eurac research

Interoperable Workflow Efficiency: Exploring the Integration of OpenEO, CWL , and EOEPCA for Seamless Data Processing and Modeling

Juraj Zvolenský, Piero Campalani, Michele Claus, Iacopo Federico Ferrario, Alexander Jacob

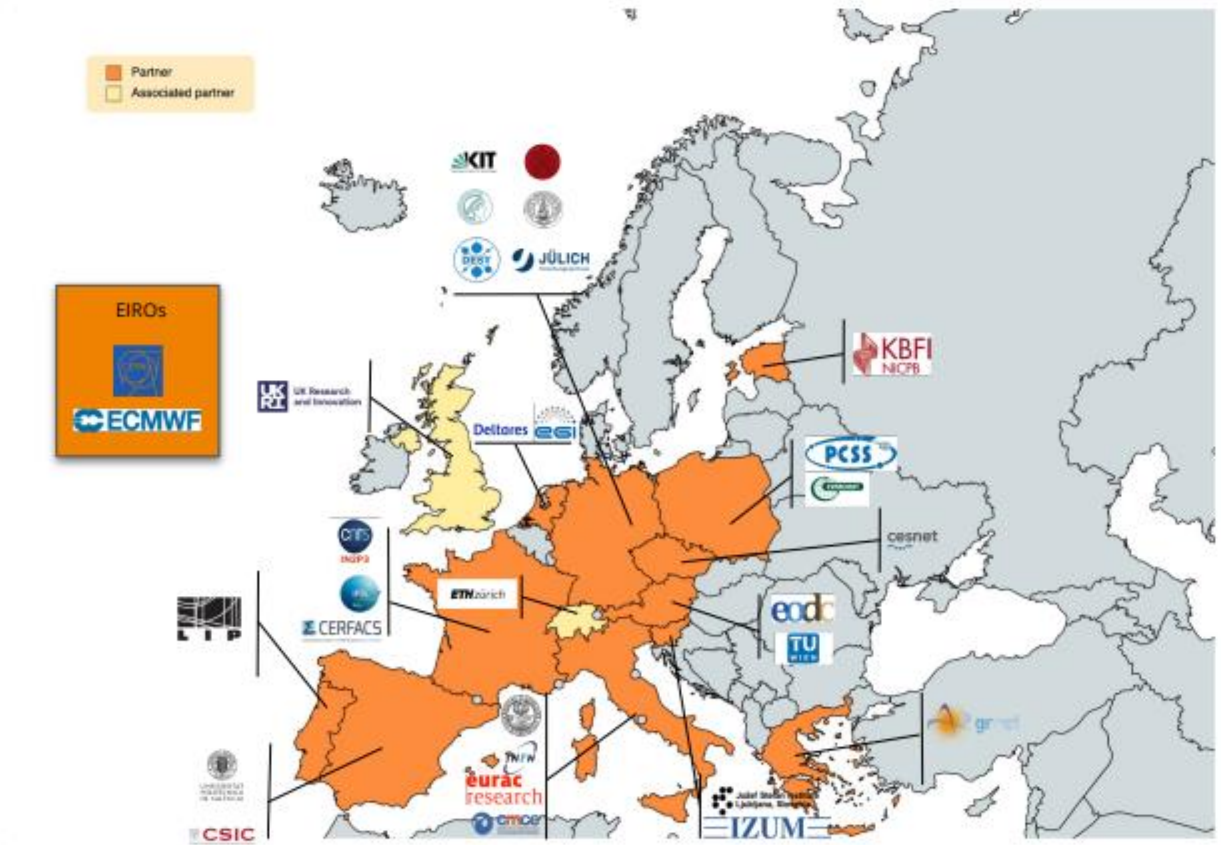
EGI 2024

Agenda

1. OpenEO
2. Interoperable workflows with CWL
3. EOEPCA ZOO-Project-DRU
4. Towards integration
5. Use case example

InterTwin (2022-2025)

An interdisciplinary Digital Twin Engine for science



Source: Andrea Manzi, EuroGeo Workshop 2023

interTwin

egi Advanced Computing for Research

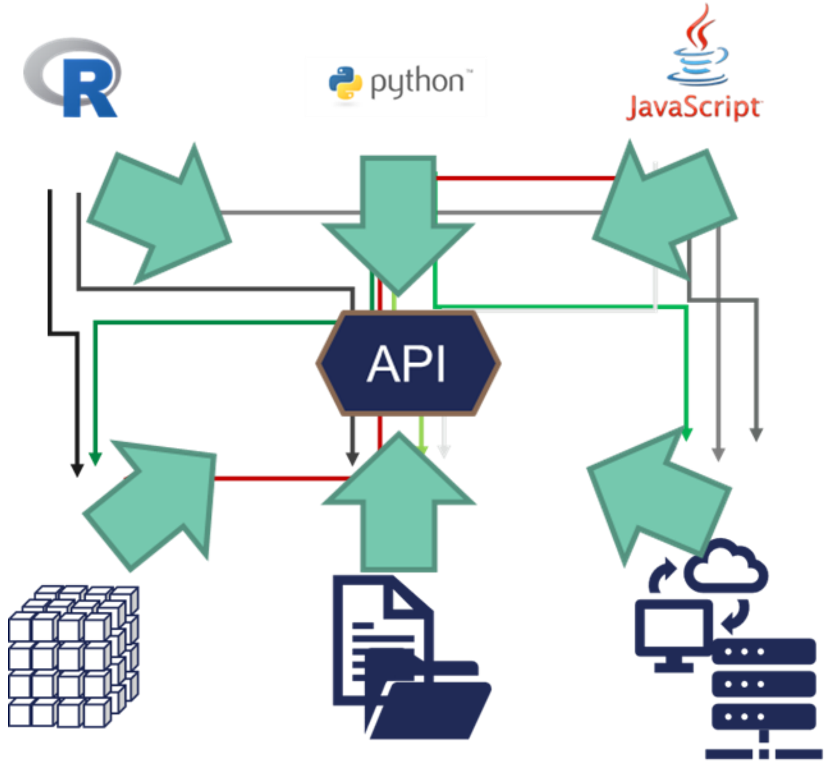
eurac
research

OpenEO

OpenEO API

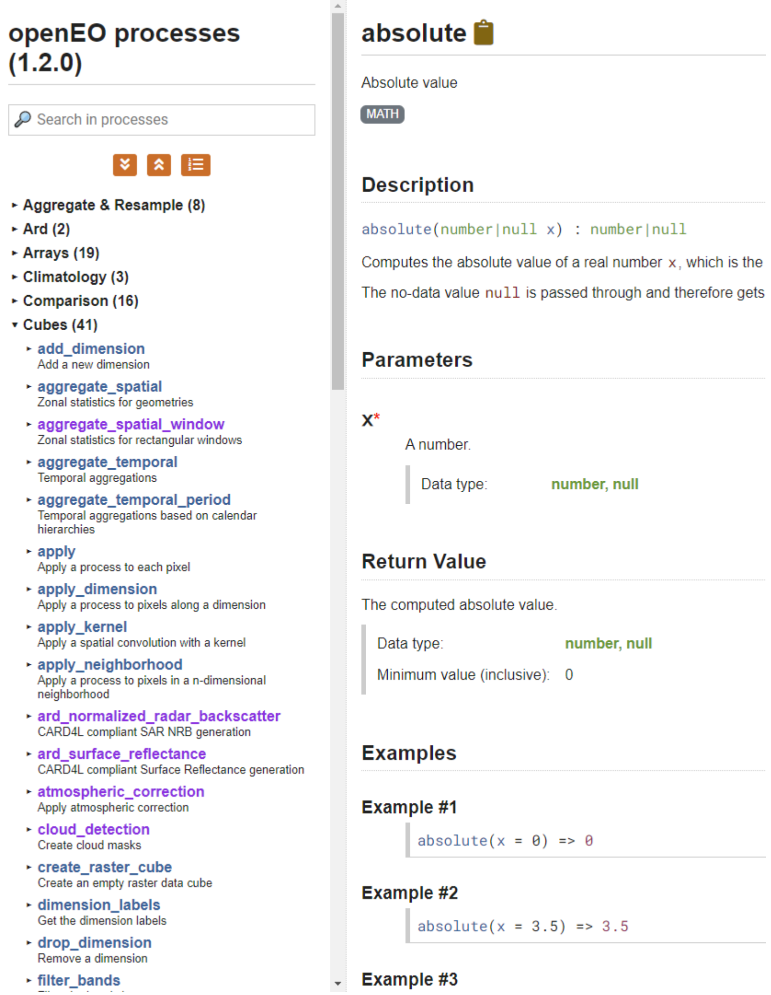
- OpenAPI specification

- Data discovery
- Auth
- Data Processing
- Workflow management
- Data export



OpenEO Processes

- Pre-defined processes
  - processes.openeo.org
  - JSON Schema
- User-defined processes
  - Combining existing processes
- User-defined functions
  - Run custom code (Python, R)
  - Several runtimes available
  - Docker images



openEO processes (1.2.0)

Search in processes

- Aggregate & Resample (8)
- Ard (2)
- Arrays (19)
- Climatology (3)
- Comparison (16)
- Cubes (41)
  - add_dimension
    Add a new dimension
  - aggregate_spatial
    Zonal statistics for geometries
  - aggregate_spatial_window
    Zonal statistics for rectangular windows
  - aggregate_temporal
    Temporal aggregations
  - aggregate_temporal_period
    Temporal aggregations based on calendar hierarchies
  - apply
    Apply a process to each pixel
  - apply_dimension
    Apply a process to pixels along a dimension
  - apply_kernel
    Apply a spatial convolution with a kernel
  - apply_neighborhood
    Apply a process to pixels in a n-dimensional neighborhood
  - ard_normalized_radar_backscatter
    CARD4L compliant SAR NRB generation
  - ard_surface_reflectance
    CARD4L compliant Surface Reflectance generation
  - atmospheric_correction
    Apply atmospheric correction
  - cloud_detection
    Create cloud masks
  - create_raster_cube
    Create an empty raster data cube
  - dimension_labels
    Get the dimension labels
  - drop_dimension
    Remove a dimension
  - filter_bands

absolute

Absolute value

MATH

Description

`absolute(number|null x) : number|null`

Computes the absolute value of a real number `x`, which is the

The no-data value `null` is passed through and therefore gets

Parameters

x*

A number.

Data type: number, null

Return Value

The computed absolute value.

Data type: number, null

Minimum value (inclusive): 0

Examples

Example #1

```
absolute(x = 0) => 0
```

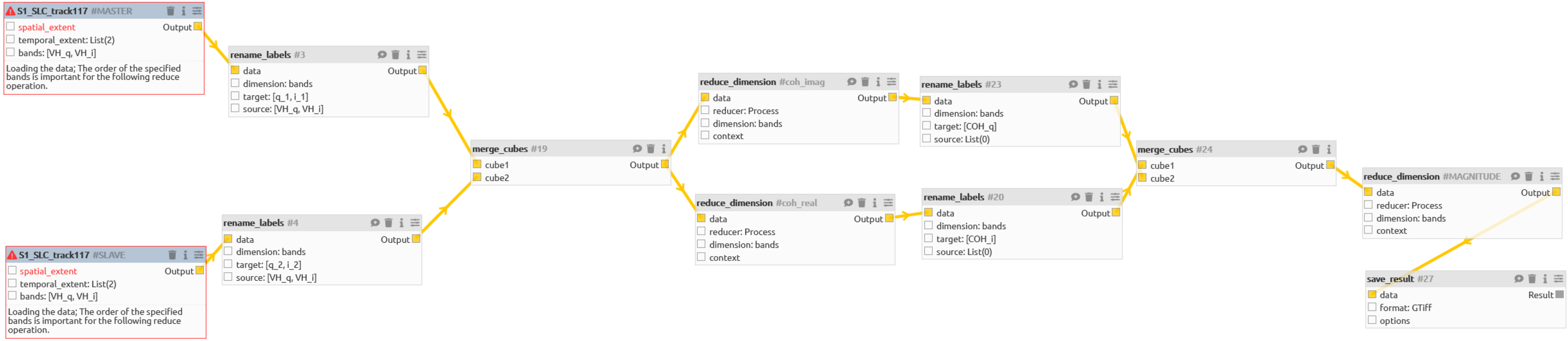
Example #2

```
absolute(x = 3.5) => 3.5
```

Example #3

EGI 2024

OpenEO Processes Graph



S1_SLC_track117 #MASTER

- □ spatial_extent — Output
- □ temporal_extent: List(2)
- □ bands: [VH_q, VH_i]

Loading the data; The order of the specified bands is important for the following reduce operation.

S1_SLC_track117 #SLAVE

- □ spatial_extent — Output
- □ temporal_extent: List(2)
- □ bands: [VH_q, VH_i]

Loading the data; The order of the specified bands is important for the following reduce operation.

rename_labels #3

- data — Output
- □ dimension: bands
- □ target: [q_1, i_1]
- □ source: [VH_q, VH_i]

rename_labels #4

- data — Output
- □ dimension: bands
- □ target: [q_2, i_2]
- □ source: [VH_q, VH_i]

merge_cubes #19

- cube1 — Output
- cube2

reduce_dimension #coh_imag

- data — Output
- □ reducer: Process
- □ dimension: bands
- □ context

reduce_dimension #coh_real

- data — Output
- □ reducer: Process
- □ dimension: bands
- □ context

rename_labels #23

- data — Output
- □ dimension: bands
- □ target: [COH_q]
- □ source: List(0)

rename_labels #20

- data — Output
- □ dimension: bands
- □ target: [COH_i]
- □ source: List(0)

merge_cubes #24

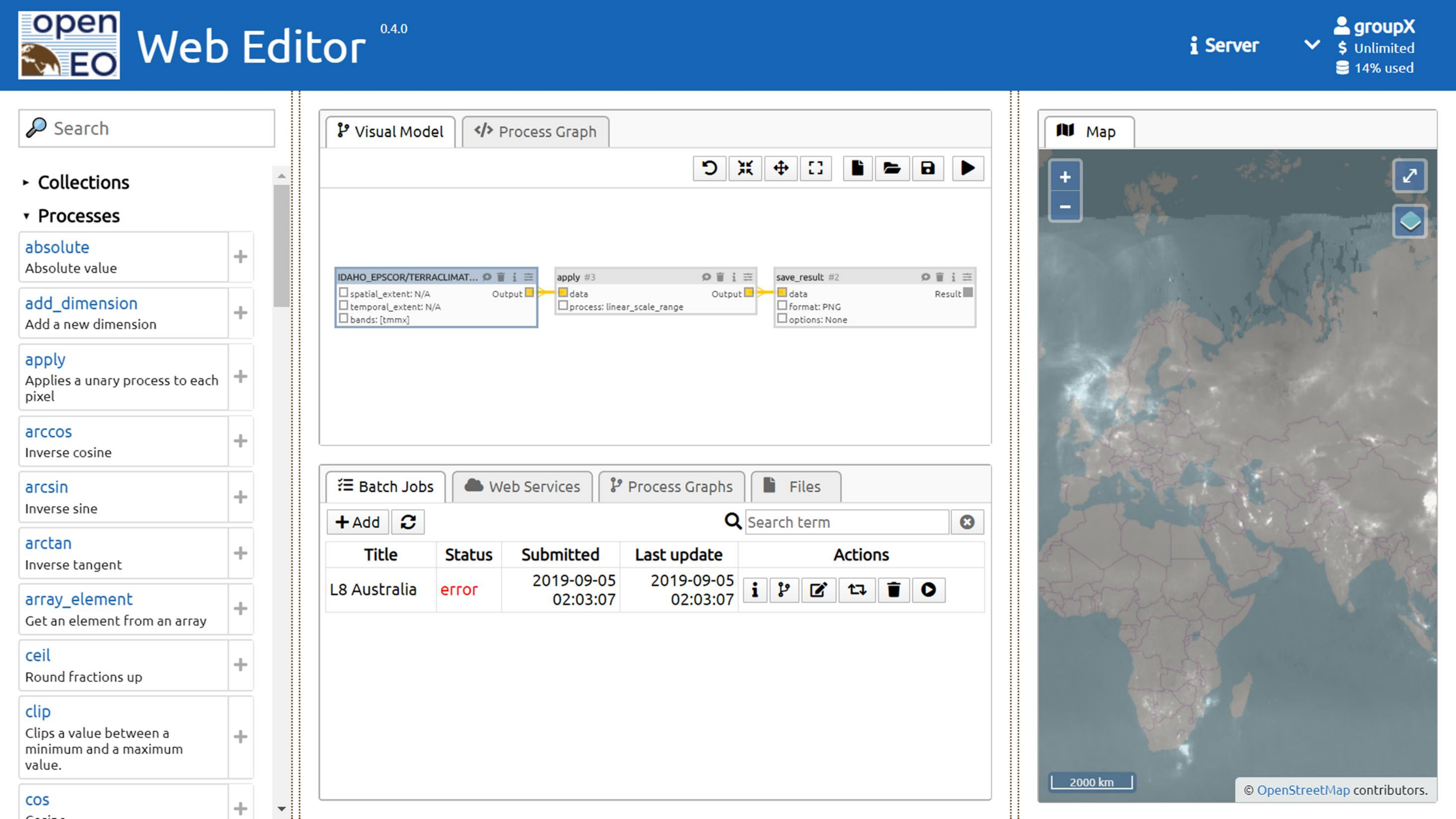
- cube1 — Output
- cube2

reduce_dimension #MAGNITUDE

- data — Output
- □ reducer: Process
- □ dimension: bands
- □ context

save_result #27

- data — Result
- □ format: GTiff
- □ options



absolute
Absolute value

add_dimension
Add a new dimension

apply
Applies a unary process to each pixel

arccos
Inverse cosine

arcsin
Inverse sine

arctan
Inverse tangent

array_element
Get an element from an array

ceil
Round fractions up

clip
Clips a value between a minimum and a maximum value.

cos

IDAHO_EPSCOR/TERRACLIMAT...
□ spatial_extent: N/A
□ temporal_extent: N/A
□ bands: [tmmx]
Output

apply #3
data
□ process: linear_scale_range
Output

save_result #2
data
□ format: PNG
□ options: None
Result

Title	Status	Submitted	Last update	Actions
L8 Australia	error	2019-09-05 02:03:07	2019-09-05 02:03:07	

eurac research

Common Workflow Language (CWL)

COMMON WORKFLOW LANGUAGE

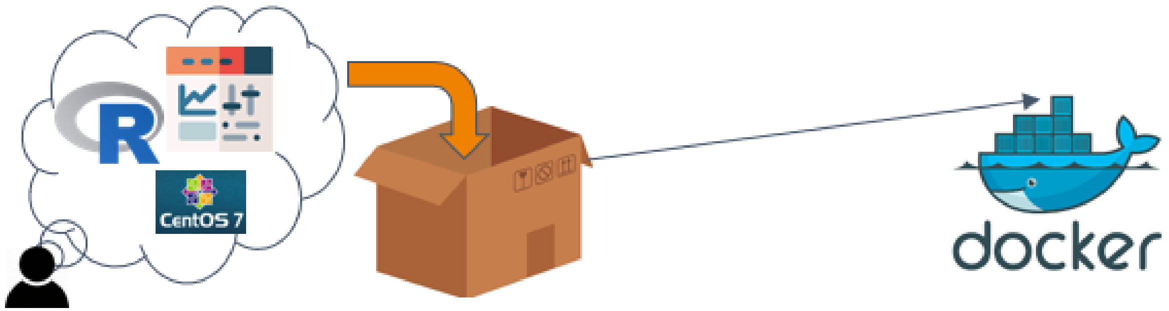
EGI 2024

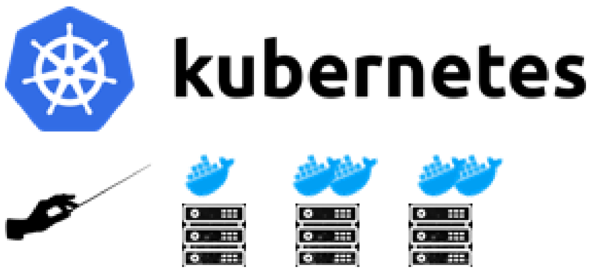
CWL

- Common Workflow Language (CWL) is an **open standard** for describing how to run command line tools and connect them **to create workflows**.

- Using CWL, it is easy to scale complex data analysis and machine learning workflows from a single developer's laptop up to massively parallel cluster, cloud and high-performance computing environments.

OGC Application Package





Container: unit of software, packaging a given code and all its dependencies.

Docker: platform designed to create and manage containers, repos of containers, etc.

Container orchestration tool to execute containers across computing resources.

- ➢ By packaging our applications and wrapping them in CWL we can automatically deploy across various processing infrastructures
- ➢ Container orchestration allows us to scale across the available infrastructure
- ➢ Well defined data input & output through STAC collections

Earth
Observation
Exploitation
Platform
Common
Architecture

EOEPCA
BETTER ACCESS TO EARTH OBSERVATION

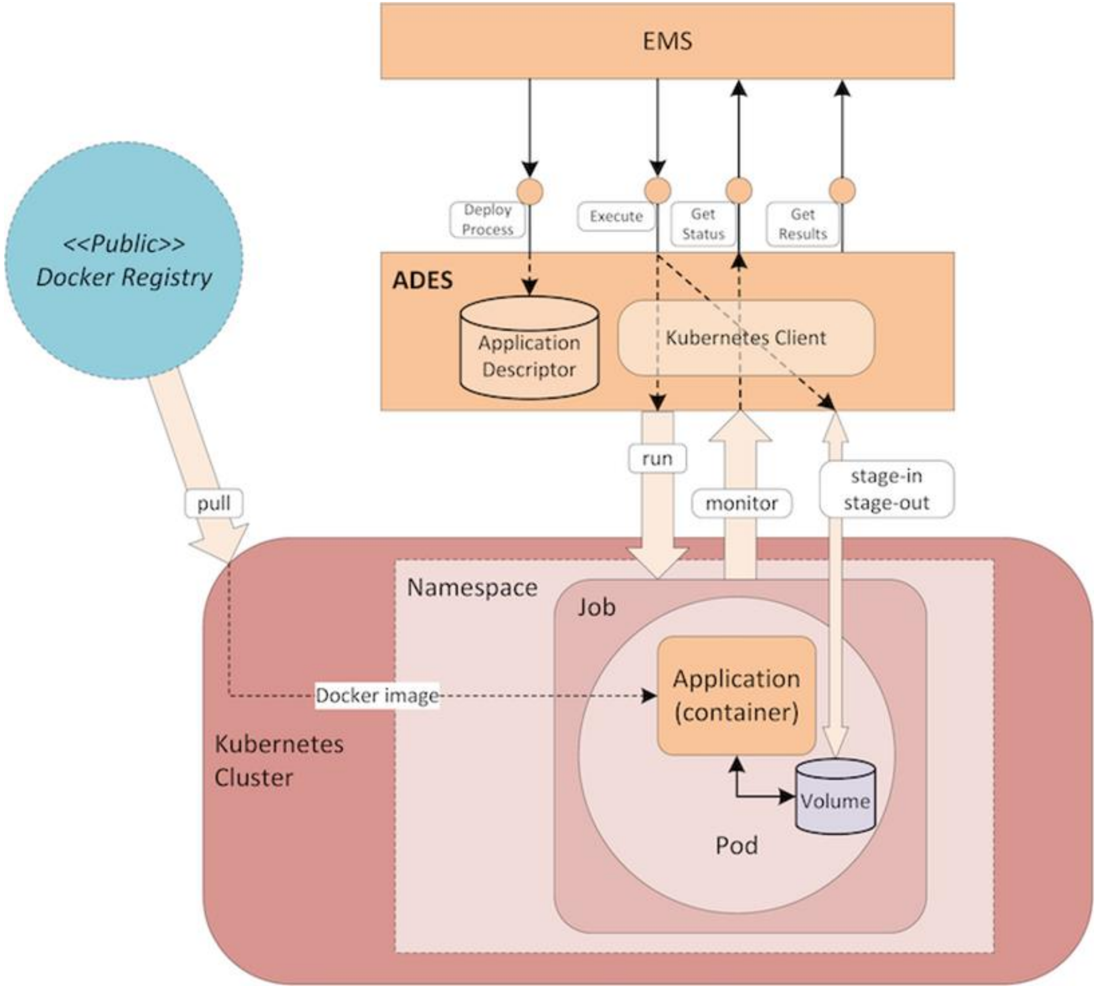
ZOO-Project-DRU

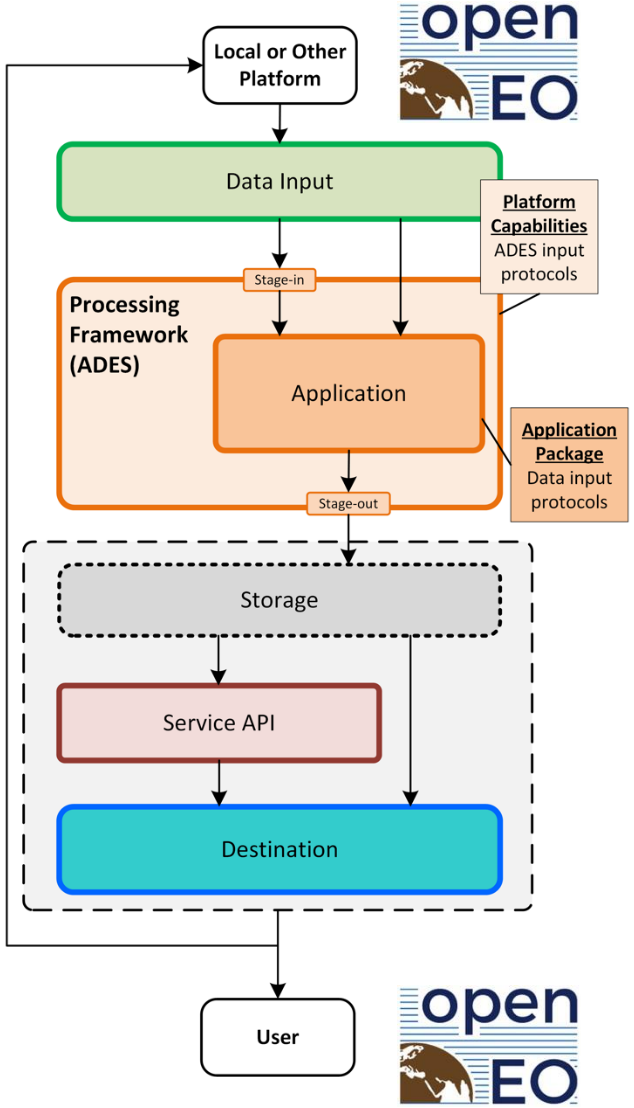
- ➢Developed as part of the Earth Observation Exploitation Platform Common Architecture (EOEPCA)

- ➢The ZOO-Project is an open-source processing platform

- ➢ZOO-Kernel, a server implementation of the Web Processing Service (WPS) (1.0.0 and 2.0.0) and the OGC API - Processes standards published by the OGC.

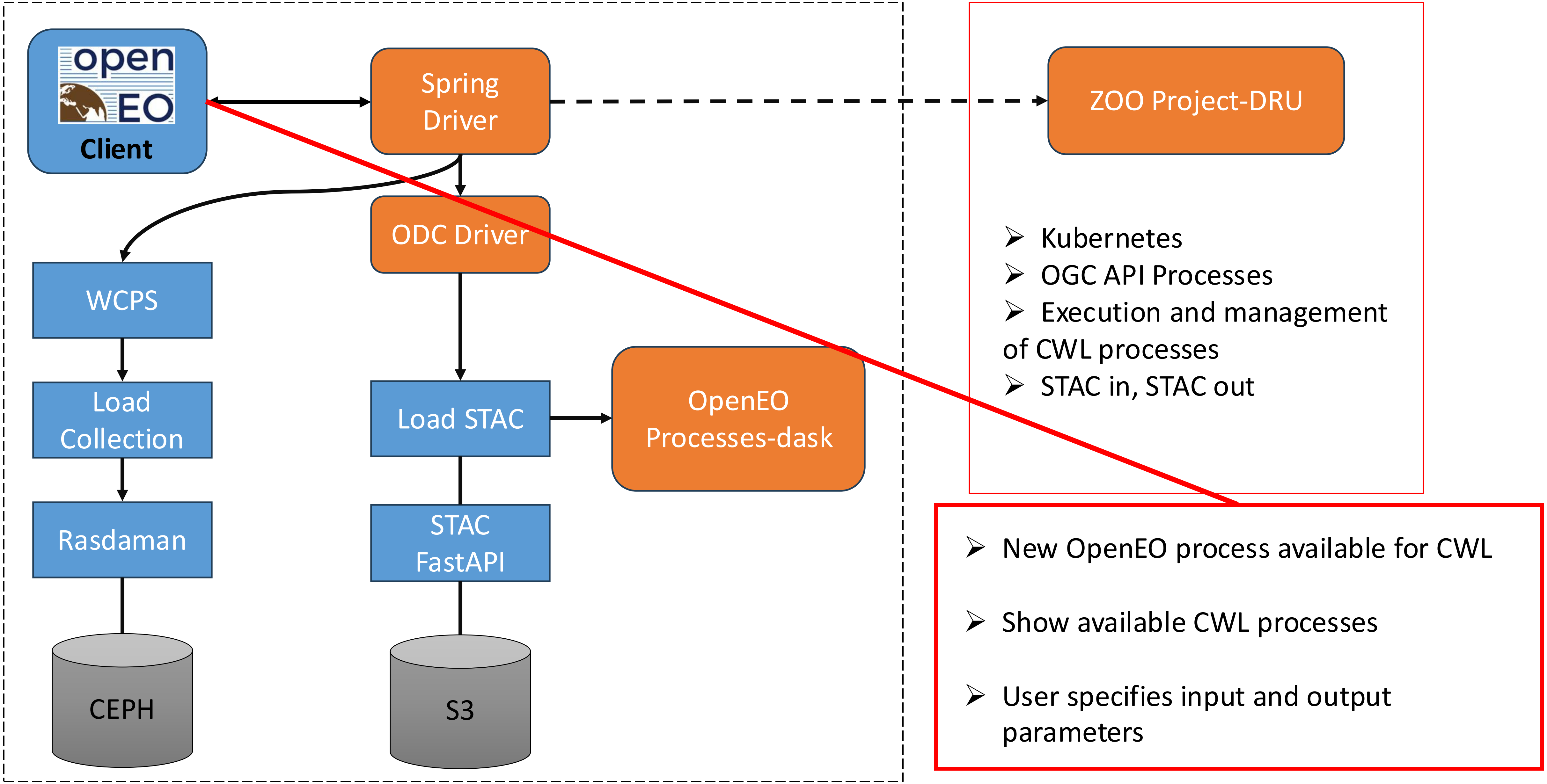
ZOO-Project-DRU





EGI 2024

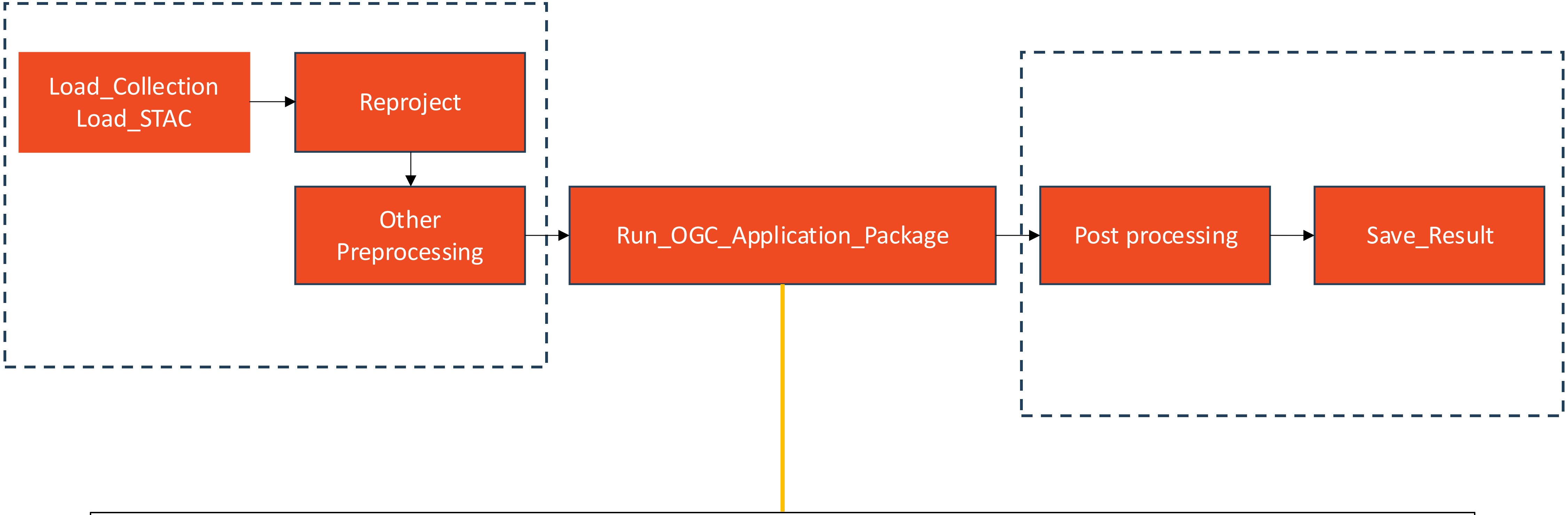
Towards Integration



- Kubernetes
- OGC API Processes
- Execution and management of CWL processes
- STAC in, STAC out

- New OpenEO process available for CWL
- Show available CWL processes
- User specifies input and output parameters

Sample Process Graph



The JSON Process graph is **split**, and the processing is redirected to the CWL executor to run the Application Package, returns a result back to OpenEO for postprocessing

Example use case: HyDroForM

eurac research

HyDroForM

- ➢Hydrological Drought Forecasting Model with HydroMT and Wflow

- ➢Preprocess data with OpenEO, and run CWL to build the model

- ➢GitHub [repo]() (WIP)

- InterTwin GitHub: interTwin Community (github.com)
- HyDroForM: interTwin-eu/HyDroForM: Hydrological Drought Forecasting Model with HydroMT and Wflow (github.com)
- Deltares HydroMT: HydroMT: Automated and reproducible model building and analysis — HydroMT documentation (deltares.github.io)
- OGC Application Package Best Practices: OGC Best Practice for Earth Observation Application Package
- Application Package Hands on tutorial: Terradue/ogc-eo-application-package-hands-on: OGC EO Application Package Hands-on (github.com)
- CWL: https://www.commonwl.org/
- EOEPCA: Earth Observation Exploitation Platform Common Architecture - EOEPCA Portal
- ZOO-Project-DRU: https://github.com/ZOO-Project/ZOO-Project

Thank you!