PanDA and RADICAL-Pilot Integration: Enabling the Pilot Paradigm on HPC Resources

Andre Merzky, Matteo Turilli, Pavlo Svirin

CHEP 2018, Sofia, Bulgaria

Motivations

- Removing the need for users to acquire resources and schedule payloads on HPC using their own means
- Investigate the advantages of scale, flexible execution and interoperability outside production constraints on HPC resources.
- Supporting backfill/regular queueing on Titan; concurrent execution of workloads on the same pilot; distribution of jobs on Titan and Summit supercomputers.

RADICAL Pilot

RADICAL-Pilot (RP) is a runtime system designed to execute multiple types of scientific workloads on pilots instantiated on one or more resources.

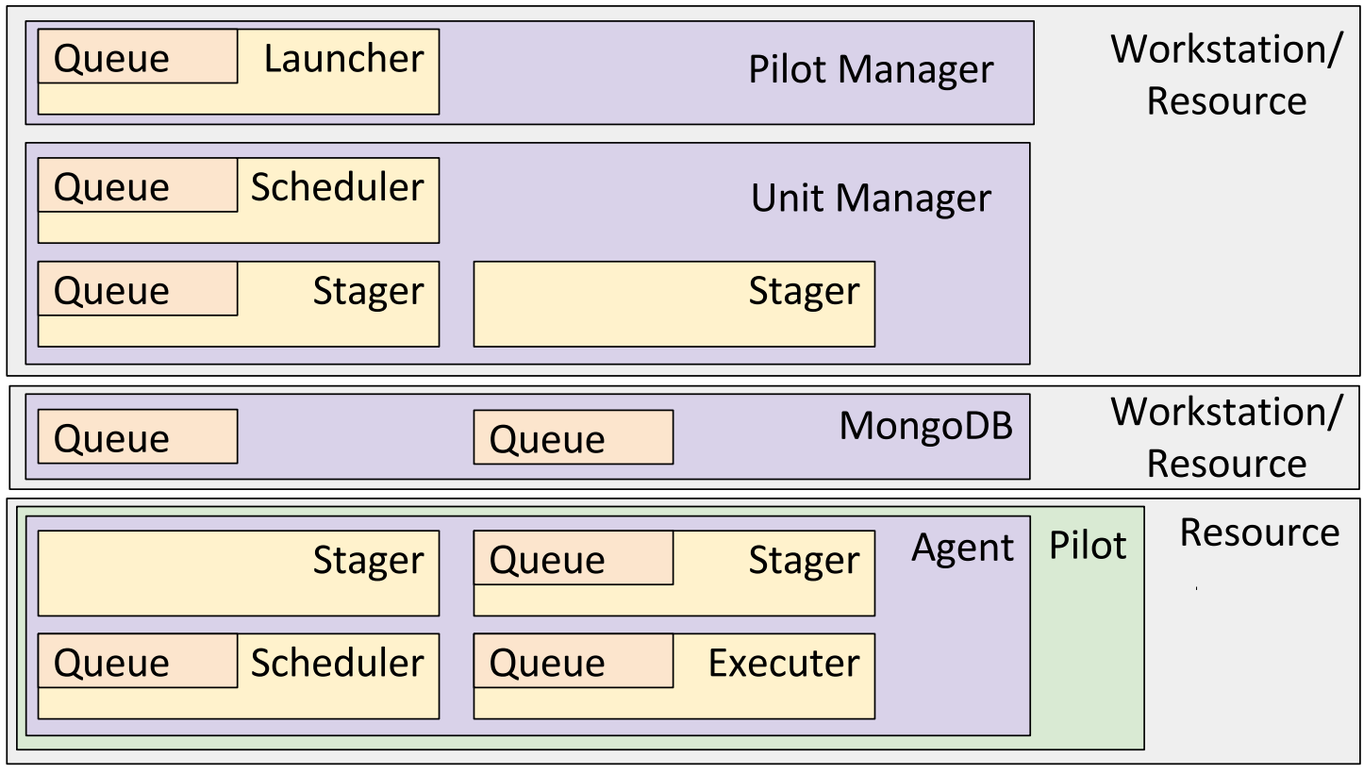
- developed by a RADICAL group of Rutgers university
- enables the description of generic workloads with one or more scalar, MPI, OpenMP, multi-process, and multi-threading tasks.
- workloads can be executed CPUs, GPUs and other accelerators, on the same pilot or across multiple pilots.
  - support for both CPU and GPU, exclusive or concurrent.
- optimization and dedicated scheduler.
- executions up to 131K cores on Titan.

RADICAL Pilot architecture

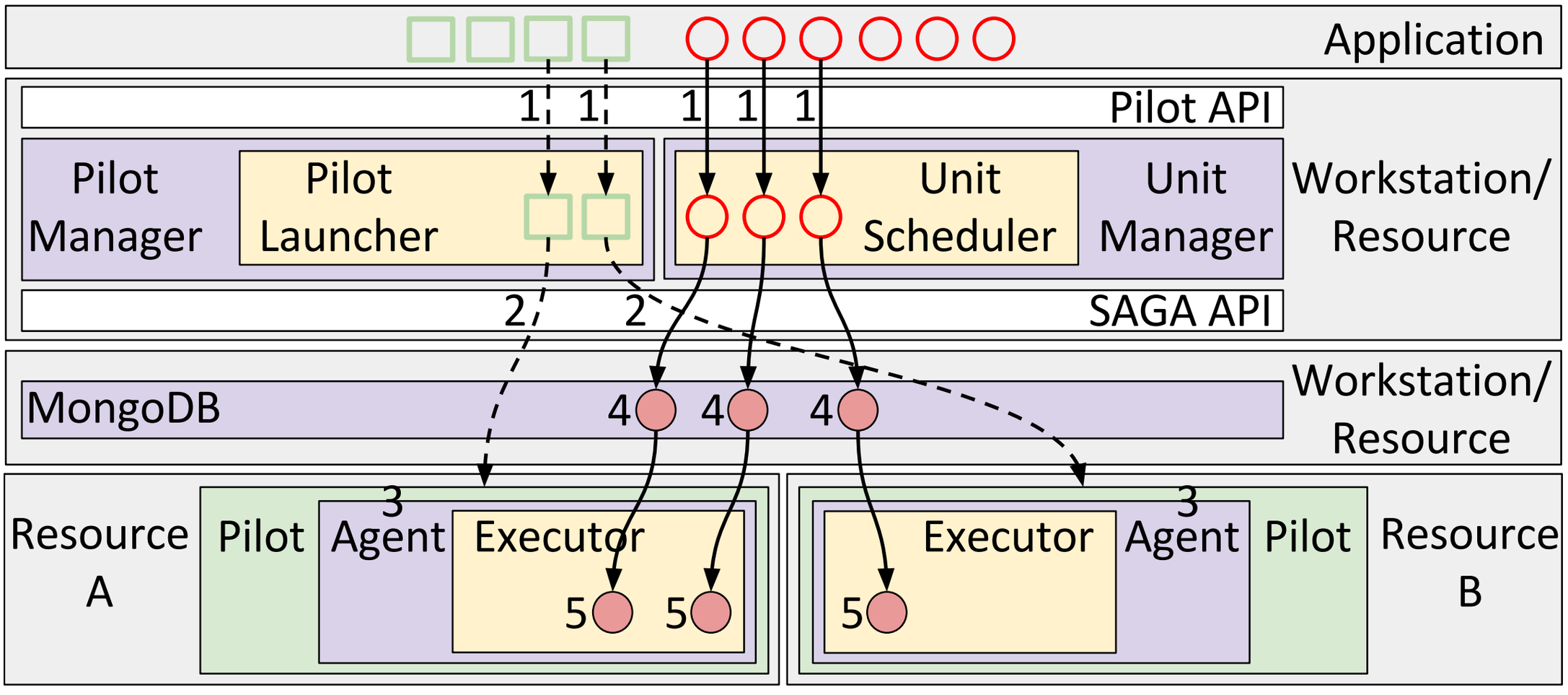
- a distributed system with four modules: PilotManager, UnitManager, Agent and DB
- modules can execute locally or remotely, communicating and coordinating over TCP/IP, and enabling multiple deployment scenarios

Pilots: placeholders for computing resources, where resources are represented independent from architecture and topological details.

Computing Units are units of work, specified as an application executable alongside its resource and execution environment requirements.



RADICAL Pilot Workflow

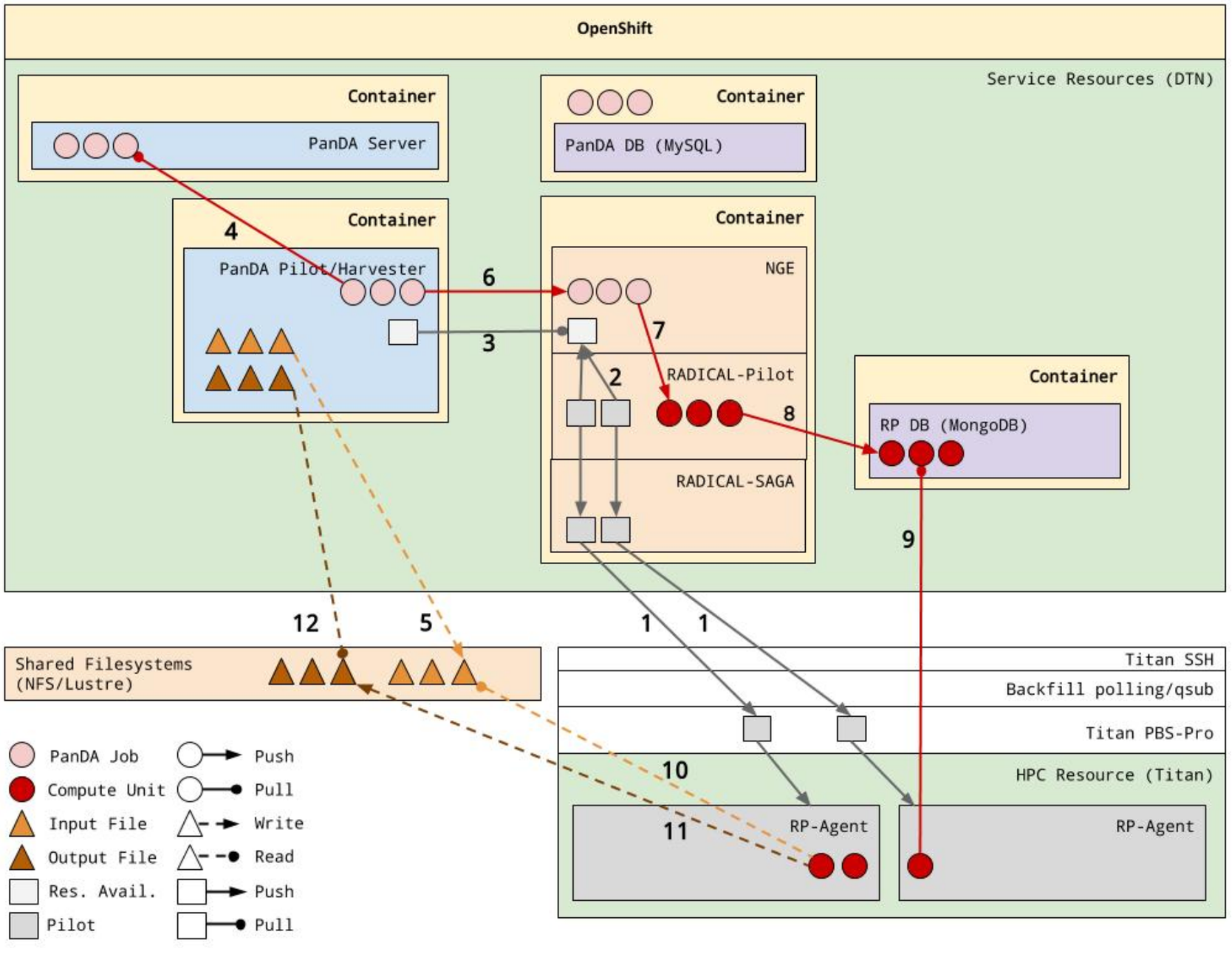


Next Generation Executor and integration with PanDA

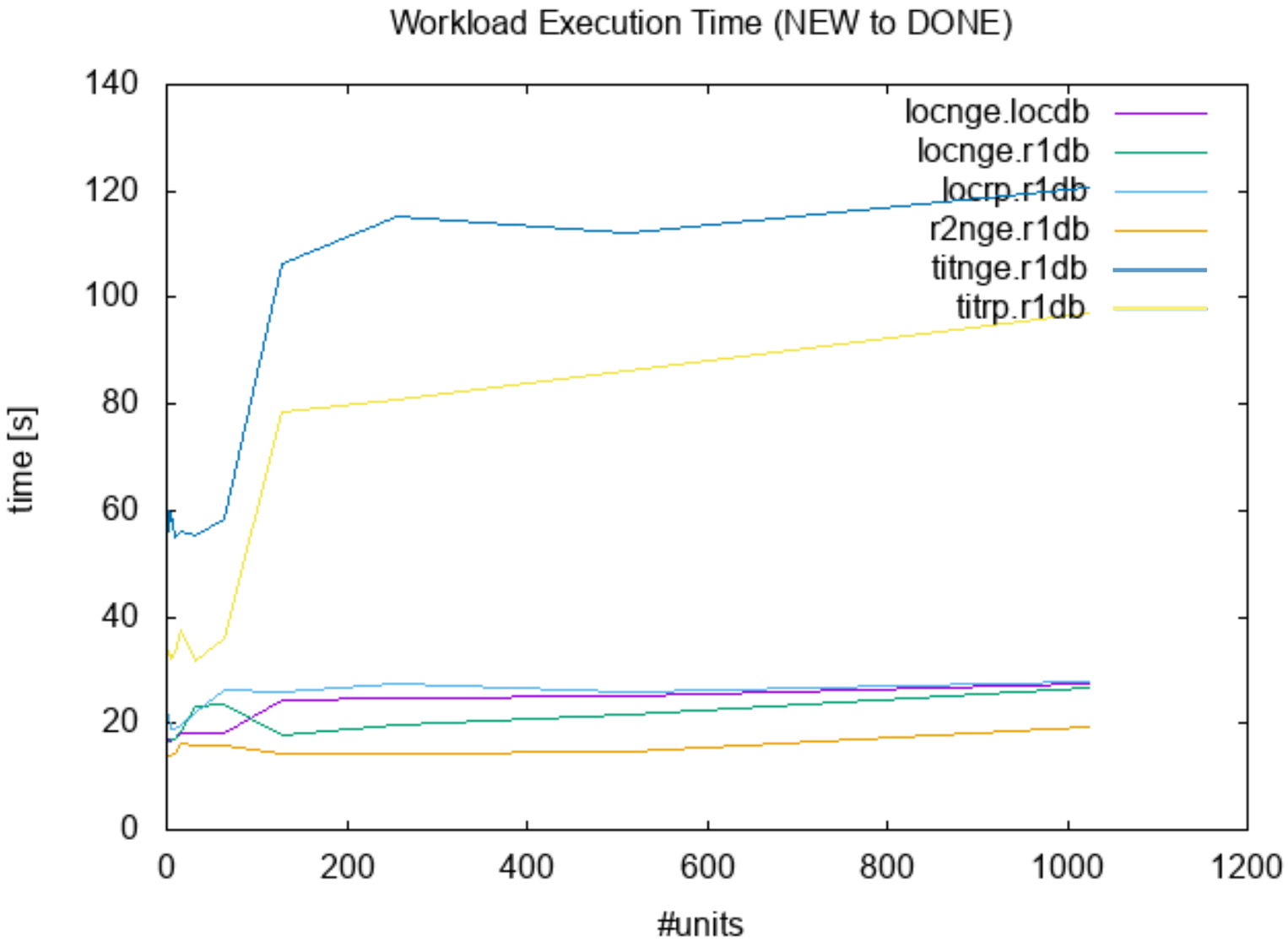
- RADICAL Next Generation Executor (NGE):
  - Persistent service with a REST interface hosted on a Titan's login node.
  - Uses a database to hold resource and task states: #cores and walltime; submitted, executing done.
  - internally uses Radical-Pilot as workload executor.

- Investigate options for integration of Harvester (Next-generation edge service for PanDA, see presentation by Tadashi Maeno) and NGE in terms of BigPanDA project:
  - Investigate the scalable, flexible execution and interoperability outside production constraints
  - Support both backfill and regular queueing capabilities at the same time on Titan
  - Support different types of workloads
  - Support concurrent execution of multiple workloads on the same pilot
  - Loose coupling among independently developed services deployed at ORNL
  - Enable concurrent distribution of jobs on multiple OLCF resources, e.g., Titan and Summit

PanDA-NGE Integration: Architecture

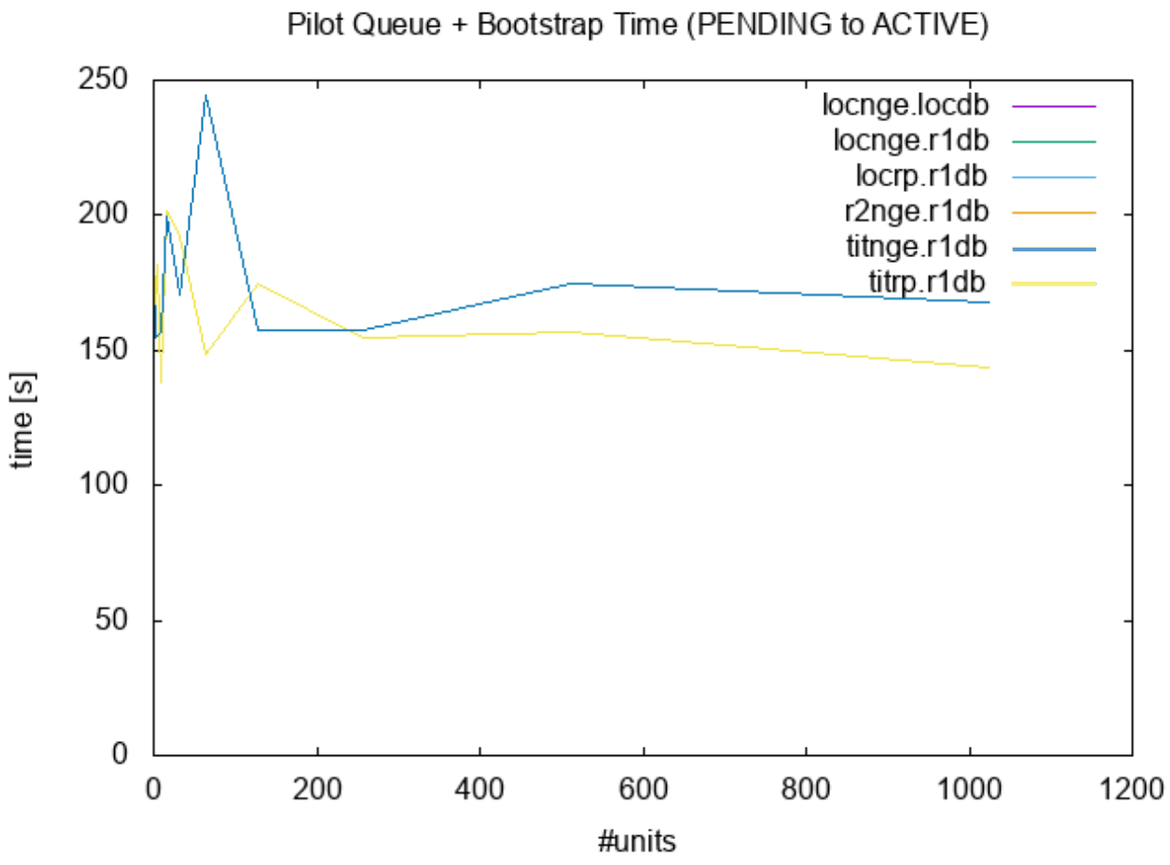


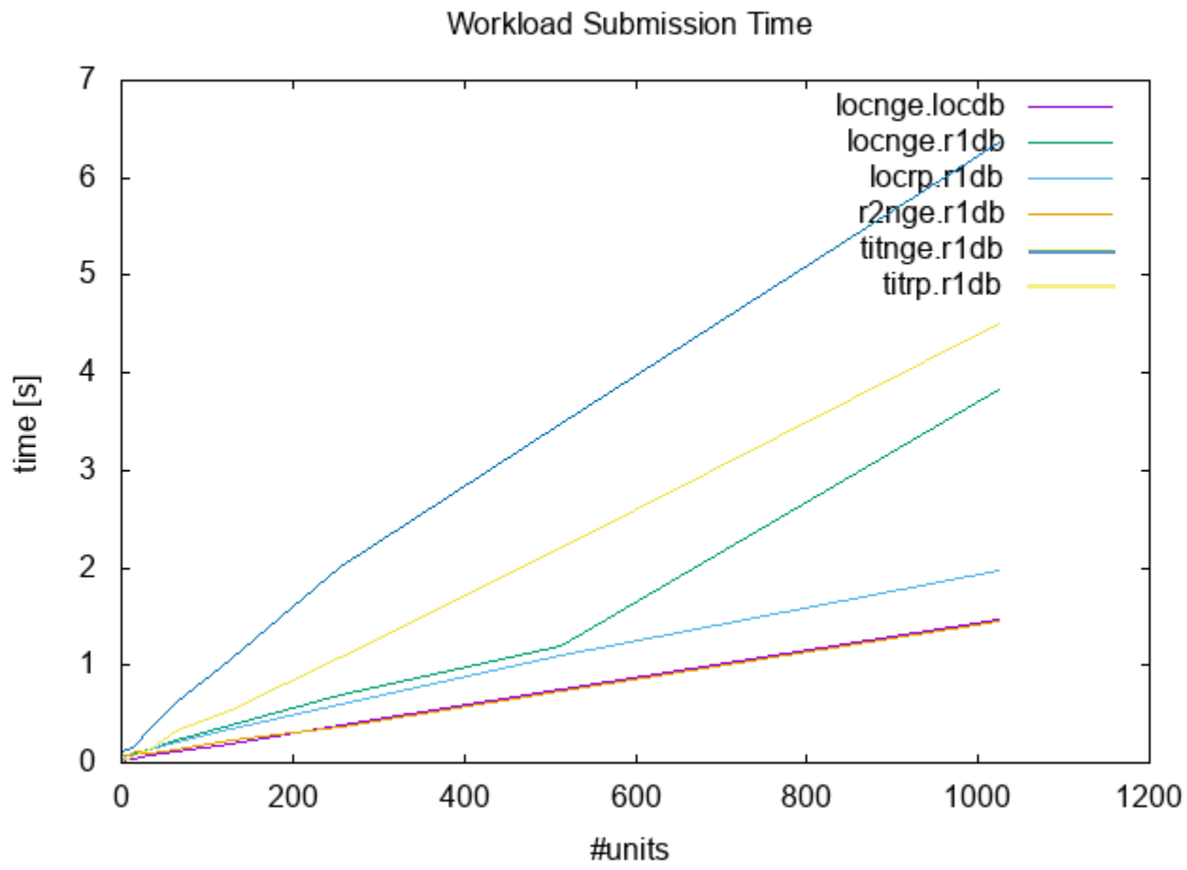
NGE Performance: Null Workload Execution



- Characterization and comparison of null workload execution time with NGE and RP in different deployment scenarios.
- NGE and RP have similar performance

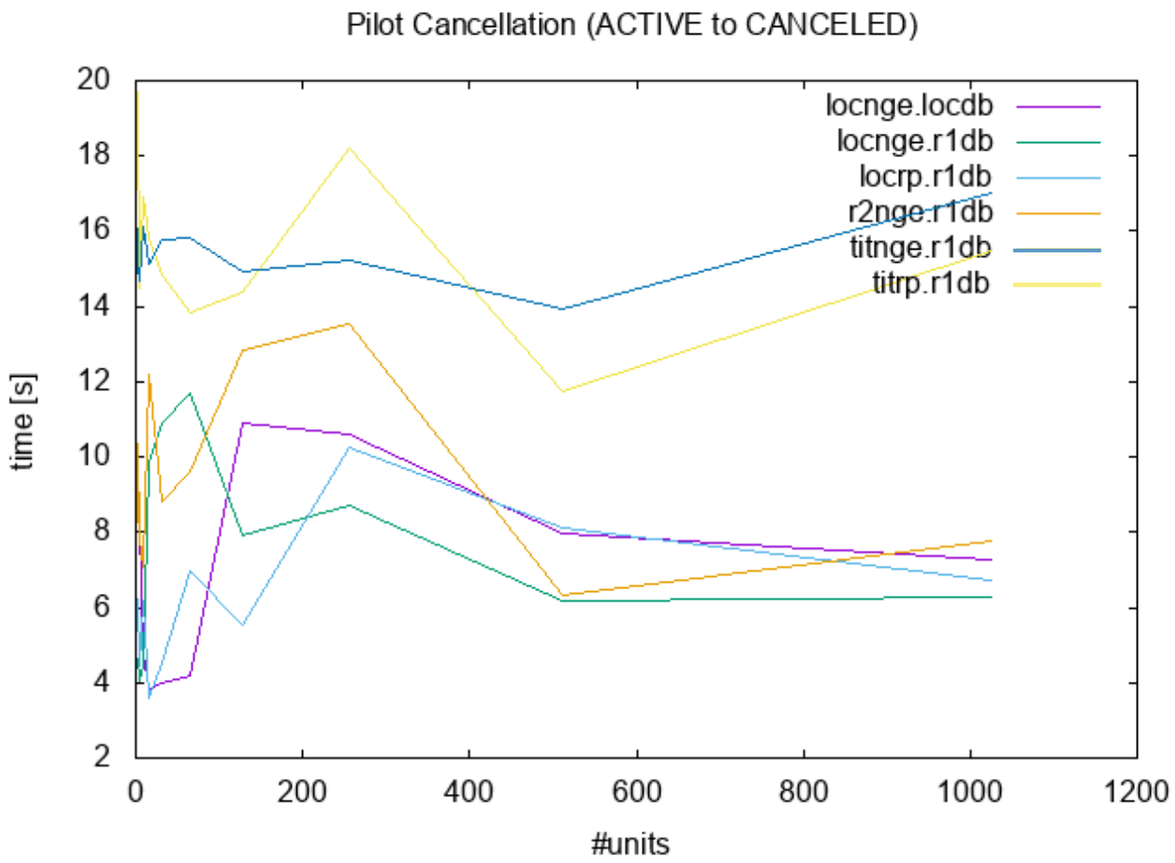
NGE Performance: Resource Acquisition

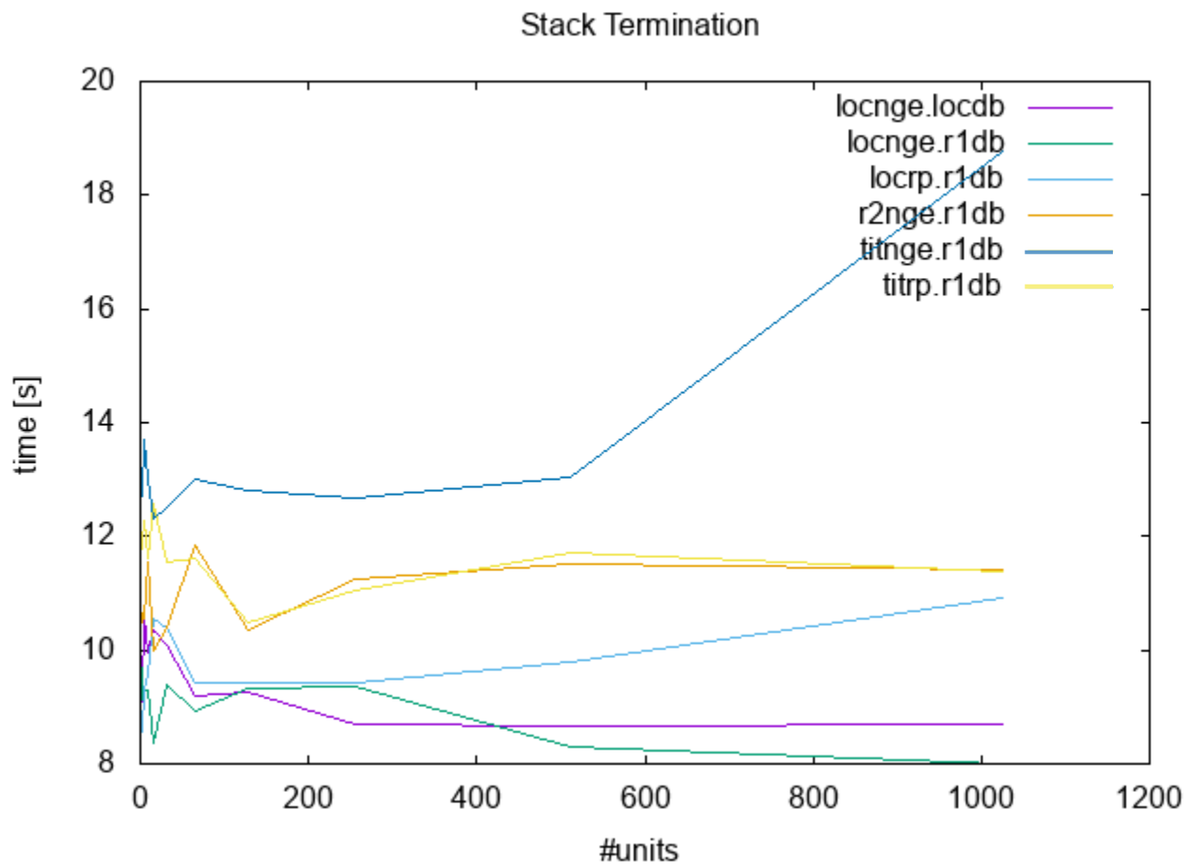




Pilot queue and bootstrap & workload submission time: NGE and RP have analogous performance

NGE Performance: Termination

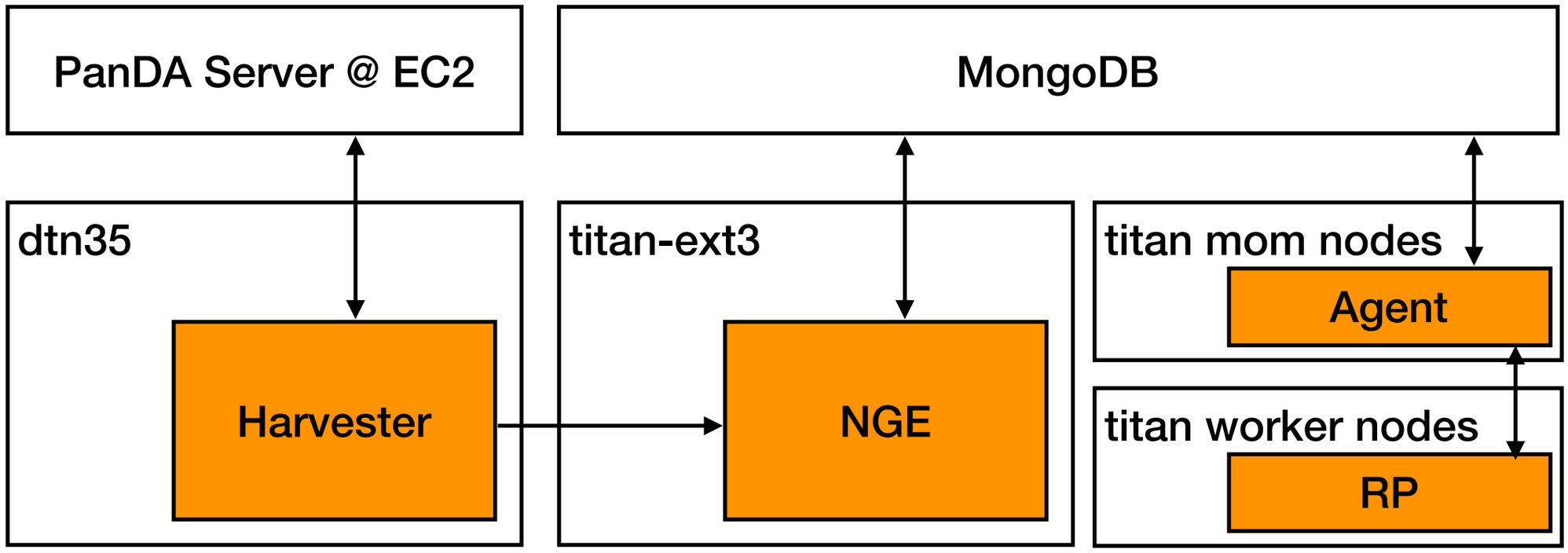




Pilot cancellation and stack termination time: NGE and RP have analogous performance.

Successful setup on Titan

- An integration between NGE and PanDA has been successfully tested on Titan
- All of the elements of the chain (job submission to PanDA Server, job fetching by Harvester, submission and execution on Titan through NGE) working



Conclusions and future plans

- Integration between PanDA and NGE was set up and tested
- Several workflows have been tested on Titan with PanDA/NGE:
  - Ready to start end-to-end execution of Molecular Dynamics use cases.
  - ATLAS simulation jobs in a Singularity container

Future plans:

- Extending profiling and analytics capabilities of NGE stack to Harvester stack, enabling fine-grained characterization of MD workload executions.
- Support Summit on NGE: no change should be needed in Harvester, showing the potential for the isolation of concerns enabled by the design of the Harverster/NGE integration.
- Port design features to Yoda on the base of the results of our characterization.