# A Scalable Object Store for Meteorological and Climate Data

Simon Smart, Tiago Quintino, Baudouin Raoult

ECMWF

simon.smart@ecmwf.int

# European Centre for Medium Range Weather Forecasts (ECMWF)

## What do we do?

Operational forecasts – **Time Critical**

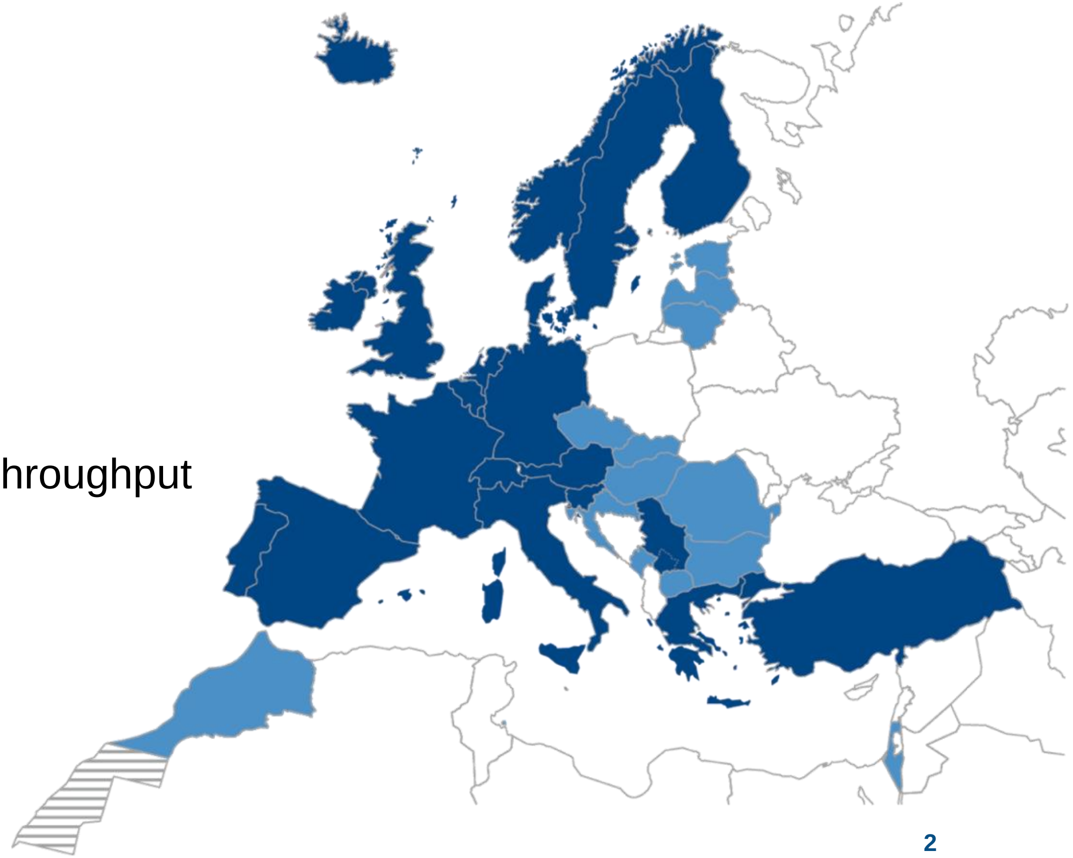
- 2 hours from satellite cut-off to deliver forecast products
- Twice per day, 00Z and 12Z

Research – **Non Time Critical**

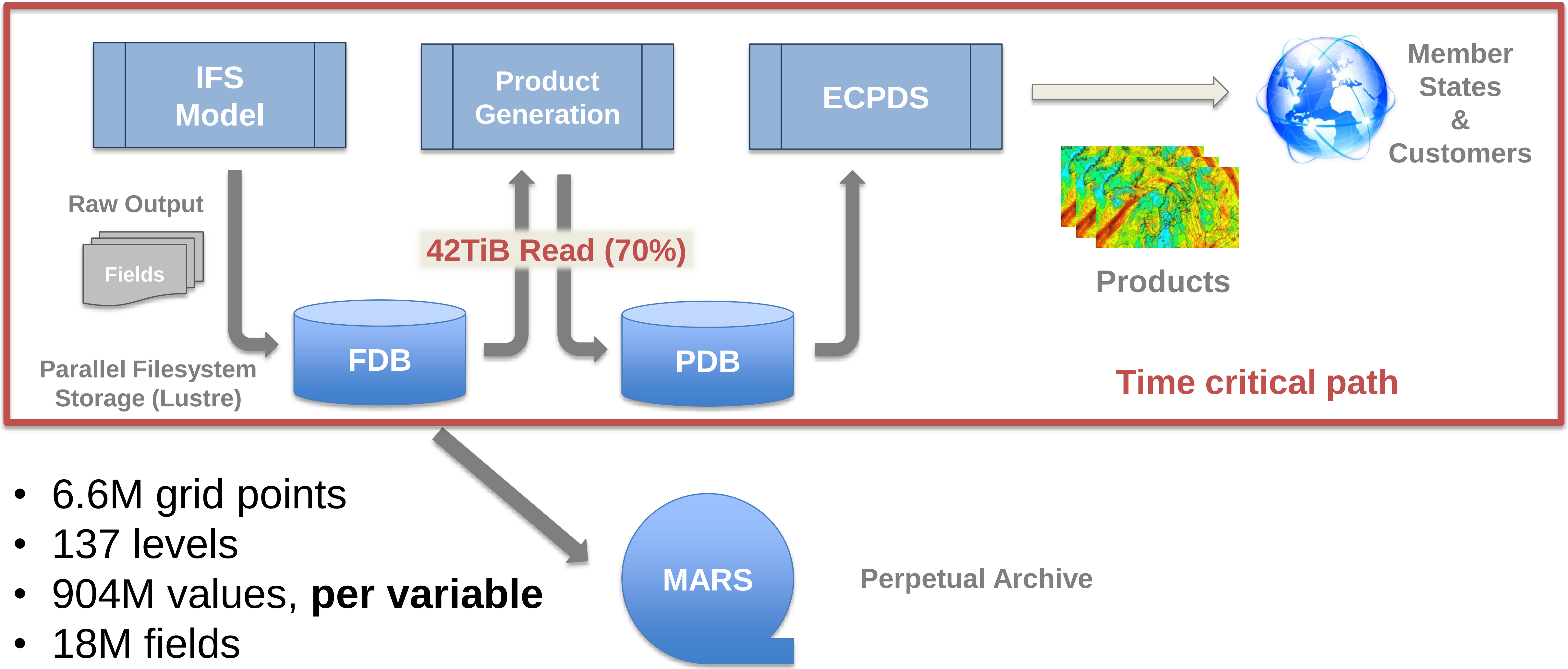
- Large part of the workload
- Re-uses current and historic analysis and forecast data

## Central trade-off

- **Performance**, minimise the time to solution, maximise throughput
- **Reliability**, minimise the worst-case runtime



# ECMWF’s Production Workflow



- 6.6M grid points
- 137 levels
- 904M values, **per variable**
- 18M fields

# Estimated Growth in Model IO

**2015**

**16km, 137 levels**

### Time critical

- 21 TB/day written
- 22 Million fields
- 85 Million products
- 11 TB/day send to customers

### Non-time critical

- 100 TB/day archived
- 400 research experiments
- 400,000 jobs / day

**2020**

**Increase: 2 horizontal, 1 upper air**

### Time critical

- 128 TB/day written
- 90 Million fields
- 450 Million products
- 60 TB/day send to customers

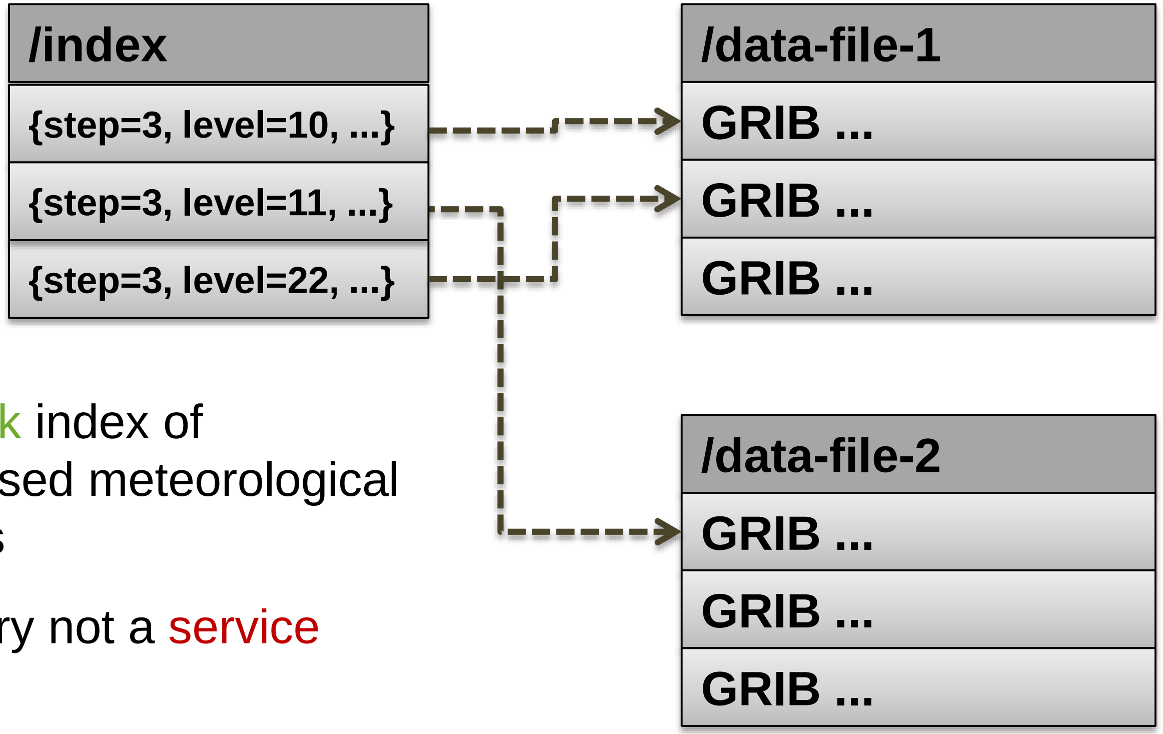
### Non-time critical

- 1 PB/day archived
- 1000 research experiments

# What is the FDB Today?

/rootpath/ database-dir/

Carefully named directories of related data



/index

{step=3, level=10, ...}

{step=3, level=11, ...}

{step=3, level=22, ...}

/data-file-1

GRIB ...

GRIB ...

GRIB ...

/data-file-2

GRIB ...

GRIB ...

GRIB ...

Each process writes to independent data files.

(MPI) Synchronisation required for writing to the index file.

- An in disk index of compressed meteorological data files
- Is a library not a service

# What are the issues with todays FDB?

- **Consistency** must be explicitly managed in parallel
- Not **transactional**. Poor behavior on failure.
- Poor traceability

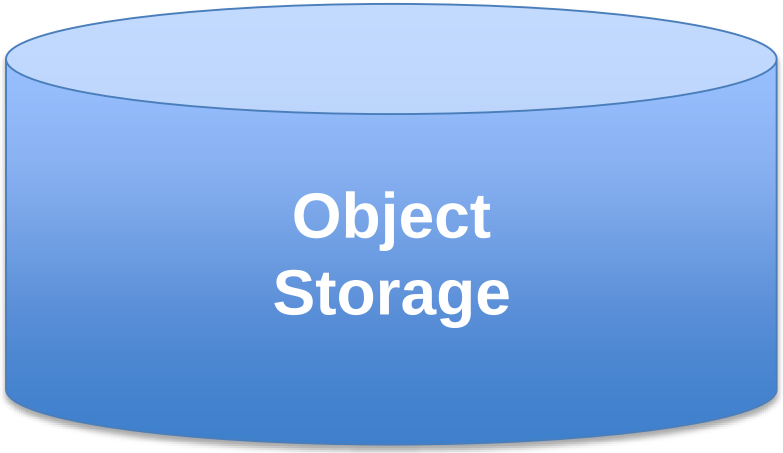
Trading consistency for performance incurs a human cost.

# Object Store

- Key-Value stores offer **scalability**
  - Just add more instances to increase capacity and throughput
- **Transactional** behaviour with minimal synchronization
- Growing popularity, namely due to **Big Data Analytics**

**Key: date=12012007, param=temp**

**Value: 101001...100101010110010**



**Object Storage**

***But ECMWF has been using key-value store for 30 years...***

**MARS**

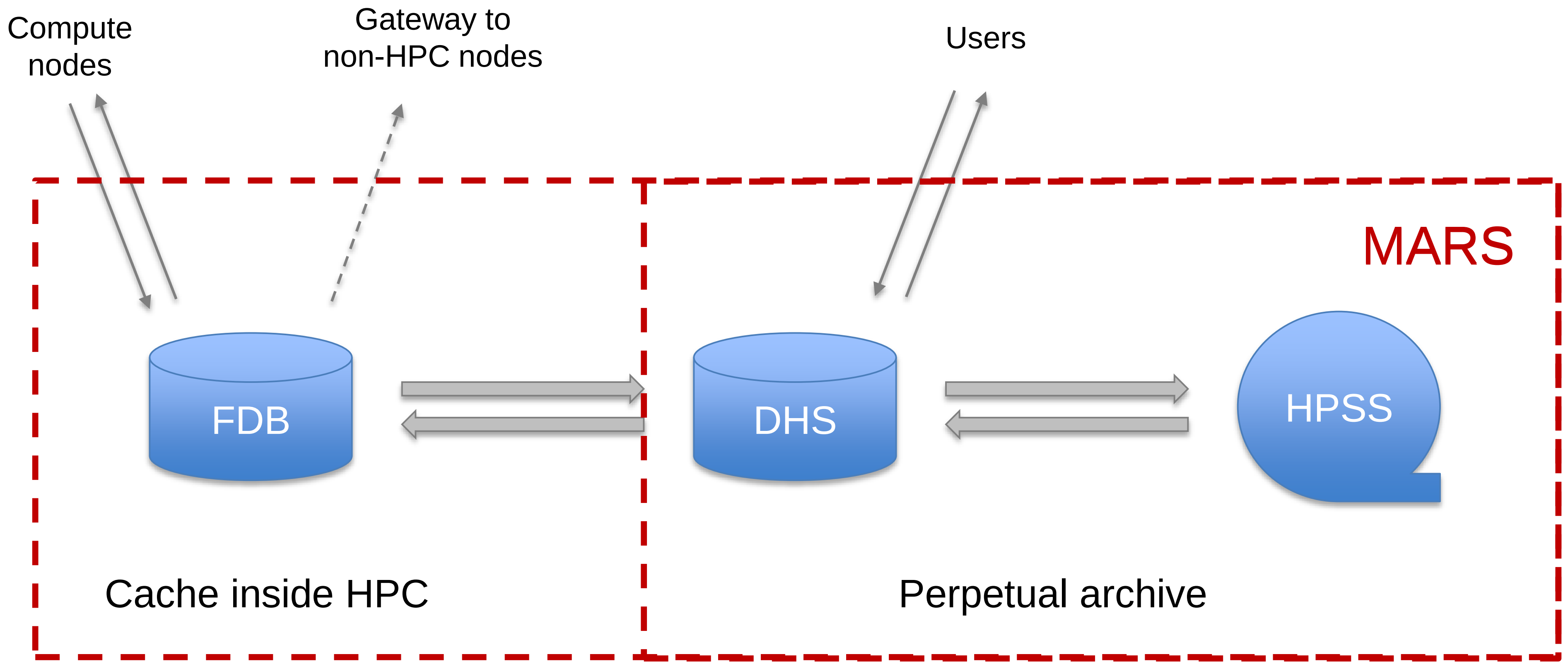
# MARS Language

```
RETRIEVE,
   CLASS    = OD,
   TYPE     = FC,
   LEVTYPE  = PL,
   EXPVER   = 0001,
   STREAM   = OPER,
   PARAM    = Z/T,
   TIME     = 1200,
   LEVELIST = 1000/500,
   DATE     = 20160517,
   STEP     = 12/24/36
```

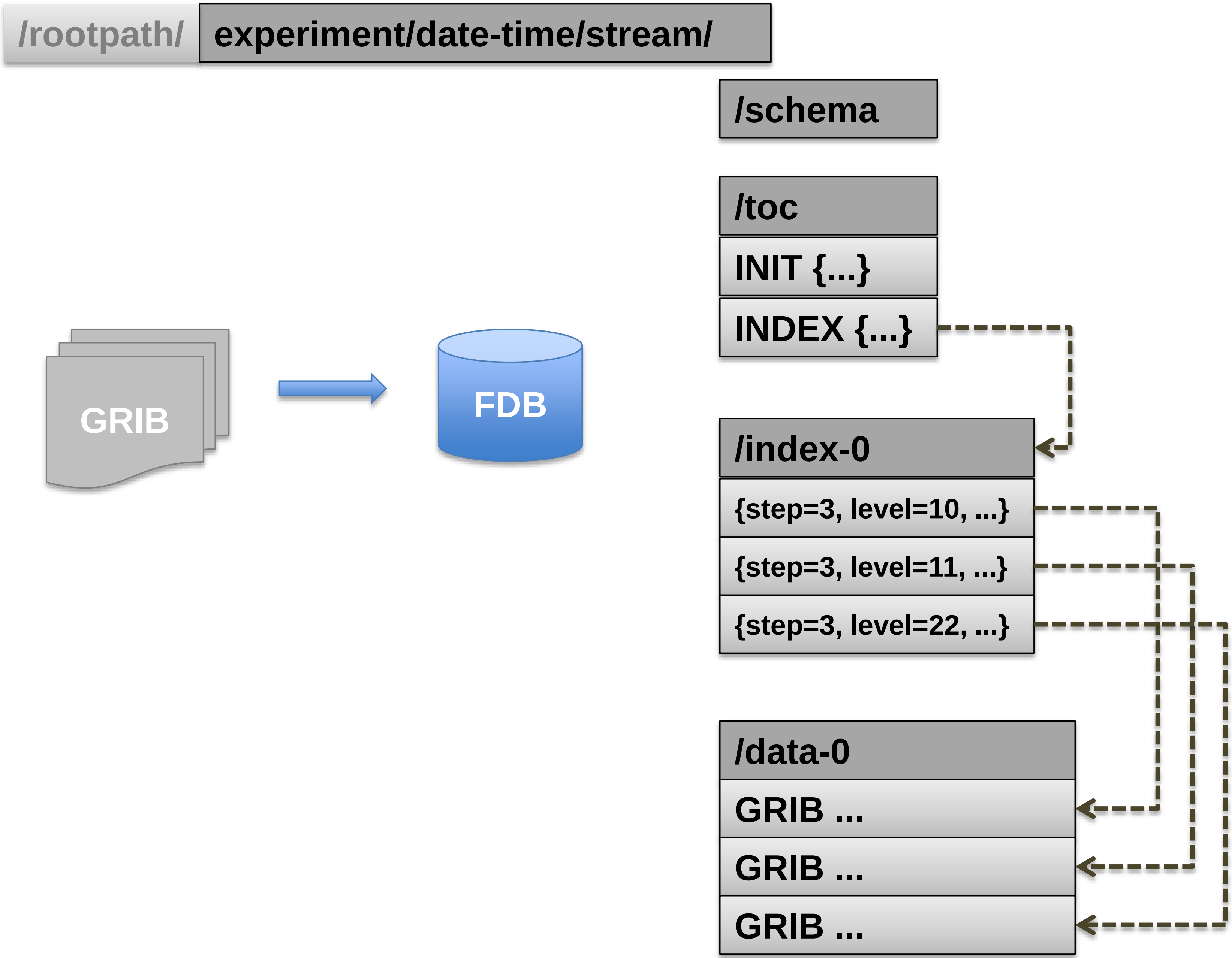
```
RETRIEVE,
   CLASS    = RD,
   TYPE     = FC,
   LEVTYPE  = PL,
   EXPVER   = ABCD,
   STREAM   = OPER,
   PARAM    = Z/T,
   TIME     = 1200,
   LEVELIST = 1000/500,
   DATE     = 20160517,
   STEP     = 12/24/36
```

**Unique** and **semantic** way to describe all ECMWF data

# What is the relationship between the FDB and MARS?



**Application controlled HSM with archive**



/rootpath/ experiment/date-time/stream/

GRIB

FDB

/schema

/toc

INIT {...}

INDEX {...}

/index-0

{step=3, level=10, ...}

{step=3, level=11, ...}

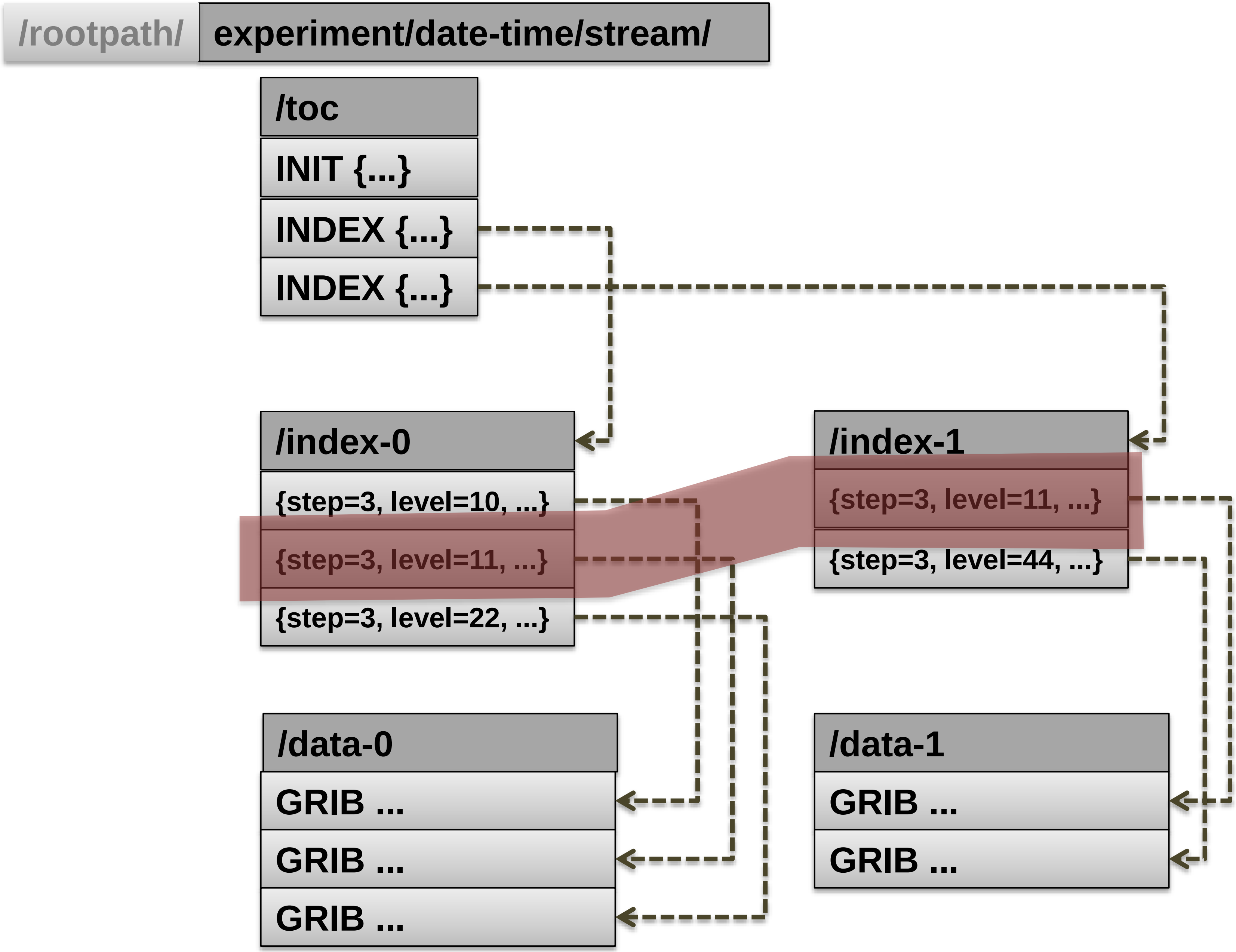
{step=3, level=22, ...}

/data-0

GRIB ...

GRIB ...

GRIB ...



/rootpath/ experiment/date-time/stream/

/toc

INIT {...}

INDEX {...}

INDEX {...}

/index-0

{step=3, level=10, ...}

{step=3, level=11, ...}

{step=3, level=22, ...}

/index-1

{step=3, level=11, ...}

{step=3, level=44, ...}

/data-0

GRIB ...

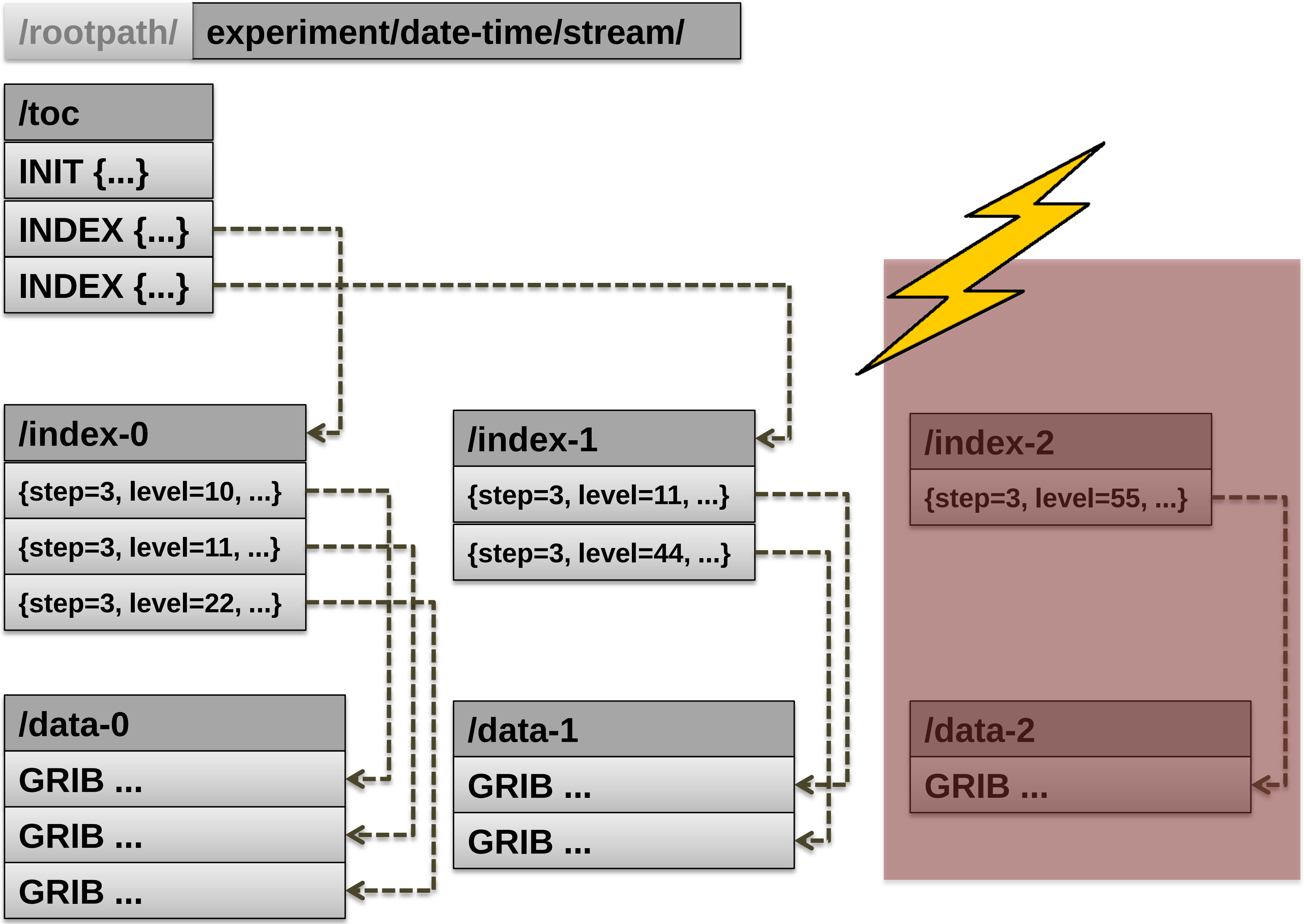
GRIB ...

GRIB ...

/data-1

GRIB ...

GRIB ...



/rootpath/ experiment/date-time/stream/

/toc

INIT {...}

INDEX {...}

INDEX {...}

/index-0

{step=3, level=10, ...}

{step=3, level=11, ...}

{step=3, level=22, ...}

/index-1

{step=3, level=11, ...}

{step=3, level=44, ...}

/index-2

{step=3, level=55, ...}

/data-0

GRIB ...

GRIB ...

GRIB ...

/data-1

GRIB ...

GRIB ...

/data-2

GRIB ...

# Schemas, appending and write performance

1. Segregate data into logically independent (re-runnable) units

```
# Ensemble forecast
[ class, expver, stream=enfo/efov, date, time, domain, number?
       [ type, levtype
               [ step, quantile?, levelist?, param ]]]

# High-resolution forecast
[ class, expver, stream=oper/dcda/scda, date, time, domain
       [ type, levtype
               [ step, levelist?, param ]]]
```

Empirically: 11 appends per second*

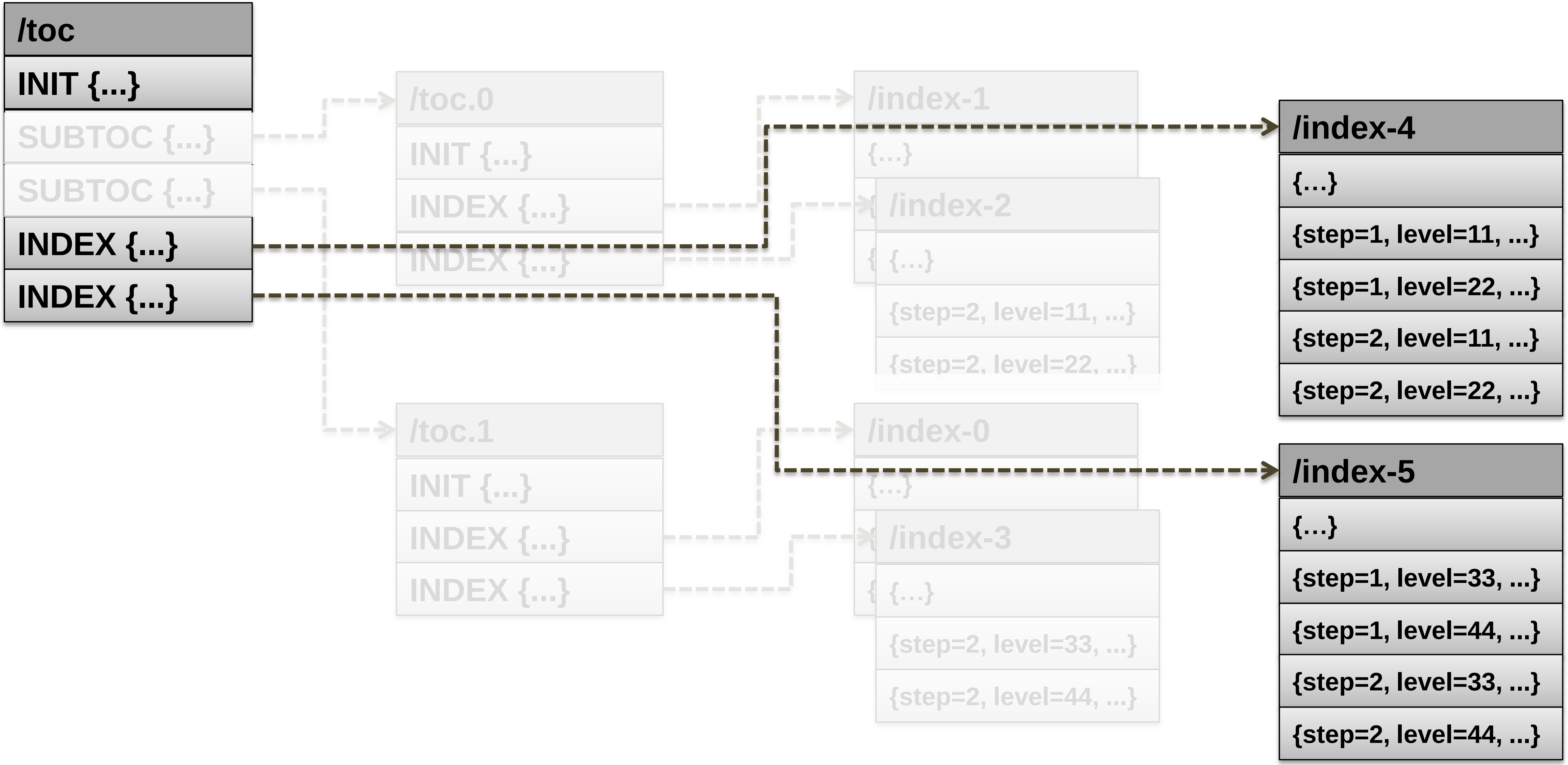
52 forecasts, 10 writers, 240 steps

→ *Minimum* 35 / second

2. Cheat …

* From 8 competing nodes, on operational Lustre filesystems at ECMWF

/rootpath/ experiment/date-time/stream/



/toc

INIT {...}

SUBTOC {...}

SUBTOC {...}

INDEX {...}

INDEX {...}

/toc.0

INIT {...}

INDEX {...}

INDEX {...}

/toc.1

INIT {...}

INDEX {...}

INDEX {...}

/index-1

{...}

/index-2

{...}

{step=2, level=11, ...}

{step=2, level=22, ...}

/index-0

{...}

/index-3

{...}

{step=2, level=33, ...}

{step=2, level=44, ...}

/index-4

{...}

{step=1, level=11, ...}

{step=1, level=22, ...}

{step=2, level=11, ...}

{step=2, level=22, ...}

/index-5

{...}

{step=1, level=33, ...}

{step=1, level=44, ...}

{step=2, level=33, ...}

{step=2, level=44, ...}

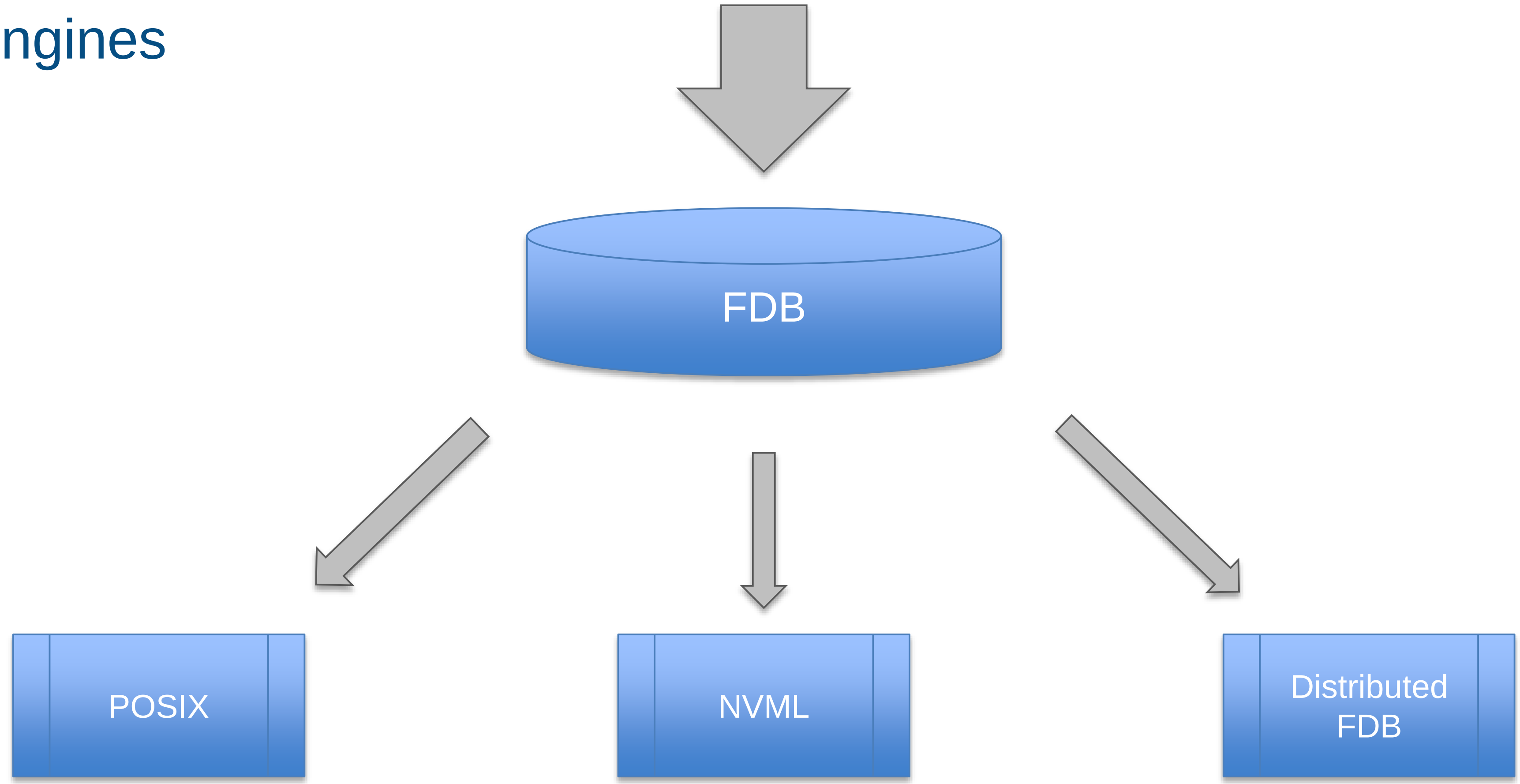
# Reading from the FDB5?

- Data transfer is similar to existing FDB. **Like for like replacement**
  - Limited by bulk movement of data from disk.

| No. Fields | Current FDB (fastest / slowest) | | FDB5 (fastest / slowest) | |
|---|---|---|---|---|
| 1 | 0.42 | 0.61 | 0.28 | 3.34 |
| 10 | 0.75 | 8.90 | 0.8 | 4.07 |
| 100 | 2.57 | 20.61 | 1.28 | 30.34 |
| 1000 | 23.53 | 533.97 | 6.95 | 418.59 |
| 10000 | 1989.38 | 2,053.75 | 2,289.28 | 2,469.92 |
| 100000 | > 170,000 (max walltime) | | 33,846 | 51,777.49 |

- Performance worse until sub-TOCs are masked.
- Sensitive to filesystem caching.
- For real requests, generally hardware bound

# FDB5 engines



Lustre
GPFS
…

Move away from files.
Avoid global consistency requirements.

**New opportunities to adapt data workflows**

# New memory technologies are coming soon

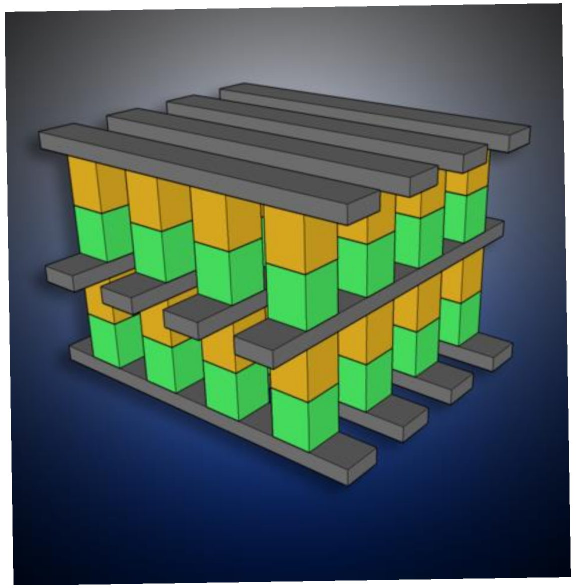
**3D XPoint** is coming soon

- Storage density similar to NAND flash memory
- Better durability than flash
- Speed and latency between NAND and DRAM
- Priced between NAND and DRAM

***Source: https://en.wikipedia.org/wiki/3D_XPoint***

Many questions

- How much will be affordable
- Likely not on every node
  - How do we distribute it, and access it remotely



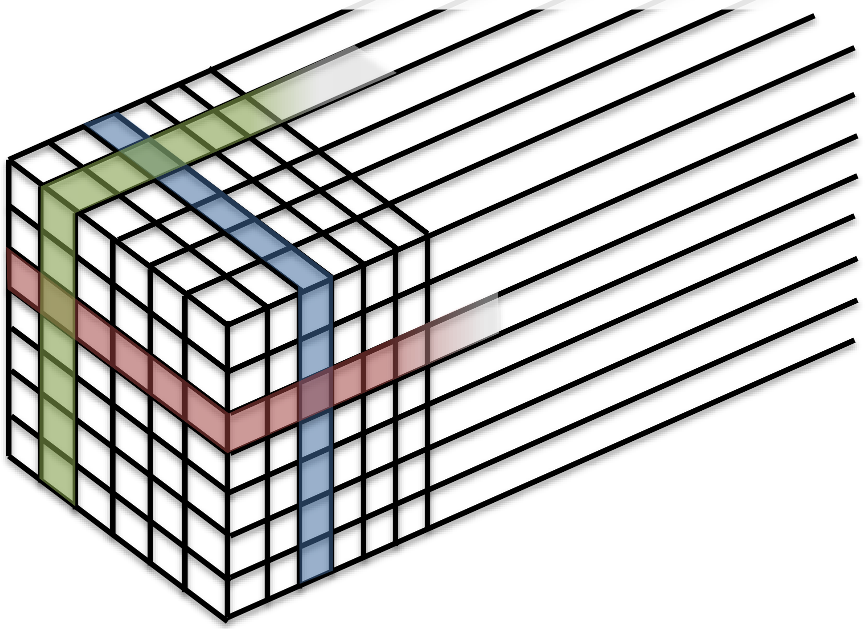
"3D XPoint" by Trolomite
Own work. Licensed under CC BY-SA 4.0

# Storing and accessing dense meteorological data

**Byte Addressable Hypercubes**

- Longitude (3600)
- Latitude (1800)
- Atmospheric levels, Physical parameters (~200)
- Time steps (~100)
- Probabilistic perturbations (50)

Clients want to do **different** analytics across **multiple** axis



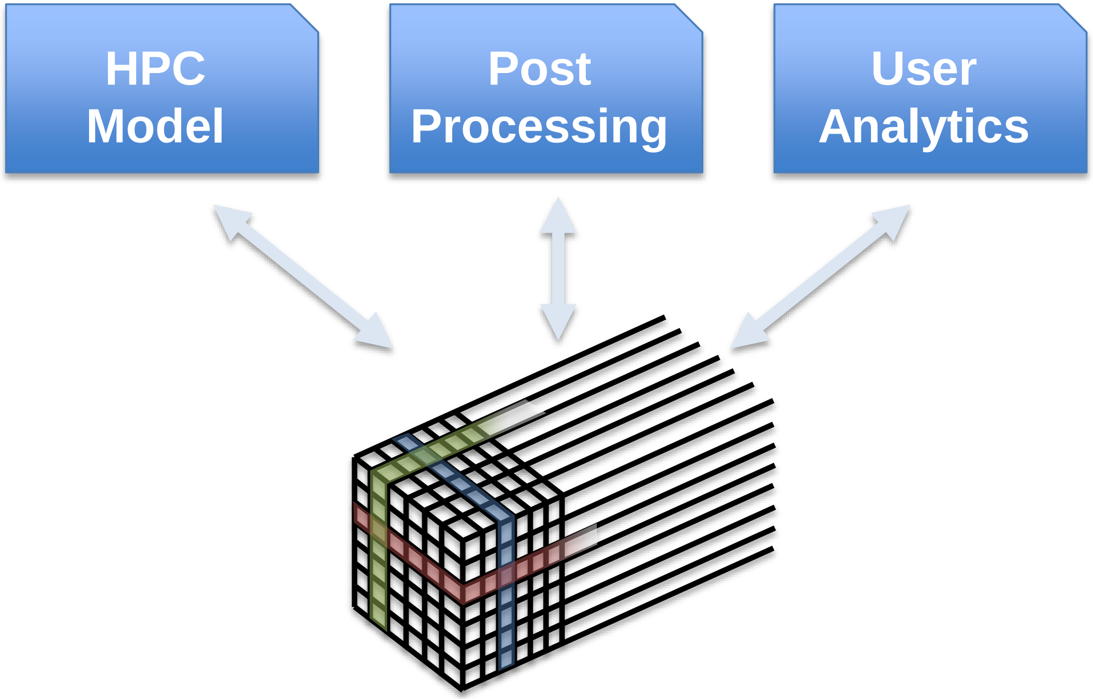
**@ double precision**

- 9km **48 TiB**
- 5km **192 TiB**
- 1.25km **1.82 PiB**

**Not** included: *historical observations, multiple models, etc...*

# Data Centric Computing

- **Producer-Consumer** model, where *HPC is producer*
- Use data while is **hot**
- Bring **users** to the data, ship ***functions***
- Don't use **files,** use **science to communicate**, use **rich metadata**
- Need to **build shared components** amongst the communities...



# ECMWF

***Part of ECMWF’s Scalability Programme***

- **Large buffers** for **time critical** applications
  - Can store **entire model output** in “memory”
  - Similar to *burst buffers* but in application space

- ***Persistence*** *until archival, for* ***non time critical***
  - *adding a new layer in the hierarchical storage system view*

- ***New workloads***
  - *Bring computation to the data for in-situ analytics.*

**Partners**

- EPCC (Proj. Leader)
- Intel
- Fujitsu
- T.U. Dresden
- Barcelona S.C.
- Allinea Software
- ARCTUR
- ECMWF

http://www.nextgenio.eu - EU funded H2020 project, runs 2015-2018

# Messages to take home

FDB5 future-proofs our storage stack

- **Consistent**, **transactional** parallel access
- Hardware limited performance

**Multiple engines** permit new usage patterns, and new ways of working

- Data pipelines
- In-situ data processing

**NVRAM** is coming

- We/you need to adapt

*NEXTGenIO has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 671951*