ENES Climate Analytics Service (ECAS)
Thematic Service, EOSC-HUB

Sofiane Bendoukha\textsuperscript{1}, Tobias Weigel\textsuperscript{1}, Sandro Fiore\textsuperscript{2}, Alessandro D’Anca\textsuperscript{2}

\textsuperscript{1}Deutsches Klimarechenzentrum (DKRZ), Hamburg, Germany
\textsuperscript{2}Centro Euro-Mediterraneo sui Cambiamenti Climatici (CMCC), Lecce, Italy

11. April 2018

EOSC-hub is EU’s Horizon 2020 project under grant agreement 777536
The ENES Climate Analytics Service (ECAS) will enable scientific end-users from climate data community and other disciplines to perform data analysis experiments on large volumes of climate data. It follows a PID-enabled, server-side and parallel approach.

**Benefits:**
- Reduce the need for local data downloads
- Reduce the effort of maintaining client-side tools
- Reduce the complexity of workflows for simple users
- Encourage flexible and open data sharing
- Enable PID-based provenance support
- More efficient use of computing resources

**Key integration milestones:**
- Enhance ENES/ESGF PID management services to be fully integrated into the EUDAT B2HANDLE service concept
- Provide/enable connectors to allow data transfer between ESGF nodes and Ophidia instances
- Integration of Ophidia with EUDAT B2HANDLE
- Make Ophidia final output available through EUDAT B2DROP and B2SHARE
- Setup and configuration of workflow repository for publishing and sharing workflows
ENES Climate Analytics Service (ECAS)

- **ECAS** is a contribution to the EOSC-hub project.
- It enables scientific end-users from climate data community and other disciplines to perform data analysis experiments on large volumes of (climate) data.
  - **Server-side, PID\(^1\)-enabled and parallel** approach
    - uplift existing computing capabilities of Ophidia framework to be available as an EOSC service
    - integrate with EOSChub services for ease of use
    - provide hands-on training to overcome adoption barriers

\(^1\)Persistent Identifier
ENES Climate Analytics Service (ECAS)

Ophidia / ENES Climate Analytics Service

source: Tobias Weigel
**Ophidia: Big Data Analytics Framework**

**Ophidia** is a research effort carried out at the Euro Mediterranean Centre on Climate Change (CMCC) to address big data challenges, issues and requirements for climate change data analytics.

- Server-side, parallel and distributed
- Extensible
- Client application available as user interface
- Workflow support
Ophidia: Server Architecture

Figure: Sandro Fiore
Ophidia: Array based Primitives

- **Ophidia** provides a wide set of array-based primitives (about 100) to perform summarization, sub-setting, predicates evaluation, statistical analysis, etc.
- Primitives comes as plugins and are applied on a single datacube chunk (fragment).
- **Primitives can be nested** to get more complex functionalities.
- New primitives can be easily integrated as additional plugins.
**Ophidia: Data Cube Operators**

<table>
<thead>
<tr>
<th>OPERATOR NAME</th>
<th>OPERATOR DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPH_APPLY(datacube_in, datacube_out, array_basedPrimitive)</td>
<td>Creates the datacube_out by applying the array-based primitive to the datacube_in</td>
</tr>
<tr>
<td>OPH_DUPLICATE(datacube_in, datacube_out)</td>
<td>Creates a copy of the datacube_in in the datacube_out</td>
</tr>
<tr>
<td>OPH_SUBSET(datacube_in, subset_string, datacube_out)</td>
<td>Creates the datacube_out by doing a sub-setting of the datacube_in by applying the subset string</td>
</tr>
<tr>
<td>OPH_MERGE(datacube_in, merge_param, datacube_out)</td>
<td>Creates the datacube_out by merging groups of merge_param fragments from datacube_in</td>
</tr>
<tr>
<td>OPH_SPLIT(datacube_in, split_param, datacube_out)</td>
<td>Creates the datacube_out by splitting into groups of split_param fragments each fragment of the datacube_in</td>
</tr>
<tr>
<td>OPH_INTERCOMPARISON(datacube_in1, datacube_in2, datacube_out)</td>
<td>Creates the datacube_out which is the element-wise difference between datacube_in1 and datacube_in2</td>
</tr>
<tr>
<td>OPH_DELETE(datacube_in)</td>
<td>Removes the datacube_in</td>
</tr>
</tbody>
</table>

**Data processing** (parallel operators, MPI & OpenMP based)

**Import/Export** (parallel operators)

**Data Access** (sequential and parallel operators)

**Metadata management** (sequential and parallel operators)

source: Sandro Fiore
Ophidia client: The Ophidia Terminal

oph_term is a terminal-like commands interpreter serving as a client for the Ophidia framework. Through the oph_term the user runs (send) commands (operators) to the Ophidia framework to manipulate datasets (datacubes).
# Ophidia client: The Ophidia Terminal


<table>
<thead>
<tr>
<th>Response:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Workflow Status</td>
</tr>
<tr>
<td>OPH_STATUS_COMPLETED</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Workflow Progress</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image-url" alt="Image" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Task List</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="table-url" alt="Table" /></td>
</tr>
</tbody>
</table>

---

**Legend:**
- **http://127.0.0.1/ophidia/sessions/**<br>
- **25251403081304837291501061213487214/**<br>
- **experiment/43081767**

---

**Notes:**
- This table shows the tasks executed in the Ophidia workflow.
- Task names and descriptions are [ophidia](ophidia-url) specific.
- Exit statuses indicate the success or failure of each task.
Ophidia client: PyOphidia

Python package to interact with the Ophidia framework

- Alternative to Oph Term
- Runs on Python 2.7, 3.3, 3.4 and 3.5
- Requires a running Ophidia instance
- It provides two main modules:
  - **client.py**: generic low level class to submit any type of requests (simple tasks and workflows), using SSL and SOAP with the client ophsubmit.py
  - **cube.py**: high level cube-oriented class to interact directly with cubes, with several methods wrapping the operators.
Ophidia client: PyOphidia

- Instantiate a client and submit a request

```python
from PyOphidia import client

ophclient = client.Client("oph-user", "oph-password", "127.0.0.1", "11732")

ophclient.submit("oph_list level=2", display=True)
```

- Set a Client for the Cube class

```python
from PyOphidia import cube

cube.Cube.setclient('oph-user','oph-password','127.0.0.1','11732')

#Export to NetCDF file
cube.Cube.createcontainer(container='test',dim='lat|lon|time',dim_type='double|double|double')

#Show a Cube structure and info
mycube1.info()

#Subset a Cube
mycube=mycube1.subset(subset_dims='lat|lon',subset_filter='1:10|20:30',subset_type='coord')
```
Workflow Support

```
{
    "name": "Example1",
    "author": "Foo",
    "abstract": "Simple workflow with three tasks",
    "exec_mode": "sync",
    "ncores": "1",
    "cube": "http://hostname/1/1",
    "tasks": [
        {
            "name": "Extract maximum value",
            "operator": "oph_reduce",
            "arguments": [ "operation=max" ]
        },
        {
            "name": "Extract minimum value",
            "operator": "oph_reduce",
            "arguments": [ "operation=min" ]
        },
        {
            "name": "Evaluate max-min range",
            "operator": "oph_intercube",
            "arguments": [ "operation=sub" ],
            "dependencies": [
                { "task": "Extract maximum value", "type": "single", "argument": "cube" },
                { "task": "Extract minimum value", "type": "single", "argument": "cube2" }
            ]
        }
    ]
}
```
**Precipitation Trend Analysis**

**Description:** Precipitation trend analysis, on different spatial and temporal scales, has received notable attention during the past century due to its relations with global climate change stated by the scientific community. For this reason, a number of models for this atmospheric variable have been defined.

**GitHub repository:** https://github.com/OphidiaBigData/ophidia-workflow-catalogue/blob/master/indigo/precip_trend_analysys/optimized_precip_trend_analysis.json

**OphidiaLab repository:** /workflows/precip_trend_analysys/optimized_precip_trend_analysis.json

**Arguments:** number of cores; model name; spatial subset; time subset in the past; time subset in the future; output grid; import type (optional); I/O server type (optional)

**Example:** ./optimized_precip_trend_analysis.json 2 CMCC-CM|CMCC-CMS rcp85 day 0.9 1976_2006 2071_2101 -90:90:0:360 r360x180

**Number of tasks:** 77
Ophidia Use Cases

Output: 5 png files: maximum temperature, minimum temperature, average, standard deviation, variance

Precipitation trend

Ophidia Use Cases
ECASLab

A user-friendly scientific data analysis environment.
ECASLab consists of:

- **Ophidia cluster**
- JupyterHub instance
  - Jupyter notebook
  - Ophidia client (PyOphidia, Ophidia terminal)
- Set of pre-installed Python libraries for running data manipulation, analysis and visualization
- Data publication service
How to use it?

1. Register
2. Login to JupyterHub
3. Perform data analysis

* At CMCC or DKRZ
Integration with EUDAT Services

source: https://eudat.eu/
Integration with EUDAT Services

- **B2DROP**: Through B2DROP, ECAS provides a secure and trusted data exchange service to store and exchange processed data outputs
- **B2SHARE**: ECAS offer users to receive, store and publish processed data outputs
- **B2HANDLE**: Through the B2HANDLE service, ECAS assigns Persistent Identifiers (PID) to Ophidia output and records kernel information according to a community profile
For more informations

- Thursday 16:45, oral presentation on ECASLab. Room M2.
- EOSC-HUB project http://eosc-hub.eu/
- EUDAT services www.eudat.eu/services
- Ophidia http://ophidia.cmcc.it/
- Ophidia use cases https://github.com/OphidiaBigData/ophidia-workflow-catalogue
- Get in touch ecas-support@dkrz.de