table 4-14. L3A processor's parameters

Parameter	Description	Type
input	the list of products descriptors (xml files)	file list
res	resolution of the output image: if the value is 0, the resolution of the input image will be kept; if the value is 10 or 20, the image will be resampled at 10 or 20.	integer
outdir	the output directory	string

Parameter	Description	Type
syntdate	the date of the synthesis, which must be defined as the central date of the synthesis time window. The format of the date should be YYYYMMDD.	integer
half_synthesis	the half synthesis period (days)	integer
applocation	processors build folder	string
bandsmapping	the file that contains the mapping of the bands. It has a structure like: primary_sensor_name,secondary_sensor_name primary_band_index,target_resolution,secondary_band_index … For example, for Sentinel 2 and Landsat 8, having the Sentinel2 as a primary mission, we will have the following structure. Please note the -1 for the Sentinel2 bands for which there is no Landsat 8 band. SENTINEL-2A,LANDSAT_8 2,10,2 3,10,3 4,10,4 5,20,-1 6,20,-1 7,20,-1 8,10,5 9,20,5 12,20,6 13,20,7	string
preproc.scatcoeffs_10m	Scattering coefficients file for 10m res. This file is requested in S2 case ONLY	file
preproc.scatcoeffs_20m	Scattering coefficient file for 20m res. This file is requested in S2 case ONLY	file
tileid	Tile id (optional)	string
siteid	The site id, represented as 2 digits (optional)	string
lut_path	Path to the colour mapping table, for example /usr/share/sen2agri/composite.map (optional)	string
synth_date_sched_offset	Backward offset from the scheduler to determine the synthesis date, in automatic mode	int
weight.aot.maxaot	Max AOT value used in weight. Default 0.8	float
weight.aot.maxweight	AOT maximum weight (default 1)	float
weight.aot.minweight	AOT minimum weight (default 0.33)	float
weight.cloud.coarseresolution	Cloud coarse resolution (default 240).	int
weight.cloud.sigmalarge	Cloud sigma value for large clouds (default 10)	float

Parameter	Description	Type
weight.cloud.sigmasmall	Cloud sigma value for small clouds (default 2)	float
weight.total.weightdatemin	Date minimum weight (default 0.5)	float

4.4.5.2 Manual execution from Web Interface

The L3A Processor can be also executed manually from the web interface by accessing the "**Custom Jobs**" tab in the Web Interface and by selecting the "**L3A processor**" tab.

This editor allows the user running the L3A processor (i) by using only a subset of the available L2A input products and/or (ii) by running it with other parameters than the default ones defined in the automated processing.

When the editor is opened (Figure 4-32), the user has to select the site of interest and then, he has to define the following parameters:

- The set of L2A input products to be used: when the user selects his site of interest, the "Available input files" list is populated with the existing L8 and S2 L2A products available in the system. The user can select a subset of products and enable / disable one of the two sensors S2 and L8;
- The synthesis date ("syntdate" in Table 4-14);
- The half synthesis interval ("synthalf"in Table 4-14): the by-default value is 25 days. It can be modified;
- Spatial resolution ("res" in Table 4-14).

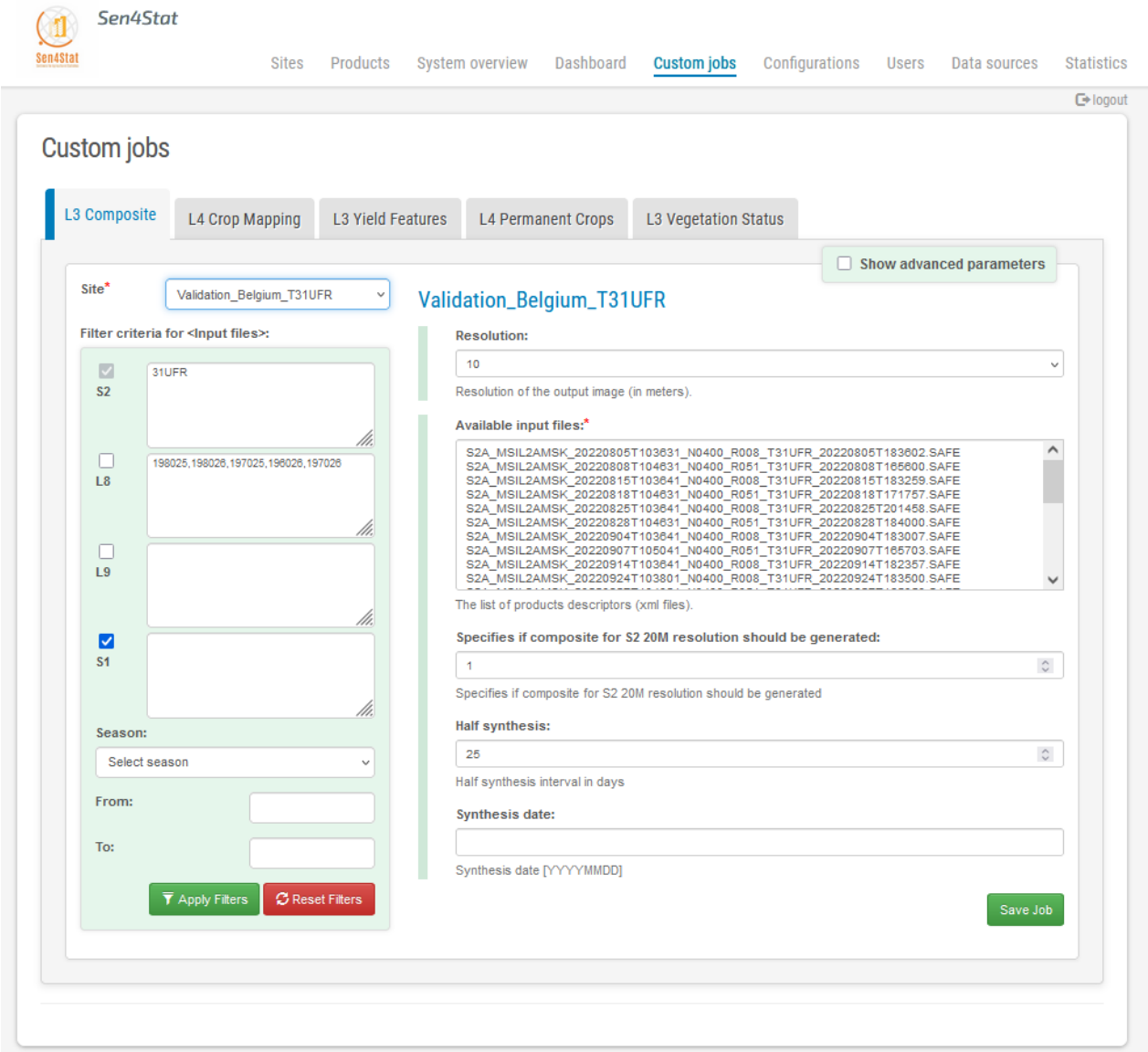


Figure 4-32. Manual execution of L3A processor from Web Site

The advanced parameters of the processor can be accessed ticking the "**Show advanced parameters**" box. The advanced parameters will be displayed as illustrated in Figure 4-33. Their definition can be found in *Appendix E9. All advanced parameters table*.

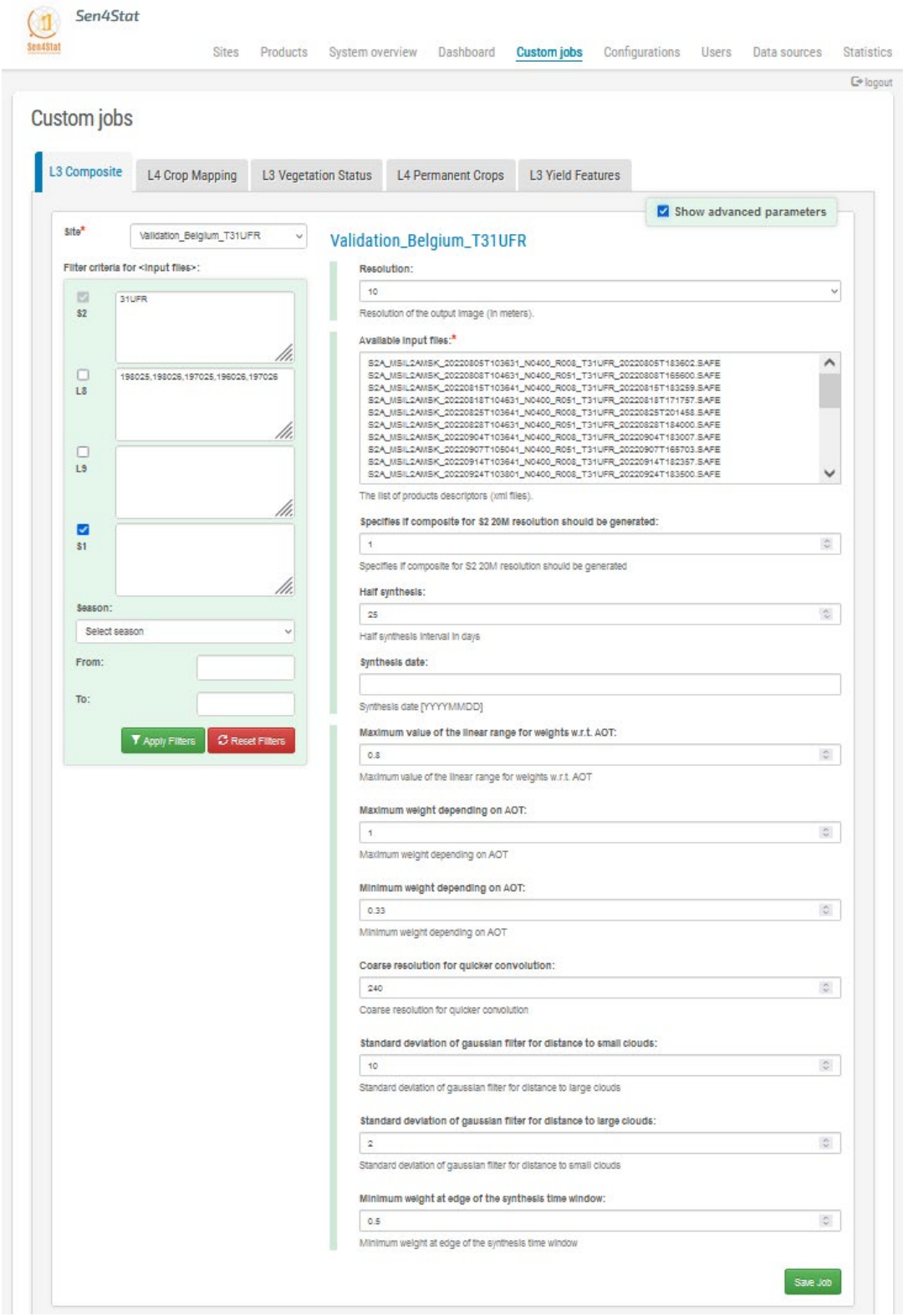


Figure 4-33. Manual execution of L3A processor from Web Site (advanced parameters)

4.4.6 Creating a Spectral Index and/or a Biophysical Indicator (L3B) product

As outlined in section 2.2.2, the NDVI, NDWI, Brightness LAI, FAPAR and FCOVER indicators are generated by the L3B processor. A more detailed description of the algorithms can be found in the corresponding ATBD document. These products are formatted according to the Product Specification Document (PSD).

4.4.6.1 Processors parameters

When a new L2A product is available or when a user requests a Spectral Index / Biophysical Indicator product, the L3B Biophysical Indicators processor can be invoked with the parameters given in Table 4-15. These parameters are read from the system database and they are prefixed with "processor.l3b.".

Table 4-15. L3B Spectral indices & Biophysical Indicators processor parameters

Parameter	Description	Type
cloud_optimized_geotiff_output	Specifies if outputs should be COG	bool
filter.produce_fapar	Produce FAPAR	bool
filter.produce_fcover	Produce FCOVER	bool
filter.produce_in_domain_flags	Produce domain input flags	bool
filter.produce_lai	Produce LAI	bool
filter.produce_ndvi	Produce NDVI	bool
filter.produce_ndvi	Produce NDWI	bool
filter.produce_brightness	Produce Brightness	bool
generate_models	Generate models (old LAI implementation only)	bool
lai.global_bv_samples_file	BV samples file	file
lai.laibandscfgfile	Config file with product bands to be used	file
lai.lut_path	LAI lut map	file
lai.modelsfolder	Generated models directory (old LAI implementation only)	file
lai.rsrcfgfile	LAI rsr file (old LAI implementation only)	file
lai.tiles_filter	Specifies the tile filter	String list
lai.use_inra_version	Use INRA version or old Sen2Agri LAI implementation (default, using INRA)	bool
lai.use_lai_bands_cfg	Config file with product bands to be used	bool

4.4.6.2 Manual execution from web interface

The L3B processor can also be executed manually from the web interface by accessing the "**Custom Jobs**" tab in the web interface and by selecting the "**L3B Spectral indices & Biophysical Indicators processor**" tab.

This tab allows the user to obtain a L3B product (i) by using only a subset of the available L2A input products and/or (ii) by running it with other parameters than the default ones defined in the automated processing.

When the editor is open (Figure 4-34), the user must select first the site and then, he has to define the following parameters:

- The site for which to process;
- Optionally, filter products based on season, satellite or specific dates;
- Set of L2A input products to be used: when the user selects his site of interest, the "Available input files" list is populated with the existing S2 and L8 L2A products available in the system; by default, all available images during the filtered time period are selected, but the user can select a subset of products and enable / disable one of the two sensors (S2 and L8);

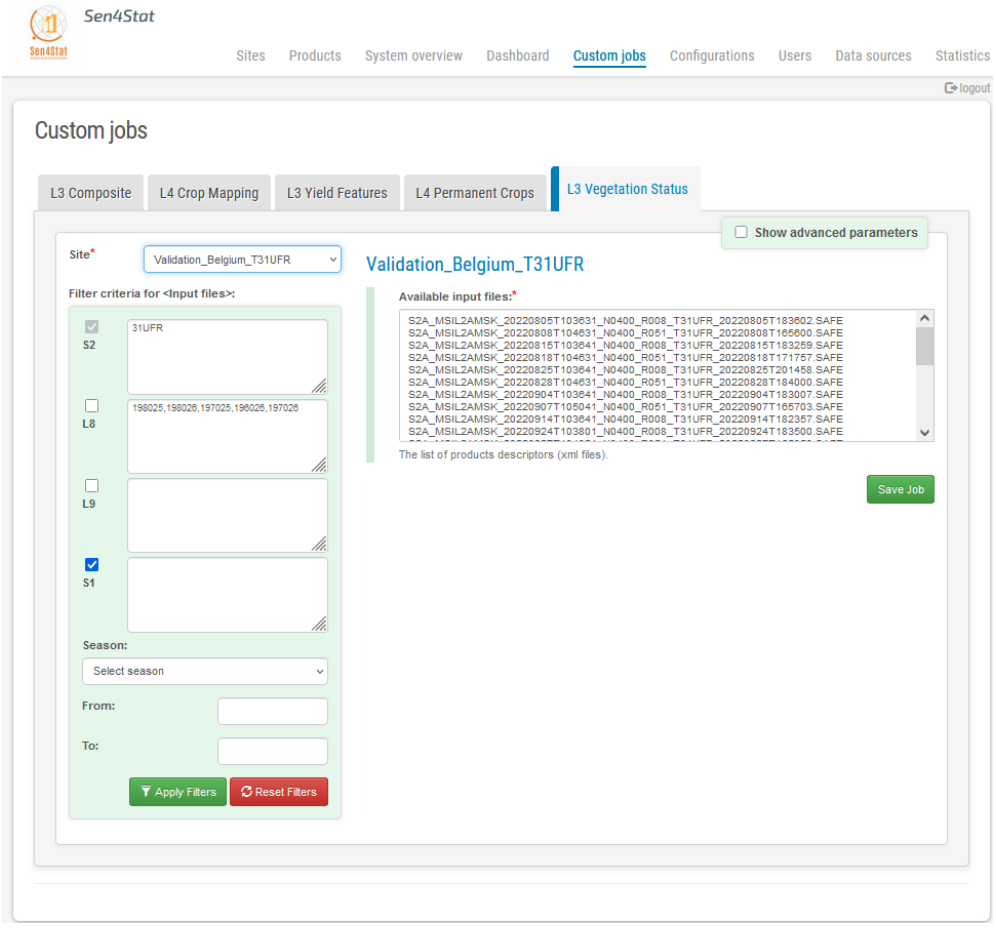


Figure 4-34. Manual execution of L3B processor from web interface

If a filtering of the indicators to be created is desired, the “Show advanced parameters” option can be selected (Figure 4-35).

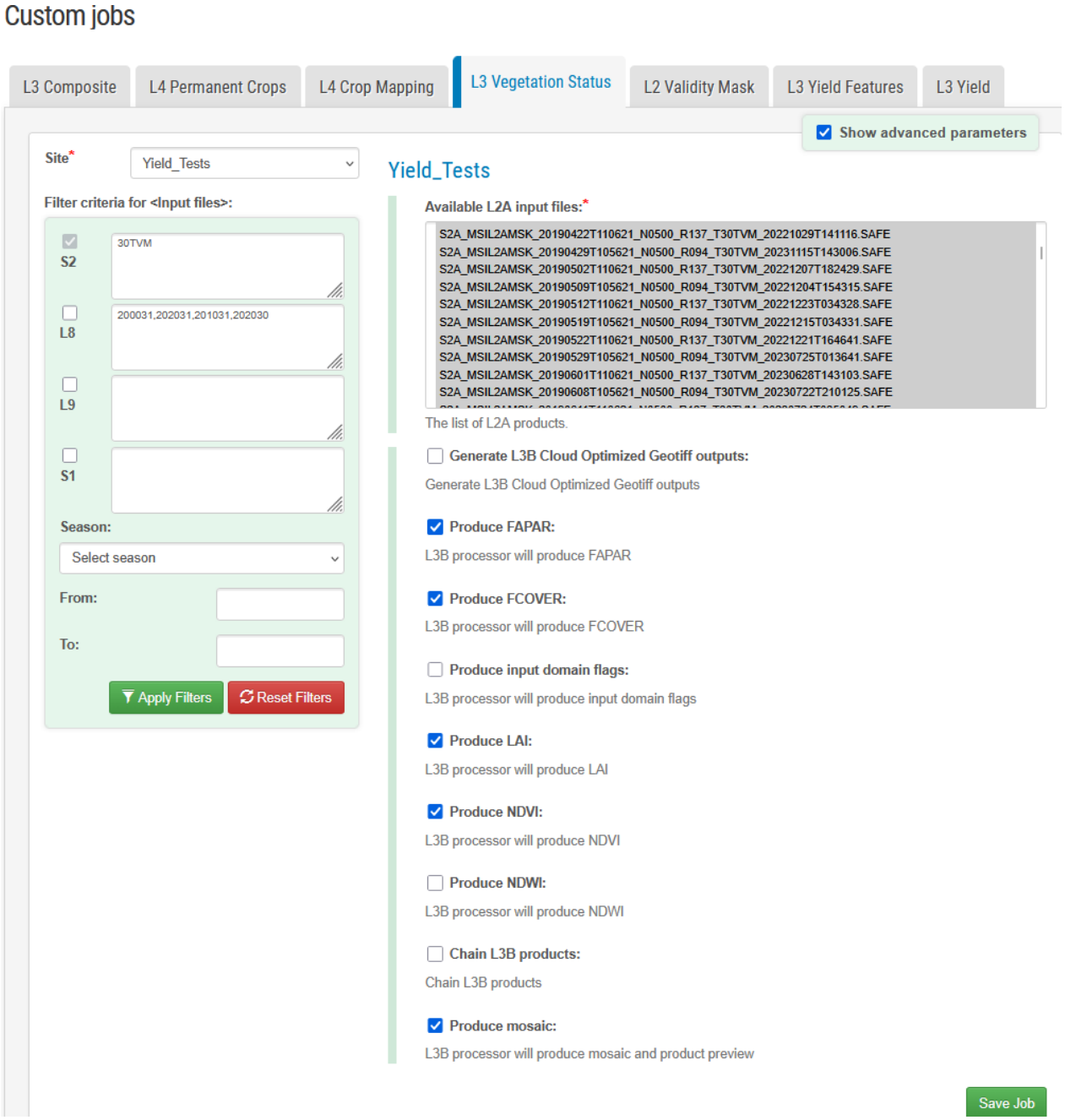


Figure 4-35. Selecting L3 indicators to be created

IMPORTANT NOTE : Since Sen4Stat version 4.0, the following L3 products are created:

- L3 LAI Biophysical Indicator
- L3 FAPAR Biophysical Indicator
- L3 FCOVER Biophysical Indicator
- L3 NDWI Vegetation Status
- L3 Brightness Indicator

These new product types are specific for each spectral index or biophysical indicator and contain only one tile, compared with the previous products that contained all the ordered indicators, containing all tiles in a day.

4.4.7 Creating a crop map

An overview of the processor is given in the section 2.2.5 and a more detailed description of the algorithms can be found in the corresponding ATBD document. This product is formatted according to the Product Specification Document (PSD).

4.4.7.1 Processor parameters

When the NSO in-situ datasets are uploaded in the system and the L2A, Backscatter and Coherence products are available for the defined season, the processor can be invoked from the Custom Jobs or as a Scheduled job, using the defined parameters in Table 4-16. These parameters are read from the system database and they are prefixed with “processor.s4s_crop_mapping”.

Table 4-16. Crop mapping processor parameters

Parameter	Description	Type
pix-ratio-min	Minimum crop to total pixel ratio	float
classifier	Classifier [rf/catboost]	string
pix-min	Minimum number of pixels of polygons	int
pix-best	Minimum number of pixels of polygons used for training	int
poly-min	Minimum number of polygons for crops	int
pix-ratio-hi	Minimum crop to total pixel ratio for strategy 1	float
pix-ratio-lo	Minimum crop to total pixel ratio for strategy 2	float
monitored-land-covers	Land cover class filter	string
monitored-crops	Crop class filter	string
smote-ratio	Synthetic sample ratio	float
sample-ratio-hi	Training pixel ratio for strategy 1	float
sample-ratio-lo	Training pixel ratio for strategies 2 and 3	float
rf.max-depth	Maximum depth of RF trees	int

Parameter	Description	Type
rf.min-samples	Minimum samples in RF tree nodes	int
rf.num-trees	Number of RF trees	int
catboost.iterations	Maximum number of trees	int
catboost.depth	Tree depth	int
catboost.early-stopping-rounds	Early stopping rounds	int
catboost.test-split	Test split fraction for early stopping	int
catboost.random-state	Random state for test split	int

4.4.7.2 Manual execution from terminal

The processor cannot be executed in this moment from a terminal but only from the Sen4Stat system web interface.

4.4.7.3 Manual execution from web interface

The crop classification processor can also be executed manually from the web interface by accessing the "**Custom Jobs**" tab and by selecting the "**Crop Mapping**" tab.

This tab allows the user to run the Crop Mapping processor using only a subset of available L2A input products by applying filters on the tiles and on the start and end dates of the monitoring period, as showed in Figure 4-36.

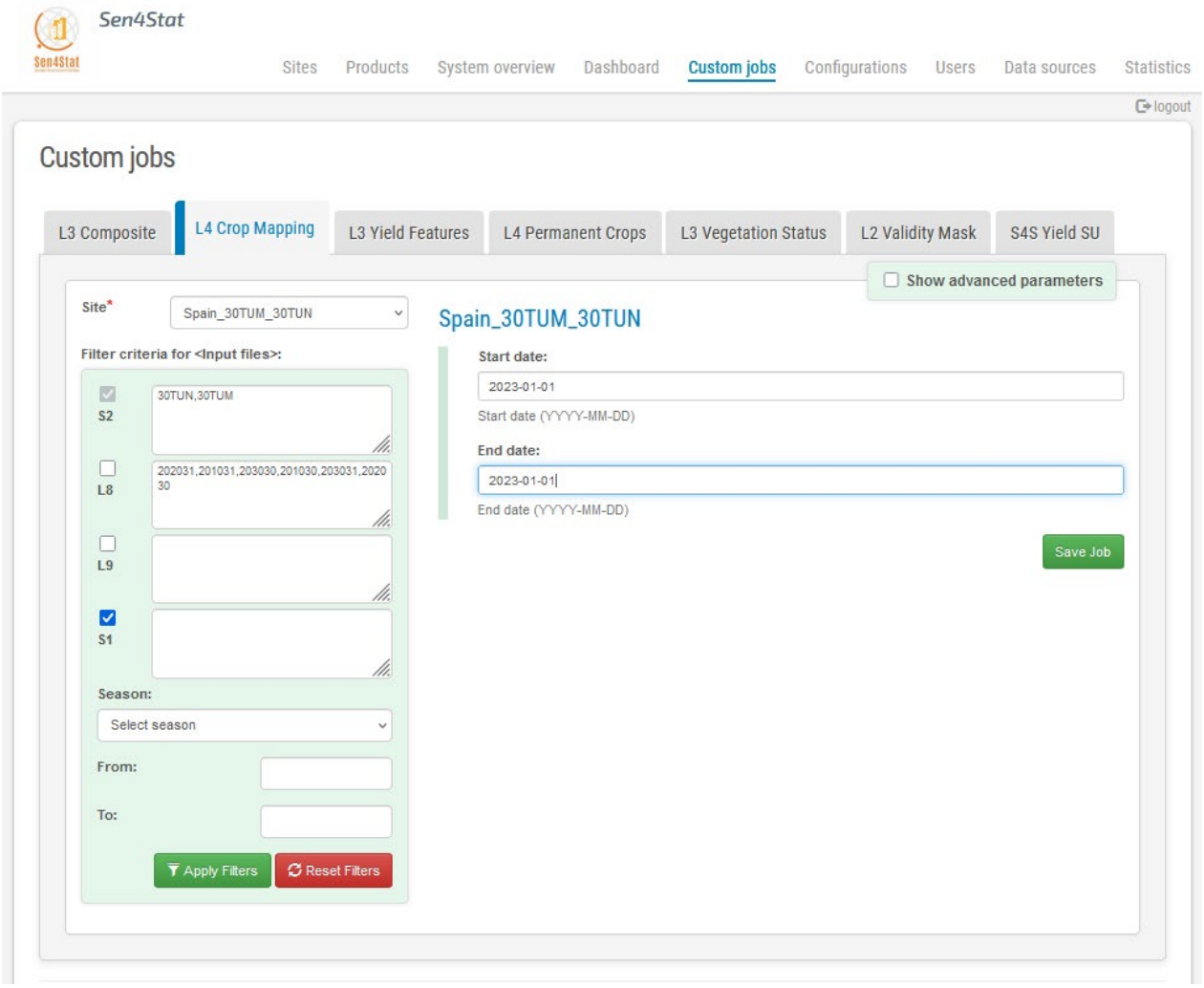


Figure 4-36. Manual execution of the Crop Mapping processor (random forest algorithm) from the web interface

By selecting the button "**Show advanced parameters**", the parameters described in the section 4.4.7.1 can also be modified (including the classification algorithm choice) before submitting the job (Figure 4-37).

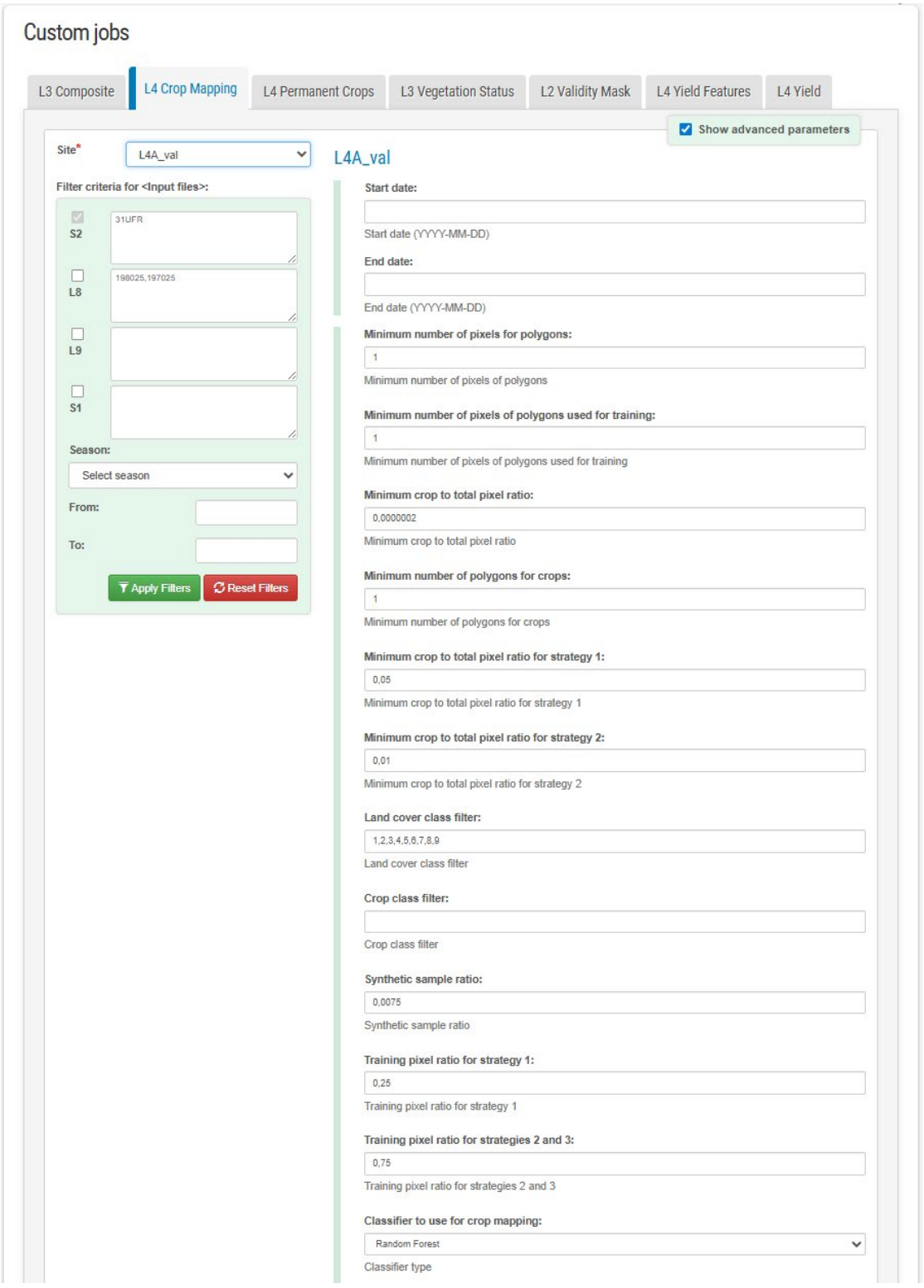


Figure 4-37. Manual execution of the Crop Mapping processor from the web interface by changing the advanced parameters

4.4.7.4 Mapping and validation outputs of the processor

Crop type mapping products for each site can be downloaded from the products tab. 7 types of files can be found in each L4A crop mapping product folder (Table 4-17), with the classification itself being split by tile. The classification report includes confusion matrices and class specific accuracy metrics generated using original and remapped classes.

Table 4-17. List of the files generated by the L4A crop mapping processor

File	Description	Type
Classified (tile specific)	Classified tile with remapped codes as labels	TIF
Classified_pre (tile specific)	Classified tile with original codes as labels	TIF
confusion_matrix	Confusion matrix and accuracy metrics of remapped classification	JSON
confusion_matrix_pre	Confusion matrix and accuracy metrics of original codes	JSON
polygons	Training and Validation polygons	GPKG
polygon-statistics	Number of training/validation pixels under each polygon	JSON
classification_report	Detailed classification Validation report	XLSX

4.4.8 Creating a permanent crop map using Broceliande

Disclaimer: Limitations of Broceliande in Accuracy and Resource Consumption

While we strive to provide the most accurate and efficient software experience with Broceliande, it is important for users to be aware of certain inherent limitations that may affect the software's performance and output.

- **Accuracy of Classification**: Currently, Broceliande utilizes a distinct processor for the Permanent Crops classification product. Efforts are underway to integrate this into the Crop Classification processor for improved efficiency and accuracy. However, due to the complexities involved in the classification process, there may be instances of incorrect or imprecise classifications. Users are advised to consider these potential inaccuracies when using the software for critical decision-making processes.
- **Resource and Time Consumption**: The operation of Broceliande, especially when dealing with large datasets or complex classification tasks, can be resource intensive. This may result in longer processing times and a substantial utilization of system resources. Users should plan accordingly, ensuring that sufficient time and system resources are allocated for the software to function effectively.

In this moment, the Permanent Crops classification product is created by a distinct processor but later on it will be included in the Crop Classification processor.

The processor can be executed from the Custom Jobs of the system or scheduled from the dashboard.

4.4.8.1 Processors parameters

When the NSO in-situ datasets are uploaded in the system and the L2A products are available for the defined season, the processor can be invoked from the Custom Jobs or as a Scheduled job, using the defined parameters in Table 4-18. These parameters are read from the system database and they are prefixed with "processor.s4s_perm_crop".

Table 4-18. Crop mapping product using Broceliande processor parameters

Parameter	Description	Type
broceliande-docker-image	The docker image for Broceliande	string
vec_field	The field in the in-situ data to be used	string

4.4.8.2 Manual execution from terminal

The processor cannot be executed in this moment from a terminal but only from the Sen4Stat system web interface.

4.4.8.3 Manual execution from web interface

The Permanent Crops processor can also be executed manually from the web interface by accessing the "**Custom Jobs**" tab and by selecting the "**S4S Permanent Crops**" tab.

This tab allows the user to run the Permanent Crops processor using only a subset of available L2A input products by applying filters on the tiles and on the start and end dates of the monitoring period, as showed in Figure 4-38.

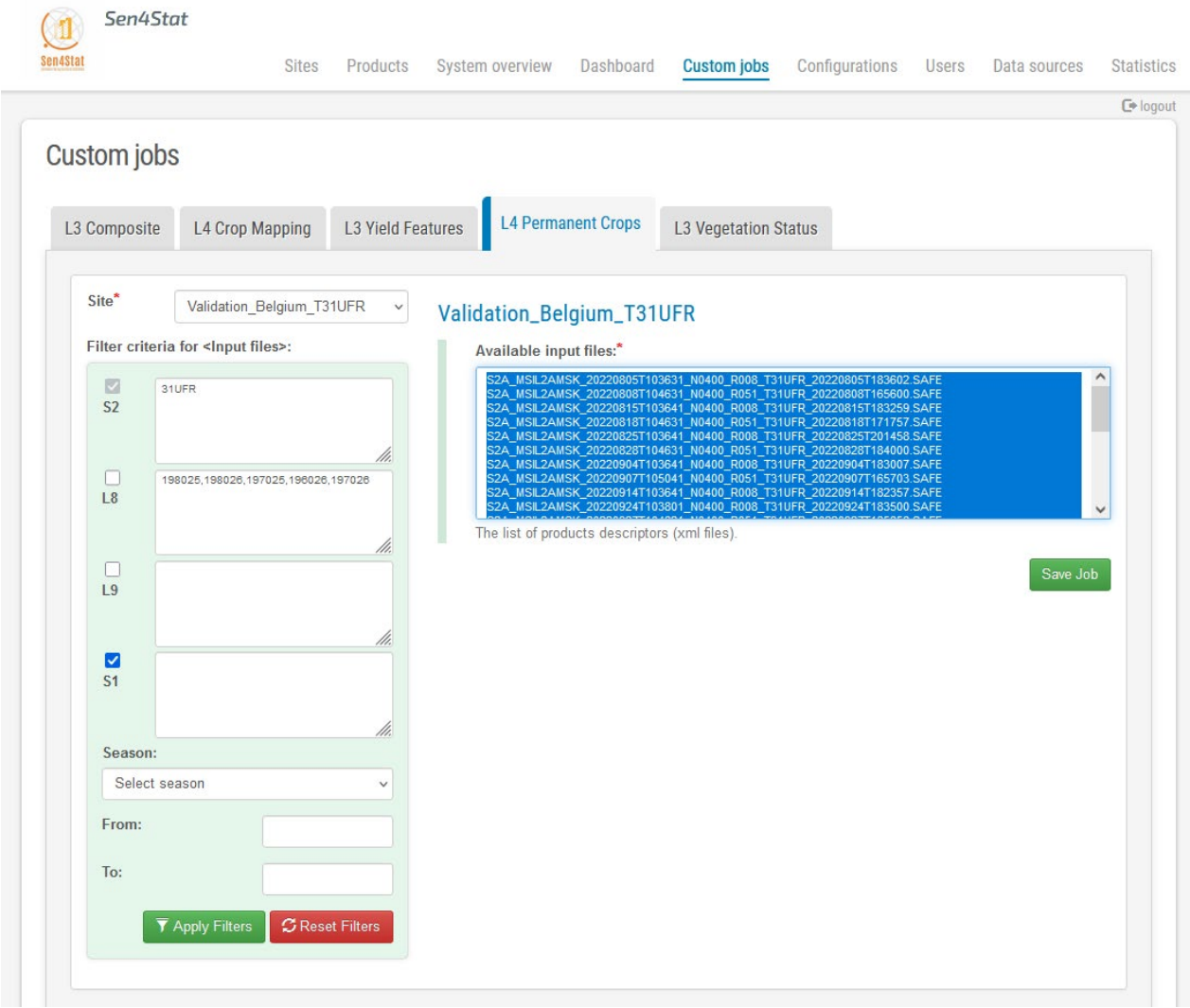


Figure 4-38. Manual execution of the Permanent Crops processor from the web interface

4.4.9 Creating a Yield Features product

An overview of the processor is given in the section 2.2.6 and a more detailed description of the algorithms can be found in the corresponding ATBD document. This product is formatted according to the Product Specification Document (PSD).

4.4.9.1 Processors parameters

When the NSO in-situ datasets, administrative boundaries and regional historical yield data are uploaded in the system and when the Sentinel-2 L2A time series and the ERA5 Weather products are available for the defined season, the processor can be invoked from the Custom Jobs or as a Scheduled job, using the defined parameters in Table 4-19. These parameters are read from the system database and they are prefixed with "processor.s4s_yield_feat".

Table 4-19.Yield features extraction processor parameters

Parameter	Description	Type
Parcel_id and crop type	The parcel id column name in the imported in-situ data.	string
safy_params_path	Path where the SAFY params file can be found for the site and configured year.	string
safy_params_upload_dir	The directory where the SAFY parameters file is uploaded from the website.	file

4.4.9.2 Manual execution from terminal

The processor cannot be executed in this moment from a terminal but only from the Sen4Stat system web interface.

4.4.9.3 Manual execution from web interface

The Yield Features processor can be executed manually from the web interface by accessing the “Custom Jobs” tab and by selecting the “L4 Yield Features” tab.

This tab allows the user to run the Yield Features processor using only a subset of available L3B input products by applying filters on the tiles and on the start and end dates of the monitoring period, as showed in Figure 4-39. User can enable or not the SAFY model for SAFY yield simulation. User can generate the feature at regional level by selecting “Generate yield SU”. A prerequisite for yield SU is historical yield of the SU, the administrative boundaries, and a L4 crop type product on the site for the appropriate year. A major feature (the trend) can only be computed for both parcel and SU level feature when the SU historical yield as well as the administrative boundaries are uploaded.

If the yield reference at parcel level is available in the uploaded in-situ data, the Yield Features at parcel level will be computed in priority. If the “Generate Yield SU” check box is selected, the Yield SU Features extraction will be performed if the about mentioned pre-requisites are fulfilled.

In order to differentiate between the Yield Features and Yield SU Features products, two types of products are created, and they can be easily identified by the tag S4SYIELDFEAT or S4SYIELDSUFEAT in their names.

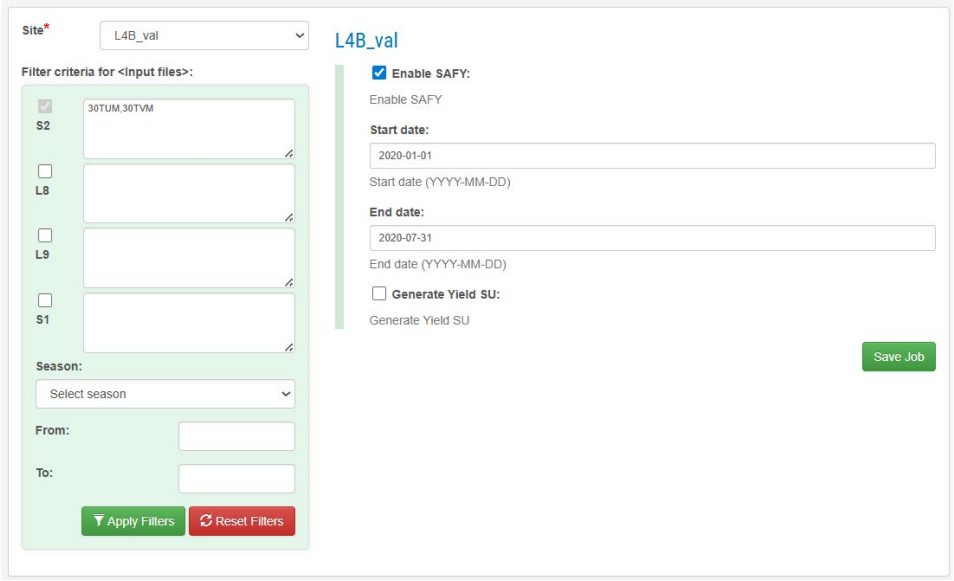


Figure 4-39. Manual execution of the yield feature extraction processor from the web interface

4.4.10 Creating a Yield Estimation product

An overview of the processor is given in the section 2.2.6 and a more detailed description of the algorithms can be found in the corresponding ATBD document. This product is formatted according to the Product Specification Document (PSD).

4.4.10.1 Processors parameters

When the Yield Features products are available, the processor can be invoked from the Custom Jobs or as a Scheduled job, using the defined parameters in Table 4-20. These parameters are read from the system database and they are prefixed with “processor.s4s_yield”.

Table 4-20. Yield estimation processor parameters

Parameter	Description	Type
Yield estimation algorithm	Selection of the regressor	string
Yield estimation selection type	Selection of the feature selection method	string
Maximum number of automatic features	Maximum number of features when the selected method is automatic	integer
Manual selection features	List of features selected by the user when the selected method in manual	string

4.4.10.2 Manual execution from terminal

The processor cannot be executed in this moment from a terminal but only from the Sen4Stat system web interface.

4.4.10.3 Manual execution from web interface

The Yield Estimation processor can be launched only if previously were created some Yield Features products. The user can select features generated on several seasons to populate the calibration set. The user has the option to choose either a set of Yield Features or Yield SU Features that can be used for calibration and for the yield estimation. If the Yield SU checkbox is checked, the selection from the "Available Yield SU Feature products" will be used in priority. It is recommended to select at least 5 years when working at SU level to ensure having enough observation in the calibration set. When more than one yield feature product is selected, the newest features product is used for the model while the older ones are used for calibration.

The advanced parameters allow to select the regressor and a feature selection method as explained in Figure 4-40.

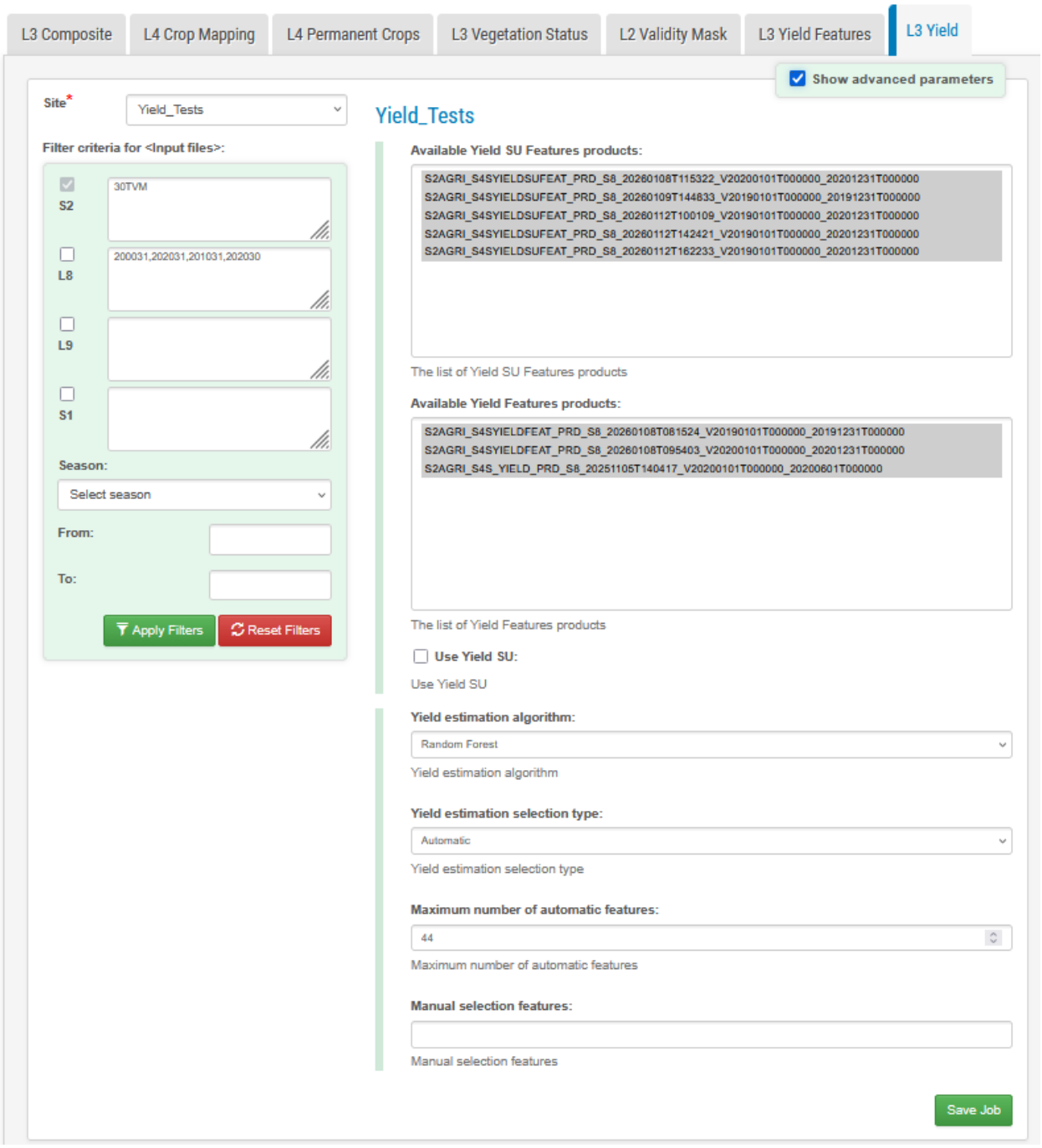


Figure 4-40. Manual execution of the yield estimation processor from the web interface

5 Maintenance of the system

5.1 Uninstall procedure

The following steps should be run as *root* user in order to uninstall the Sen4Stat system from the computer.

- Stop and disable the running applications:

```
# systemctl stop sen2agri-orchestrator sen2agri-scheduler sen2agri-executor sen2agri-http-listener sen2agri-services sen2agri-monitor-agent

# systemctl disable sen2agri-orchestrator sen2agri-http-listener sen2agri-monitor-agent
```

- Uninstall the application packages:

```
# yum remove sen2agri-processors sen2agri-downloaders-demmaccs sen2agri-app

# rm -fR /usr/share/sen2agri/sen2agri-services
```

- Stop and disable SLURM and MUNGE:

```
# systemctl stop Slurmd Slurmctld Slurmdbd munge

# systemctl disable Slurmd Slurmctld Slurmdbd munge
```

- Uninstall SLURM and MUNGE:

```
# yum remove Slurm Slurm-munge Slurm-Slurmdb-direct Slurm-sql Slurm-torque Slurm-sjstat Slurm-plugins Slurm-Slurmdbd Slurm-devel Slurm-sjobexit Slurm-perlapi Slurm-pam_Slurm munge-devel munge-libs munge
```

- Stop and disable MariaDB:

```
# systemctl stop mariadb

# systemctl disable mariadb
```

- Uninstall MariaDB:

```
# yum remove mariadb-server mariadb-devel mariadb
```

- Remove the PostgreSQL database:

```
# su -l postgres -c 'psql -c "drop database sen4stat;"'
```

- Uninstall OTB and GDAL 2.0:

```
# yum remove otb gdal-local
```

- Stop and disable Apache:

```
# systemctl stop http

# systemctl disable http
```

- Uninstall Apache and PHP:

```
# yum remove httpd php
```

- Uninstall dependencies:

```
# yum remove boost tinyxml tinyxml-devel qt qt5-qtbase qt-x11 fftw gdal geos
libgeotiff libsvm muParser opencv openjpeg2 openjpeg2-tools proj proj-epsg swig
qt5-qtbase-postgresql gsl cifs-utils
```

- Uninstall MAJA:

Follow the guidelines available in the MAJA user manual.

NOTE: Other applications on the system might depend on some of the packages above. Before continuing, please make sure to double-check the list of packages to be removed.

5.2 Update procedure

Upgrading versions of the Sen4Stat system is possible.

All upgrades can be performed using the same script Sen4StatDistribution/install_script/update.sh:

```
# cd <SEN4STAT_ROOT>/Sen4StatDistribution/install_script/

# sudo ./update.sh
```

The MAJA installation is not performed automatically by the system and should be performed manually by the user according to the instructions described in the MAJA documentation.

Nevertheless, after a reinstall of MAJA, the following operations should be performed in the Sen4Stat system in order to use the new MAJA version:

```
$ rm -rf /mnt/archive/gipp_maja/*

$ cp <SEN4STAT_ROOT>/Sen4StatDistribution/gipp_maja/* /mnt/archive/gipp_maja

$ psql -U admin sen4stat
```

```
sen4stat=# update config set value = '/opt/maja/3.2.2/bin/maja' where key
='demmaccs.maccs-launcher';
sen4stat=# \q
```

5.3 Recover runs

5.3.1 SLURM issues

SLURM is used in the Sen4Stat system for executing the job steps of almost all processors. This is why it is an important component of the system but it might fail sometimes, especially when no disk space errors occur. In order to reset SLURM, restarting its services might solve the issue:

```
$ sudo systemctl restart slurmd slurmdbd slurmctld mariadb
```

Also, if the SLURM gives the following error when executing “srun”:

```
srun: Required node not available (down, drained or reserved)

srun: job 3257 queued and waiting for resources
```

then the following command should be executed, followed by a restart of the SLURM services, as above:

```
$ sudo -u sen2agri-service scontrol update NodeName=localhost State=RESUME
```

5.3.2 Download issues

Sometimes, datasources might return errors while querying or downloading products or some products might become available later than other for the same date. As the downloading process is an iterative one and there exist also a buffer in days for querying for new products, some of them might still not be available at a given moment of time. To force re-querying for these products, the following commands can be executed:

```
$ psql -U admin sen4stat -c "update config set value = true where key = 'downloader.s1.forcestart'"

$ psql -U admin sen4stat -c "update config set value = true where key = 'downloader.s2.forcestart'"

$ psql -U admin sen4stat -c "update config set value = true where key = 'downloader.l8.forcestart'"
```

5.3.3 Pre-processing issues

When pre-processing issues occur, most common being disk full errors but also unavailability of external services like for downloading SRTMs, the level 1 products should be reset in order to be reprocessed. To do this, the corresponding entries for the failed products should be removed from the table “l1_tile_history” and the statuses in the “downloader_history” table from the database should be reset to “Not processed” as follows:

```
$ psql -U admin sen4stat -c "delete from l1_tile_history where downloader_history_id in (select id from downloader_history where status_id = 6 and site_id = <YOUR_SITE_ID>)"

$ psql -U admin sen4stat -c "update downloader_history set status_id = 2 where status_id = 6 and site_id = <YOUR_SITE_ID>"
```

5.3.4 Resume a failed processing job

The most frequent errors in the Sen4Stat are due to the disk full errors. If an execution job failed due to this error, the following command can be executed in order to resume the job, if enough disk space was made available:

```
$ job_operations -j <JOB_ID> -o reset_failed
```

Appendix A - Additional tools available in the system

Appendix A.1 Sen4Stat Executor parameters

The Sen4Stat Executor is based on the Sen2Agri Executor and is in charge of the execution of the processors applications, using SLURM for executing these applications.

Parameter Name	Description	Configuration Method
Executor IP Address	The address where the executor is found	Sen4Stat Configurator
Executor Port	The port where the executor is found	Sen4Stat Configurator

Appendix A.2 Sen4Stat Persistence Manager parameters

The Sen4Stat Persistence Manager offers an API for other components to access the database (configuration but also the execution tables).

Parameter Name	Parameter Value	Description	Configuration Method
Database/HostName	localhost	The name of the server where the database is hosted	Config file /etc/sen2agri/sen2agri-persistence.conf
Database/DatabaseName	sen4stat	The name of the Sen4Stat database	Config file /etc/sen2agri/sen2agri-persistence.conf
Database/UserName	Admin	The username used to connect to the database	Config file /etc/sen2agri/sen2agri-persistence.conf
Database/Password		The password used to connect to the database	Config file /etc/sen2agri/sen2agri-persistence.conf

Appendix A.3 Sen4Stat Services Download parameters

The Sen4Stat Services Download module is a component that manages the download of level-1 products (Sentinel-1, Sentinel-2 and Landsat8) and Sentinel-2 level-2 product from Copernicus DAS and USGS or from a local store. Also, if configured accordingly, the ESA level-2 products can be directly downloaded from Copernicus DAS or fetched from a local repository.

Parameter Name	Parameter Value	Description	Configuration Method
Write Directory for S2	/mnt/dwn_def/s2	The directory where the downloaded Sentinel-2 products will be saved	Sen4Stat Configurator
Write Directory for L8	/mnt/dwn_def/l8	The directory where the	Sen4Stat Configurator

		downloaded Landsat-8 products will be saved	
Write Directory for S1	/mnt/dwn_def/s1	The directory where the downloaded Sentinel-1 products will be saved	Sen4Stat Configurator
site	Ex: "1";"NLD_test" "2"; "Italy"	The sites defined in the database, from which the downloaders are searching online products; the polygon of each site is projected in WGS84	Sen4Stat database, site table; this table is prefilled and the manual editing in the database is not advised.
shape_tiles_s2	Shapes defined for the acquisition plan	The shapes defined for the acquisition plan will be intersected with the polygon of each site, and the resulting shapes will be interrogated by the downloader	Sen4Stat database, shape_tiles_s2 table; this table is prefilled and the manual editing in the database is not advised.
shape_tiles_l8	Shapes defined for the acquisition plan of Landsat 8	The shapes defined for the acquisition plan will be intersected with the polygon of each site, and the resulting shapes will be interrogated by the landsat downloader	Sen4Stat database, shape_tiles_l8 table; this table is prefilled and the manual editing in the database is not advised.
satellite	"1";"sentinel2" "2";"landsat8" "3"; "sentinel1"		Sen4Stat database, site table; this table is prefilled and the manual editing in the database is not advised
downloader.use.esa.l2a	true	Specifies if the ESA L2A products should be downloaded from SciHub instead of downloading L1C	

		and pre-process them with MAJA.	

The Sen4Stat Services Download is able to query for the list of products available from one source and to download the products from another source. For example, the list of available products for a season can be queried from Copernicus DAS but the download of the products to be performed from a local store. The "datasources" table contains the data sources used for querying and for downloading level-1 or ESA level-2 products. These parameters can be edited from the "Datasource" tab in the system web interface as described in section 4.2.1. The structure of the table is the following:

Column name	Description
satellite_id	The satellite ID for which the datasource is configured
name	The name of the datasource
scope	The scope of the datasource and may have the following values: 1 = query, 2 = download, 3 = query & download
fetch_mode	The product extraction mode: 1 - OVERWRITE - Products are downloaded from the remote site and the corresponding local product, if exists, is overwritten 2 - RESUME - Products are downloaded from the remote site and, if a corresponding local product exists, the download is resumed from the current length of the local product 3 - COPY - Products are copied from a local (or shared) folder into the output folder. No remote download is performed 4 - SYMLINK - Only a symlink to the product file system location, into the output folder, is created. No remote download is performed 5 – Direct link to product – The path of the product is directly inserted into the database. No remote download or symbolic linking is performed
username	The user used for the remote connection
password	The password used for the remote connection
download_path	The path where the products are downloaded (should not be changed)
specific_params	Specific parameters for the datasource (should not be changed)
maximum_connexions	Maximum connections to be used at a moment to the datasource
local_root	The local root repository in the case of the products already downloaded in a local store
enabled	Specifies if the datasource is enabled or not
site_id	Binds the record to a specific site. This allows using a different data source (than the one globally configured for a satellite) for the specified site

Some of the parameters can also be edited in the /usr/share/sen2agri/sen2agri-services/config/ services.properties.

In the case of using a local repository that does not have the structure in the format .../2018/01/08, the following keys can be added/updated in the configuration file:

```
AWSDataSource.Sentinel2.usePadding=false
AWSDataSource.Landsat8.usePadding=false
```

In this way, repositories having the folder structure like ".../2018/1/8" will be considered.

Other keys that can be configured via the services.properties are given below.

Parameters that should not be modified unless administration is performed:

`server.port = 8081`

➔ This is the port on which the server listens for calls from the interface. Should not be changed unless port is already taken by another application. In this case, the port should be also updated in the file /var/www/html/ConfigParams.php

`spring.datasource.url`

➔ This key gives the connexion string to the database

`spring.datasource.username=admin`

`spring.datasource.password=sen2agri`

➔ Credentials for the sen2agri database

`resolve.links.locally = false`

➔ If set to "true", it "expands" symbolic links by copying locally the files and folders

Parameters that can be optionally changed:

`database.config.polling=15`

➔ Interval in minutes for checking for changes in the database and for execution jobs (ex. queries for datasources to check for new products or retry products interval). 0 means disabled

`network.connexions.timeout=30`

➔ If some datasources are responding very slow, this parameter can be set to a higher value. Pay attention that this applies to all datasources and might impact the performances of the system. To be increased if it is absolutely necessary.

NOTE: Changing one of the keys in the "services.properties" file will require restarting the application using:

```
sudo systemctl restart sen2agri-services
```

Also, after changing datasources parameters in "services.properties" and restarting the application, should be checked the correct update of the "datasource" table using:

```
psql -U admin sen4stat -c "select * from datasource;"
```

If the parameters do not match, manual update of the datasource table might be necessary and restart the application.

NOTE: In the sen2agri-services, compared with the old downloaders, the philosophy of downloading changed as searches are not performed anymore each time from the beginning of the season(s) but instead it searches in an incremental manner, from the last downloaded/aborted product.

With the sen4stat-services, in order to try to retry the aborted products they should be set to status 3 (FAILED) and in the same time reset the no_of_retries column to 0 (there is a retry job in the application that will perform all the retries automatically).

Nevertheless, in order to force a query from the beginning of the season (in the case of missed products), a small plugin was added to the application in order to be able to perform this operation.

This option should be activated only in exceptional situations or if the user really wants this and is aware about the disadvantages (much more requests to datasources, more processing time consuming on machine, possible higher delays in product availability and incompatibility with the near-realtime mode). This is why, this option is disabled by default.

To use this option, add in the "config" table the key "scheduled.lookup.all_products.enabled = true".

Once you notice that the products are up-to-date, you should remove (or set to false) the key from the database as it is not compatible with the near-realtime download mode.

In order to change the logging level of the sen2agri-services, the file /usr/share/sen2agri/sen2agri-services/config/application.properties can be edited for changing especially the following keys:

```
logging.level.ro.cs.tao=TRACE

logging.level.org.esa.sen2agri=TRACE
```

Where the possible values are for log levels are: ERROR, WARN, INFO, DEBUG, or TRACE.

Appendix A.4 Sen4Stat Optical Products L2 Pre-processing Parameters

If configured, the optical level 2 products pre-processor is represented by the demmaccs pre-processor that can support the following configurations:

- Pre-processing of Sentinel-2 L1C and Landsat-8 L1T products into L2A products using MAJA
- Pre-processing of Sentinel-2 L1C products into L2A products using Sen2Cor

The change from MAJA to Sen2Cor is made simply by changing the image of the pre-processor to be used, represented by the key 'processor.l2a.s2.implementation' from 'maja' into 'sen2cor' or viceversa.

Parameter Name	Parameter Value	Description
processor.l2a.maja.gipp-path	/mnt/archive/gipp/maja	MAJA GIPP path
processor.l2a.maja.remove-fre	0	Remove FRE files from resulted L2A product
processor.l2a.maja.remove-sre	1	Remove SRE files from resulted L2A product
processor.l2a.optical.cog-tiffs	0	Produce L2A tiff files as Cloud Optimized Geotiff

processor.l2a.optical.compress-tiffs	0	Compress the resulted L2A TIFF files
processor.l2a.optical.max-retries	3	Number of retries for the L2A processor
processor.l2a.optical.num-workers	4	Parallelism degree of the L2A processor
processor.l2a.optical.output-path	/mnt/archive/maccs_ def/{site}/{processor}/	path for L2A products
processor.l2a.optical.retry-interval	1 day	Retry interval for the L2A processor
processor.l2a.s2.implementation	maja	L2A processor to use for Sentinel-2 products (`maja` or `sen2cor`)
processor.l2a.sen2cor.gipp-path	/mnt/archive/gipp/sen2cor	Sen2Cor GIPP path
processor.l2a.srtm-path	/mnt/archive/srtm	Path to the DEM dataset
processor.l2a.swbd-path	/mnt/archive/swbd	Path to the SWBD dataset
processor.l2a.working-dir	/mnt/archive/demmaccs_tmp/	Working directory
processor.l2a.processors_image	sen4x/l2a-processors:0.2.3	L2a processors image name
processor.l2a.sen2cor_image	sen4x/sen2cor:2.10.01-ubuntu-20.04	Sen2Cor image name
processor.l2a.maja_image	sen4x/maja:4.5.4-centos-7	MAJA image name
processor.l2a.gdal_image	osgeo/gdal:ubuntu-full-3.4.1	GDAL image name
processor.l2a.l8_align_image	sen4x/l2a-l8-alignment:0.1.2	L8 align image name
processor.l2a.dem_image	sen4x/l2a-dem:0.1.3	DEM image name

Appendix A.5 Sen4Stat Services Sentinel-1 Pre-processing Parameters

The Sen4Stat Services S1 Pre-processing module is a component that is doing the amplitude and coherence extraction from the Sentinel-1 SLC products.

Some of its parameters are found in the “config” database table:

Parameter Name	Parameter Value	Description
processor.l2s1.enabled	true	Enables or disables S1 pre-processing
processor.l2s1.acquisition.delay		
processor.l2s1.compute.amplitude	true	Enables or disables the S1 amplitude generation
processor.l2s1.compute.coherence	true	Enables or disables the S1 coherence generation
processor.l2s1.convert.int		Perform a conversion from float to int into output products, using a quantification value.

processor.l2s1.path	/mnt/archive/{site}/l2a-s1	The path where the L2 products will be produced
processor.l2s1.polarisations	VV;VH	The polarisations to be processed
processor.l2s1.projection	EPSG:3035	The projection to be applied to S1 L2 products, either as EPSG code or as WKT
processor.l2s1.gpt.tile.cache.size	256	The size, in MB, used by SNAP tile scheduler
processor.l2s1.gpt.parallelism	8	The number of processors used by SNAP
processor.l2s1.crop.output	true	If true, the L2 coherence products will be cropped by the intersection of S1 SLC products
processor.l2s1.keep.intermediate	false	If true, intermediate processing results are kept
processor.l2s1.crop.nodata	true	If true, the S1 L2 products NoData borders are cropped
processor.l2s1.output.format	GeoTIFF	The SNAP output format name
processor.l2s1.output.extension	.tif	The extension of S1 L2 products
processor.l2s1.parallel.steps.enabled	true	If true, some processing steps can be executed in parallel
processor.l2s1.pixel.spacing	20.0	The pixel spacing of S1 L2 products, in meters
processor.l2s1.resolve.links	false	If true, symlinks are replaced by actual folders and files (copy locally the S1 L1 products)
processor.l2s1.parallelism	1	Number of processings (not steps) that can run in parallel
processor.l2s1.master	S1B	The co-registration master sensor
processor.l2s1.min.intersection	0.05	The minimum overlap required to process a S1 L1 product pair (between 0 and 1)
processor.l2s1.work.dir		If set, intermediate results are done in this location
processor.l2s1.min.memory	8192	The minimum free memory required by a step, in MB
processor.l2s1.extract.histogram	true	If true, histogram is extracted from S1 L2 products and saved into database
processor.l2s1.copy.locally	false	Copy products locally before starting the pre-processing
processor.l2s1.ignore.previous.orbit.failure	false	
processor.l2s1.interval	60	
processor.l2s1.join.amplitude.steps	false	
processor.l2s1.join.coherence.steps	false	

processor.l2s1.min.disk	16384	Minimum disk space needed to execute the processor.
processor.l2s1.min.s2.intersection	0.05	Minimum intersection with a S2 tile to cut for.
processor.l2s1.otb.enable.compression	True	
processor.l2s1.otb.min.memory	2048	The minimum memory to be used by the OTB applications.
processor.l2s1.overwrite.existing	false	Overwrite existing output products. If not set, and the output product already exists, it will not be regenerated
processor.l2s1.process.newest	false	Process the newest products first
processor.l2s1.step.timeout	60	
processor.l2s1.temporal.filter.interval	1	
processor.l2s1.temporal.offset	6	Interval between master and slave acquisitions, in days.
processor.l2s1.terrain.flattening.enabled	false	Enable terrain flattening.
processor.l2s1.use.other.site.products	false	If products already pre-processed on other sites, they will be automatically imported on the current site, without pre-processing.
processor.l2s1.version	2	Version of the S1 pre-processor

Other parameters are found in the same configuration file as for the downloader module:

Parameters that should not be modified unless administration is performed:

```
plugins.use.docker = false
```

➔ The services need GDAL 2+ to perform certain operations. By default, it is expected that GDAL 2+ is found in the system path.

If set to "true", then Docker is expected to be installed and having mounted a container with GDAL 2+.

```
docker.gdal.image = geodata/gdal
```

➔ If the previous setting is "true", it indicates the name of the Docker container that holds GDAL 2+

Parameters that can be optionally changed:

```
ScheduleManager.forced.site = <site_id>
```

➔ If present and not empty, it forces the download and Sentinel-1 processing just for the given site_id. Usually it is not set.

Appendix A.6 Using Sentinel1 and Sentinel2 Download Services with Local L1C Store

If Sentinel-1, Sentinel-2 and Landsat8 low-level products are already downloaded and available in a store , the Download Services can be configured in order to use directly these stores instead of downloading the low-level products from SciHub and USGS. In this case, after the successful installation of the Sen4Stat system, the "*datasource*" table can be edited in order to specify the store.

In order to activate the local store mode, the line corresponding to the download datasource should be updated. For example, for the S2 products (having the column satellite_id = 1 and the scope=2) :

- Set the "local_root" column to the root directory where the L1C products are stored (see "<PATH_TO_LOCAL_EO_DATA>" below);
- Set the "fetch_mode" column value to 4 (symlink mode) or 3 (copy products mode)

The "<PATH_TO_LOCAL_EO_DATA>" should be the root directory where the L1C products are stored in a hierarchical folder structure YYYY/MM/DD/<Products> like in the following example, where the first level is the year, the second level of folders is the month of the year, the third level of folders is the day of the month:

```
2015
2016
2017
    01
    02
    03
       01
       02
       03
       04
          S2A_MSIL1C_<xxxx1>.SAFE
          S2A_MSIL1C_<xxxx1>.SAFE
```

These parameters can also be edited in the /usr/share/sen2agri/sen2agri-services/config/ services.properties by setting, for example, for IPT repository and the option of creating symlinks:

```
AWSDataSource.Sentinel2.local_archive_path=/eodata/Sentinel-2/MSI/L1C
AWSDataSource.Sentinel2.fetch_mode=4
```

The general settings for a DIAS have the following form:

```
<DataSourceClassName>.<Satellite>.path.builder.class                                    =
org.esa.sen4cap.dias.<dias>.Sentinel1PathBuilder
<DataSourceClassName>.<Satellite>.local.archive.path.format = <tokens>
<DataSourceClassName>.<Satellite>.path.suffix = none|.SAFE|<other>
```

```
<DataSourceClassName>.<Satellite>.product.format = folder|zip
```

The configuration file of the services contain sections for all the DIASes (except CreoDIAS, which doesn't need one). It is enough to uncomment the items from a section (and comment the others).

After changing configuration file or the database, the changes normally should be refreshed in maximum 1 minute otherwise the Sen2Agri Services service should be restarted using:

```
sudo systemctl restart sen2agri-services
```

Appendix B - Sen4Stat database tables for manual configuration

The table below presents the main configuration tables that might present interest for the advanced users.

Table name	Description
Site	The table contains all sites that are used in the system
Processor	The table contains all processors that are used in the system
Config_category	The table contains the configuration for the type of applications that can be configured. Examples: Archiver, Executor, L4A Crop Mask Processor, etc.
Config_metadata	The table contains the parameters that can be configured for each type of application contained in Config_category table. The following columns can be modified: - the friendly name for the parameter - the type of the parameter - is_advanced – specifies if the parameter can be configured by an user with admin role - config_category_id – the id from Config_category table
Config	The table contains default values for the parameters of each configured parameters: - key – the name of the parameter - site_id – the id of the site, if the parameter is configured for a specific site - value – the default value of the parameter - last_updated – the time when the parameter was last updated

Apart from these tables, there are also the execution tables that are used by the system but these tables not recommended to be edited by the users.

In order to access the database, the following command can be issued in order to access to the PostgreSQL console:

```
$ psql -U admin sen4stat

sen4stat=#
```

The prompt “sen4stat=#” can be used to issue SQL commands like the following that displays the content of the table “config”:

```
sen4stat=# select * from config;
```

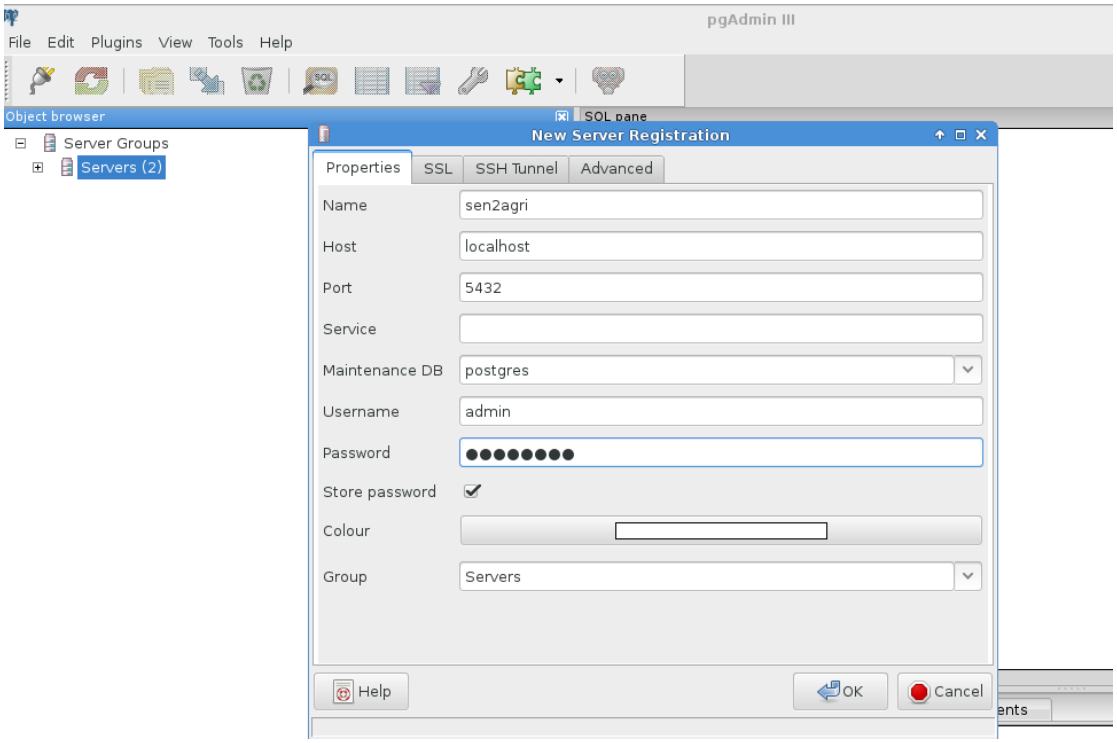
Please note that all commands in this console should end by a semicolon.

Another option, if the Linux was installed with an X server, is to use the much easier to use application, *pgadmin3*. In order to install and run this application, the following commands can be executed:

```
$ sudo yum install -y pgadmin3

$ pgadmin3 &
```

In this application, a new connexion to the database server can be created using the parameters from the picture below (database password is *"sen2agri"*):



Appendix C - System performance example

System performances on an Creodias machine used as calculation basis (8 vCores, 64 gb ram)

	Products	Tile(s) / Orbit(s)	Duration
MAJA	**1**	**1**	**30 min**
FMask	**1**	**1**	**20 min**
L3A cloud free composite	**19**	**1**	**2 hours**
L3B processing	**1**	**1**	**20 min**
Crop Mapping	**NA**	**NA**	**NA**
Yield feature extraction	**109**	**1**	**14 hours**
Yield processor	**NA**	**NA**	**NA**

System performances on an Creodias machine used as calculation basis (16 vCores, 128 gb ram)

	Products	Tile(s) / Orbit(s)	Duration
S1 Preprocessing BCK without terrain flattening	**1**	**1/Polarisation**	**10 min**
S1 Preprocessing BCK with terrain flattening	**1**	**1/Polarisation**	**8 min**
S1 Preprocessing COHE	**1**	**1/Polarisation**	**18 min**

Appendix D - Building from source

In order to build the system, the script Sen2AgriBuildAll.sh (from the git sources sen2agri/packaging) can be used. This script presents to the user the following menu:

```
*****************************************
** 1) Install BUILD PREREQUISITES
** 2) BUILD Sen2AgriPlatform : OTB and GDAL
** 3) BUILD Sen2AgriProcessors, Downloaders and Demmaccs
** 4) BUILD Sen2AgriApplication
** 5) BUILD Sen2AgriWebsite
** 6) BUILD Selective: Enter list of options (ex: platform, processors, website, app)
** 7) BUILD Sen2Agri All Components
*****************************************
Please enter a menu option and enter or enter to exit.
```

The user can choose one of the options in order to build the desired component(s).

In the next 2 paragraphs are presented the options to build the system. Both options can be also implemented using the above interface and by selecting the corresponding menus.

Building the core components

For building the core components (used for manual execution) run the steps 1, 2 and 3 from the general installation script Sen2AgriBuildAll.sh.

Building the components for automated usage

Run step 7 from the general installation script Sen2AgriBuildAll.sh.

Preparing the distribution

In order to create a Sen4Stat system distribution the following steps should be followed:

- Execute Sen2AgriBuildAll.sh

Execute install_platform/sen2agriCreateDistribution.sh by providing the previously created folder with the Sen4Stat system RPM files.

Appendix E – Advanced system configuration items

Appendix E.1 Processing System Configuration Utility

The Sen4Stat Configurator is present in the web interface, in the “Configurations” tab.

The parameters that will appear in the Sen4Stat Configurator are managed through the Sen4Stat database and, depending on their type, they can be modified either by an admin or non-admin user.

Some of the parameters are site specific while other ones are system specific.

The values presented into the controls from each configuration section are the system values, applicable as defaults for all sites. The user has the possibility to change the global values directly in the configuration section and then press the “Save” button.

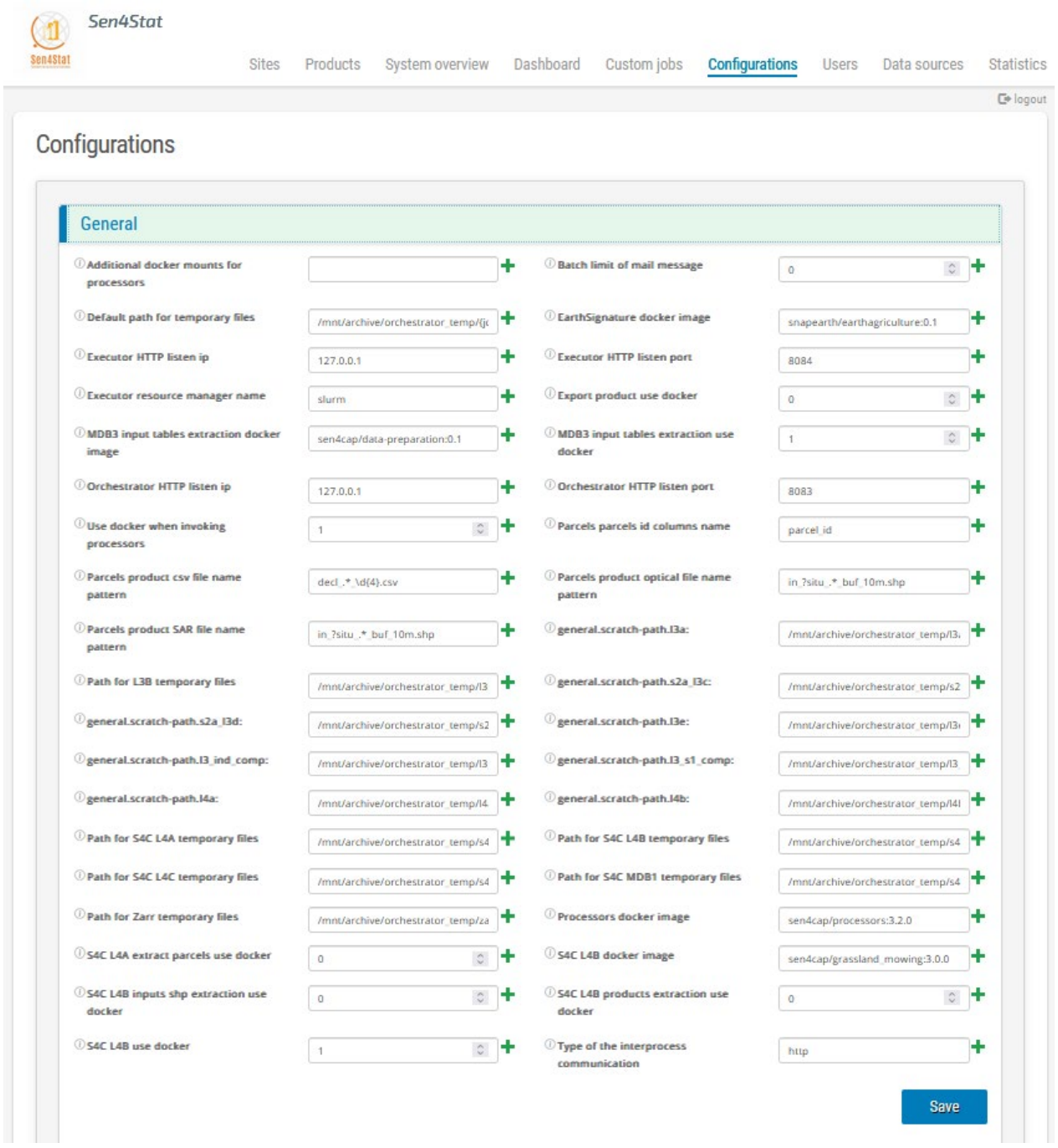


Figure 0-1. Changing configuration values

All parameters values that are modified are validated by the configurator (see Figure 0-2):

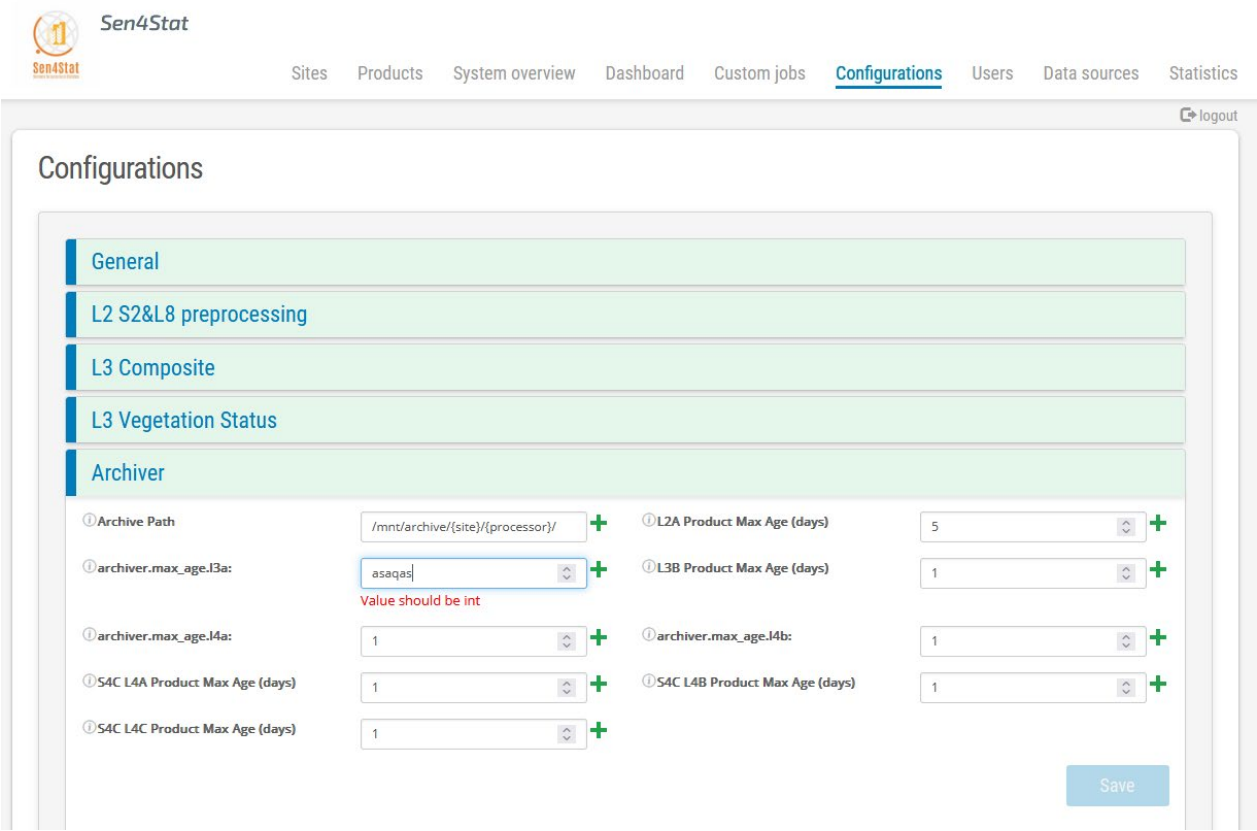


Figure 0-2. Sen4Stat Configurator – parameter validation

Whenever an invalid value is inserted for a configuration, the "Save" button will be disabled until all values for the current section are valid.

For the site specific parameters, the user has the possibility to select the site for which he is configuring the parameter (Figure 0-3), by pressing the + corresponding to the parameter to be changed, operation that opens a new window :

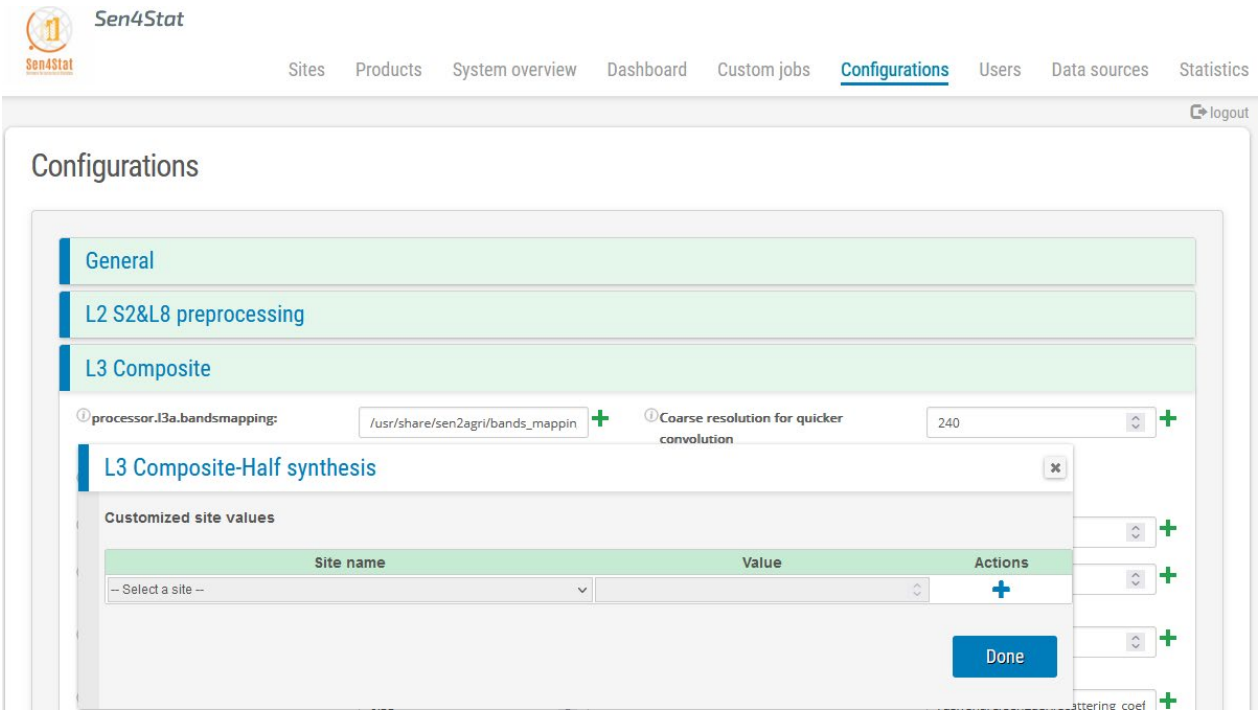


Figure 0-3. Sen4Stat Configurator – customising site parameter value

By pressing again the + button in the new window, it will be offered the possibility to edit that line:

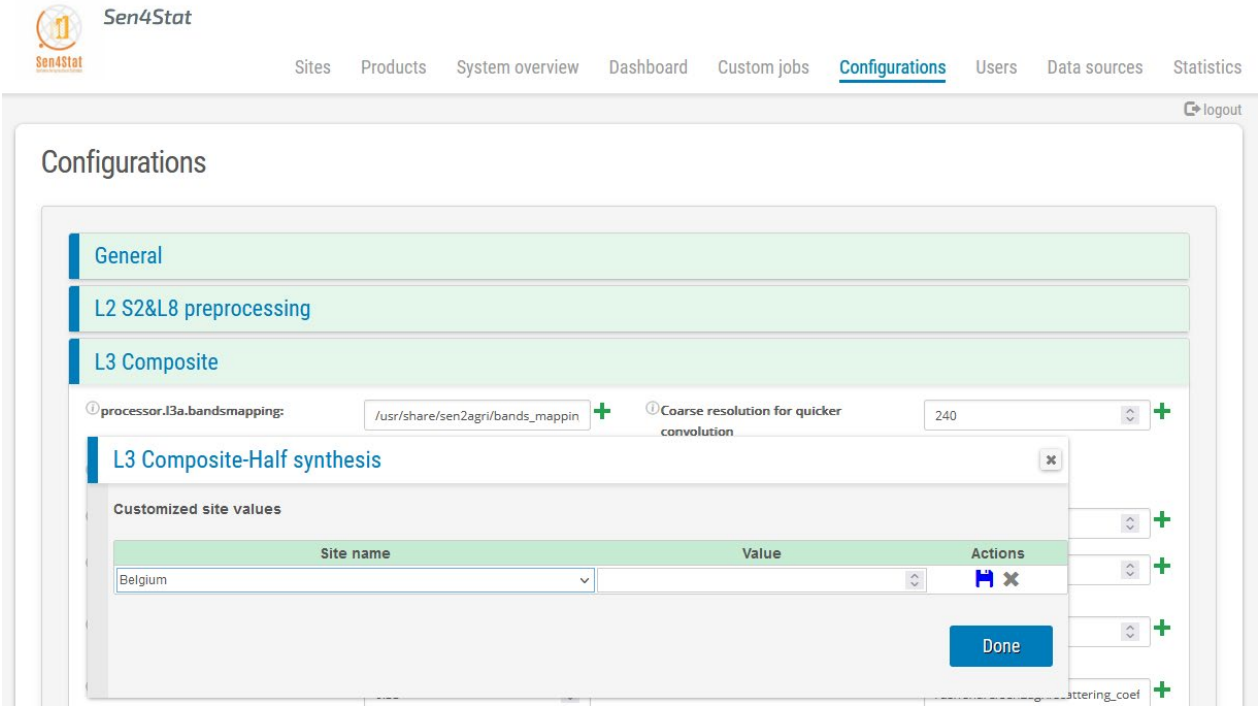
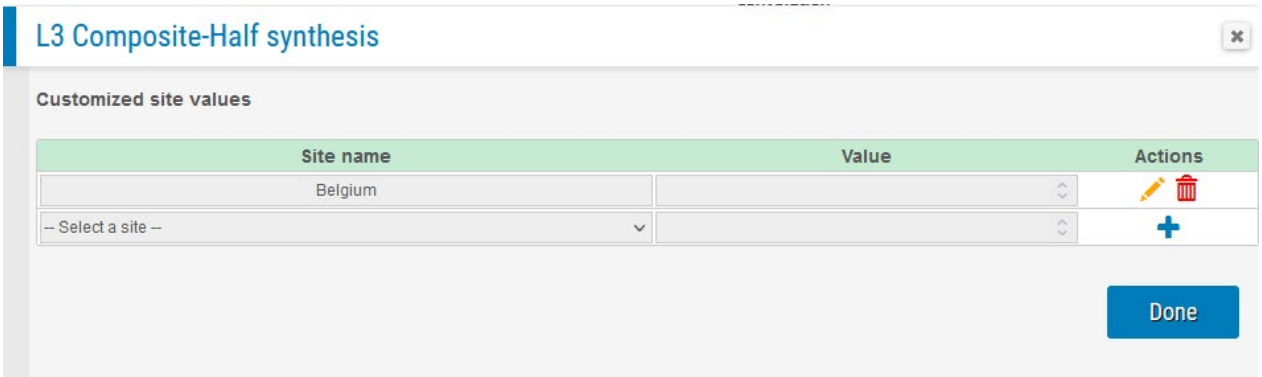


Figure 0-4. Sen4Stat Configurator - setting site specific value

Once the value is set, the configuration can be saved by pressing the button. The existing site specific values can be edited at any time (or removed) by accessing again the button from the main configuration section and then pressing the button.



Appendix E.2 Sen4Stat Archiver

The Sen4Stat Archiver is a system service which interrogates the execution database for products that need to be archived. It will receive a list of products, each product having a current path (source directory) and a destination path (destination directory).

The Configurator contains a tab dedicated to configure the Archiver (Figure 0-5):

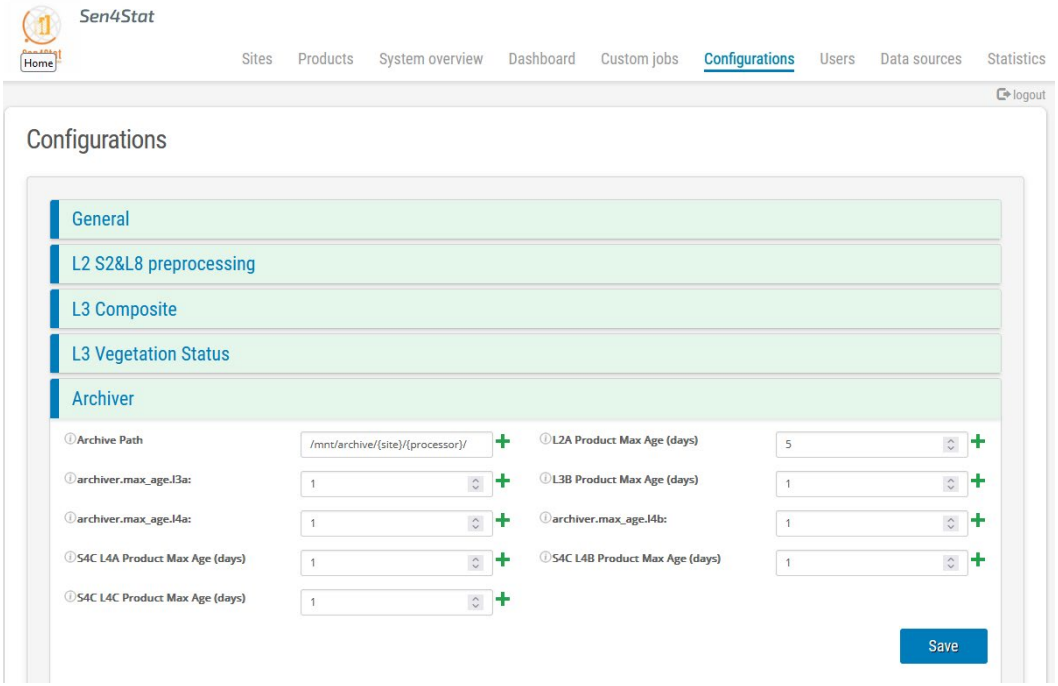


Figure 0-5. Sen4Stat Configurator – Archiver tab

Appendix E.3 Sen4Stat HTTP Listener

The Sen4Stat HTTP Listener is a system service used to offer the information displayed in the Execution Dashboard and to communicate with the Monitor Agent to receive the information that it will send to the dashboard.

The Configurator contains a tab dedicated to the HTTP Listener configuration, called Dashboard (Figure 0-6).

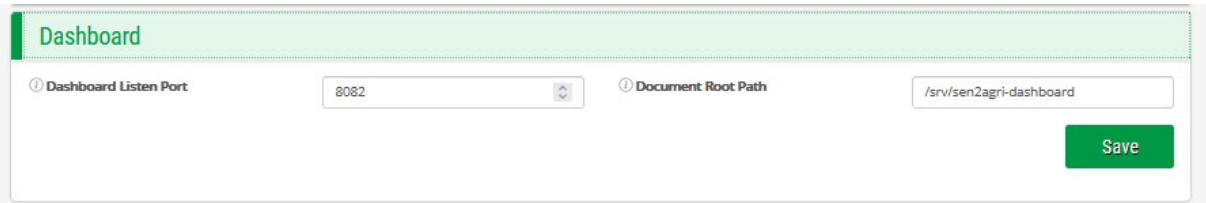


Figure 0-6. Sen4Stat Configurator – Dashboard tab

Appendix E.4 Sen4Stat Executor

The Sen4Stat Executor is a system service that uses SLURM to execute the processors.

The Configurator contains a tab dedicated to the Executor configuration (Figure 0-7).

The parameters used by the Executor are described in Appendix A.1 Sen4Stat Executor parameters.

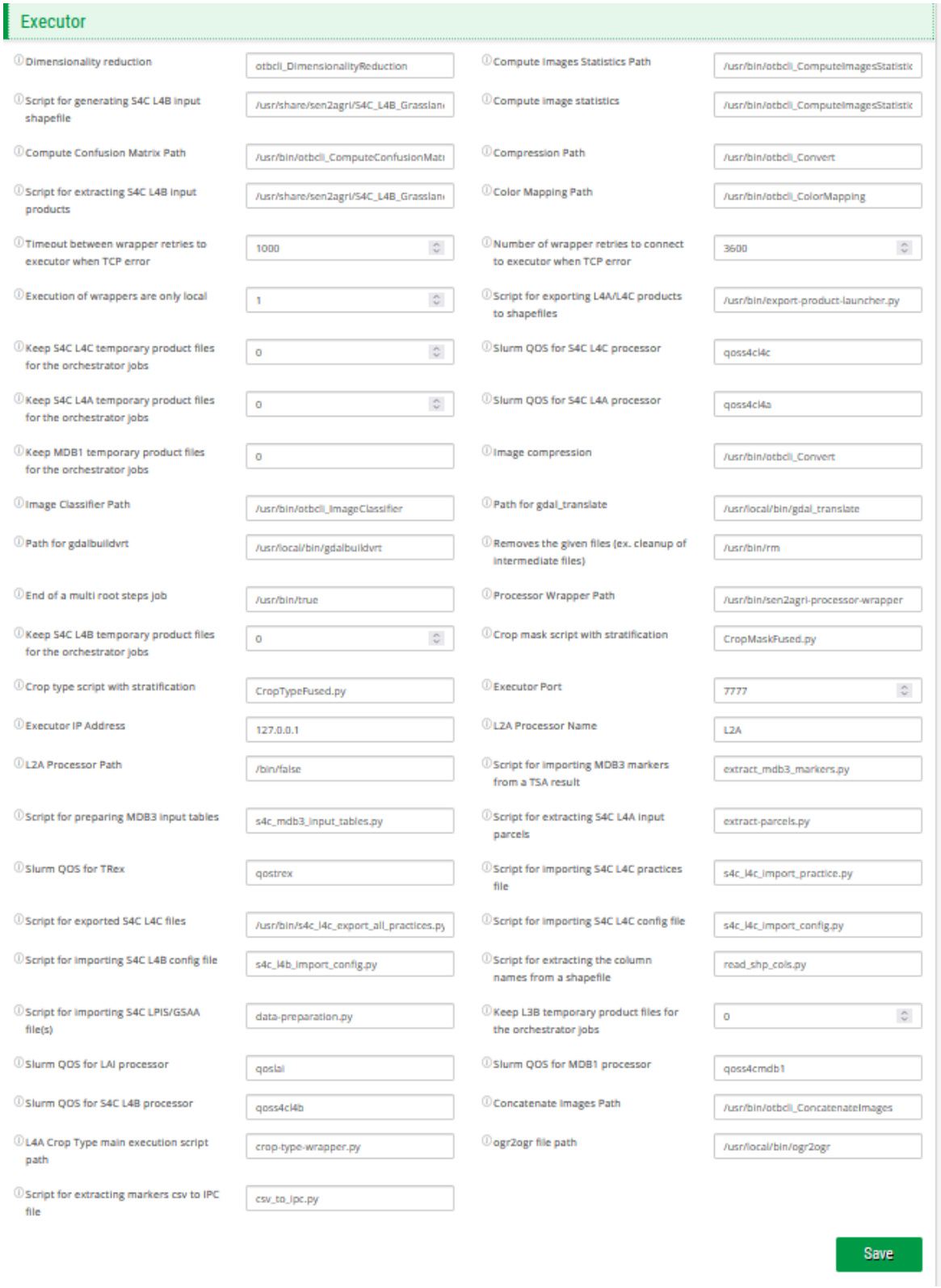


Figure 0-7. Sen4Stat Configurator – Executor tab

Appendix E.5 Sen4Stat Monitor Agent

The Monitor Agent service is used to monitor the Sen4Stat processing nodes and sends metrics to the HTTP Listener in order to be displayed in the Execution Dashboard.

The Configurator contains a tab dedicated to configure the Monitor Agent (Figure 0-8).

Figure 0-8. Sen4Stat Configurator – Monitoring Agent tab

Appendix E.6 Sen4Stat Persistence Manager

The Persistence Manager exchanges information between the database and the other system components (Executor, Monitor Agent, Dashboard and Archiver).

The parameters used by the Persistence Manager are described in Appendix A.2 Sen4Stat Persistence Manager parameters.

Appendix E.7 Sen4Stat Downloader

The S2 L1C, S2 L2A and L8 L1T products are downloaded automatically by the downloader processor installed and configured by the installation script described in section 3.3. This processor is scheduled to run periodically (hourly), as configured by the installation script. Its job is to download the available products/tiles from the ESA Copernicus DAS (https://browser.dataspace.copernicus.eu) for S2 products and from https://ers.cr.usgs.gov/login (for L8 products) according to the parameters described in Appendix A.3 Sen4Stat Services Download parameters.

The configuration parameters (from **config** table) can have values for each site id, and also a global value, for all sites. In the case the key is missing for a certain site, the global value is considered. Even if the database parameters are described here, they can be changed also from the Sen4Stat Configurator application, so the direct database manipulation is not advised.

Some manual configuration should be performed in order to set up a proxy configuration if needed, please see 3.3.3.1.

To configure the application by using the Sen4Stat Configurator, just launch the configurator as shown in Figure 0-9.

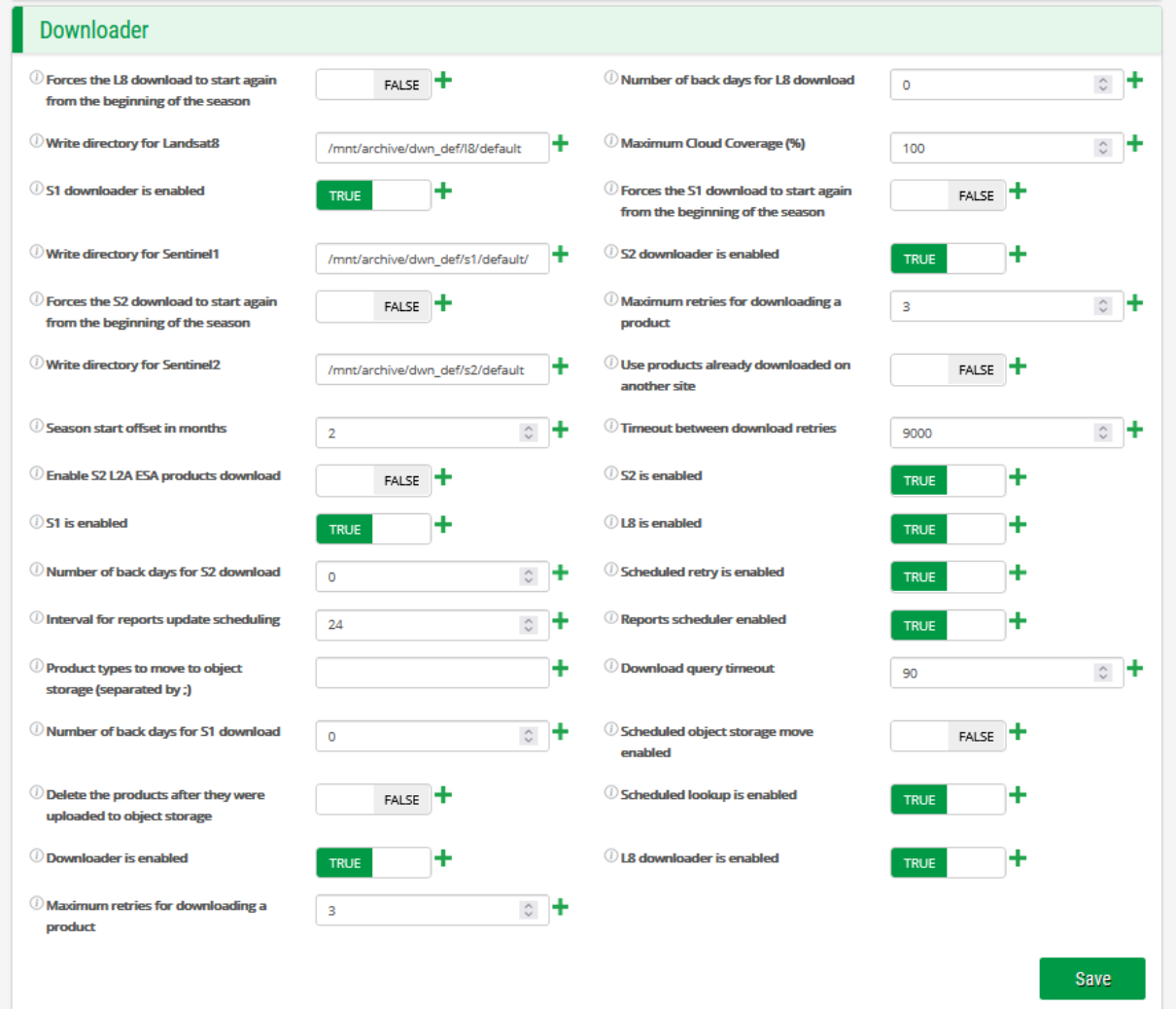


Figure 0-9. Sen4Stat Configurator – Downloader tab

When the downloader processor is launched, it interrogates the database to intersect the polygons of the sites with the shapes defined in the sentinel tiles (**shape_tile_s2**) table (see Appendix B - Sen4Stat database tables for manual configuration for database tables). For the resulting tiles, the downloader interrogates Copernicus DAS site https://browser.dataspace.copernicus.eu and the Landsat site https://ers.cr.usgs.gov/login about the available products for the defined season period for each configured site. From the available products, the downloader checks which of them were already downloaded (this information is located in table **downloader_history)**. After the new products are downloaded, they are added in the **downloader_history** table, with the status **not processed**. Later on, the demmaccs processor (see next section) will handle all the **not processed** products found in the **downloader_history** table.

Appendix E.8 Sen4Stat Demmaccs_launcher

The demmaccs_launcher processor is also installed and configured by the installation script described in section 3.3. It handles all the fully downloaded products from the **downloader_history** table with the status **not_processed** and its main goal is to check the database, to create a list with the unprocessed L1C and L1T

products, to launch with the proper arguments the **demmaccs** processor which in its turn will launch the dem and maccs processors (see section 4.4.1) and to write the results in the database in both **downloader_history** and **product** tables. The demmaccs processor is able to launch the dem and maccs processors in parallel to minimize the processing time (see section 4.4.1: *–processes-number-dem* and *–processes-number-maccs* input arguments).

Appendix E.9 All advanced parameters table

The following configurations are used in the automatic mode of the Sen4Stat system. They can be changed using the Procesing System Configuration Utility.

The advanced processor parameters are prefixed with:

- LAI: “processor.l3b.*”. These parameters can be changed in the "L3 Vegetation Status" of the System Cofiguration Utility;
- CropType: “processor. s4s_l4a.*”. These parameters can be changed in the "L4 Crop Mapping" of the System Cofiguration Utility;
- Cloud free composite: “processor.l3a.*”. These parameters can be changed in the "L3 Composite" of the System Cofiguration Utility;
- Yield Features: “processor.s4s_yield_feat.*”. These parameters can be changed in the "L4 Yield Features" of the System Cofiguration Utility;
- Yield: “processor.s4s_yield.*”. These parameters can be changed in the "L4 Yield" of the System Cofiguration Utility;

Appendix E.10 Advanced useful scripts

- **insert_l2a_product_to_db.py** – allows inserting into a site the L2A products processed in another location. See the help of the script for details.

Appendix F – Endpoint Services description

Appendix F1. Data source configuration

`GET: /downloader/sources/`

This method allows the user to retrieve the list of predefined configurations of data sources modules.

Note: There are no mandatory or optional parameters for this function.

Example: /downloader/sources/

`GET: /downloader/sources/{satellite}`

This method allows the user to retrieve the list of predefined configurations of data sources modules for the given satellite (*satellite*).

Parameter	Description
satellite: *short*	The unique identifier of a sentinel; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);

Example: /downloader/sources/1

`GET: /downloader/sources/{satellite}/{name}`

This method allows the user to retrieve the configuration of the given data source (*name*) for the given satellite (*satellite*).

Parameter	Description
satellite: *short*	The unique identifier of a sentinel; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);
name: *String*	The name of the data source from which the user wants to download data.

Example: /downloader/sources/1/SciHub

`POST: /downloader/sources/{satellite}/{name}`

This method allows the user to update a certain configuration of the given data source (*name*) for the given satellite (*satellite*).

Parameter	Description
satellite: *short*	The unique identifier of a sentinel; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);
name: *String*	The name of the data source from which the user wants to updata data.

Example: /downloader/sources/1/SciHub

`GET: /refresh/`

This method allows the user to refresh the configuration.

Note: There are no mandatory or optional parameters for this function.

Example: /refresh/

`GET: /refresh/info`

This method allows the user to view informations regarding the memory, threads and jobs.

Note: There are no mandatory or optional parameters for this function.

Example: /refresh/info

`GET: /parcel`

This method allows the user to retrieve information about specific parcel. This method performs a verification of all the input parameters, in order to provide the correct information.

Parameter	Description
site: *String*	The name of the site for which we want to retrieve information about the parcel.
id: *String*	The id of the parcel for which we want to retrieve information.
practice: *Practice*	
year: *int*	

Example: /parcel

Appendix F2. Download control

`GET: /downloader/`

This method returns information about all the downloads in progress.

Note: There are no mandatory or optional parameters for this function.

Example: /downloader/

`GET: /downloader/{id}`

This method returns information about the downloads in progress for a specific site (*id*).

Parameter	Description
id: *short*	The unique identifier of a site; if the value of the parameter is: • id = 0: it returns the list with all the downloads in progress; • id != 0: it returns the list with all the downloads in progress for the given site;

Example: /downloader/1235

`GET: /downloader/{code}/{satellite}`

This method returns information about the download in progress for a specific site (*id*) and a specific satellite (*satellite*).

Parameter	Description
code: *String*	The unique identifier of a site; if the value of the parameter is: • id = 0: it returns the list with all the downloads in progress; • id != 0: it returns the list with all the downloads in progress for the given site;
satellite: *String*	The name of the sentinel from which the user wants to retrieve the information regarding the status of the download in progress.

Example: /downloader/netherlands/S2

`GET: /downloader/{id}/count`

This method retrieves information about the estimated number of products to download for a specific site (*id*)

Parameter	Description
id: *short*	The unique identifier of the site for which the user wants to retrieve the information regarding the estimated number of products to download.

Example: /downloader/1235/count

`GET: /downloader/stop`

This method stops all the downloads and marks the downloader as disabled for all sites.

Note: There are no mandatory or optional parameters for this function.

Example: /downloader/stop

`GET: /downloader/stop/{id}`

This method stops all the downloads and marks the downloader as disabled for a specific site (*id*).

Parameter	Description
id: *short*	The unique identifier of the site for which the user wants to stop all the downloads; if the value of the parameter is: • id = 0: it stops all the downloads in progress; • id != 0: it stops all the downloads in progress for the given site;

Example: /downloader/stop/1235

`GET: /downloader/stop/{id}/{satelliteId}`

This method stops all the downloads and marks the downloader as disabled for the specific site (*id*) and specific sensor (*satelliteId*).

Parameter	Description
id: *short*	The unique identifier of the site for which the user wants to stop all the downloads for the given sensor (*satelliteId*);
satelliteId: *short*	The unique identifier of the sensor from which the user stops all the downloads, for the given site.

Example: /downloader/stop/1235/2

`GET: /downloader/start`

This method starts all the downloads and marks the downloader as enabled for all sites.

Note: There are no mandatory or optional parameters for this function.

Example: /downloader/start

`GET: /downloader/start/{id}`

This method starts all the downloads and marks the downloader as enabled for a specific site (*id*).

Parameter	Description
id: *short*	The unique identifier of the site for which the user wants to enable all the downloads; if the value of the parameter is: • id = 0: it enables all the downloads in progress; • id != 0: it enables all the downloads in progress for the given site;

Example: /downloader/start/1235

`GET: /downloader/start/{id}/{satelliteId}`

This method starts all the downloads and marks the downloader as enabled for a specific site (*id*) and a specific satellite (*satelliteId*).

Parameter	Description
id: *short*	The unique identifier of the site for which the user wants to start all the downloads, for the given sensor (*satelliteId*);

satelliteId: *short*	The unique identifier of the sensor from which the user enables all the downloads, for the given site (*id*).

Example: /downloader/start/1235/3

`GET: /downloader/forcestart`

This method forces the downloader for the specific job (*job*), site (*siteId*) and sensor (*satelliteId*) to start from the beginning of the first defined season.

Parameter	Description
job: *String*	The unique identifier of the job type that the user wants to force to restart;
siteId: *short*	The unique identifier of a site; if the value of the parameter is: • siteId = 0: it forces the downloader to start from the beginning for all the sites; • siteId != 0: it forces the downloader to start from the beginning for the given site;
satelliteId: *short*	The unique identifier of the sensor from which the user forces the restart of the downloader;

Example: /downloader/forcestart

Appendix F3. Import

`GET: /products/import/l1`

This method allows the user to import data into the system.

Parameter	Description
siteId: *short*	The name of the site for which we want to import L1 products into the system.
folder: *String*	The relative folder where we want to upload L1 products.
satelliteId: *short*	The name of the site for which we want to upload L1 products.
link: *boolean*	If: • true: is linking folder to target path; • false: is copying the folder to target path;

Example: /products/import/l1

`GET: /import/s4sparcels`

This method allows the user to import the NSO in-situ datasets into the system. At least a NSO in-situ geographical dataset (as zipped shapefile) and a NSO in-situ statistical dataset file must be present.

Parameter	Description
siteId: *short*	The name of the site for which we want to import the LPIS file.

year: *Optional <Integer>*	
parcelsFile: *Optional <String>*	
parcelStatsFile: *Optional <String>*	

Example: /import/s4sParcels

`GET: /import/{siteShortName}`

This method allows the user to check if a certain site exists into the system and, if so, update its data. This method performs a file size check in order validate the verification.

Parameter	Description
siteShortName: *String*	The name of the site for which we want update data.
file: *String*	The file with new data, which we want to add to the system.
type: *DataType*	The file type (*LPIS* or *DECLARATION*).
offset: *Optional <Long>*	

Example: /import/ netherlands_1

`GET: /import/{siteShortName}/{year}`

This method allows the user to check if a certain site exists into the system and, if so, update its data. This method performs a file size check in order validate the verification.

Parameter	Description
siteShortName: *String*	
year: *int*	
file: *String*	
type: *Optional <DataType>*	
crs: *Optional <String>*	
offset: *Optional <Long>*	

Example: /import/ netherlands_1/2018

`GET: /s1l2/config`

This method allows the user to retrieve the actual configuration values for the Sentinel-1 L2 processor.

Note: There are no mandatory or optional parameters for this function.

Example: /s1l2/ config

`GET: /s1l2/`

This method allows the user to retrieve the actual metadata of the product specified in the given path.

Parameter	Description
path: *String*	The path of the product for which the user wants to retrieve the metadata.

Example: /s1l2/

`GET: /s1l2/reprocess`

This method allows the user to trigger the reprocessing of the Sentinel-1 L1 SLC acquisition, for a given site and for a given polarisation.

Parameter	Description
siteCode: *String*	The path of the product for which the user wants to retrieve the metadata.
product: *String*	The name of the acquisition.
polarisation: *Optional <String>*	The name of the polarisation for which the user wants to do the reprocessing. By default: VV and VH.

Example: /s1l2/

`GET: /s1l2/import/l2/{site}`

This method allows the user to import in the system database the L2 products from a specific folder.

Parameter	Description
site: *String*	The site name for which we trigger the import.
folder: *String*	The path of the folder from which we want to import data.

Example: /s1l2/import/l2/ netherlands_1

Appendix F4. Output products

`GET: /sites/`

This method allows the user to retrieve the list of all existing sites.

Note: There are no mandatory or optional parameters for this function.

Example: /sites/

`POST: /sites/`

This method allows the user to create a new site; it returns the unique identifier of the created site (*siteId*).

Parameter	Description
name: *String*	The name of the site we want to create.
zipFilePath: *String*	The relative path of the site shapefile.
enabled: *boolean*	Provides information concerning the site status (enabled/disabled).

Example: /sites/

`DELETE: /sites/`

This method allows the user to delete an existing site and all related products.

Parameter	Description
siteId: *short*	The unique identifier of the site we want to delete.

Example: /sites/

`GET: /sites/seasons/{id}`

This method allows the user to retrieve the list of all seasons for a specific site (*id*).

Parameter	Description
siteId: *String*	The unique identifier of the site for which we want to retrieve of all seasons.

Example: /sites/seasons/15

`POST: /login`

This method allows the user to connect to the application using a valid username and password.

Parameter	Description
user: *String*	The unique identifier of the name/entity/person who wants to use the application.
pwd: *String*	The unique identifier that allows the user (*user*) to access the application.

Example: /login

`GET: /products/`

This method allows the user to retrieve the list of all the existing products.

Parameter	Description
userId: *int*	The unique identifier of the logged user.
siteId: *int*	The unique identifier of the site for which we want to retrieve the products.

productTypeId: *int*	The unique identifier of the product types we want to retrieve.
token: *String*	The unique identifier that authenticates the users who attempt to log into the application.

Example: /products/

`GET: /products/disable/{satellite}/{id}`

This method allows the user to disable the downloading or processing products for a specific satellite (*satellite*) and site (*id*).

Parameter	Description
satellite: *short*	The enum value of sensor we want to disable; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);
id: *short*	The unique identifier of the site we want to disable; if the value of the parameter is: • id = 0: it disables the downloads or processing in progress for all sites; • id != 0: it disables the downloads or processing in progress for the given site;

Example: /products/disable/3/1235

`GET: /products/enable/{satellite}/{id}`

This method allows the user to enable the downloading or processing products for a specific satellite (*satellite*) and site (*id*).

Parameter	Description
satellite: *short*	The enum value of sensor we want to enable; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);
id: *short*	The unique identifier of the site we want to enable; if the value of the parameter is: • id = 0: it enables the downloads or processing in progress for all sites; • id != 0: it enables the downloads or processing in progress for the given site;

Example: /products/enable/3/1235

`GET: /products/enable/status/{satellite}/{id}`

This method allows the user to view the status (enable/disable) of a specific satellite (*satellite*) and site (*id*).

Parameter	Description
satellite: *short*	The enum value of sensor for which we want to check the status; possible values: • 1 = S2 (Sentinel 2); • 2 = L8 (Landsat8); • 3 = S1 (Sentinel 1);
id: *short*	The unique identifier of the site for which we want to check the status.

Example: /products/enable/status/3/1235

`GET: /products/objectstorage/get`

This method returns information about the product types for all stored objects

Note: There are no mandatory or optional parameters for this function.

Example: /products/objectstorage/get

`GET: /products/objectstorage/set/{productsTypeIds}`

This method returns information about a specific product type.

Parameter	Description
productsTypeIds: *String[]*	The unique identifier of the site products types we want to retrieve.

Example: /products/objectstorage/get/12;15

`GET: /products/types`

This method returns a list with all the existing product types

Note: There are no mandatory or optional parameters for this function.

Example: /products/types

`GET: /products/download`

This method allows the user to download a specific product (*name*) for a specific site (*siteId*), in ".zip" format.

Parameter	Description
name: *String*	The name of the product we want to download.
siteId: *int*	The unique identifier of the site for which we want to download the given product.

Example: /products/download

`POST: /products/upload`

This method allows the user to upload files into the system.

Parameter	Description
siteId: *int*	The name of the site for which we want to upload a given file.
file: *MultipartFile*	The file we want to upload.
folder: *String*	The relative folder we want to upload the file.

Example: /products/upload

`GET: /objectstorage/productTypes`

This method allows the user to retrieve the product type ids that should be moved to Object Storage buckets.

Note: There are no mandatory or optional parameters for this function.

Example: /objectstorage/productTypes

`POST: /objectstorage/productTypes`

This method allows the user to set the product type ids that should be moved to Object Storage buckets.

Note: There are no mandatory or optional parameters for this function.

Example: /objectstorage/productTypes

`GET: /objectstorage/delete`

This method allows the user to retrieve the number of deleted objects.

Parameter	Description
container: *String*	The Object Storage bucket.
filter: *String*	The name of the applied filter.

Example: /objectstorage/delete

`GET: /objectstorage/copy`

This method allows the user to copy the given file, associated to the given site and representing the given product type, to Object Storage.

Parameter	Description
siteId: *short*	The unique identifier of the site we want to copy.
productType: *ProductType*	The product type identifier.
year: *int*	The year of the product.
file: *String*	The file representing the product.

Example: /objectstorage/copy

`GET: /objectstorage/move`

This method allows the user to move the given file, associated to the given site and representing the given product type, to Object Storage.

Parameter	Description
siteId: *short*	The unique identifier of the site we want to move.
productType: *ProductType*	The product type identifier.
year: *int*	The year of the product.
file: *String*	The file representing the product.

Example: /objectstorage/copy