IS-ENES3 Virtual Workshop
on Compute and Analytics

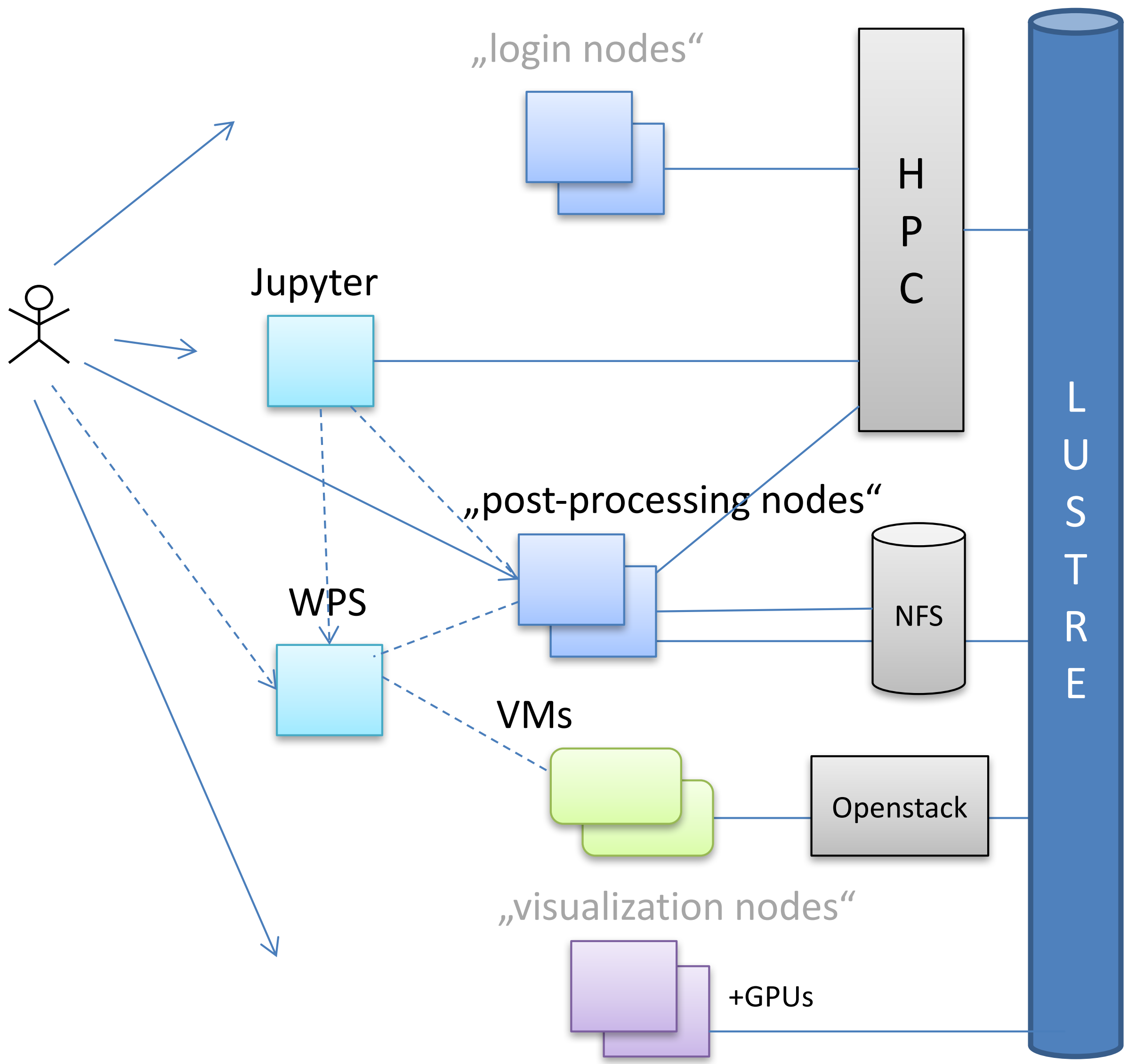
Compute and Analytics Services at DKRZ: status and plans

Stephan Kindermann, Carsten Ehbrecht

Overview

- High level overview of DKRZ compute services
- Interactive data analysis: Jupyter, etc.
- VM based compute services
- WPS service enabling: Birdhouse
- Data infrastructure – processing integration: EOSC ECAS example

DKRZ compute services: Status



Major usage scenario

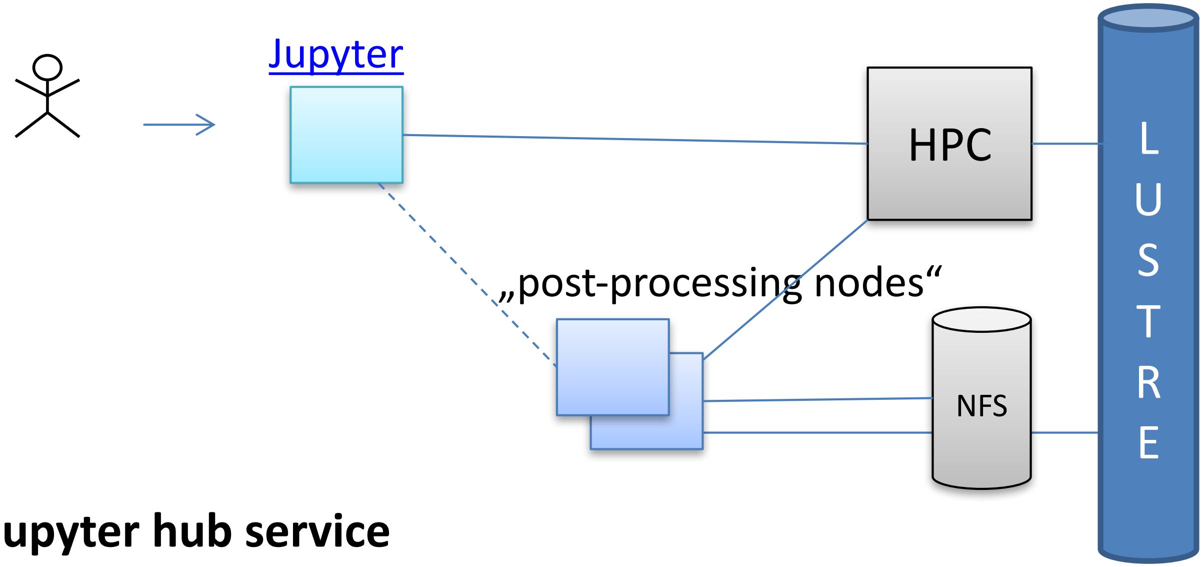
Batch processing

Interactive processing

service provisioning

visualization

DKRZ compute services: Jupyter



Jupyter hub service

- Exclusiv resource allocation (slurm adapter)
- Pre-configured environments (kernels, packages)

Direct jupyter notebook access:

- Shared resource usage (helper scripts for connecting)
- Usage of self configured environments

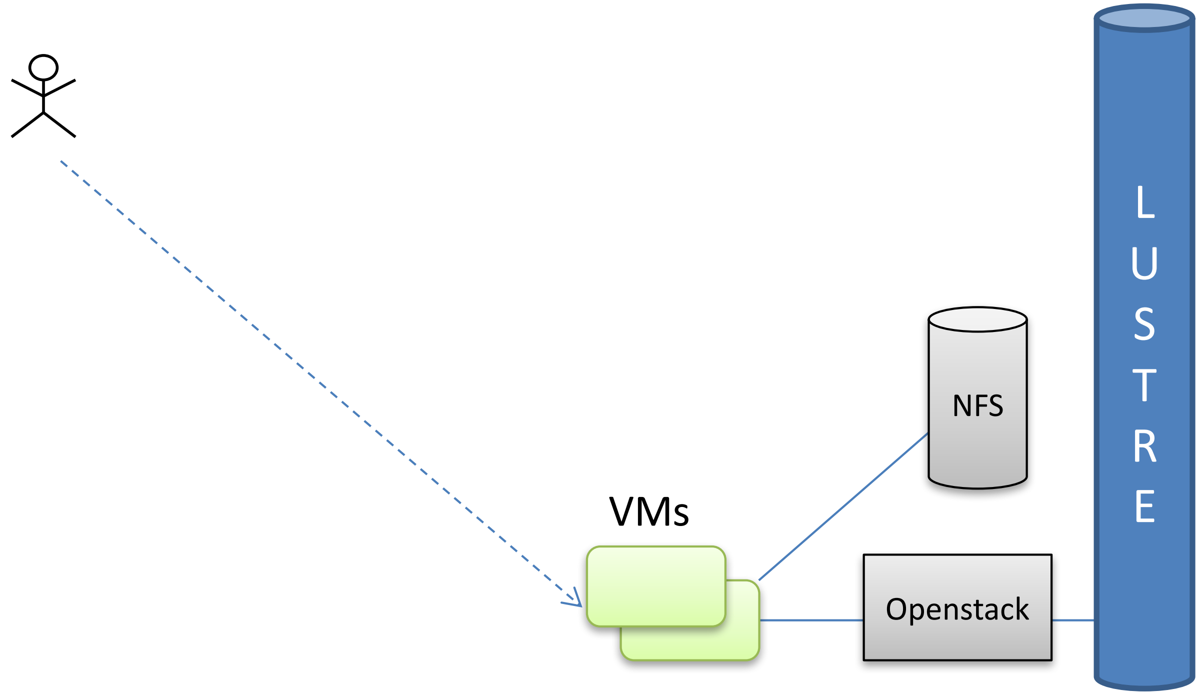
Jupyter hub EOSC service (ECAS)

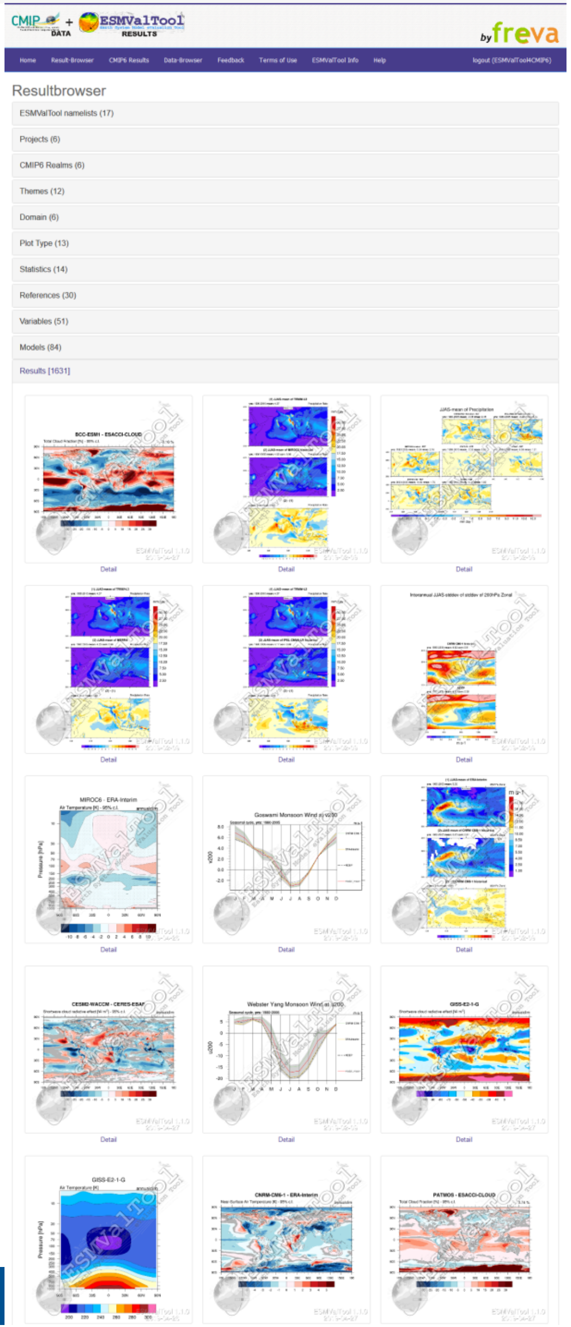
- Flexible env allocation (docker)
- Pre-configured environments (kernels, packages)
- Ophidia compute backend
- Integration with EOSC B2Share, B2Drop, B2Access services

Key challenges:

- Flexibility + exclusive resource allocation (→ docker, singularity ..)

DKRZ processing service example: EsmValTool





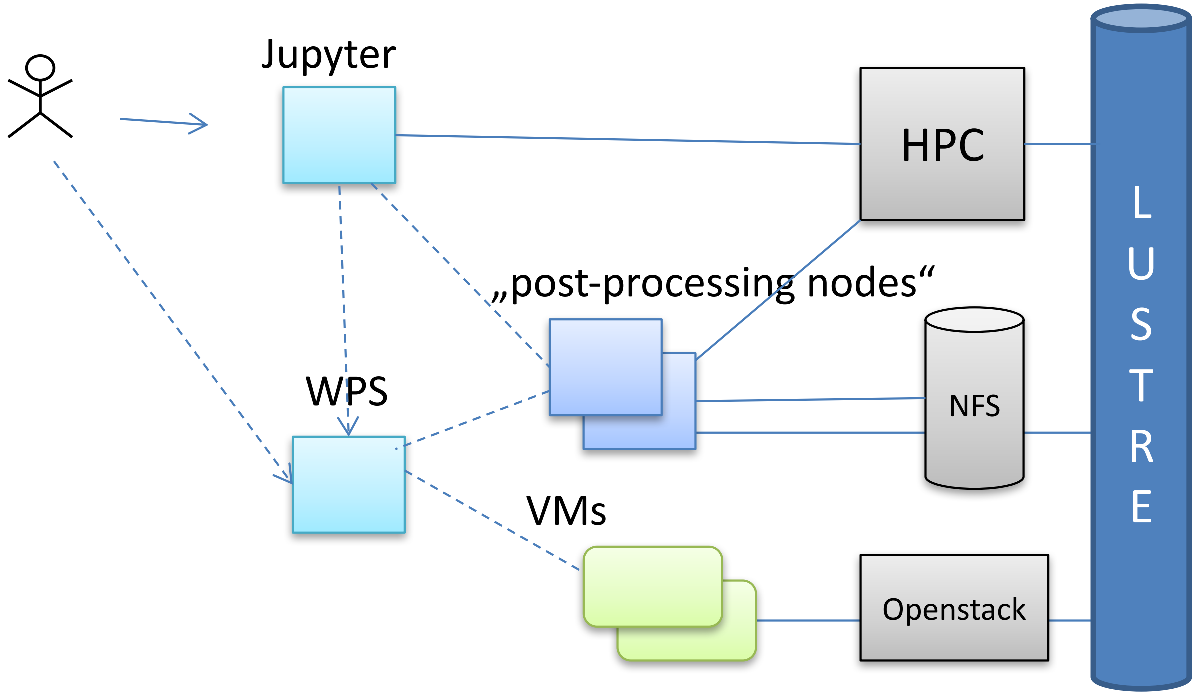
Resultbrowser

- ESMValTool namelists (17)
- Projects (6)
- CMIP6 Realms (6)
- Themes (12)
- Domain (6)
- Plot Type (13)
- Statistics (14)
- References (30)
- Variables (51)
- Models (84)
- Results [1631]

Service example: CMIP6 model evaluation result generation

- Deployed at https://cmip-esmvaltool.dkrz.de/
- Uses ESMValTool
- Continuous processing of CMIP6 evaluation results based on DKRZ CMIP data pool
- Integrated with CMIP6 replication

DKRZ processing service developments: Birdhouse



Birdhouse open source project:

- Modular components supporting OGC WPS service deployments
- Used for Copernicus WPS service deployments at CEDA, IPSL and DKRZ
- Planed usage for climate4impact backend services

Key challenges:

- Processing code base: maintenance, testing, sustainability
- „WPS service establishing workflow“: e.g. VM service → community → WPS service → example jupyter notebooks ..

Processing service developments: Birdhouse

IS-ENES3:

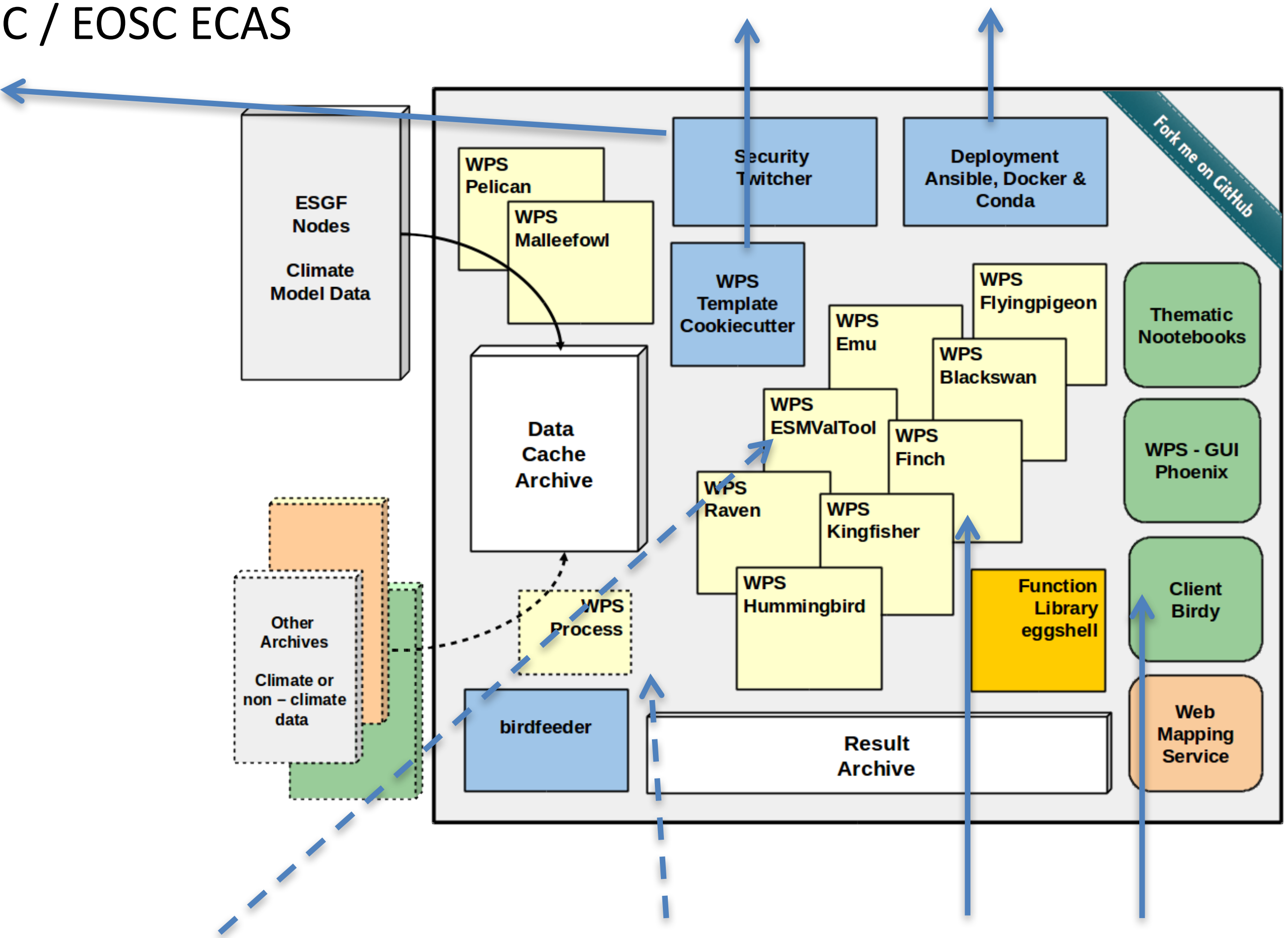
- Interop. with C4Impact
- Interop. with CMCC / EOSC ECAS

ESGF:

- CWT discussions

Copernicus:

- WPS service deployment



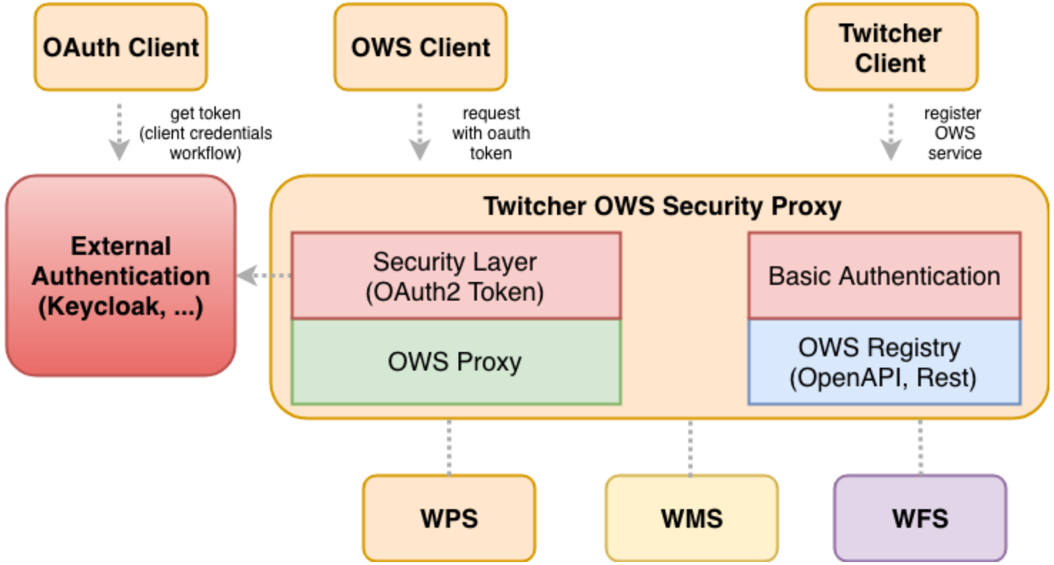
EsmValTool

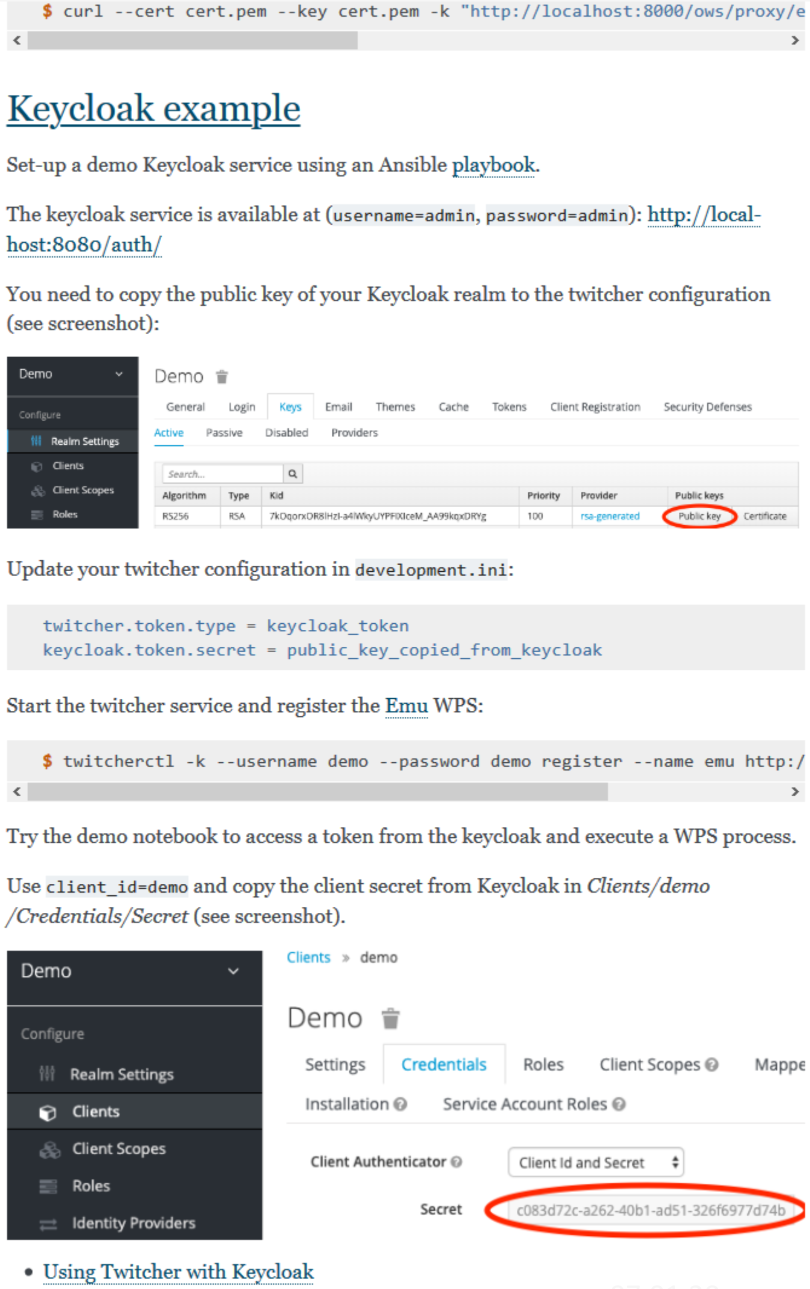
Copernicus
ESGF CWT

Ouranos/CRIM

Twitcher security proxy

- https://twitcher.readthedocs.io/en/latest/


OAuth Client
OWS Client
Twitcher Client
get token (client credentials workflow)
request with oauth token
register OWS service
External Authentication (Keycloak, ...)
Twitcher OWS Security Proxy
Security Layer (OAuth2 Token)
OWS Proxy
Basic Authentication
OWS Registry (OpenAPI, Rest)
WPS
WMS
WFS



```
$ curl --cert cert.pem --key cert.pem -k "http://localhost:8000/ows/proxy/e
```

Keycloak example

Set-up a demo Keycloak service using an Ansible playbook.

The keycloak service is available at (`username=admin`, `password=admin`): http://localhost:8080/auth/

You need to copy the public key of your Keycloak realm to the twitcher configuration (see screenshot):

Algorithm	Type	Kid	Priority	Provider	Public keys	
RS256	RSA	7kDqorxDR8lHzl-a4lWkyUYPFIXlceM_AA99kqxDRYg	100	rsa-generated	Public key	Certificate

Update your twitcher configuration in `development.ini`:

```
twitcher.token.type = keycloak_token
keycloak.token.secret = public_key_copied_from_keycloak
```

Start the twitcher service and register the Emu WPS:

```
$ twitcherctl -k --username demo --password demo register --name emu http:/
```

Try the demo notebook to access a token from the keycloak and execute a WPS process.

Use `client_id=demo` and copy the client secret from Keycloak in *Clients/demo/Credentials/Secret* (see screenshot).

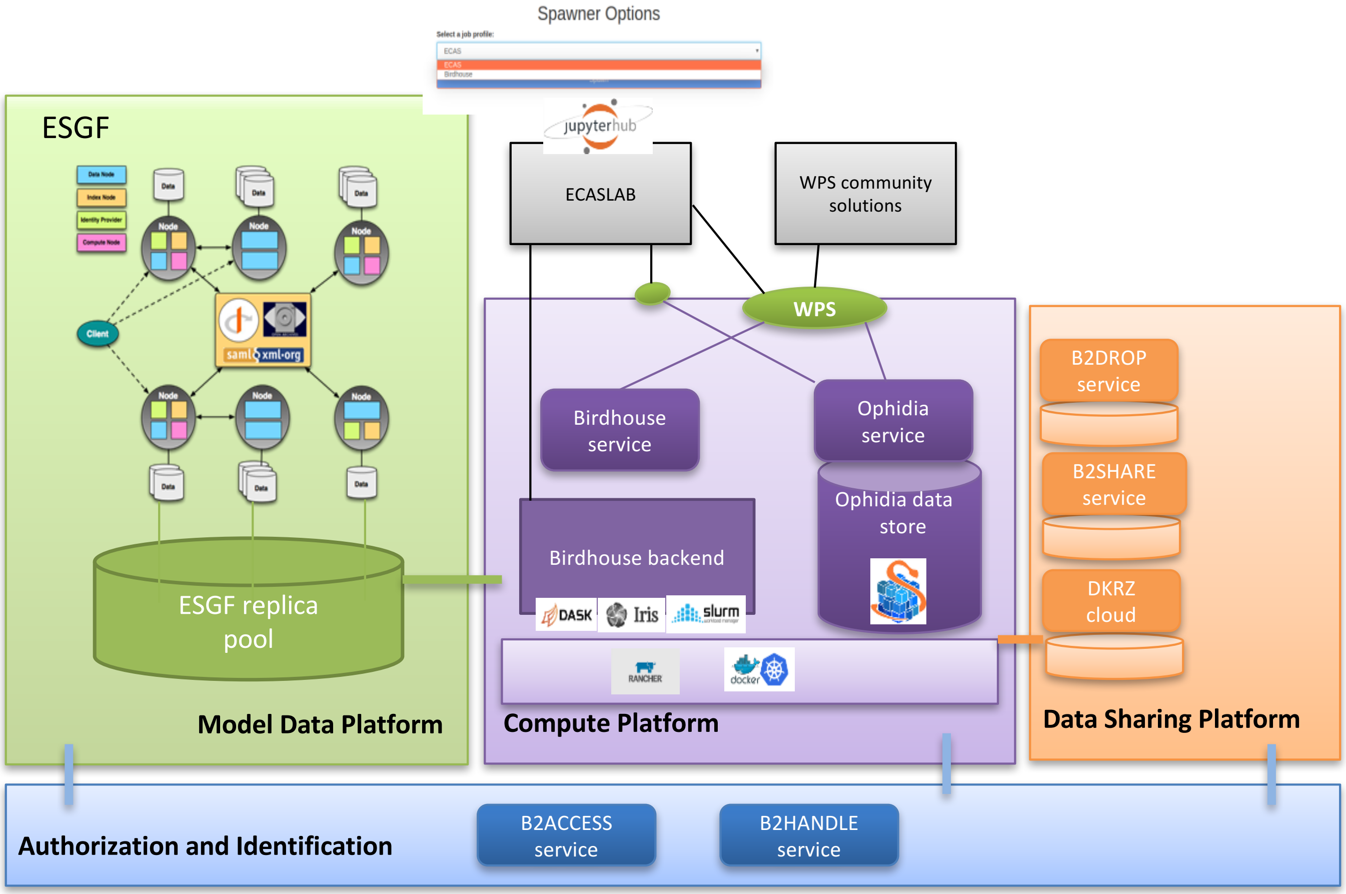
Client Authenticator: Client Id and Secret

Secret: c083d72c-a262-40b1-ad51-326f6977d74b

- Using Twitcher with Keycloak

ECAS: Data service – compute service integration

Example: EOSC data services ←→ ECAS compute service ←→ ESGF data



Challenges:

- Standardized interoperable interfaces to search, access and share data
  - ESGF bespoke search API, better something like opensearch/Intake catalogs
- Data provenance support

Summary: processing services at DKRZ

- Transition phase from „platform service“ to „tailored compute service“ offerings
  - New expertise needed:
    - Emerging compute layer (xarray, iris, dask, + ML packages)
    - Emerging infra layer (openstack, kubernetes, ..)
    - ML applications (new ML group established , > 7 people)
  - Integration of „semantic“ data access layer with compute
    - Model → data layer / data layer → data analysis / data layer → archival system (ECMWF field data base fdb5 analogies)
  - Requirement to support data analysis on much less standardized data collections (in comparison to CMIPs ..)

Some Links

- DKRZ jupyter notebook support: https://www.dkrz.de/up/systems/mistral/programming/jupyter-notebook
- CMIP6 Evaluation service: https://cmip-esmvaltool.dkrz.de/
- Bird-house – WPS related components: doc, github, publications
  - C. Ehbrecht, T. Landry, N. Hempelmann, D. Huard, and S. Kindermann. Projects based on the web processing service framework birdhouse. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-4/W8:43–47, 2018. URL: https://www.int-arch-photogramm-remote-sens-spatial-inf-sci.net/XLII-4-W8/43/2018/, doi:10.5194/isprs-archives-XLII-4-W8-43-2018.
  - Hempelmann, C. Ehbrecht, C. Alvarez-Castro, P. Brockmann, W. Falk, J. Hoffmann, S. Kindermann, B. Koziol, C. Nangini, S. Radanovics, R. Vautard, and P. Yiou. Web processing service for climate impact and extreme weather event analyses. flyingpigeon (version 1.0). *Computers & Geosciences*, 110(Supplement C):65 – 72, 2018.
    URL: http://www.sciencedirect.com/science/article/pii/S0098300416302801, doi:https://doi.org/10.1016/j.cageo.2017.10.004.
- EOSC ECAS service at DKRZ: https://www.dkrz.de/mms/pdf/talks/EGU2018-12549_ecas_Bendoukha.pdf (EGU 2019 presentation)