8th Annual Earth System Grid Federation

# High Performance Data Analytics and Machine Learning developments in Ophidia


**Sandro Fiore**
Head of the Data Science and Learning Research Team
CMCC Advanced Scientific Computing Division

**Donatello Elia**
CMCC Advanced Scientific Computing Division

**Giovanni Aloisio**
Director of the CMCC SuperComputing Centre
CMCC Strategic Council member


December 4-7, 2018

cmcc Centro Euro-Mediterraneo sui Cambiamenti Climatici

NextData

is-enes INFRASTRUCTURE FOR THE EUROPEAN NETWORK FOR EARTH SYSTEM MODELLING

# Introduction

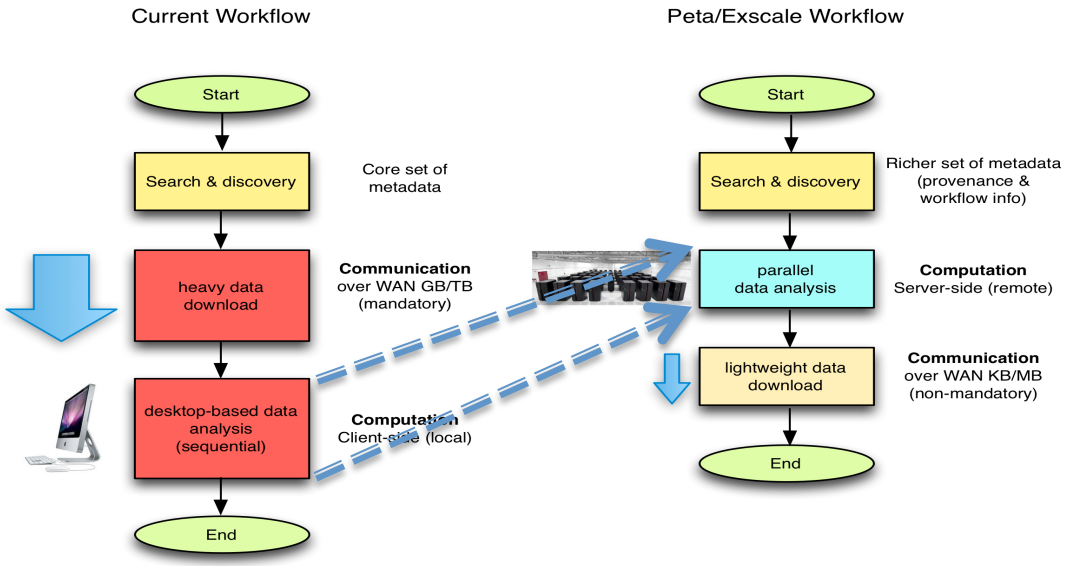
- Research challenge: HPC & big data convergence at extreme-scale for eScience
- Goal: Scalable HPDA&ML software stack for eScience
- International Context: IESP (2008-2011), BDEC (2012-2018) and now BDEC2 (2018-2020)
- European projects: EESI (2008-2011), EESI2, EXDCI and now EXDCI2 (2018-2020)
- Research project: Ophidia (design study 2011, implementation 2012 - present)

- This talk focus: Ophidia2.0

# Ophidia

- Ophidia is a CMCC research project addressing fast and big data challenges for eScience
- It provides a paradigm shift, from sequential & client- to parallel & server-side data analysis

## Paradigm shift at scale




*S. Fiore, A. D'Anca, C. Palazzo, I. Foster, D. N. Williams, G. Aloisio, "**Ophidia: toward bigdata analytics for eScience**", ICCS2013 Conference, Procedia Elsevier, Barcelona, June 5-7, **2013***

# Key Features in Ophidia

- eScience framework
- Server-side
- Parallel
- In-memory
- Declarative
- Datacube oriented (multi-dimensional OLAP support)

- (Shared) Sessions
- Workflows and applications
- Interactive and batch support
- HPC and HTC tasks
- Both domain-oriented (e.g. nc) and domain-agnostic support (e.g. OLAP)

# Multi-dimensional data and storage model

- OLAP-based (ndim-datacube)
- Dimension-independent
- Implicit and explicit dimensions
- Partition & distribution
- Flexible and scalable

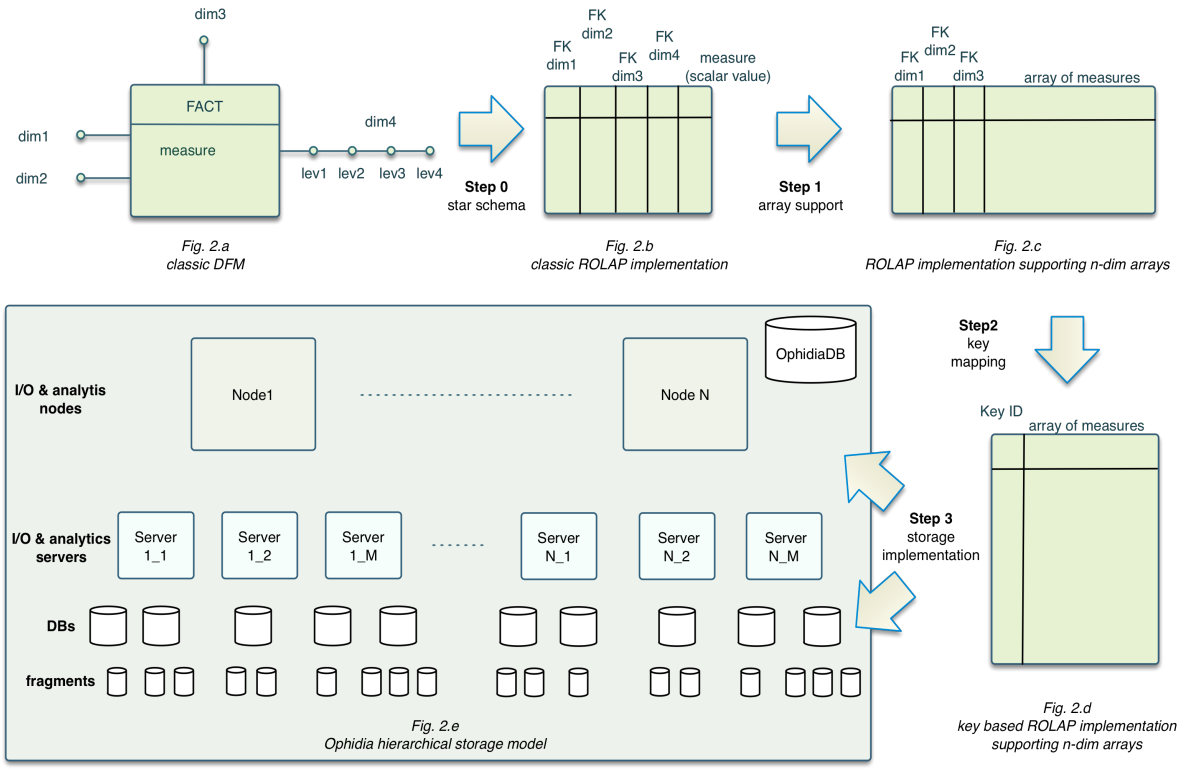



*Fig. 2.a*
*classic DFM*

*Fig. 2.b*
*classic ROLAP implementation*

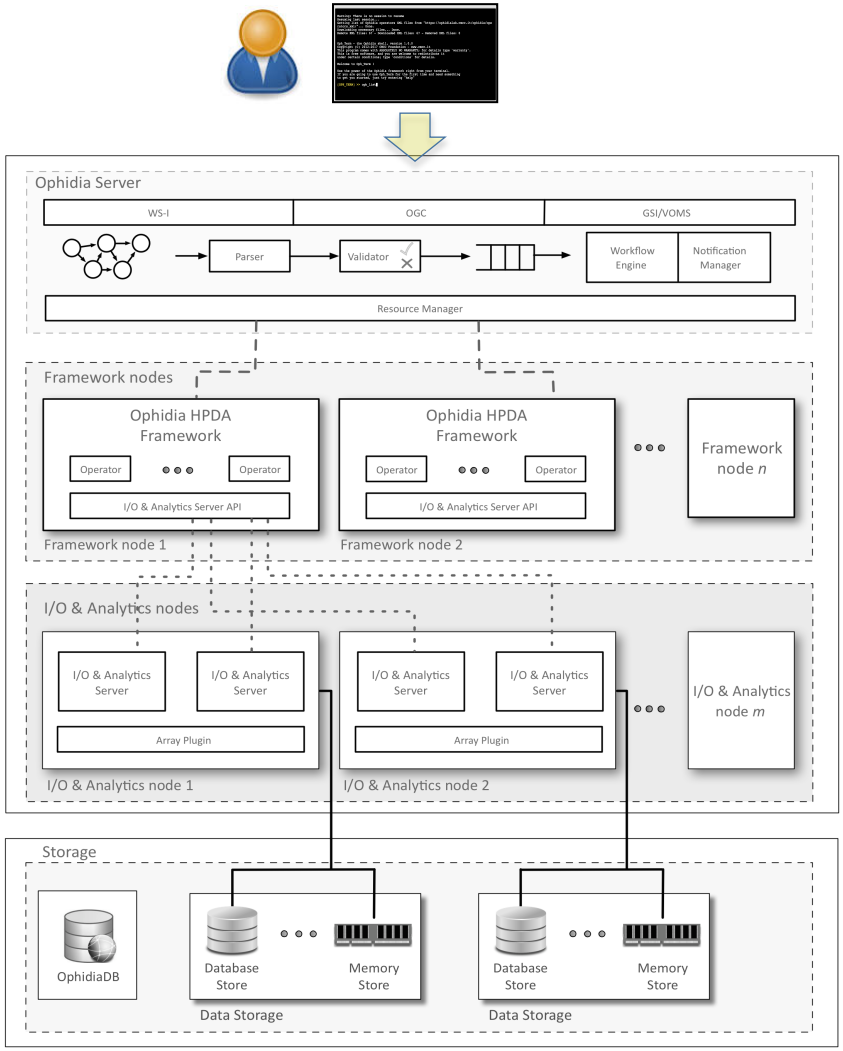
*Fig. 2.c*
*ROLAP implementation supporting n-dim arrays*

*Fig. 2.e*
*Ophidia hierarchical storage model*

*Fig. 2.d*
*key based ROLAP implementation supporting n-dim arrays*

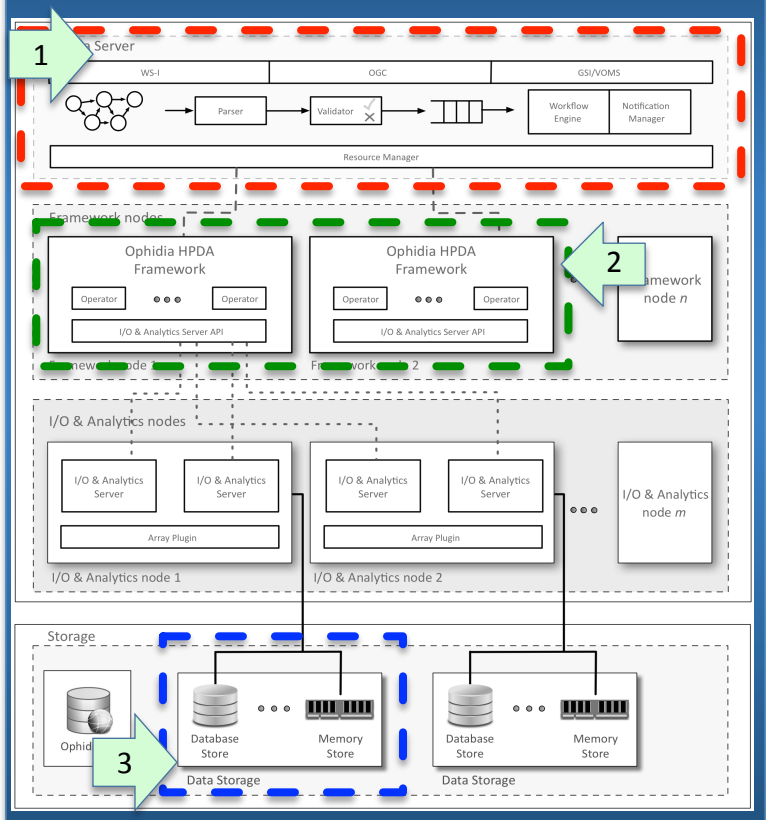
# Ophidia2.0 Architecture

- Interoperable interface (e.g. OGC WPS)
- Modular and extensible
- Parallel framework runtime
- I/O & analytics runtime
- Multiple storage back-ends
- Supports ranging from single operators to complex analyses

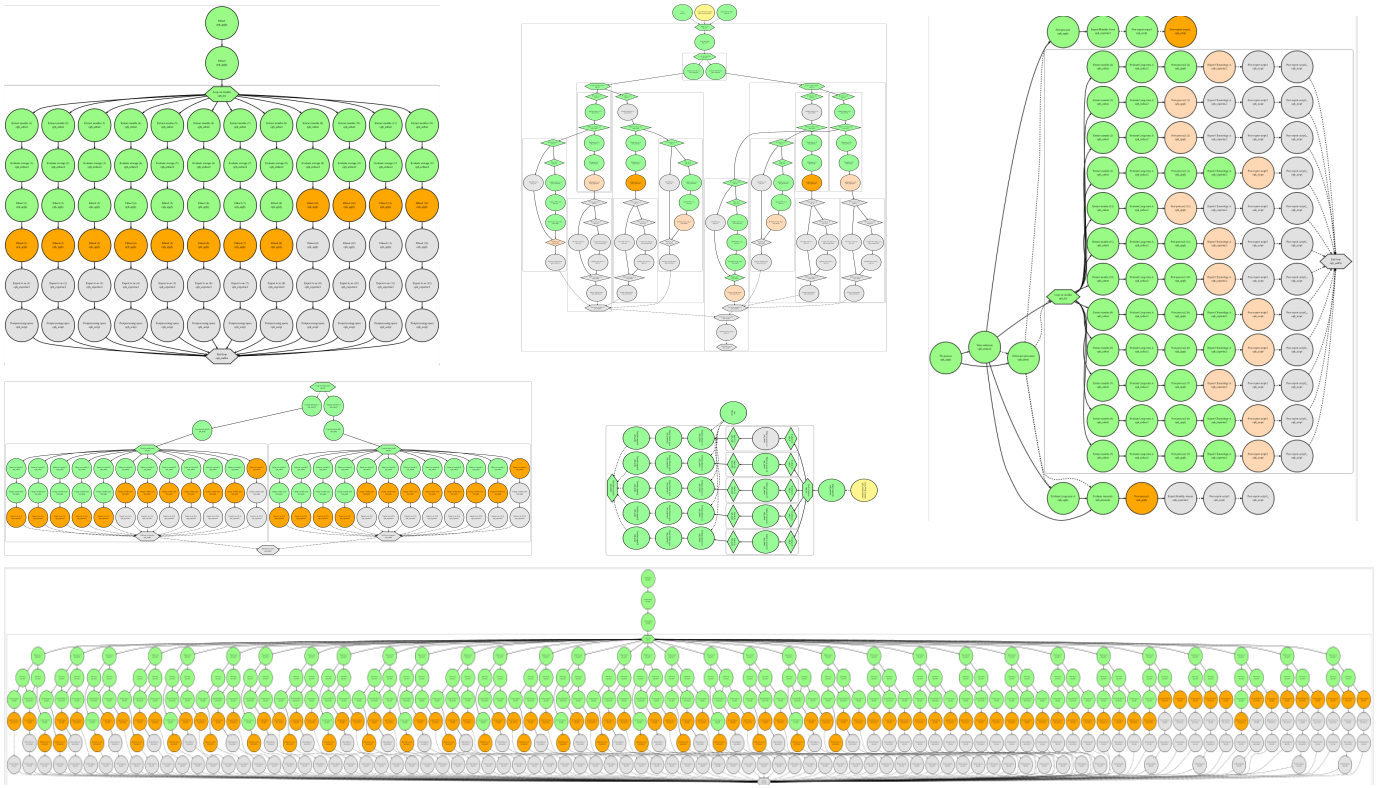


# Three levels of parallelism

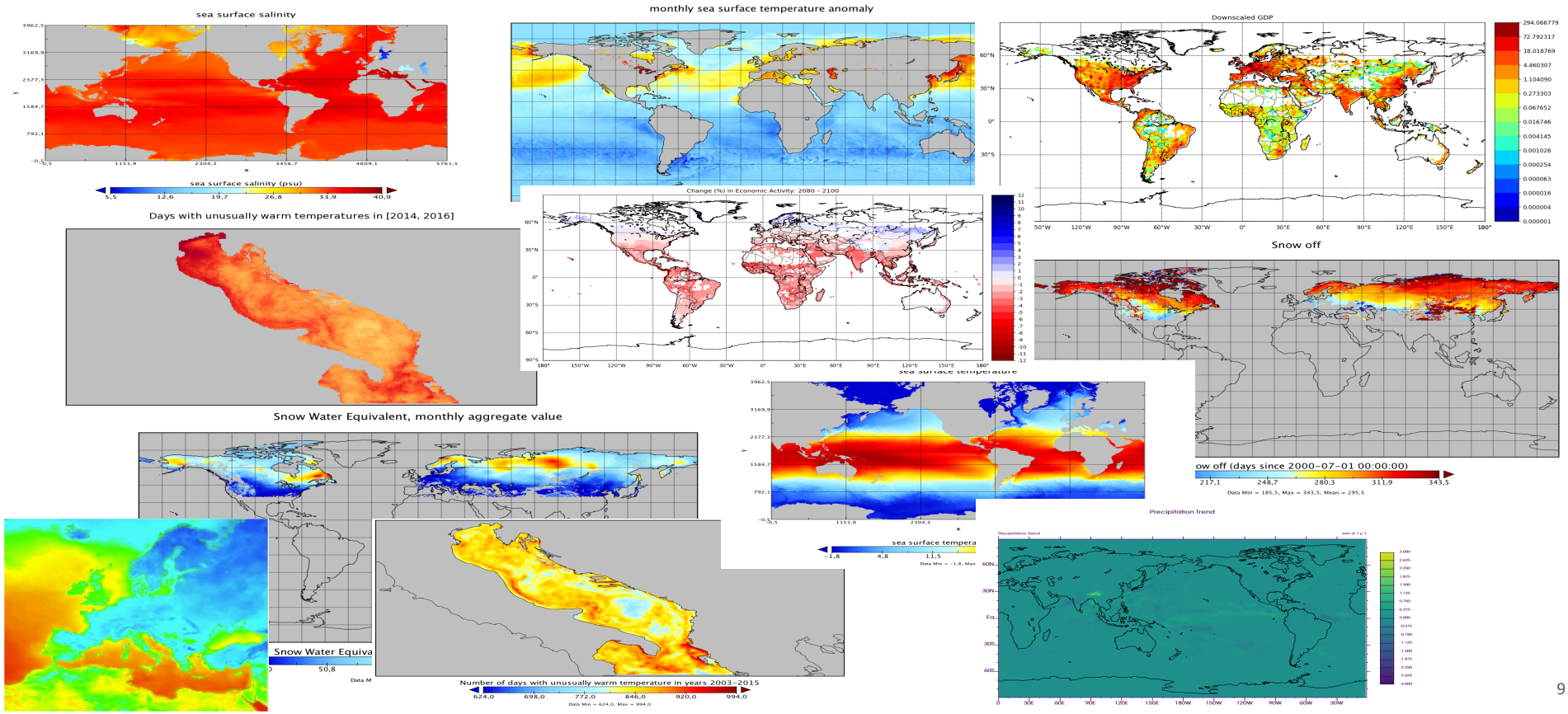
- Datacube-level parallelism
  - HTC paradigm
  - At the front-end level
  - Based on the “massive” operator concept
- Fragment-level parallelism
  - HPC paradigm
  - MPI/Pthread
  - At the HPDA framework level
- Array-level parallelism
  - HPC paradigm
  - OpeMP based
  - At the I/O & analytics server level



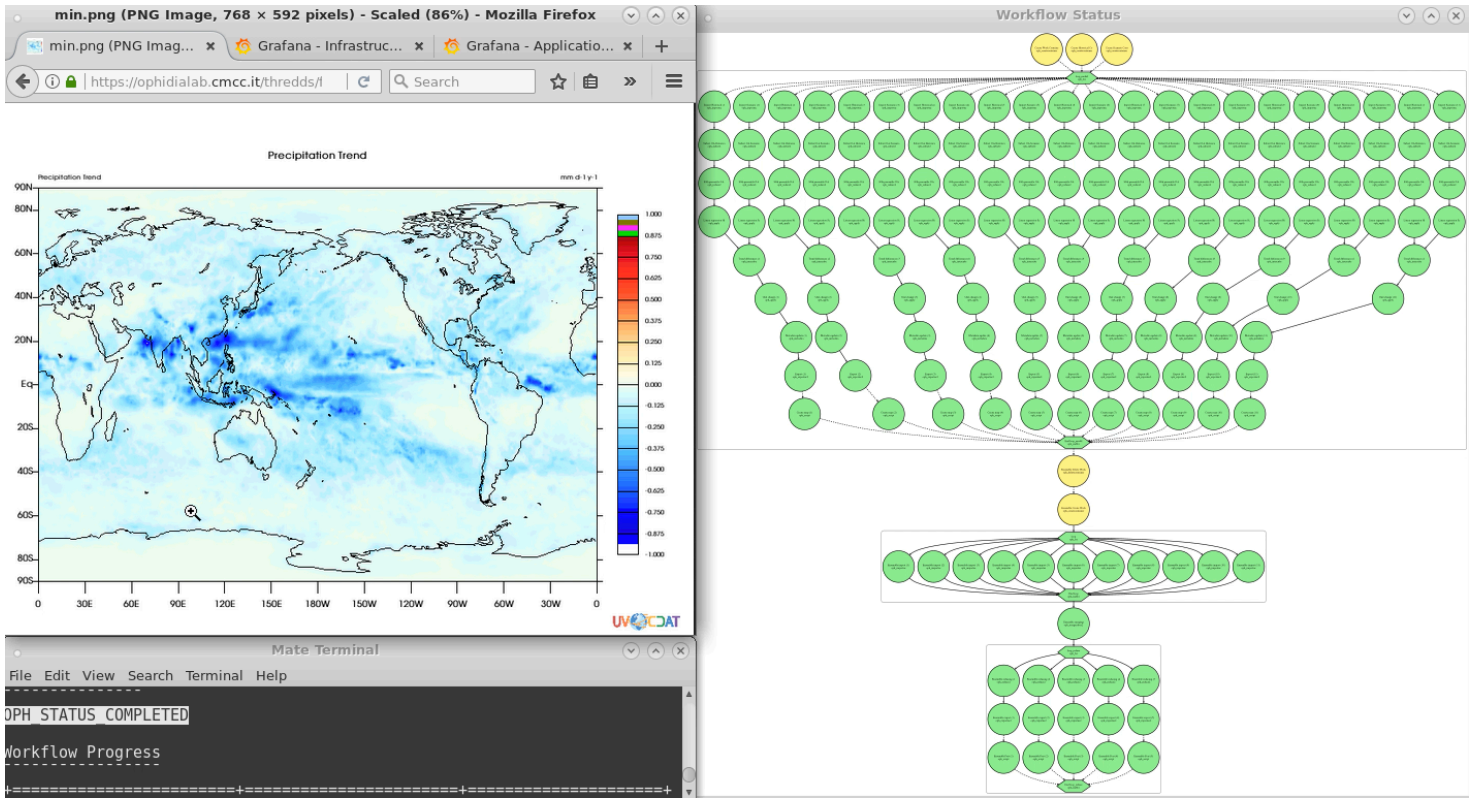
# Analytics Workflow support (runtime perspective)



# Analytics Workflow support (end-user perspective)



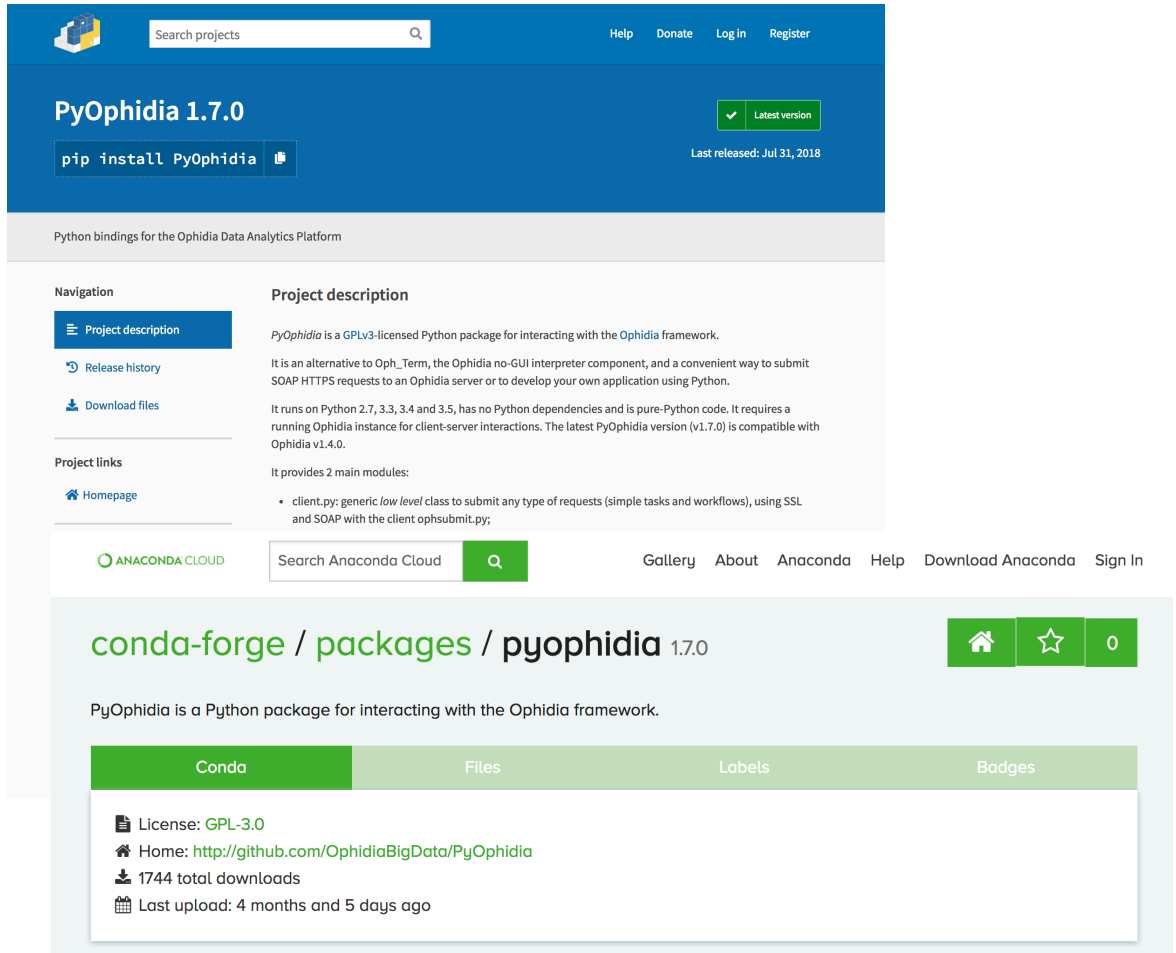
# Multi-model analytics workflow case study (CMIP5-based)




*The ESiWACE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 675191 http://www.esiwace.eu*

# Programmable framework

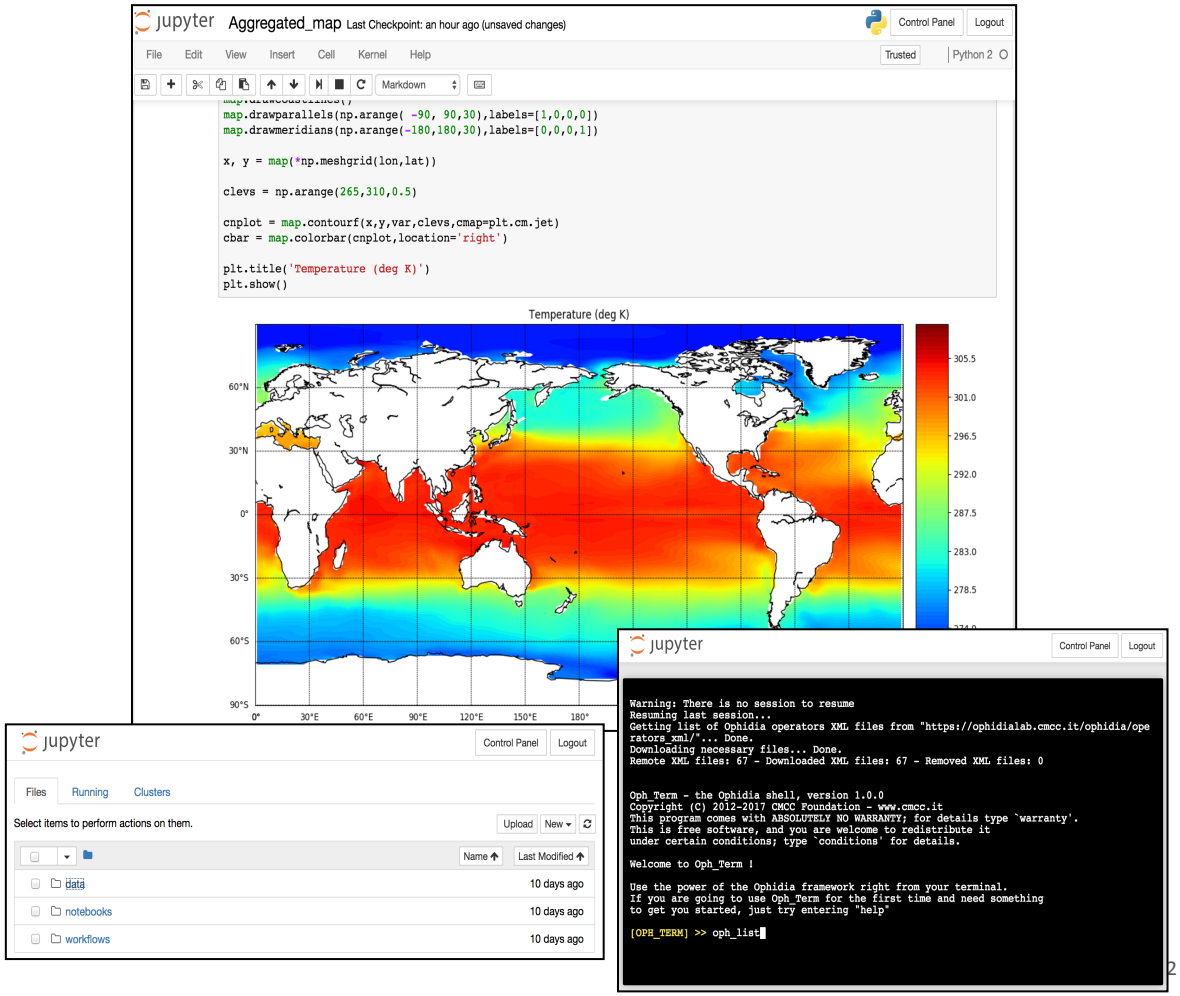
- PyOphidia provides a Python binding to Ophidia
- Consists of two classes:
  - Client class: connections, submissions, workflows, sessions
  - Cube class: datacube operators
- Available in conda-forge




https://pypi.org/project/PyOphidia/

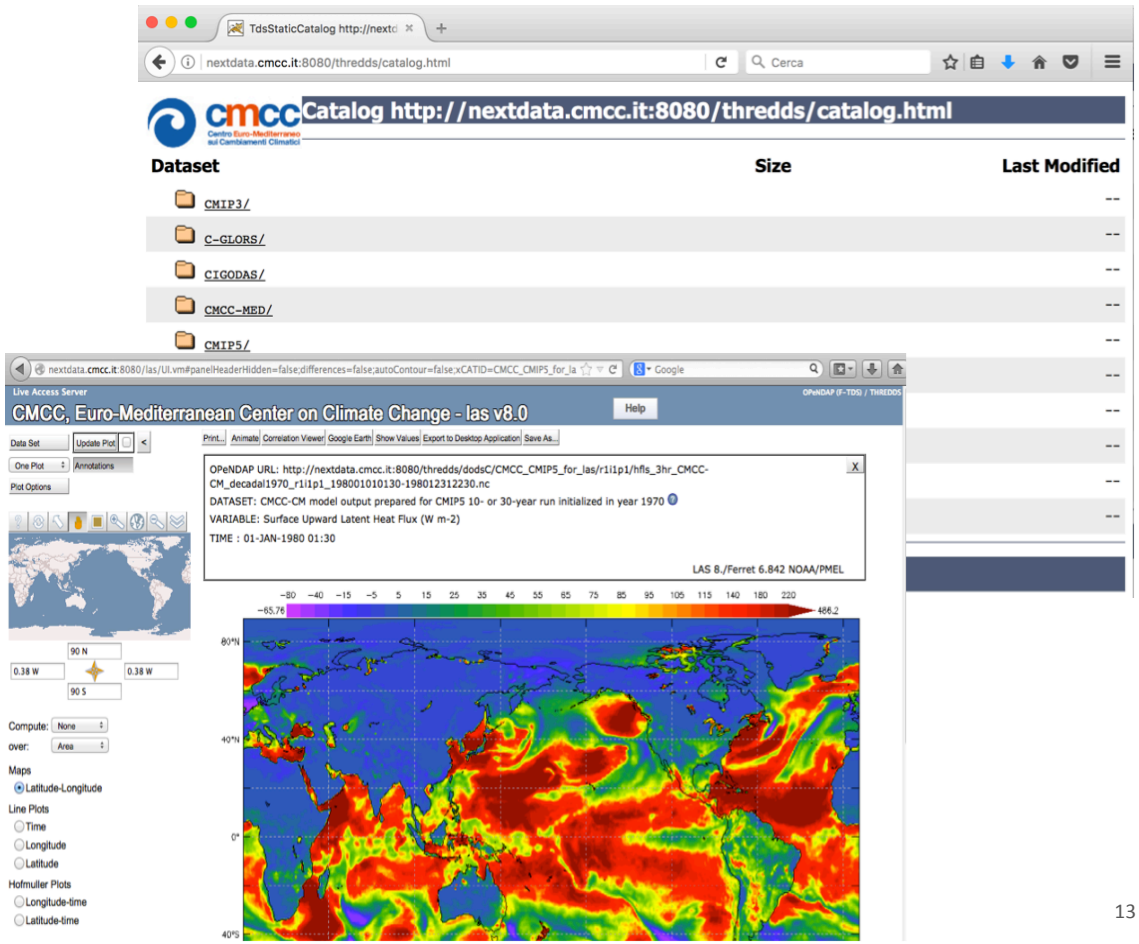
# JupyterHub front-end

- Web access/env
- Integrated into the HPC environment (e.g. GPFS)
- Access via VPN
- Multiplexed front-end
- Fat-node hosting
- Terminal features available too (embedded)
- Integrated into the EOSC-hub project (ECAS thematic service)



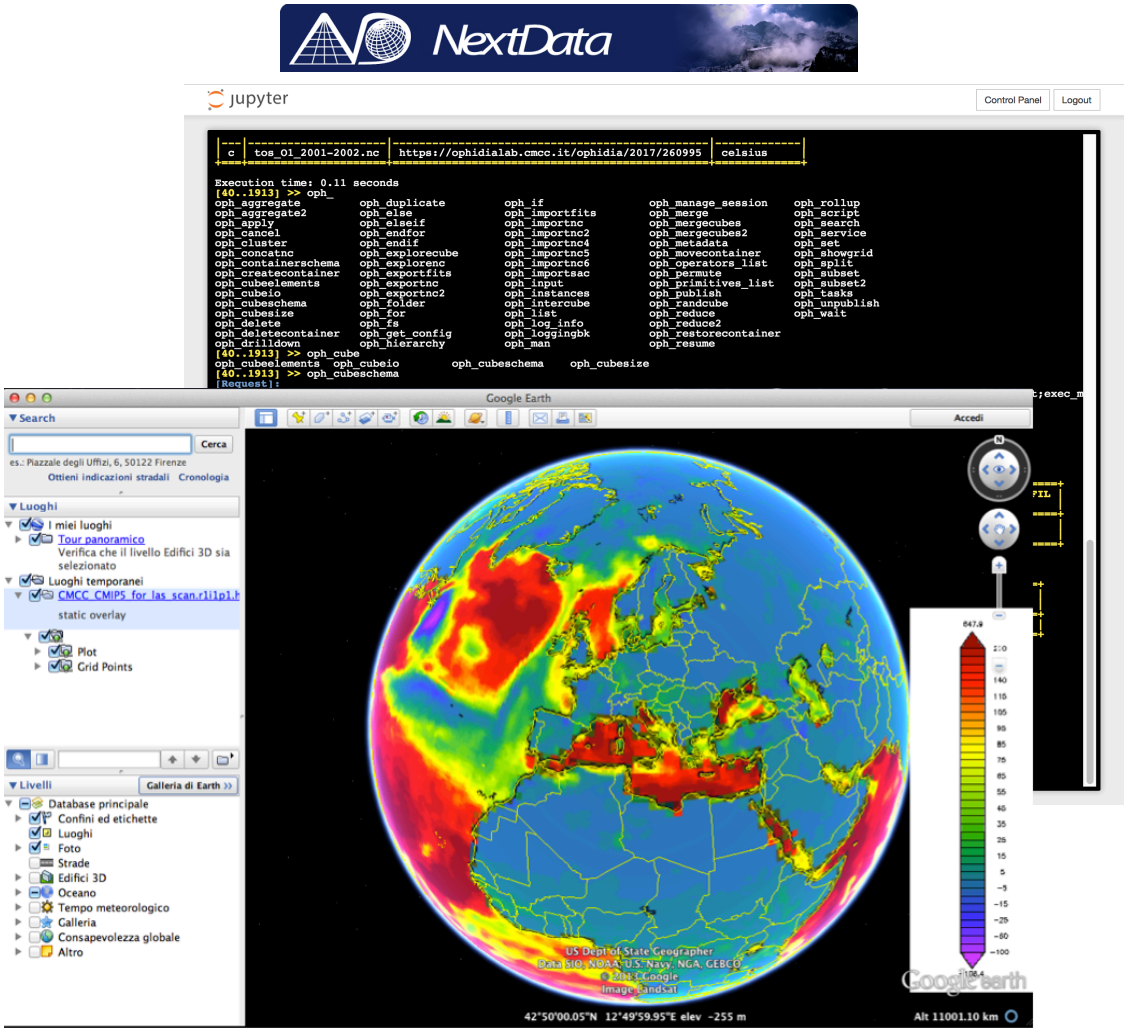
# Analytics back-end for the NextData project (I)

- Ophidia is also exploited in the NextData project
- Computing & data capabilities integrated into the same environment
  - Several datasets published by CMCC modelling divisions
  - Catalogue browsing
  - Data download
  - Visualization capabilities
  - Computing



# Analytics back-end for the NextData project (II)

- Computing & data capabilities integrated into the same environment
- Terminal-based capabilities
- Google Earth (viz)



# Analytics back-end for the NextData project (III)

- Programmatic access to the back-end datasets
  - Based on Python language
  - Jupyter Notebooks
  - Through PyOphidia

NextData

File Edit View Insert Cell Kernel Help Trusted Python 3

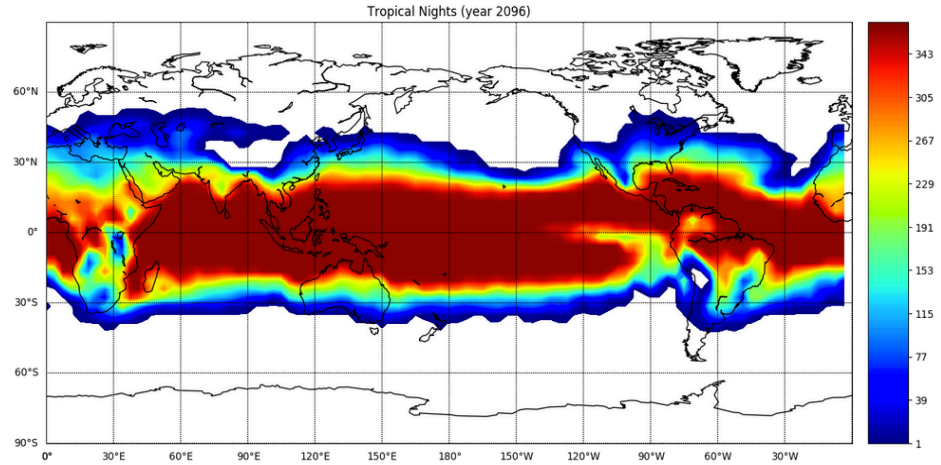
```
map.drawcoastlines()
map.drawparallels(np.arange( -90, 90,30),labels=[1,0,0,0])
map.drawmeridians(np.arange(-180,180,30),labels=[0,0,0,1])

x, y = map(*np.meshgrid(lon,lat))

clevs = np.arange(1,372,2)

cnplot = map.contourf(x,y,var,clevs,cmap=plt.cm.jet)
cbar = map.colorbar(cnplot,location='right')

plt.title('Tropical Nights (year 2096)')
plt.show()
```



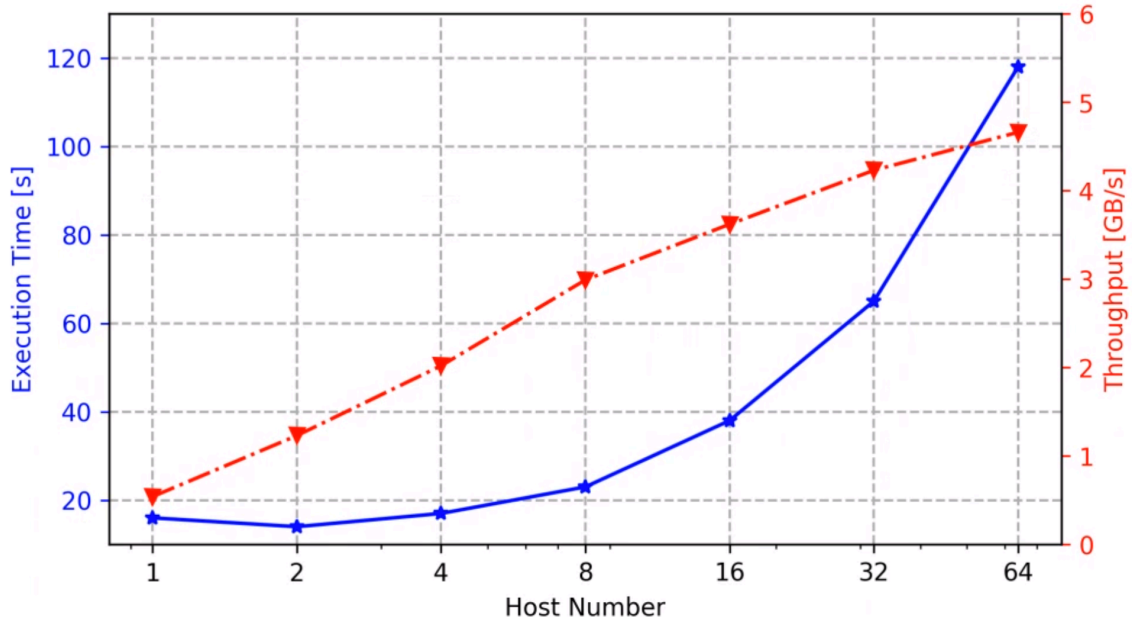

In [ ]:

# Parallel Import Monitoring

- ophclient.submit(”oph_importnc2 src_path=[path=*.nc;]”)
- *Three scenarios on a I/O and analytics node*
  - *Single import*
  - *Multiple imports/delete*
  - *Multiple imports on the same node*
- *Network traffic*
  - *Inbound burst*
  - *No outbound data*
- *Memory increase*
- *Cache effects*

# Parallel Import Benchmark (preliminary insights)

- *Importnc2 operator*
- *HTC importnc2 statements*
- *Weak scalability benchmark*
- *Metrics*
  - *Execution time [s]*
  - *Throughput [GB/s]*
- *From 1 to 64 input files*
  - *1 HPDAnode/1 input file*
  - *16 fragments/node*
  - *1 file 8.5GB (0.53TB)*
- *Up to 64 nodes → 1024 cores*




| Cores | 16 | 32 | 64 | 128 | 256 | 512 | 1024 (12%) |
|---|---|---|---|---|---|---|---|
| Size | 8,5 | 17 | 34 | 68 | 136 | 272 | 544 |
| Time [s] | 16 | 14 | 17 | 23 | 38 | 65 | 118 |
| BW [GB/s] | 0,54 | 1,23 | 2,02 | 2,99 | 3,62 | 4,23 | 4,66 (76%) |

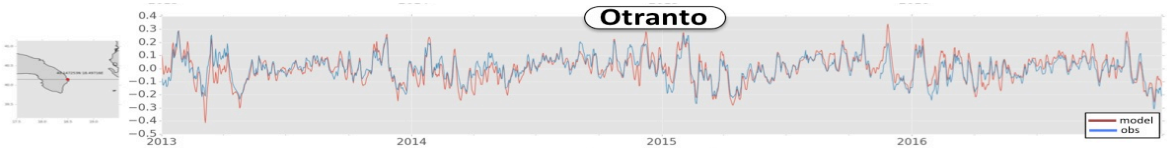
*Robustness test up to 4096 threads and 2.1TB on 1024 cores have been performed*

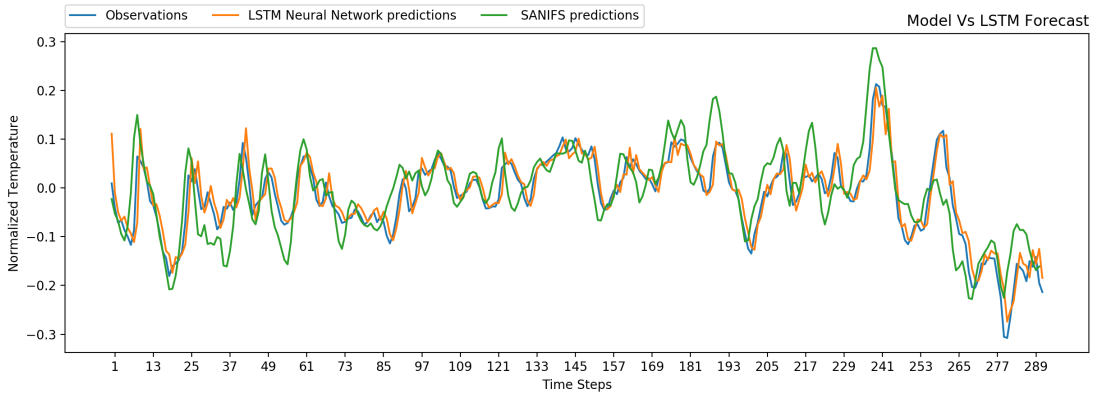
# LSTM Primitives and SANIFS use case

- Time-series predictive analytics
- The algorithm has been divided in two phases: training and test/prediction
- Implementation based on the KANN library
- After the training, the resulting neural network with updated parameters is saved as a binary array in a datacube. It can then be reused in the test phase
- For test and prediction we defined another primitive
- The primitives can run in the *oph_apply* operator

```
oph_lstm(input_OPH_TYPE, output_OPH_TYPE, measure,
dim_in, dim_out, n_h_layers, n_h_neurons, [dropout],
[learning_rate], [unrolled_len], [minibatch_size],
[max_epoch])
```

```
oph_lstm_predict(input_OPH_TYPE, output_OPH_TYPE,
measure_a, measure_b, test)
```



# Conclusions

- Ophidia2.0 provides a stronger integration of big data and HPC
  - Major step forward w.r.t. Ophidia1.4 and HPDA paradigm
  - Tight integration with HPC eco-system
  - New flexible and dynamic deployment approach
  - New release of the native I/O and analytics server
  - Improved management of multi-user scenarios
  - Three levels of parallelism to address higher scalability
  - Process-level (vs thread-level) connection to the OphidiaDB
- Planned to be released by this month (January 2019 at the latest). Stay tuned!

# Thanks!

## Do you want to join this effort?

Please get in touch with us soon

sandro.fiore@cmcc.it