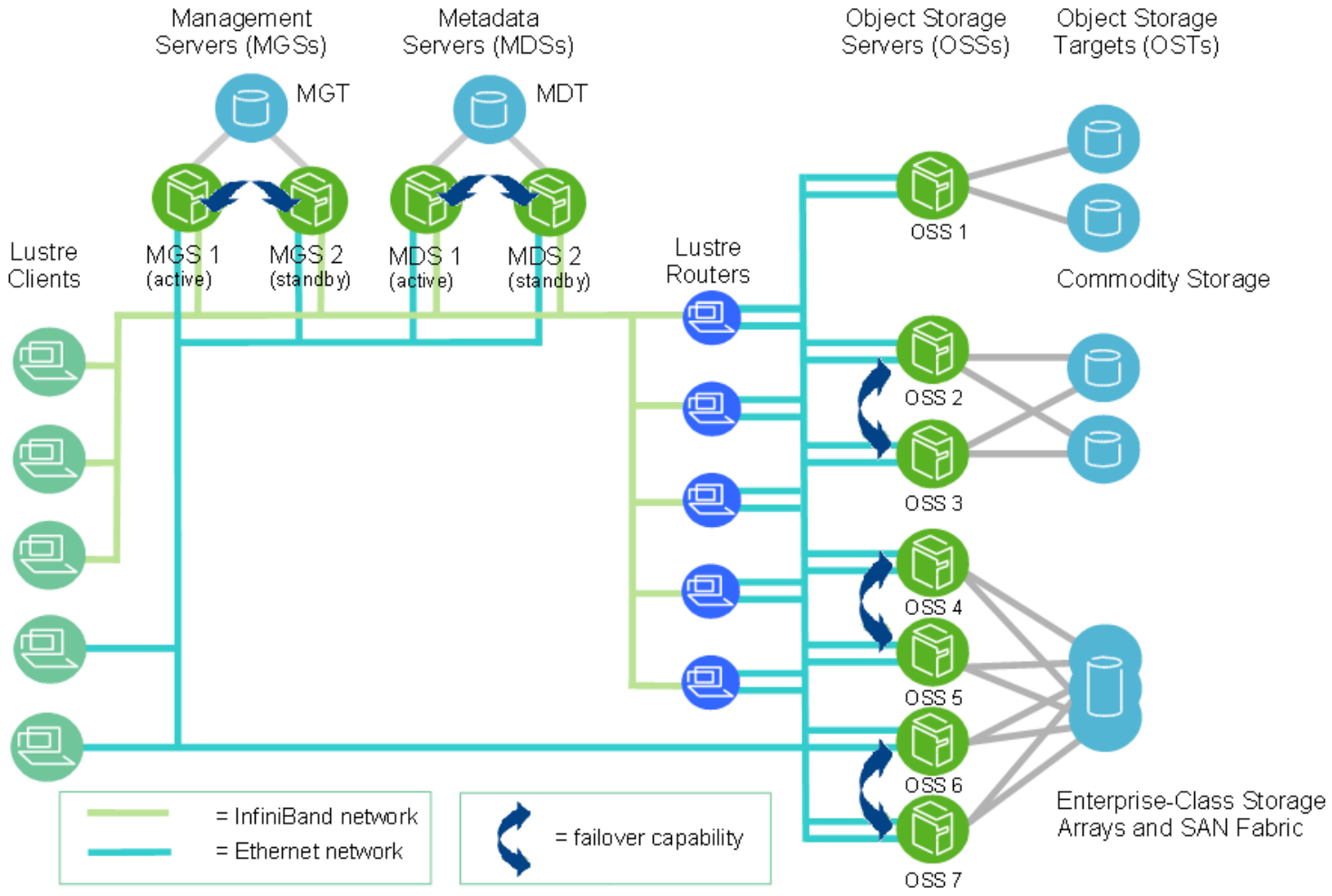
OPENFABRICS ALLIANCE

Update on Lustre*, OpenSFS and FastForward

Doug Oucharek

Intel® High Performance Data Division

Lustre: Overview



Lustre: Ongoing Projects

- Addressing MDS Bottleneck (OpenSFS + Intel)
  - SMP Affinity in LNet (2.3)
  - ptlrpc improvements for locks and threading (2.3)
  - Distributed Namespace (DNE) (phase 1: 2.4, ongoing)
- Cloud/Enterprise Enhancements (Intel)
  - Hadoop Optimizations
- Consistency (OpenSFS + Intel)
  - LFSCK (2.3, 2.4, ongoing)

Lustre: Ongoing Projects

- Performance
  - Network Request Scheduler (2.4) (Intel + Xyratex)
  - 4MB Bulk RPC (2.4) (Xyratex)
- More flexible backend support (Intel + LLNL)
  - ZFS Support (2.4, ongoing)
- Archiving
  - HSM (CEA + Intel) (2.4, ongoing)

Lustre: Upcoming Projects

- Improve Small File I/O
  - Data on MDS (multi-phase project)
- OSS Striping
  - Replication/Migration (multi-phase project)
- Cloud/Enterprise Enhancements
  - Improve Target to NID mapping (proposed)
- Easier/More Flexible Configuration
  - Dynamic LNet Config
  - Channel Bonding (IB or Generic)

Lustre: Upcoming Projects

- Performance
  - Storage Tiers
- Scalable Monitoring
  - LNet Health Network

Exascale I/O technology drivers

	2012	2020
Nodes	10-100K	100K-1M
Threads/node	~10	~1000
Total concurrency	100K-1M	100M-1B
Object create	**100K/s**	**100M/s**
Memory	1-4PB	30-60PB
FS Size	10-100PB	600-3000PB
MTTI	1-5 Days	6 Hours
Memory Dump	< 2000s	< 300s
Peak I/O BW	**1-2TB/s**	**100-200TB/s**
Sustained I/O BW	**10-200GB/s**	**20TB/s**

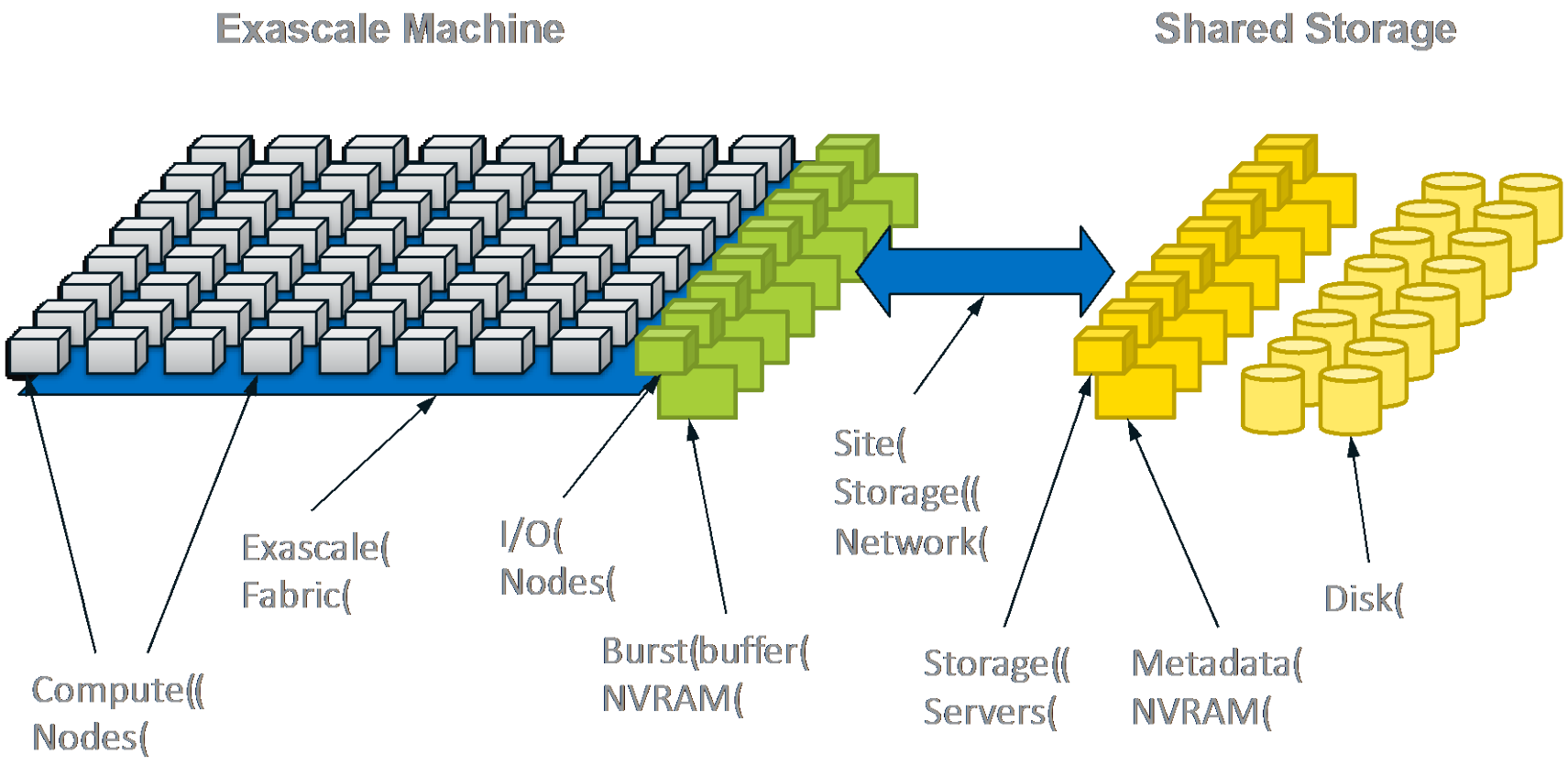
Department of Energy - Fast Forward Challenge

- FastExascale research and development
- Sponsored Forward RFP provided US Government funding for by 7 leading US national labs
- Aims to solve the currently intractable problems of Exascale to meet the 2020 goal of an Exascale machine
- RFP elements were Processor, Memory and Storage
- Whamcloud won the Storage (filesystem) component
  - HDF Group – HDF5 modifications and extensions
  - EMC – Burst Buffer manager and I/O Dispatcher
  - Cray - Test
- Contract renegotiated on Intel acquisition of Whamcloud
  - Intel - Arbitrary Connected Graph Computation
  - DDN - Versioning OSD

Exascale I/O Requirements

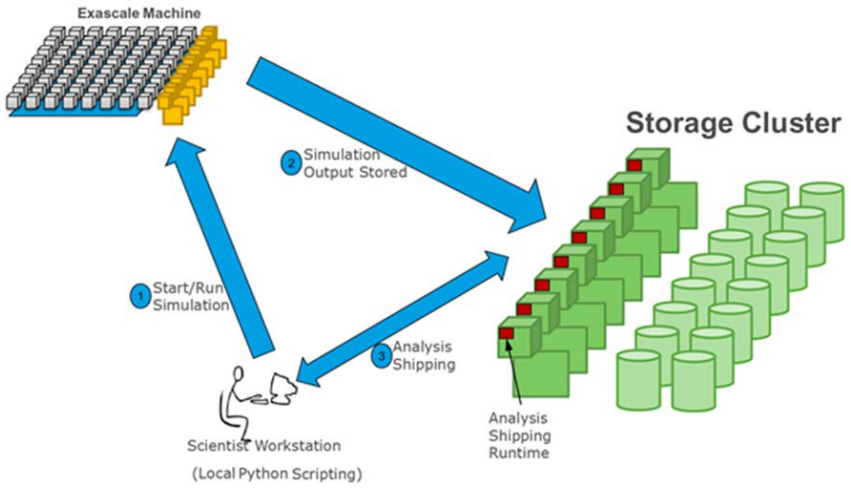
- Constant failures expected at exascale
- Filesystem must guarantee data and metadata consistency
  - Metadata at one level of abstraction is data to the level below
- Filesystem must guarantee data integrity
  - Required end-to-end
- Filesystem must always be available
  - Balanced recovery strategies
  - Transactional models for fast cleanup on failure
  - Scrubbing for repair / resource recovery ok to take days-weeks

Exascale I/O Architecture



Project Goals

- Make storage tool of the scientist
- Move compute to data or data to compute as appropriate
- Provide unprecedented fault tolerance



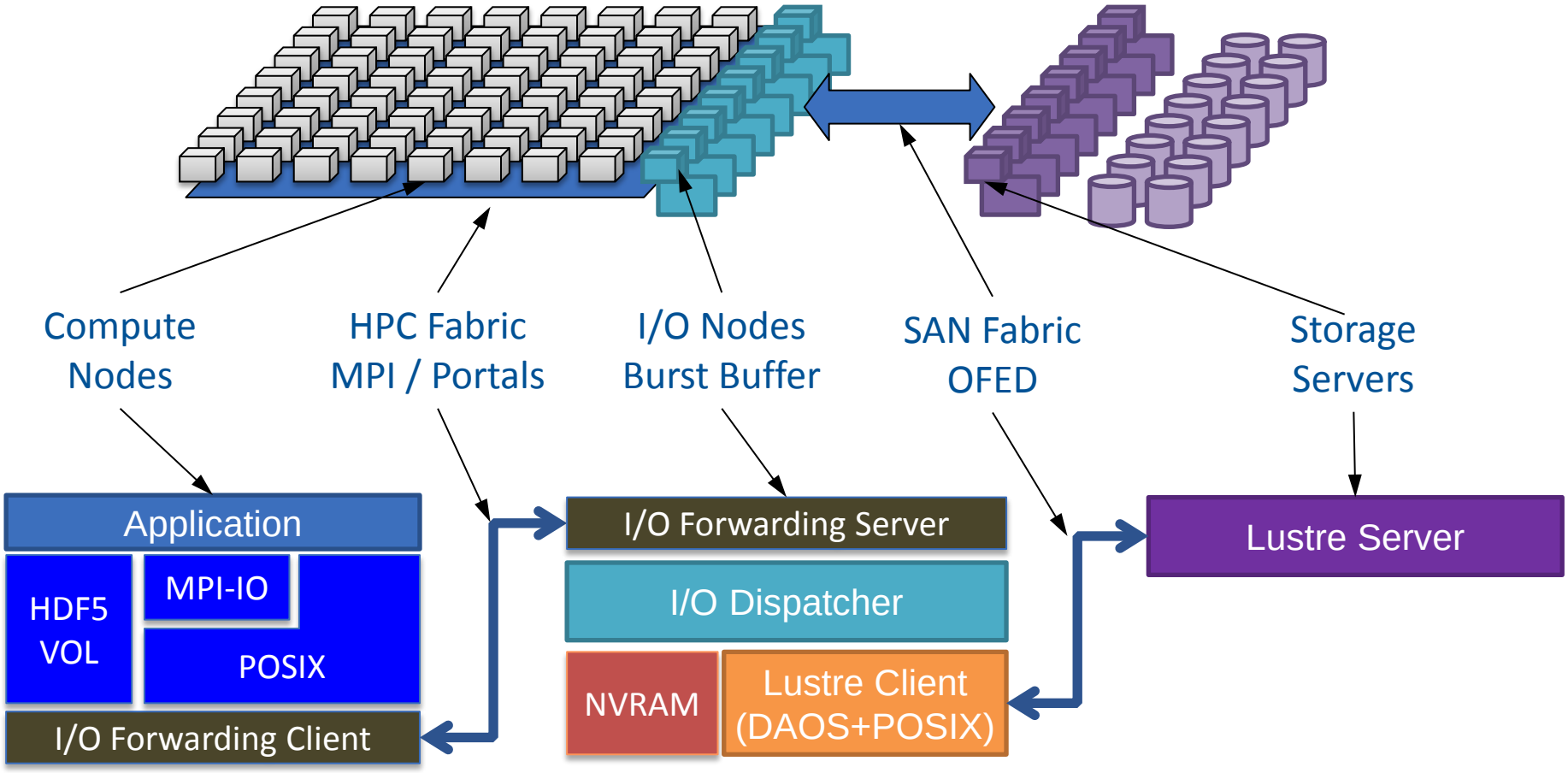
I/O Stack: Features + Requirements

- Non-blocking APIs
  - Asynchronous programming models
- Transactional == consistent thru failure
  - End-to-end application data & metadata integrity
- Low latency / OS bypass
  - Fragmented / Irregular data

I/O Stack: Layered

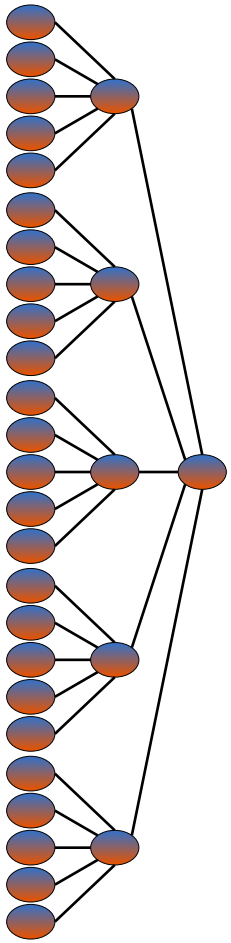
- Application I/O
  - Multiple top-level APIs to support general purpose or application-specific I/O models
- I/O Dispatcher
  - Match conflicting application and storage object models
  - Manage NVRAM burst buffer / cache
- DAOS
  - Scalable, transactional global shared object storage

Fast Forward I/O Architecture



Server Collectives

- Gossip protocols
  - Fault tolerant O(log n) global state distribution latency
  - Peer Discovery
- Tree overlay networks
  - Fault tolerant
    - Collective completes with failure on quorum change
  - Scalable server communications
    - DAOS transaction collectives
    - Collective client eviction
    - Distributed client health monitoring



Thank You