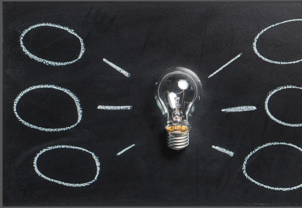
Department of Computer Science

Data-Centric I/O and Next Generation Interfaces



Limitless Storage
Limitless Possibilities

https://hps.vi4io.org

Julian M. Kunkel and partners from the NGI forum

HPC-IODC Workshop

2019-06-20

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

Outline

University of Reading

1 Workflows

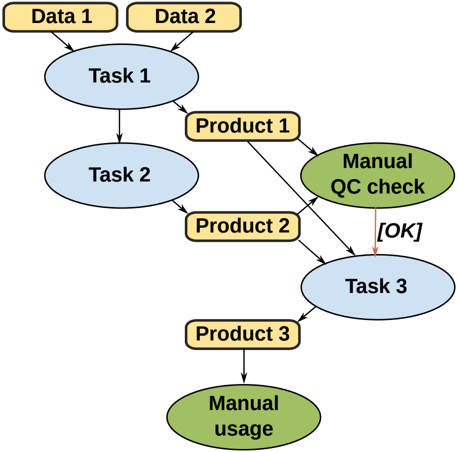
2 The NGI Strategy

3 Summary

Workflows

University of Reading

- Consider workflow from 0 to insight
  - Needs/produces data
  - Uses tasks
    - Parallel apps?
    - Big data tools?
    - Manual analysis
  - May need month to complete
  - Manual tasks are unpredictable
  - What are users interested in?
- Not well described in HPC
  - Mostly hardcoded in scripts
- Can we exploit workflows?
  - Does it matter where data is?
  - Vendors simulations?
  - Enforce ILM as needed by users



Planning HPC Resources

University of Reading

Planning for Cern/LHC and other big experiments

- A detailed planning of activities is performed
- Experiments are proposed with plans (time, resource utilization)

Planning for Data Centers

- May include: Time needed, CPU (GPU) hours, storage space
- After resources are granted scientists do what they want
  - Some limitations, e.g., quota, compute limit
  - But access patterns?
  - The system is not aware what possibly could happen
  - The data center does not know suffiently what users do
- Additionally: Execution uses often tools with 40year old concepts

Planning HPC Resources: An Alternative Universe

University of Reading

- Scientists deliver
  - detailed but abstract workflow orchestration
  - containers with all software
  - data management plan with data lifecycle
  - time constraints and budget
- Data centers and vendors
  - Simulate the execution before workflow is executed
  - Estimate costs, energy consumption
  - Determine if it is the best option to run
- Systems
  - Utilize the information to orchestrate I/O
  - Make decisions about data location and placement:
    - Trade compute vs. storage and energy/costs vs. runtime
  - Ensure proper execution
- Provocing: Big data is ahead in such an agenda!

An Alternative Universe: Separation of Concerns

University of Reading

Decisions made by users

- Defining relevant 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)
- Monitoring and if needed modifying workflows on the fly
- Analyzing data interactively, e.g., Visual Analytics
- Declaring value of data (logfile, data-product, observation)

Separation of Concerns

Programmers of models/tools

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

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, your connected laptop

Outline

University of Reading

1 Workflows

2 The NGI Strategy

3 Summary

The Next Generation Interfaces Initiative

University of Reading

Goals

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

Next Generation Interfaces

Towards a new data centric compute/IO 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)

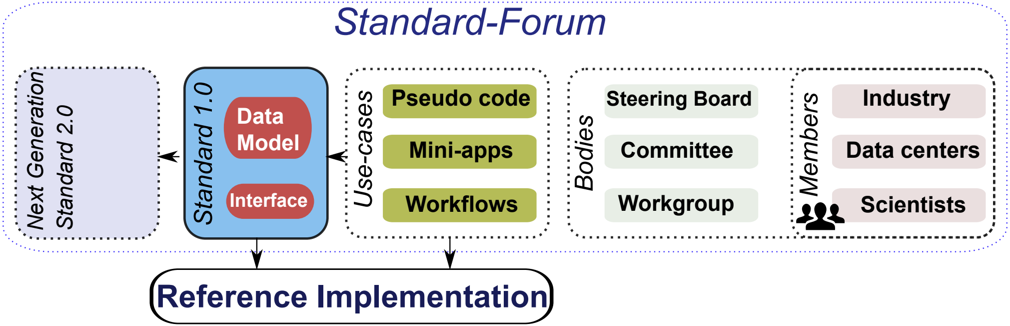
Why do we need a new domain/funding independent API?

- Many domains have similar issues; projects are competitive
- It is a hard problem approached by countless approaches
- Harness RD&E effort across domains

Development of the Data Model and API

University of Reading

- Establishing a Forum similarly to MPI
- Define data model for HPC
- Open board: encourage community collaboration



One Year NGI in Retrospective

University of Reading

- One year ago, we started NGI as informal collaboration
- Fair interest in individuals from
  - Institutions (UCAR, Sandia, Argonne, ...)
  - Vendors (NVIDIA, (Mellanox), Kove, DDN, ...)
- Issue: converting committments into actions (without funding)
- Started to work on white-papers (Minipapers)
  - Use-cases, visions, APIs, coordination
  - Everyone is welcome to contribute
- We will publish the first ones before SC

Summary

University of Reading

- There is a huge potential for the next-generation interface
- Can the community work together to define next generation APIs?
- Can the community work together to define vision and next generation APIs?

Participate defining NG interfaces

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

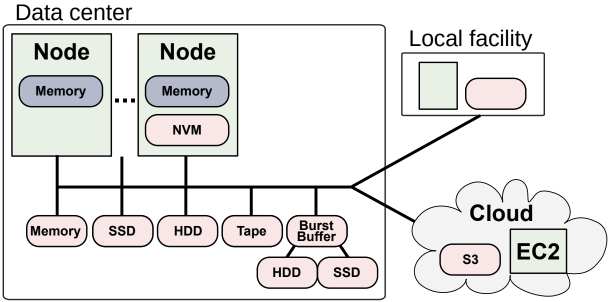
Appendix

Challenges Faced by HPC I/O

- Difficulty to analyze behavior and understand performance
  - Unclear access patterns (users, sites)
- Coexistence of access paradigms in workflows
  - File (POSIX, ADIOS, HDF5), SQL, NoSQL
- Semantical information is lost through layers
  - Suboptimal performance, lost opportunities
  - All data treated identically (up to the user)
- Re-implementation of features across stack
  - Unpredictable interactions
  - Wasted resources
- Restricted (performance) portability
  - Optimizing each layer for each system?
  - Users lack technological knowledge for tweaking
- Utilizing the future storage landscapes
  - No performance awareness, manual tuning and mapping to storage needed

Future Systems: Coexistence of Storage Systems

University of Reading



- 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

Alternative Software Stack

University of Reading

Some examples of the zoo of alternatives

- High-level abstractions: Dataclay, Dataspaces, Mochi
- Data models: ADIOS, HDF5, NetCDF, VTK
- Standard API across file formats: Silo, VTK, CDI, HDF5
- Low-level libraries: SIONlib, PLFS
- Storage interfaces: MPI-IO, POSIX, vendor-specific (e.g., CLOVIS), S3
- Big-data: HDFS, Spark, Flink, MongoDB, Cassandra
- Research: Countless storage system prototypes every year
- Projects: EXAHDF, Maestro (FET Proactive)

Outline

University of Reading

4 Challenges

5 Potential Interfaces

A Pragmatic View

University of Reading

- Take existing data model like VTK (or NetCDF) as baseline
- With a hint of:
  - Scientific metadata handling
  - Workflow and processing interface
  - Information lifecycle management
  - Hardware model interface (hardware provides its own performance models)
- First prototype utilizes existing software stack
  - Like Cylc for workflows
  - Like MongoDB for metadata
  - Like a parallel file system (or object storage)
- Work on:
  - Scheduler for performant mapping of data/compute to storage/compute
  - A FUSE client for flexible data mappings on semantic metadata
  - Importer/Exporter tools for standard file formats
- Add magic (knowledge of experts developing APIs)
- Next prototype: move on with true implementation

Next-Generation HPC IO API Key Features

- High-level data model for HPC
  - Storage understands data structures vs. byte array
  - Relaxed consistency
- Semantic namespace and storage-aware data formats
  - Organize based on domain-specific metadata (instead of file system)
  - Support domain-specific operations and addressing schemes
- Integrated processing capabilities
  - Offload data-intensive compute to storage system
  - Managed data-driven workflows supporting events and services
  - Scheduler maps compute and I/O to hardware
- Enhanced data management features
  - Information life-cycle management (and value of data)
  - Embedded performance analysis
  - Resilience, import/export, ...