HPC I/O for Computational Scientists: General Principles

Presented to
ATPESC 2017 Participants

Rob Latham and Phil Carns
Mathematics and Computer Science Division
Argonne National Laboratory

Q Center, St. Charles, IL (USA)
8/4/2017

EXASCALE COMPUTING PROJECT

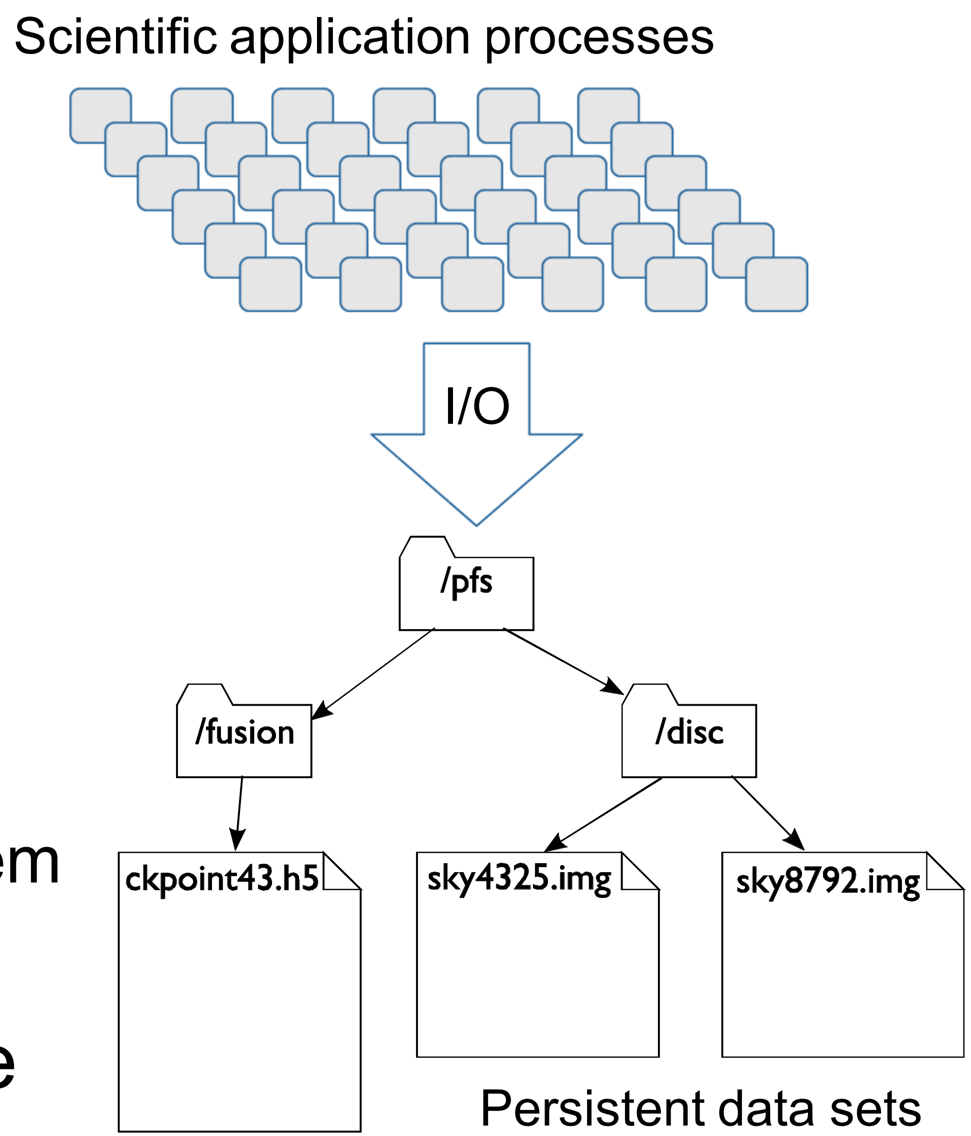
National Nuclear Security Administration

U.S. DEPARTMENT OF ENERGY | Office of Science

Argonne NATIONAL LABORATORY

HPC I/O 101

- HPC I/O: storing and retrieving persistent scientific data on a high performance computing platform
  - Encompasses hardware components, system software, and applications
  - Data is usually stored on a **parallel file system**
  - On the surface this looks like any other file system
- Optimized for high-volume parallel access: many application processes accessing large data sets at the same time



HPC I/O Systems

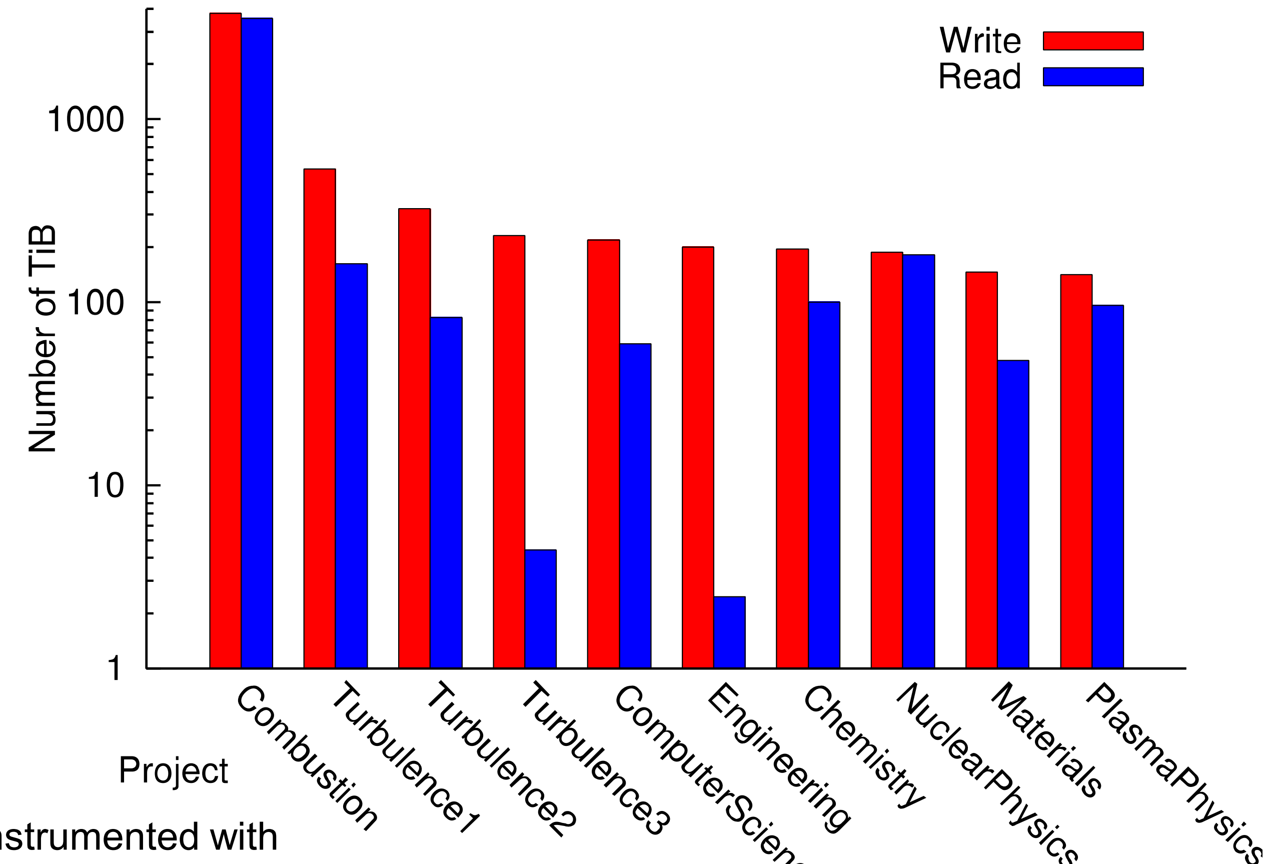
- Hardware: disks, disk enclosures, servers, and networks
- Software: parallel file system, libraries, parts of the operating system
- Applications: how applications use the storage system

Most common reasons for reading or writing data:

- Productive I/O: storing scientific results
- Defensive I/O: saving state in case the application or system crashes
- Analysis I/O: scientific discovery from previous results

HPC I/O system usage

It's not just checkpoints – scientists are reading large volumes of data *into* HPC systems as part of their science.



Top 10 data producer/consumers instrumented with Darshan from August 2014 to January 2015 (Mira).

So you want to store data on an HPC system?

Let's talk about the basics (applicable to any system):

- **What is unique about HPC I/O?**
- **How do you account for those things in your application?**

What is unique about HPC I/O?
#1: multiple storage systems to choose from

- **Most HPC systems have different file systems for different purposes**
- **Step 1: pick the right resource for your needs**
- **Consult site documentation, ask support if you aren't sure**

Example: NERSC file systems, 2017

File System	Path	Type	Peak Performance	Default Quota	Backups	Purge Policy
Global homes	$HOME	GPFS	Not For IO Jobs	40 GB 1,000,000 Inodes	Yes	Not purged
Global project	/project/projectdirs/*projectname*	GPFS	130GB/Second	1 TB 1,000,000 Inodes	Yes if ≤ 5 TB quota. No if quota is > 5 TB.	Not purged
Edison local scratch	$SCRATCH	Lustre	168GB/Second (across 3 file systems)	10 TB 5,000,000 Inodes	No	Files not accessed for 8 weeks are deleted
Cori local scratch	$SCRATCH, $CSCRATCH (from other systems)	Lustre	700GB/Second	20 TB 10,000,000 Inodes	No	Files not accessed for 12 weeks are deleted
Cori Burst Buffer	$DW_JOB_STRIPED, $DW_PERSISTENT_STRIPED_*XXX*	DataWarp	1.7 TB/s, 28M IOP/s	none	No	Data is deleted at the end of
Archive (HPSS)	Typically accesse inside of NERSC					

File Systems' Intended Use

File System	Intended Use	File Optimization
Global homes	Hold source code, executables, configuration files, etc. NOT meant to hold the output from your application runs; the scratch or project file systems should be used for computational output.	Optimized for small to medium sized files.
Global project	Sharing data within a team or across computational platforms. Store application output files. Intended for actively used data.	Optimized for high-bandwidth, large-block-size access to large files.
Scratch file systems	Edison and Cori each have large, local, parallel scratch file systems dedicated to that systems. The scratch file systems are intended for temporary uses such as storage of checkpoints or application input and output. If you need to retain files longer than the purge period, the files should be copied to global project or to HPSS.	Optimized for high-bandwidth, large-block-size access to large files.
Burst Buffer	Cori's Burst Buffer provides very high performance I/O on a per-job or short-term basis. It is particularly useful for codes that are IO-bound, for example, codes that produce large checkpoint files, or that have small or random I/O reads/writes.	Optimized for high-bandwidth access for all size files and all access patterns.
Archive	Long term archival of important and unique data, such as data from published results.	Optimized for file sizes or bundles of files in the 100s of GB range.

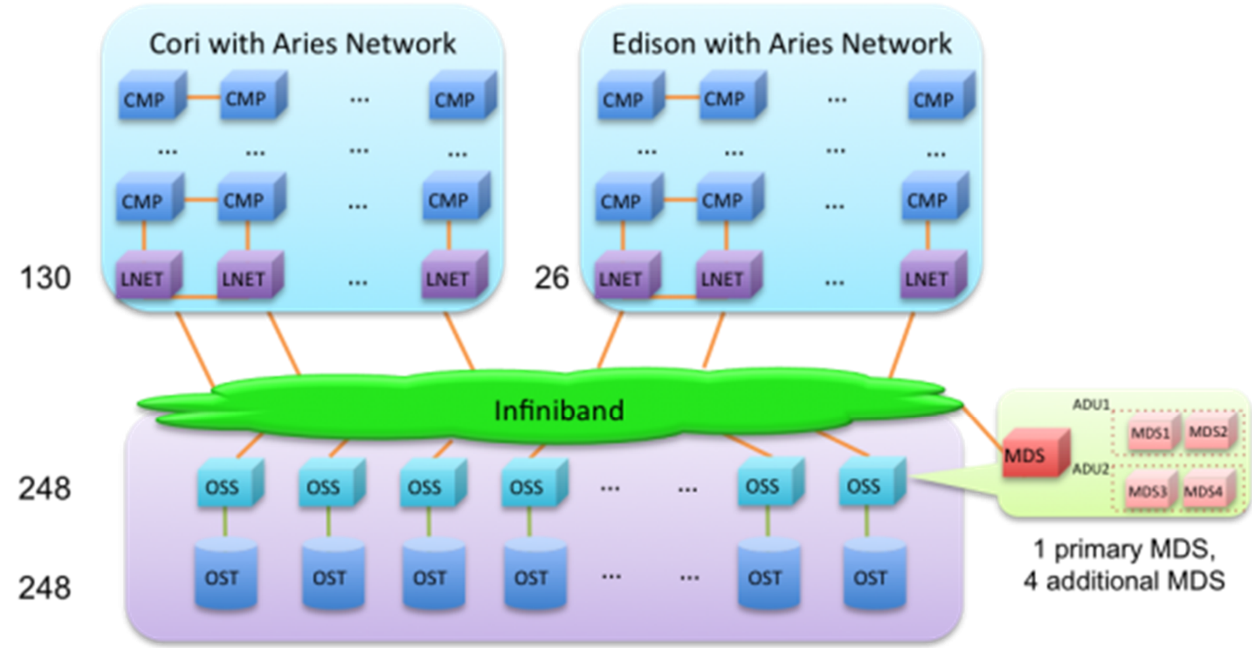
Example: choosing the right storage system

- **Home** file system on Mira:
  – 24 servers, extra level of replication, 3 storage appliances (DDN couplet)
- **FS0** project file system on Mira:
  – 128 servers, no extra replication, 16 storage appliances (DDN couplet)
  – Also more disk drives per server
- Both are accessible to your job, but:
  – The former is tuned for small file, login node activity, high availability
  – The latter is tuned for > 6x the performance for large parallel jobs

What is unique about HPC I/O?
#2: the storage system is large and complex

- It looks like a normal file system
- But there are 10,000 or more disk drives!
- This means that an HPC file system will often behave differently from a "normal" file system

Cori scratch file system diagram
NERSC, 2017

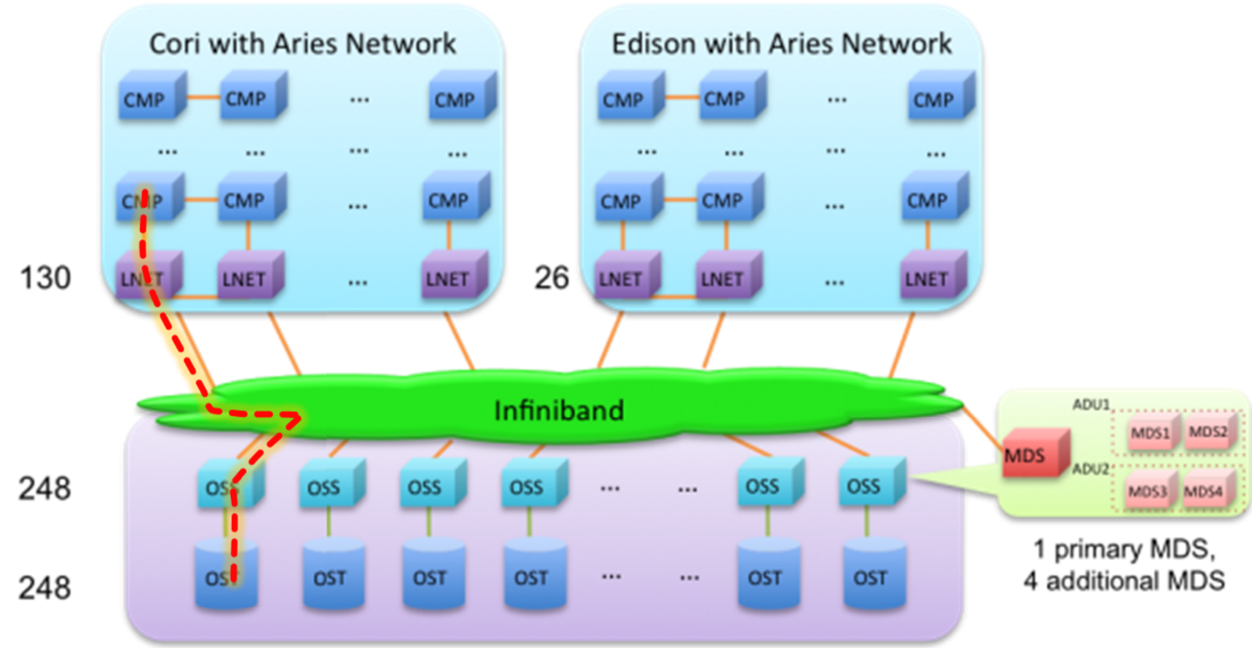


Each OSS controls one OST. The Infiniband connects the MDS, ADUs and OSSs to the LNET routers on the Cray XC System. The OSTs are configured with GridRAID, similar to RAID6, (8+2), but can restore failure 3.5 times faster than traditional RAID6. Each OST consists of 41 disks, and can deliver 240TB capacity.

What is unique about HPC I/O?
#2: the storage system is large and complex

- **Moving data from one compute node to a disk drive takes several "hops"**
- **Therefore, the *latency*, or time to complete a single small operation by itself, is relatively poor**

Cori scratch file system diagram
NERSC, 2017

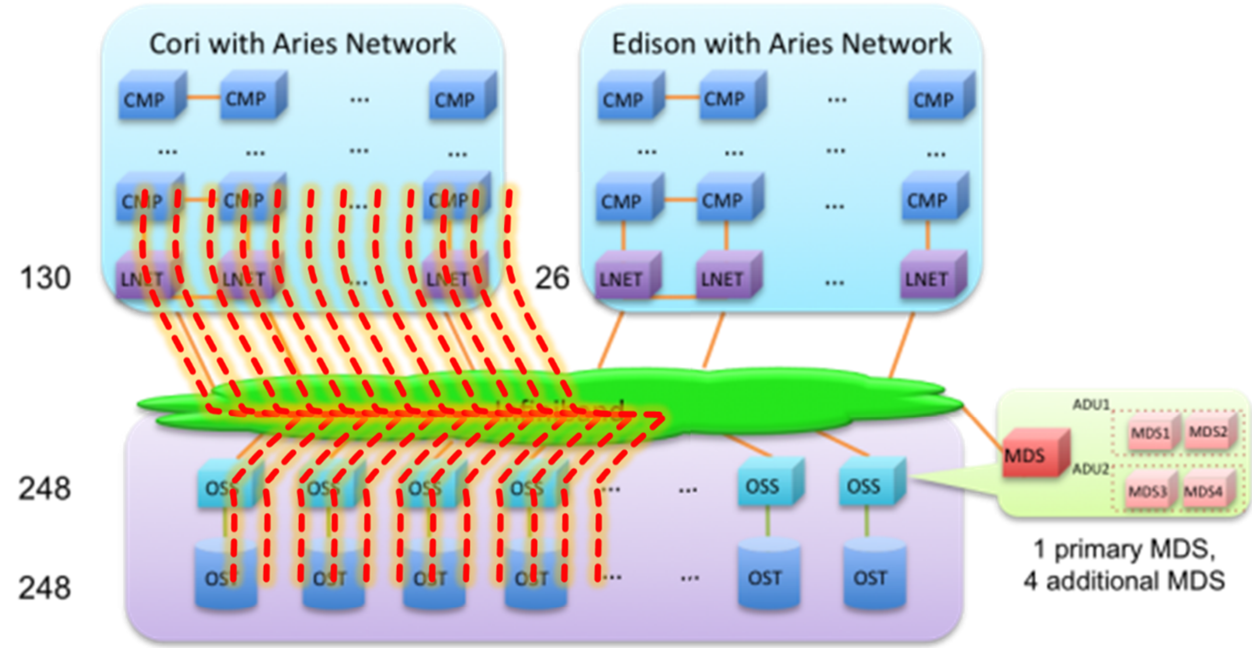


Each OSS controls one OST. The Infiniband connects the MDS, ADUs and OSSs to the LNET routers on the Cray XC System. The OSTs are configured with GridRAID, similar to RAID6, (8+2), but can restore failure 3.5 times faster than traditional RAID6. Each OST consists of 41 disks, and can deliver 240TB capacity.

What is unique about HPC I/O?
#2 the storage system is large and complex

- **But the network is fast, and you can do many I/O operations simultaneously**
- **Therefore, the *aggregate bandwidth,* or rate of parallel data access, is tremendous**
- **Step 2: Parallel I/O tuning is all about playing to the system's strengths:**
  - **Move data in parallel with big operations**
  - **Avoid sequential small operations**

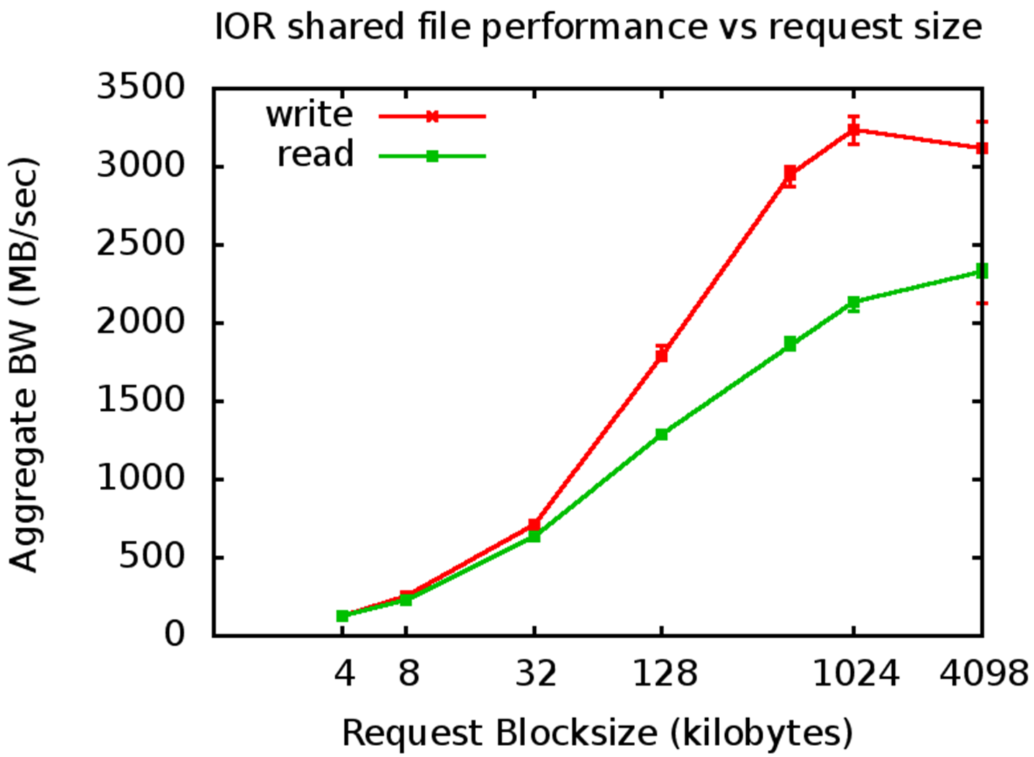
Cori scratch file system diagram
NERSC, 2017



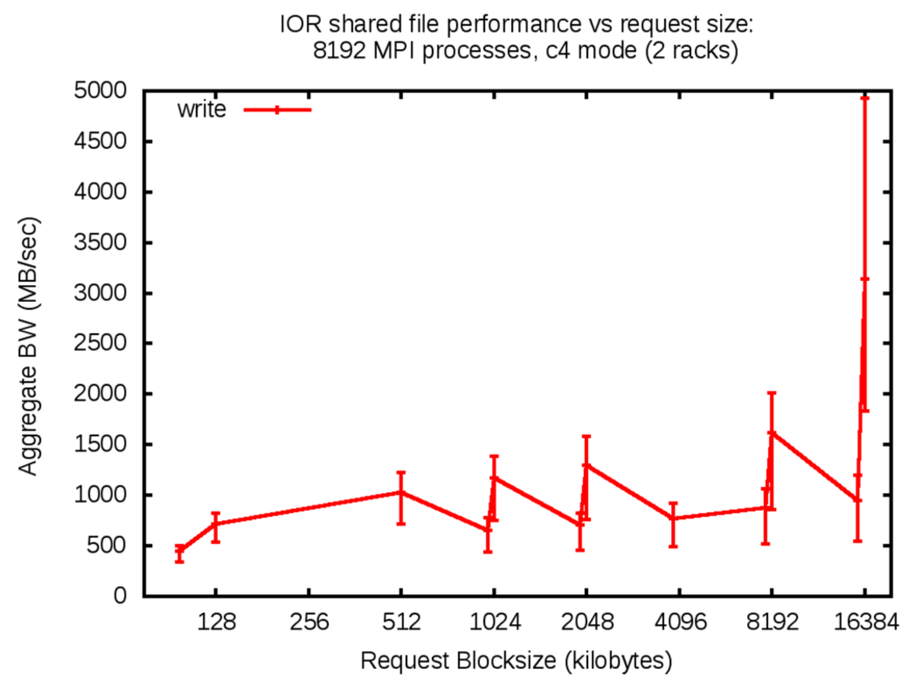
Each OSS controls one OST. The Infiniband connects the MDS, ADUs and OSSs to the LNET routers on the Cray XC System. The OSTs are configured with GridRAID, similar to RAID6, (8+2), but can restore failure 3.5 times faster than traditional RAID6. Each OST consists of 41 disks, and can deliver 240TB capacity.

Example of HPC I/O strengths and weaknesses

Interconnect latency has a significant impact on effective rate of I/O. Typically I/Os should be in the O(Mbytes) range.



2K processes of IBM Blue Gene/P at ANL.



8k processes of IBM Blue Gene /Q at ANL

Why? For small operations it takes too much time to coordinate the devices (i.e., startup cost, handshaking) relative to the amount of useful work done per operation.

Don’t prep a space shuttle to get groceries ;-)

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

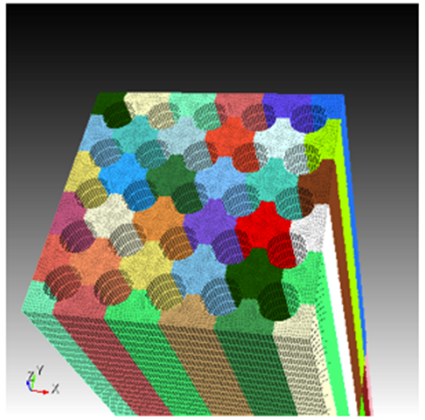
What is unique about HPC I/O?
#3 sophisticated application data models

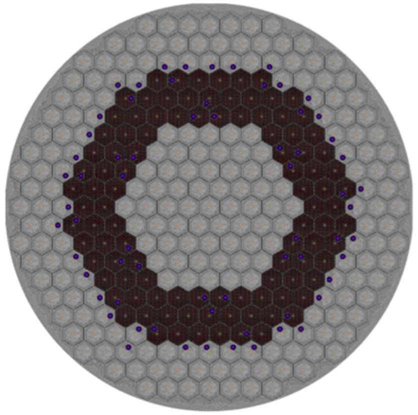
Images from T. Tautges (ANL) (upper left), M. Smith (ANL) (lower left), and K. Smith (MIT) (right).

- Applications use advanced data models to fit the problem at hand
  - Multidimensional typed arrays, images composed of scan lines, …
  - Headers, attributes on data
- I/O systems have very simple data models
  - Tree-based hierarchy of containers
  - Some containers have streams of bytes (files)
  - Others hold collections of other containers (directories or folders)

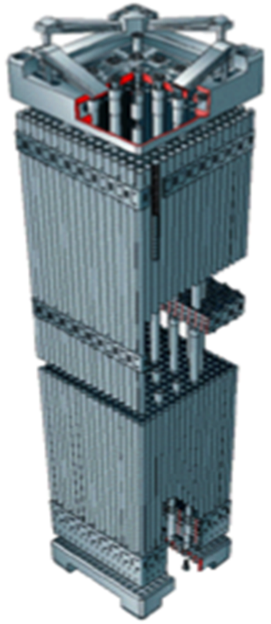
Step 3: Use data libraries that help to efficiently map between your data model and the file system.

We'll learn more about this as the day goes on!





Model complexity:
Spectral element mesh (top) for thermal hydraulics computation coupled with finite element mesh (bottom) for neutronics calculation.



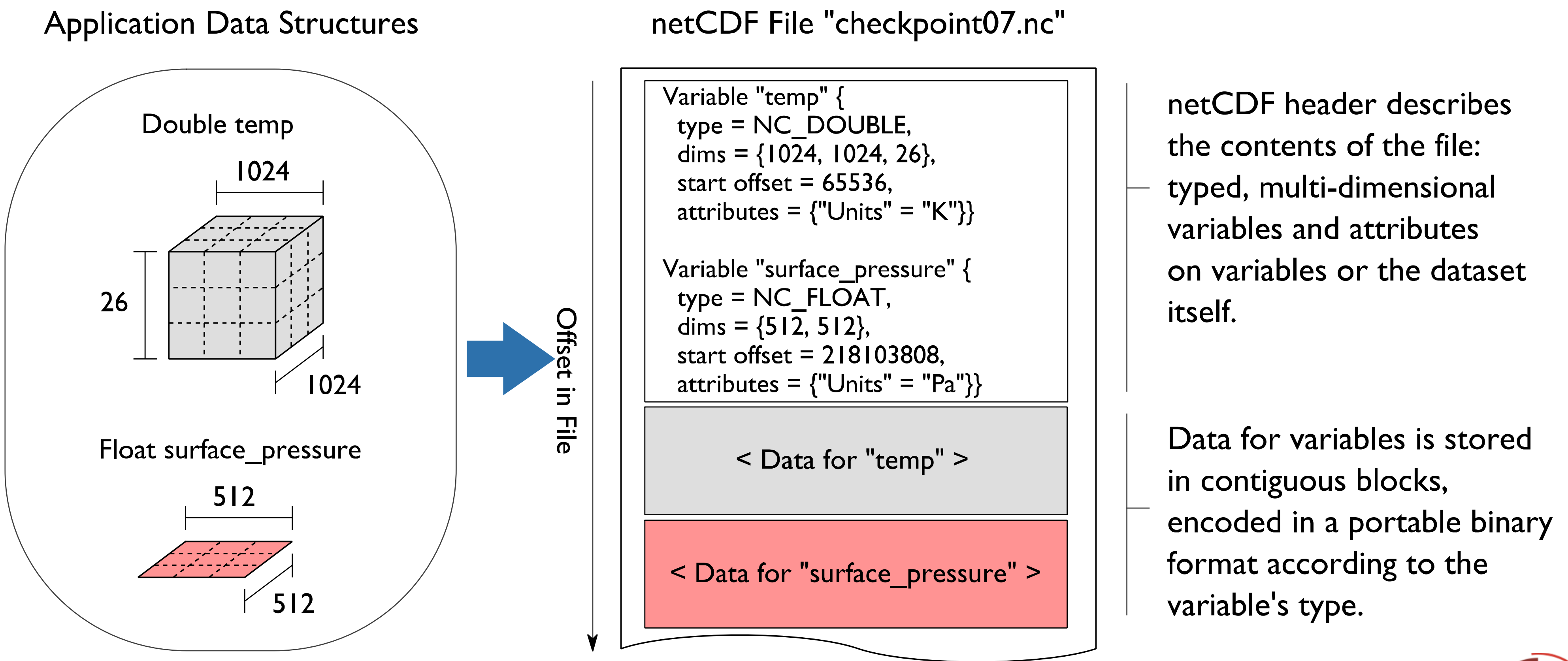
Scale complexity:
Spatial range from the reactor core in meters to fuel pellets in millimeters.

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Example of organizing application data

Application data models are supported via libraries that map down to files (and sometimes directories).



What is unique about HPC I/O?
#4: each HPC facility is different

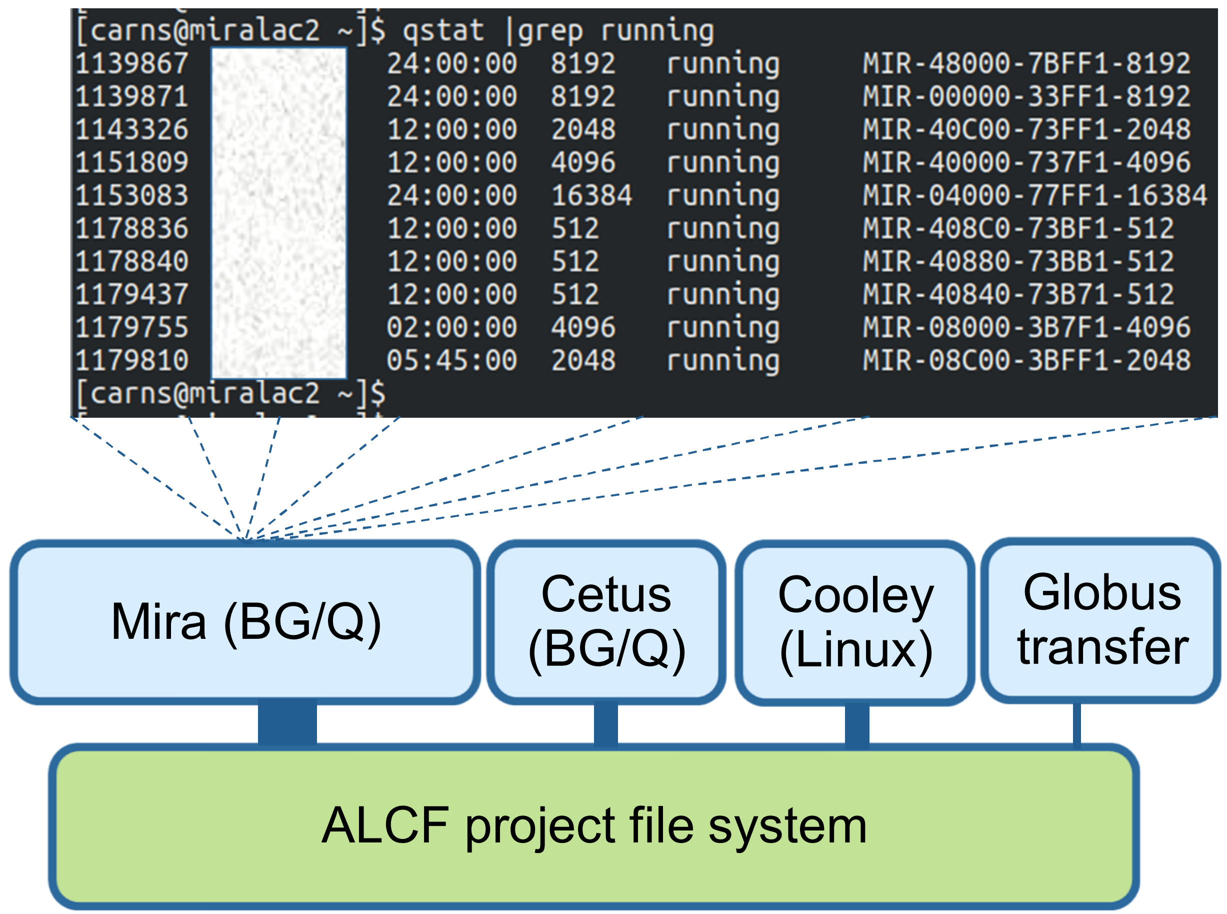
- HPC systems are purpose-built by a handful of different vendors

… and more

- Their storage systems are no different. Major storage platforms in the DOE include GPFS (IBM), Lustre (Intel), PanFS (Panasas), and Datawarp (Cray)
- …In some cases with different hardware integrators, and almost always with different performance characteristics
- Step 4: use portable tools and libraries to handle platform optimizations, learn performance debugging basics (more later)

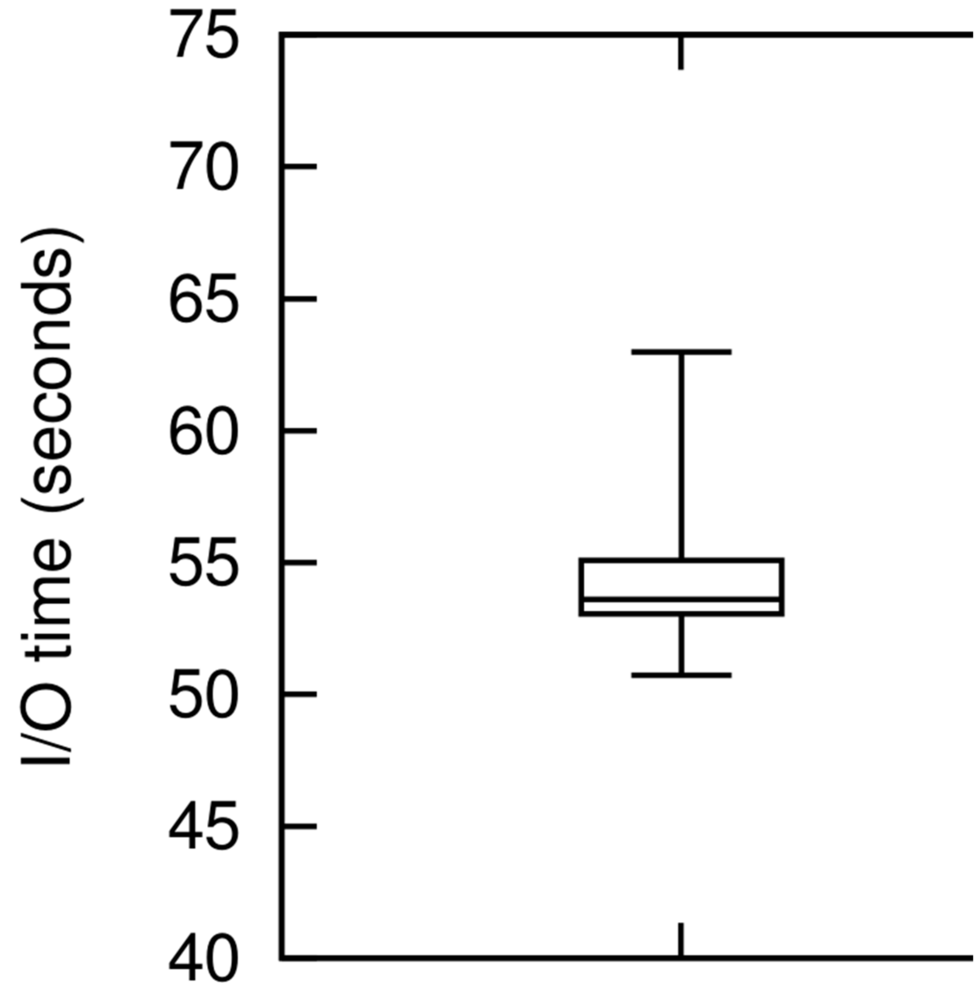
What is unique about HPC I/O?
#5: Expect some performance variability

- **Why:**
  - **Thousands of hard drives never perform perfectly at the same time**
  - **You are sharing storage with many other users**
  - **Not just computation jobs, but remote transfers, tape backups, etc.**
  - **The storage is shared with multiple systems**
- **Some performance variance is normal**



What is unique about HPC I/O?
#5: Expect some performance variability

- **Step 5: when measuring I/O performance, take multiple samples and/or look for trends over time**
- **Example shows 15 samples of I/O time from a 6,000 process benchmark on Edison system, with a range of 51 to 63 seconds**



Putting it all together for HPC I/O happiness

1. Check site documentation to find appropriate storage resources
2. Move big data in parallel, and avoid waiting for individual small operations
3. Use I/O libraries that are appropriate for your data model
4. Rely on existing tools for optimizations, and learn how to do some basic performance debugging
5. Be aware that sometimes performance fluctuates for reasons that you cannot control

HOW IT WORKS: TODAY’S I/O SYSTEMS

An example system: Mira (ALCF)

- Mira is the flagship HPC system at Argonne National Laboratory
- 48 racks
- 786,432 processors
- 768 terabytes of memory

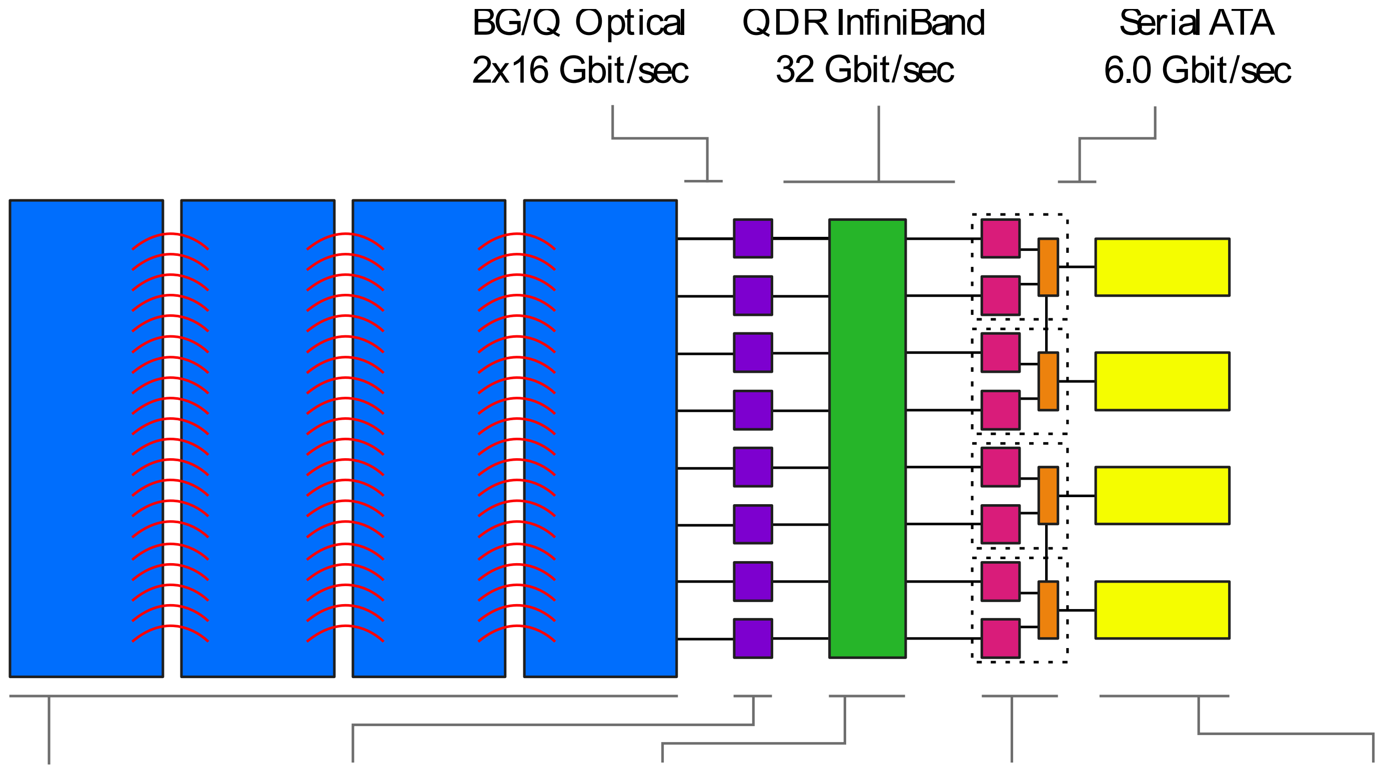
“Mira is 20 times faster than Intrepid, its IBM Blue Gene/P predecessor”

Argonne NATIONAL LABORATORY
ECP EXASCALE COMPUTING PROJECT

Mira storage system

- 384 I/O nodes (relay file system operations from compute nodes to the storage system)
- 3024 port InfiniBand switch complex
- Largest file system:
  - 16 DDN storage systems
  - 8,960 SATA disks
  - 512 SSDs
  - 12 PiB formatted storage
  - 240 GiB/s performance

Mira storage hardware layout



Compute nodes run applications and some I/O middleware.

768K cores with 1 Gbyte of RAM each

Gateway nodes run parallel file system client software and forward I/O operations from HPC clients.

384 16-core PowerPC A2 nodes with 16 Gbytes of RAM each

Commodity network primarily carries storage traffic.

QDR Infiniband Federated Switch

Storage nodes run parallel file system software and manage incoming FS traffic from gateway nodes.

SFA12KE hosts VM running GPFS servers

Enterprise storage controllers and large racks of disks are connected via InfiniBand.

32 DataDirect SFA12KE; 560 3 Tbyte drives + 32 200 GB SSD; 16 InfiniBand ports per pair

Reviewing the data access path (conceptual)

A simple example:

Application	In-memory Data Model
On-disk Formatting	
Files (POSIX)	
I/O Hardware	

Logical (data model) view of data access.

* We're simplifying the story here somewhat …

What really happens on Mira

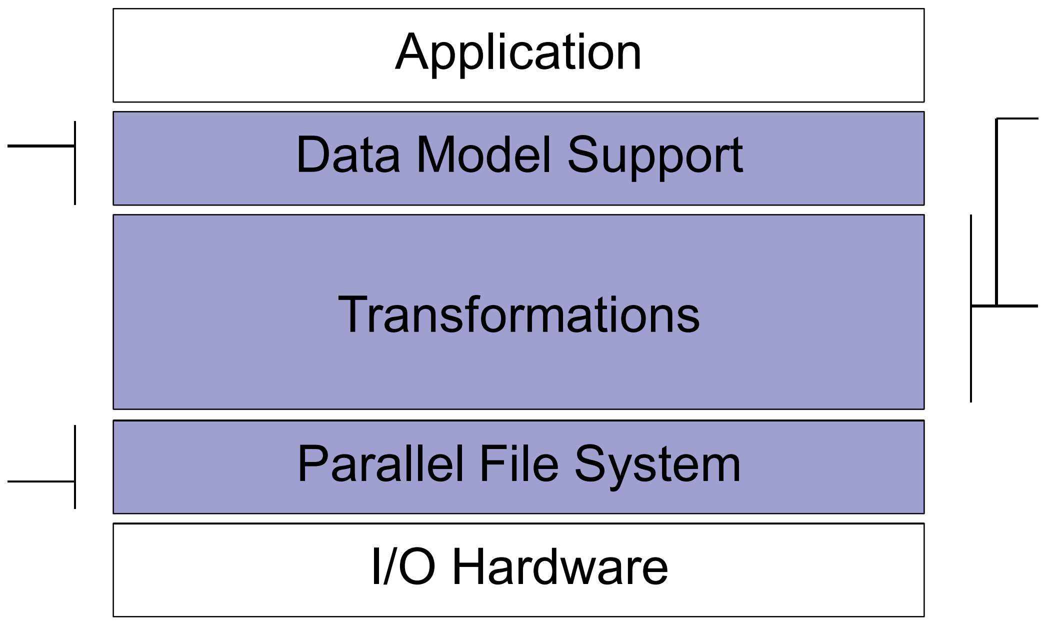
The software used to provide data model support and to transform I/O to better perform on today's I/O systems is often referred to as the *I/O stack.*

Data Model Libraries map application abstractions onto storage abstractions and provide data portability.

HDF5, Parallel netCDF, ADIOS

Parallel file system maintains logical file model and provides efficient access to data.

GPFS



I/O Middleware organizes accesses from many processes, especially those using collective
I/O.

MPI-IO

I/O Forwarding transforms I/O from many clients into fewer, larger request; reduces lock contention; and bridges between the HPC system and external storage.

IBM ciod

What really happens on Mira

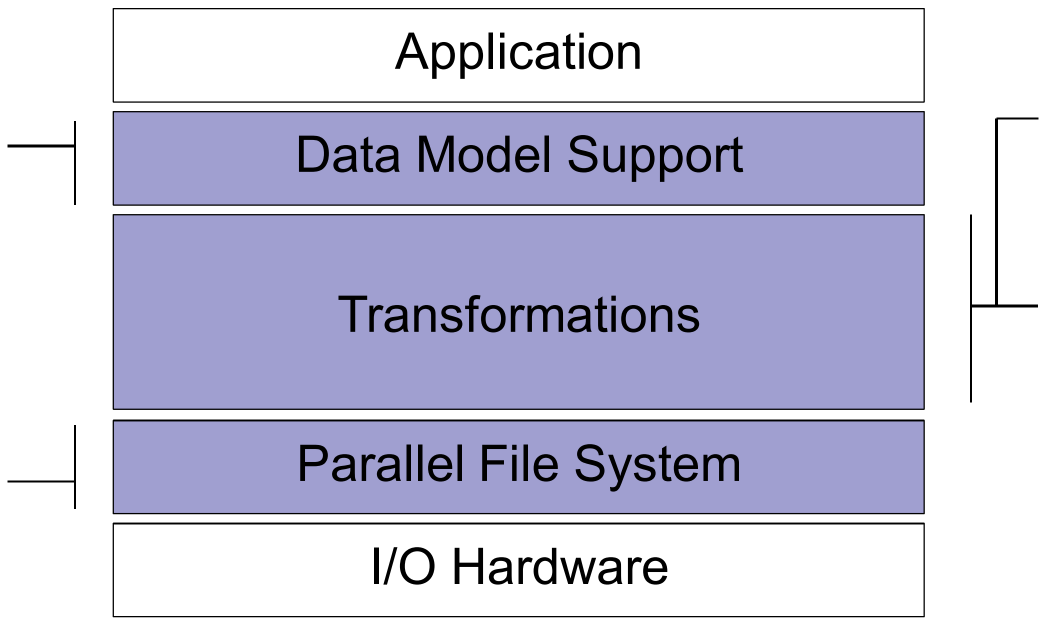
The I/O stack has a lot of software components (not to mention hardware), but data model libraries can protect applications most of this complexity

Data Model Libraries map application abstractions onto storage abstractions and provide data portability.

HDF5, Parallel netCDF, ADIOS

Parallel file system maintains logical file model and provides efficient access to data.

GPFS



I/O Middleware organizes accesses from many processes, especially those using collective
I/O.

MPI-IO

I/O Forwarding transforms I/O from many clients into fewer, larger request; reduces lock contention; and bridges between the HPC system and external storage.

IBM ciod

What about Theta?

Key parts of the software and hardware stack are different

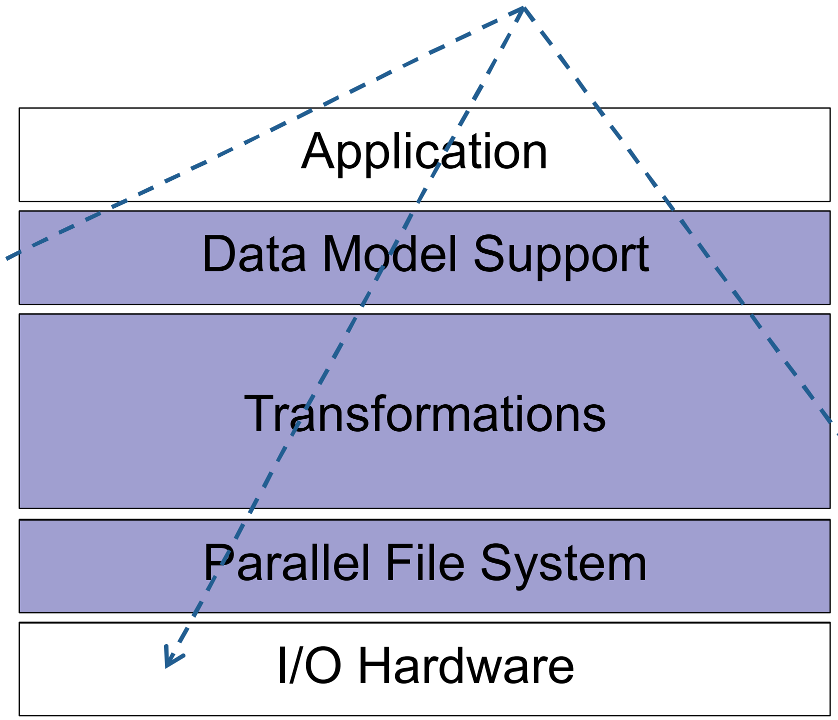
Different optimizations are needed to account for block sizes, storage device types, locking algorithms, etc.

Data Model Libraries map application abstractions onto storage abstractions and provide data portability.

HDF5, Parallel netCDF, ADIOS

Parallel file system maintains logical file model and provides efficient access to data.

Lustre



I/O Middleware organizes accesses from many processes, especially those using collective
I/O.

MPI-IO

I/O Forwarding transforms I/O from many clients into fewer, larger request; reduces lock contention; and bridges between the HPC system and external storage.

Lnet routers

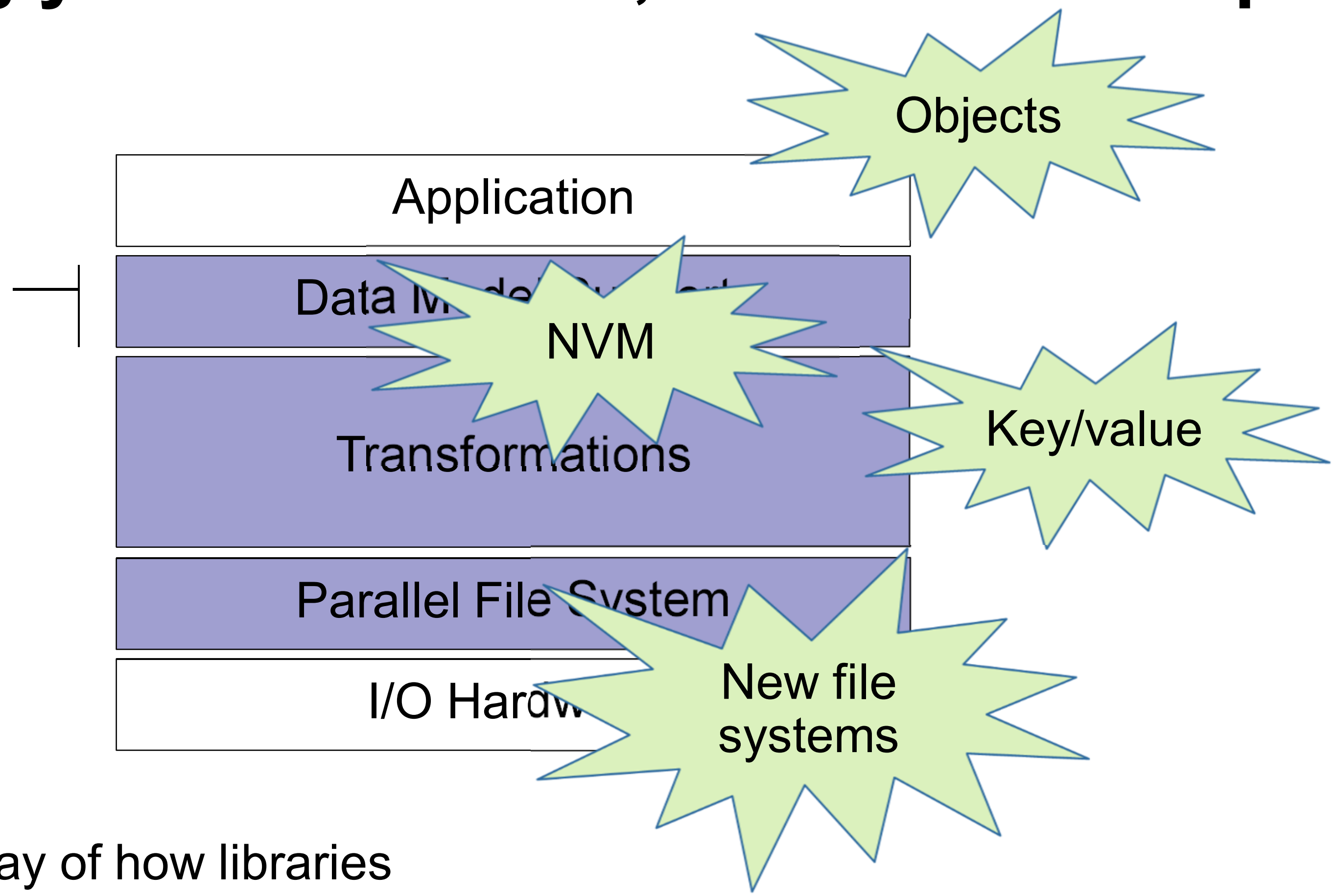
The high level library APIs used by applications are still the same, though!

What about the future?

Choosing the right libraries and interfaces for your application isn't just about fitting your data model, but also future-proofing your application.

Data Model Libraries map application abstractions onto storage abstractions and provide data portability.

HDF5, Parallel netCDF, ADIOS



We'll see examples later in the day of how libraries are adapting to storage technology.

Next up!

- This presentation covered general principles of HPC I/O and how to use it
- The next presentation will go into more detail on “I/O transformations”: how your data path can be tuned to traverse an HPC storage system more effectively.