Semantically-driven data management solution for I/O intensive HPC workflows

(FOSDEM 2024, Brussels)

Metin Cakircali

Forecasts & Services Department, ECMWF, Bonn

metin.cakircali@ecmwf.int

Acknowledgement:
Metin Cakircali, Jenny Wong, Olivier Iffrig, Simon Smart, James Hawkes and Tiago Quintino

EuroHPC Joint Undertaking

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955811. The JU receives support from the European Union's Horizon 2020 research and innovation programme and France, the Czech Republic, Germany, Ireland, Sweden, and the United Kingdom.

ECMWF
European Centre for Medium-Range Weather Forecasts

IO-SEA

About ECMWF

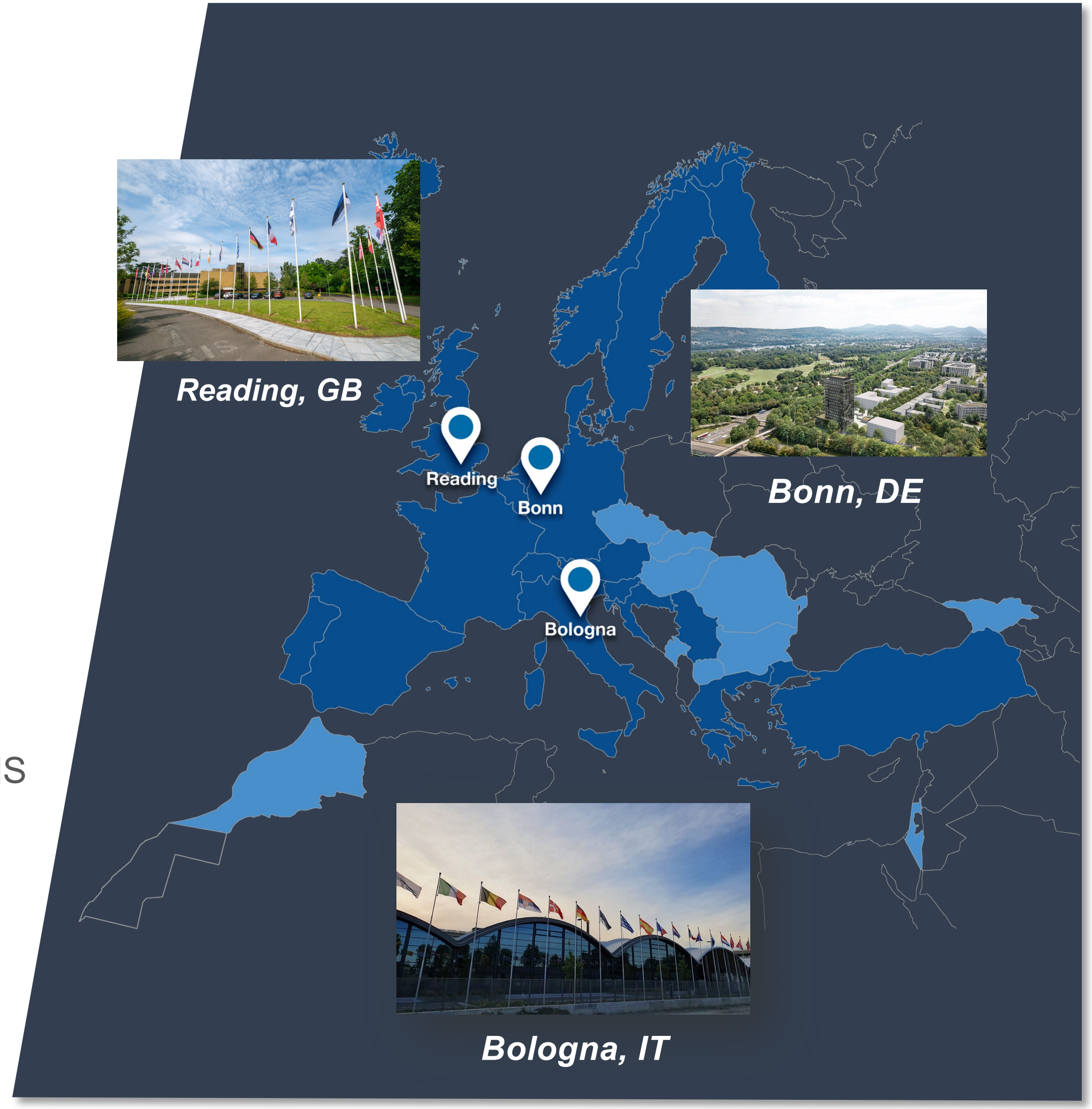
Established in 1975, Intergovernmental Organisation
- 23 Member States | 12 Cooperating States
- 450+ staff

24/7 operational service
- Operational NWP – 4x HRES+ENS forecasts / day
- Supporting NWS (coupled models) and businesses

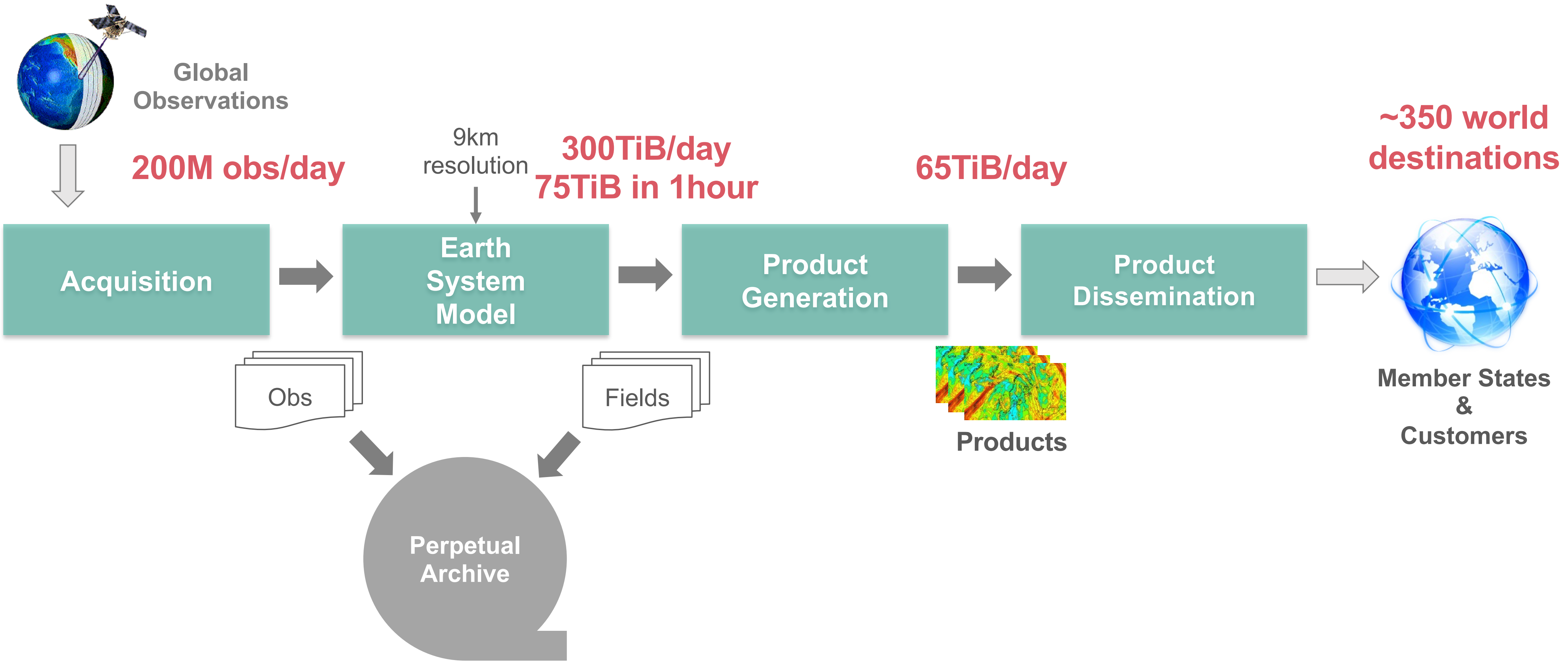
Research institution
- Experiments to continuously improve our models
- Reforecasts and Climate Reanalysis

Operate 2 EU Copernicus Services
- Climate Change Service (C3S)
- Atmosphere Monitoring Service (CAMS)
- Support Copernicus Emergency Management Service CEMS

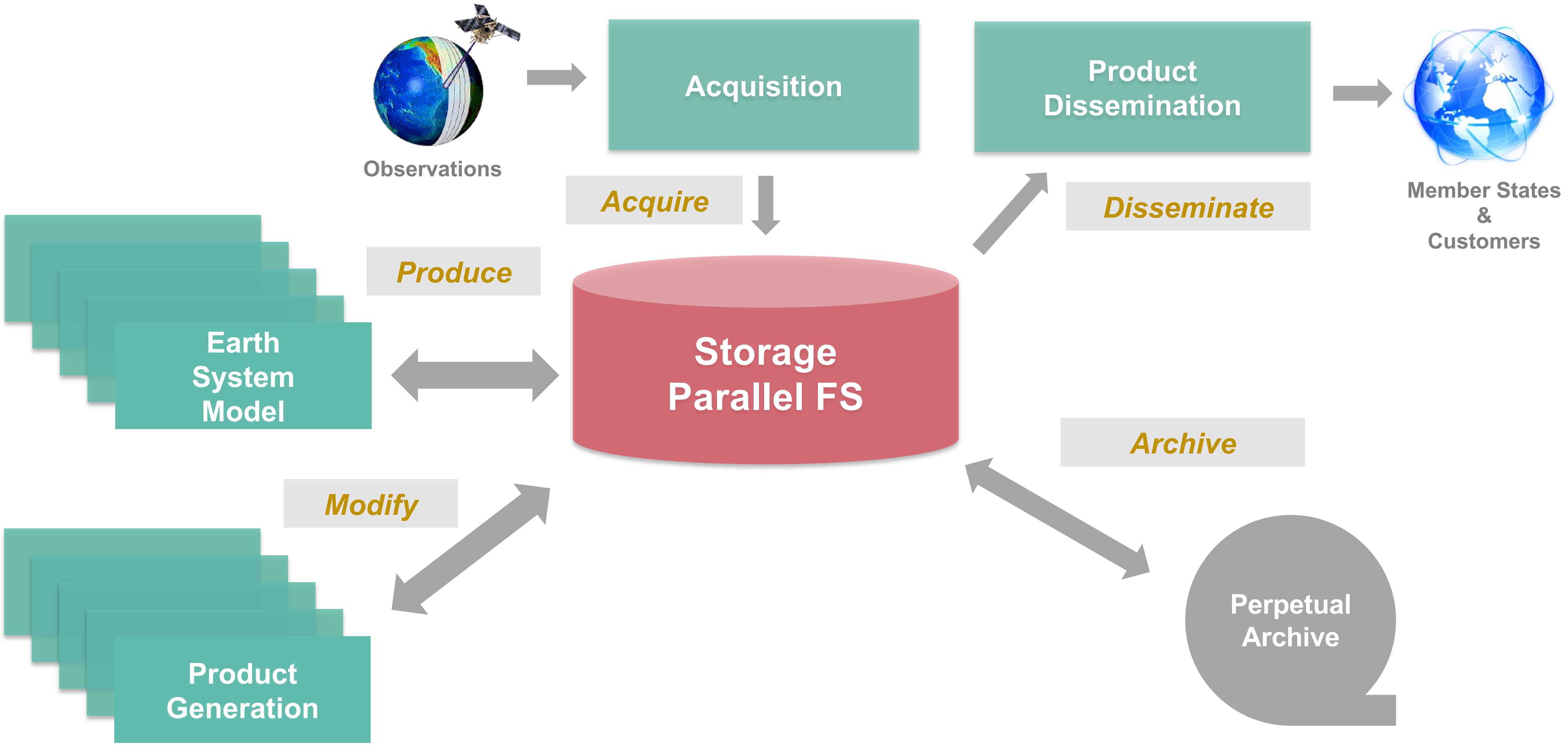
Destination Earth
- Operates two Digital Twins
- Operates the DestinE Digital Twin Engine (DTE)



ECMWF's Production Workflow



ECMWF’s Information System



Semantically-driven Data Management

Managing data based on its semantic (meaningful) description.

- abstract where/how data is stored

Avoid nested folder structures or UUIDs …

👎

```
/home/user/projects/ecmwf/42/20240203/…
```

👎

```
my_bucket/experiment-20240203-0000a211878e73ef26393003ea
```

… use meaningful metadata:

👍

```
project: ECMWF
experiment: 42
date: 20240203
parameter: Pressure
level: 0
```

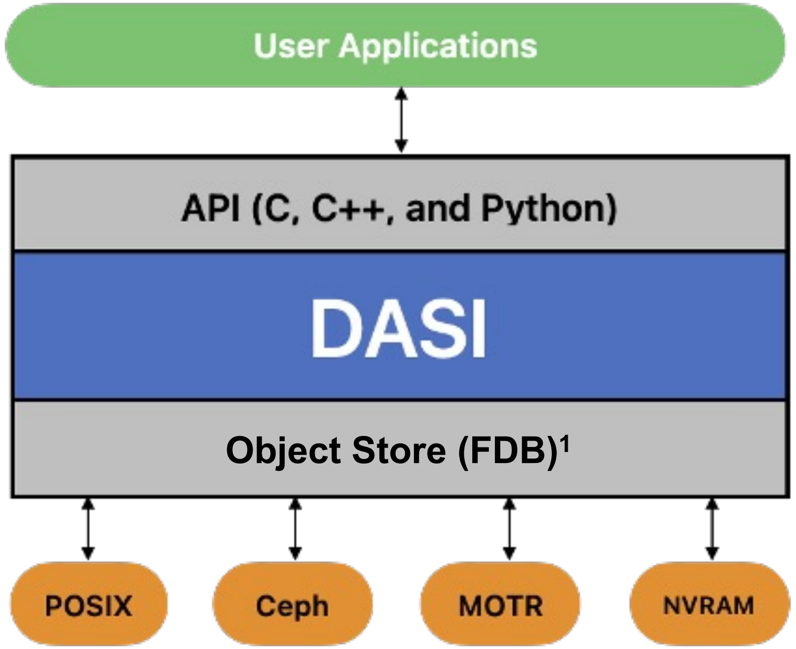
DASI (Data Access and Storage Interface)

- **semantically-driven data store**
  - index and identify using meaningful metadata
  - fast and efficient search/retrieve algorithms

- **abstracts storage technologies**
  - POSIX, DAOS, Motr, Ceph

- **based on the ECMWF's object store (FDB)[1]**
  - same data language since 1984 (>600PiB)

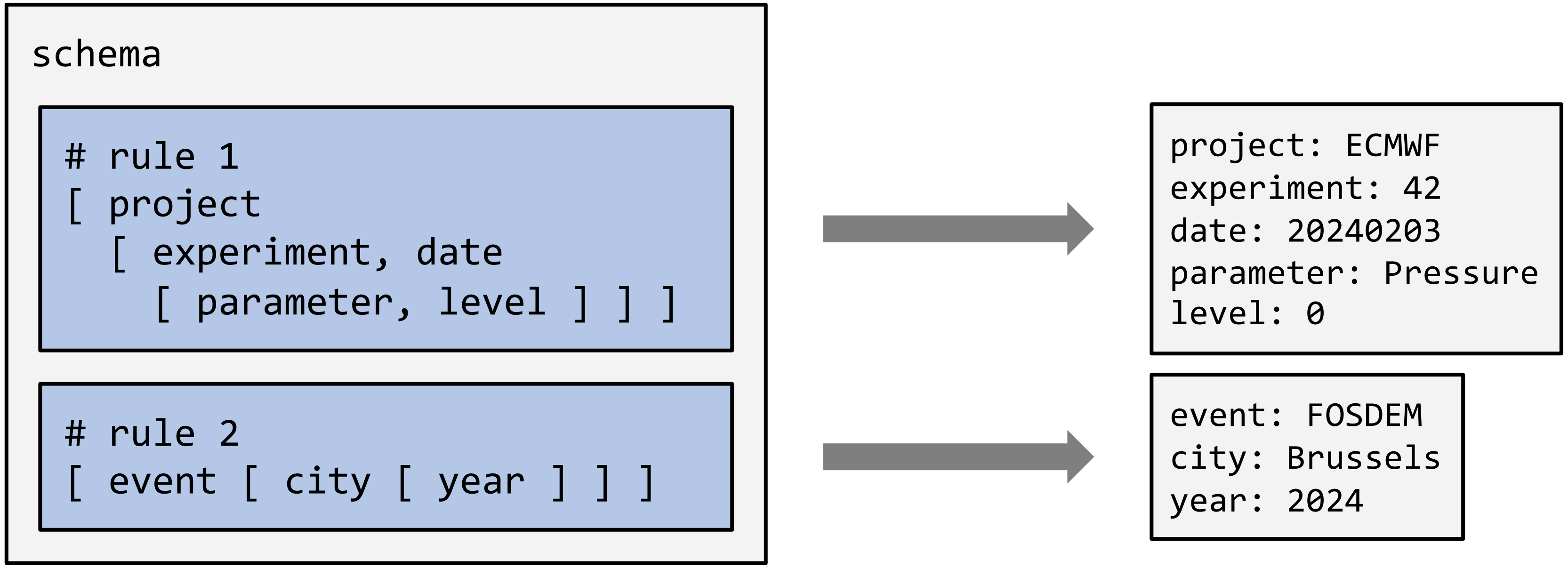
- **developed as part of IO-SEA project**
  - open source: github.com/ecmwf-projects/dasi



[1] Fields Database (FDB), https://github.com/ecmwf/fdb

DASI: Schema

- *Schema* is a collection of rules that describe the database structure
  - rule is a hierarchical tree of attributes



```
schema
```

```
# rule 1
[ project
  [ experiment, date
    [ parameter, level ] ] ]
```

```
# rule 2
[ event [ city [ year ] ] ]
```

```
project: ECMWF
experiment: 42
date: 20240203
parameter: Pressure
level: 0
```

```
event: FOSDEM
city: Brussels
year: 2024
```

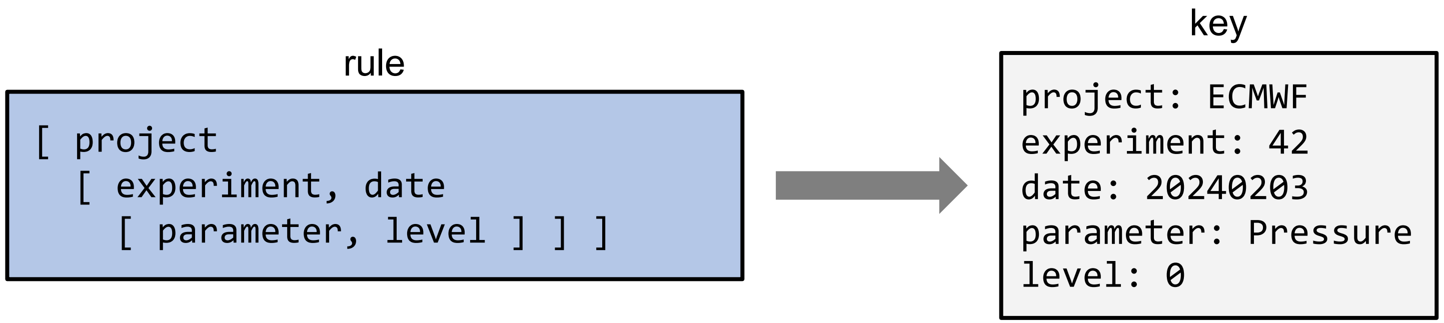
DASI: Rule

***Rule* is a hierarchical tree of attributes that …**

- has three levels
- can have multiple attributes per level

How to make a rule?

- unique and complete: how to identify data from others?
- locality: which data to store together?



rule

```
[ project
  [ experiment, date
    [ parameter, level ] ] ]
```

key

```
project: ECMWF
experiment: 42
date: 20240203
parameter: Pressure
level: 0
```

DASI: Configuration

config file

```
---
schema: /path/to/schema/file
store: file
spaces:
  - roots:
    - path: /path/to/data/output1
      retrieve: false
    - path: /path/to/data/output2
      wipe: true
```

- path to schema
- database paths (roots)
- backend storage technologies
- behavior; read, write, archive, retrieve, wipe

DASI: Key vs. Query

Key	Query
single object	any number of objects
project: ECMWF experiment: 42 date: 20240203 parameter: Pressure level: 0	project: ECMWF experiment: 42 date: 20240203 parameter: Pressure level: {0,1,3}

DASI: Usage

Command line tools and C, C++, and Python APIs are available to use.

archive: store data by a key

```
# Python API
data = b“some text”
key = { “User”: “metin”, “Project”: “IOSEA”, “Date”: “20231101”, “City”: “Bonn” }
dasi.archive(key, data)
```

list: search data by a query

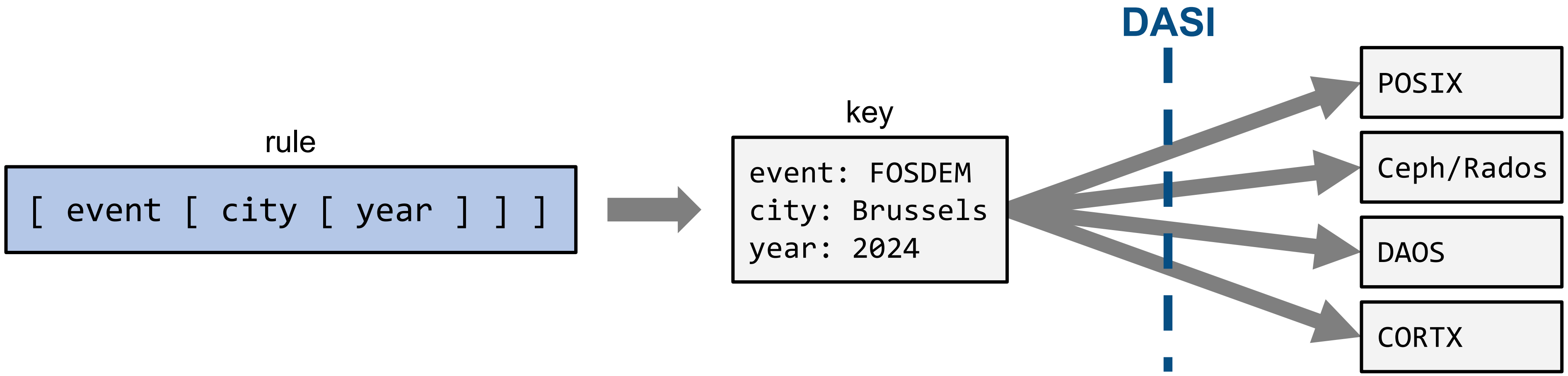
```
# Python API
query = { “User”: {“metin”}, “Project”: {“IOSEA”}, “Date”: {“20231101”, “20231102”}, “City”: {“Bonn”} }
keys = dasi.list(query)
```

retrieve: fetch data by a key

```
# Python API
key = { “User”: “metin”, “Project”: “IOSEA”, “Date”: “20231101”, “City”: “Bonn” }
data = dasi.retrieve(key)
```

Summary

- semantic (meaningful) description of data
  - no UUIDs, nested directories
- index and identify data
  - fast and efficient algorithms
- abstract where/how data is stored
  - transparent support for backend storage technologies



rule

```
[ event [ city [ year ] ] ]
```

key

```
event: FOSDEM
city: Brussels
year: 2024
```

DASI

POSIX

Ceph/Rados

DAOS

CORTX

More About DASI

- **Open-Source Code**
  - https://github.com/ecmwf-projects/dasi
  - Example: Histogram (Python API)
  - Example: Weather (C API)

- **Binary Packages**
  - https://github.com/ecmwf-projects/dasi/releases

- **Documentation**
  - https://dasi.readthedocs.io

Questions

Thank you for your attention…

Acknowledgement:
Metin Cakircali, Jenny Wong, Olivier Iffrig, Simon Smart, James Hawkes and Tiago Quintino

EuroHPC Joint Undertaking

This project has received funding from the European High-Performance Computing Joint Undertaking (JU) under grant agreement No 955811. The JU receives support from the European Union's Horizon 2020 research and innovation programme and France, the Czech Republic, Germany, Ireland, Sweden, and the United Kingdom.