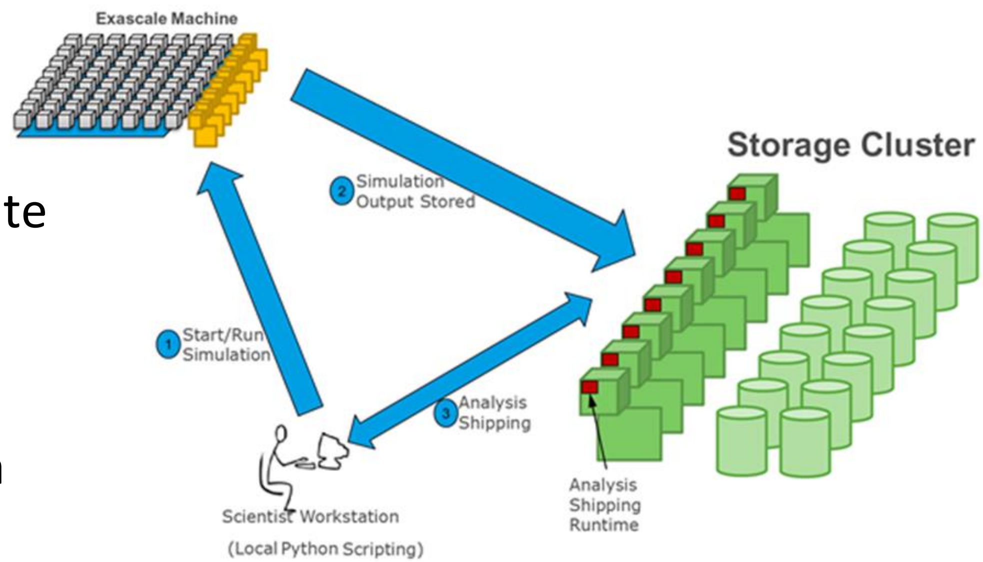
A Vision of Storage for Exascale Computing

Eric Barton

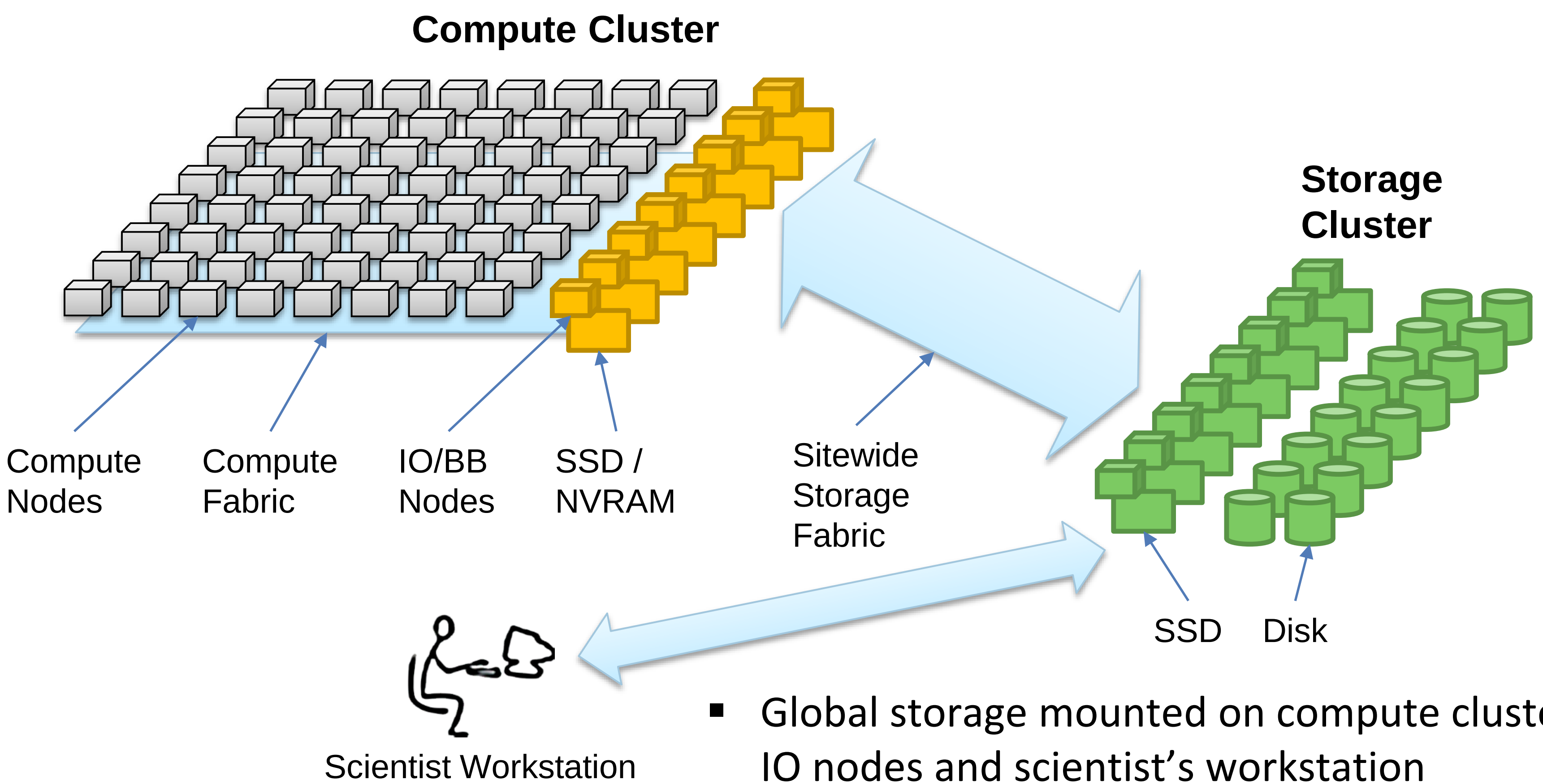
Fast Forward Storage & IO Project Goals

- Make Exascale storage a tool of the Scientist
  - Tractable data management
  - Comprehensive interaction
  - Move compute to data or data to compute as appropriate



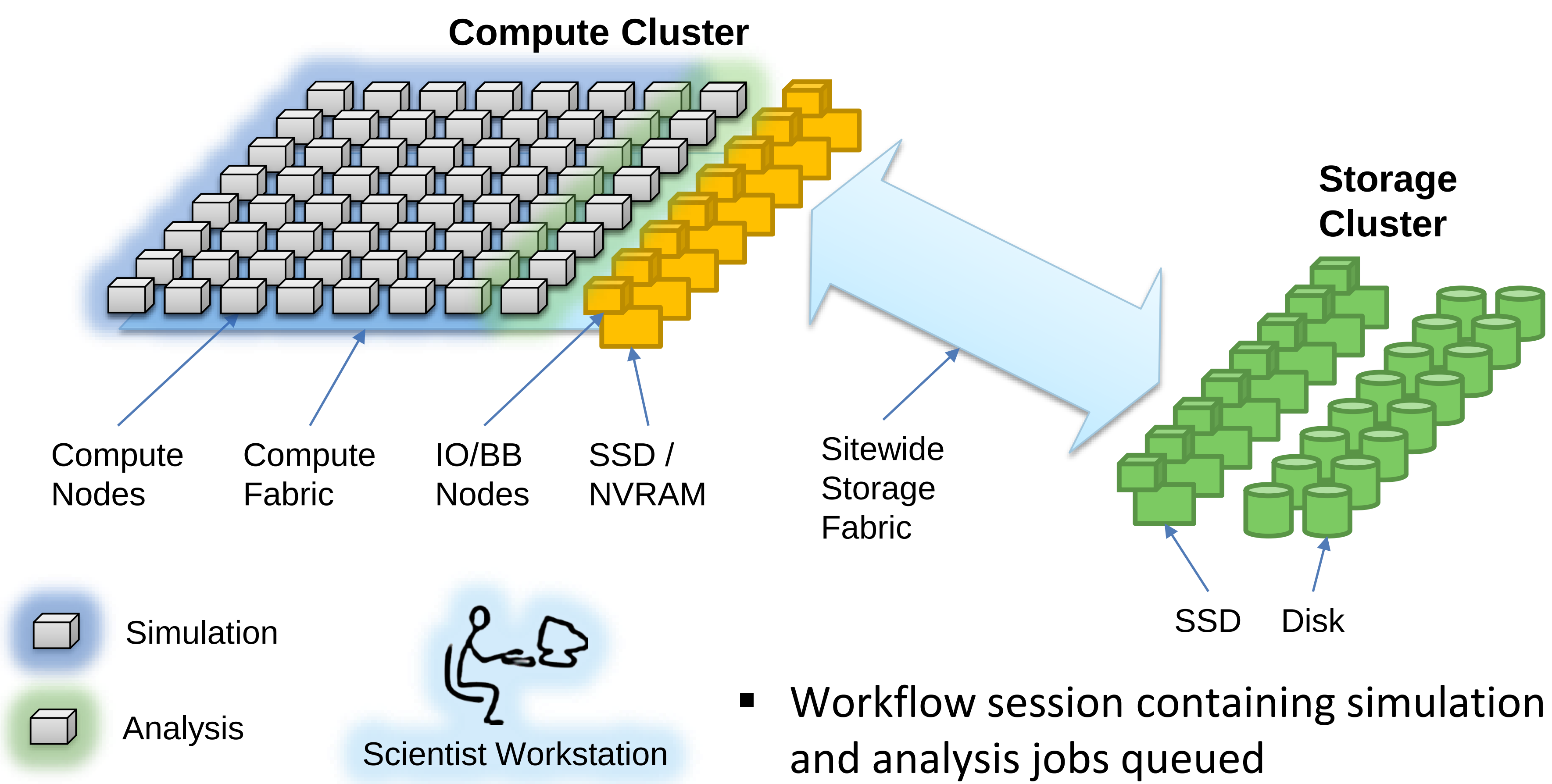
- Overcome today's IO limits
  - Multi-petabyte datasets
  - Explosive growth of metadata
  - Horizontal scaling & jitter
- Support unprecedented fault tolerance
  - Deterministic interactions with failing hardware & software
  - Fast & scalable recovery
  - Enable multiple redundancy & integrity schemes

Fast Forward I/O Architecture



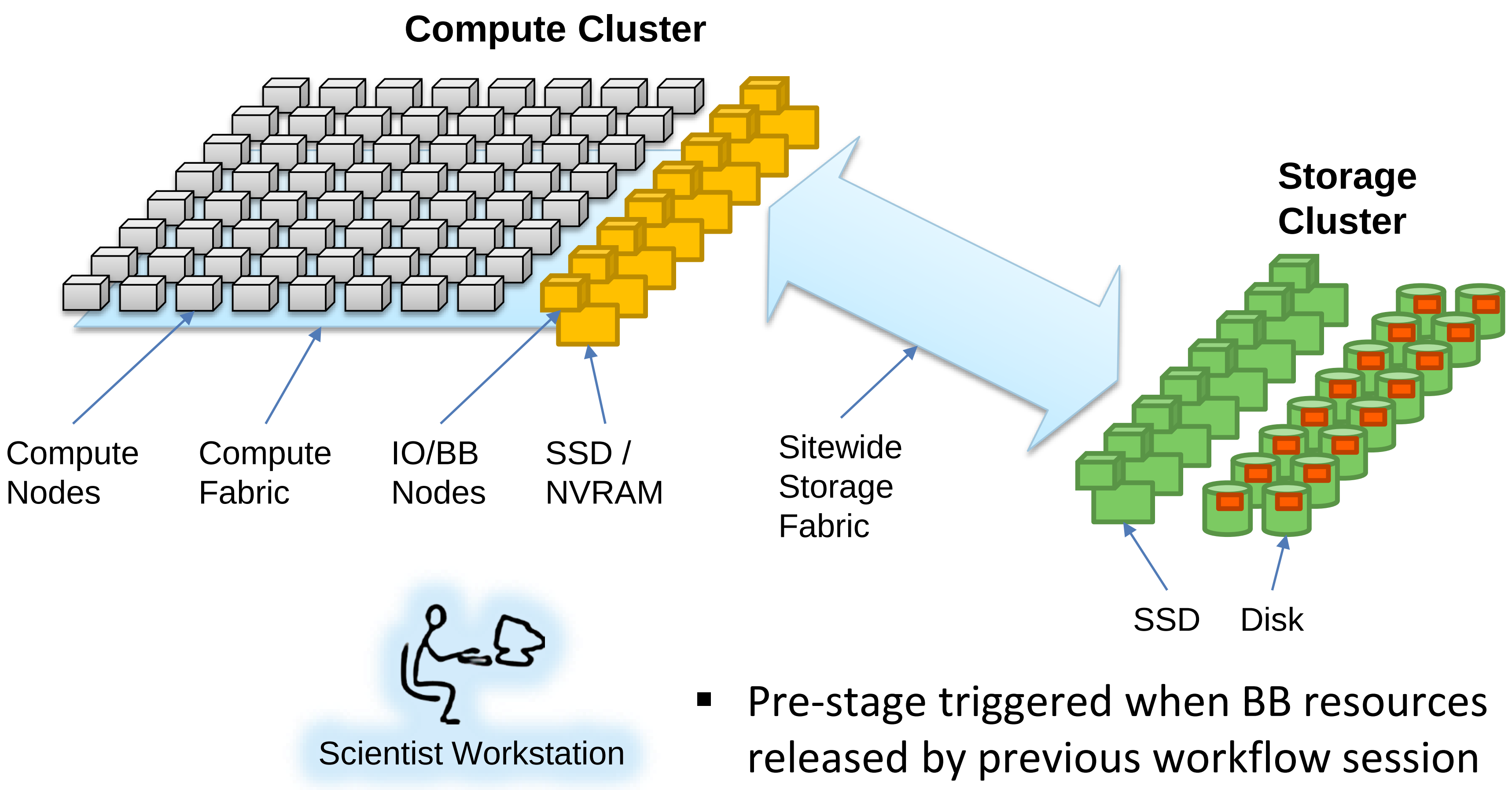
- Global storage mounted on compute cluster IO nodes and scientist's workstation
- I/O forwarding from compute to IO nodes

Workflow: Simulation + In-transit Analysis



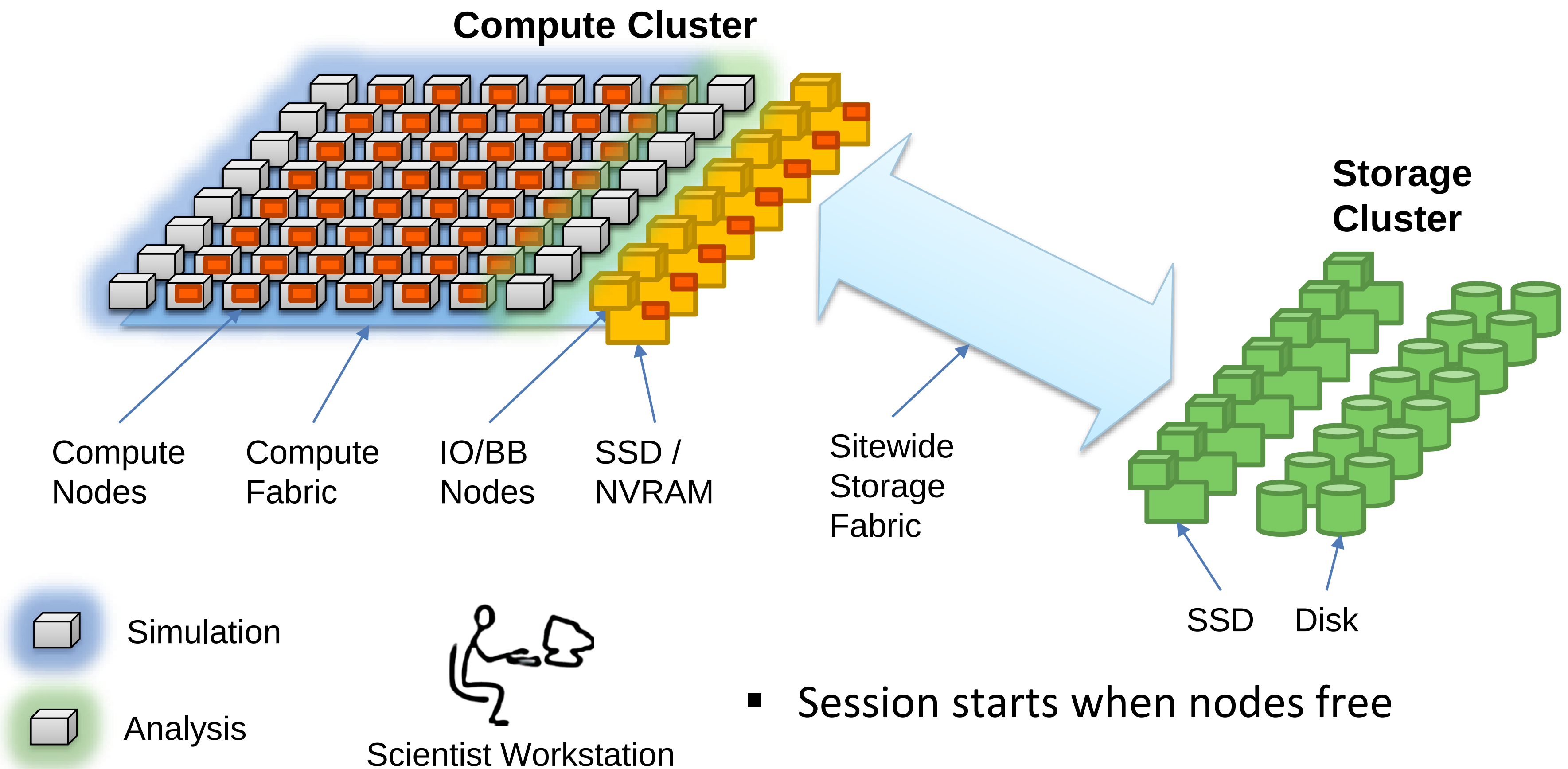
- Workflow session containing simulation and analysis jobs queued

Workflow: Pre-stage to Burst Buffer



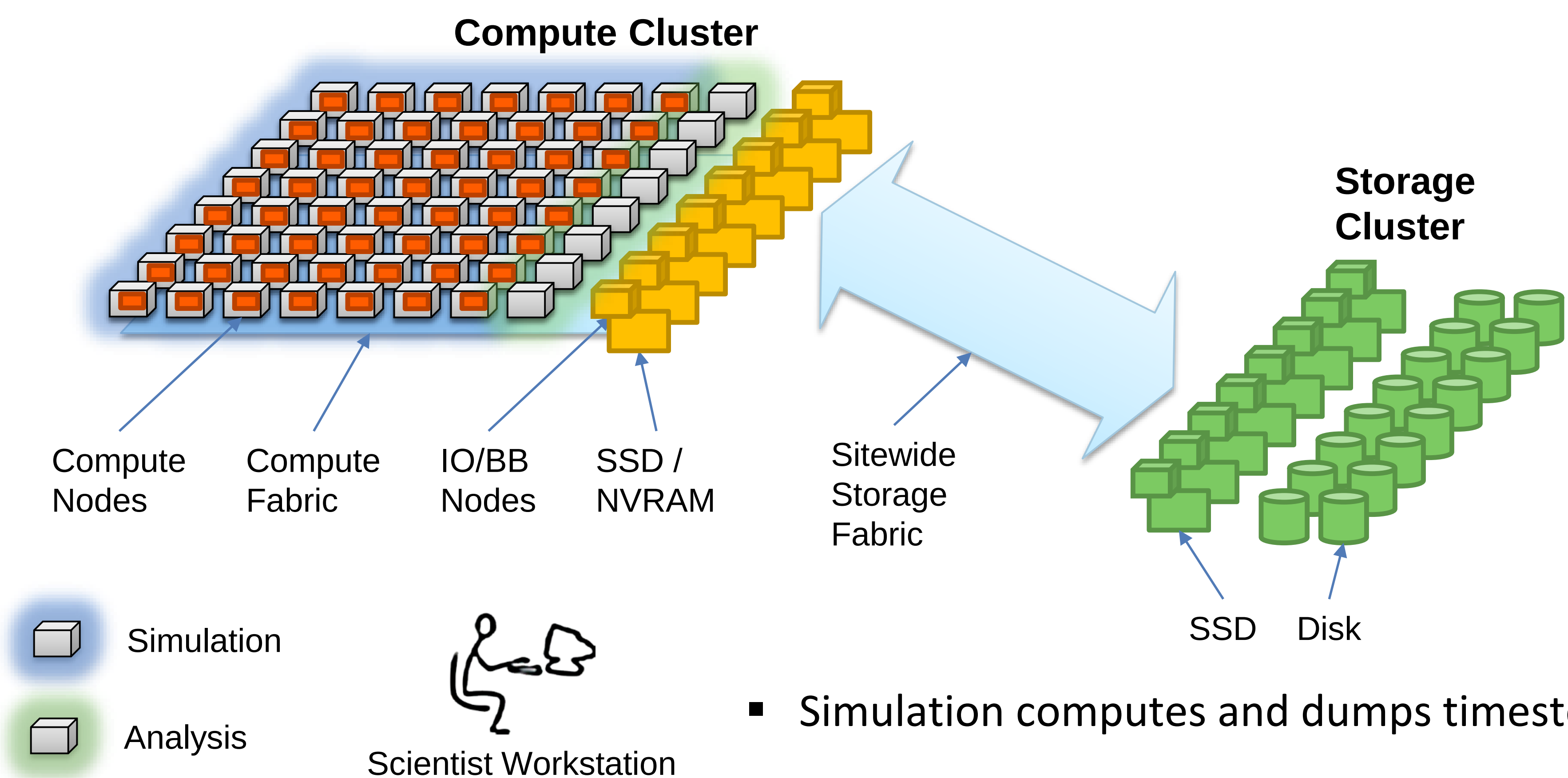
- Pre-stage triggered when BB resources released by previous workflow session

Workflow: Start Session



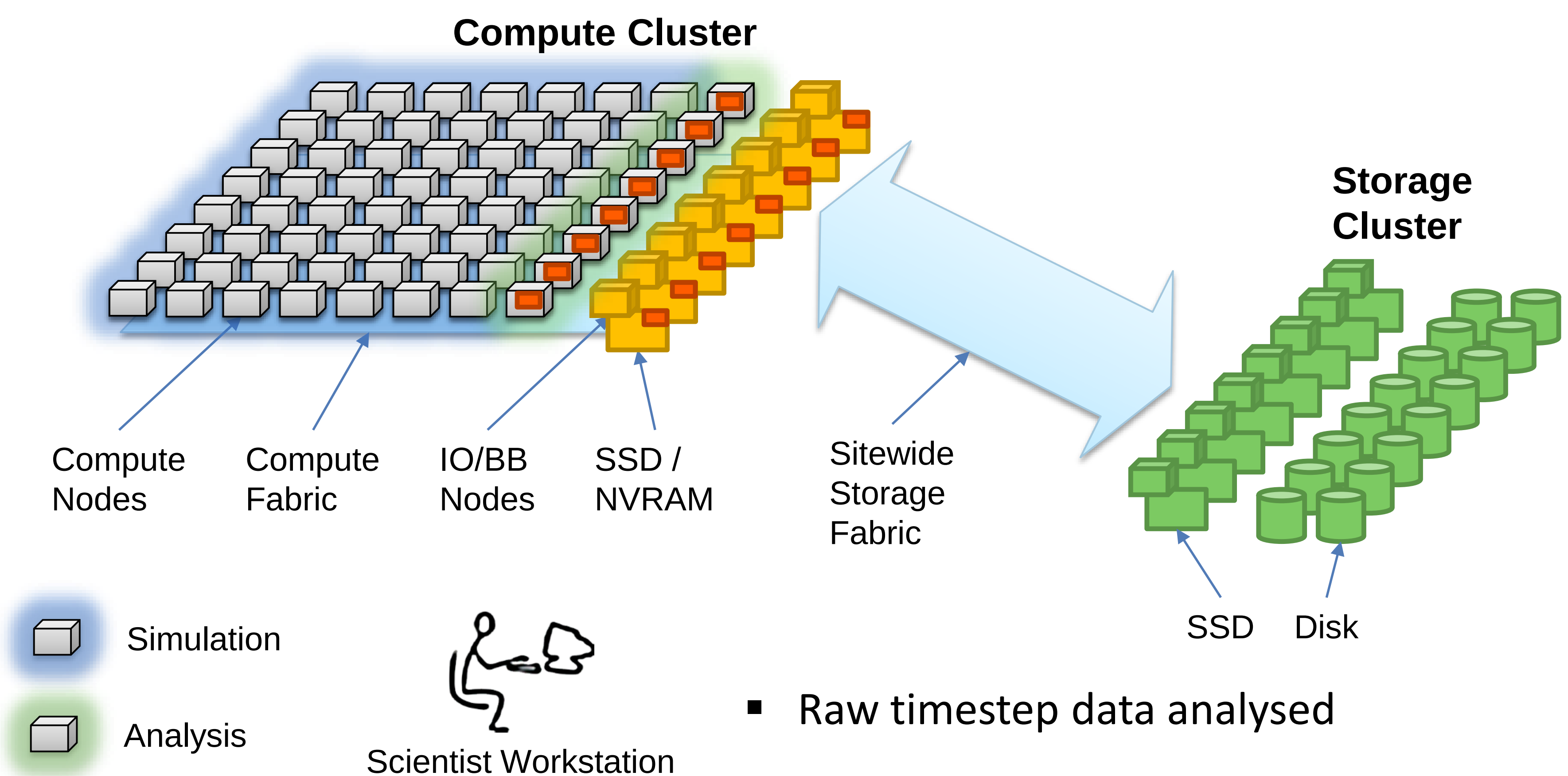
- Session starts when nodes free
- Previous session may still be persisting data from BB to global storage

Workflow: Dump Timestep



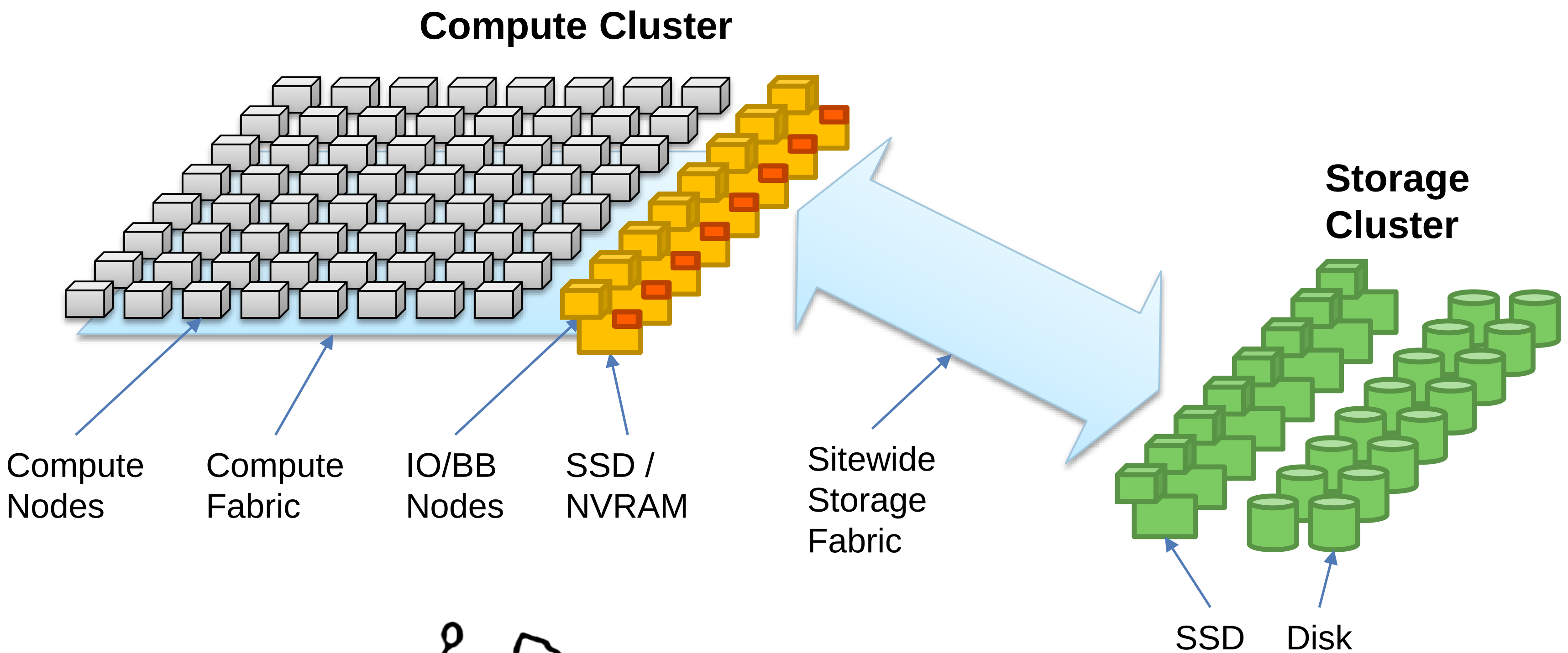
- Simulation computes and dumps timestep

Workflow: In-transit Analysis



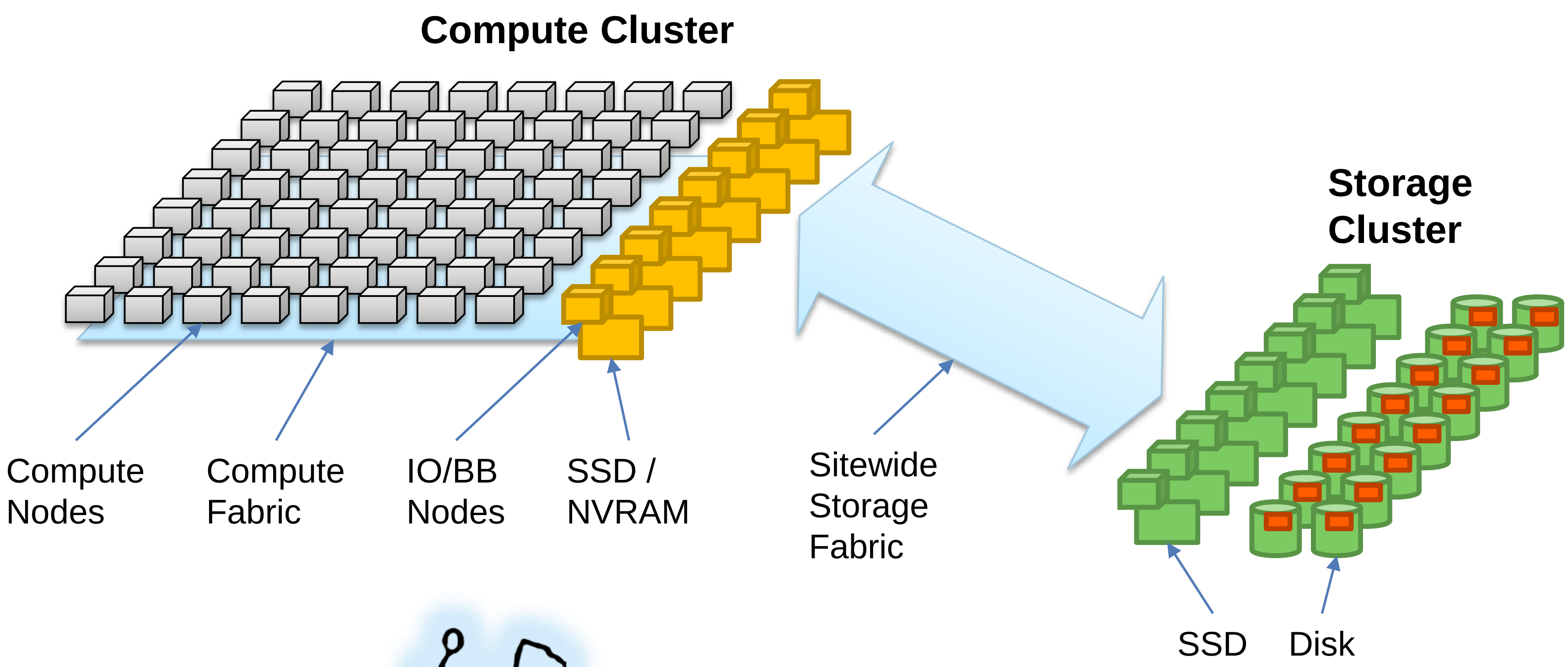
- Raw timestep data analysed
- Analysis data saved to BB
- Raw timestep may be discarded

Workflow: Persist to Global Storage



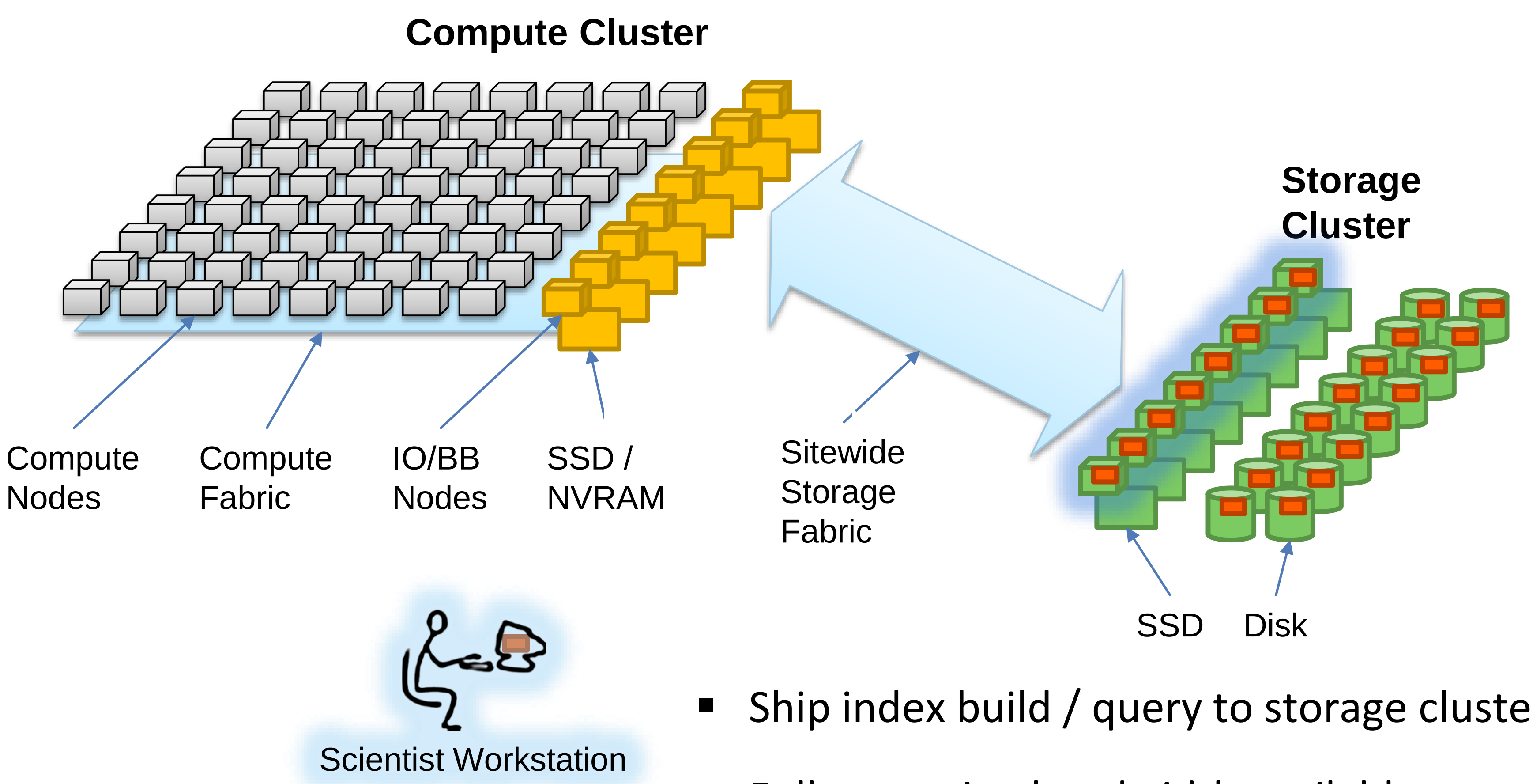
- Analysis output saved to global storage

Workflow: Browse



- Scientist browses simulation output
- Insufficient bandwidth for brute-force query or index build

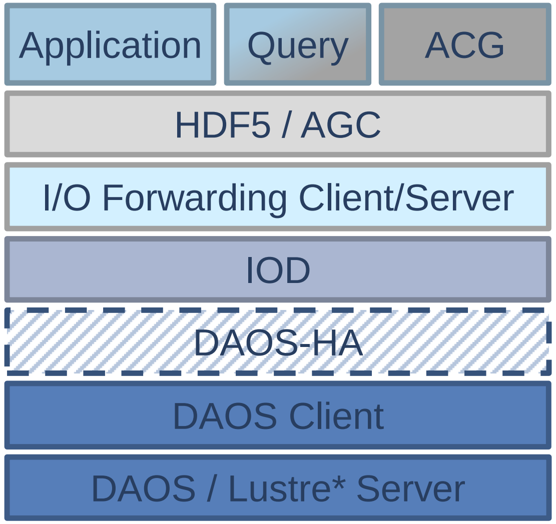
Workflow: Analysis Shipping

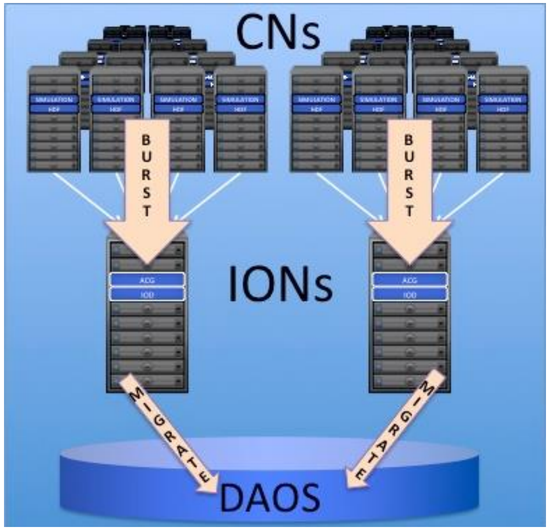


- Ship index build / query to storage cluster
- Full streaming bandwidth available
- Query results returned to workstation

Stackable components

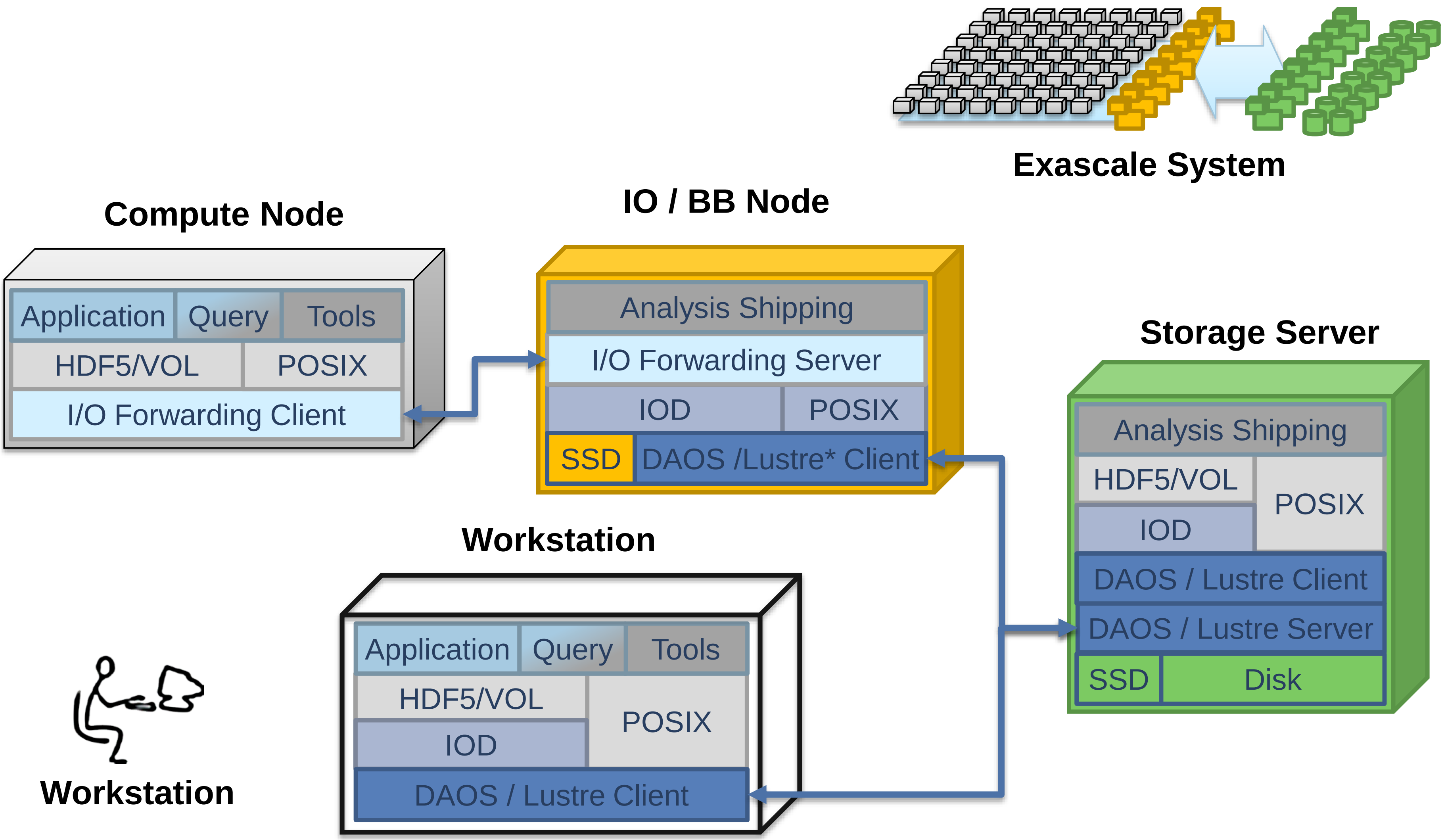
- Application I/O APIs
  - Multiple domain-specific API styles & schemas
  - HDF5+extensions & Graph Computation libraries
- I/O forwarding
  - Keeps top level APIs on Compute Nodes when IOD runs on the Burst Buffer
- I/O Dispatcher (IOD)
  - Impedance match application I/O to storage capabilities
  - Semantic resharding
  - Burst Buffer management
- DAOS-HA
  - High-availability scalable object storage
  - Follow-on project from Fast Forward
- DAOS Containers
  - Virtualized shared-nothing object storage
  - Unpolluted storage system namespace





*other names and brands may be claimed by others

I/O Stack Configurations



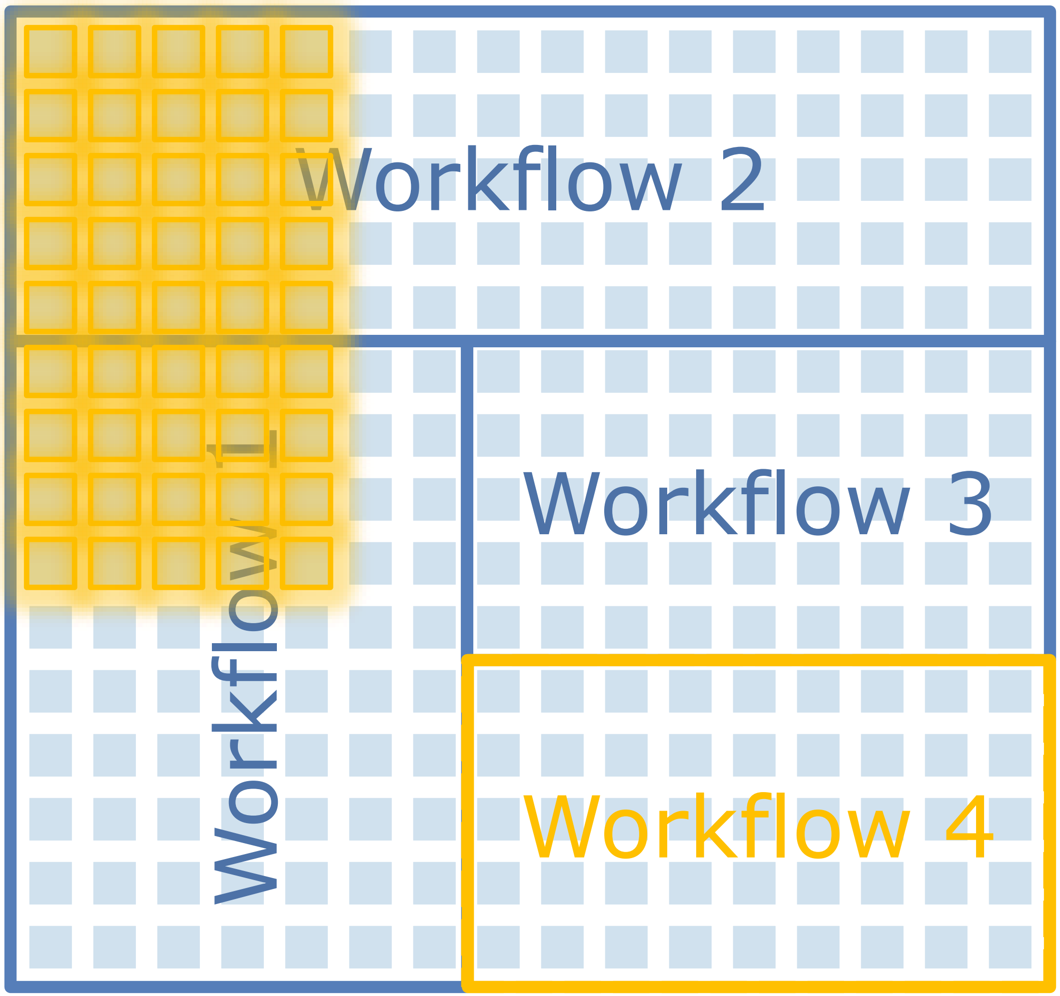
*other names and brands may be claimed by others

Ubiquitous NVRAM

- O(1TB) compute node-local storage
- Instant-on
  - 0 power standby
- Load-store byte-granular access
  - Invites Distributed Persistent Memory programming models
  - Order of magnitude larger in-core working sets
- Storage fully leverages fabric

	Disk	Edge BB	NVRAM
Checkpoint / Search	1 hour	6 minutes	6 seconds
Capacity (# datasets)	30	3-5	10-30

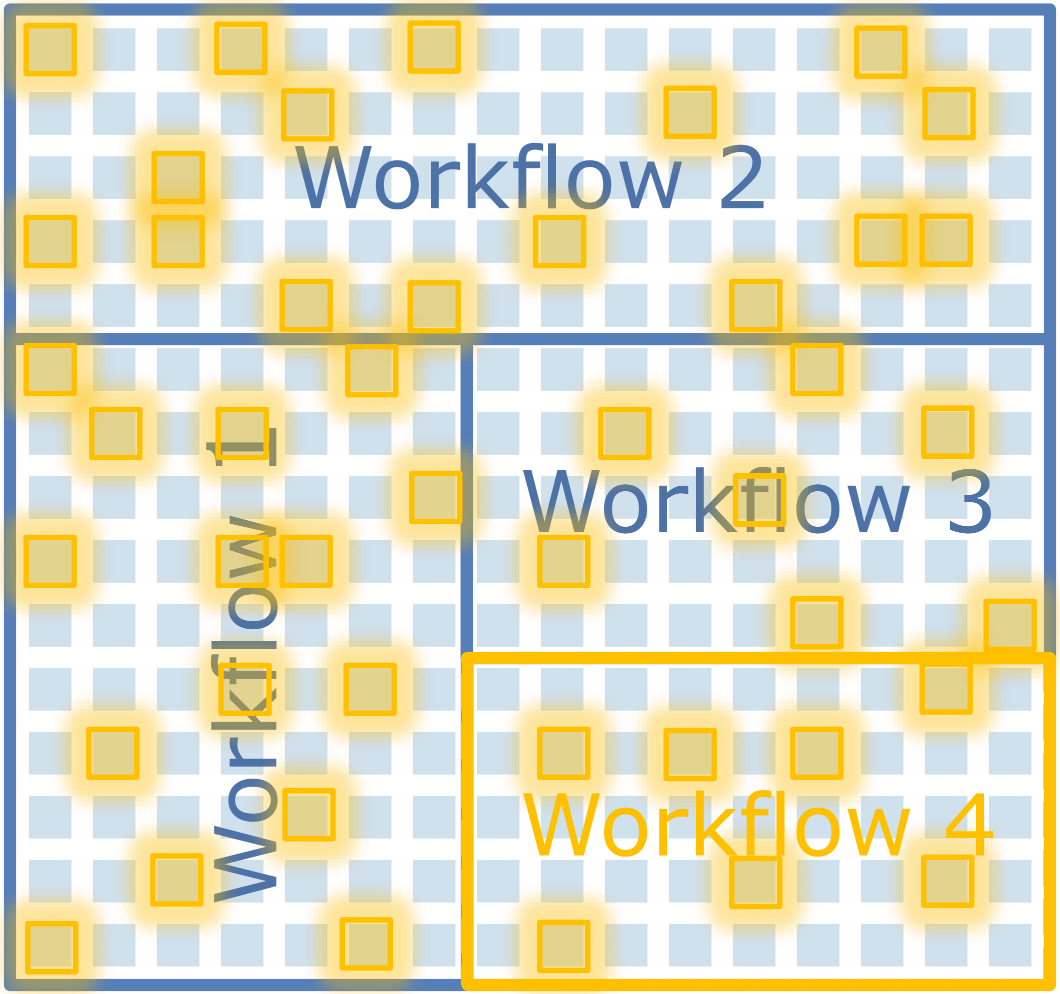
Scheduling Persistent Memory



- Workflow Session 4 ready to run
- Data not local
- Migrate
- Workflow Session 4 started

- Issues
  - Space at destination
  - Comms Interference

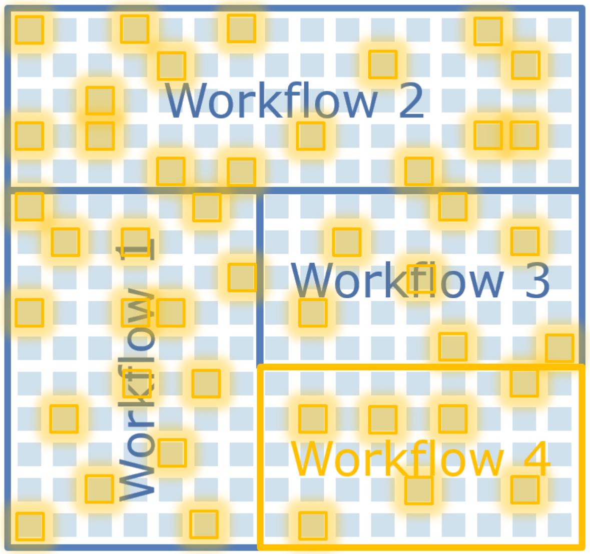
Scheduling Persistent Memory



- Workflow Session 4 ready to run
- Data not local
- Migrate
- Workflow Session 4 started

- Issues
  - Space at destination
  - Comms Interference

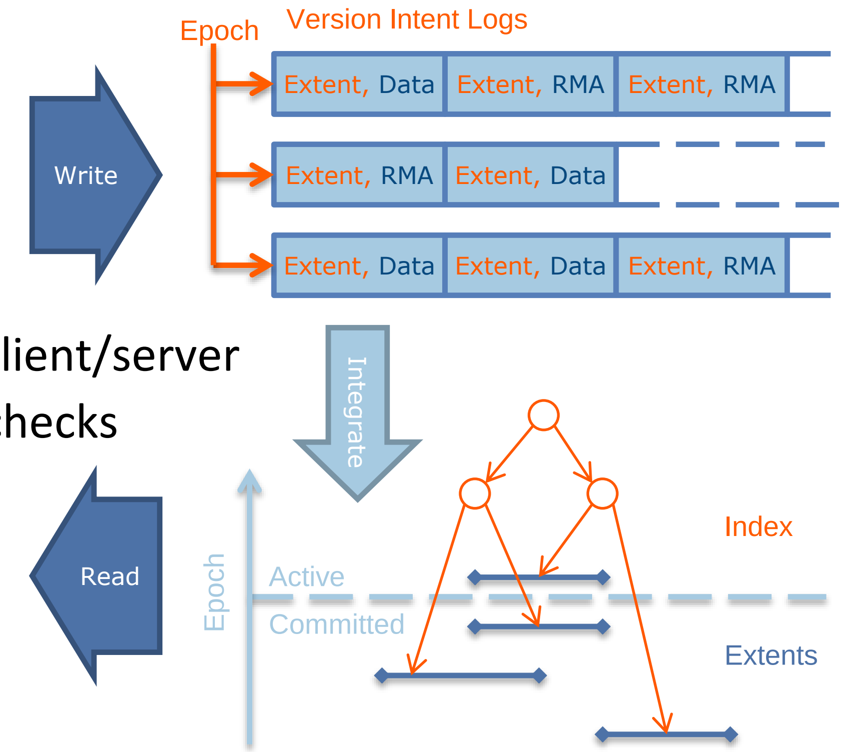
Persistent Memory v. Storage



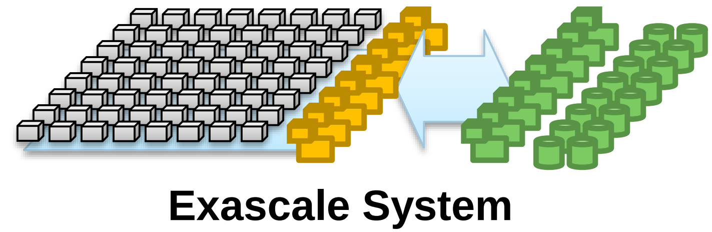
- Persistent Memory is fast but it's…
  - Local to the process using it
  - Inaccessible on node failure
  - Fixed schema
- Storage may be slower but it's…
  - Globally accessible
  - Consistent & durable
  - Snapshotable / Cloneable / Migrateable
- APIs required to…
  - Convert PM ⇔ Storage
    - Persist / Instantiate Distributed Persistent Memory images
    - PM schema conversion
  - Support workflow scheduler integration
    - Data-aware process instantiation
    - Process-aware data migration

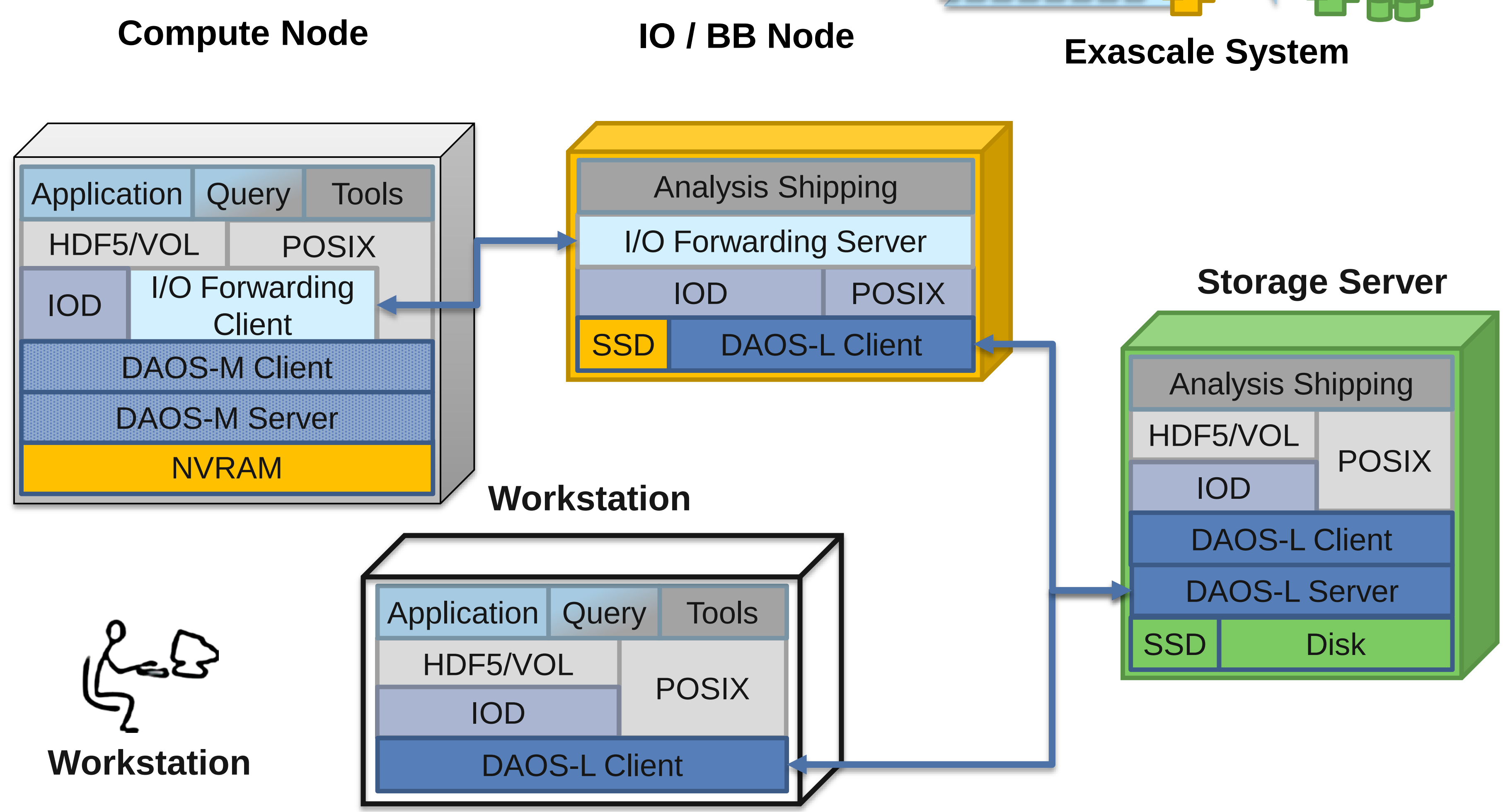
DAOS-M

- Client & Server OS bypass
- Connectionless
  - Peer-to-peer connectivity = ~100x client/server
  - Heavyweight security / ownership checks once on container open
- Memory VOSD
  - PM programming model
    - No block I/O stack latency
    - Byte granular
  - Read
    - Extremely low latency
    - committed writes integrated on index traversal
  - Write
    - Incoming data and metadata logged
    - Integration processes inserts into index



I/O Stack Configurations





Summary

- Ubiquitous NVRAM changes the game
- 3 order of magnitude step change in performance from disk
  - Terabytes/s -> Petabytes/s
  - mS latency -> µS latency
- Workflows will change to exploit
  - Persistent Memory programming models
  - Data aware workflow scheduling
- Storage software must change to exploit
  - Same transactional guarantees required
  - End-to-end OS bypass required
  - Scalable comms/security context establishment
  - More I/O stack configuration flexibility