

# **birdhouse: supporting web processing services for climate data**

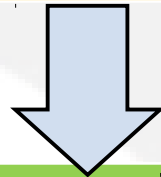
**Stephan Kindermann<sup>1</sup>, Carsten Ehbrecht<sup>1</sup>, Nils Hempelmann<sup>2</sup> et. al.**

**1. German Climate Computing Center, Germany**

**2. Le Laboratoire des Sciences du Climat et de l'Environnement, France**

# Climate Data volume grows quickly

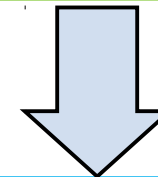
But on client side:  
Limited storage/compute capacities



**“download and  
process at home”**



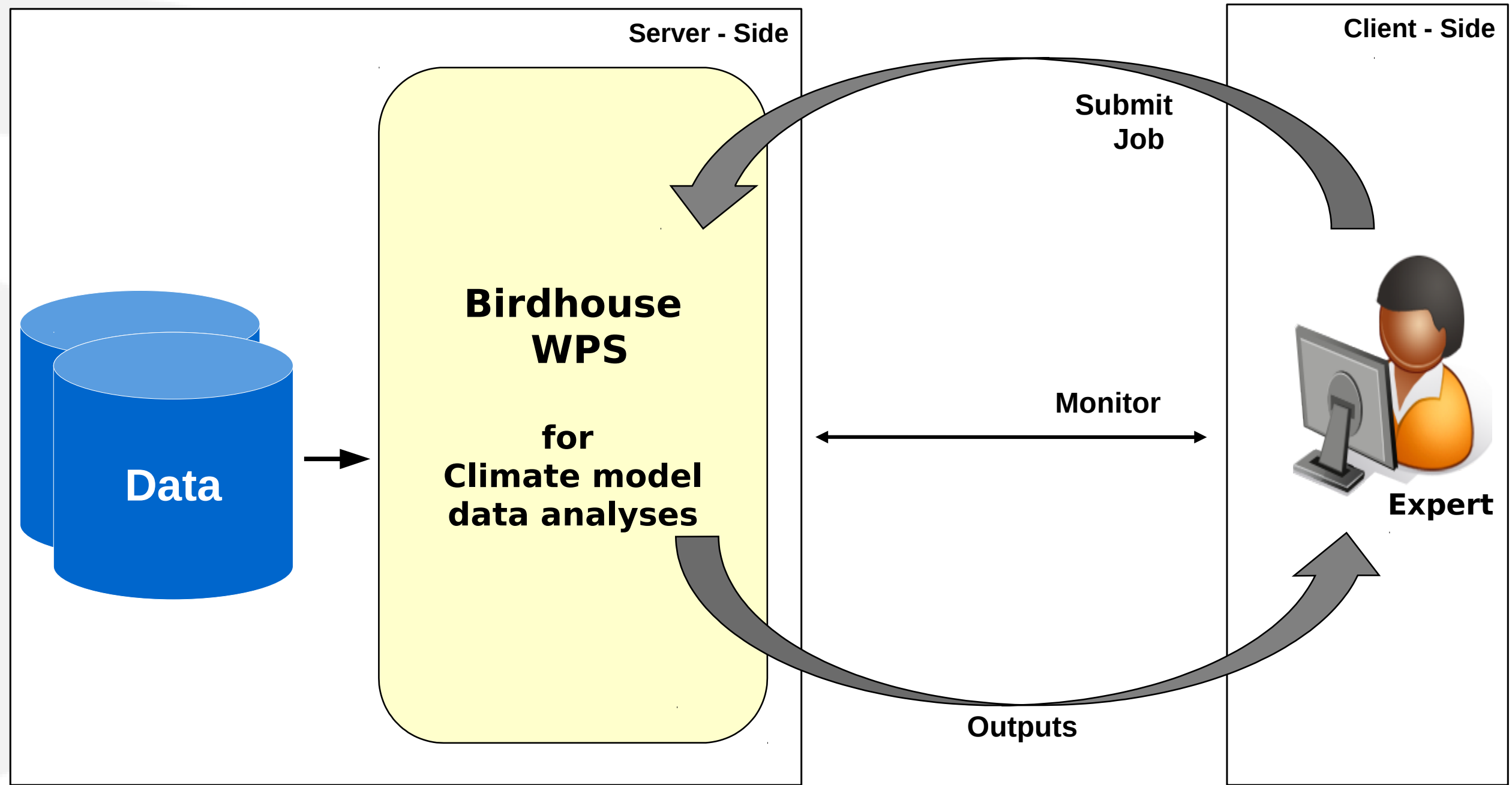
**Processing  
in or close to  
Data archives**



**Web Processing Service**

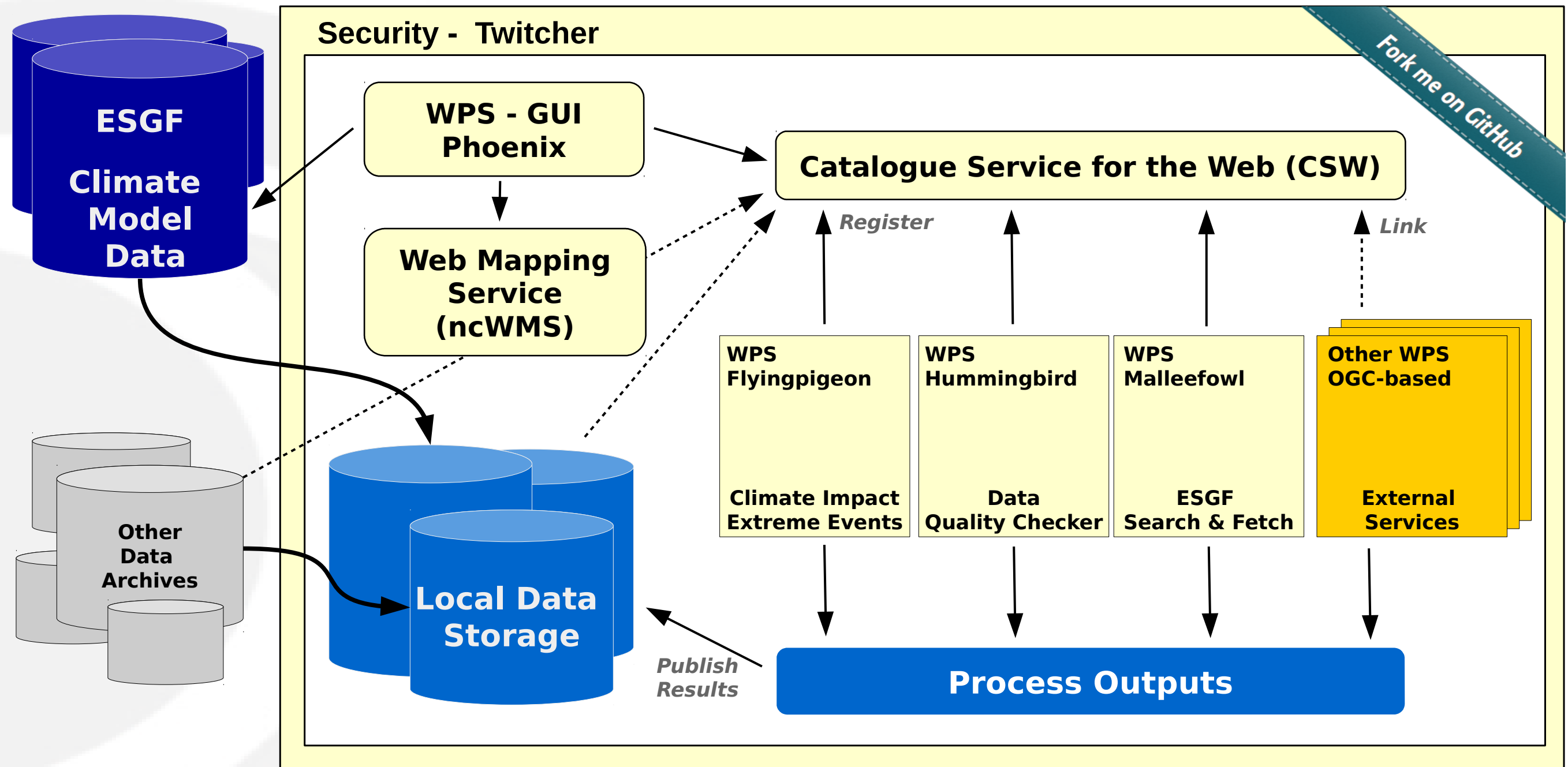
**Submit jobs on a Server  
close to the data**

# Server-Client Side



# What does Birdhouse provide ?

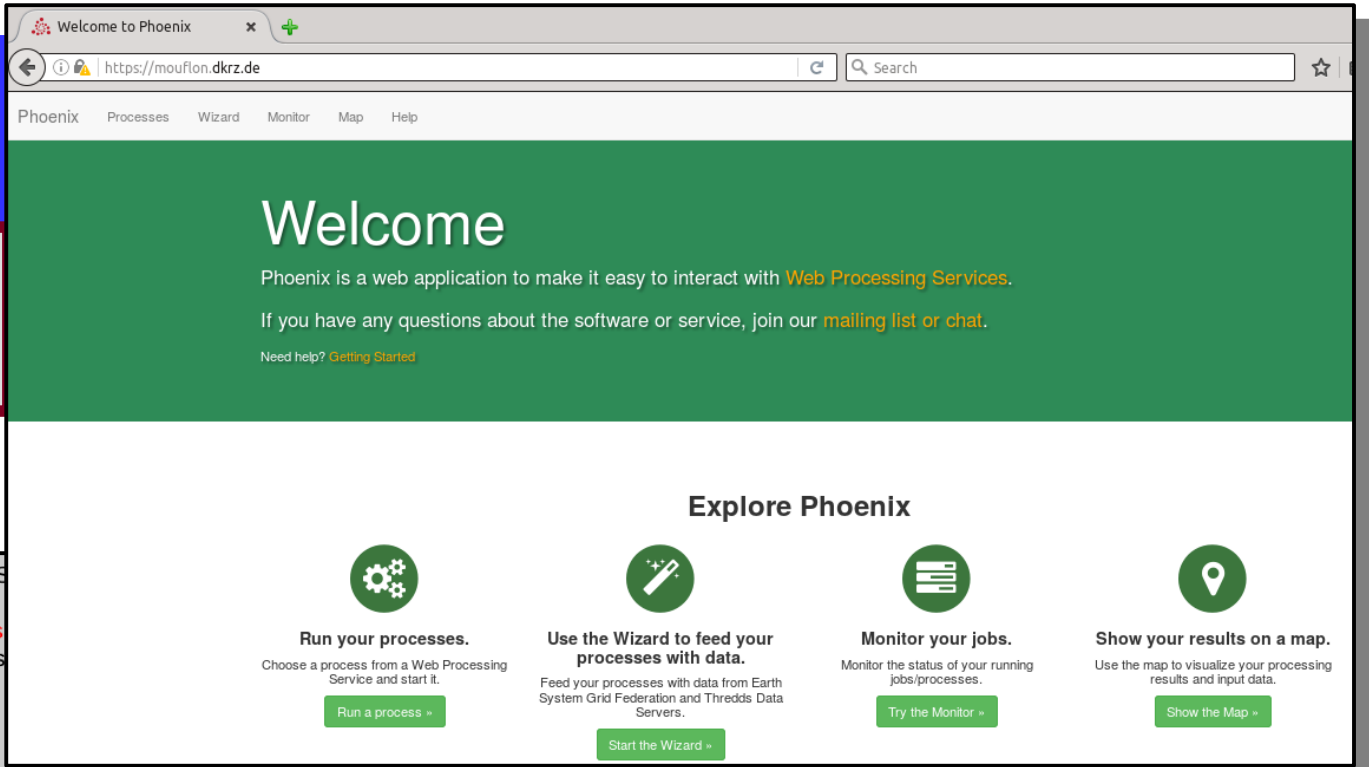
- **Customizable installation of Web Processing Services using conda, buildout and ansible.**
- **Provides WPS as Docker Container.**
- **Web-based and Terminal WPS clients.**
- **Security Proxy Twitcher for OGC/WPS services.**
- **Data Access : ESGF, Thredds, OpenStack Cloud, ...**
- **WPS for compliance checks and climate impact.**
- **Supports PyWPS 3.x and 4.x ... but not restricted to it (Zoo, GeoServer WPS, COWS, 52North, ...)**



# Client Side

## Web Browser GUI

Authentication with OAuth or OpenID



## Script language Terminal Call

Token authentication

```
[nhempel@lsce3199 ~]$ export WPS_SERVICE=https://mouflon.dkrz.de:8090
[nhempel@lsce3199 ~]$ birdy -h

usage: birdy [<options>] <command> [<args>]

Flyingpigeon: Processes for climate data, indices and extrem events

optional arguments:
-h, --help            show this help message and exit
--debug              enable debug mode

command:
List of available commands (wps processes)

{visualisation,sdm,segetalflora,indices_single,subset_countries,eobs_to_cordex,ensembleRobustness,analogs,fetch}

visualisation  Visualisation of netcdf files:
sdm            Species distribution model:
segetalflora   Segetal Flora:

indices_single Calculation of climate indice (single variable):
subset_countries Subset netCDF files:
eobs_to_cordex  EObs to CORDEX:
ensembleRobustness Calculation of the robustness of an ensemble:

analogs        Days with analog pressure pattern:
fetch          Download Resources:
```

```
from owslib.wps import WebProcessingService
wps = WebProcessingService(url="https://mouflon.dkrz.de:8090",
                           verbose=False,
                           token="your_token")

execute = wps.execute(
    identifier="niceprocess",
    inputs=[
        ("parameter_1", "argument"),
        ("parameter_2", "42"),
        ("parameter_3", "0.987"), # use the default value
        ("file_identifier", "https://thredds/fileServer1/test/file1.nc"),
        ("file_identifier", "https://thredds/fileServer1/test/file2.nc"),
        ("file_identifier", "https://thredds/fileServer2/test/file3.nc")],
    output=[("output", True)])

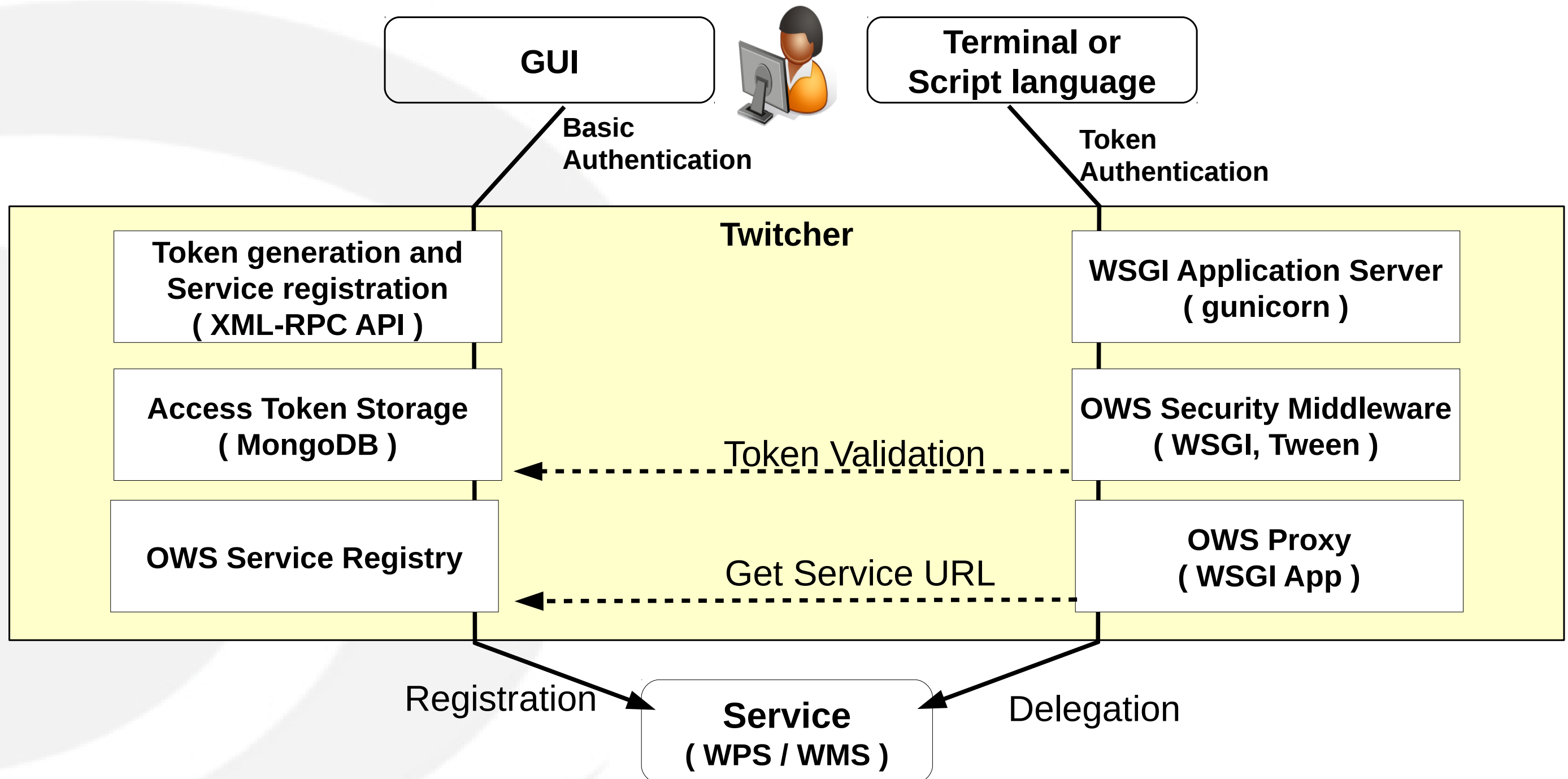
# time for a coffee

for o in execute.processOutputs:
    print o.reference

https://mouflon.dkrz.de:8090/wpsoutputs/flyingpigeon/output_graphic-697dee76-d722-93ae-9789bf75cf44.png
https://mouflon.dkrz.de:8090/wpsoutputs/flyingpigeon/output_netCDF-697dee76-d722-93ae-9789bf75cf44.nc
https://mouflon.dkrz.de:8090/wpsoutputs/flyingpigeon/output_text-697dee76-d722-93ae-9789bf75cf44.txt
```


Just testing a nice script to visualise some variables  
Species distribution model  
Species biodiversity of segetal flora. Input files: variable:tas , domain: EUR-11 or EUR-44  
This process calculates climate indices based on one single variable.  
This process returns only the given polygon from input netCDF files.  
downloads EObs data in adapted CORDEX format  
Calculates the robustness as the ratio of noise to signal in an ensemble of timeseries  
Search for day with analog pressure pattern  
This process downloads resources (limited to 50GB) to the local file system and returns a textfile with appropriate paths

# Security



# Security Token

Phoenix Processes Wizard Monitor Map Help

       Phoenix ▾

## Phoenix

Profile

Personal access token

C4I access token

ESGF access token

Group Permission

Personal access token

Generate Token

**Twitcher access token**

1907471138d44bec9ef71fc9ecd9d72e

**Expires**

2016-11-28 01:51:23 UTC

Powered by [Birdhouse](#) | Get the code on [GitHub](#) | Version v0.6.2



# Script language

```
from owslib.wps import WebProcessingService, monitorExecution

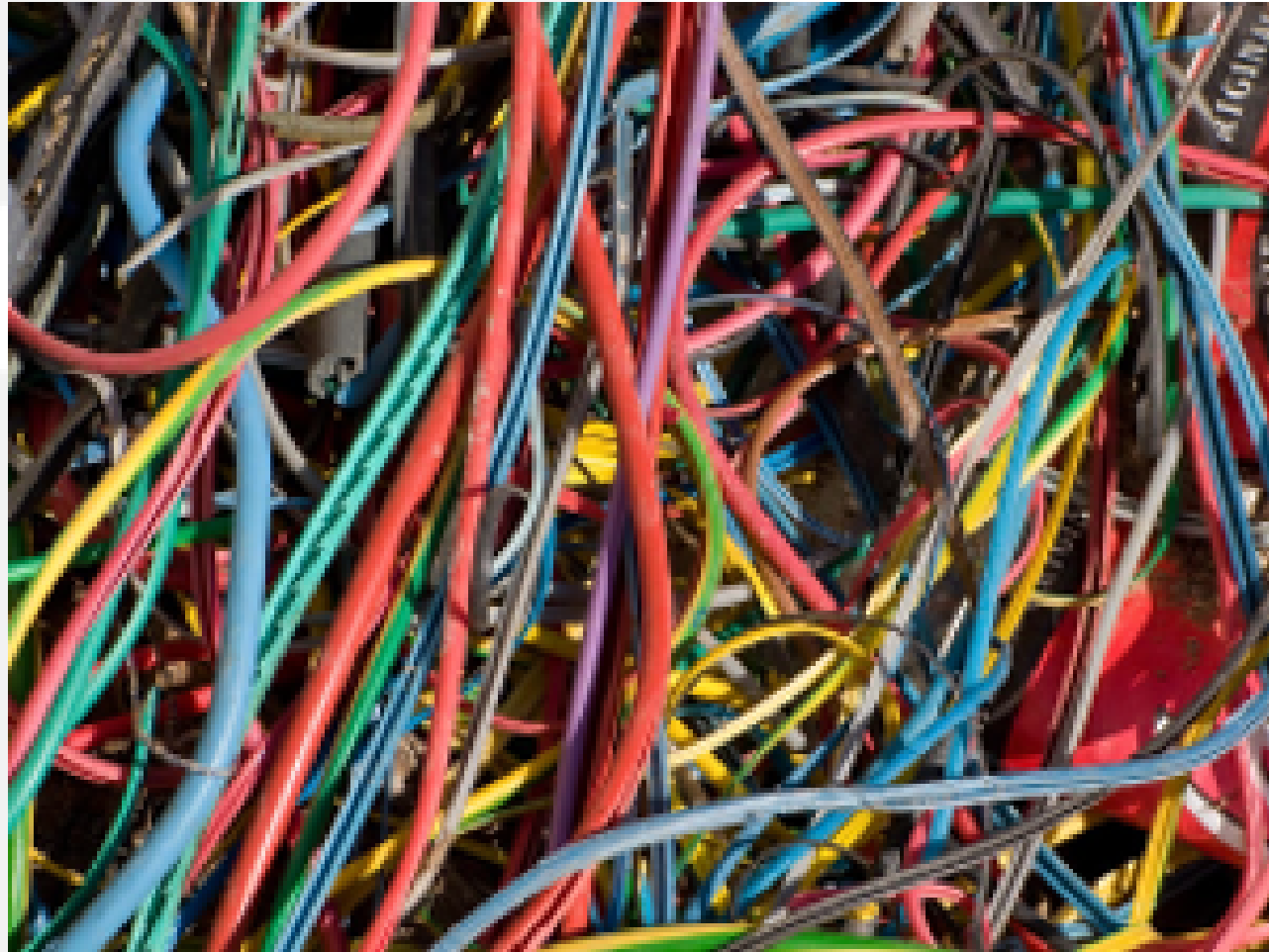
# using wps url with access token db6c...
wps = WebProcessingService(
    url="https://mouflon.dkrz.de/ows/proxy/flyingpigeon/db6c1293d0444d919dcc3ce48fa610f7 ", \
    verify=False,
    verbose=False, skip_caps=False,
)

execute = wps.execute(
    identifier="niceprocess",
    inputs=[
        ("parameter_1", "argument"),
        ("parameter_2", "42"),
        # ("parameter_3", "0.987"), # use the default value
        ("file_identifier", "https://thredds/fileServer1/test/file1.nc"),
        ("file_identifier", "https://thredds/fileServer1/test/file2.nc"),
        ("file_identifier", "https://thredds/fileServer2/test/file3.nc")],
    output=[("output", True)])

for o in execute.processOutputs:
    print o.reference

https://mouflon.dkrz.de:8090/wpsoutputs/flyingpigeon/output\_graphic-697dee76-d722-93ae-9789bf75cf44.png
https://mouflon.dkrz.de:8090/wpsoutputs/flyingpigeon/output\_netCDF-697dee76-d722-93ae-9789bf75cf44.nc
```

# Deployment with conda and buildout



Using conda package manager to setup an environment with all used software components (python, R, matplotlib, PyWPS, ...)

Using buildout to setup PyWPS with all services (supervisor, gunicorn, nginx) and configuration files.

To install a *Bird* just run :

```
$ git clone https://github.com/bird-house/emu.git  
$ cd emu  
$ make clean install  
$ make start
```

<http://conda.pydata.org/docs/>

<http://www.buildout.org/en/latest/>

<http://birdhouse.readthedocs.io/en/latest/installation.html>

# ESGF – search

[Wizard](#) [Monitor](#) [Map](#) [Help](#)

**ESGF Search \***

Datasets found: 18

➤ Search Options

➤ Freetext Search

▼ Your keyword selections

project:CORDEX × domain:EUR-11 × experiment:historical × experiment:rcp85 × time\_frequency:day × variable:tas ×

▼ Categories

access data\_node driving\_model ensemble experiment experiment\_family institute rcm\_name rcm\_version version

▼ Keywords: variable

tas

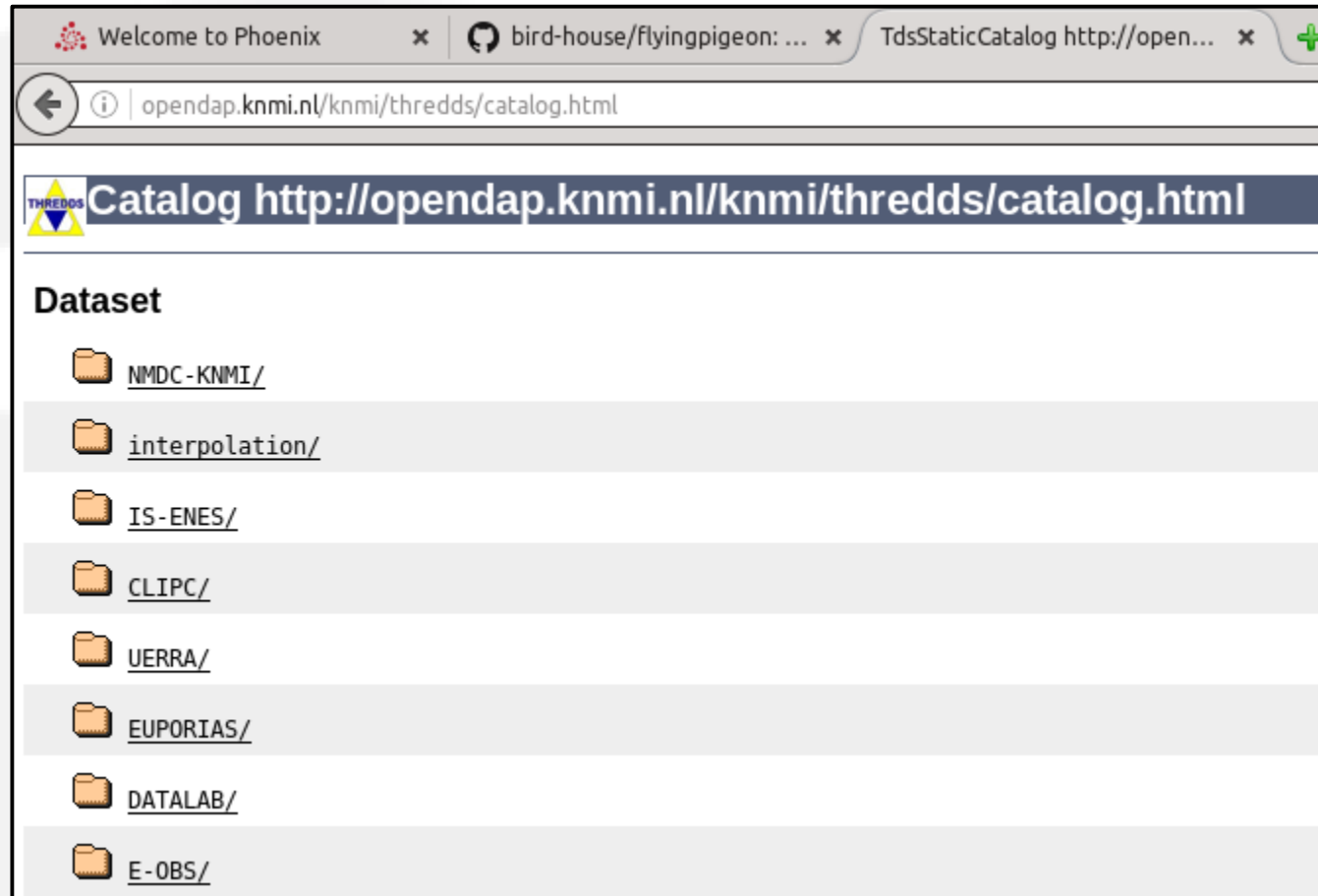
➤ Date

Previous

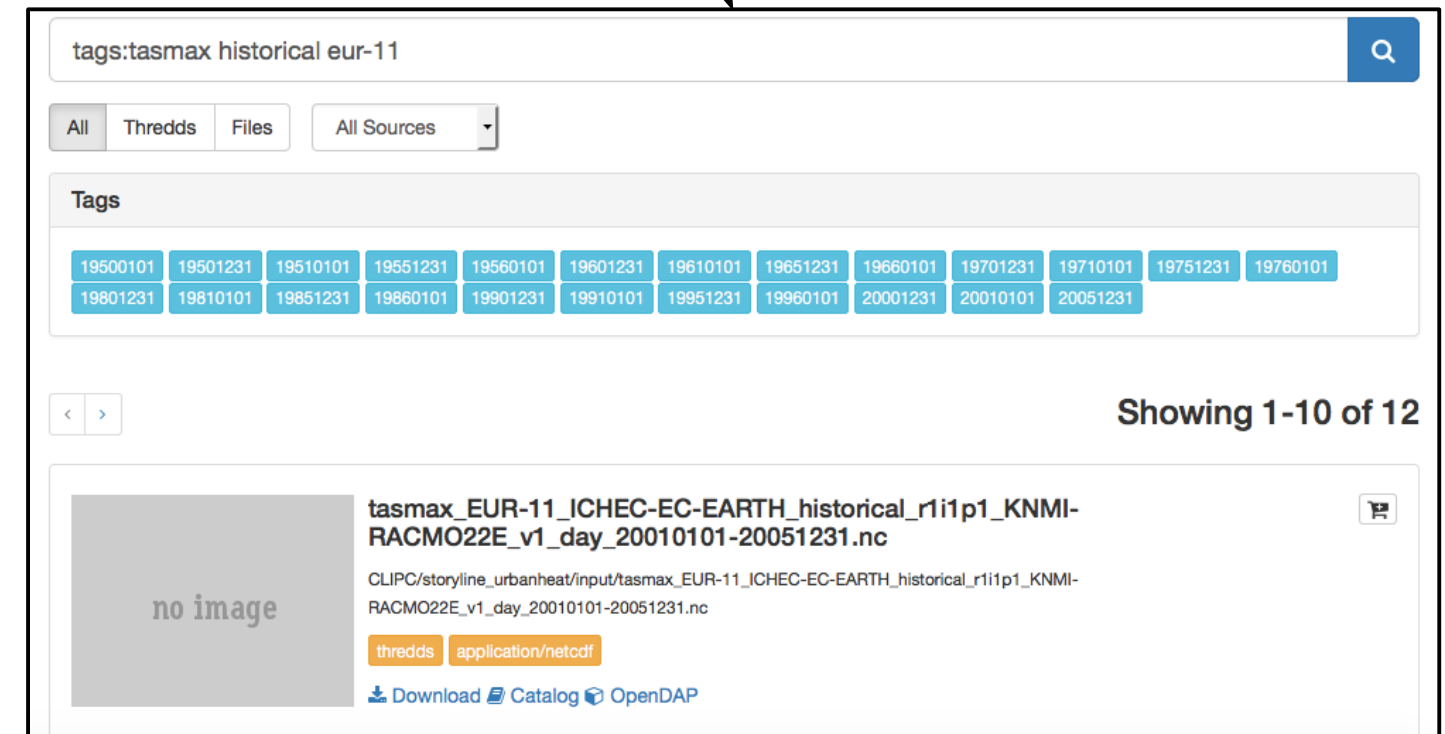
Cancel

Next

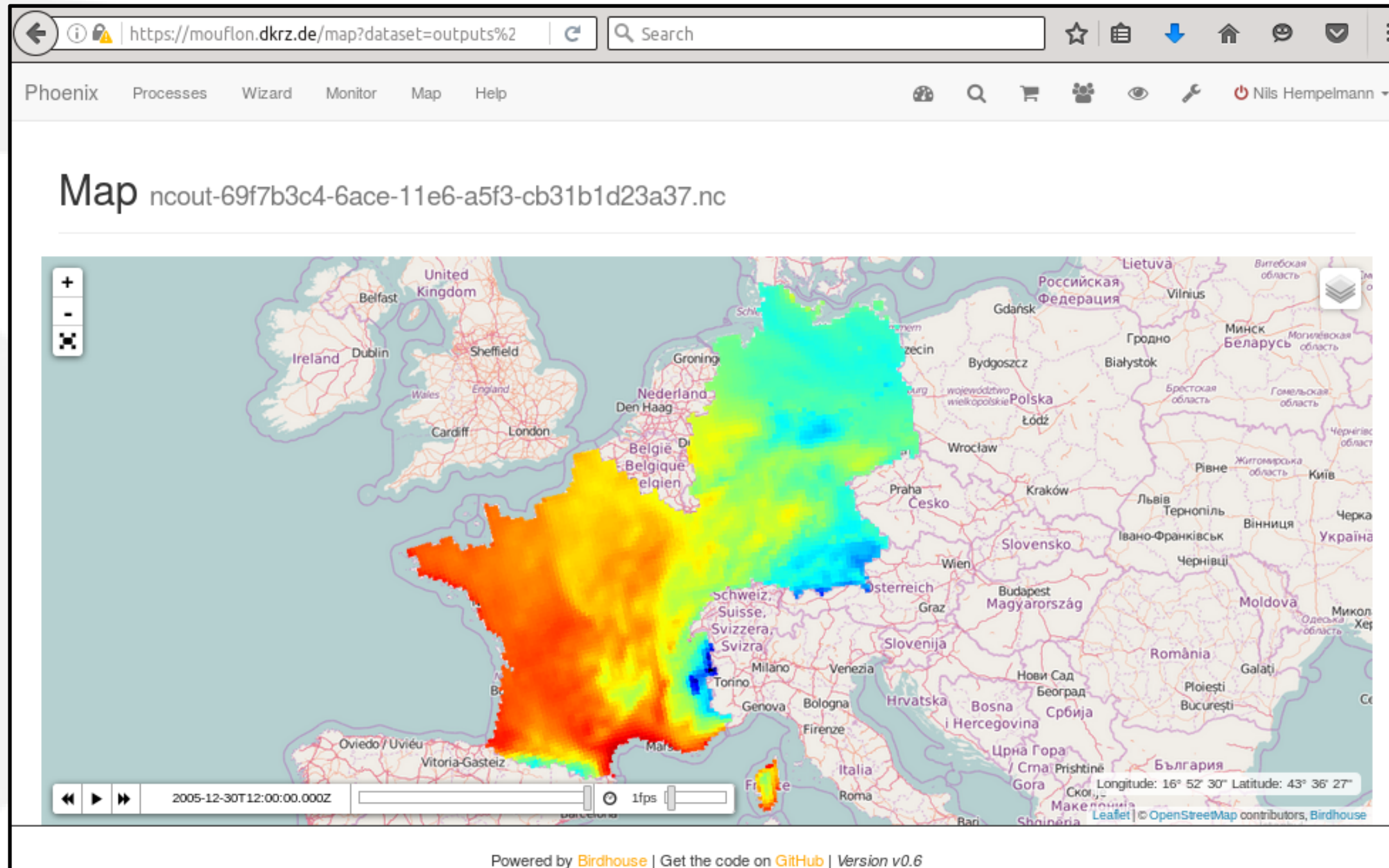
# Solr Index for Data ( bird-feeder )



Run bird-feeder to create Solr search Index for Thredds Data Catalogs and local data



# Web Mapping Server





# SpotChecker : Metadata Compliance Checks

**Spot Checker** Please complete the form below and submit a job.

The Spot Checker is a Python tool to check local/remote datasets against a variety of compliance standards. Each compliance standard is executed by a Check Suite, which functions similar to a Python standard Unit Test. A Check Suite runs one or more checks against a dataset, returning a list of Results which are then aggregated into a summary. Available compliance standards are the Climate and Forecast conventions (CF) and project specific rules for CMIP6 and CORDEX.

[View as XML](#) [Birdhouse](#) [User Guide](#) [CF Conventions](#) [IOOS Compliance Online Checker](#)

Test Suite \*

cf

Select the test you want to run. Default: cf (climate forecast conventions)

NetCDF File

URL Enter a URL pointing to your resource

Enter a URL pointing to a NetCDF file (optional)

Remote OpenDAP Data URL

il/knmi/thredds/dodsC/CLIPC/gerics/gerics-members/tasmax/su\_cdo-1-6-3\_GERICS\_ens-multiModel\_rcp85\_mixed\_ens-multiModel\_v1\_EUR-11\_yr\_20060101-20991231.nc

Or provide a remote OpenDAP data URL, for example: <http://my.opendap/thredds/dodsC/path/to/file.nc>

Execute

**Your dataset scored 239 out of 244 points**

During the cf check

For dataset [http://opendap.knmi.nl/knmi/thredds/dodsC/CLIPC/gerics/gerics-members/tasmax/su\\_cdo-1-6-3\\_GERICS\\_ens-multiModel\\_rcp85\\_mixed\\_ens-multiModel\\_v1\\_EUR-11\\_yr\\_20060101-20991231.nc](http://opendap.knmi.nl/knmi/thredds/dodsC/CLIPC/gerics/gerics-members/tasmax/su_cdo-1-6-3_GERICS_ens-multiModel_rcp85_mixed_ens-multiModel_v1_EUR-11_yr_20060101-20991231.nc)

Scoring Breakdown:

High Priority  1

| Name   | Score |
|--|-------|
| §2.2 Valid netCDF data types   | 11/11 |
| §2.4 Unique dimensions   | 11/11 |
| §3.1 Variable lat contains valid CF units                                | 3/3   |
| §3.1 Variable lat's units are appropriate for the standard_name latitude | 1/1   |

Run SpotChecker on NetCDF file :  
File URL, OpenDAP URL or uploaded File.

Perform compliance checks :  
CF conventions, CORDEX, CMIP5, ...

Using IOOS Compliance-Checker and  
DKRZ Quality Assurance Checker.

Check reports in HTML and YAML format.

# Subsetting : Region USA

Enter NetCDF File URL and USA clipping Region

**Resource \***

Resource

URL

[Add Resource](#)

**NetCDF File**

**Region \***

Region

[Add Region](#)

Execute Job

**Monitor Job Execution**

Running 1 Finished 7 Matching 8 Sort [C](#)

| <input type="checkbox"/> | Status | User    | Process          | Service      | Caption | Finished | Duration | Labels  |   |
|--------------------------|--------|---------|------------------|--------------|---------|----------|----------|---|---|
| <input type="checkbox"/> | ⚙      | Phoenix | subset_countries | flyingpigeon | ???     | ???      | 0:00:31  | <a href="#">dev</a> <a href="#">single</a> <a href="#">async</a><br><a href="#">edit labels</a> | <a href="#">Details</a> <a href="#">Restart</a> |

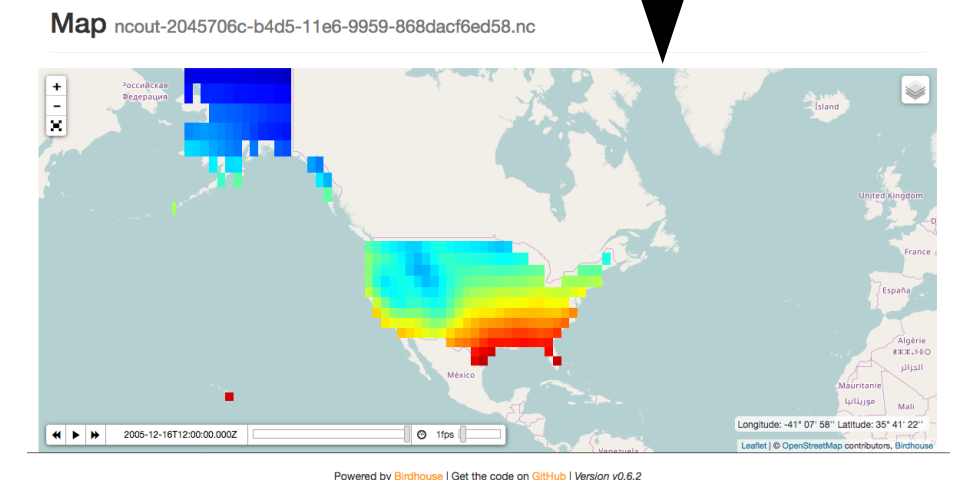
Show  
Outputs

[Log](#) [Inputs](#) [Outputs](#) [View as XML](#)

**Subsets** Parameter **output**, a WPS ComplexType  
Tar archive containing the netCDF files  
[output-2045706c-b4d5-11e6-9959-868dac6ed58.tar](#)  
[application/x-tar](#)  
[Download](#) [Share](#)

**Subsets for one dataset** Parameter **ncout**, a WPS ComplexType  
NetCDF file with subsets of one dataset.  
[ncout-2045706c-b4d5-11e6-9959-868dac6ed58.nc](#)  
[application/x-netcdf](#)  
[Download](#) [Share](#) [Show on Map](#)

Show Map



# ESMValTool Diagnostics as Web Processing Service

## 1. Choose ESMVal Process

Home / Processes / esmvaltool

**Description**

WPS processes for ESMValTool.

[XML](#) [Provider: EsmValTool](#)

**Processes**

- ⚙️ **ESMValTool: surface contour plot for precipitation 1.0**  
Tutorial contour plot used in the doc/overview.pdf.
- ⚙️ **ESMValTool: tutorial diagnostic. 1.0**  
Tutorial diagnostic used in the doc/toy-diagnostic-tutorial.pdf.

## 2. Enter Input Parameters

**Description**

Tutorial contour plot used in the doc/overview.pdf.

[XML](#) [Birdhouse](#) [ESMValTool](#)

**Run async \***

☒

Check this to run process async.

**Model \***

MPI-ESM-LR

Choose a model like MPI-ESM-LR.

**Experiment \***

historical

Choose an experiment like historical.

**Ensemble \***

r1i1p1

Choose an ensemble like r1i1p1.

**Start year \***

1990

Start year of model data.

**End year \***

2000

End year of model data.

**Execute**

## 3. Outputs : plot, namelist, log

Log Inputs Outputs

**Output plot** Parameter **output**, a WPS ComplexType  
Generated output plot of ESMValTool processing.

surfconplot\_simple\_pr\_T2Ms\_ANNjcrpMJ.pdf

[application/pdf](#)

[Download](#) [Share](#)

**namelist** Parameter **namelist**, a WPS ComplexType  
ESMValTool namelist used for processing.

namelistvGe9v4.xml

[text/plain](#)

[Download](#) [Share](#)

**Log File** Parameter **Log**, a WPS ComplexType  
Log File of ESMValTool processing

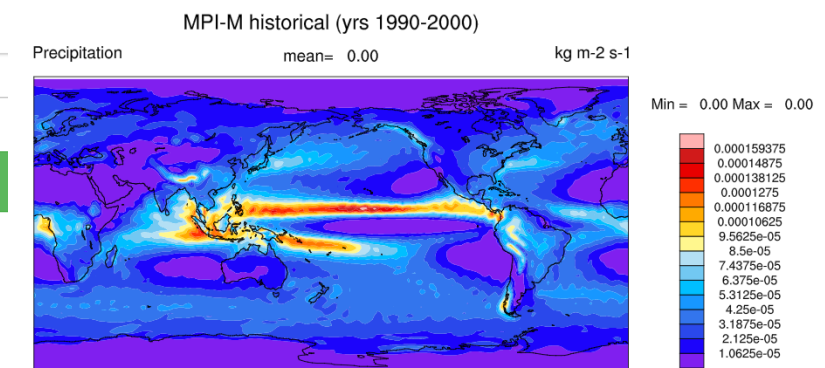
loge1GhQG.txt

[text/plain](#)

[Download](#) [Share](#)

Show Plot

ANN



ESMValTool namelist is generated by the WPS process.  
Data is retrieved by the ESMValTool ESGF coupling module.



# Accessing Remote WPS : Climate4Impact WPS at KNMI

## Register WPS URL

Home / Settings / Services

< KNMI Remove Service

Name: KNMI

URL: <http://climate4impact.eu/impactportal/WPS>

Service Type: WPS

Abstract: See <http://pywps.wald.intevation.org> and <http://www.opengeospatial.org/standards/wps>

Keywords: GIS WPS

References:

Rights: none

Creator: IS-ENES

Public Access:

Service Name: knmi

## Update C4I Access Token

Phoenix

Profile

C4I access token Generate C4I Token

Personal access token

C4I access token

ESGF access token

Group Permission

C4I access token

91d80687-01c6-4944-936e-c667f64f8369

Update C4I Token

## Run a Process : operation on two numbers

PyWPS Server Please choose one of the processes to submit a job.

See <http://pywps.wald.intevation.org> and <http://www.opengeospatial.org/standards/wps>

Capabilities (XML) IS-ENES

### CLIPC Create statistics per NUTS region Identify 1.0

Identify process for statistics per NUTS region calculations

★ 3

### CLIPC Create statistics per NUTS region Execute 1.0

The NUTS extractor calculates statistics for any NetCDF file by extracting geographical areas defined in a GeoJSON file. The statistics per geographical area include minimum, maximum, mean and standard deviation. The statistics are presented in a CSV table and a NetCDF file.

★ 3

### CLIPC Combine Identify 1.0

Lists possible operations for two resources for the CL

### CLIPC Combine Execute 1.0

Performs operation on two nc files and returns the an

### CLIPC ICCLIM simple indicator calculat

Identify function for ICCLIM simple indicator calculat

### CLIPC ICCLIM simple indicator calculat

Using ICCLIM, single input indices of temperature TG R20mm, RX1day, RX5day; and of snowfall: SD, SD1, :

### CLIPC DRS Checker 1.0

Checks file for correct DRS

Perform operation on two numbers Please complete the form below and submit a job.

Performs operation on two numbers and returns the answer

View as XML

Run async \*

☐

Check this to run process async.

Input 1

2.0

Input 1

Input 2

5.0

Input 2

operator

multiply

Execute

Log Inputs Outputs View as XML

no image

Binary operator result Parameter **answer**, a WPS LiteralType

No summary

10.0

- **Using PyWPS-4 : ready for WPS 2.0 (pause, resume, delete)**
- **Attach batch processing with SLURM etc ...**
- **Optionally run processes in Docker Container.**
- **Delegation to SLURM and Docker is handled internally of PyWPS (new feature in PyWPS-4).**
- **Process code and Process definition is not changed when run as batch process or in a docker container.**

- **<https://github.com/bird-house>**
- **<http://birdhouse.readthedocs.org/en/latest/>**
- **<https://gitter.im/bird-house/birdhouse>**
- **<https://lists.dkrz.de/mailman/listinfo/wps>**
- **<https://lists.dkrz.de/mailman/listinfo/wps-dev>**
- **DEMO GUI: <https://mouflon.dkrz.de>**



---

**Contact :**

wps@dkrz.de

**Thanks to :**

Carmen Alvarez-Castro, Katharina Berger, Patrick Brockmann, Carsten Ehbrecht, Wolfgang Falk, Nils Hempelmann, Heinz-Dieter Hollweg, Jörg Hoffmann, Nikolay Kadygrov, Stephan Kindermann, Florian Klemme, Nikolay Koldunov, Ben Koziol, Cathy Nangini, Sabine Radanovics, Seckmag, Robert Vautard, Pascal Yiou , .... , et. al.

# Terminal Call

```
[nhempel@lsce3199 ~]$ conda install -c birdhouse birdhouse-birdy
```

```
[nhempel@lsce3199 ~]$ birdy -h
```

```
usage: birdy [<options>] <command> [<args>]
```

**Flyingpigeon: Processes for climate data, indices and extreme events**

**optional arguments:**

**-h, --help** show this help message and exit

**--debug** enable debug mode

**--token TOKEN, -t TOKEN**

Token to access the WPS service.

**command:**

List of available commands (wps processes)

# Terminal Call

```
[nhempel@lsce3199 ~]$ export WPS_SERVICE=https://mouflon.dkrz.de/ows/proxy/flyingpigeon
```

```
[nhempel@lsce3199 ~]$ birdy -token 0c6d305b0f42452cbdcf31c7ac74f1e1 \  
analogs_detection --experiment 'NCEP_slp'
```

```
INFO:Execution status: ProcessAccepted
```

```
INFO:Execution status: ProcessStarted
```

```
INFO:Execution status: ProcessSucceeded
```

```
INFO:Output:
```

```
INFO:analogs=http://mouflon.dkrz.de/wpsoutputs/flyingpigeon/analogs-08bce60c-6a41-11e6-be7a-8fdf4b12fcf5.txt (text/plain)
```

```
INFO:config=http://mouflon.dkrz.de/wpsoutputs/flyingpigeon/config-08bce60c-6a41-11e6-be7a-8fdf4b12fcf5.txt (text/plain)
```

```
[nhempel@lsce3199 ~]$
```

<http://twitcher.readthedocs.io/en/latest/tutorial.html>

# Analogue of atmospheric Circulation

