Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Swiss Confederation

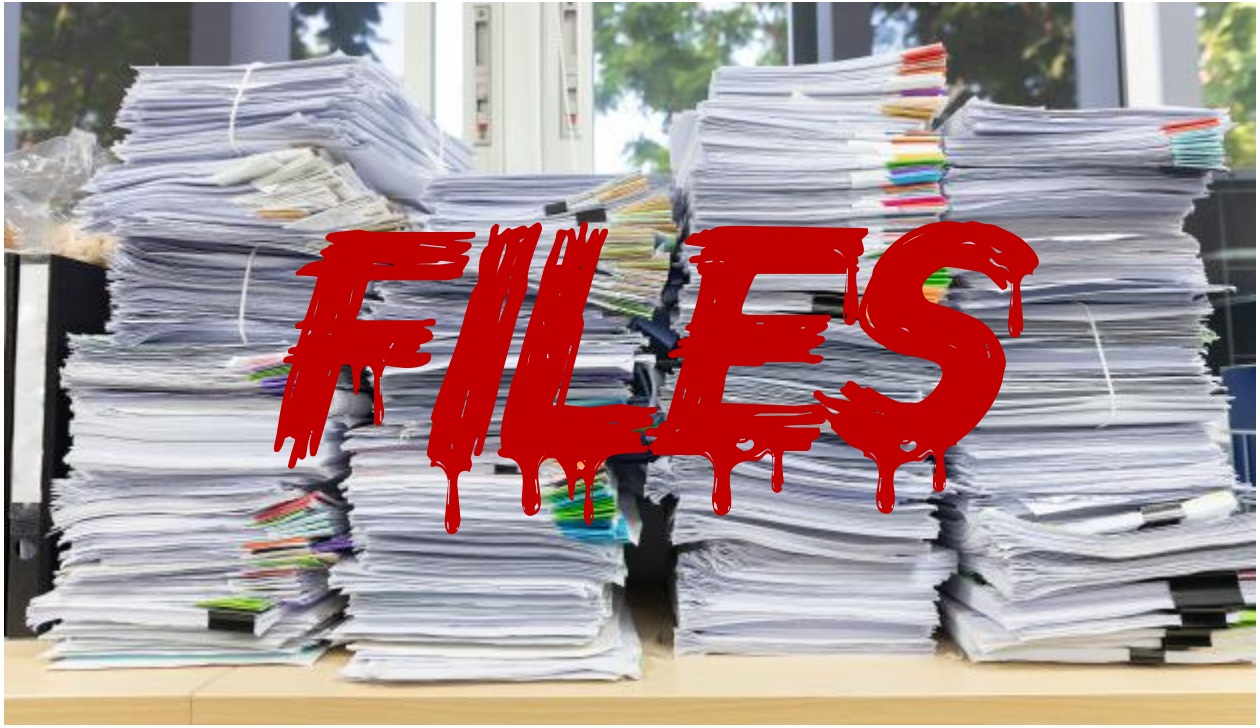
Federal Department of Home Affairs FDHA
Federal Office of Meteorology and Climatology MeteoSwiss

Data service for gridded data

Code name: Gridefix

Philipp Falke, Matthieu Bernard, Gabriela Aznar

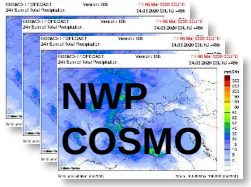
In the beginning there was nothing but …

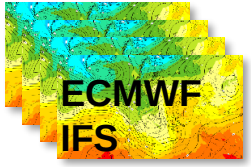


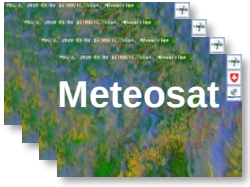
First prototype of database for grid data

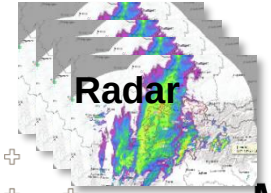
- Catalogue for files
- Supports NWP model and satellite data
- Abstract the concept of files though OPeNDAP API
  - Always 2 API calls: **catalogue** and **dataset**
  - Redundant slicing of arrays
  - Lots of on-the-fly processing
  - Hard to access data in random manner

Unified access to feed data pipelines







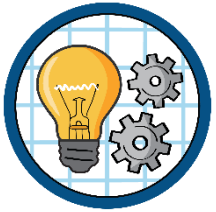


Grid database

Production

Development

Transfer of methods



MeteoSwiss

Two main use cases

Production

- Near real-time availability of data
- Rolling archive
- Publish gridded data
- Operational real-time processing

Development

- Historical archive of model output and other gridded data
- Development and Training of Postprocessing (incl. Machine learning), Verification, Climatology

➢ Requirements

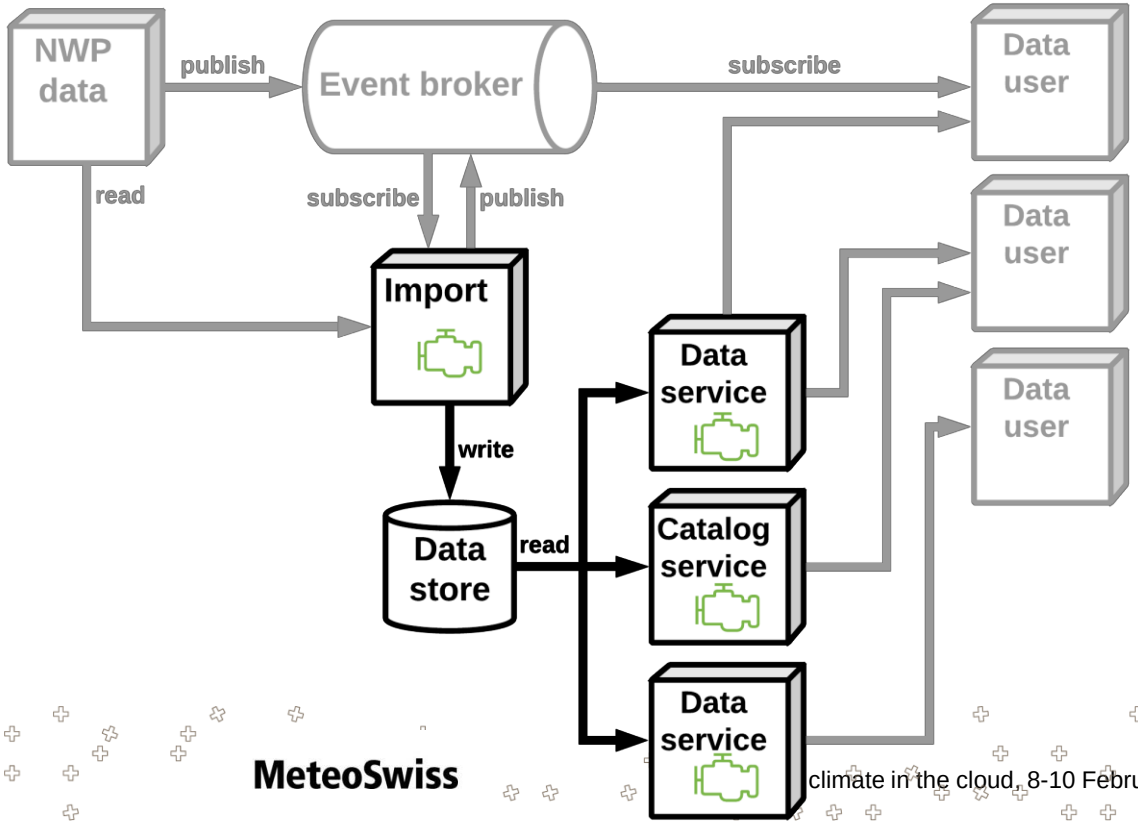
- Store multi-dimensional data (reference time, lead time, members, levels, x, y)
- Import heterogeneous data
- Language independent data access
- Data catalog for finding the right data
- Real-time availability of data
- Sufficient performance for data access along every dimension

Implementation choices

- «Cloud-native»
  - Horizontal scalability
  - Services exposing APIs
  - Object store
- *Import* rather than *indexing*

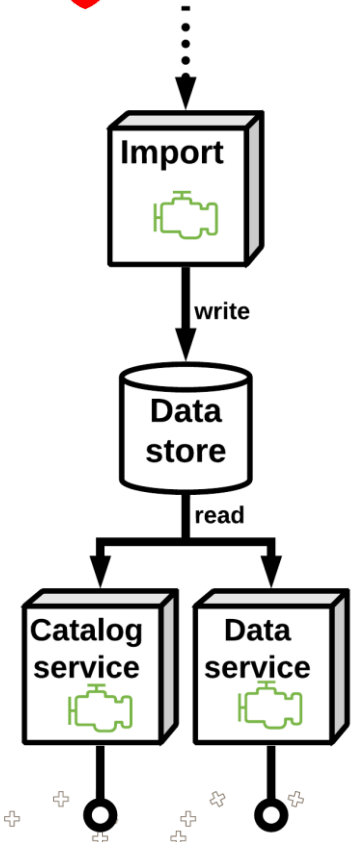
OPENAPI
INITIATIVE

Architecture overview



- Common data access layer
- Independent services providing different views on data
- Potentially multiple data services

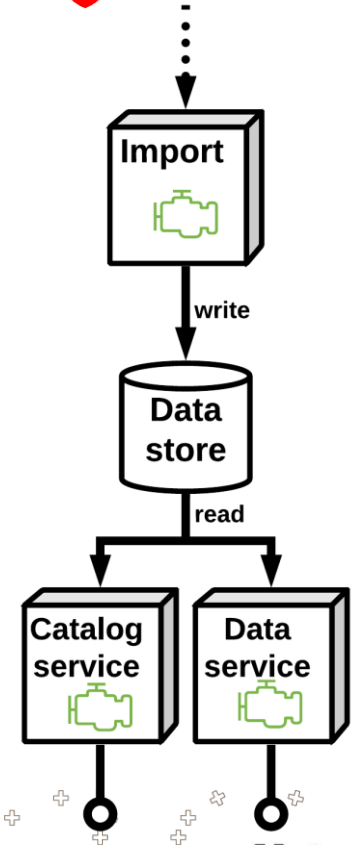
Data access layer: gridefix-core



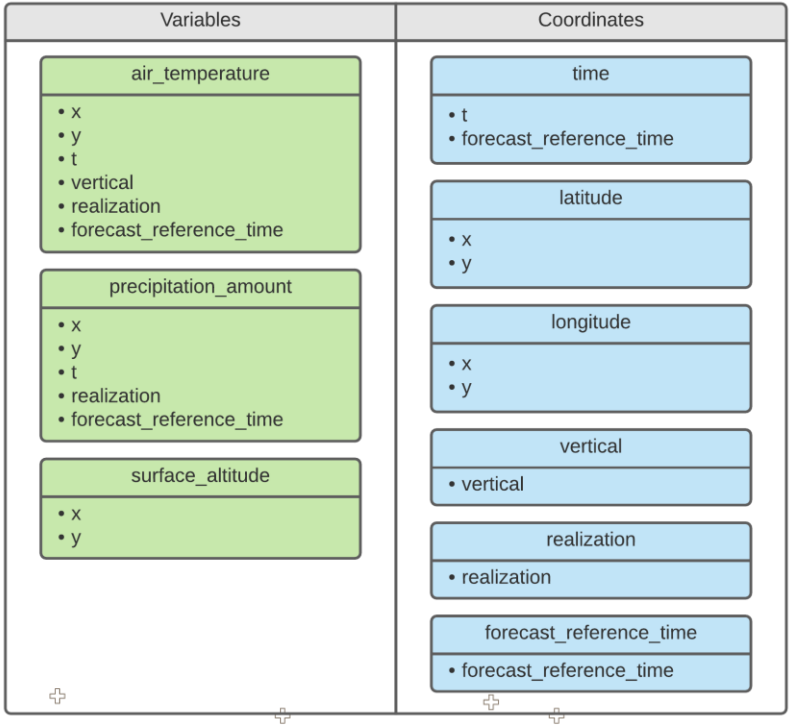
- Abstraction layer for data store access
- Python library for efficient interaction with storage
- Implements storage format
- Enforces data model
- Exposes stored data as xarray datasets
- Makes use of xarray accessors

MeteoSwiss

Data Model

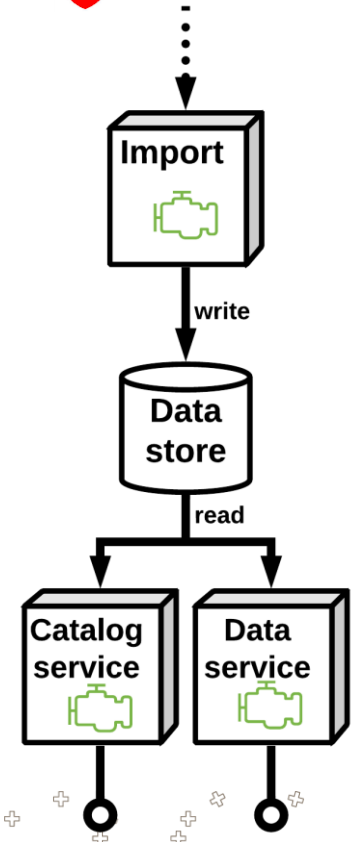


- Variable names follow NetCDF CF Metadata Convention
- Coordinates are stored in latitude / longitude
- Attributes can be set on arrays, variables, coordinates, the major time axis



MeteoSwiss

Selecting datasets

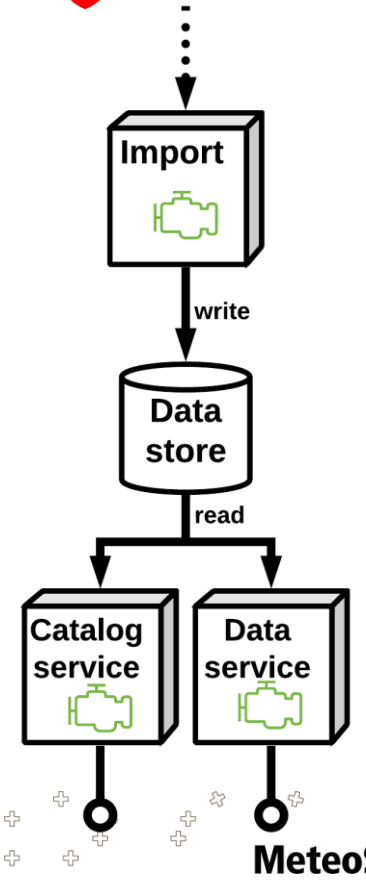


- No hierarchy (almost):
  - A dataset is identified by a source name (i.e. the model name) and a unique set of tags (*operational forecast, re-forecast, analysis, observation*)

We define:

- **Array**: multi-dimensional array with multiple variables
- **Dataset**: Can consist of multiple arrays, merged along the major time axis to avoid storing/transmitting blocks of NaN

Storage format



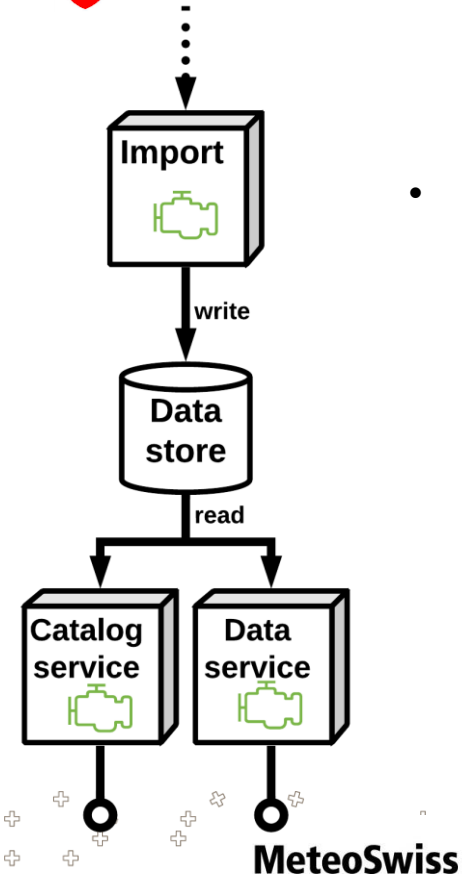
- Leveraging **zarr** and **dask**
- Rolling archive along major time axis
- Attributes on relevant hierarchy levels

Performance optimisations:

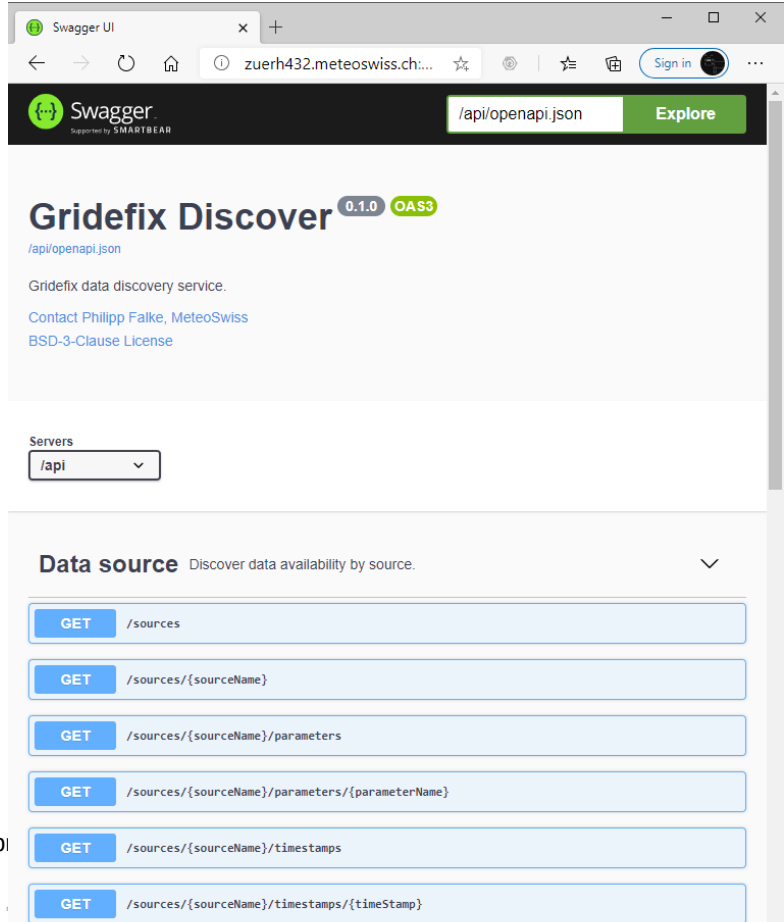
- zarr meta-data in MongoDB or Redis
- zarr chunks in object store

```
COSMO-1                                                    source
├── 30a23917-11e7-4950-82b2-e8690184ac55                   array
│   ├── coordinates
│   │   ├── latitude
│   │   ├── longitude
│   │   ├── time
│   │   └── vertical
│   ├── 2020-01-01T00:00:00                                time axis group
│   │   ├── air_temperature
│   │   ├── eastward_wind
│   │   ├── mass_fraction_of_cloud_liquid_water_in_air
│   │   └── northward_wind
│   └── 2020-01-01T06:00:00
│       ├── air_temperature
│       ├── eastward_wind
│       ├── mass_fraction_of_cloud_liquid_water_in_air
│       └── northward_wind
├── 9ed433bb-ebfc-4f00-8f31-f0441d8c703f
│   ├── coordinates
│   │   ├── latitude
│   │   ├── longitude
│   │   ├── time
│   │   └── vertical
│   └── 2020-01-01T03:00:00
│       ├── air_temperature
│       ├── eastward_wind
│       ├── mass_fraction_of_cloud_liquid_water_in_air
│       └── northward_wind
├── 482505bf-f2c3-43bf-867c-e34f6156bb21
...
```

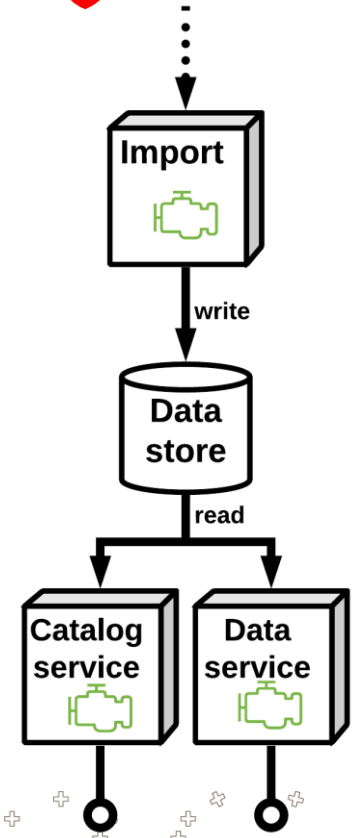
Data catalogue: gridefix-discover



- High level interface to programmatically access all meta-data



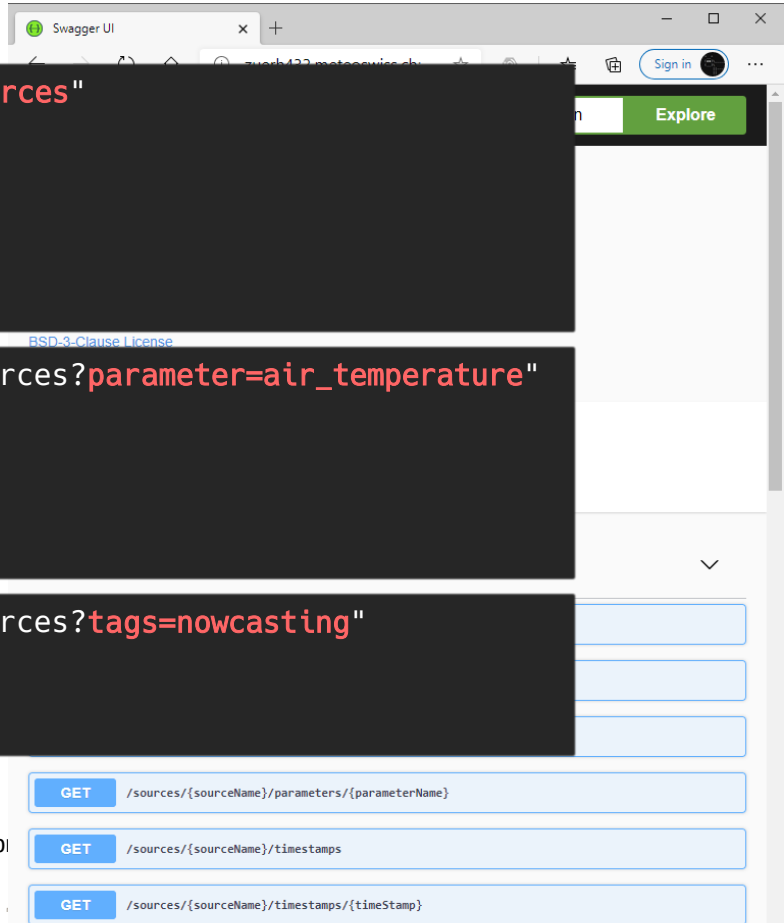
Data catalogue: gridefix-discover




```
❯ curl "http://gride.fix/discover/api/sources"
[
  "COSMO-1E",
  "COSMO-2E",
  "ECMWF_IFS",
  "INCA"
]
```

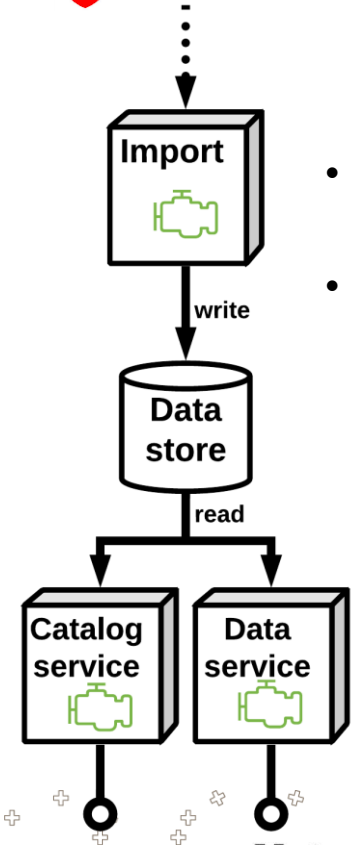
```
❯ curl "http://gride.fix/discover/api/sources?parameter=air_temperature"
[
  "COSMO-1E",
  "COSMO-2E",
  "ECMWF_IFS"
]
```

```
❯ curl "http://gride.fix/discover/api/sources?tags=nowcasting"
[
  "INCA"
]
```



MeteoSwiss

NetCDF over the net: gridefix-retrieve



- Own implementation of OPeNDAP
- Rely on lazy loading principle
  - Open archive data as single data object
  - Perform slicing through NetCDF library

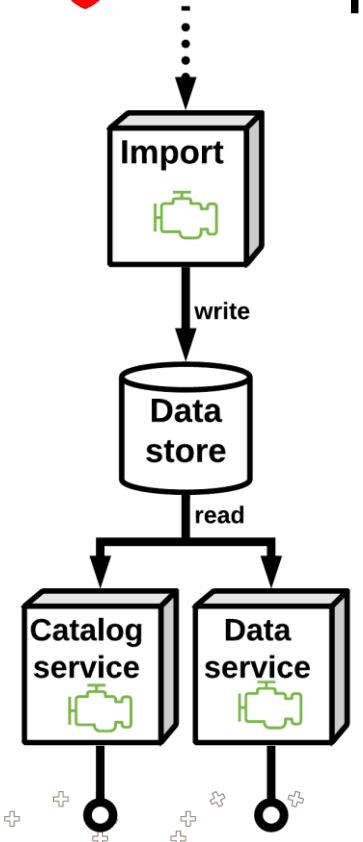
```
import xarray as xr
data = xr.open_dataset('http://gride.fix/api/sources/INCA/tags/nowcasting,surface')
data
```

xarray.Dataset

▸ Dimensions:	(**forecast_reference_time**: 82, t: 37, x: 631, y: 461)		
▾ Coordinates:			
forecast_refer…	(forecast_reference_time)	datetime64[ns]	2021-02-05T02:00:0…
time	(forecast_reference_time, t)	datetime64[ns]	…
longitude	(y, x)	float64	…
latitude	(y, x)	float64	…
▾ Data variables:			
rainfall_rate	(forecast_reference_time, t, y, x)	float32	…
▾ Attributes:			
bounding_box :	[3.0949137676025105, 44.44404860532198, 11.633508044478594, 48.655163205944284]		
crs :	EPSG:4326		
description :	MeteoSwiss Nowcasting system		
source :	INCA		
source_crs :	EPSG:4326		
tags :	["forecast", "1 kilometer", "nowcasting", "surface", "10 min timesteps", "operational", "Switzerland"]		

https://github.com/MeteoSwiss/opendap-protocol

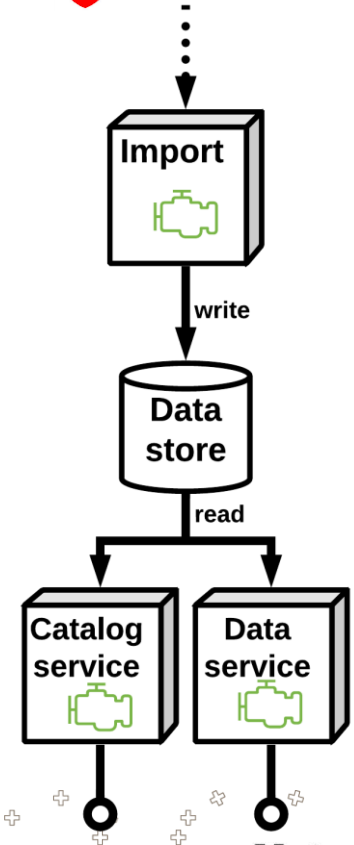
Import data: gridefix-ingest



- Independent entities for importing diverse data
- Data import usually developed by data provider
- Helper library supporting most common tasks
  - Coordinate manipulations
  - Transformation to target data model
  - Validate data model
  - Validate CF-compliance

MeteoSwiss

A word about deployment



PoC deployment on AWS

- EKS: API services (ingest, discover, retrieve)
- S3: object store
- DocumentDB: MongoDB compatible

On premise

- OpenShift cluster: API services + DB
- Minio: object store

MeteoSwiss

Outlook

Technical

- Complete deployment including event based part
- Consider other serialisation formats
  - WMS
  - Protocol Buffers
- Accomodate irregular grids

Non-technical

- Publication as OSS
- Use as general solution for gridded data at MeteoSwiss
- Deployment on EWC ☺