A COMMON, OPEN SOURCE INTERFACE BETWEEN EARTH OBSERVATION DATA INFRASTRUCTURES AND FRONT-END APPLICATIONS

Deliverable 06

Version 1.0 from 2018/03/26

Two early prototype back-ends

Change history

Issue	Date	Author(s)	Description
0.1	2018/03/15	Gunnar Busch, EODC	First draft.
0.2	2018/03/19	openEO consortium	Revision of back-end driver descriptions.
0.3	2018/03/20	Alexander Jacob, EURAC	Updated introduction with programming language section and added graph to description of WCPS back-end driver. Typo correction throughout the document.
0.4	2018/03/22	Miha Kadunc, Sinergise	Reviewed the deliverable and revised the section on Sentinel Hub back-end
1.0	2018/03/26	Matthias Schramm, TU Wien	Last Review and creation of final version

For any clarifications please contact openEO@list.tuwien.ac.at.

Number of pages: **24**

Disclaimer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 776242. Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Table of Contents

List of Figures

List of Acronyms

API	Application Programming Interface
EO	Earth Observation
EODC	Earth Observation Data Centre
EURAC	Eurac Research
GIS	Geo Information System
GRASS	Geographic Resources Analysis Support System
HTTP	Hypertext Transfer Protocol
NDVI	Normalized Difference Vegetation Index
NFS	Network File System
OGC	Open Geospatial Consortium
openEO	Name of the project for which this report was written
PoC	Proof of Concept
POSIX	Portable Operating System Interface
REST	Representational State Transfer
WCPS	Web Coverage Processing Service
WCS	Web Coverage Service
WWU	Westphalian Wilhelms-University

1 Executive Summary

This document is intended to provide an overview of the achievements of the openEO deliverable “D4.2 Two early prototype back-end drivers (file-based, SciDB)“. The deliverable is a demonstrator and one of four deliverables (D3.1, D4.1, D4.2 and D5.1) in month 6 of the openEO project that together result in the milestone “M2 Proof of Concept (PoC)“. The overall aim of the PoC is to provide a first prototypical implementation of the openEO interface, which includes a python client, a first prototype of the Core API and two back-end prototypes.

2 Introduction

The first six months of the openEO project were efficiently used to develop back office drivers in parallel, that implement the openEO core API specification and provide API based access to available data and operations. Two weeks of intense collaboration in Belgium and Germany and bi-weekly video conferences, together with the Core API (WP3) and Client (WP5) developers, allowed close collaboration, coordination of common problems and finding common solutions. As already discussed in the project kick off meeting, the exact variation of implemented back-end drivers for D4.2, that were defined as one file-based and one SciDB driver, varied. The consortium decided to replace the development of a SciDB driver with a WCPS/rasdaman driver, given that none of the back-office project partners are currently using SciDB. While the implemented drivers diversified, the amount of currently available prototypical back-end driver implementations widely exceeds the proposed two back-end drivers.

The openEO proposal defines three different kinds of back-end categories that are intended to implement the openEO core API specification: File-based, mid-level and high-level back-offices. File-based back-ends provide access to raw data via POSIX, NFS or Object Storage. Early prototype drivers for file-based back-ends were implemented for Sentinel Hub (Sinergise) and OpenShift (EODC). Mid-level API back office drivers provide an interface between software systems that can utilise file-based data processing. Currently available prototypical mid-level drivers were implemented for GeoTrellis (VITO) and GRASS GIS (Mundialis). Furthermore, drivers for high-level back offices, that provide access to data and processes through API calls (e.g. as defined by OGC standards) were implemented for WCPS (EURAC) on top of rasdaman. Moreover, a back-end driver in R was developed by WWU to provide a lightweight and simplistic version of an openEO compliant back-end driver for local development and testing.

Apart from the three general categories of back-end types also a number of different programming languages have been used for implementing the drivers, ranging from Python over R and JavaScript to Java. These implementations provide a good starting point for any developer aiming to develop drivers in various back-ends and software development environments.

The following sections will give a short overview of available back-end driver prototypes, the current status and the technical specifications of the individual drivers. Interfaces that are currently implemented by the back-ends with respect to the version 0.0.2 of the openEO core API are listed for each driver. A complete overview of specified routes, that are included in the openEO Core API v0.0.2 is given below. Further details on the Core API can be found at https://open-eo.github.io/openeo-api/.

For each of the back-end driver prototypes, each subsection contains a specification of the respective capabilities. The following list gives a general description of the structure of these subsections:

Capabilities

- **API information:** */capabilities*
- **Output formats:** */capabilities/output_formats*
- **Web service types:** */capabilities/services*

EO Data Discovery

- **Data discovery:** */data*
- **Product information:** */data/<product_id>*

Process Discovery

- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*

User Defined Functions (UDF)

- **UDF runtimes:** */udf_runtimes*
- **UDF process description:** */udf_runtimes/<lang>/>udf_type>*

User Content

- **Jobs submitted by user**: */users/<user_id>/jobs*
- **Services submitted by user:** */users/<user_id>/services*
- **Credits available to user:** */users/<user_id>/credits*
- **Submit, retrieve process graphs of user:** */users/<user_id>/process_graphs*
- **Manage process graph of user:** */users/<user_id>/process_graphs/<process_graph_id>*
- **List user-uploaded files:** */users/<user_id>/files*
- **Manage user-uploaded files:** */users/<user_id>/files/<path>*

Authentication

- **Authentication of user:** */auth/login*
- **Register new user:** */auth/register*

Job Management

- **Submit new job:** */jobs*

- **Synchronous execution of job:** */execute*
- **Manage submitted job:** */jobs/<job_id>*
- **Subscribe to job updates:***/jobs/<job_id>/subscribe*
- **Running job in batch mode:** */jobs/<job_id>/queue*
- **Pause a job:** */jobs/<job_id>/pause*
- **Cancel a job:** */jobs/<job_id>/cancel*
- **Download links to job results:** */jobs/<job_id>/download*

Service Management

- **Publish a new service:** */services*
- **Manage services:** */services/<service_id>*

Instructions for installing and using the back-end drivers, e.g. for local development or testing, are included in the GitHub repositories of the individual driver. References to the GitHub repositories are given as this document is not intended to provide specific instructions on how to implement and run the drivers. These instructions are available in the “README“ files of the GitHub repositories. Some back-ends provide a publicly accessible URL to their openEO API implementation that can be used to execute HTTP request examples. URLs to the public endpoints, if available and example HTTP-requests, are given in the technical specifications and may be used to demonstrate the back-office prototypes. Specific API routes may be secured, and the user needs to request a token to access these routes. For the purpose of sending requests on secured routes and for any other questions concerning a specific back-end one may contact the back-office provider or the openEO project or work package coordinators.

3 Back-End Driver Demonstrators

3.1 GeoPySpark VITO

Technical Specification	
Platform	Geotrellis / GeoPySpark
Language	Python 3.5
GitHub Link	https://github.com/Open-EO/openeo-geopyspark-driver
Public Endpoint	Not yet available

3.1.1 Back-End Description

GeoTrellis is an open source Scala library that facilitates fast operations on raster data using Scala and Apache Spark. It enables distributed computing on large data sets using map algebra techniques and supports vector, raster and point cloud data. GeoTrellis provides RESTful endpoints and raster processing at web speeds (sub-second or less) [1]. The driver implements a direct (non-REST) version of the openEO client API on top of GeoPySpark. The back-end has been tested with Spark on a Yarn cluster and with Apache Accumulo as the tile storage back-end for Geotrellis. The openEO driver could, with minor modifications, be adapted to work with other Geotrellis back-ends, such as S3.

3.1.2 Current Capabilities

- **Data discovery:** */data*
- **Product information:** */data/<product_id>*
- **Synchronous execution of job:** */execute*
- **Download links to job results:** */jobs/<job_id>/download*
- **Services**: TMS (Tile Map Service)
- **Filters**: Bands, Date Range, Bounding Box
- **Operations**: NDVI, Min Time, Max Time

3.2 GRASS GIS Mundialis

Technical Specification	
Platform	GRASS GIS
Language	Python 3.5
GitHub Link	https://github.com/Open-EO/openeo-grassgis-driver
Public Endpoint	http://openeo.mundialis.de:5000

3.2.1 Back-End Description

The GRASS GIS driver implements the openEO Core API interface for the GRASS GIS as a Service (Actinia Core; available from https://github.com/mundialis/actinia_core) - a software solution for parallel, large scale geodata processing. It is a highly scalable REST interface to process geodata with the GRASS GIS in a distributed environment. GRASS GIS is a free and open source software package providing geospatial processing engines in a single integrated environment for raster, vector, and 3d-voxel processing as well as image processing capabilities [2, 3].

3.2.2 Current Capabilities

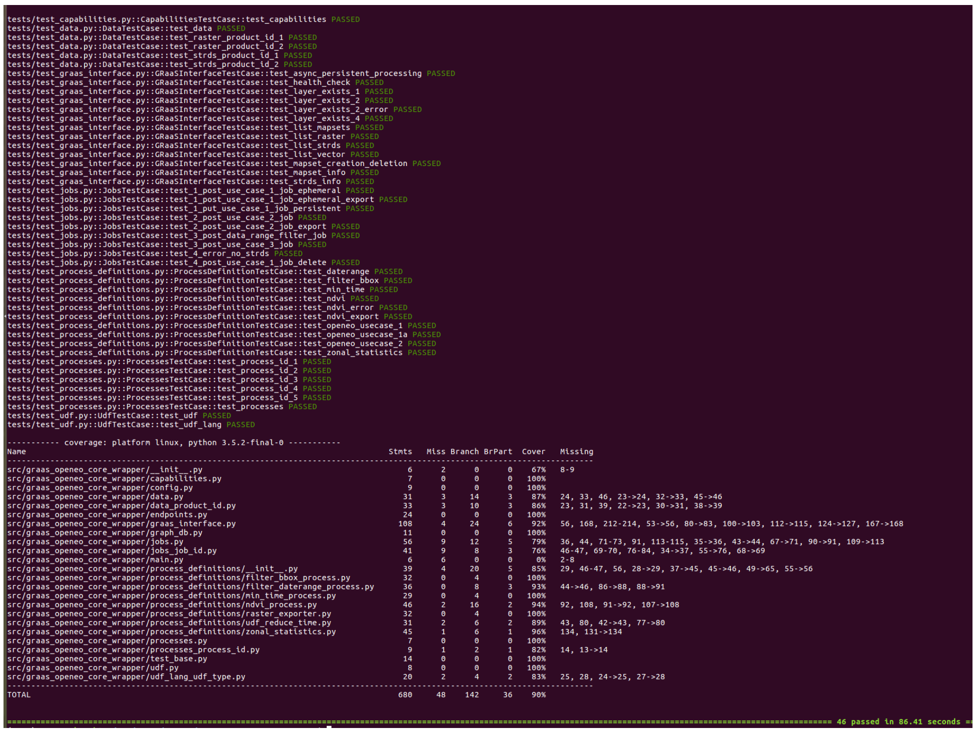
- **API information:** */capabilities*
- **Data discovery:** */data*
- **Product information:** */data/<product_id>*

- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*
- **UDF runtimes:** */udf_runtimes*
- **UDF process description:** */udf_runtimes/<lang>/>udf_type>*
- **Submit new job:** */jobs*
- **Manage submitted job:** */jobs/<job_id>*
- **Filters**: Bands, Date Range, Bounding Box
- **Operations**: NDVI, Min Time, Max Time, Zonal Statistics, Raster Exporter
- **UDFs**: Reduce Time

3.2.3 Example POST Request

```
{
  "process_graph": {
    "process_id": "min_time",
    "args": {
      "collections": [{
        "process_id": "NDVI",
        "args": {
          "collections": [
            {
              "process_id": "filter_daterange",
              "args": {
                "collections": [{
                  "process_id": "filter_bbox",
                  "args": {
                    "collections": [{"product_id": "LL.
                       sentinel2A_openeo_subset.strds.S2A_B04"}],
                    "bottom": 38.9,
                    "left": -4.8,
                    "right": -4.6,
                    "top": 39.1,
                    "ewres": 0.0001,
                    "nsres": 0.0001,
                    "srs": "EPSG:4326"
                  }
                }],
                "from": "2017-04-12 11:17:08",
                "to": "2017-09-04 11:18:26"
              }
            },
            {
              "process_id": "filter_daterange",
              "args": {
                "collections": [{
                  "process_id": "filter_bbox",
                  "args": {
                    "collections": [{
                    "product_id": "LL.sentinel2A_openeo_subset.strds.S2A_B08"}],
                    "bottom": 38.9,
                    "left": -4.8,
                    "right": -4.6,
                    "top": 39.1,
                    "ewres": 0.0001,
                    "nsres": 0.0001,
                    "srs": "EPSG:4326"
                  }
                }],
                "from": "2017-04-12 11:17:08",
                "to": "2017-09-04 11:18:26"
              }
            }],
          "red": "S2A_B04",
          "nir": "S2A_B08"
        }
      }]
    }
  }
}
```

3.2.4 Visualisation



```
tests/test_capabilities.py::CapabilitiesTestCase::test_capabilities PASSED
tests/test_data.py::DataTestCase::test_data PASSED
tests/test_data.py::DataTestCase::test_raster_product_id_1 PASSED
tests/test_data.py::DataTestCase::test_raster_product_id_2 PASSED
tests/test_data.py::DataTestCase::test_strds_product_id_1 PASSED
tests/test_data.py::DataTestCase::test_strds_product_id_2 PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_async_persistent_processing PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_health_check PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_layer_exists_1 PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_layer_exists_2 PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_layer_exists_2_error PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_layer_exists_4 PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_list_mapsets PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_list_raster PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_list_strds PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_list_vector PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_mapset_creation_deletion PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_mapset_info PASSED
tests/test_graas_interface.py::GRaaSInterfaceTestCase::test_strds_info PASSED
tests/test_jobs.py::JobsTestCase::test_1_post_use_case_1_job_ephemeral PASSED
tests/test_jobs.py::JobsTestCase::test_1_post_use_case_1_job_ephemeral_export PASSED
tests/test_jobs.py::JobsTestCase::test_1_put_use_case_1_job_persistent PASSED
tests/test_jobs.py::JobsTestCase::test_2_post_use_case_2_job PASSED
tests/test_jobs.py::JobsTestCase::test_2_post_use_case_2_job_export PASSED
tests/test_jobs.py::JobsTestCase::test_3_post_data_range_filter_job PASSED
tests/test_jobs.py::JobsTestCase::test_3_post_use_case_3_job PASSED
tests/test_jobs.py::JobsTestCase::test_4_error_no_strds PASSED
tests/test_jobs.py::JobsTestCase::test_4_post_use_case_1_job_delete PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_daterange PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_filter_bbox PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_min_time PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_ndvi PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_ndvi_error PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_ndvi_export PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_openeo_usecase_1 PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_openeo_usecase_1a PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_openeo_usecase_2 PASSED
tests/test_process_definitions.py::ProcessDefinitionTestCase::test_zonal_statistics PASSED
tests/test_processes.py::ProcessesTestCase::test_process_id_1 PASSED
tests/test_processes.py::ProcessesTestCase::test_process_id_2 PASSED
tests/test_processes.py::ProcessesTestCase::test_process_id_3 PASSED
tests/test_processes.py::ProcessesTestCase::test_process_id_4 PASSED
tests/test_processes.py::ProcessesTestCase::test_process_id_5 PASSED
tests/test_processes.py::ProcessesTestCase::test_processes PASSED
tests/test_udf.py::UdfTestCase::test_udf PASSED
tests/test_udf.py::UdfTestCase::test_udf_lang PASSED

----------- coverage: platform linux, python 3.5.2-final-0 -----------
Name                                                                     Stmts   Miss Branch BrPart  Cover   Missing
---------------------------------------------------------------------------------------------------------------------
src/graas_openeo_core_wrapper/__init__.py                                    6      2      0      0    67%   8-9
src/graas_openeo_core_wrapper/capabilities.py                                7      0      0      0   100%
src/graas_openeo_core_wrapper/config.py                                      9      0      0      0   100%
src/graas_openeo_core_wrapper/data.py                                       31      3     14      3    87%   24, 33, 46, 23->24, 32->33, 45->46
src/graas_openeo_core_wrapper/data_product_id.py                            33      3     10      3    86%   23, 31, 39, 22->23, 30->31, 38->39
src/graas_openeo_core_wrapper/endpoints.py                                  24      0      0      0   100%
src/graas_openeo_core_wrapper/graas_interface.py                           108      4     24      6    92%   56, 168, 212-214, 53->56, 80->83, 100->103, 112->115, 124->127, 167->168
src/graas_openeo_core_wrapper/graph_db.py                                   11      0      0      0   100%
src/graas_openeo_core_wrapper/jobs.py                                       56      9     12      5    79%   36, 44, 71-73, 91, 113-115, 35->36, 43->44, 67->71, 90->91, 109->113
src/graas_openeo_core_wrapper/jobs_job_id.py                                41      9      8      3    76%   46-47, 69-70, 76-84, 34->37, 55->76, 68->69
src/graas_openeo_core_wrapper/main.py                                        6      6      0      0     0%   2-8
src/graas_openeo_core_wrapper/process_definitions/__init__.py               39      4     20      5    85%   29, 46-47, 56, 28->29, 37->45, 45->46, 49->65, 55->56
src/graas_openeo_core_wrapper/process_definitions/filter_bbox_process.py    32      0      4      0   100%
src/graas_openeo_core_wrapper/process_definitions/filter_daterange_process.py 36    0      8      3    93%   44->46, 86->88, 88->91
src/graas_openeo_core_wrapper/process_definitions/min_time_process.py       29      0      4      0   100%
src/graas_openeo_core_wrapper/process_definitions/ndvi_process.py           46      2     16      2    94%   92, 108, 91->92, 107->108
src/graas_openeo_core_wrapper/process_definitions/raster_exporter.py        32      0      4      0   100%
src/graas_openeo_core_wrapper/process_definitions/udf_reduce_time.py        31      2      6      2    89%   43, 80, 42->43, 77->80
src/graas_openeo_core_wrapper/process_definitions/zonal_statistics.py       45      1      6      1    96%   134, 131->134
src/graas_openeo_core_wrapper/processes.py                                   7      0      0      0   100%
src/graas_openeo_core_wrapper/processes_process_id.py                        9      1      2      1    82%   14, 13->14
src/graas_openeo_core_wrapper/test_base.py                                  14      0      0      0   100%
src/graas_openeo_core_wrapper/udf.py                                         8      0      0      0   100%
src/graas_openeo_core_wrapper/udf_lang_udf_type.py                          20      2      4      2    83%   25, 28, 24->25, 27->28
---------------------------------------------------------------------------------------------------------------------
TOTAL                                                                      680     48    142     36    90%


========================================================================================== 46 passed in 86.41 seconds ==
```

Figure 3.1: openEO GRASS GIS driver test suite (openEO wrapper test)

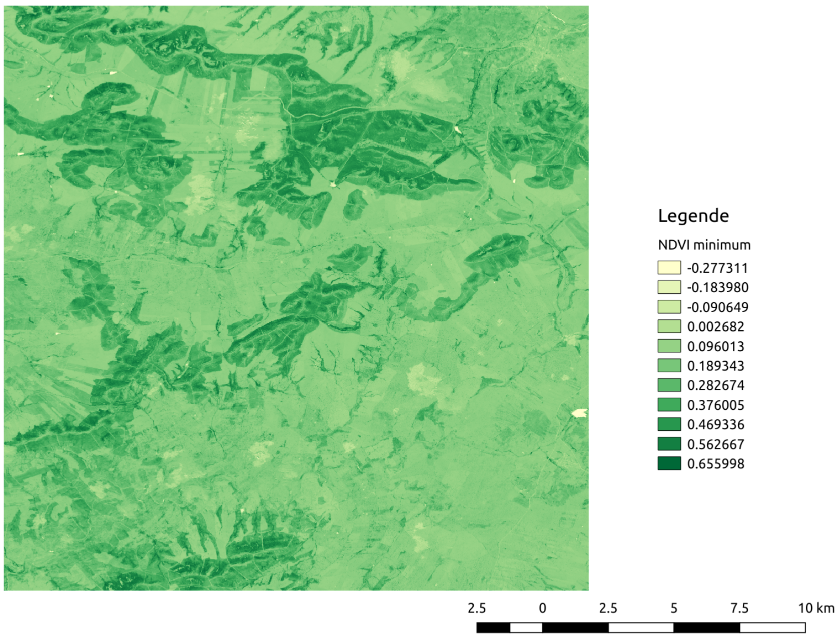


Figure 3.2: Resulting NDVI raster layer as GeoTiff

3.3 OpenShift EODC

Technical Specification	
Platform	OpenShift Origin / EODC Storage
Language	Python 3.5
GitHub Link	https://github.com/Open-EO/openeo-openshift-driver
Public Endpoint	http://openeo.eodc.eu

3.3.1 Back-End Description

OpenShift Origin is an open source and community driven Platform-as-a-Service (PaaS) by RedHat that is based on Kubernetes and Docker containers to provide cluster and container life-cycle management. OpenShift includes Kubernetes for container orchestration and offers multiple extensive layers on top of it to realise user management, container building, integrated networking and persistent storage [4]. The OpenShift cluster at EODC is hosted in an integrated environment that allows direct access to the file-based storage of EODC and enables to perform operations on the data by executing single Docker containers that have mounted persistent storage. Web services based on Flask provide the endpoints to the openEO API.

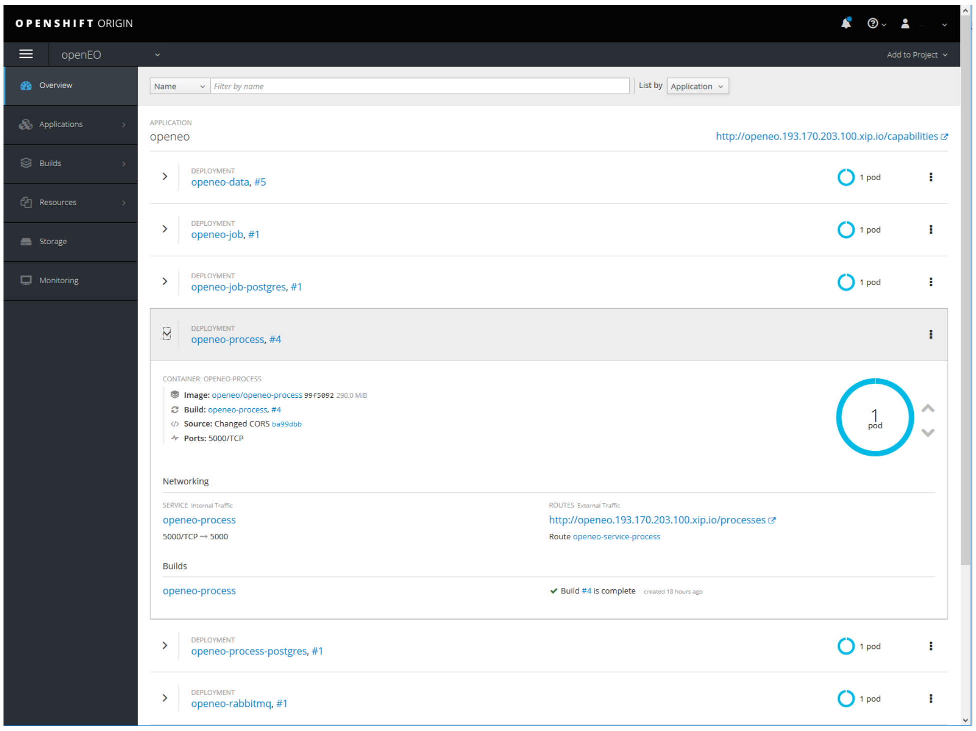
3.3.2 Current Capabilities

- **API information:** */capabilities*
- **Output formats:** */capabilities/output_formats*
- **Data discovery:** */data*
- **Product information:** */data/<product_id>*
- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*
- **Authentication of user:** */auth/login*
- **Submit new job:** */jobs*
- **Running job in batch mode:** */jobs/<job_id>/queue*
- **Manage submitted job:** */jobs/<job_id>*
- **Download links to job results:** */jobs/<job_id>/download*
- **Filters**: Bands, Date Range, Bounding Box
- **Operations**: NDVI, Min Time, Sentinel-2 Extraction

3.3.3 Example POST Request

```
{
  "process_graph":{
    "process_id":"min_time",
    "args":{
      "imagery":{
        "process_id":"NDVI",
        "args":{
          "imagery":{
            "process_id":"filter_daterange",
            "args":{
              "imagery":{
                "process_id":"filter_bbox",
                "args":{
                  "imagery":{
                    "product_id":"s2a_prd_msil1c"
                  },
                  "left":652000,
                  "right":672000,
                  "top":5161000,
                  "bottom":5181000,
                  "srs":"EPSG:32632"
                }
              },
              "from":"2017-01-01",
              "to":"2017-01-08"
            }
          },
          "red":"B04",
          "nir":"B08"
        }
      }
    }
  }
}
```

3.3.4 Visualisation



APPLICATION
openeo
http://openeo.193.170.203.100.xip.io/capabilities

DEPLOYMENT
openeo-data, #5
1 pod

DEPLOYMENT
openeo-job, #1
1 pod

DEPLOYMENT
openeo-job-postgres, #1
1 pod

DEPLOYMENT
openeo-process, #4

CONTAINER: OPENEO-PROCESS
Image: openeo/openeo-process 99f5092 290.0 MiB
Build: openeo-process, #4
Source: Changed CORS ba99dbb
Ports: 5000/TCP

1 pod

Networking

SERVICE Internal Traffic
openeo-process
5000/TCP → 5000

ROUTES External Traffic
http://openeo.193.170.203.100.xip.io/processes
Route openeo-service-process

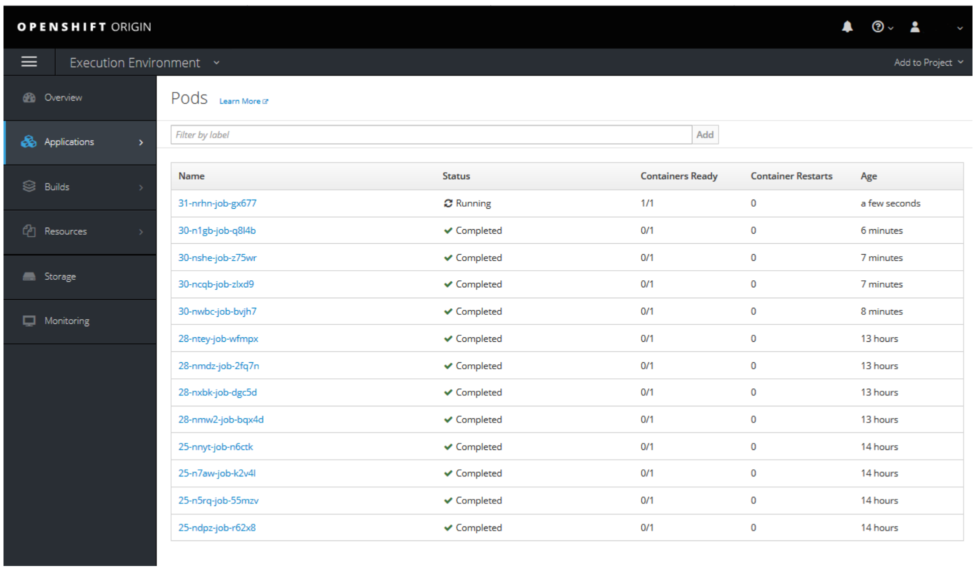
Builds

openeo-process
✓ Build #4 is complete created 18 hours ago

DEPLOYMENT
openeo-process-postgres, #1
1 pod

DEPLOYMENT
openeo-rabbitmq, #1
1 pod

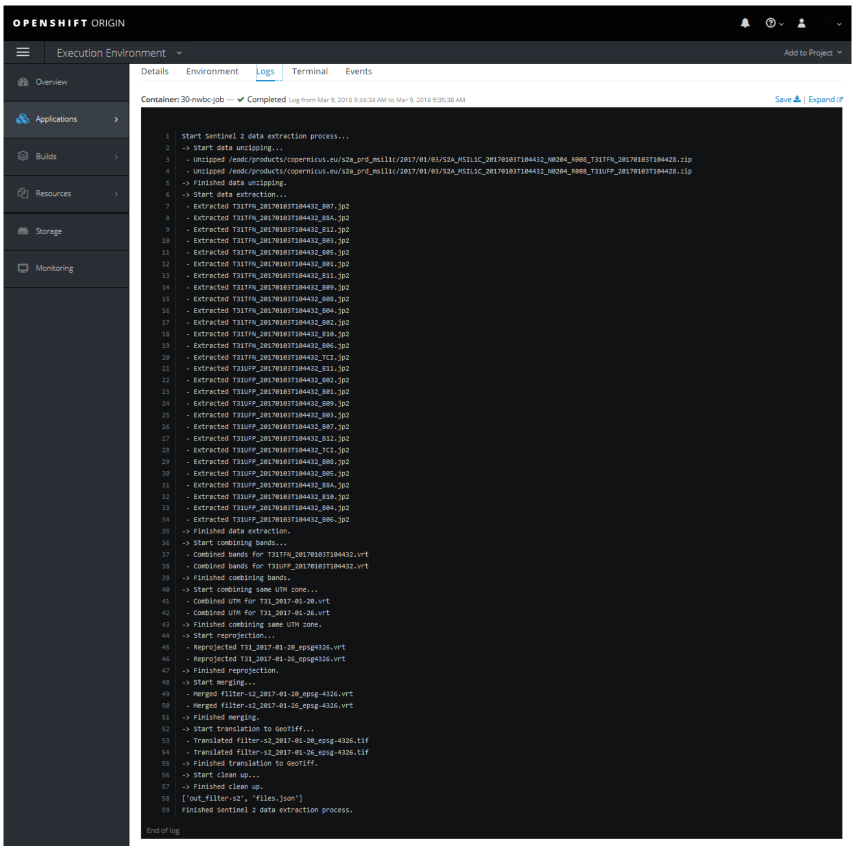
Figure 3.3: openEO service endpoints on OpenShift



Pods

Name	Status	Containers Ready	Container Restarts	Age
31-nrhn-job-gx677	Running	1/1	0	a few seconds
30-n1gb-job-q8l4b	✓ Completed	0/1	0	6 minutes
30-nshe-job-z75wr	✓ Completed	0/1	0	7 minutes
30-ncqb-job-zlxd9	✓ Completed	0/1	0	7 minutes
30-nwbc-job-bvjh7	✓ Completed	0/1	0	8 minutes
28-ntey-job-wfmpx	✓ Completed	0/1	0	13 hours
28-nmdz-job-2fq7n	✓ Completed	0/1	0	13 hours
28-nxbk-job-dgc5d	✓ Completed	0/1	0	13 hours
28-nmw2-job-bqx4d	✓ Completed	0/1	0	13 hours
25-nnyt-job-n6ctk	✓ Completed	0/1	0	14 hours
25-n7aw-job-k2v4l	✓ Completed	0/1	0	14 hours
25-n5rq-job-55mzv	✓ Completed	0/1	0	14 hours
25-ndpz-job-r62x8	✓ Completed	0/1	0	14 hours

Figure 3.4: Running and completed jobs submitted on /jobs



Container: 30-nwbc-job — ✓ Completed Log from Mar 9, 2018 9:34:34 AM to Mar 9, 2018 9:35:38 AM

```
Start Sentinel 2 data extraction process...
-> Start data unzipping...
 - Unzipped /eodc/products/copernicus.eu/s2a_prd_msil1c/2017/01/03/S2A_MSIL1C_20170103T104432_N0204_R008_T31TFN_20170103T104428.zip
 - Unzipped /eodc/products/copernicus.eu/s2a_prd_msil1c/2017/01/03/S2A_MSIL1C_20170103T104432_N0204_R008_T31UFP_20170103T104428.zip
-> Finished data unzipping.
-> Start data extraction...
 - Extracted T31TFN_20170103T104432_B07.jp2
 - Extracted T31TFN_20170103T104432_B8A.jp2
 - Extracted T31TFN_20170103T104432_B12.jp2
 - Extracted T31TFN_20170103T104432_B03.jp2
 - Extracted T31TFN_20170103T104432_B05.jp2
 - Extracted T31TFN_20170103T104432_B01.jp2
 - Extracted T31TFN_20170103T104432_B11.jp2
 - Extracted T31TFN_20170103T104432_B09.jp2
 - Extracted T31TFN_20170103T104432_B08.jp2
 - Extracted T31TFN_20170103T104432_B04.jp2
 - Extracted T31TFN_20170103T104432_B02.jp2
 - Extracted T31TFN_20170103T104432_B10.jp2
 - Extracted T31TFN_20170103T104432_B06.jp2
 - Extracted T31TFN_20170103T104432_TCI.jp2
 - Extracted T31UFP_20170103T104432_B11.jp2
 - Extracted T31UFP_20170103T104432_B02.jp2
 - Extracted T31UFP_20170103T104432_B01.jp2
 - Extracted T31UFP_20170103T104432_B09.jp2
 - Extracted T31UFP_20170103T104432_B03.jp2
 - Extracted T31UFP_20170103T104432_B07.jp2
 - Extracted T31UFP_20170103T104432_B12.jp2
 - Extracted T31UFP_20170103T104432_TCI.jp2
 - Extracted T31UFP_20170103T104432_B08.jp2
 - Extracted T31UFP_20170103T104432_B05.jp2
 - Extracted T31UFP_20170103T104432_B8A.jp2
 - Extracted T31UFP_20170103T104432_B10.jp2
 - Extracted T31UFP_20170103T104432_B04.jp2
 - Extracted T31UFP_20170103T104432_B06.jp2
-> Finished data extraction.
-> Start combining bands...
 - Combined bands for T31TFN_20170103T104432.vrt
 - Combined bands for T31UFP_20170103T104432.vrt
-> Finished combining bands.
-> Start combining same UTM zone...
 - Combined UTM for T31_2017-01-20.vrt
 - Combined UTM for T31_2017-01-26.vrt
-> Finished combining same UTM zone.
-> Start reprojection...
 - Reprojected T31_2017-01-20_epsg4326.vrt
 - Reprojected T31_2017-01-26_epsg4326.vrt
-> Finished reprojection.
-> Start merging...
 - Merged filter-s2_2017-01-20_epsg-4326.vrt
 - Merged filter-s2_2017-01-26_epsg-4326.vrt
-> Finished merging.
-> Start translation to GeoTiff...
 - Translated filter-s2_2017-01-20_epsg-4326.tif
 - Translated filter-s2_2017-01-26_epsg-4326.tif
-> Finished translation to GeoTiff.
-> Start clean up...
-> Finished clean up.
['out_filter-s2', 'files.json']
Finished Sentinel 2 data extraction process.
```

End of log

Figure 3.5: Result log of a finished data extraction job

3.4 R Back-End WWU

Technical Specification	
Platform	R / plumber
Language	R
GitHub Link	https://github.com/Open-EO/openeo-r-backend
Public Endpoint	Not yet available

3.4.1 Back-End Description

The R back-end is a reference implementation for the openEO core API as a proof-of-concept written in R, utilising the *plumber* package as a lightweight web server. The web server is not final and in terms of security aspects not optimised. The goal of this package is to provide a simplistic version of local server back-end compliant with openEO [5].

3.4.2 Current Capabilities

- **API information:** */capabilities*
- **Output formats:** */capabilities/output_formats*

- **Data discovery:** */data*
- **Product information:** */data/<product_id>*
- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*
- **Jobs submitted by user**: */users/<user_id>/jobs*
- **Submit, retrieve process graphs of user:** */users/<user_id>/process_graphs*
- **Manage process graph of user:** */users/<user_id>/process_graphs/<process_graph_id>*
- **List user-uploaded files:** */users/<user_id>/files*
- **Manage user-uploaded files:** */users/<user_id>/files/<path>*
- **Authentication of user:** */auth/login*
- **Submit new job:** */jobs*
- **Download links to job results:** */jobs/<job_id>/download*
- **Synchronous execution of job:** */execute*

3.4.3 Example POST Request

```
{
  "process_graph": {
    "process_id": "find_min",
    "args": {
      "imagery": {
        "process_id": "calculate_ndvi",
        "args": {
          "imagery": {
            "process_id": "filter_daterange",
            "args": {
              "imagery": {
                "product_id": "sentinel2_subset"
              },
              "from": "2017-04-01",
              "to": "2017-05-01"
            }
          },
          "nir": 8,
          "red": 4
        }
      }
    }
  },
  "output": {
    "format": "GTiff"
  }
}
```

```
{
  "process_graph": {
    "process_id": "zonal_statistics",
    "args": {
      "imagery": {
        "process_id": "filter_daterange",
        "args": {
          "imagery": {
            "process_id": "filter_bands",
            "args": {
              "imagery": {
                "product_id": "sentinel2_subset"
              },
              "bands": "8"
            }
          },
          "from": "2017-04-01",
          "to": "2017-07-01"
        }
      },
      "regions": "/users/me/files/polygons.geojson",
      "func": "median"
    }
  },
  "output": {
    "format": "GPKG"
  }
}
```

3.5 Sentinel Hub Sinergise

Technical Specification	
Platform / Framework	Sentinel Hub / NodeJS
Programming Language	JavaScript
GitHub Link	https://github.com/Open-EO/openeo-sentinelhub-backend
Public Endpoint	Not yet available

3.5.1 Back-End Description

Sentinel Hub (https://www.sentinel-hub.com/) is a service-oriented platform that aims to facilitate processing and exploitation of satellite imagery by providing efficient access to several Earth observation data sets via easy-to-integrate web services.

The main features of the system are: automated archiving process, ensuring that data are always synchronised with official sources; full resolution access over web services, with automatic re-projection, mosaicking, ortho-rectification and other standard processing; user-defined scripts to define spectral band expressions and other remote sensing products; multi-temporal processing for on-the-fly computation of user-defined algorithms on time-series; service for statistical analysis of a time-series over an area or point of choice; and APIs for advanced feature integration.

Sentinel Hub works with original EO data, avoiding intensive pre-processing and additional storage for processed tiles. Some of the data accessible through Sentinel Hub are stored as AWS Public Datasets[1] (Sentinel-1, Sentinel-2, Landsat 8, MODIS) and some are hosted on CloudFerro EO Cloud[2] (Sentinel-1, Sentinel-2, Sentinel-3, ENVISAT, ESA's archive of Landsat 5, 7 and 8) [6].

3.5.2 Current Capabilities

- **API information:** */capabilities*
- **Web service types:** */capabilities/services*
- **Data discovery:** */data*
- **Product information:** */data/<product_id>*
- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*
- **Jobs submitted by user:** */users/<user_id>/jobs*
- **Services submitted by user:** */users/<user_id>/services*
- **Submit new job:** */jobs*
- **Publish a new service:** */services*
- **Manage services:** */services/<service_id>*
- **Filters:**: Date Range
- **Operations**: NDVI, Min Time, Max Time

[1] https://aws.amazon.com/earth/
[2] https://finder.eocloud.eu/

3.5.3 Example POST Request

```json
{
  "process_graph": {
    "process_id": "min_time",
    "args": {
      "imagery" : {
        "process_id": "NDVI",
        "args": {
          "imagery": {
            "process_id": "filter_daterange",
            "args": {
              "imagery": {
                "product_id":"Sentinel2A-L1C"
              },
              "from": "2017-01-01",
              "to": "2017-01-31"
            }
          },
          "red": "B04",
          "nir": "B08"
        }
      }
    }
  }
}
```

3.5.4 Multi-temporal Processing Script Generated by the Back-end

```javascript
const findMinIndex = arr => {
  let min = Infinity;
  let minIdx = -1;
  for (var i = 0; i < arr.length; i++) {
    let cur = arr[i];
    if (cur < min) {
      min = cur;
      minIdx = i;
    }
  }
  return minIdx;
};
const findMin = arr=> arr[findMinIndex(arr)];
const index = (a, b) => (a - b) / (a + b);
const dateRangeFilter = (from, to) => {
    return (scene => from < scene.date.getTime() && scene.date.getTime() < to);
};
function setup(dss) {
  setInputComponents([dss.B04,dss.B08]);
  setOutputComponentCount(1);
}

function filterScenes(scenes, inputMetadata) {
  scenes = scenes.filter(
    dateRangeFilter(Date.parse('2017-01-01'),Date.parse('2017-01-31'))
  );
  return scenes;
}

function eval(samples, scenes) {
  samples = samples.map(s => index(s.B08, s.B04));
  samples = [findMin(samples)];
  return samples;
}
```

3.6 WCPS EURAC

Technical Specification	
Platform	rasdaman 9.5
Language	Java
GitHub Link	https://github.com/Open-EO/openeo-wcps-driver
Public Endpoint	http://saocompute.eurac.edu/openEO_WCPS_Driver/openeo/

3.6.1 Back-End Description

WCS (Web Coverage Service) is an interface standard by the OGC (Open Geospatial Consortium) that supports the retrieval of geospatial data as coverages representing space and time-varying phenomena. The coverage data of a WCS is returned in forms that are useful for client-side rendering, as input into scientific models, and for other clients [7]. WCPS (Web Coverage Processing Service) extends this standard by defining a language that is capable of processing and retrieval of multi-dimensional geospatial coverages that represent images, sensor observations or statistical data. The WCPS language is independent from any particular request and response encoding, as no concrete request/response protocol is specified by the standard. For setting up a WCPS instance, therefore, a separate, additional specification establishing the concrete protocol is required. This allows embedding of WCPS into different target service frameworks [8]. The openEO WCPS driver is built on top of rasdaman ("raster data manager"), an Array Database that allows the storing and querying of multi-dimensional arrays. Rasdaman can process arrays residing in file system directories as well as in databases. The developed driver itself however is operational with all WCPS compliant web services.

3.6.2 Current Capabilities

- **API information:** */capabilities*
- **Output formats:** */capabilities/output_formats*
- **Data discovery:** */data*
- **Product information:** */data/<product_id>*

- **Process discovery:** */processes*
- **Process information:** */processes/<process_id>*
- **Authentication of user:** */auth/login*
- **Submit new job:** */jobs*
- **Synchronous execution of job:** */execute*
- **Manage submitted job:** */jobs/<job_id>*
- **Running job in batch mode:** */jobs/<job_id>/queue*
- **Download links to job results:** */jobs/<job_id>/download*
- **Filters**: Bands, Date Range, Bounding Box
- **Operations**: NDVI, Min Time, Max Time

3.6.3 Example POST Request

```
{
  "process_graph":{
    "process_id":"max_time",
    "args":{
      "imagery":{
        "process_id":"NDVI",
        "args":{
          "imagery":{
            "process_id":"filter_daterange",
            "args":{
              "imagery":{
                "process_id":"filter_bbox",
                "args":{
                  "imagery":{
                    "product_id":"S2_L2A_T32TPS_20M"
                  },
                  "left":652000,
                  "right":672000,
                  "top":5161000,
                  "bottom":5181000,
                  "srs":"EPSG:32632"
                }
              },
              "from":"2017-01-01",
              "to":2017-01-31"
            }
          },
                "red":"B04",
                "nir":"B8A"
            }
        }
    }
  }
}
```

3.6.4 Visualisation

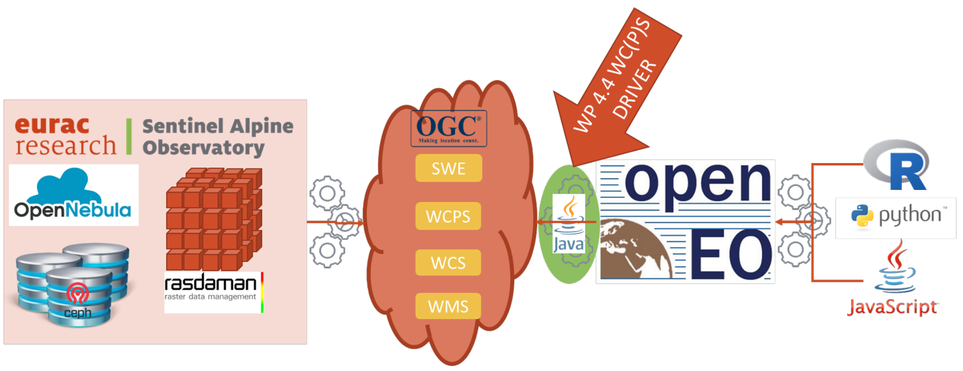


Figure 3.6: Overview of the WCPS driver in the context of a high level backend inside the Eurac Research Sentinel Alpine Observatory

4 Outlook

The Proof of Concept demonstrates the feasibility of openEO core functionality. These features, that are crucial for the success of the project, were implemented on the basis of investigations in standardising interfaces connecting the back-ends with the clients. These developed driver implementations serve as a basis for further extending the current features in accordance with the Core API specification. Major functionality like user management and advanced job processing and monitoring are yet to be implemented. Therefore, from month 7, the back-end providers will work towards the implementation of the first full API integration, which will be available in month 20. The proof of concept itself, together with the user requirements meeting and the hackathon in month 9 are serving as continued feedback and validation streams to improve openEO and assure that the realisation meets the user and back-end requirements.

5 References

[1] “What is GeoTrellis? GeoTrellis 1.0.0 documentation,” accessed: 2018-03-15. [Online]. Available: https://geotrellis.readthedocs.io/en/latest/

[2] “GRASS GIS OSGeo-Live 11.0 Documentation,” accessed: 2018-03-15. [Online]. Available: https://live.osgeo.org/en/overview/grass_overview.html

[3] M. Neteler, M. H. Bowman, M. Landa, and M. Metz, “GRASS GIS: A multi-purpose open source GIS,” *Environmental Modelling & Software*, vol. 31, pp. 124–130, May 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364815211002775

[4] “OpenShift Origin Documentation,” accessed: 2018-03-15. [Online]. Available: https://docs.openshift.org/

[5] “openEO Backend in R for proof-of-concept,” accessed: 2018-03-15. [Online]. Available: https://github.com/Open-EO/openeo-r-backend

[6] “Sentinel Hub - The Next Generation Satellite Imagery Service,” accessed: 2018-03-15. [Online]. Available: https://www.sentinel-hub.com/sites/default/files/about.pdf

[7] OGC, *OGC WCS OGC WCS 2.0 Interface Standard - Core, version 2.0.1; 09-110r4*. Open Geospatial Consortium, 2012. [Online]. Available: http://www.opengeospatial.org/standards/wcs

[8] ——, *OGC WCS OpenGIS Web Coverage Processing Service (WCPS) Language Interface Standard; 08-068r2*. Open Geospatial Consortium, 2009. [Online]. Available: http://www.opengeospatial.org/standards/wcps