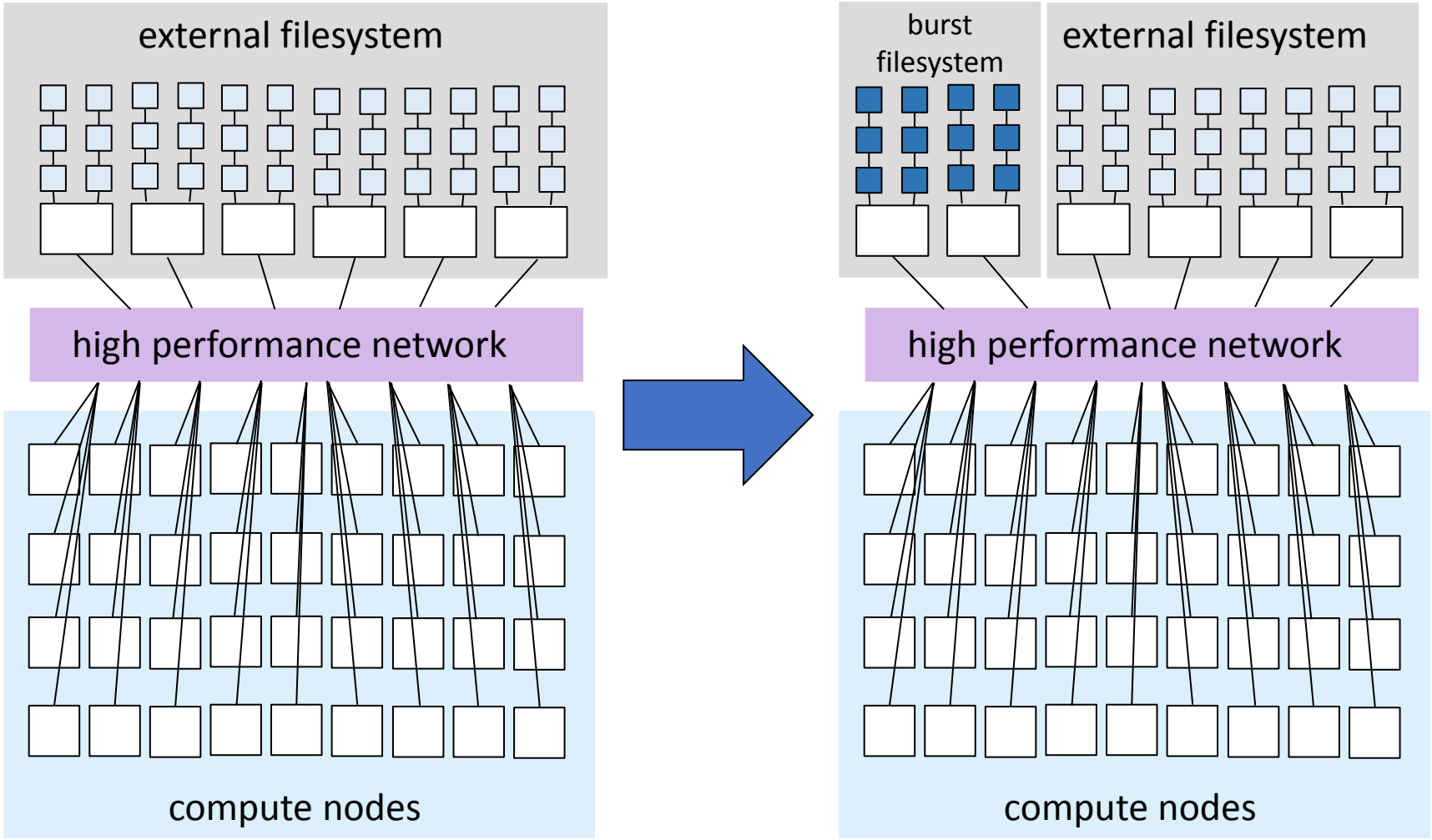
The NEXTGenIO Project

Dr David Henty
d.henty@epcc.ed.ac.uk

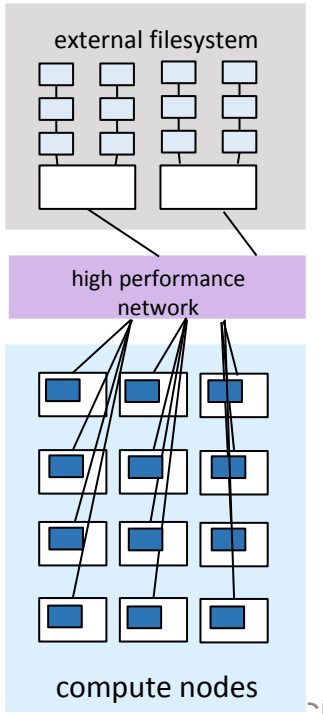
Current trends & approaches

Burst buffer



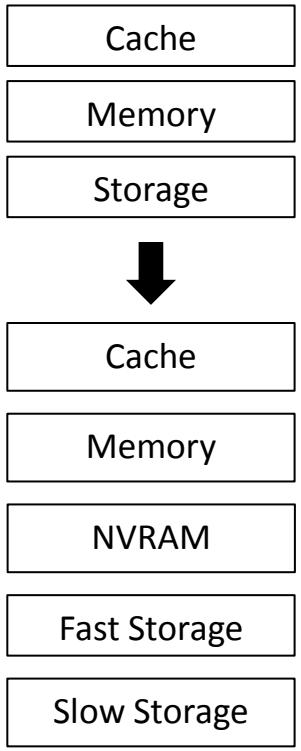
Moving beyond burst buffer

- Non-volatile is coming to the node rather than the filesystem
- Argonne Theta machine has 128GB SSD in each compute node



Non-volatile memory

- Non-volatile RAM
  - Intel's DC Persistent Memory technology is one example
- Much larger capacity than DRAM
  - Hosted in the DRAM slots (DIMM form factor), controlled by a standard memory controller
- Slower than DRAM by a small factor, but significantly faster than SSDs



The NEXTGenIO approach

NEXTGenIO key facts

- FETHPC Research & Innovation Action
- 48 months, 1 year to go
- 8 partners, covering
  - Hardware
  - HPC centres and users
  - Software
  - Tools developers

Our objectives

- Develop a hardware platform prototype
  - Demonstrate applicability of NV memory for both HPC and data centric applications
- Exascale I/O investigation
  - Understanding how best to exploit NVRAM
- Systemware development
  - Producing the necessary software to enable (Exascale) application execution on the hardware platform
- Application co-design
  - Understanding individual application I/O profiles & typical I/O workloads and their requirements

NVRAM properties

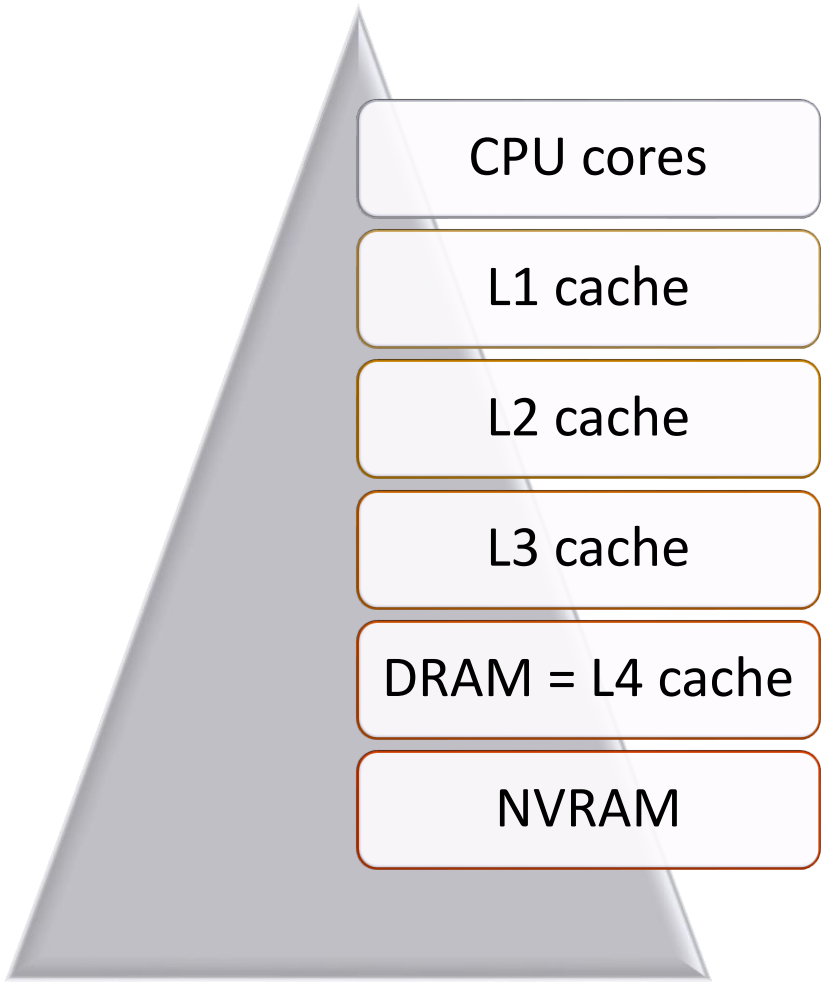
Modes of operation

- **One-level memory**
  - DRAM and NVRAM are two separate memory spaces
    - A bit like Flat Mode on KNL
  - NVRAM is persistent
- **Two-level memory**
  - DRAM acts as cache
    - A bit like Cache Mode on KNL
  - NVRAM is not persistent in this case

☛ Reboot required to change between modes

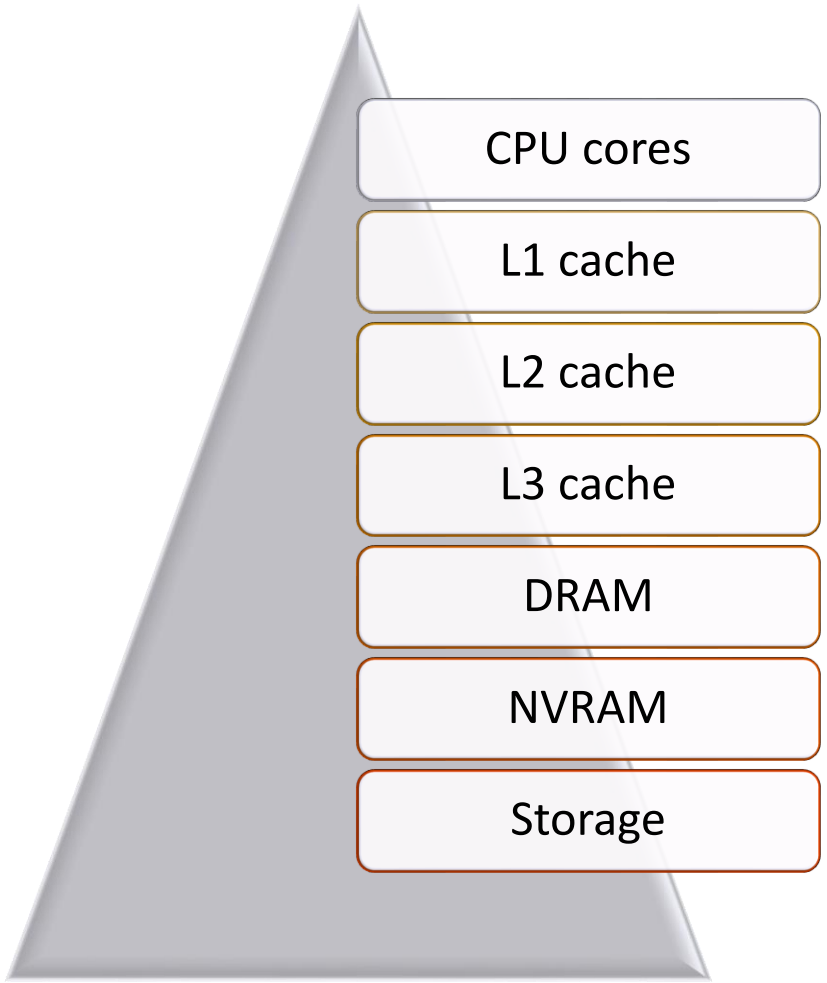
☛ Possible to partition NVRAM and create “app direct” space for two-level memory

Two-level memory



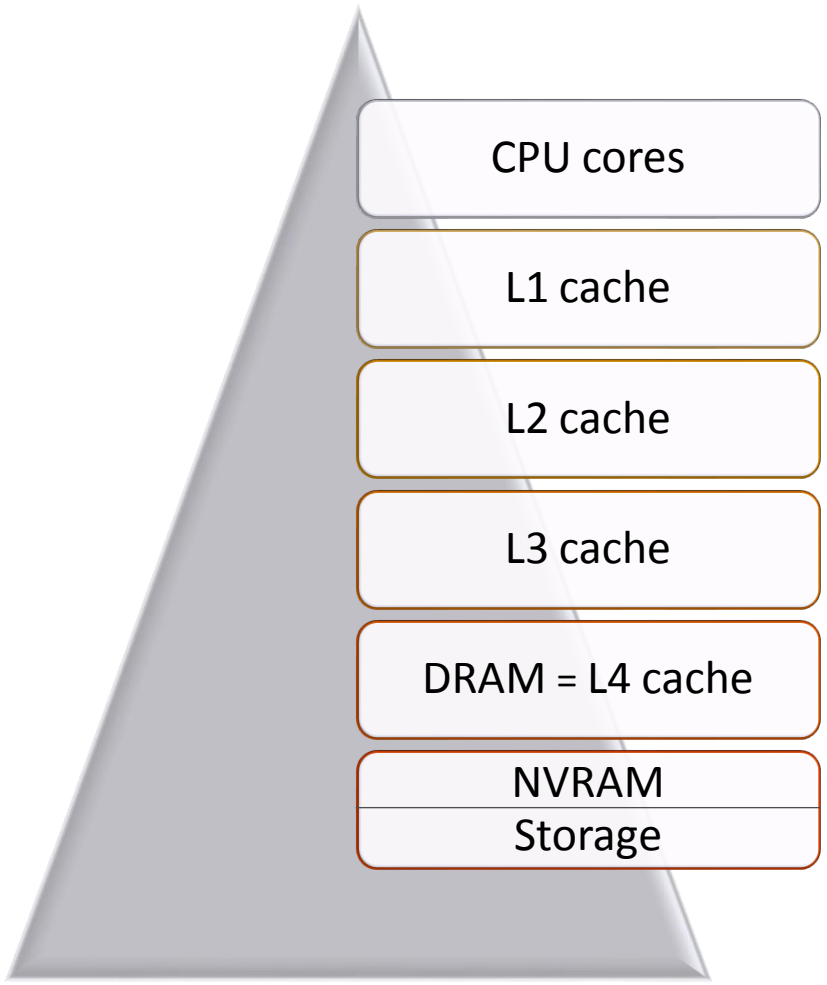
- Memory controller views DRAM like any other cache
- Applications only see NVRAM as large memory space
- Latency of byte-addressable SCM ~5-10x of DDR4 memory when connected to the same memory channels
- Not necessary to change applications to exploit SCM

One-level memory



- Only DRAM visible as memory
- Direct loads/stores from/to NVRAM
- Use NVRAM as very fast byte-addressable persistent local storage
  - local or distributed file system
  - object store
  - check-pointing
- Two options for support of applications
  - Modify applications
  - Enable via system software

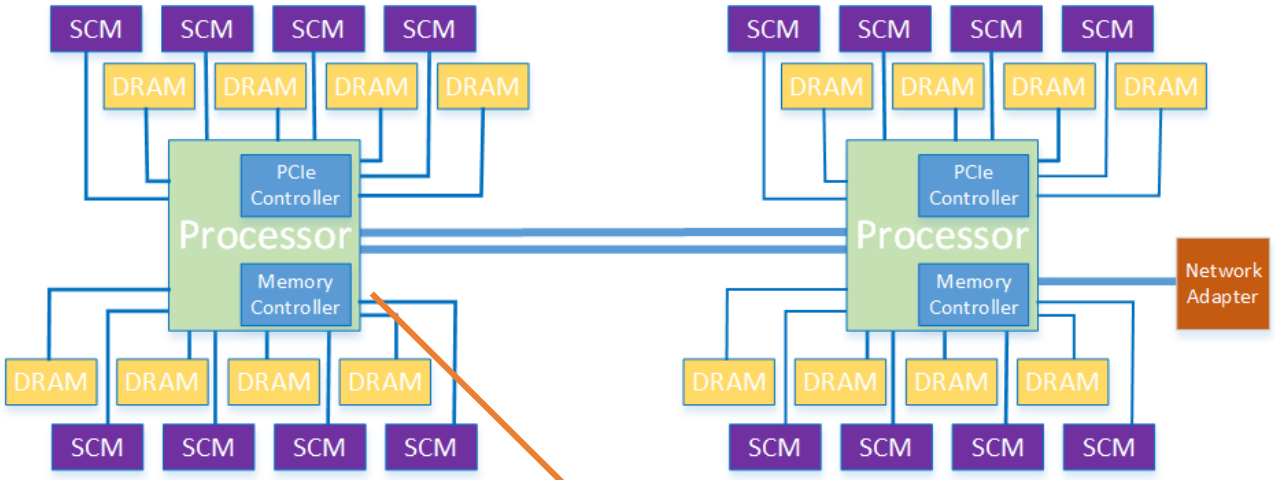
Two-level memory with partitioning

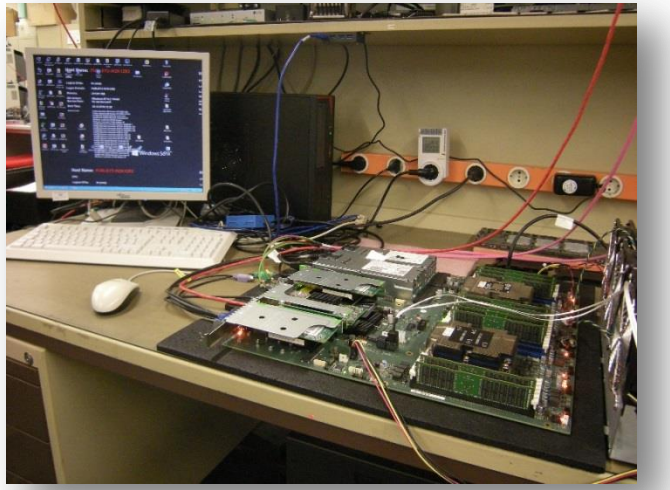


- NVRAM space can be partitioned
- Partially used as memory (not persistent) and as “app direct” (persistent)
  - Think “hybrid” mode on KNL

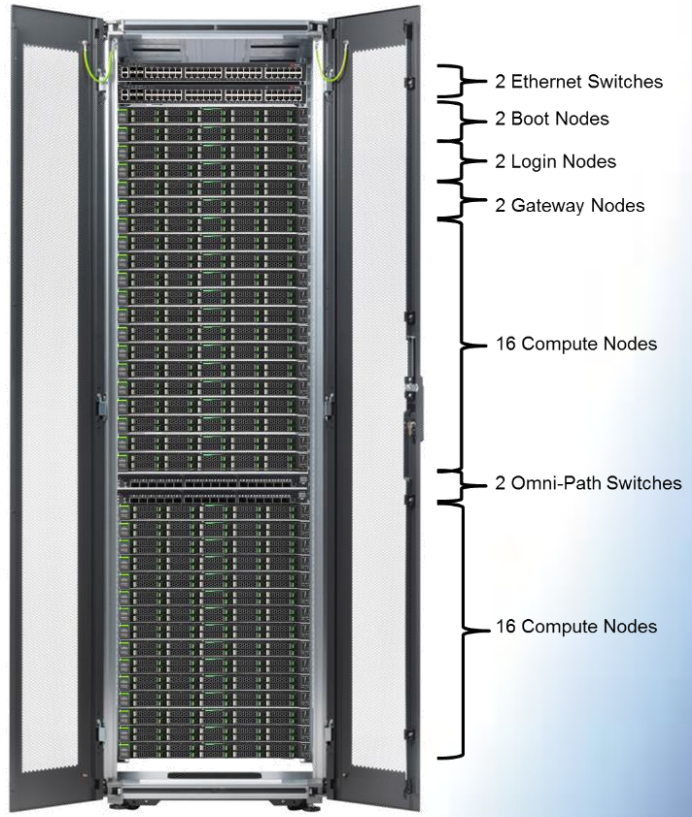
Hardware architecture

Hardware

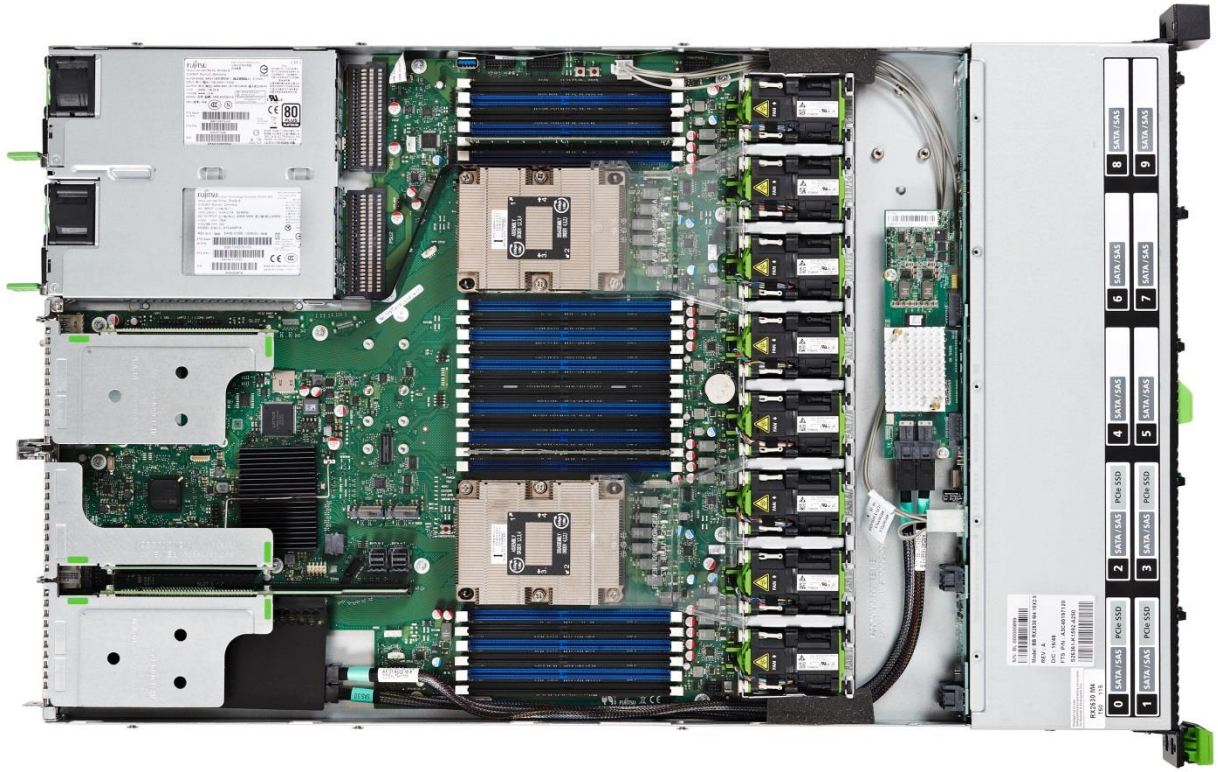


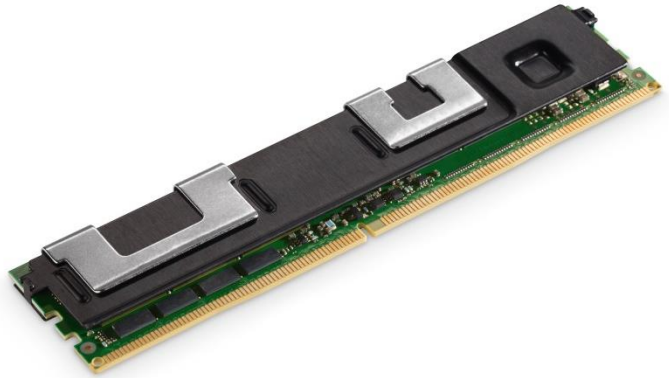


Customised memory controller deals with differences in performance characteristics between DDR and NVRAM



Hardware (2)





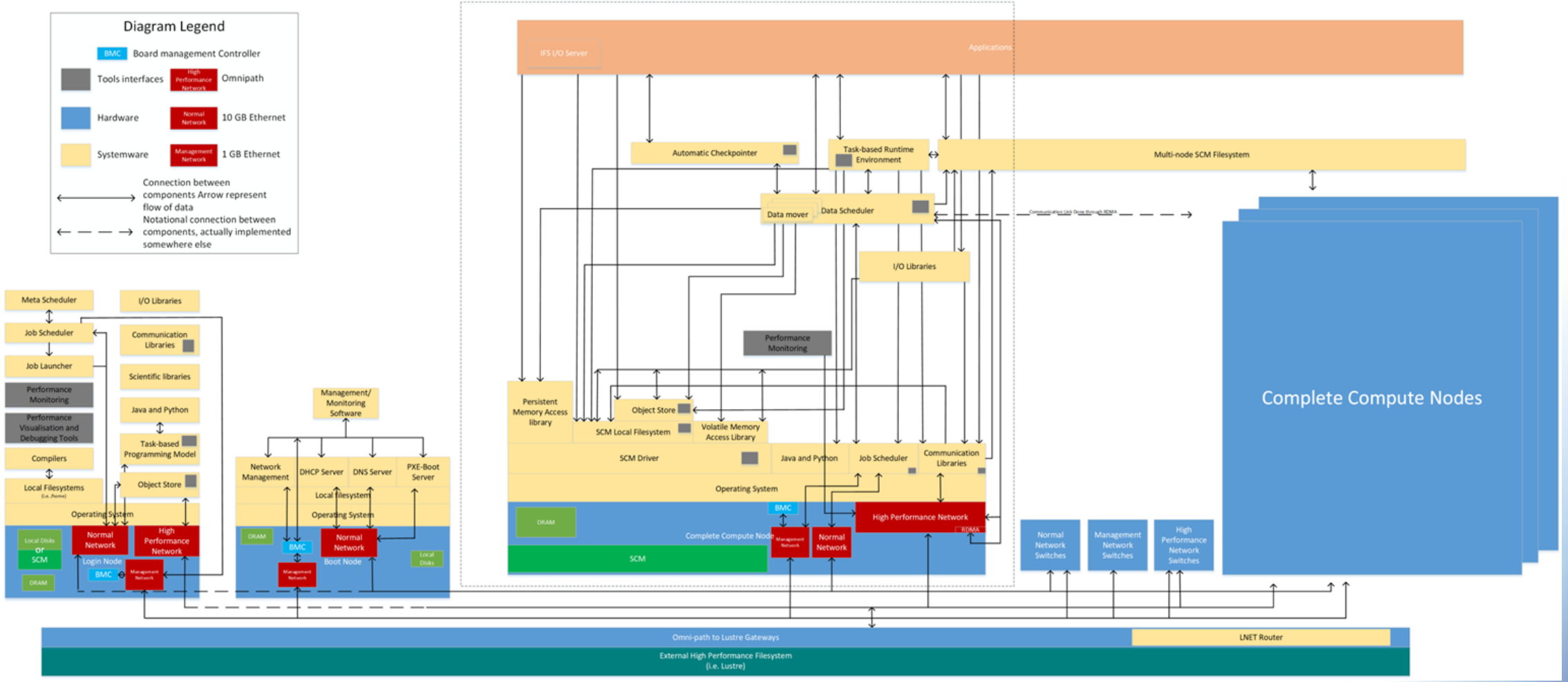
intel OPTANE™ DC PERSISTENT MEMORY

Use cases & systemware architecture

Use cases & scenarios for NVRAM

Pre-load and/or post-move data	No application changes required, dealt with in systemware
Distributed filesystem	No application changes required
Object/key-value store	Application changes might be required
Share data between applications	No application changes required, dealt with in systemware
Check-pointing & resilience	Application changes might be required
Direct memory access	Application changes required

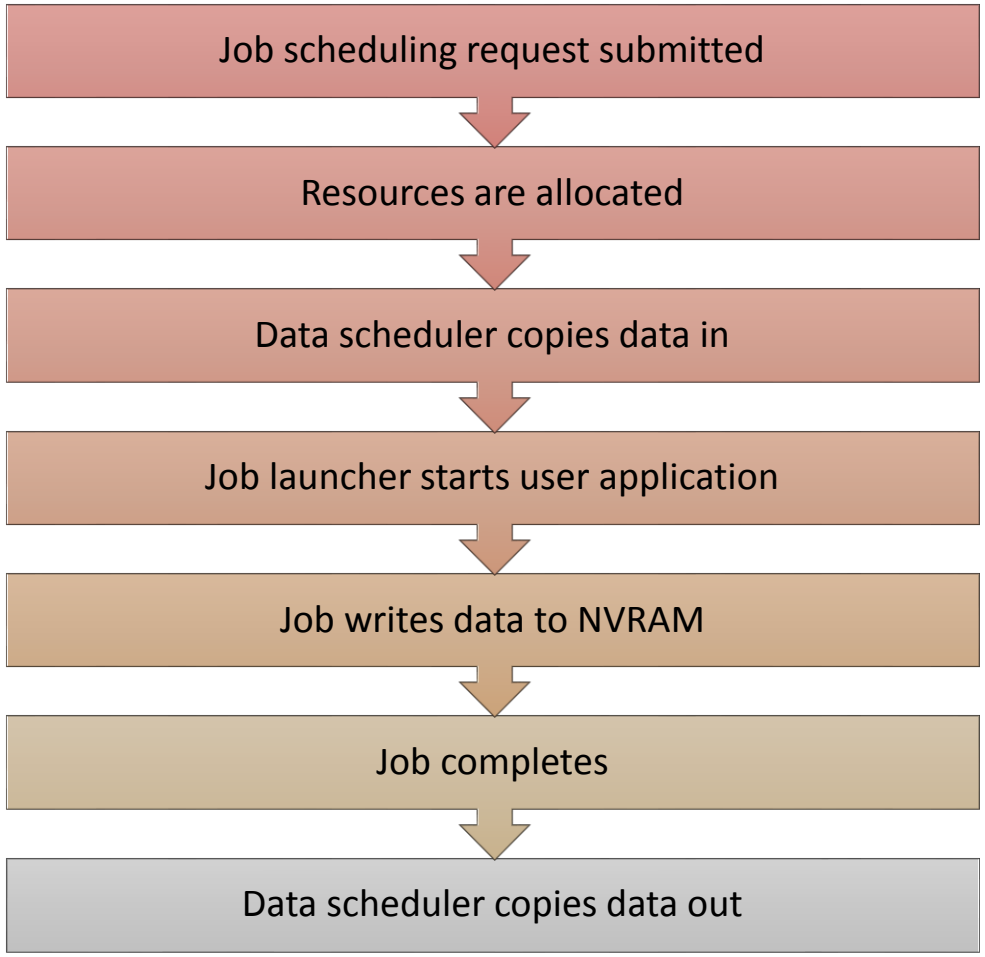
Systemware overview diagram

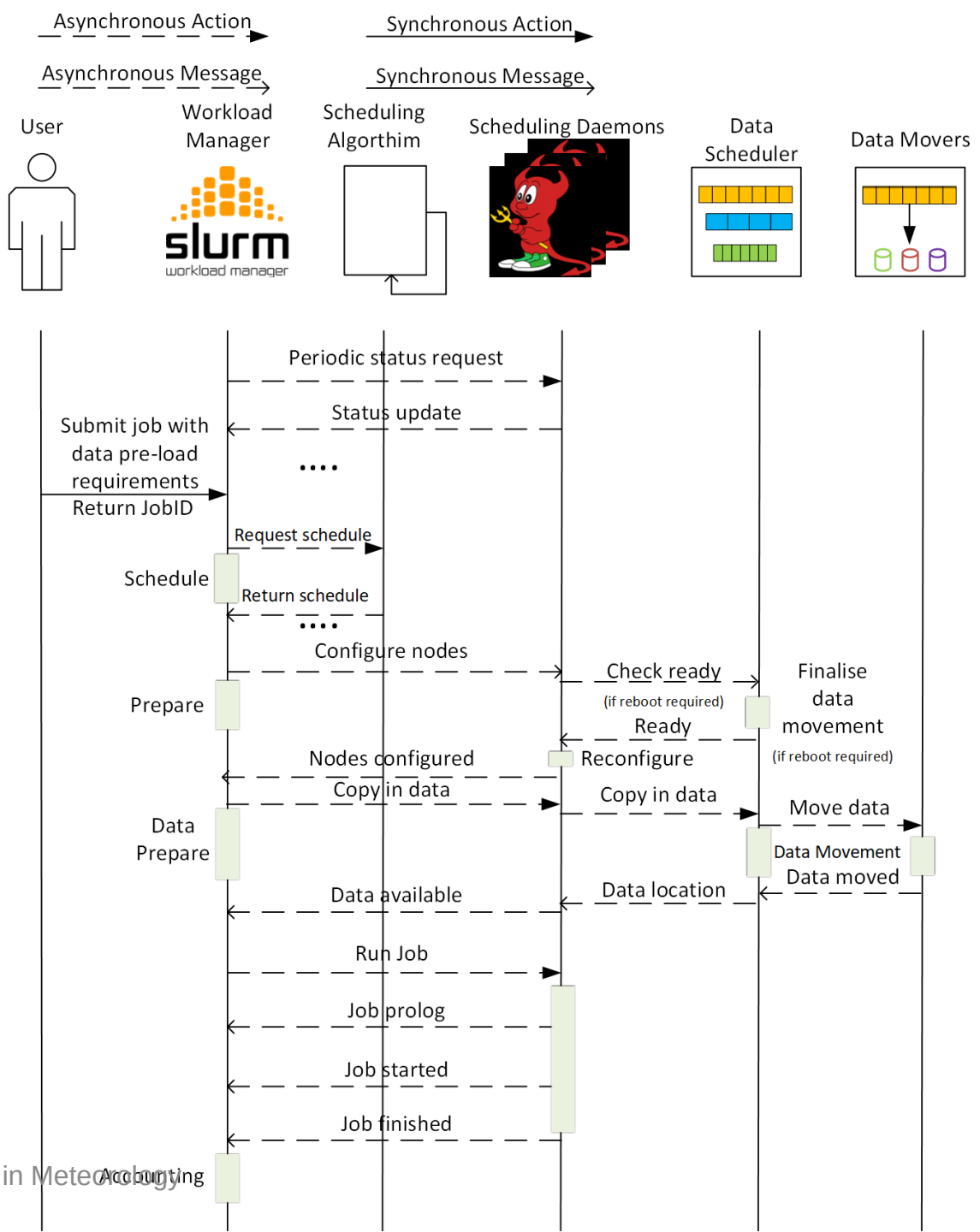


Systemware components

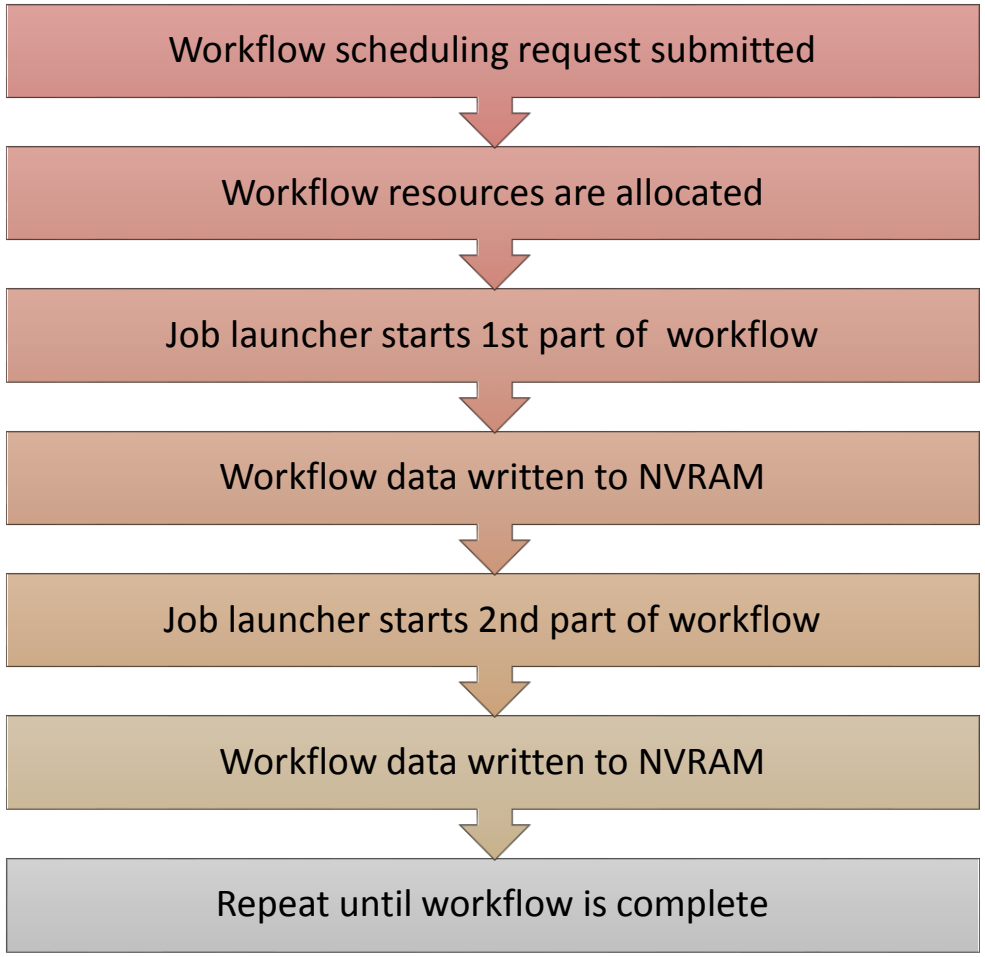
Job scheduler	• NVRAM resource monitored and managed by scheduler • Data locality and energy awareness
Data scheduler	• Data movement and shepherding • Between nodes, and to/from external storage
Object store	• Support for NVRAM • DAOS, dataClay
File system	• Local, on-node (scratch) • Distributed across NVRAM
Programming environment	• I/O libraries (MPI I/O, HDF5, NetCDF) • Tasking frameworks

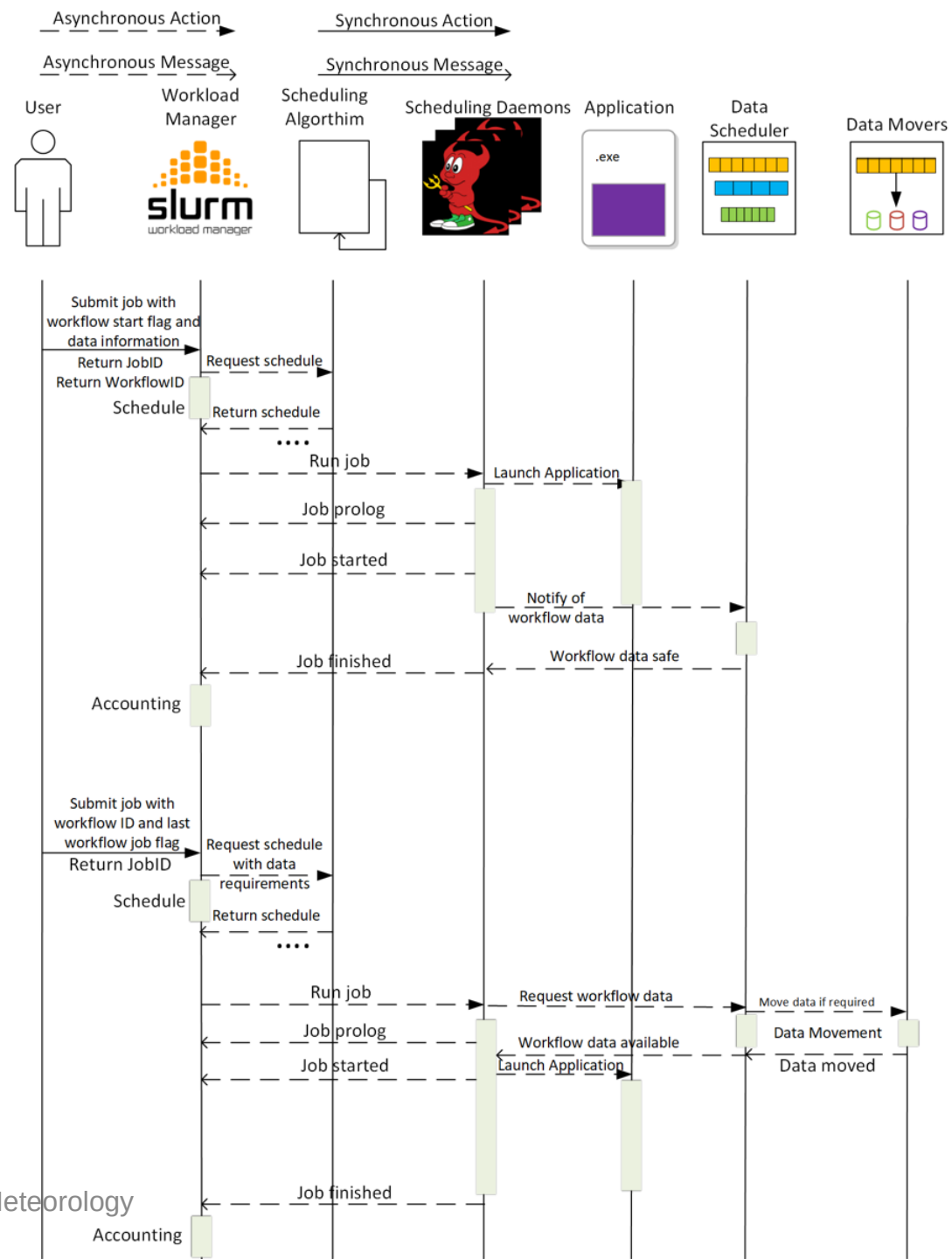
Example: pre-loading files





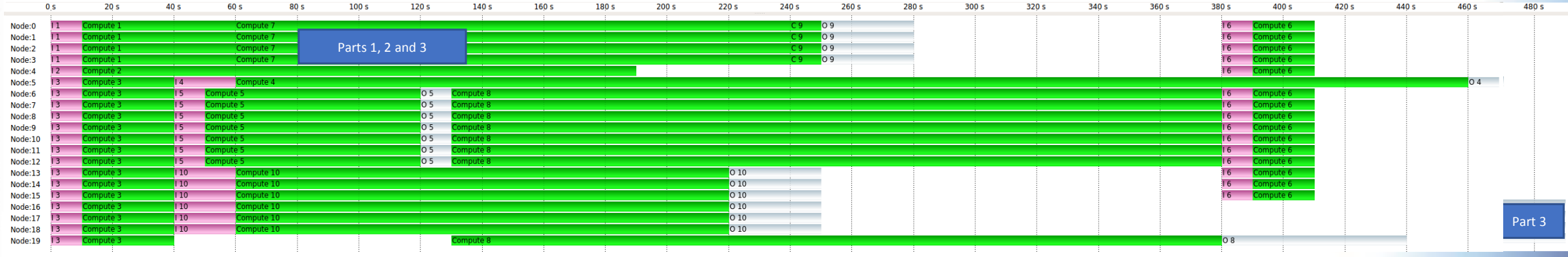
Example: workflows





Example: workflows (2)

- 3-part workflow
  - Output from Part 1 → input to Part 2
  - Output from Part 2 → input to Part 3



The challenge of distributed storage

- Enabling all the use cases in multi-user, multi-job environment is the real challenge
  - Heterogeneous scheduling mix
  - Different requirements on the NVRAM
  - Scheduling across these resources
  - Enabling sharing of nodes
  - Not impacting on node compute performance
- Enabling applications to do more I/O
  - Large numbers of our applications do not heavily use I/O at the moment
  - What can we enable if I/O is significantly cheaper?

Persistence

- Scenarios that exploit the capacity of NVRAM are clear
- Use cases for persistence however are much less obvious
- Possible examples
  - Workflows *could* benefit from persistence
  - In-memory databases for long term use
  - Datasets shared between multiple users for long running jobs
  - Resilience
  - “Closing the lid” on your supercomputer...?

More information

Michèle Weiland, Adrian Jackson, Nick Johnson & Mark Parsons, *"**Exploiting the Performance Benefits of Storage Class Memory for HPC and HPDA Workflows**".* Supercomputing Frontiers and Innovations, Vol 5, no. 1, pp. 79-94. doi.org/10.14529/jsfi180105

Adrian Jackson, Michèle Weiland, Mark Parsons, Bernhard Homölle, *"**Architectures for High Performance Computing and Data Systems using Byte-Addressable Persistent Memory**",* http://arxiv.org/abs/1805.10041

Questions?

www.nextgenio.eu

@nextgenio