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

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.
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

Global Observations

200M obs/day

9km resolution

300 TiB/day

75 TiB in 1 hour

65 TiB/day

~350 world destinations

Member States & Customers

Acquisition

Earth System Model

Product Generation

Product Dissemination

Perpetual Archive

Obs

Fields

Products

200M observations per day are acquired, processed, and distributed globally through a network of member states and customers. The data processing involves Earth System Models generating products at a rate of 300 TiB per day, with 75 TiB processed in one hour. The final products are disseminated to approximately 350 destinations worldwide.
ECMWF’s Information System

- **Observations**
- **Acquisitions**
- **Product Dissemination**
- **Member States & Customers**

- **Acquire**
- **Disseminate**
- **Archive**

- **Storage Parallel FS**
- **Earth System Model**
- **Product Generation**
- **Modify**
- **Perpetual Archive**
Semantically-driven Data Management

Managing data based on its semantic (meaningful) description.
• abstract where/how data is stored

Avoid nested folder structures or UUIDs …

... 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
**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?

![Diagram of a rule key]
DASI: Configuration

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

```yaml
---
schema: /path/to/schema/file
store: file
spaces:
  - roots:
    - path: /path/to/data/output1
      retrieve: false
    - path: /path/to/data/output2
      wipe: true
```
### DASI: Key vs. Query

<table>
<thead>
<tr>
<th>Key</th>
<th>Query</th>
</tr>
</thead>
<tbody>
<tr>
<td>single object</td>
<td>any number of objects</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key Details</th>
<th>Query Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>project: ECMWF</td>
<td>project: ECMWF</td>
</tr>
<tr>
<td>experiment: 42</td>
<td>experiment: 42</td>
</tr>
<tr>
<td>date: 20240203</td>
<td>date: 20240203</td>
</tr>
<tr>
<td>parameter: Pressure</td>
<td>parameter: Pressure</td>
</tr>
<tr>
<td>level: 0</td>
<td>level: {0,1,3}</td>
</tr>
</tbody>
</table>
DASI: Usage

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

**archive**: store data by a key

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

**retrieve**: fetch data by a key

```python
# 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
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

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.