The Next Community Cluster: Designs & Policies

With latent (and increasing) demand for local HPC resources, the University's IT HPC team has been busy designing the third community cluster. Lessons from prior clusters that inform the design of the next cluster include:

- **Cluster #3 will use latest generation Intel processors.** AMD has not yet made its entry back into the HPC marketplace, and alternative architectures would present support and compatibility issues for a general-purpose machine.
- **Cluster #3 must increase average usage.** Both Mills and Farber have historically been underutilized.
- **Cluster #3 will use a storage design similar to that of Farber.** Lustre will be used for high-speed shared scratch storage. NFS (with off-site snapshotting) will be used for home directory and workgroup storage.
- **Cluster #3 must have greater growth flexibility (not just capacity).** Being dependent on funding as unpredictable as grants can be, the cluster must be able to grow (and scale) many time its original size with a longer open period for growth than prior clusters.

Stakeholders in previous clusters also provided feedback:

- **Frequent large-scale purchases of hardware are difficult to sustain.** Researchers cannot be expected to have ongoing access to large grant allocations earmarked to purchase significant amounts of local HPC resources. Budgeting smaller amounts annually is a more sustainable model for most established researchers.
- **Cluster lifespans on the order of four to five years are too short.** In some units, the average span of a graduate student's research program easily outlasts the clusters. Some complex research workflows would benefit from *not* having to be reengineered every few years as new clusters are built.

All of these factors are considered important to the success of the next community cluster.

Design Goals for the Next Community Cluster

In late 2016, an RFI (Request For Information) was submitted to a group of prominent HPC vendors on behalf of University IT. Dell, HPE, Lenovo, and Penguin returned information packets that provided insight into how each would satisfy the design goals:

- **Increased compute density.** Previous clusters used classic 1U nodes, but current designs from all vendors allow for at least two nodes per 1U. *Higher density* solutions:
    - Optimize power conversion (and minimize energy loss)
    - Decrease the number of racks and amount of floorspace necessary

- **Intel processors suited for HPC environments.** More cores per processor and better overall performance of those cores versus Farber.
- **Continue to provide optional add-ons.** Offering system memory sizes above the baseline and GPU (and other) coprocessors is mandatory.
- **More reusable infrastructure versus previous clusters.** Allow for lowest-cost upgrade of components of the cluster in later years of operation.
- **Storage present in each rack.** As racks are added to the cluster, NFS and Lustre storage capacity increase; Lustre bandwidth increases; and underlying efficiency increases as SSD begins to replace mechanical media.
- **Scalable well beyond Mills and Farber sizes.** The high-speed network must be sized to allow for simple expansion by adding racks, up to ca. 11 computational racks.
    - In-rack network switches must use a ratio of node:uplink ports no worse than 4:1.

After several months of interaction with these vendors, the following specifications were produced by IT HPC staff for the first-generation hardware in the next community cluster:

Processor	2 x 18C Intel Xeon E5-2695v4
Memory	128 GB DDR4-2400 ECC (8 x 16 GB)
	(options for 256 GB, 512 GB)
GPU	2 x nVidia Tesla P100 (one per socket)
High-speed network	100 Gbps Intel OmniPath
Storage	120*N* TB Lustre (high-speed scratch)
	60*N* TB NFS (home & workgroup directories)
Operational lifespan	8 years
Maximum size	*N* = 11 racks (700+ nodes, 20000+ E5-2695v4 cores)

Realizing an Extended Lifespan

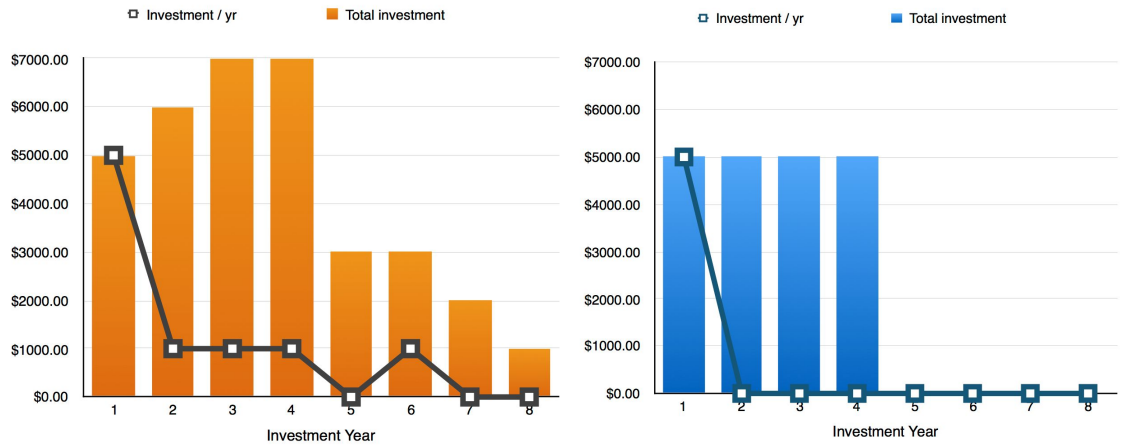
Increasing the operational lifespan of the next community cluster relies on a *rolling-upgradeable* model for expansion and replacement of hardware over time. As demand increases, additional racks of compute hardware are added to the cluster. When compute hardware exits warranty, it can be replaced. IT will periodically work with the vendors to refresh the design specification, so new hardware added to the cluster will remain cutting-edge as time goes by.

With this kind of design and stakeholders' desiring lower-cost investment opportunities, the previous investment scheme of purchases by node-oriented monetary increments must be altered. A *minimum*

buy-in of e.g. one node is still mandated, but IT hopes to make smaller increments (e.g. $1000) possible thereafter. The dollars invested in the cluster represent a workgroup's *share* of the cluster resources.

Investments in the next cluster will continue to have a fixed term on the order of the warranty period for the compute hardware. This relationship is necessary in order to ensure every workgroup gets the share of resources to which it is entitled. But the cost savings associated with a higher-density design and an increased amount of reusable infrastructure over the operational lifespan of the cluster is expected to produce a lower per-node price tag for stakeholders, netting them far more computational power per dollar than available on Mills or Farber.

The possibility for reinvestment in the cluster at lower levels (e.g. $1000) is hoped to provide an opportunity for workgroups to maintain their presence in the cluster over its full operational lifespan. Committing smaller amounts each year, a workgroup could increase (or just sustain) its computational capacity over time. If that annual reinvestment is not possible one year, the workgroup loses a fraction of its capacity — not all of it:



Each workgroup depicted bought a single node (at a hypothetical $5000) when the cluster was built. The orange workgroup reinvested $1000 annually after that: that workgroup's share peaks at $7000 in the third and fourth year. In year five the annual reinvestment was neglected, dropping the share to $3000. Another $1000 invested in year six leaves the workgroup with a diminishing share through year eight.

The blue workgroup, on the other hand, bought a node and did not reinvest. In year five that $5000 share expires, leaving no total investment. However, workgroups will maintain a nominal *single share* in the cluster through its operational lifespan, in the hopes that they will find the funding to reinvest. That single share maintains the workgroup's presence, data, and (in a limited regard) ability to run jobs

on the cluster. Naturally, the importance of that workgroup's jobs will be affected by its meager share count.

NOTE ON CAPITAL INVESTMENTS

In some cases, grants may require that the computing hardware purchased be treated as a capital asset, which can be reclaimed by the granting agency. In such cases, an investor would be required to buy in increments of per-node cost and the serial numbers of those nodes would be noted for tagging by University Procurement. Any investors having grants with such provisions should contact the IT HPC staff for additional details regarding what exactly would be "owned" by the granting agency.

Share-based Job Scheduling

To enforce the delivery of each workgroup's share of resources, the next community cluster will undergo a major redesign in terms of job scheduling. Foremost to these changes is a switch from Univa Grid Engine to the SLURM job scheduler. A growing disparity in HPC-friendly features between Grid Engine and its competitors forced this change. SLURM is the scheduler of choice in many HPC environments, including XSEDE, the OpenHPC stack, and many systems on the Top100 list. It is free and open source, which presents the IT HPC team with the opportunity to debug and address problems more effectively, as well as contribute back to that software community.

On both Mills and Farber, each workgroup had an *owner queue* that fed jobs to the specific nodes purchased by that group. On Farber, a *spillover queue* was introduced that allowed jobs in an owner queue to be matched to similar nodes not owned by the workgroup: this feature was partly responsible for the increase in average utilization of Farber (a spillover queue was later added to Mills, as well). The next cluster will eliminate owner queues altogether, allowing jobs to always run on any hardware that satisfies the job's resource needs.

Communicating jobs' resource requirements is an important aspect of how jobs are submitted for execution on the community clusters. Each cluster has mandated additional information about each job's resource requirements:

Mills	Farber	Next Cluster
Number of processor cores	Number of processor cores	Number of processor cores
	Amount of memory per core	Amount of memory per core
		Amount of time

The next community cluster will require jobs to specify the amount of real-world time required (also known as *walltime*). Each queue will have a maximum value inherited by any job not explicitly specifying a value. Users will be asked to make conservative estimates (informed by previous runs of similar jobs, for example). With every job having a known execution time, SLURM can work to arrange jobs in the most efficient order of execution (a technique called *backfill*), maximizing the ongoing utilization of the cluster and guaranteeing each workgroup its purchased share of cluster resources.

Getting Stakeholders Involved

Throughout the month of June, 2017, IT HPC staff met with groups of researchers to present the design and policy information documented above. These meetings were meant to solicit feedback — objections, concerns, even praise — with respect to the direction that IT would like to take the next community cluster. Groups included:

- Center for Applied Demography and Survey Research (CADSR)
- College of Agriculture and Natural Resources (CANR)
- Catalysis Center for Energy Innovation (CCEI)
- Civil and Environmental Engineering (CE)
- School of Marine Science and Policy (CEOE)
- Chemical and Biomolecular Engineering (ChE)
- College of Health Sciences (CHS)
- Computer and Information Sciences (CISC)
- Delaware Biotech Institute (DBI)
- Electrical and Computer Engineering (ECE)
- Linguistics and Cognitive Science
- Physics and Astronomy
- Psychological and Brain Sciences

Reactions were overwhelmingly positive. The changes to job scheduling elicited the most discussion, with the reusability and longevity of the design being a close second. A selection of specific questions generated within these groups follows:

Q	**If every workgroup reinvests $1000 in the cluster, my percent share will not change — and neither will my job priority. So what would that $1000 actually get me?**
A	The short answer: more compute cycles. Behind the scenes, dollars spent equate to CPU cycles. When the number of purchased CPU cycles exceeds the capacity present in the

cluster, another rack must be added to satisfy demand. So, in effect, all those small investments aggregate toward the purchase of additional capacity.

Q **On other clusters where share-based scheduling is in effect, I routinely see extremely long wait times in the queue: for example, a month from the time the job was submitted to when it finally executes. Will that be how this community cluster behaves?**

A Long wait times typically indicate that a job scheduling system is either poorly configured or oversubscribed. IT will work to keep the overall investment of stakeholders well-matched with the amount of computational resources present in the cluster, which will combat oversubscription. The configuration of SLURM will be informed by real-world examples (XSEDE systems, etc.) and will hopefully function well from its inception. However, change will very likely need to happen over time, and IT HPC staff are committed to keeping your workflows running properly.

Q **If I have jobs that I need to test — to be sure I have the right flags or my input file is setup correctly — will there be a way to do that without waiting in a long queue?**

A Many HPC environments implement *development queues* for this sort of thing. A development queue typically embodies:

- A relatively short maximum execution time (e.g. four hours)
- A low maximum number of cores (e.g. two, for minimal parallelism)
- Hosts that are not as performant (e.g. a node with slower processors or hyperthreading enabled)

Any job that fits those guidelines could be run in a development queue. Vetting a job's startup or running it for a short time to estimate full runtime are both a good fit.

The ratios of node types (baseline, large memory, GPU) must reflect demand plus some amount of "overflow capacity." IT HPC staff utilized a Google form to collect basic node criteria from interested faculty/staff (see graphs on next page). Responses indicated a node count ranging between 33 to 58,

with some responders providing no count (there were also a number of interested faculty/staff present at face-to-face meetings who did not respond to the survey form).