Data Systems @DKRZ: Tech Tussle

or:
The start of an
improved collaborative process to
ease data handling for
DRKZ users

22 Feb 2023, support@dkrz.de

DKRZ

Goal - why are we here today?

ESM output data is becoming so voluminous and complex that its organisation and analysis has become extremely difficult

This is why today, we'd like to

➔ **Establish a process** which enables **organized exchange and collaboration** between **DKRZ and DKRZ users** to develop and provide a more suitable **service portfolio** meeting the **data handling requirements** of DKRZ users.

What will happen now…

Setting the scene

-> team, boundary conditions, possible concept

Tech Tussle -> Brainstorming for constructive exchange

-> given the boundary conditions of our infrastructure

Who we are (at the moment)

- Anna Fuchs (storage)
- Florian Ziemen (visualization, workflow support for nextGEMS, ESiWACE2)
- Panos Adamidis (ICON I/O, HPC)
- Martin Bergemann (freva)
- Daniel Heydebreck (HSM expert/support)
- Fabian Wachsmann (data processing and data distribution)
- Karsten Peters-von Gehlen (a little bit of everything, connecting the pieces)
- Sven Willner and team (MPI-M, semantic data storage)

With support of various other colleagues with specific expertise in data handling aspects (hardware, analysis, metadata, catalogs)

Books may indeed be quite useful - but remember the saying:

Ignorance is the mother of all adventures

Time to get to know you!

What is your biggest problem in your daily data handling workflow?

Enter one keyword, please! **this-is-a-long-keyword** is also possible… ;-)

Participants can vote at
Slido.com with
#4256749
or
https://rb.gy/4yyii9

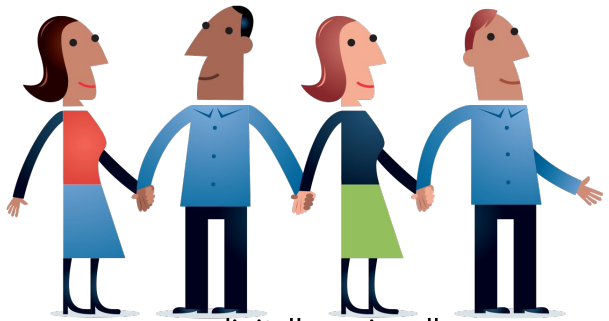
Our impression so far…

Workflow problem involves

- Data selection
- Multiple Tiers of Storage (fast, slow, very slow)
- Multiple Types of Storage (tape, object, posix etc)
- Performance is an issue.
- Compression is needed.
- Lots of technology available, or potentially available, but hard to harness and/or not clearly useful.
- **Balance of “difficulty” tilting away from “simulation hard, analysis easy” to “simulation hard, analysis just as hard”.**

After B Lawrence

We feel you and that’s why we’re here today!



Boundary conditions defined by available infrastructure components

Levante HPC

Specs:

2832 CPU Nodes
100 Gbit/s per node
(compute, shared, interactive)

60 GPU Nodes (GPU)

No storage on nodes.

Rank 53 of the TOP500-List (11/2022)

Storage systems

Property	File System	Object Storage *	HSM
Visible as	/ … on levante (normal File system)	https://swiftbrowser.dkrz.de *	via slk command line tool on levante.
Capacity	130 PB	1 PB *	~ 240 PB (new library) ~ 2 PB disk cache.
Time to first byte			

We need the tape system because it is bigger than the disk storages.

* successor:
object storage (S3)
using disk storage
from Mistral planned

Tape has high latencies. Most of the **working data** must be on the file system since it is much bigger than the other disk storage systems.

Parallel file system (*Lustre* /work/ /home/ /…)

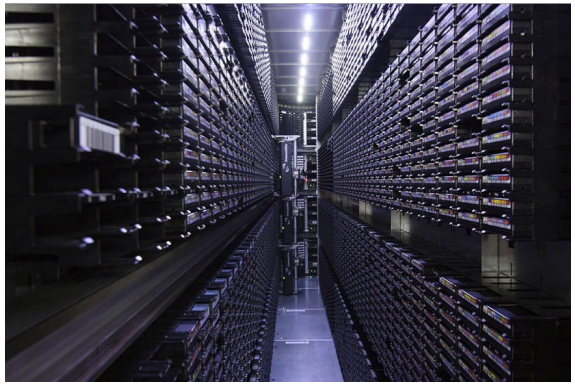
- Strength:
  - Fast access to large amounts of data
  - Parallel read/write of files
- Used for most of the data storage
  - “luxurious” conditions compared to other HPC systems
  - a lot of *old* data has not been read in years
- Every file is distributed across many disks (HW RAID + Lustre FS)

Object storage (*swift*)

- Strength:
  - Anonymous access from outside of DKRZ possible
- Used for:
  - Sharing files with outside users
  - Hosting Cloud Optimized Data (zarr) for e.g. easy reduction of data transfers
  - Gitlab LFS extension
- Every file is distributed across 6 disks per (5GB) chunk
- **Bigger S3-interfaced successor using old Mistral disks is planned**

Tape storage (*HSM / slk / Stronglink*)

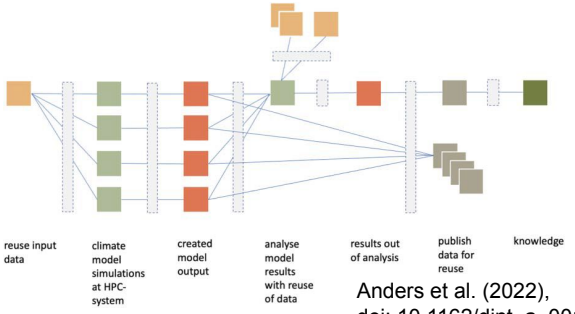
- Strength
  - Minimal energy usage and cost
  - automated harvesting of netCDF metadata
- Used for
  - Archival of simulations after analysis
  - Archival of restart files
  - Long-term storage of data
  - A few minutes latency for fetching a tape and forwarding to the read location
- ~300MB/s read speed per tape
- New library scheduled to enter operation in April



Requesting small amounts of data is inefficient.
Multi-TB files take long to read.

Work in progress

current workflows (non-exclusive)



Anders et al. (2022),
doi: 10.1162/dint_a_00127

- user performs model run on Levante,
  analyzes the data at DKRZ and stores the output data at DKRZ
- data is stored in the DKRZ tape archive;
  user wishes to retrieve and copy them to an external location
- user performs a simulation at another HPC site and copies the data to DKRZ for further analysis and long term storage (tape)
- data is stored at DKRZ (Lustre or tape archive);
  a user wishes to analyze the data

What would be of biggest value a data handling system could give to you?

Pick 4 of the provided options

Participants can vote at
Slido.com with
#4256749

Plan for the concept I (VERY high-level)

Data, produced by models, are stored at DKRZ in different storage locations and are indexed in centrally maintained catatalogue(s)

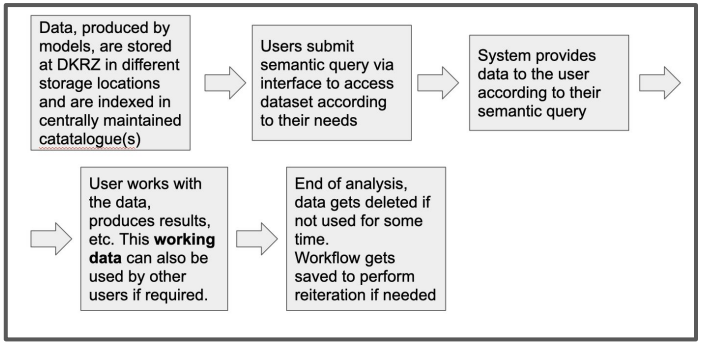
Users submit semantic query via interface to access dataset according to their needs

System provides data to the user according to their semantic query

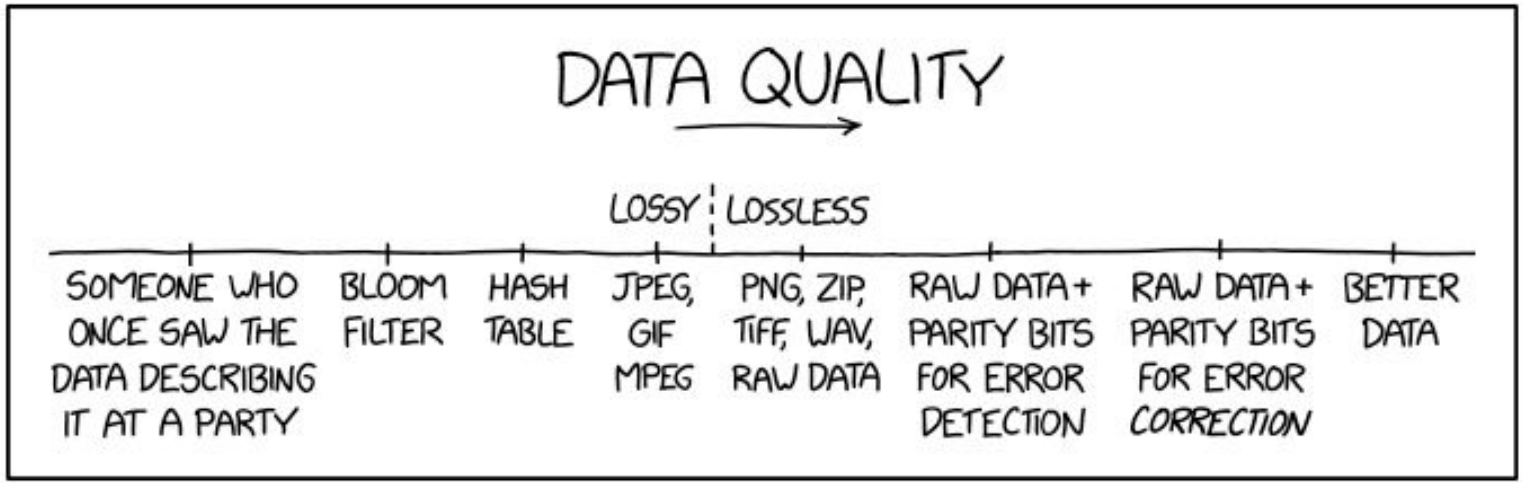
User works with the data, produces results, etc. This **working data** can also be used by other users if required.

End of analysis, data gets deleted if not used for some time.
Workflow gets saved to perform reiteration if needed

Plan for the concept II



- centrally maintained and administered
- catalogued data requires that metadata follow at least system-wide requirements
  - support will be available
- system use should not be mandatory, but support for people not using the system will not be provided to the same extent as if the system was used
- future data handling will/has to involve a shift in the way we work
  - -> cultural / social aspect which cannot be ignored

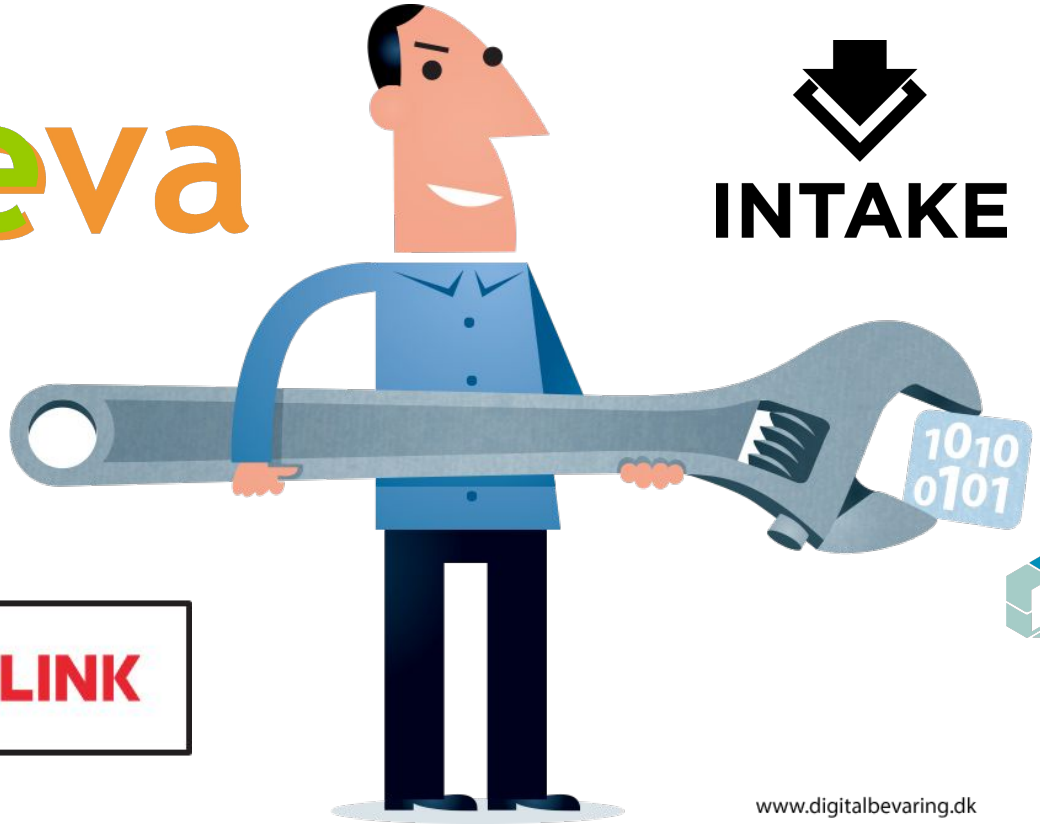


https://xkcd.com/2739/

What do we have? What is in progress?

by freva

INTAKE



Zarr

STRONG LINK

ACROSS
HPC Big DAta ArtifiCial intelligence cross Stack PlatfoRm TOwardS ExaScale

BORGES

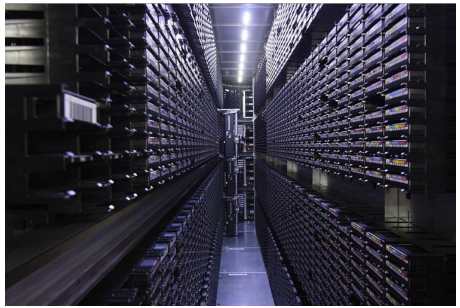
www.digitalbevaring.dk

How familiar are you with current efforts to improve data handling in the DKRZ environment?

Pick 1 of the provided options

Participants can vote at
Slido.com with
#4256749

HSM / StrongLink



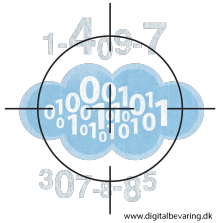
metadata feature

- netCDF metadata automatically harvested
  - standard_name + long_name + name of variable
  - time variable: first and last value
  - ≅100 global attributes
- manual metadata updates possible
- manual Grib2 metadata import tool in preparation
- search files based on metadata

details: https://docs.dkrz.de/doc/datastorage/hsm/metadata.html

questions: support@dkrz.de

Semantic data access



write

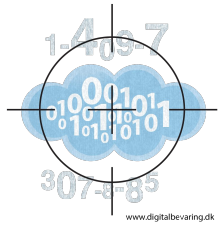
load_data(variable=’tas’, simulation=’ngc2009’, time=’2020-02-22’)

instead of

load_data(‘/work/bm1235/k203123/experiments/ngc2009/run_20200219T000000-20200304T235920/ngc2009_atm_2d_30min_inst_20200222T000000Z.nc’)

needs a logic that identifies data according to meaningful properties, usually some form of table/database.

Intake(-ESM)



Intake(-ESM) uses a catalog file of dataset properties and identifiers; at least 5 large catalogs of climate model simulation output are maintained at DKRZ.

As identifier, anything works that python can be taught to load data from.

e.g.

variable_id=ta, simulation_id=ngc2013, time_min=2020-02-.*, frequency=23hour, level_type=ml

yields

slk:///arch/bu1213/NGC2/ngc2013/outdata/ngc2013_atm_ml_23h_inst_1_20200201T000000Z.nc

Intake-ESM / slkspec / find_files / outtake

- Intake catalogs provide semantic data access
- slkspec allows python to download files from the HSM system
- find_files (shell script) provides command line access to the catalog and slkspec
- outtake (py module) provides high-level interface for the combination of intake catalogs and slk.

Pro:

- Provides analysis-ready data by reduction of data preparation tasks for users e.g. by automated aggregation of files to data sets
- Enables data access independent of file format and storage location from one catalog

Con:

- Only integrates seamlessly with python.
- Slkspec: no asynchronous tape retrieval - other operations are blocked

Freva - Free Evaluation System

by freva

- Freva is a data search and analysis platform
- The system comes with a web, command line and python user interface
- Con not only search for data but also run scientific analysis code

Pros:

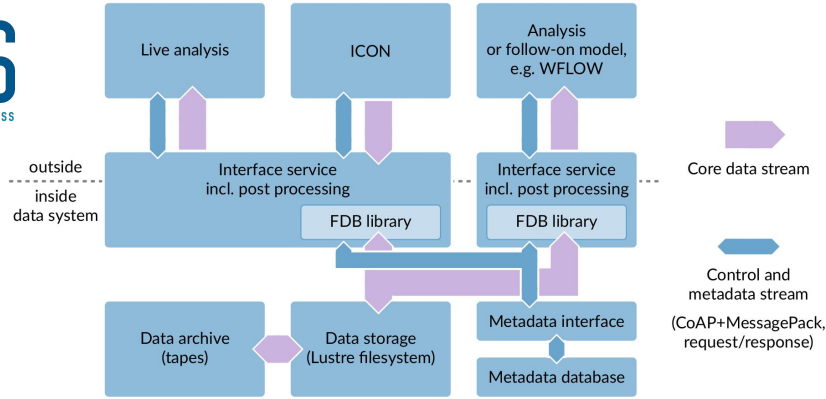
- Central cataloging system via apache solr -> fast and scalable.
- Reproducible data analysis because configs are stored in databases.
- Intuitive web UI -> click.

Cons:

- RESTful workflow API only in dev stage -> currently hard to design thin clients in other languages - but this will change.
- There is no centralised DKRZ instance of Freva.

https://openresearchsoftware.metajnl.com/article/10.5334/jors.253/

Borges data system



- Goal: Make data use fully independent from storage (location, format, ...)
- Central, global index of data in the system
  - Make data fully discoverable and searchable
  - Focus for prototype: ICON-output data with automatic metadata (incl. model version, configuration, ...) using GRIB standard
- Data-access only in a semantic way ("what" instead of "where"); no files visible/accessible, as if uploaded to a cloud storage
- All access through a middleware to
  - Keep central index and storage consistent
  - Abstract from storage back-end in a seamless way: Classical file system, tape, other HPC centers
  - Make data easily shareable among users
  - Integrate with current resource accounting (storage use, compute time) and systems (Lustre, HSM, …)
  - With minimal overhead compared to "direct file access"
- Lightweight client library to integrate into existing interfaces: cdo, Python (Intake-like), ?

Who is responsible for managing the data storage?

	“The system”	“The user”
Pros	- Defined interfaces and abstractions ensure consistency and robustness - Data access is known and data can be moved around automatically as needed or demanded - Allows for optimization in the background	- Most libraries support reading from file-based data and can be continued to be used - Little adaptation by the user needed - Low-level control for users if desired
Cons	- No direct file access, only via system interface - Dependence not only on the underlying back-ends but also on the data system - Users must be able to trust the data system as much as e.g. Lustre or HSM	- Little auto-”magic”, users have to move data around themselves (though supported by tools) - Users have to manage access permissions low-level - Danger of continuing to be messy and uneconomic

…but compromises possible(?)

Let’s discuss!!

- Do you have questions?
- Do you work on a similar project?
- Do you have a specific use case which we might have overlooked?
- Do you have any other feedback?

More technical backup-slides also available ;-)