Los Alamos NATIONAL LABORATORY

Bringing the Cloud to HPC Before Taking HPC to the Cloud

Quincy Wofford

qwofford@lanl.gov

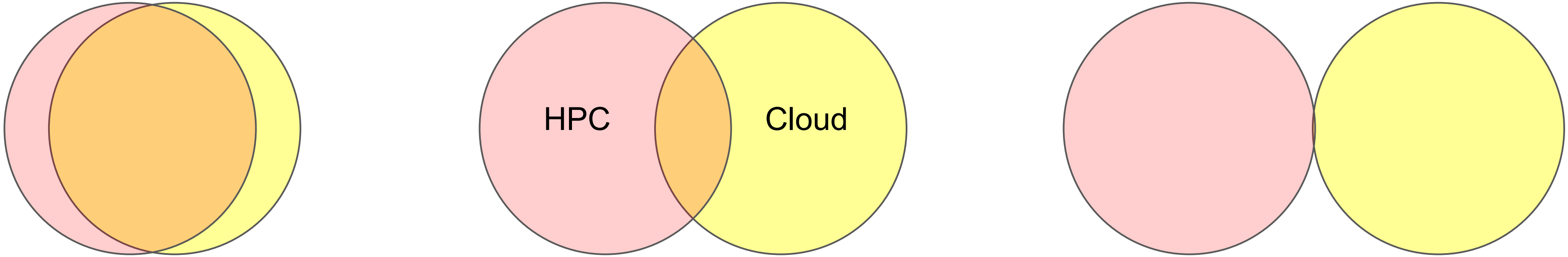
Scientist

Applied Computer Science (CCS-7)

Los Alamos National Laboratory

Image generated by DALL-E 2

Development Operations is central for all Cloud and HPC workflows



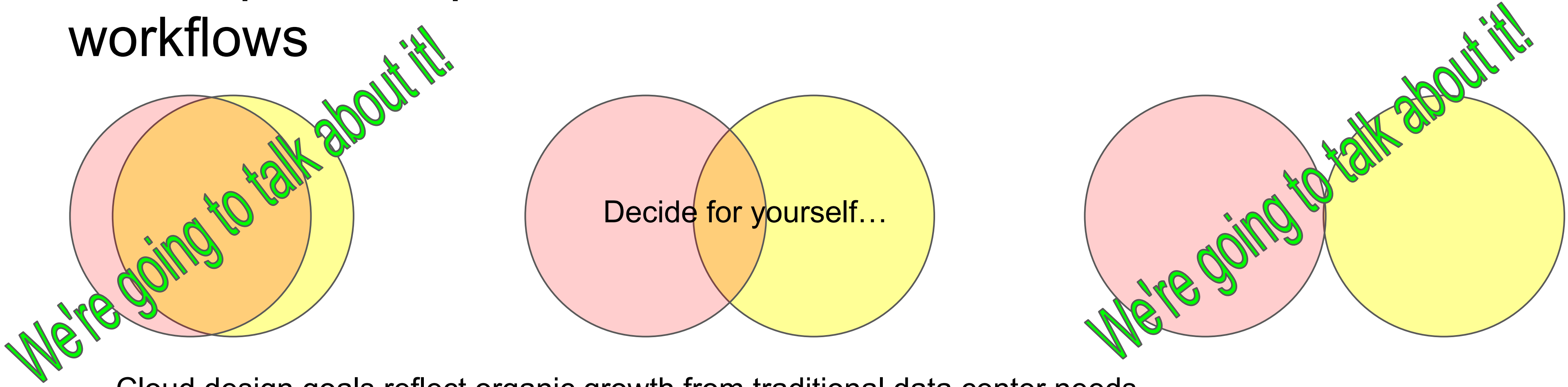
Cloud design goals reflect organic growth from traditional data center needs.

Development Operations is bigger than software. It’s an operational standard common to any process, implicit or explicit.

Typical data center apps strain compute resources but generally tolerate memory and network latency better than HPC apps, and tools reflect this by offering scalability at the cost of latency.

How much convergence makes sense, and what are the requirements for convergence?

Development Operations is central for all Cloud and HPC workflows



Cloud design goals reflect organic growth from traditional data center needs.

Development Operations is bigger than software. It’s an operational standard common to any process, implicit or explicit.

Typical data center apps strain compute resources but generally tolerate memory and network latency better than HPC apps, and tools reflect this by offering scalability at the cost of latency.

How much convergence makes sense, and what are the requirements for convergence?

Middle Solutions Don’t Work for All Operational Needs

Development Operations := An active process refined by clear levels of detail.

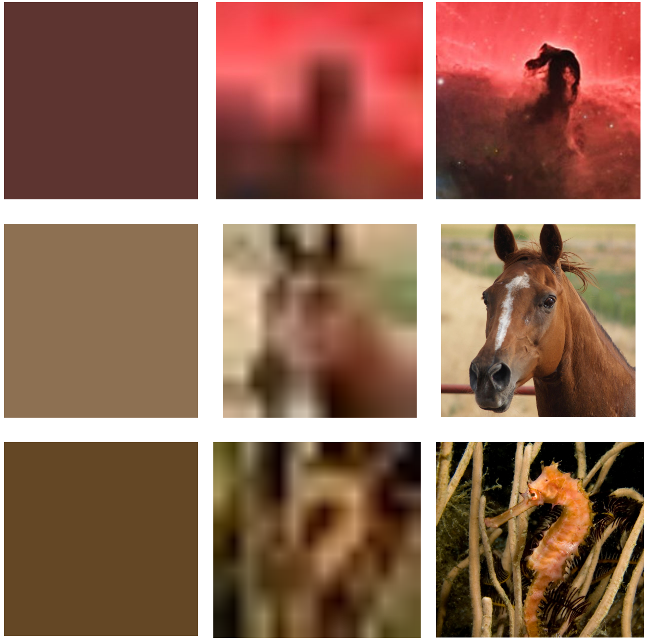
The Same Operational Task Has Different Deliverable Outcomes

Development Operations := An active process refined by clear levels of detail.

Operation success depends on context

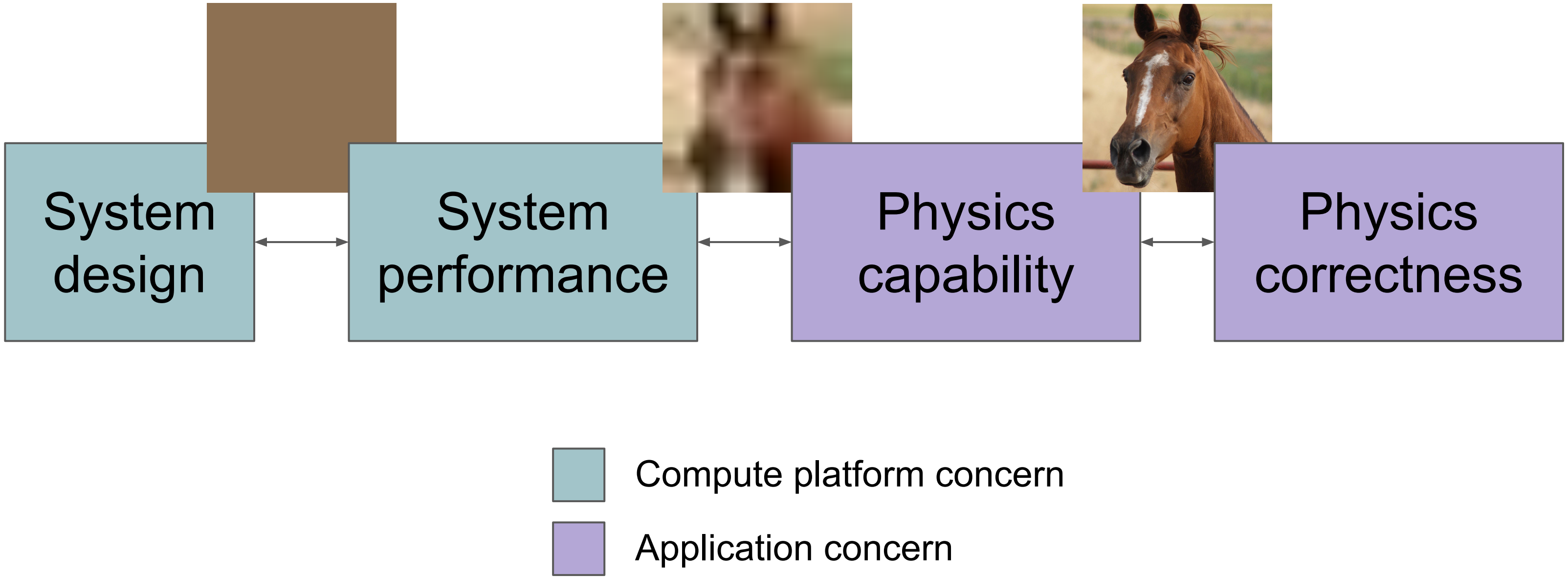
Development Operations := An active process refined by clear levels of detail.

Automation patterns are applicable to problems of a similar type

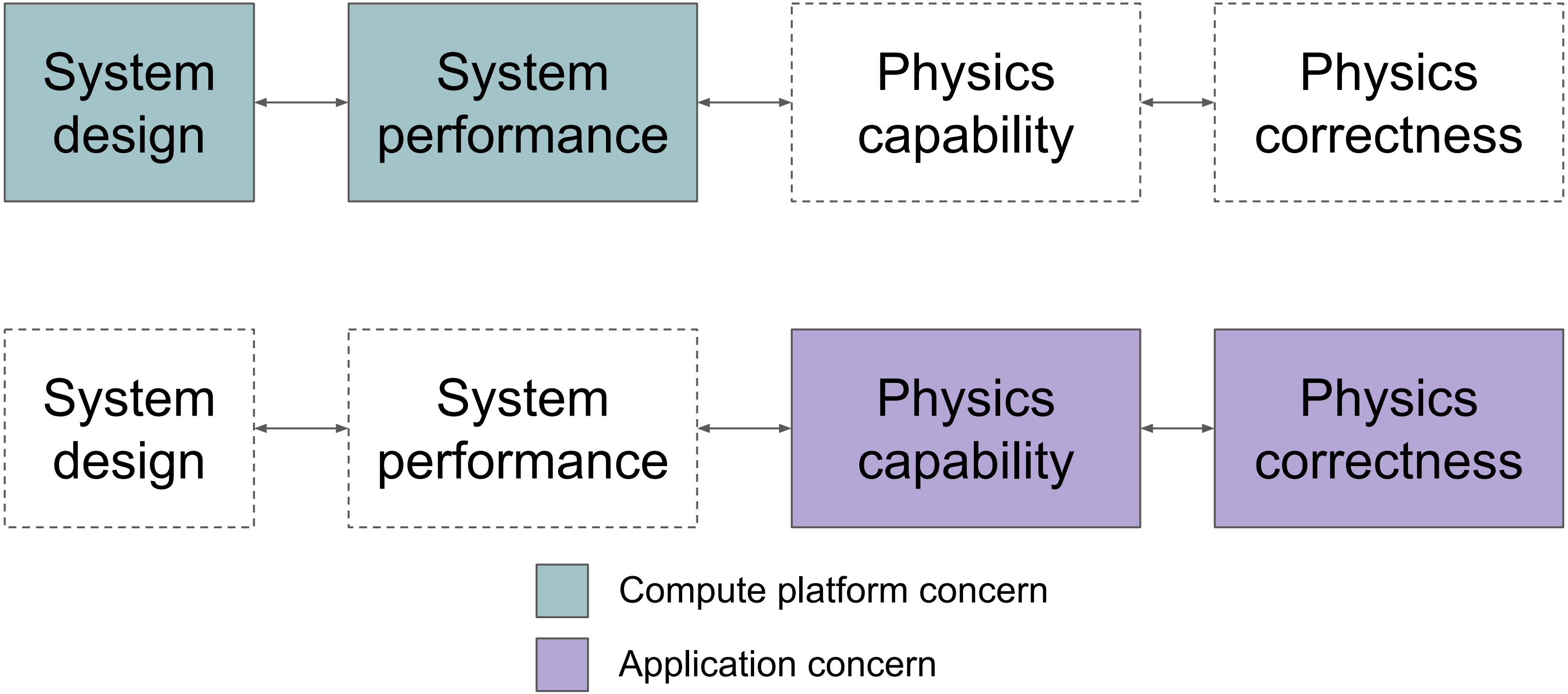


Development Operations := An active process refined by clear levels of detail.

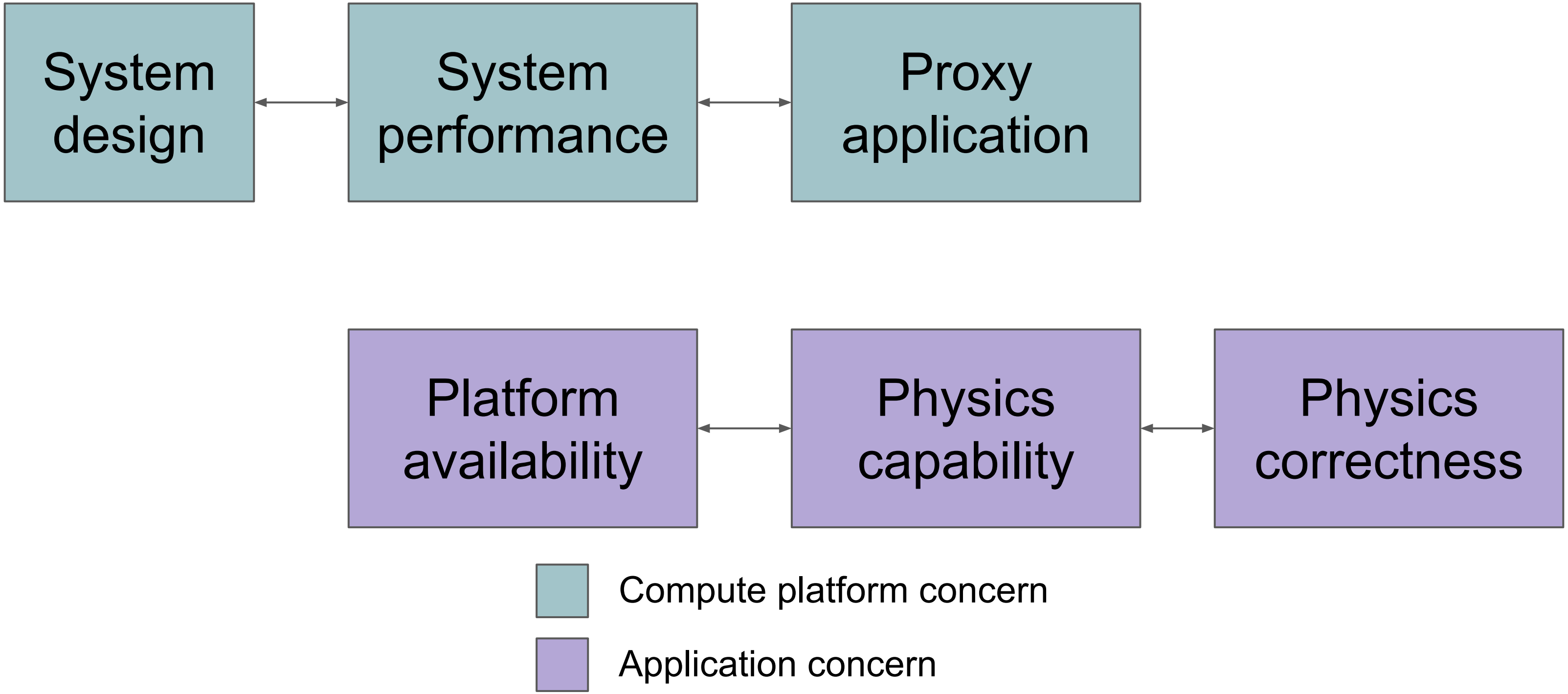
Application experiments have computing and physics goals.



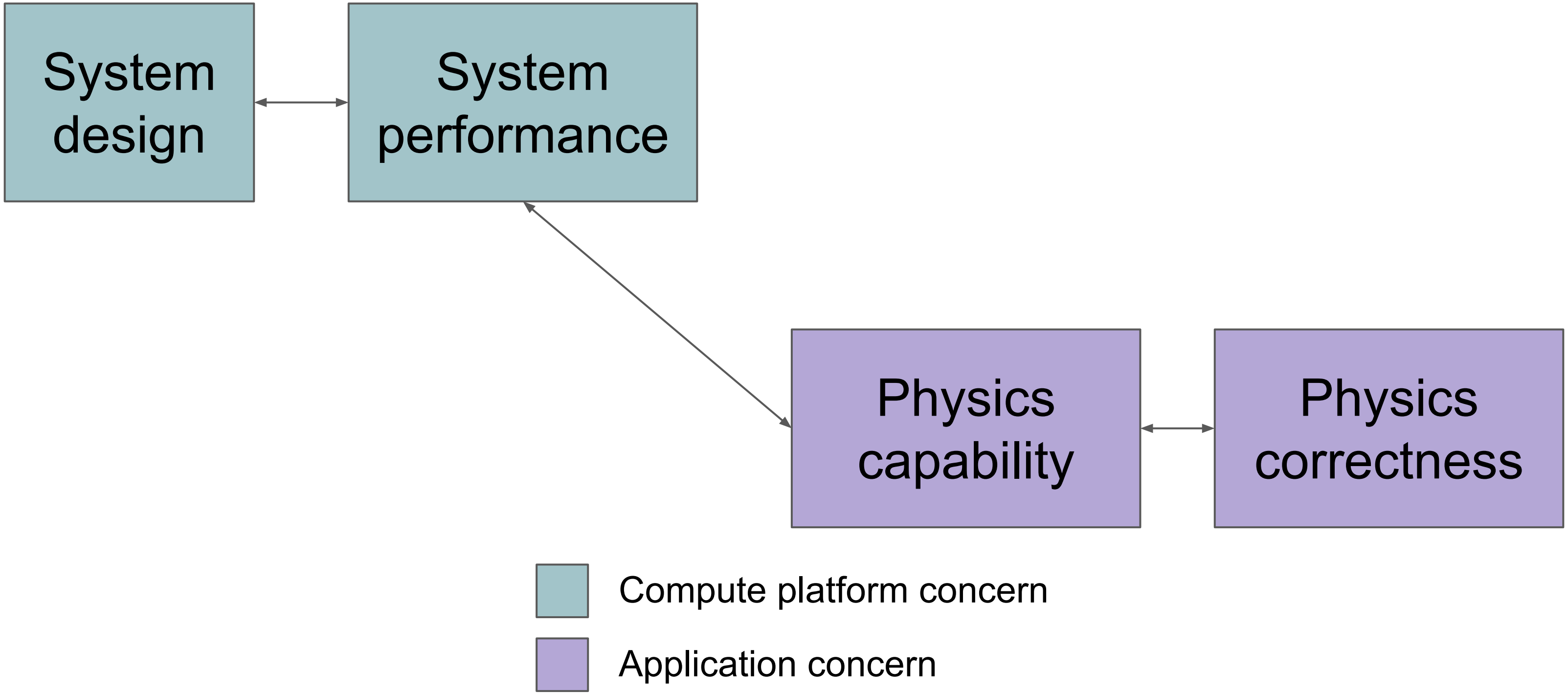
System performance may be evaluated separately from application performance.



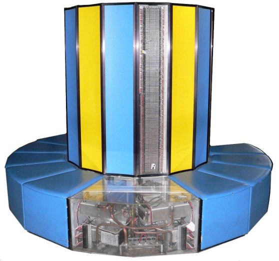
System performance may be evaluated separately from application performance



We’re in the business of Application Experiments, which are inextricably linked to hardware. We can embrace that.

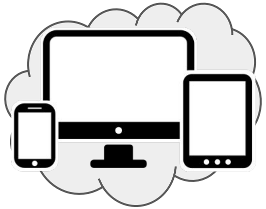


“DevOps” tools are mature, but not designed for HPC DevOps



System design ↔ System performance ↔ Physics capability ↔ Physics correctness

≠

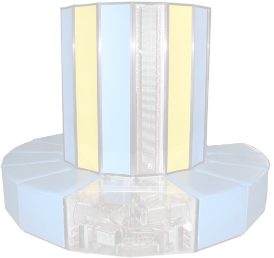


System design ↔ System performance ↔ Consumer content ↔ Consumer accessibility

Compute platform concern

Application concern

HPC and Cloud DevOps Can Still Use the Same Tools



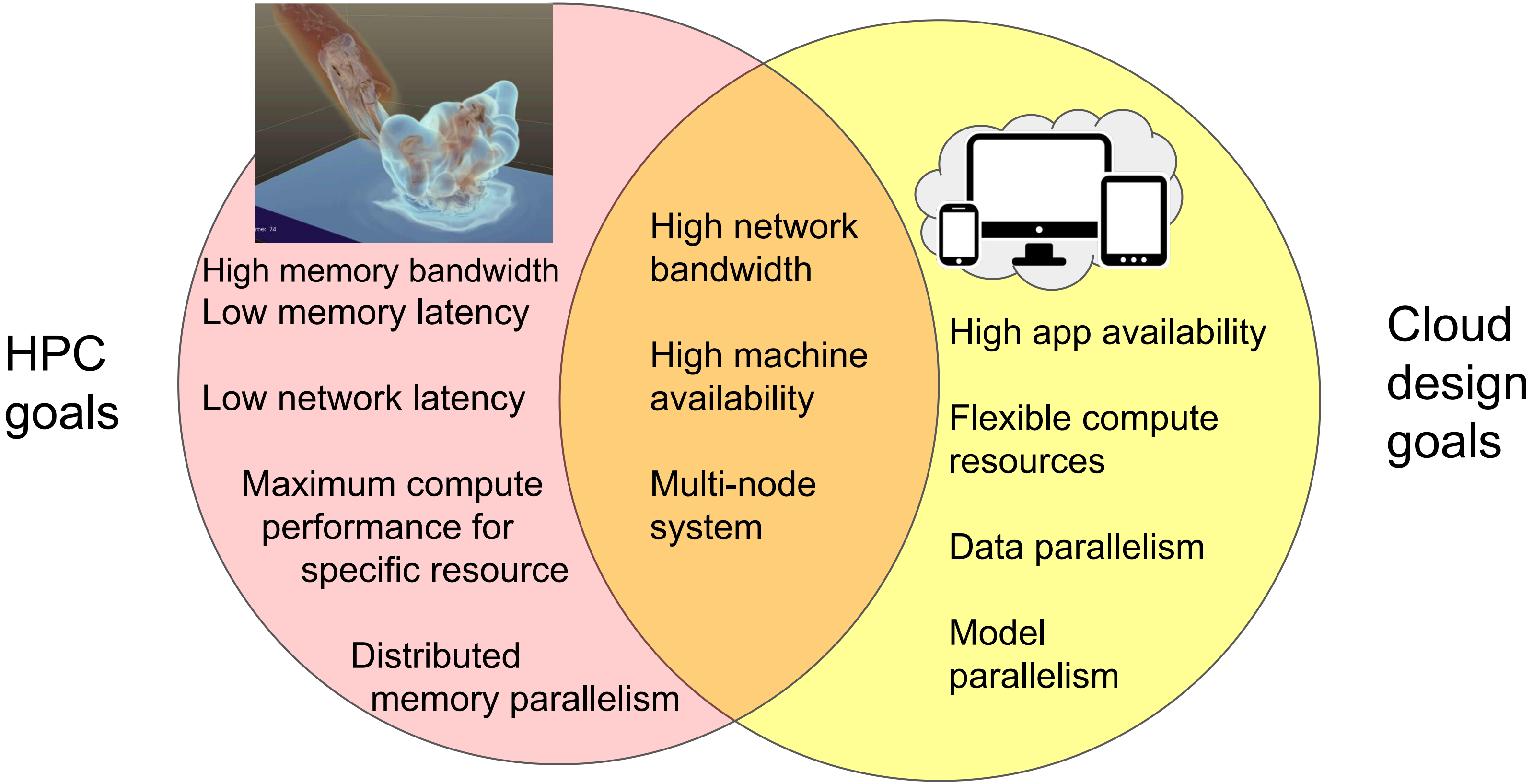
What is the cost of failed application experiment?
- Minimize error.

Today’s code automation works with both.

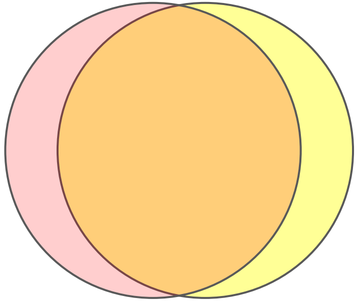
What is the cost of failed customer engagement?
- Minimize downtime.

GitLab RED HAT® Quay.io

Merging Capabilities requires managing trade-offs



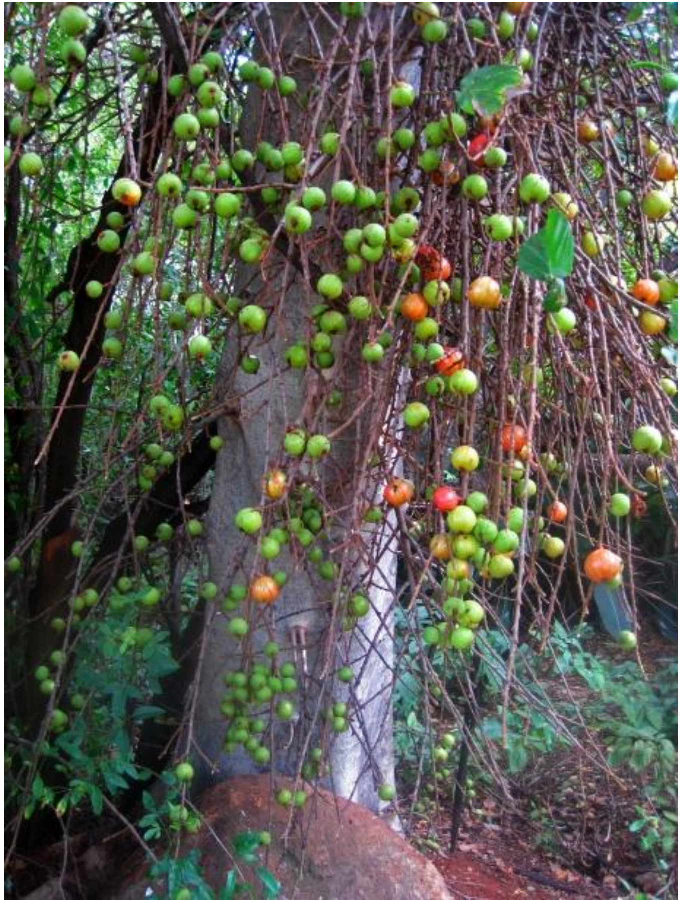
HPC (mostly) in the cloud



We demonstrate technical feasibility for the most complex apps and problems

First steps for HPC workflows in the Cloud

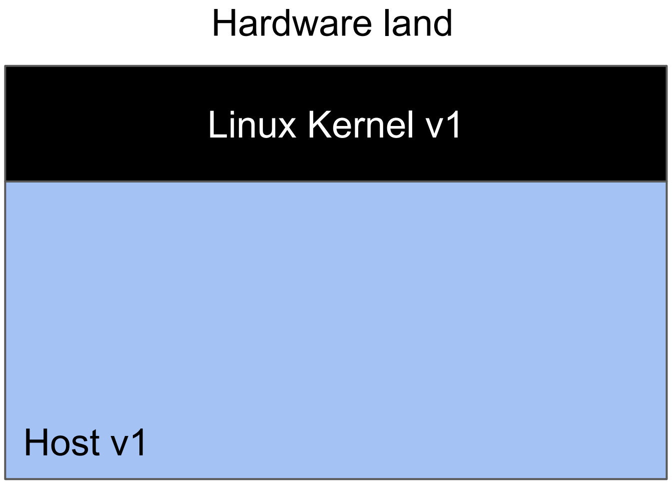
- Embarrassingly parallel apps/problems
- On-node GPU parallel apps/problems
- Latency tolerant or latency hiding distributed memory parallel apps/problems (they exist!)
- User interfaces
- These apps are more likely to benefit from existing cloud tools, and offer opportunities to evaluate security and performance for running very tough apps.



Requirements for very challenging HPC workflows in the Cloud:

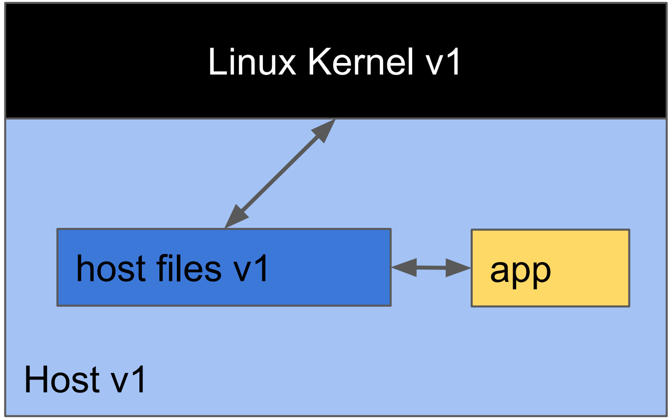
- Bless the kernel, and make it known.
- Support arbitrary containers.
  (kernel user namespace makes this easy)
- Ability to inspect host's hardware.
- Ability to inspect parallel orchestration environment.
- Container acceptance tests in distributed memory parallel.
  (performance tests and physics tests)

Why do we need a blessed kernel?

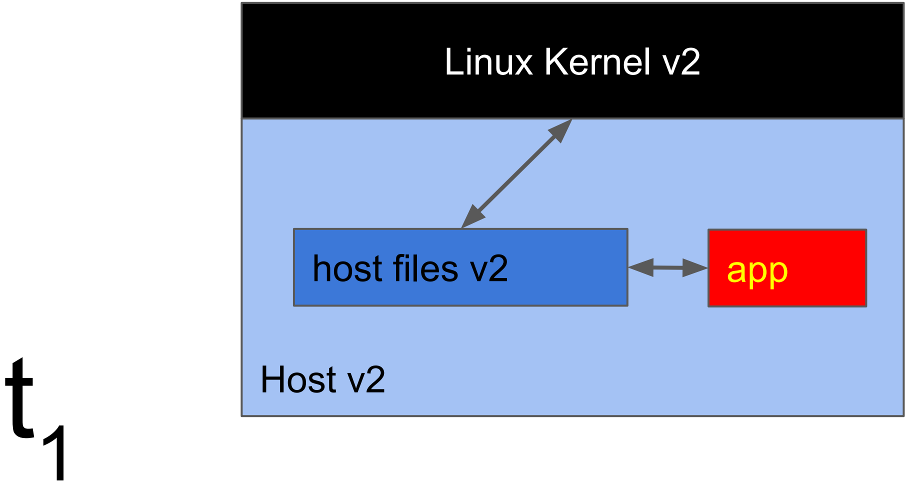


Why do we need a blessed kernel?

t_0



Why do we need a blessed kernel?



Answer:

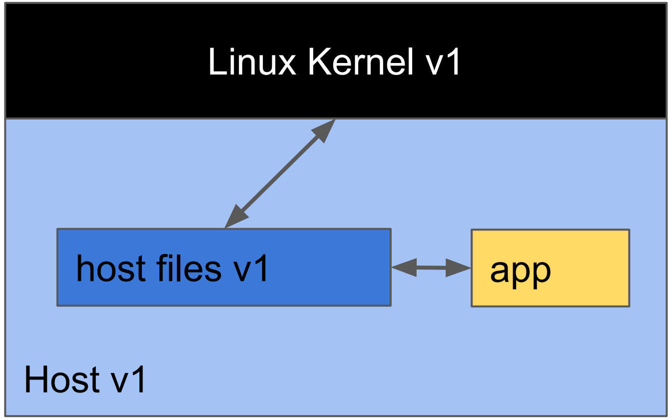
Hardware interface is a provenance concern for scientific experiments.

Performance degradation breaks HPC application requirements.

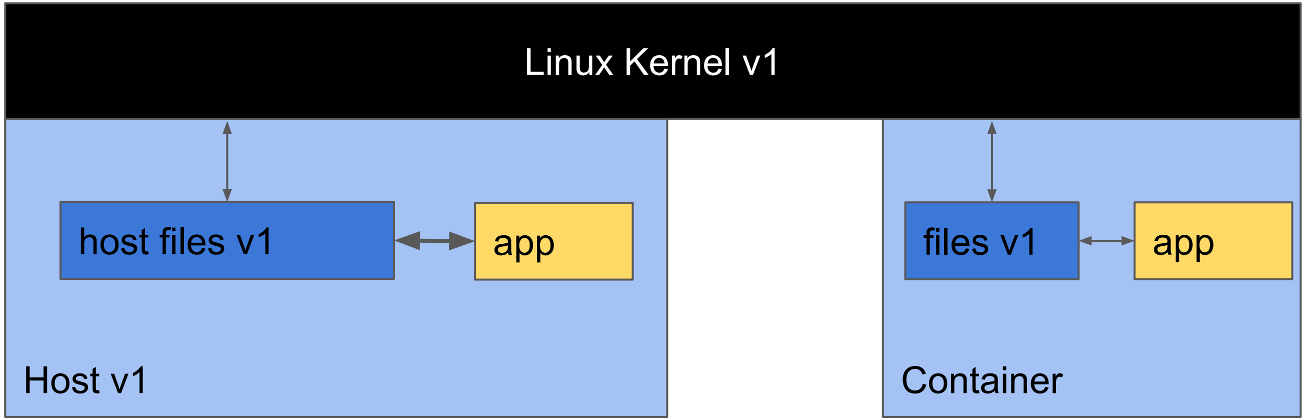
We need information about the kernel for experiment comparability and performance assurances.

Why do we need an arbitrary container?

t_0

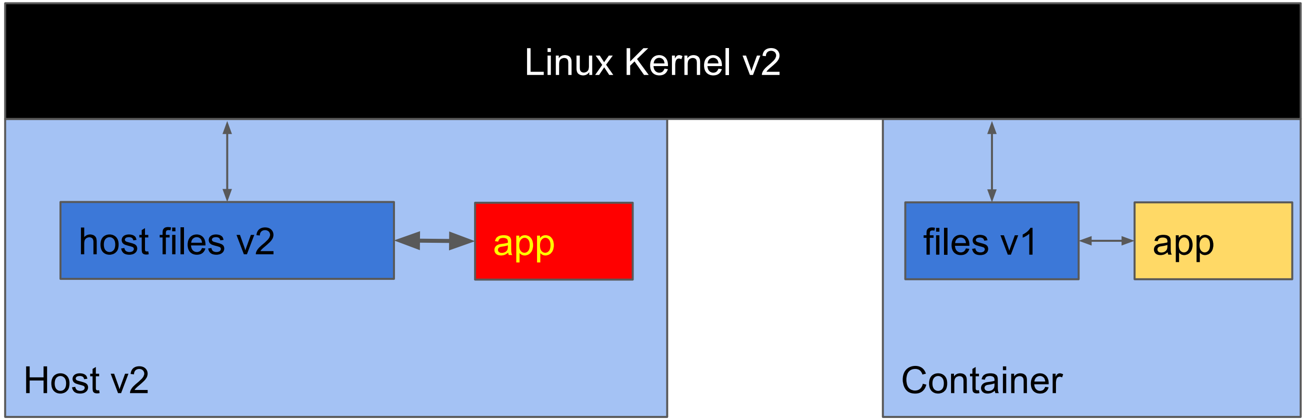


Why do we need an arbitrary container?



t_0

Why do we need an arbitrary container?



t_1

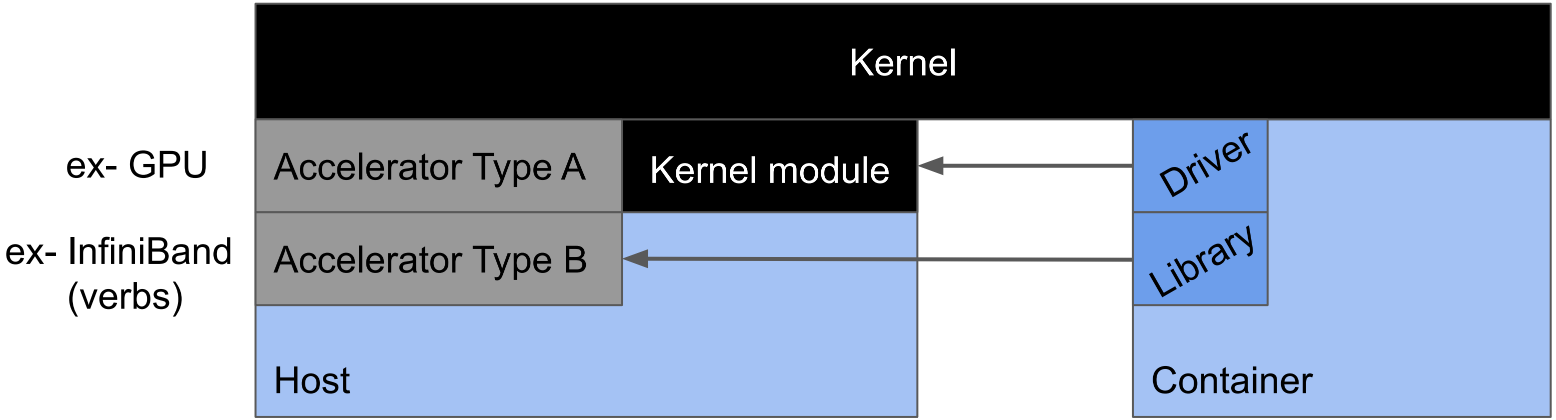
Answer: glibc ABI compatibility and isolated application dependencies can buy us some portability, which saves on DevOps cycles.

Why do we need the ability to inspect host's hardware?

Answer:

Containerized applications have 3 possible interfaces to the host and its devices:

1. The Linux kernel
2. Kernel modules
3. Kernel bypass libraries



Source: *Complete Provenance for Application Experiments with Containers and Hardware Interface Metadata*

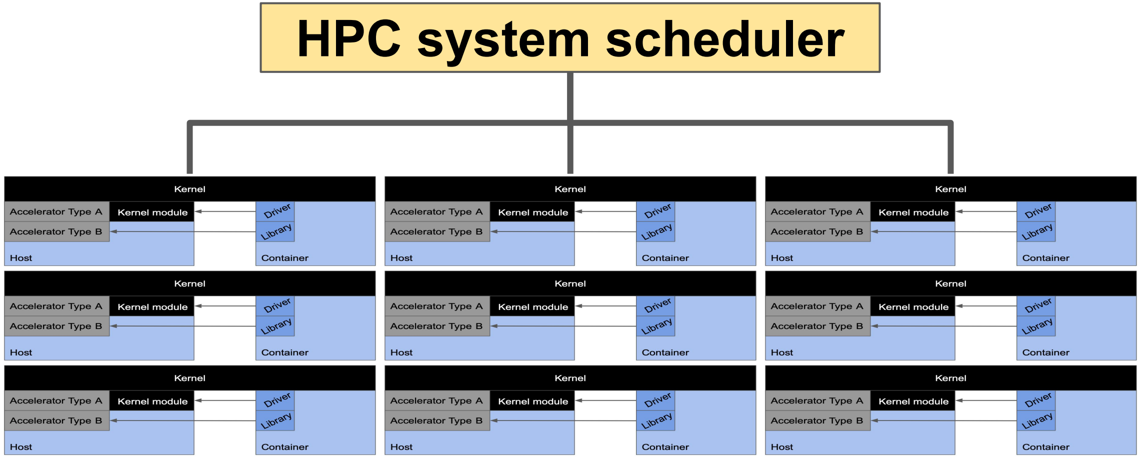
Why do we need the ability to inspect parallel orchestration environment?

Answer:

Our containers require an appropriate abstraction for distributed parallel execution.

Abstractions that work so far:

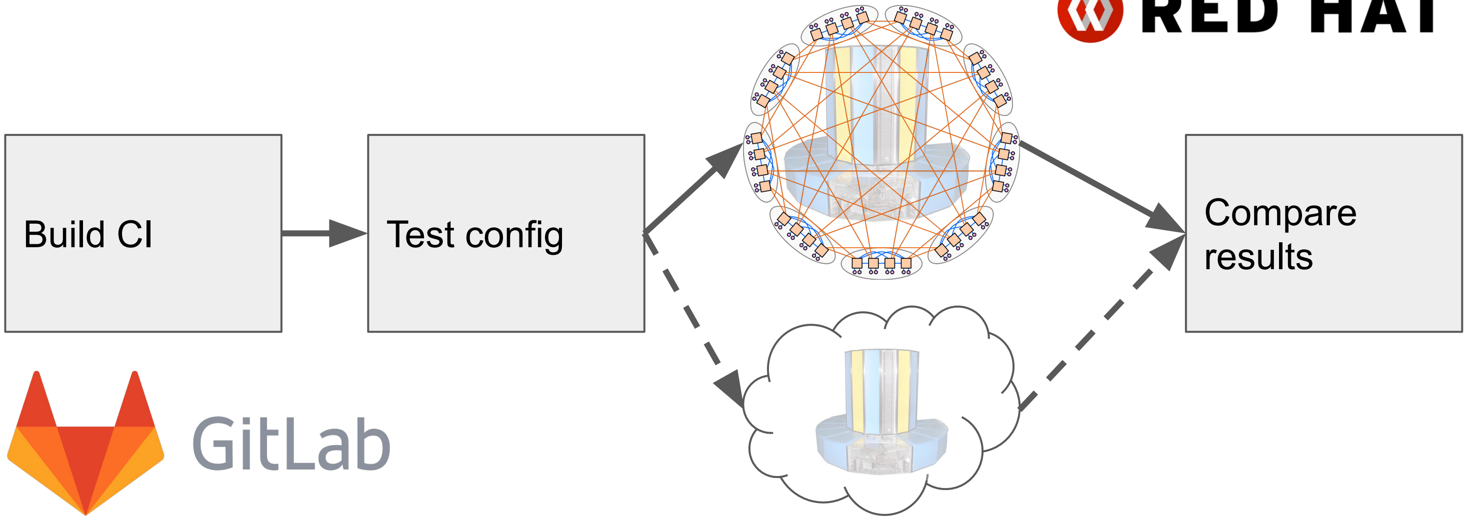
- Matching host MPI (harder to maintain)
- Matching host PMI (less admin control)



Why do we need container acceptance tests in distributed memory parallel?

Answer:

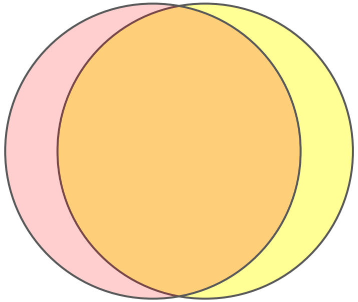
We do this at LANL using Gitlab, Slurm, and Jacamar.
This is compatible with any cloud software stack, on-prem or remote.



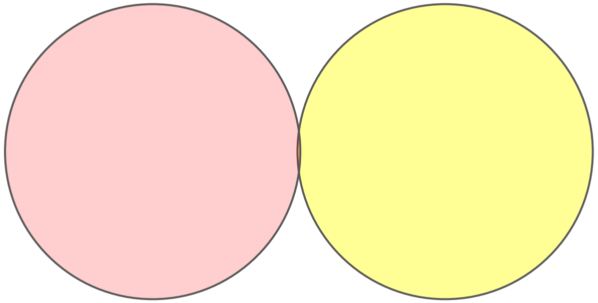
Two extremes:

- HPC completely in the cloud
- Cloud as an accelerator

HPC (mostly) in the cloud

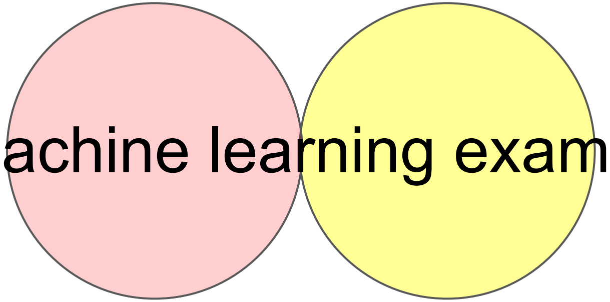


HPC+Cloud



HPC+Cloud

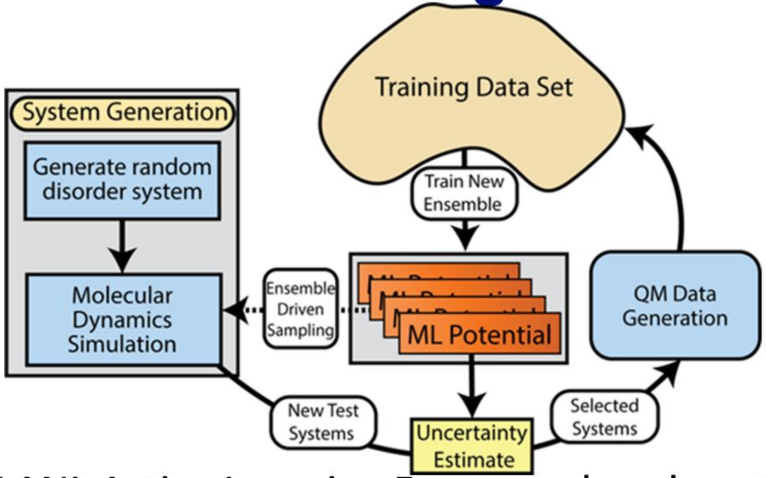
Machine learning example

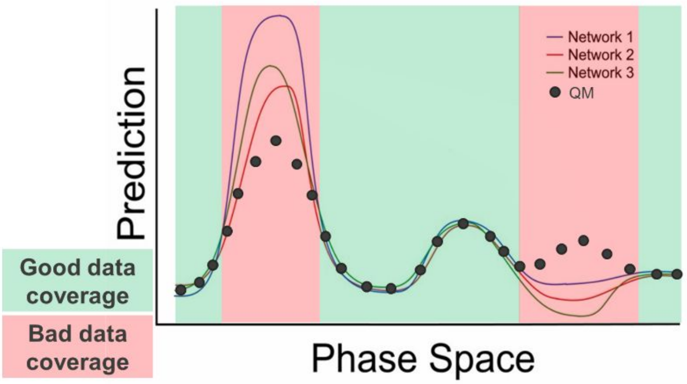


- Leverages existing HPC infrastructure with minimal changes.
- Leverages cloud infrastructure strengths with minimal changes.

- Two primary challenges, and example solutions:
  1. HPC/Cloud interface – Active Learning Framework (ALF), LANL
  2. Machine-actionable data curation – Data Science Infrastructure (DSI) Project, LANL

Active Learning and Data Selection





LANL Active Learning Framework code automates sampling, electronic structure, and training tasks using a Parsl (product of ECP) backend for job management.

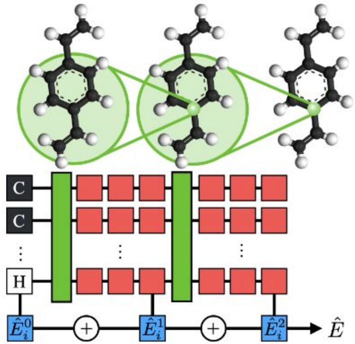
Relies on Query By Committee to determine structures with high uncertainty.

http://www.github.com/lanl/alf

Los Alamos NATIONAL LABORATORY

Smith, J. S.; Nebgen, B.; Mathew, N.; Chen, J.; Lubbers, N.; Burakovsky, L.; Tretiak, S.; Nam, H. A.; Germann, T.; Fensin, S.; Barros, K., Automated discovery of a robust interatomic potential for aluminum. *Nature Comm.* **2021,** *12, 1257*.

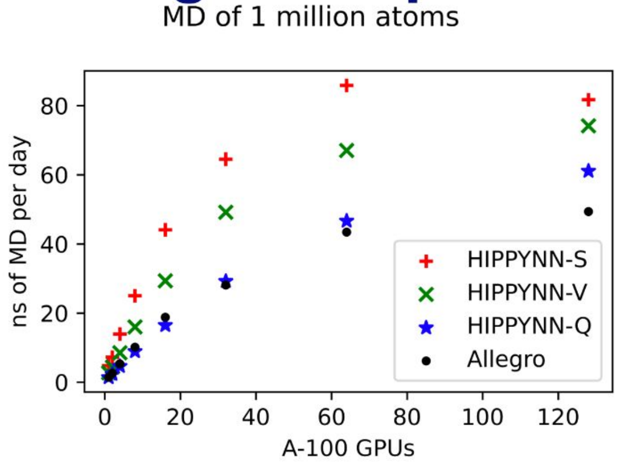
Active learned ML potentials can then be run large scale MD simulations of dynamic processes using Lammps



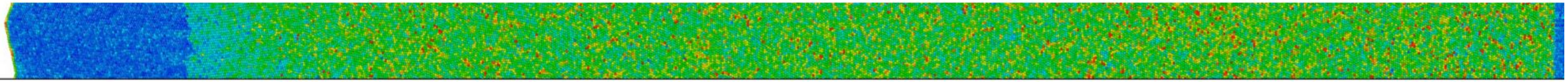
Left: the HIPPYNN neural network architecture.

github.com/lanl/hippynn

Right: Strong scaling of Lammps with the HIPPYNN neural network potential



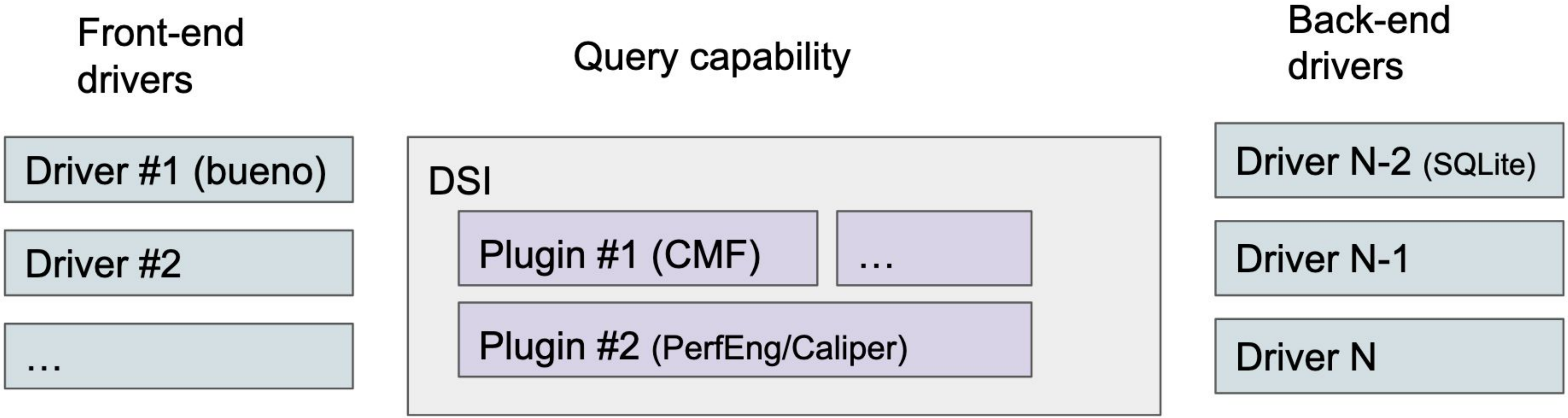
Bottom: multi-million atom shock simulation of Zinc performed with active learned HIPPYNN potential.



Data Science Infrastructure Supports Cross-Cutting Data Interoperability with Plugins and Drivers

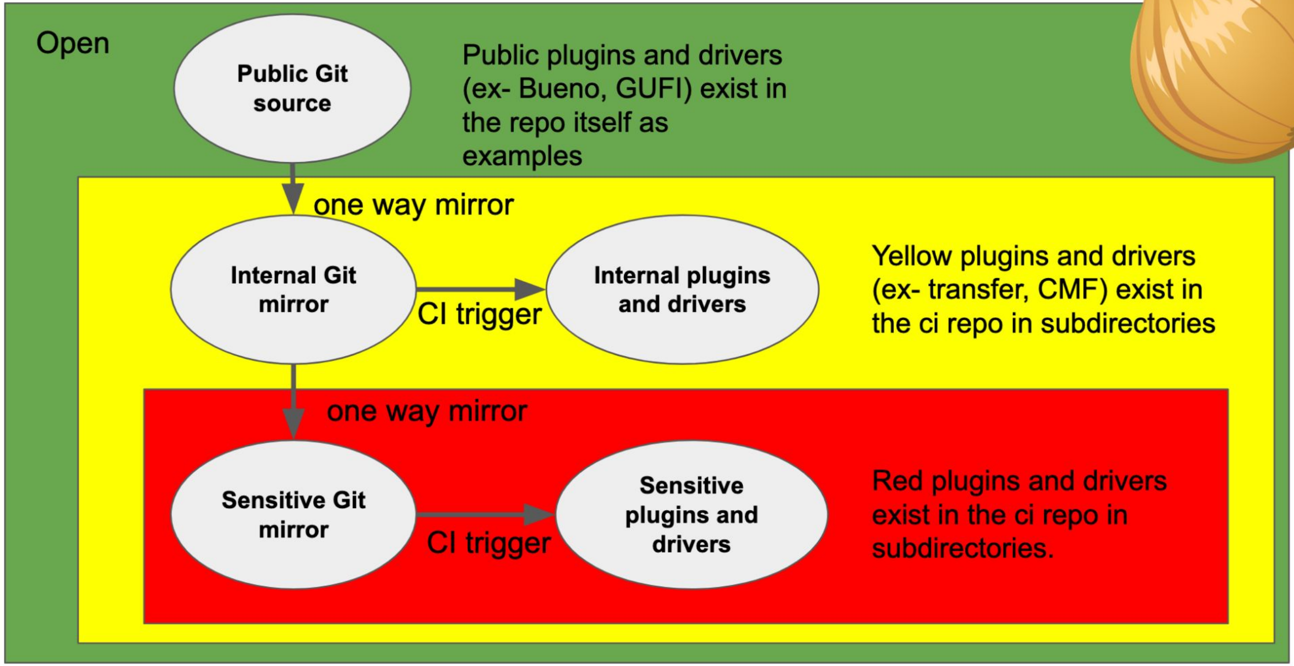
Not prescriptive – goal is to meet the users where they are

- Read their formats, work with their existing processes
- Provide templates of best practices



Data Science Infrastructure Supports Cross-Cutting Data Interoperability with POSIX-enforced security compliance

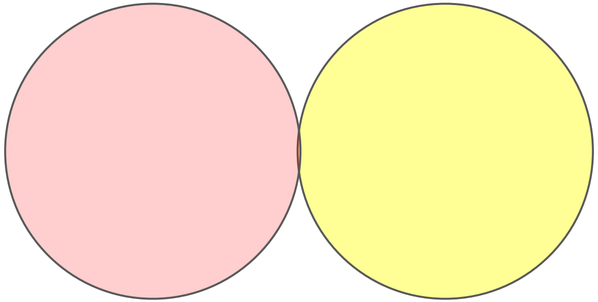
github.lanl.gov/lanl/dsi



Converged cluster success for ML depends on metrics TBD

1. Is there a framework to support it (working on it) ?
2. Is the linkage fast enough (bandwidth/latency) ?
3. Is the linkage secure enough (encryption sufficient) ?
4. Is the design advantage worth it (co-located or not) ?

HPC+Cloud



Summary

- DevOps is more general than tools, and it serves as common ground for a discussion about converged computing.
- Studying the needs for minimal convergence and maximal convergence reveals converged cluster design trade-spaces.
- The most challenging apps could technically run in the cloud, if some gaps are filled. Security is not considered here.