HEP-CCE All Hands Meeting
April 11-12, 2023

HEP Workflows and Leadership Facilities

Brookhaven National Laboratory

Argonne NATIONAL LABORATORY

Complex Workflows in HEP-CCE: Overview

- To an external observer, HEP community has many workflow technologies, which often serve specific use-cases, monolithic, and difficult to extend.
- Performance on leadership platforms is complex even for simple workflows; HPC workflows will only get more important, but increasingly harder
    - See few slides at end for illustrative challenges
- Opportunity to harmonize use of experiment-agnostic components, integrable solutions for extensibility and modularity on leadership platforms.
- Challenges:
    - No single approach as experiments at different maturity-levels, development cycle, and readiness to adopt external tools
    - Difficult to integrate new software into existing software infrastructure

Complex Workflows in HEP-CCE: Analysis

There are some valid underlying reasons for experiment specialization:

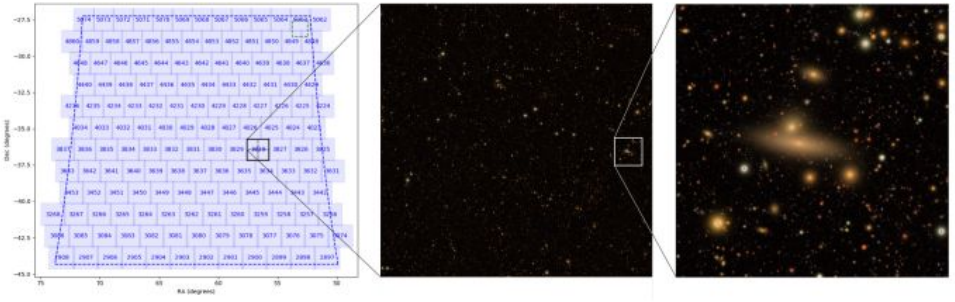
- Use cases with completely different workflow structures
- Resources with very different optimization goals
- Execution systems with fundamentally different capabilities

There are missed opportunities for interoperability and reuse across experiments

- Workflow components/sub-systems be reused
- Workflows be ported between systems
- Well defined performance, portability, provenance
- **Leverage HPC-workflow experience** on DOE Leadership Platforms

We discuss two prime examples ..

Extreme-scale Cosmology with LSST DESC



Enhancing portability and scalability of the LSST processing pipelines to run on DOE facilities

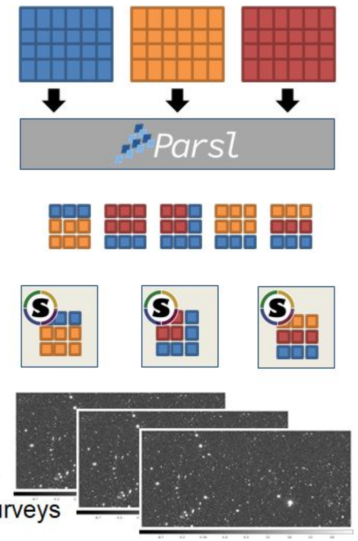
Parsl to compose scalable and portable workflows from the Python DRP components (run at NERSC)

189 sensors x ~10K catalogs

Node-sized bundles

Executed on Theta and Cori

Millions of core hours to deliver synthetic sky surveys



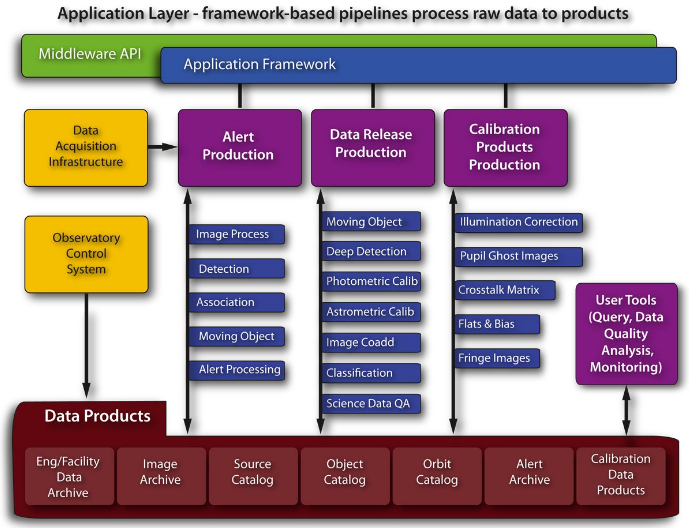
Data challenge 2 (DC2)

Simulating LSST images across ALCF and NERSC

Driven from Jupyter notebooks with containerized code

DESC et al, The LSST DESC DC2 Simulated Sky Survey, The Astrophysical Journal, 2021.

Villarreal et al. Extreme scale survey simulation with Python Workflows, eScience 2021



Rubin LSST image processing steps

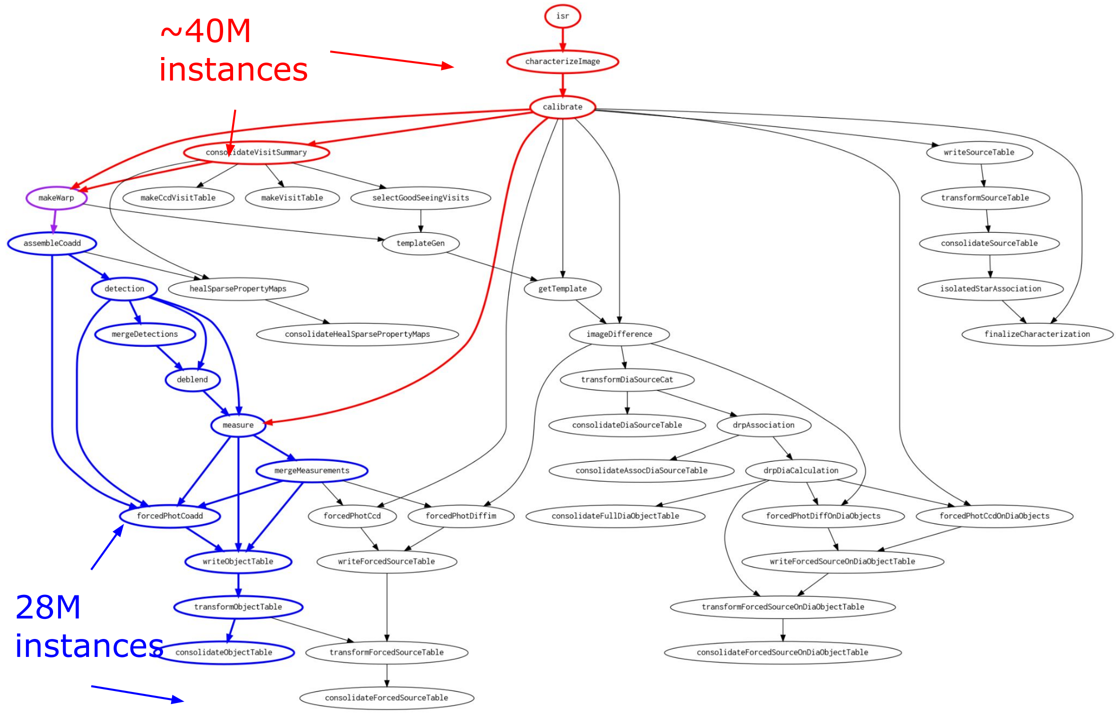
Rubin Data Management provides a package,ctrl_bps, to support processing of graphs on multi-node resources.

ctrl_bps accepts plugins based on specific workflow systems to manage execution, e.g., HTCondor, Pegasus, Parsl, PanDA.

DESC implemented a plugin based on Parsl for running on NERSC Cori and Perlmutter systems.

- Various Parsl executors: ThreadPool, HighThroughput, WorkQueue
- Rubin pipeline jobs have a large range of resource requirements: sub-GB to 10s of GB memory usage, few minute to several hours of runtime ⇒ WorkQueue is the most useful.Slurm and Local providers used

Brookhaven National Laboratory



Graph showing dependencies between task types for Rubin image processing. red operate on CCD-visits, blue tasks on patches, and purple on both. Credit: Jim Chiang

CW Working Group | **Early Exploration**

- Prototype effort to explore remote execution with funcX
    - funcX-based remote machine learning training for ATLAS (https://github.com/ValHayot/funcXtraining/)
- Explored middleware integration capabilities:
    - Developed RESTful API prototype for RADICAL-Pilot (high-performance task execution system).
- Early runs of adapted HEP frameworks on HPC production resources Theta & ThetaGPU (ALCF):
    - FastCaloSim GPU (https://github.com/cgleggett/FCS-GPU): adapted version of the ATLAS Fast Calorimeter Simulation framework.
    - HEPscore (https://gitlab.cern.ch/hep-benchmarks/hep-score): a benchmark based on containerized HEP workloads.

Brookhaven National Laboratory

CW Working Group | **Outputs**

- Representation from HEP experiments:
  - DUNE, LSST, DESC, Coffea, ATLAS/Panda
- Most systems are defining a task graph representation

  ⇒ Opportunity to choose a common task graph representation that can be ingested by others
- Support for streamlined remote execution

  ⇒ Building on tools like funcX to enable such execution

  ⇒ Representation of the cross system execution of workflows
- Support for malleable MPI jobs

  ⇒ Workflow systems (e.g.,RCT) able to exploit scheduler support and other interfaces for managing MPI jobs
- Guarantee maximum resource utilization

  ⇒ Allocation of resources with regards of task parameters (to avoid underutilization of resources with long-tail executing tasks)
- Diverse and distributed monitoring information

  ⇒ Common monitoring approach (consolidating information from various layers)

Formalizing HEP Requirements on LCF

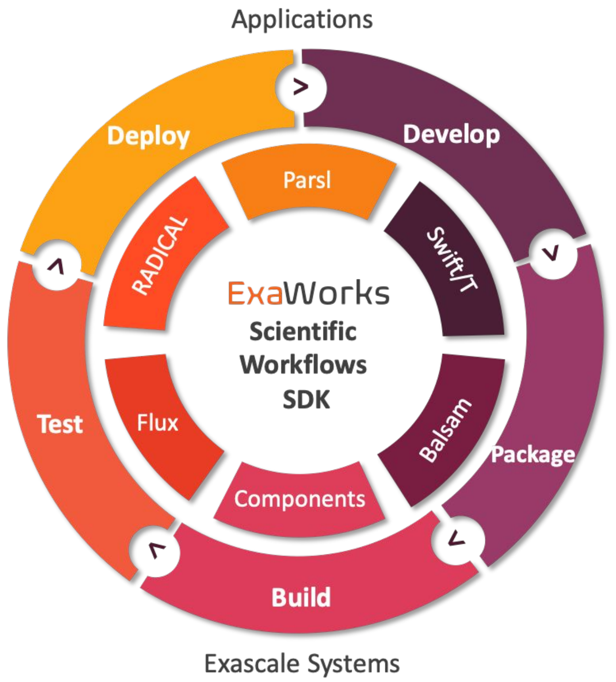
- Software requirement specification (SRS) complete (survey) for July 2023
  - Mikhail Titov (BNL), Valerie Hayot (ANL), others welcome
- Workflow requirements: (i) PST and/or DAG, (ii) groups of tasks with no dependencies, (iii) static, dynamic, adaptive, stream, etc.
- Workloads requirements: (i) number of tasks, (ii) static, dynamic, adaptive, stream, etc. (iii) heterogeneous/homogeneous.
- Task requirements: (i) kind (executable/function), (ii) amount of resources (CPU cores, GPUs, Memory).
- Data requirements: (i) input/output size total/per task, (ii) location of input/output data, (iii) staging, (iv) read/write I/O rate, (v) format
- Performance and scale requirements

ExaWorks is *not* developing a workflow system!

ExaWorks is providing a production-grade Software Development Kit (SDK) for exascale workflows

ExaWorks SDK democratizes access to hardened, scalable, and interoperable workflow technologies and components, for both developers and users

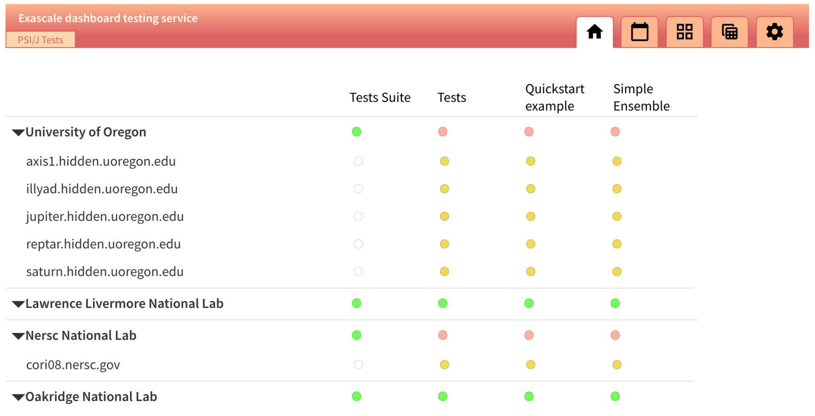
- E4S-based community policies for software quality
- Open community-based design and implementation process
- Scalability of components on **Exascale Systems**
- Standard packaging and testing
- Work toward shared capabilities in the SDK



Brookhaven National Laboratory

CI & deployment infrastructure for workflow tools

- Developed a GitLab CI infrastructure
- Set up CI at LLNL, ORNL, and ANL for the SDK components
- Testing PSI/J on an ECP testing cluster
- Developed a testing server to collect results of tests and enable dashboards and reporting from multiple sites
- Creating **Status Dashboard** to view what tests have been run on which systems



Exascale dashboard testing service

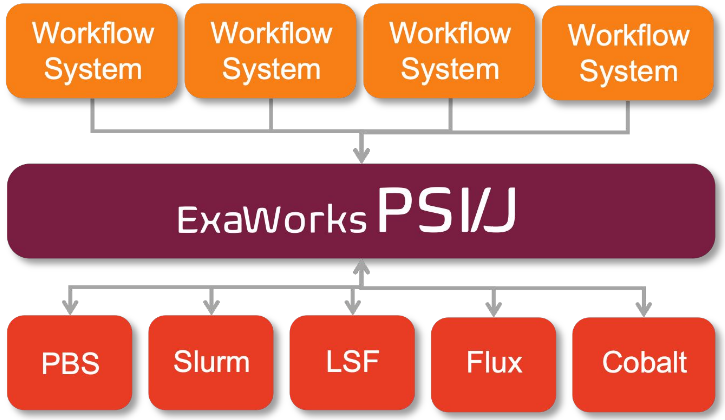
PSI/J Tests

	Tests Suite	Tests	Quickstart example	Simple Ensemble
University of Oregon	🟢	🔴	🔴	🔴
axis1.hidden.uoregon.edu	○	🟡	🟡	🟡
illyad.hidden.uoregon.edu	○	🟡	🟡	🟡
jupiter.hidden.uoregon.edu	○	🟡	🟡	🟡
reptar.hidden.uoregon.edu	○	🟡	🟡	🟡
saturn.hidden.uoregon.edu	○	🟡	🟡	🟡
Lawrence Livermore National Lab	🟢	🟢	🟢	🟢
Nersc National Lab	🟢	🔴	🔴	🔴
cori08.nersc.gov	○	🟡	🟡	🟡
Oakridge National Lab	🟢	🟢	🟢	🟢

Brookhaven National Laboratory

PSI/J: Portable Submission Interface for Jobs

- PSI/J Python binding
  - Python library with asynchronous interface for interacting with job schedulers
  - Supports Slurm, LSF, Cobalt, Flux; PBS and other schedulers coming
  - Architected to allow seamless contributions from the community
- Eventually the PSI/J specification will cover more advanced job-management functionality, such as job submission on remote clusters ("layer 1").
- We have integrated PSI/J into RADICAL-Cybertools and Parsl



Brookhaven National Laboratory

Why ExaWORKS Components?

1) Reduce overheads
   a) Share common components and reduce development, testing, maintenance, etc. costs
   b) Build upon robust capabilities
   c) Enhance portability

2) Establish an ecosystem of complementary capabilities
   a) Simplify adoption of different workflow tools for different problems
   b) Improve user experience

3) Enable innovation
   a) Focus on innovation at higher levels of the workflows stack

4) Post-ECP: Central to “Workflows and Application Services” seed

Brookhaven National Laboratory

Summary

- HEP-CCE can guide adoption of HPC workflows on LCFs and support new workflow development (e.g., DUNE, GPU workflows)
- Exploration of HEP workloads (and corresponding frameworks) and their test runs on HPC testbeds is ongoing.
- The CW group will construct a curated list of workflow requirements on a per-experiment basis to determine specific and overlapping needs in the HEP community.

HPC-Workflows: Challenges at Scale

- Managed extreme heterogeneity of applications and resources **concurrently**
  - **Application:** Type of task: executable/function
    - Size and duration of tasks: 1-N cores/GPUs
    - Type of parallelism: thread/process, OpenMP, MPI
  - **Resource:** Type, scale, cost of resources used by tasks
    - Edge-to-Exascale: Computing across the heterogeneous continuum

- Redefine what it means to be performant!
  - Measure & optimize **collective performance**, not single task performance
    - More than makespan optimization; non-traditional trade-offs
  - Complex interplay of traditional computational & scientific performance
    - AI-coupled-HPC WF **effective** performance > 10^N traditional WF
  -

Brookhaven National Laboratory

HPC-Workflows: Challenges at Scale (2)

- Different layers of resilience:
  - Management system itself;
    - Control the state of its components and the number of instances per each component;
  - Management of allocated resources;
    - Track the resources performance and adjust their availability for payload placement;
  - Management of execution processes (i.e., computing tasks).
    - Trade-off between level of resilience and throughput
  - Makespan optimization under performance uncertainty