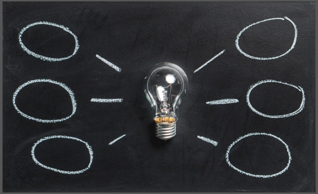
Computer Science Department

University of Reading

Overcoming Storage Issues of Earth-System Data with Intelligent Storage Systems



Julian Kunkel, ESiWACE WP4 team, NGI team

http://aces.cs.reading.ac.uk

18th Workshop on high performance computing in meteorology

2018-09-27

LIMITLESS **POTENTIAL** | LIMITLESS **OPPORTUNITIES** | LIMITLESS **IMPACT**

Outline

1 Motivation

2 Workflows

3 ESDM

4 Outlook

5 Summary

Climate/Weather Workflows

Challenges

- Programming of efficient workflows
- Efficient analysis of data
- Organizing data sets
- Ensuring reproducability of workflows/provenance of data
- Meeting the compute/storage needs in future complex hardware landscape

Expected Data Characteristics in 2020+

- Velocity: Input 5 TB/day (for NWP; reduced data from instruments)
- Volume: Data output of ensembles in PBs of data
- Data products are used by 3rd parties
- Various file formats

I/O Challenges

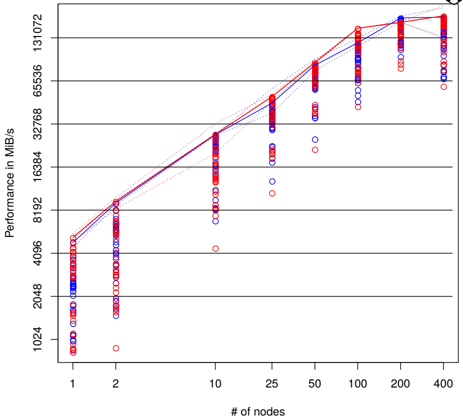
University of Reading

- Large data volume and high velocity
- Data management practice does not scale and are not portable
  - Cannot easily manage file placement and knowledge of what file contains
  - Hierarchical namespaces does not reflect use cases
  - Individual strategies at every site
- The storage stack becomes more inhomogeneous
  - Non-volatile memory, SSDs, HDDs, tape
  - Node-local, vs. global shared, partial access (e.g., racks)
- Suboptimal performance & performance portability
  - Users cannot properly exploit the hardware / storage landscape
  - Tuning for file formats and file systems necessary at the *application* level
- Data conversion/merging is often needed
  - To combine data from multiple experiments, time steps, ...

Manual Tuning is Non-Trivial

University of Reading

- Goal: Identify good settings for I/O
- Measured on Mistral Phase1; Lustre
- Run: IOR, indep. files, 10 MiB blocks
  - Proc. per node: 1,2,4,6,8,12,16
  - Stripes: 1,2,4,16,116
- Note: Slowest client stalls others



Best settings for read (excerpt)

Nodes	PPN	Stripe	W1	W2	W3	R1	R2	R3	Avg. Write	Avg. Read	WNode	RNode	RPPN
1	6	1	3636	3685	1034	4448	5106	5016	2785	4857	2785	4857	809
2	6	1	6988	4055	6807	8864	9077	9585	5950	9175	2975	4587	764
10	16	2	16135	24697	17372	27717	27804	27181	19401	27567	1940	2756	172

A Typical Current Software Stack for NWP/Climate

University of Reading

- Domain semantics
  - XIOS writes independent variables to one file each
  - 2nd servers for performance reasons
- Issues with the Data model in NetCDF4/HDF5
  - Performant mappings to files are limited
    - Map data semantics to one "file"
    - HDF5 shared file format notorious inefficient
  - Domain metadata is treated like normal data
    - Need for higher-level databases like Mars
  - Interfaces focus on variables but lack features
    - Workflows
    - Information life cycle management

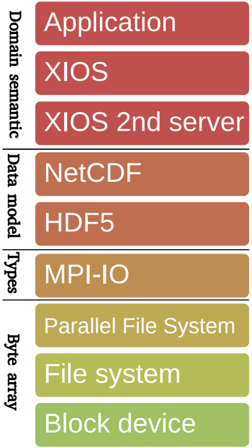
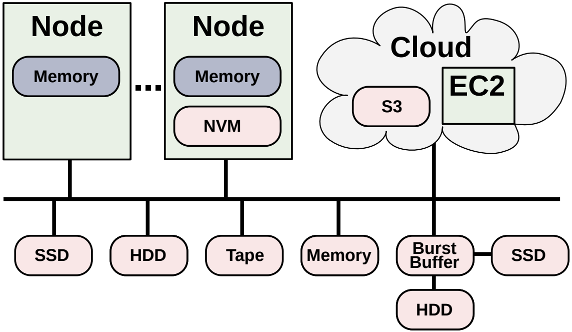


Figure: Typical I/O stack

Problem: Future Systems Coexistence of Storage

University of Reading



- Goal: We shall be able to use all storage technologies concurrently
  - Without explicit migration etc. put data where it fits
  - Administrators just add a new technology (e.g., SSD pool) and users benefit
- Why no manual configuration, e.g., partitioning by file?
  - Reminds on implementing manual RAID across HDDs
  - Increases burden of data management

Outline

1 Motivation

2 Workflows

3 ESDM

4 Outlook

5 Summary

Scenario: Large Simulation

- Assume large scale simulation, timeseries (e.g., 1000 y climate)
- Assume manual data analysis needed (but time consuming)
- We need all 1000 y for detailed analysis!

A typical workflow execution

- Run simulation for 1000 y
  - Store various data on (online) storage
  - Keep checkpoints to allow reruns
  - Maybe backup data in archive
- Explore data to identify how to analyze data
- At some point: Run the analysis on all data
- Problem: Occupied storage capacity

Alternative Workflows Done by Scientists

University of Reading

Recomputation

- Run simulation
  - Store checkpoints
  - Store only selected data (wrt. resolution, section, time)
- Explore data
  - Run recomputation to create needed data (e.g., last year)
- At some point: run analysis across all data needed
- This is a manual process, must consider
  - Runtime parameters
  - System configuration/available resources

Another Alternative Workflows

Provided by more intelligent storage and better workflows

- Run simulation
  - Store checkpoints on node-local storage
    - Redundancy: from time to time restart from another node
  - Store selected data on online storage (e.g., 1% of volume)
    - Also store high-resolution data sample (e.g., 1% of volume)
  - Store high-resolution data directly on tape
- Explore data on snapshot
- Month later: schedule analysis of data needed
  - The system retrieves data from tape
  - Performs the scheduled operations on streams while data is pulled in
  - Informs user about analysis progress
- Some people do this manually or use some tools to achieve similarly
  - Aim for domain & platform independence and heterogenous HPC landscapes

Scenario: Data Organization

University of Reading

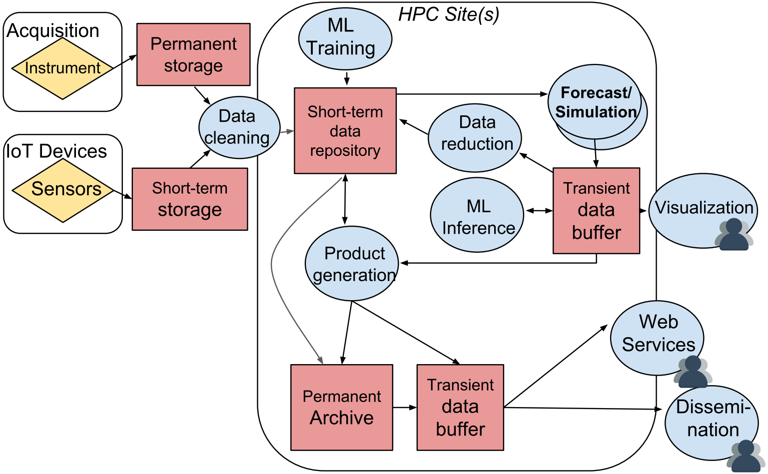
Goal: Semantic Namespace

- Provide features of data repositories (e.g., MARS) to explore data
- User-defined properties but provide means to validate schemas
- Similar to MP3 library ...

High-Level questions addressed by them

- What experiments did I run yesterday?
- Show me the data of experiment X, with parameters Z...
- Cleanup unneeded temporary stuff from experiment X
- Compare the mean temperature of one model for one experiment across model versions

Smarter Climate/Weather Workflows in 2020+



- IoT (and mobile devices)
  - Additional data provider
  - Improves short-term weather prediction
- Machine learning support
  - Localize known patterns
  - Interactive use
    Visual analytics
- Data reduction
  - Output is triggered by events (ML)
  - Compress data of ensembles

Outline

1 Motivation

2 Workflows

3 ESDM

4 Outlook

5 Summary

Earth-System Data Middleware

Part of the ESiWACE Center of Excellence in H2020

Design Goals of the Earth-System Data Middleware

1. Relaxed access semantics, tailored to scientific data generation
   - Avoid false sharing (of data blocks) in the write-path
   - Understand application data structures and scientific metadata
   - Reduce penalties of **shared** file access
2. Site-specific (optimized) data layout schemes
   - Based on site-configuration and performance model
   - Site-admin/project group defines mapping
   - Flexible mapping of data to multiple storage backends
   - Exploiting backends in the storage landscape
3. Ease of use and deployment particularly configuration
4. Enable a configurable namespace based on scientific metadata

Expected Benefits

University of Reading

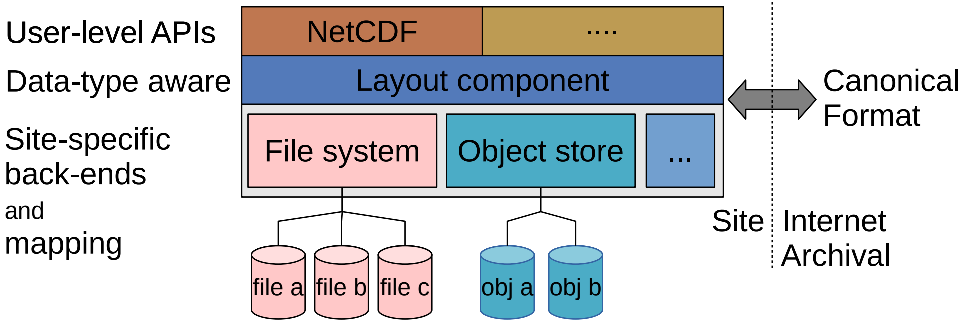
- Independent, share-nothing lock-free writes from parallel applications
- Storage layout is optimized to local storage
  - Exploits characteristics of diver storage
  - Preserve compatibility by creating platform-independent file formats on the site boundary/archive
- Less performance tuning from users needed
  - One data structure can be fully or partially replicated with different layouts
  - Using multiple storage systems concurrently
- (Expose/access the same data via different APIs)[1]
- (Flexible and automatic namespace)

[1] Not shown in ESiWACE scope

Architecture

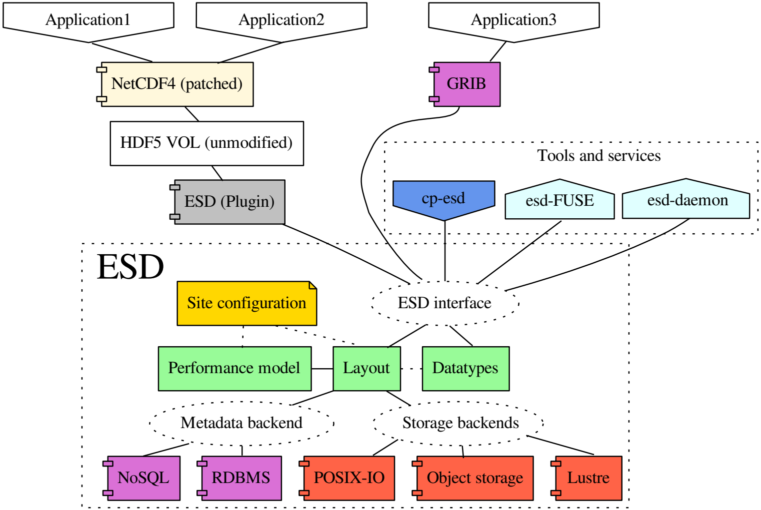
Key Concepts

- Middleware utilizes layout component to make placement decisions
- Applications work through existing API (currently: NetCDF library)
- Data is then written/read efficiently; potential for optimization inside library



Architecture: Detailed View of the Software Landscape

University of Reading



Evaluation

System

- Test system: DKRZ Mistral supercomputer
- Nodes: 200 (we have also other measurements)

Benchmark

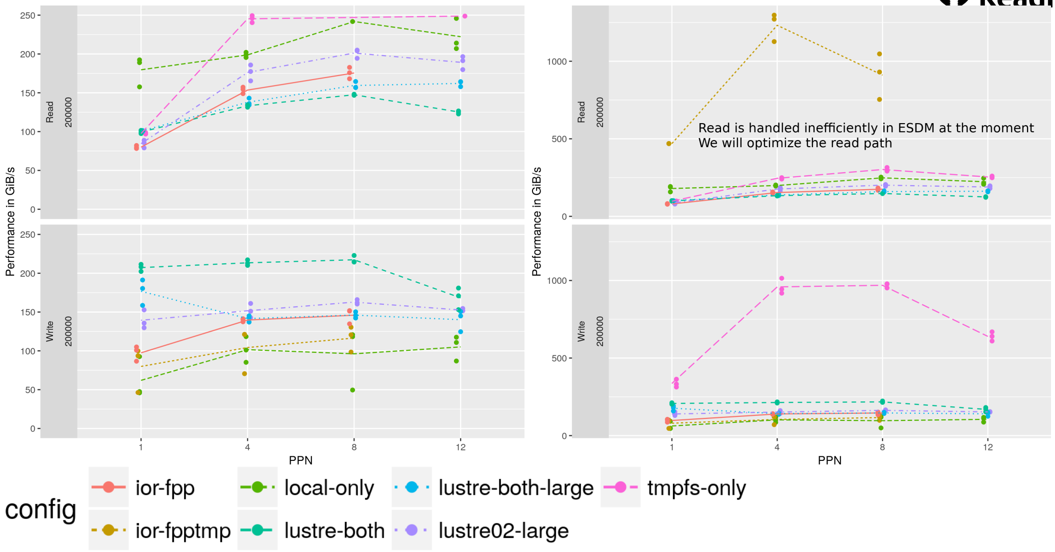
- Uses ESDM interface directly; Metadata on Lustre
- Write/read a timeseries of a 2D variable
- Grid size: $200k \cdot 200k \cdot 8Byte \cdot 10iterations$
- Data volume: size = 2980 GiB; compared to IOR performance

ESDM Configurations

- Splitting data into fragments of 100 MiB (or 500)
- Use different storage systems
- Uses 8 threads per node (max per application 400)

Measured Performance

University of Reading



Outline

1 Motivation

2 Workflows

3 ESDM

4 Outlook

5 Summary

My Long Term Vision: Separation of Concerns

University of Reading

Decisions made by scientists

- Scientific metadata
- Declaring workflows
  - Covering data ingestion, processing, product generation and analysis
  - Data life cycle (and archive/exchange file format)
  - Constraints on: accessibility (permissions), ...
  - Expectations: completion time (interactive feedback human/system)
- Modifying workflows on the fly
- Interactive analysis, e.g., Visual Analytics
- Declaring value of data (logfile, data-product, observation)

Separation of Concerns

Programmers of models/tools (e.g., Ophidia)

- Decide about the most appropriate API to use (e.g., NetCDF + X)
- Register compute snippets (analytics) to API
- Do not care **where** and **how** computation is done

Decisions made by the (compute/storage) system

- Where and how to store data, including file format
- Complete management of available storage space
- Performed data transformations, replication factors, storage to use
- Including scheduling of compute/storage/analysis jobs (using, e.g., ML)
- Where to run certain data-driven computations (**Fluid-computing**)
    - Client, server, in-network, cloud

A Proposed Software Stack for NWP/Climate

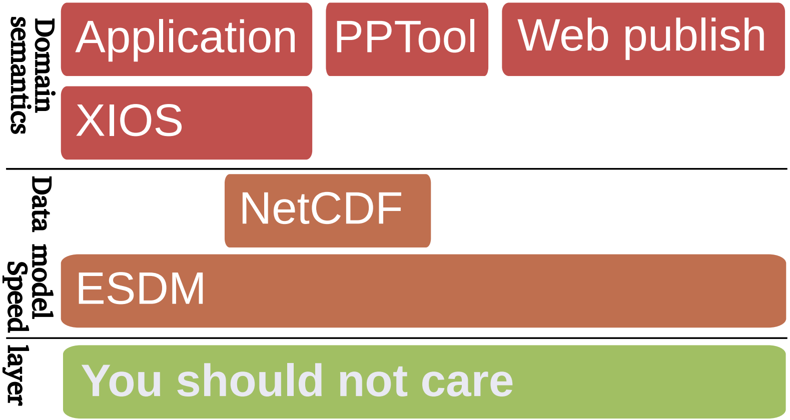


Figure: ESDM Stack

ESiWACE2 Plans

University of Reading

- Hardening of ESDM
- Integrate an improved performance model
- Direct NetCDF integration (instead of HDF5)
- Improvements on compression (also for NetCDF)
- Optimized backends for, e.g., Clovis, IME, S3
- Integrate Cylc and ESDM
  - Extensions to Cylc to cover data lifecycle, I/O performance needs
  - Cylc to provide information about workflow to ESDM
  - ESDM to make superior placement decisions
- Industry proof of concepts for EDSM
- Supporting post-processing, analytics and (in-situ) visualization
  - Support of computation workflows within ESDM
  - Integration with analysis tools, e.g., Ophidia, CDO

ESDM is just the Beginning: Next Generation Interfaces

University of Reading

Towards a new I/O stack considering:

- Smart hardware and software components
- Storage and compute are covered together
- User metadata and workflows as first-class citizens
- Self-aware instead of unconscious
- Improving over time (self-learning, hardware upgrades)

Why do we need a new domain-independent API?

- Other domains have similar issues
- It is a hard problem approached by countless approaches
- Harness RD&E effort across domains

Pursued Community Strategy

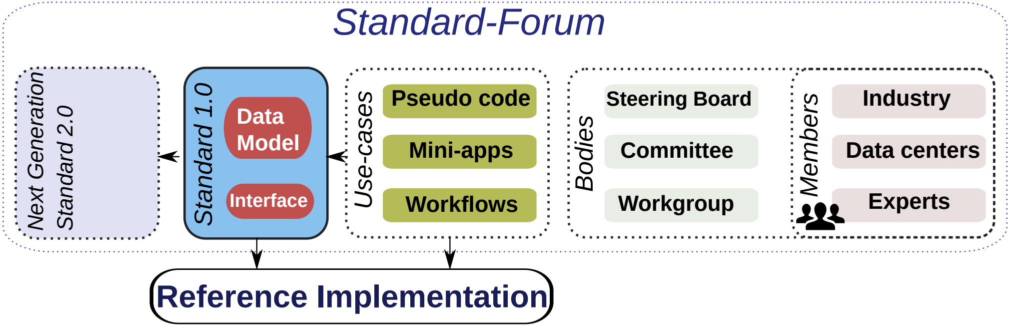
University of Reading

- The **standardization** of a high-level *data model & interface*
  - Targeting data intensive and HPC workloads
  - To have a future: must be beneficial for Cloud/Big Data + Desktop, too
  - Lifting semantic access to a new level
- Development of a reference implementation of a **smart runtime system**
  - Implementing key features
- Demonstration of benefits on socially relevant data-intense apps

Development of the Data Model and API

University of Reading

- Establishing a Forum (similarly to the Message Passing Interface – MPI)
- Model targets High-Performance Computing and data-intensive compute
- Open board: encourage community collaboration



Summary

University of Reading

- The I/O stack is complex
- Integrated and smart compute & storage is the future
- ESDM is a transitional approach towards next-gen interfaces

Participate defining NG interfaces

- Join the mailing list
- Visit: `https://ngi.vi4io.org`