

Resource Management Operators support



EOX IT Services <https://eox.at>



EOfarm P.C. <http://eofarm.com>

EOEPCA Operators demo 2022-09-20



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).

Overview

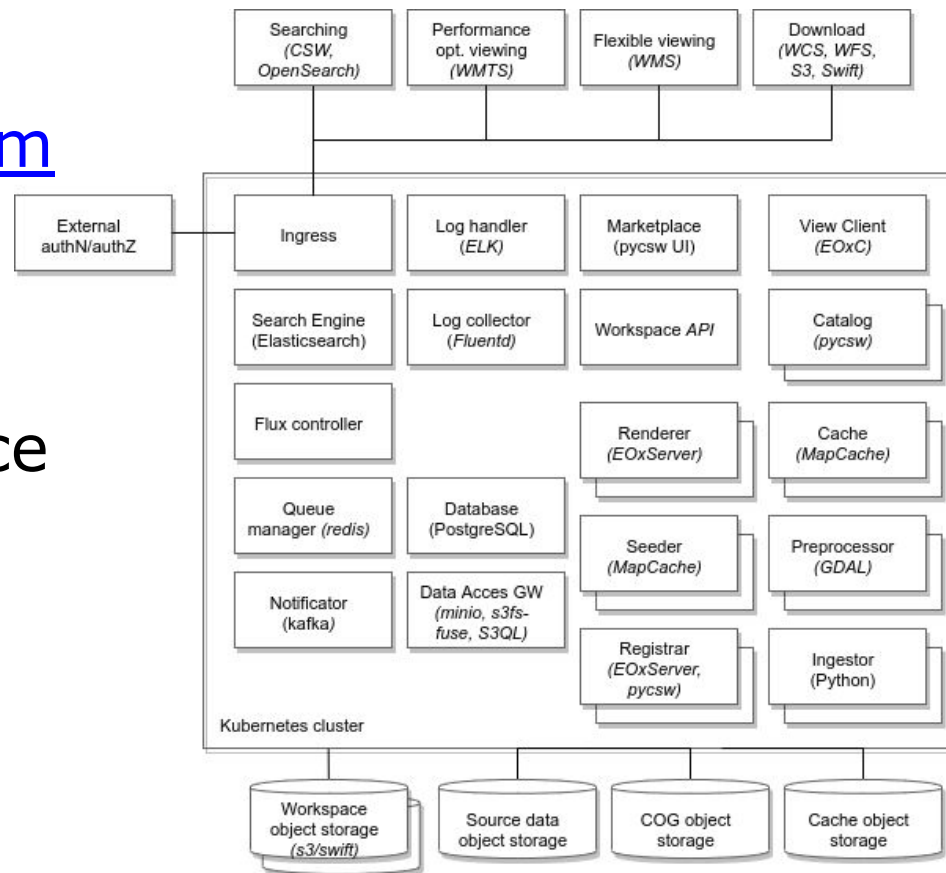
- Team
- Resource Management Architecture
- Resource Catalogue
- Data Access Service & Ingestion
- User Workspace
- Summary

Team



Resource Management Architecture

- Heavily based on [Python geospatial stack/ecosystem](#)
- Kubernetes mainly using helm charts
- Automation via flux
- System and per Workspace deployments



Resource Catalog Overview

- Discover/Search, Harvest, Distributed/Federated Search (user and system catalog)
- Metadata
- Standards
 - ISO 19115/19139
 - OGC CSW 2.0.2 / 3.0.0
 - Core Profile
 - ISO Application Profile
 - OGC OpenSearch Geo/Time, EO Profiles
 - OGC API - Records - Part 1: Core
 - STAC API (1.0.0-rc1), STAC Item (1.0.0)

Resource Catalog Technologies

- [pycsw](#)
 - Python OGC Reference Implementation Server
 - Multiple metadata formats (ISO, DC etc.)
 - Flexible backends (RDBMS)
- [OWSLib](#)
 - Swiss army knife Python client library for OGC Web Services and APIs
- [pygeometa](#)
 - Metadata composition
- [pygeoapi](#)
 - Python OGC Server Reference Implementation
 - evolving OGC API standards

Resource Catalogue Technologies

- PostGIS
 - PostgreSQL Spatial extension
 - OGC SFSQL
 - Metadata Store for pycsw, EOxServer
- Elasticsearch (in Roadmap)
 - Full Text Search Engine
 - Highly Distributed

Resource Catalogue Configuration

- Helm Chart available:
<https://github.com/EOEPCA/helm-charts/releases/tag/rm-resource-catalogue-1.1.0>
- pycsw
 - Configuration via helm chart values:
<https://github.com/EOEPCA/helm-charts/blob/main/charts/rm-resource-catalogue/values.yaml#L53-L131>
 - Can be scaled vertically and horizontally since it is a stateless service, just add instances.
 - Populate the catalogue through:
 - pycsw-admin CLI:
<https://docs.pycsw.org/en/latest/administration.html#loading-records>
 - OGC CSW-T:

Resource Catalogue Configuration

- PostgreSQL
 - Configuration via helm chart values:
<https://github.com/EOEPCA/helm-charts/blob/main/charts/rm-resource-catalogue/values.yaml#L5-L45>
 - Can be highly optimized for millions of metadata records using spatial index and full text search support.
 - Can be scaled vertically
 - Horizontal scaling can be achieved in various scenarios: (i) Read Scalability through pgpool or pgbouncer by using PostgreSQL replication (ii) Read-Write Scalability using FDW and Partitioning or tools like Postgres-XL and Citus.
 - Replacing the default helm chart with a high-availability one, like Bitnami:

Resource Catalogue Examples

← → ↻ 🏠 🔒 https://resource-catalogue.demo.eoepca.org ★ 📧 🔍 Search



pycsw

EOEPCA Resource Catalogue

[Home](#) [JSON](#) | [Contact](#)

Based on pycsw, a Python OGC CSW server implementation

[Collections](#)

OpenAPI

- [Swagger](#)
- [JSON](#)

[Conformance](#)

- [CSW 3.0.0](#)
- [CSW 2.0.2](#)
- [OpenSearch](#)
- [STAC API](#)
- [OAI-PMH](#)
- [SRU](#)

Powered by  pycsw 3.0.dev0

← → ↻ 🏠 🔒 https://resource-catalogue.demo.eoepca.org/csw ... 📧 🔍 Search

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
<!-- pycsw 3.0.dev0 -->
<csw30:Capabilities version="3.0.0" updateSequence="1631886261" xsi:schemaLocation="http://www.opengis.net/cat/csw/3.0 http://schemas.opengis.net/ows/3.0/csw/3.0/owsGetCapabilities.xsd">
  <ows20:ServiceIdentification>
    <ows20:Title>pycsw Geospatial Catalogue</ows20:Title>
    <ows20:Abstract>
      pycsw is an OGC CSW server implementation written in Python
    </ows20:Abstract>
    <ows20:Keywords>
      <ows20:Keyword>catalogue</ows20:Keyword>
      <ows20:Keyword>discovery</ows20:Keyword>
      <ows20:Keyword>metadata</ows20:Keyword>
      <ows20:Type codeSpace="ISOTC211/19115">theme</ows20:Type>
    </ows20:Keywords>
    <ows20:ServiceType codeSpace="OGC">CSW</ows20:ServiceType>
    <ows20:ServiceTypeVersion>2.0.2</ows20:ServiceTypeVersion>
    <ows20:ServiceTypeVersion>3.0.0</ows20:ServiceTypeVersion>
    <ows20:Profile>http://www.opengis.net/cat/csw/apiso/1.0</ows20:Profile>
    <ows20:Fees>None</ows20:Fees>
    <ows20:AccessConstraints>None</ows20:AccessConstraints>
  </ows20:ServiceIdentification>
  <ows20:ServiceProvider>
    <ows20:ProviderName>Organization Name</ows20:ProviderName>
    <ows20:ProviderSite xlink:type="simple" xlink:href="https://pycsw.org/">
  </ows20:ServiceContact>
    <ows20:IndividualName>Lastname, Firstname</ows20:IndividualName>
    <ows20:PositionName>Position Title</ows20:PositionName>
  </ows20:ContactInfo>
    <ows20:Phone>
      <ows20:Voice>+xx-xxx-xxx-xxxx</ows20:Voice>
      <ows20:Facsimile>+xx-xxx-xxx-xxxx</ows20:Facsimile>
    </ows20:Phone>
    <ows20:Address>
      <ows20:DeliveryPoint>Mailing Address</ows20:DeliveryPoint>
      <ows20:City>City</ows20:City>
      <ows20:AdministrativeArea>Administrative Area</ows20:AdministrativeArea>
      <ows20:PostalCode>Zip or Postal Code</ows20:PostalCode>
    </ows20:Address>
  </ows20:ServiceProvider>
</csw30:Capabilities>
```

Resource Catalogue Examples

The screenshot shows a web browser at the URL `https://resource-catalogue.demo.eoezca.org/?f=json`. The page displays a JSON document with the following structure:

```
{
  "stac_version": "1.0.0-beta.4",
  "id": "pycsw-catalogue",
  "type": "Catalog",
  "conformsTo": [
    "http://www.opengis.net/spec/ogcapi-common-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-common-2/1.0/conf/collections",
    "http://www.opengis.net/spec/ogcapi-features-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-features-3/1.0/req/filter",
    "http://www.opengis.net/spec/ogcapi-records-1/1.0/conf/core",
    "http://www.opengis.net/spec/ogcapi-records-1/1.0/conf/sorting",
    "http://www.opengis.net/spec/ogcapi-records-1/1.0/conf/json",
    "http://www.opengis.net/spec/ogcapi-records-1/1.0/conf/html",
    "https://api.stacspec.org/v1.0.0-beta.4/core",
    "https://api.stacspec.org/v1.0.0-beta.4/item-search",
    "https://api.stacspec.org/v1.0.0-beta.4/item-search#filter",
    "https://api.stacspec.org/v1.0.0-beta.4/item-search#sort"
  ],
  "links": [
    {
      "rel": "self",
      "type": "application/json",
      "title": "This document as JSON",
      "href": "https://resource-catalogue.demo.eoezca.org/?f=json",
      "hreflang": "en-US"
    },
    {
      "rel": "root",
      "type": "application/json",
      "title": "The root URI as JSON",
      "href": "https://resource-catalogue.demo.eoezca.org/?f=json",
      "hreflang": "en-US"
    },
    {
      "rel": "conformance",
      "type": "application/json",
      "title": "Conformance as JSON",
      "href": "https://resource-catalogue.demo.eoezca.org/conformance?f=json"
    },
    {
      "rel": "service-doc",
      "type": "text/html"
    }
  ]
}
```

The screenshot shows the Swagger UI for the EOEPCA Resource Catalogue API. The title is "EOEPCA Resource Catalogue" with version "1.0" and "OAS3" specification. The URL is `https://resource-catalogue.demo.eoezca.org/openapi?f=json`. The description states: "Based on pycsw, a Python OGC CSW server Implementation". The "Servers" section shows the base URL: `https://resource-catalogue.demo.eoezca.org - Based on pycsw, a Python OGC CSW server implementation`. The "Capabilities" section lists the essential characteristics of the API:

- GET** `/` Landing page
- GET** `/conformance` Conformance page
- GET** `/collections` Collections page
- GET** `/collections/metadata:main` Collection page

The "Data" section lists the access to data (records):

- GET** `/collections/metadata:main/items` Records search items page
- GET** `/search` Records search items page

Resource Catalogue Examples

← → ↺ 🏠 🔒 https://resource-catalogue.demo.eoepca.org/collections/metadata 📄 ☆ 📧 ⬇️ 🔍 Search 🏠 ABR 🌐


pyCSW
EOEPCA Resource Catalogue

[Home](#) / [Collections](#) / [EOEPCA Resource Catalogue](#) / [Items](#) [JSON](#) | [Contact](#)



Leaflet | Map data © OpenStreetMap contributors

[Prev](#) [Next](#)

Title	Type
S2A_MSIL1C_20200930T092031_N0209_R093_T34TFK_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T35TLF_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SFF_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34TGG_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SEH_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T35SKD_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SFH_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SFE_20200930T105731.SAFE	dataset
S2A_MSIL1C_20200930T092031_N0209_R093_T34SEF_20200930T105731.SAFE	dataset

[Prev](#) [Next](#)

Resource Catalogue Example

https://resource-catalogue.demo.eoepca.org/collections/metadata



pycsw

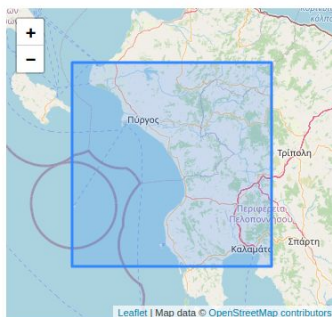
EOEPCA Resource Catalogue

[Home](#) / [Collections](#) / [EOEPCA Resource Catalogue](#) / [Items](#) /

[JSON](#) | [Contact](#)

[S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE](#)

S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE



Property	Value
datetime	2020-09-30T10:57:31.000000Z
start_datetime	2020-09-30T09:20:31.024Z
end_datetime	2020-09-30T09:20:31.024Z
externalId	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
recordUpdated	2021-12-06T16:21:09Z
type	dataset
created	2020-09-30T10:57:31.000000Z
updated	2020-09-30T10:57:31.000000Z
title	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
description	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
keywords	['Orthoimagery', 'Land cover', 'Geographical names', 'data set series', 'processing', 'eo:productType:', 'eo:orbitNumber:', 'eo:orbitDirection:', 'eo:processingLevel:Level-1C']
associations	[{'href': 's3://EODATA/Sentinel-2/MSI/L1C/2020/09

https://resource-catalogue.demo.eoepca.org/collections/metadata/main/it

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```

{
  "id": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
  "type": "Feature",
  "geometry": {
    "type": "Polygon",
    "coordinates": [
      [
        [
          [
            21,
            36.95
          ],
          [
            21,
            37.95
          ],
          [
            22.23,
            37.95
          ],
          [
            22.23,
            36.95
          ],
          [
            21,
            36.95
          ]
        ]
      ]
    ]
  },
  "properties": {
    "datetime": "2020-09-30T10:57:31.000000Z",
    "start_datetime": "2020-09-30T09:20:31.024Z",
    "end_datetime": "2020-09-30T09:20:31.024Z",
    "externalId": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "recordUpdated": "2021-12-06T16:21:09Z",
    "type": "dataset",
    "created": "2020-09-30T10:57:31.000000Z",
    "updated": "2020-09-30T10:57:31.000000Z",
    "title": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "description": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "keywords": [
      "Orthoimagery",
      "Land cover",
      "Geographical names",
      "data set series",
      "processing",
      "eo:productType:",
      "eo:orbitNumber:",
      "eo:orbitDirection:",
      "eo:processingLevel:Level-1C"
    ]
  },
  "associations": [
    {
      "href": "s3://EODATA/Sentinel-2/MSI/L1C/2020/09
    }
  ]
}

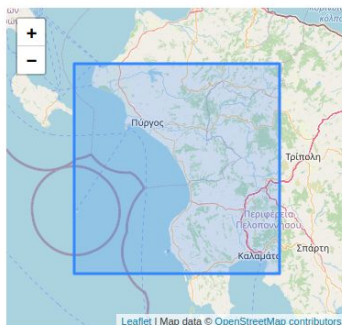
```


Resource Catalogue Example

Home / Collections / EOEPKA Resource Catalogue / Items / S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE

[JSON](#) | [Contact](#)

S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE



Property	Value
datetime	2020-09-30T10:57:31.000000Z
start_datetime	2020-09-30T09:20:31.024Z
end_datetime	2020-09-30T09:20:31.024Z
externalId	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
recordUpdated	2021-12-06T16:21:09Z
type	dataset
created	2020-09-30T10:57:31.000000Z
updated	2020-09-30T10:57:31.000000Z
title	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
description	S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE
keywords	['Orthoimagery', 'Land cover', 'Geographical names', 'data set series', 'processing', 'eo:productType:/', 'eo:orbitNumber:/', 'eo:orbitDirection:/', 'eo:processingLevel:Level-1C']
extent	{ 'spatial': { 'bbox': [[21.0, 36.95, 22.23, 37.95]], 'crs': 'http://www.opengis.net/def/crs/OGC/1.3/CRS84' } }

← → ↺ 🏠 https://resource-catalogue.demo.eoepca.org/stac/collections/metadata:ma

JSON Raw Data Headers

Save Copy Collapse All Expand All Filter JSON

```

{
  "id": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
  "type": "Feature",
  "geometry": {
    "type": "Polygon",
    "coordinates": [
      [
        [
          [
            21,
            36.95
          ],
          [
            21,
            37.95
          ],
          [
            22.23,
            37.95
          ],
          [
            22.23,
            36.95
          ],
          [
            21,
            36.95
          ]
        ]
      ]
    ]
  },
  "properties": {
    "datetime": "2020-09-30T10:57:31.000000Z",
    "start_datetime": "2020-09-30T09:20:31.024Z",
    "end_datetime": "2020-09-30T09:20:31.024Z",
    "externalId": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "recordUpdated": "2021-12-06T16:21:09Z",
    "type": "dataset",
    "created": "2020-09-30T10:57:31.000000Z",
    "updated": "2020-09-30T10:57:31.000000Z",
    "title": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "description": "S2A_MSIL1C_20200930T092031_N0209_R093_T34SEG_20200930T105731.SAFE",
    "keywords": [
      "Orthoimagery",
      "Land cover",
      "Geographical names",
      "data set series",
      "processing",
      "eo:productType:/",
      "eo:orbitNumber:/",
      "eo:orbitDirection:/",
      "eo:processingLevel:Level-1C"
    ]
  },
  "extent": {
    "spatial": {
      "bbox": [
        [
          21.0,
          36.95,
          22.23,
          37.95
        ]
      ],
      "crs": "http://www.opengis.net/def/crs/OGC/1.3/CRS84"
    }
  }
}
```

Resource Catalogue Examples

```

Launcher
x 04 Resource Catalogue.ipyn
+ ✂ 📄 ▶ ■ ⌂ ⏪ ⏩ Markdown v

[11]: #bbox_query = BBox([37.8, 23.4, 38.8, 24.5])
      bbox_query = BBox([37, 13.9, 37.9, 15.1])

[12]: begin = PropertyIsGreaterThanOrEqualTo(propertyname='apiso:TempExtent_begin', literal='2021-04-02 00:00')

[13]: end = PropertyIsLessThanOrEqualTo(propertyname='apiso:TempExtent_end', literal='2021-04-03 00:00')

[14]: cloud = PropertyIsLessThanOrEqualTo(propertyname='apiso:CloudCover', literal='20')

[15]: filter_list = [
      And(
        [
          bbox_query, # bounding box
          begin, end, # start and end date
          cloud # cloud
        ]
      )
    ]

```

The filter is then applied to the GetRecords request and results are shown:

```

[17]: csw.getrecords2(constraints=filter_list, outputschema='http://www.isotc211.org/2005/gmd')
      csw.results

[17]: {'matches': 1, 'returned': 1, 'nextrecord': 0}

[18]: selected_record = list(csw.records)[0]

[19]: for rec in csw.records:
      print(f'identifier: {csw.records[rec].identifier}\ntype: {csw.records[rec].identification.identtype}\n')

identifier: S2B_MSIL1C_20210402T095029_N0300_R079_T33SVB_20210402T121737.SAFE
type: dataset
title: S2B_MSIL1C_20210402T095029_N0300_R079_T33SVB_20210402T121737.SAFE

```

```

Launcher
x 04 Resource Catalogue.ipyn
+ ✂ 📄 ▶ ■ ⌂ ⏪ ⏩ Markdown v Python 3

[ ]: for ref in rec.references:
      url = ref['url']
      if 'WCS' in url:
          print(msg(geolink=sniff_link(url), **ref))
          break

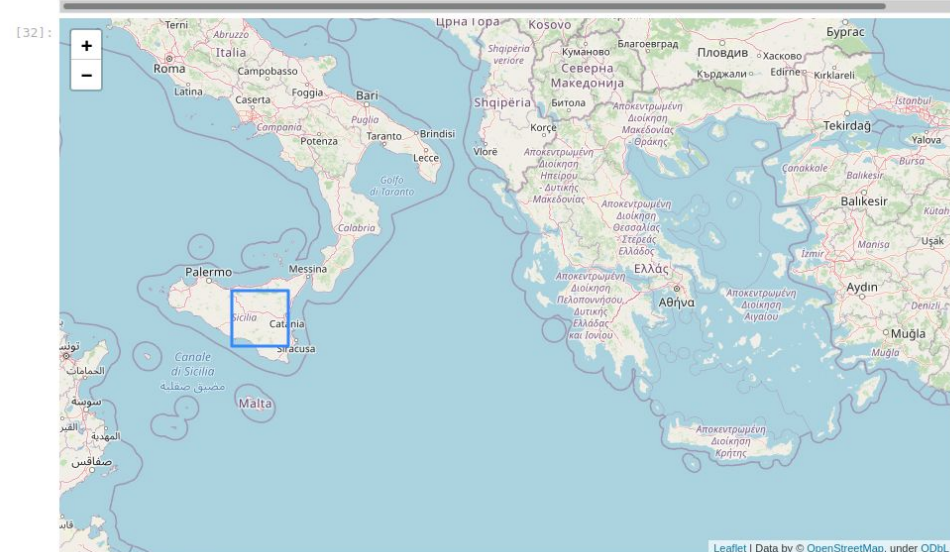
```

Finally we demonstrate how to show the record footprint on a map, using the **Folium** Python library:

```

[32]: m = folium.Map(location=[38, 20], zoom_start=6, tiles='OpenStreetMap')
      folium.Rectangle(bounds=[[float(rec.bbox.miny), float(rec.bbox.minx)], [float(rec.bbox.maxy), float(rec.bbox.maxx)]]).
      m

```



Resource Catalogue Examples

The left screenshot displays the 'Edit Catalog Service' dialog box. It contains the following fields and options:

- Name:** EOEPCA CREODIAS
- URL:** http://resource-catalogue.185.52.193.87.nip.io/
- Authentication:** If the service requires basic authentication, enter a user name and optional password.
 - User name:**
 - Password:**
- Buttons:** OK, Cancel, Help, Close.

The right screenshot displays the 'MetaSearch' window. It includes a search bar, a 'Find' section with coordinates (Xmax, Ymax, Xmin, Ymin), and a 'Results' section showing a list of datasets.

Search Parameters:

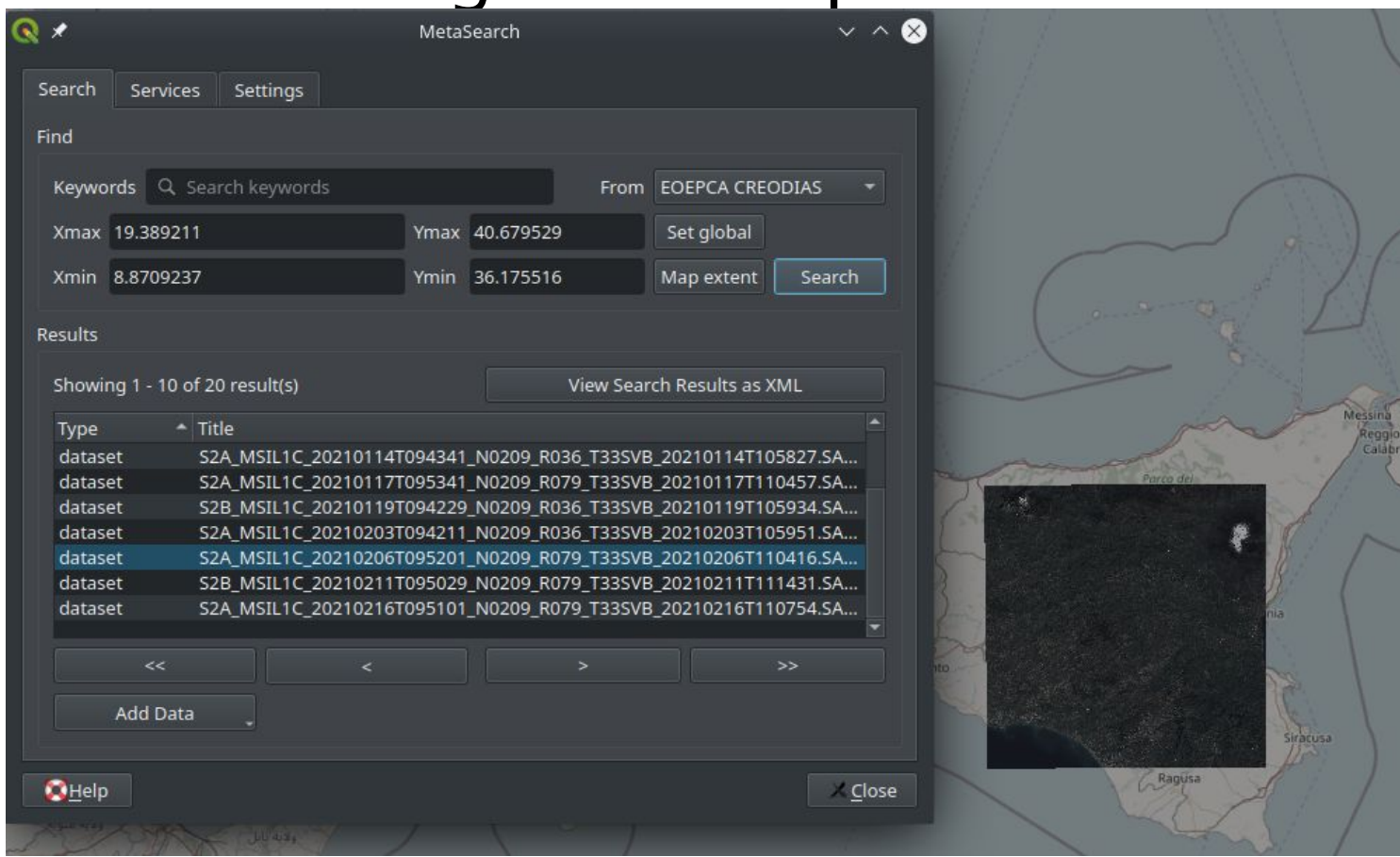
- Keywords:** Search keywords
- From:** EOEPCA CREODIAS
- Xmax:** 19.389211
- Ymax:** 40.679529
- Xmin:** 8.8709237
- Ymin:** 36.175516
- Buttons:** Set global, Map extent, Search

Results: Showing 1 - 10 of 20 result(s). View Search Results as XML

Type	Title
dataset	S2A_MSIL1C_20210114T094341_N0209_R036_T33SVB_20210114T105827.SA...
dataset	S2A_MSIL1C_20210117T095341_N0209_R079_T33SVB_20210117T110457.SA...
dataset	S2B_MSIL1C_20210119T094229_N0209_R036_T33SVB_20210119T105934.SA...
dataset	S2A_MSIL1C_20210203T094211_N0209_R036_T33SVB_20210203T105951.SA...
dataset	S2A_MSIL1C_20210206T095201_N0209_R079_T33SVB_20210206T110416.SA...
dataset	S2B_MSIL1C_20210211T095029_N0209_R079_T33SVB_20210211T111431.SA...
dataset	S2A_MSIL1C_20210216T095101_N0209_R079_T33SVB_20210216T110754.SA...

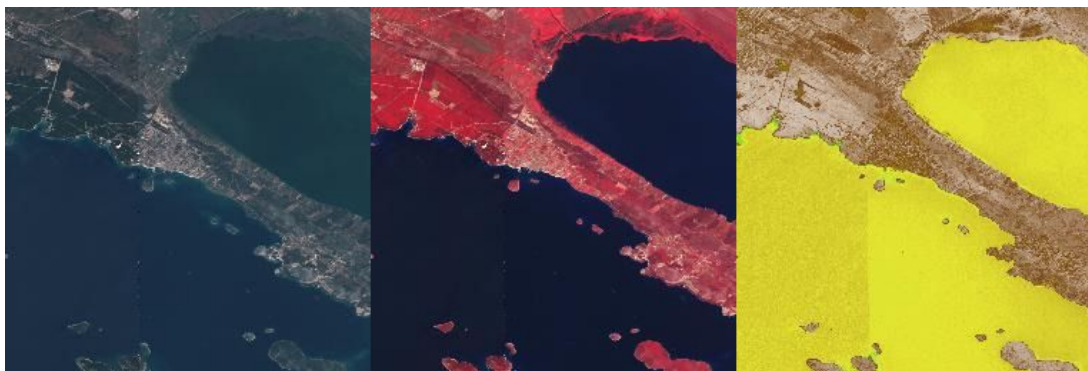
Buttons: <<, <, >, >>, Add Data, Help, Close.

Resource Catalogue Examples



Data Access Service Overview

- View Server (EOxServer as core)
- Flexible WMS viewing interface
 - raster expressions, outlines, masking, property filtering
- Performance optimized WM(T)S cached viewing



Data Access Service Overview

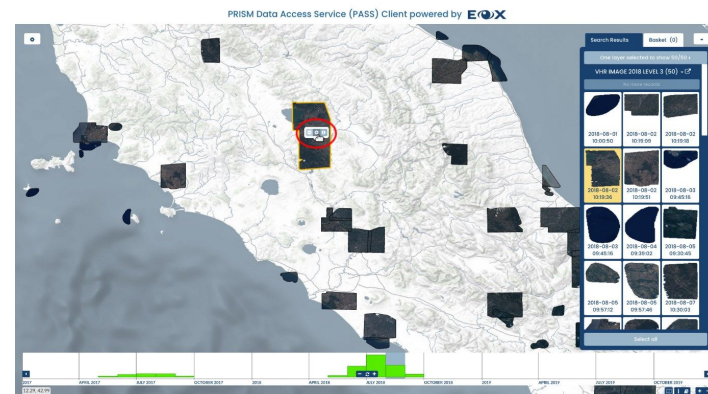
- Download interfaces
 - WCS + EO Application Profile
 - DSEO
- OGC OpenSearch catalog interface
 - Geo Extension
 - Time Extension
 - EO Extension
 - CQL filtering

Data Access Ingestion

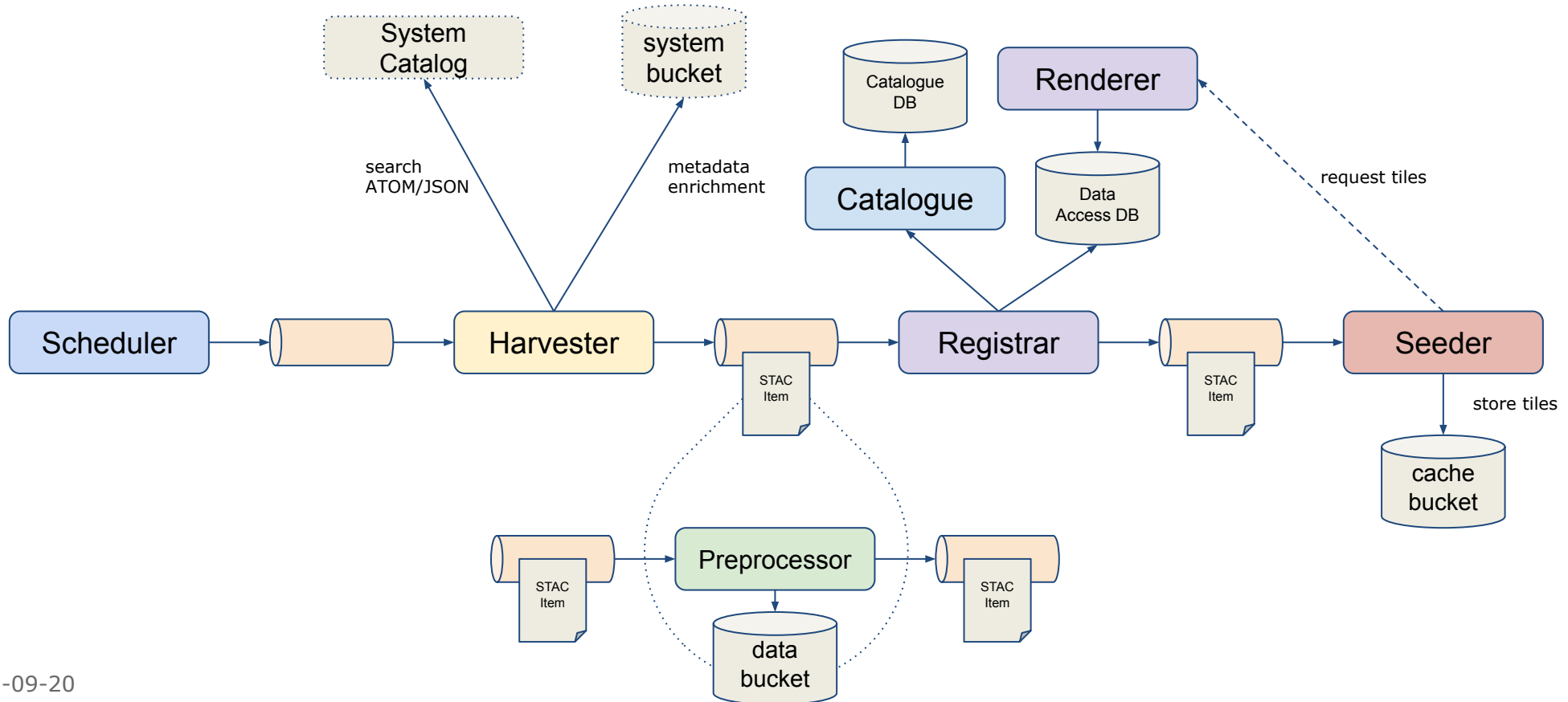
- Harvesting from System catalog
- Metadata enrichment
- Export to STAC Item
- Registration into Data Access and Resource Catalog

Data Access Service & Ingestion Components

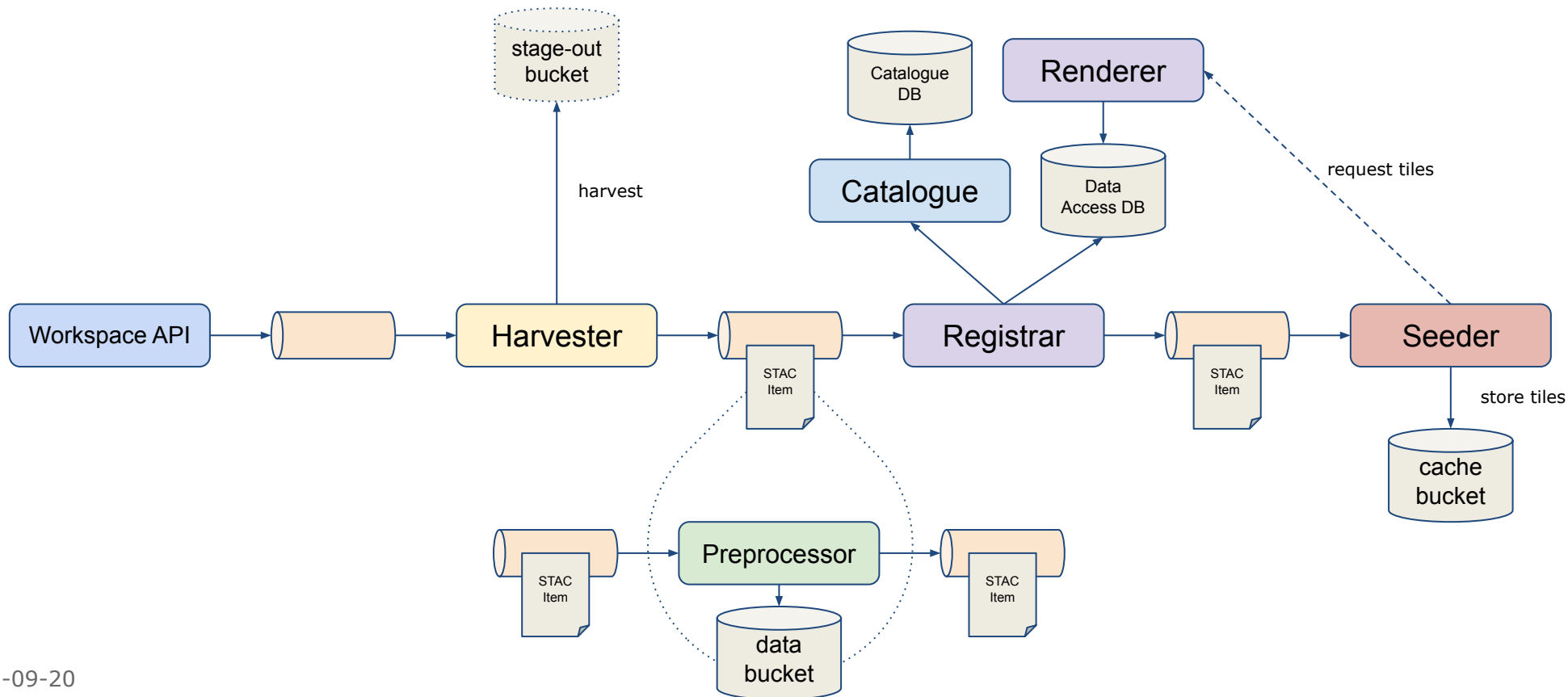
- Database - [PostgreSQL](#)
- Renderer - [EOxServer](#)
- Cache - [Mapcache](#)
- Registrar - [Python](#) + [EOxServer](#)
- Harvester - [Python](#)
- [Redis](#) queues
- Webclient based on [EOxC](#)
- [Elasticsearch](#)/[Fluentd](#)/[Kibana](#) logging stack



Data Access Ingestion



Data Access Ingestion - Workspace



Harvesting configuration

- Several harvesters can be configured

```
harvesters:
- name: Creodias-Opensearch
  resource:
    url: https://finder.creodias.eu/resto/api/collections/Sentinel2/describe.xml
    type: OpenSearch
    format_config:
      type: 'application/json'
    property_mapping:
      start_datetime: 'startDate'
      end_datetime: 'completionDate'
      productIdentifier: 'productIdentifier'
    query:
      time:
        property: sensed
        begin: 2019-09-10T00:00:00Z
        end: 2019-09-11T00:00:00Z
      collection: null
      bbox: 14.9,47.7,16.4,48.7
    filter: {}
    postprocess:
      - type: harvester_eopca.postprocess.CREODIASOpenSearchSentinel2Postprocessor
    queue: register
```

Harvesting configuration

- `resource`: defines from where shall be harvested
- `filter`: filter harvested items using CQL2-JSON
- `postprocess`:
 - apply postprocessing (Python script)
 - transformation to STAC if not already
 - metadata enrichment

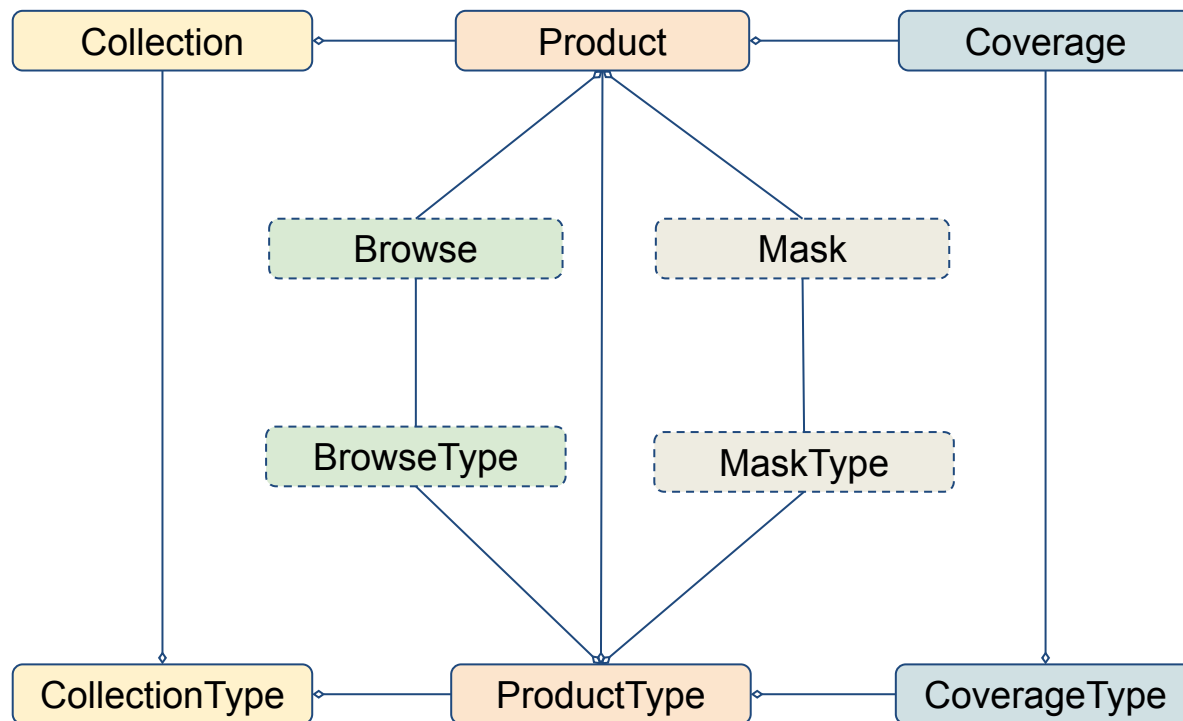
```
class MyPostprocessor(Postprocessor):  
    def postprocess(self, item: dict) -> dict:  
        ...
```

- `queue`: output queue

Data Access configuration

- View Server can be used with rigid type system
 - Products
 - main Record item
 - Coverages
 - sub-item of Products (rel OGC Coverages)
 - Browsers
 - visual representation of a Product
 - either materialized or virtual
 - Masks
 - Collections

Data Access configuration



Data Access configuration

- Helm chart values
- global values:
 - `productTypes`
 - `collections`
- `and registrar` specific values

Data Access ProductType configuration

- Product type definition:

```
productTypes:  
  - name: S2MSI1C  
    filter:  
      s2:product_type: S2MSI1C  
    collections:  
      - S2L1C  
    metadata_assets: []  
    coverages:  
      S2L1C_B01:  
        assets:  
          - B01  
    ...
```

Data Access ProductType configuration

- Browse setup:

```
...
defaultBrowse: TRUE_COLOR
browses:
  TRUE_COLOR:
    asset: visual
    red:
      expression: B04
      range: [0, 4000]
      nodata: 0
    green:
      expression: B03
      range: [0, 4000]
      nodata: 0
    blue:
      expression: B02
      range: [0, 4000]
      nodata: 0
```

```
browses:
  ...
  NDVI:
    grey:
      expression: (B08-B04) / (B08+B04)
      range: [-1, 1]
  masks:
    clouds:
      validity: false
```

Data Access ProductType configuration

- `filter`: when registering only matching products use this product type
- `collections`: place this product in the collection
- `coverages`: all coverages to be created for the product and the asset name to find it in the STAC Item
- `browses`:
 - the asset name of the browse and/or
 - the red, green, blue or greyscale sources or expressions
 - value ranges to stretch

Data Access Collection configuration

- Collection definition:

```
collections:  
  S2L1C:  
    product_types:  
      - S2MSI1C  
    coverage_types:  
      - S2L1C_B01  
      - S2L1C_B02  
      - S2L1C_B03  
      - ...
```

Data Access Collection configuration

- `product_types`: product types that can be inserted into this collection
- `coverage_types`: coverage types that can be inserted into this collection

Registration configuration

```
config:
  defaultBackends:
    - path: registrar_pycsw.backend.ItemBackend
      kwargs:
        repository_database_uri: postgresql://postgres:mypass@resource-catalogue-db/pycsw
        ows_url: https://data-access.develop.eoepca.org/ows
  defaultSuccessQueue: seed_queue
```

Registration configuration

```
disableDefaultRoute: true
routes:
  items:
    path: registrar.route.stac.ItemRoute
    queue: register_queue
    replace: true
  backends:
    - path: "registrar.backend.eoxserver.ItemBackend"
      kwargs:
        instance_base_path: "/var/www/pvs/dev"
        instance_name: "pvs_instance"
        product_types: []
        auto_create_product_types: True
    - path: "registrar_pycsw.backend.ItemBackend"
      kwargs:
        repository_database_uri: "postgresql://postgres:mypass@resource-catalogue-db/pycsw"
        ows_url: "https://data-access.{{ workspace_name }}.develop.eoepca.org/ows"
        public_s3_url: "https://cf2.cloudferro.com:8080/{projectid}:{bucket}"
```


Registration configuration

```
collections:
  path: registrar.route.stac.CollectionRoute
  queue: register_collection_queue
  replace: true
  backends:
    - path: registrar_pycsw.backend.CollectionBackend
      kwargs:
        repository_database_uri : postgresql://postgres:mypass@resource-catalogue-db/pycsw

ades:
  path: registrar.route.json.JSONRoute
  queue: register_ades_queue
  replace: true
  backends:
    - path: registrar_pycsw.backend.ADESBBackend
      kwargs:
        repository_database_uri : postgresql://postgres:mypass@resource-catalogue-db/pycsw

application:
  path: registrar.route.json.JSONRoute
  queue: register_application_queue
  replace: true
  backends:
    - path: registrar_pycsw.backend.CWLBackend
      kwargs:
        repository_database_uri : "postgresql://postgres:mypass@resource-catalogue-db/pycsw"
        ows_url : "https://data-access.{{ workspace_name }}.develop.eoepca.org/ows"
        public_s3_url : "https://cf2.cloudferro.com:8080/{projectid}:{bucket}"
```

Registration configuration

- “Routes”
 - Listens on a specific queue
 - Parses to a specific type (e.g.: STAC Item)
 - Connected to backends (e.g.: pycsw)
 - Modes (register, replace, deregister)
 - Configurable output queue
 - Default Route (additional backends)

Seeder Configuration

- Re-uses cache configuration
- ``minzoom` / `maxzoom``

Renderer Configuration

- Re-uses global ProductType setup
- EO-WCS:
 - Collections -> advertised DatasetSeries
 - Products -> non-advertised DatasetSeries
 - Coverages -> non-advertised Coverages
 - Typical flow:
 - GetCapabilities -> DatasetSeries
 - DescribeEOCoverageSet -> Coverages
 - GetCoverage

Renderer Configuration

- EO-WMS:
 - Collections: advertised layer structure
 - Products: non-advertised layer structure
 - Coverage: non-advertised layer structure
- Basic layers:
 - "`<name>`" -> default rendering
 - "`<name>__outlines`" -> rendered footprint
 - "`<name>__outlined`" -> default rendering with imposed footprint

Renderer Configuration

- Collection/Products:
 - "<name>__<browse>": specific visualization
 - e.g: "S2L1C__NDVI"
 - "<name>__<mask>": polygon mask
 - e.g: "S2L2A__clouds"
 - "<name>__masked_<mask>": default visualization with masks subtracted

Renderer Configuration

- Coverages:
 - `dim_bands` parameter for RGB composition
 - e.g: `...&dim_bands=B08,B04,B03&...`
 - `dim_ranges` parameter
 - e.g: `...&dim_range=0 4000&...`

Renderer Configuration

- Additional parameters:
 - `cql`: Allows to pass CQL filters
 - **e.g:** `...&cql=cloudCover < 10&...`
 - `sortBy`: Allows to sort rendered images
 - **e.g:** `...&sortBy=cloudCover A&...`

Renderer Configuration

- OpenSearch
- Automatically exposed as set of endpoints
- `/opensearch` -> root OSDD
- `/opensearch/<format>` -> collection level search
- `/opensearch/collections/<collection>` -> collection OSDD
- `/opensearch/collections/<collection>/<format>` -> product level search

Cache/Client Configuration

```
layers:
  - id: S2L1C
    title: Sentinel-2 Level 1C True Color
    abstract: Sentinel-2 Level 2A True Color
    displayColor: '#eb3700'
    grids:
      - name: WGS84
        zoom: 13
    parentLayer: S2L1C
    search:
      histogramBinCount: 15
      histogramThreshold: 80
  - id: S2L1C__TRUE_COLOR
    title: Sentinel-2 Level 1C True Color
    abstract: Sentinel-2 Level 2A True Color
    grids:
      - name: WGS84
        zoom: 13
    parentLayer: S2L1C
```

Cache/Client Configuration

- `global.layers`
- **Metadata:** `id`, `title`, `abstract`
- `displayColor`: used in the client
- `grids`: what grids shall be cached and until which `zoomlevel`
- `parentLayer`: to build layer hierarchies
- `search`: search settings
- `global.storage.cache`:
 - storage configuration
 - cache bucket credentials as secret

Cache Configuration

- Endpoints:
 - `/cache/` -> main entry point
 - `/cache/wmts` -> WMTS
 - `/cache/wms` -> WMS endpoint

User Workspace

- EOEPKA Model
 - Object storage bucket
 - Data Access Service
 - Resource Catalog with federated system catalog
 - Deployment scenario(s) up to operator
- [Workspace API](#) based on FastAPI
 - create, delete, etc.
 - using “Bucket” claims via K8s CRD & K8s Secret
 - register data & processes
- [Bucket operator](#) for OpenStack

User Workspace

FastAPI 0.1.0 OAS3

/openapi.json

default

GET	/probe	Probe	⌵
POST	/workspaces	Create Workspace	⌵
GET	/workspaces/{workspace_name}	Get Workspace	⌵
DELETE	/workspaces/{workspace_name}	Delete Workspace	⌵
PATCH	/workspaces/{workspace_name}	Patch Workspace	⌵
POST	/workspaces/{workspace_name}/redploy	Redeploy Workspace	⌵
POST	/workspaces/{workspace_name}/register	Register	⌵
POST	/workspaces/{workspace_name}/deregister	Deregister	⌵
POST	/workspaces/{workspace_name}/register-collection	Register Collection	⌵
POST	/workspaces/{workspace_name}/create-container-registry-repository	Create Container Registry Repository	⌵
POST	/grant-access-to-container-registry-repository	Grant Container Registry Access View	⌵

User Workspace Customization

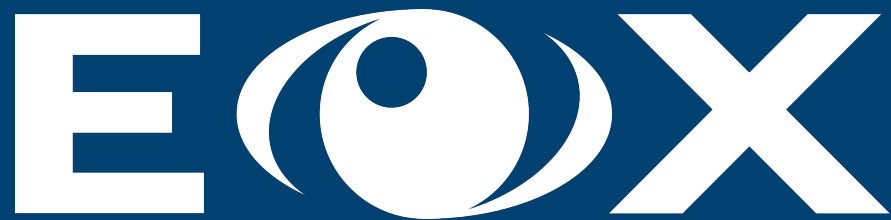
- [Configuration options in detail here](#)
- Central configs:
 - `workspaceChartsConfigMap`
References a config map containing HelmReleases which define the user workspace functionality
 - `s3Endpoint, s3Region`
Bucket server location
 - `harborUrl, harborUsername, harborPassword`
Access for workspace api to harbor (container registry)

Harbor (container registry)

- Installed via [helm release](#)
- Central configs:
 - `harborAdminPassword`
Change in interface after first install and configure in workspace api
 - `storageClass`
Set an appropriate storage class for all harbor services

Summary

- Open Standards (OGC, ISO)
- Free and Open Source (FOSS)
 - Core components
 - Reference implementation
 - Geospatial (FOSS4G)
- Best of breed Python geospatial components
- Download, configure, and run
 - <https://github.com/EOEPCA/eoepca>
- Open communities
- Open to contributions



Thanks for your attention!



EOX IT Services <https://eox.at>



EOfarm P.C. <http://eofarm.com>

EOEPCA Operators demo 2022-09-20



This work is licensed under a [Creative Commons Attribution-ShareAlike 4.0 International License](https://creativecommons.org/licenses/by-sa/4.0/).