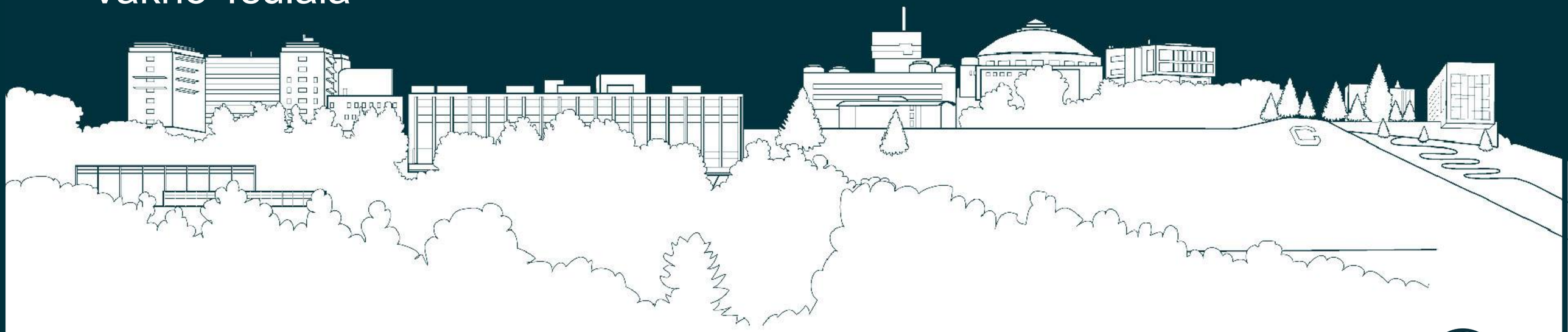
Raythena integration with Harvester, Pilot, and Panda

Doug Benjamin, Miha Muškinja,
Paul Nilsson, Asoka De Silva,
Vakho Tsulaia

ADC WFMS weekly
Thursday, 28 May 2020



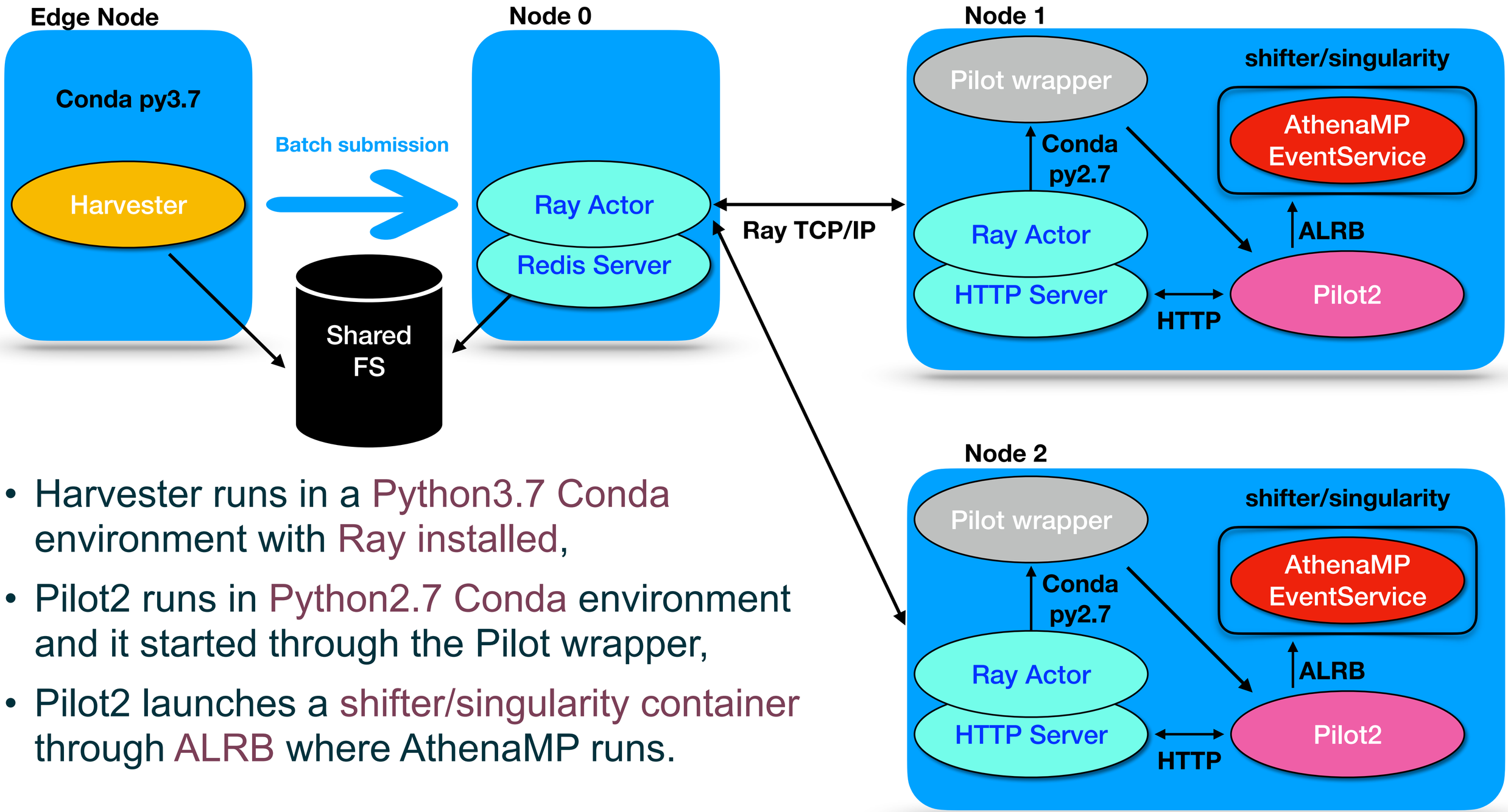
BERKELEY LAB

ATLAS EXPERIMENT

Introduction

- Follow up to Raythena discussion in April,
- Raythena orchestrates Event Service (ES) Jumbo Jobs on HPCs based on the Ray framework and utilizes Harvester and Pilot2,
- Main developments since last presentation:
  - Implement and validate Raythena ES plugin for Pilot2,
  - Adapt Raythena and Pilot2 to the new workflow where Pilot2 starts a container through ALRB and runs AthenaMP in it,
  - Fix log file stage-out / zipping for ES jobs in Pilot2,
  - Test the new scheme on Cori through the Panda/Harvester queue,
  - Validate that zipped **HITS and log files get staged** out (i.e. we located HITS transferred to SLAC from Cori),
  - Get successful merge jobs with generated output.

Raythena Scheme on Cori KNL



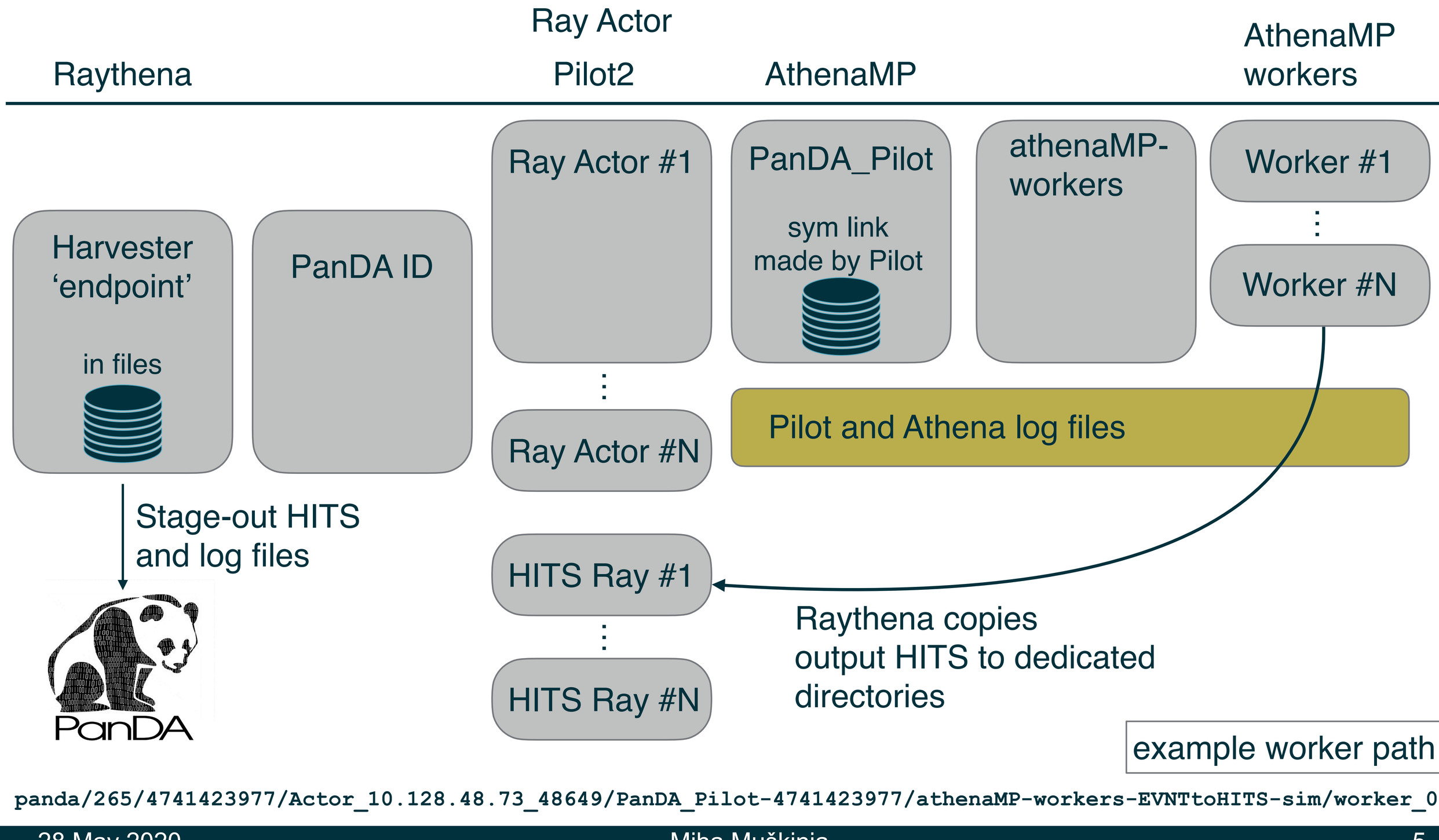
- Harvester runs in a Python3.7 Conda environment with Ray installed,
- Pilot2 runs in Python2.7 Conda environment and it started through the Pilot wrapper,
- Pilot2 launches a shifter/singularity container through ALRB where AthenaMP runs.

Common Code for all Packages

- Raythena: https://github.com/PanDAWMS/ray
- Pilot2 (currently using the ‘next’ branch): https://github.com/PanDAWMS/pilot2
- NERSC_Cori_Raythena Harvester/Pilot configuration: https://github.com/PanDAWMS/harvester_configurations/tree/master/NERSC_Cori_Raythena

- Some site-specific configuration is needed,
- Minimal code changes needed in Pilot2.

Directory Structure



`panda/265/4741423977/Actor_10.128.48.73_48649/PanDA_Pilot-4741423977/athenaMP-workers-EVNTtoHITS-sim/worker_0`

Raythena tests on Cori and BNL

- Currently running Raythena Event Service test tasks on Cori: https://bigpanda.cern.ch/tasknew/21433080/,
- Jobs finish successfully and zipped output HITS and log files staged-out and merged,
- Using atlas/athsimulation:21.0.109_100.0.2-noAtlasSetup image,
  - Could not get older images (e.g. 21.0.15) to work with ALRB + shifter,
- Plan to also run a test Event Service Raythena Harvester queue at BNL,
  - This would enable us to test both shifter and singularity containers,
  - At BNL, so far successfully tested a multi-node stand-alone Ray example, did not yet launch Raythena.

Example job

Latest event range records:

File ID	Min event	Max	Status	ProcessID	Attempt	DataSet ID	ObjectStore ID	Jobset ID
21238610752	2076	2076	finished	2076	9	313933299		4741542317
21238610752	2066	2066	finished	2066	9	313933299		4741542317
21238610752	2064	2064	finished	2064	9	313933299		4741542317
21238610752	2062	2062	finished	2062	9	313933299		4741542317
21238610752	2061	2061	finished	2061	9	313933299		4741542317

Issue With the --inputEVNTFile Argument

- AthenaMP is started with an '—inputEVNTFile' argument,
  - Use to initialize EventSelector,
  - Not actually processed in an Event Service job,
- Value for this argument comes from the job definition, and is always a local file in the harvester endpoint, e.g.: —inputEVNTFile=file1.EVNT,
- This works because Pilot creates links to input files in Athena run directory during the 'stage-in' process,
- Once AthenaMP is initialized and forked into worker processes, worker processes receive input files from Pilot2 through yampl,
  - Path has to be absolute because input files are not copied to worker directory,
- However, starting Athena with —inputEVNTFile=file1.EVNT and later giving same file to worker process with a different path causes it to fail opening the file.

Solution for the --inputEVNTFile Argument

- Worker processes run fine if the same file path is used in AthenaMP ‘—inputEVNTFile’ and in the yampl communication,
- We cannot give absolute path in job definition, because directory name is dynamical,
- **Current solution**: Raythena modifies the Sim_tf.py command before sending it to Pilot to change the local path to absolute path in ‘—inputEVNTFile’,
- **Pros**: it works,
- **Cons**: very fragile because Sim_tf.py is manipulated with a RegExp and it is impossible to cover all valid ways of writing down the —inputEVNTFile argument with values,
- With this solution, it is not necessary for Pilot to copy / stage-in EVNT files,
- In any case, Pilot does not need to add files to Pool File Catalog,
- Finding a better solution for this would be desired,
- Good to understand why reading the input file fails in the first place.

Next Steps

- Find a better solution for —inputEVNTFile argument,
- Run merged output through Reco; produce AOD, check for duplicates,
  - Will contact ADC DPA team to setup a validation task (corresponding to s3558 tag),
- Upscale Event Service tasks to >100 nodes for basic benchmark,
- Would like to use a newer R21 release for certification (e.g. 21.0.109).

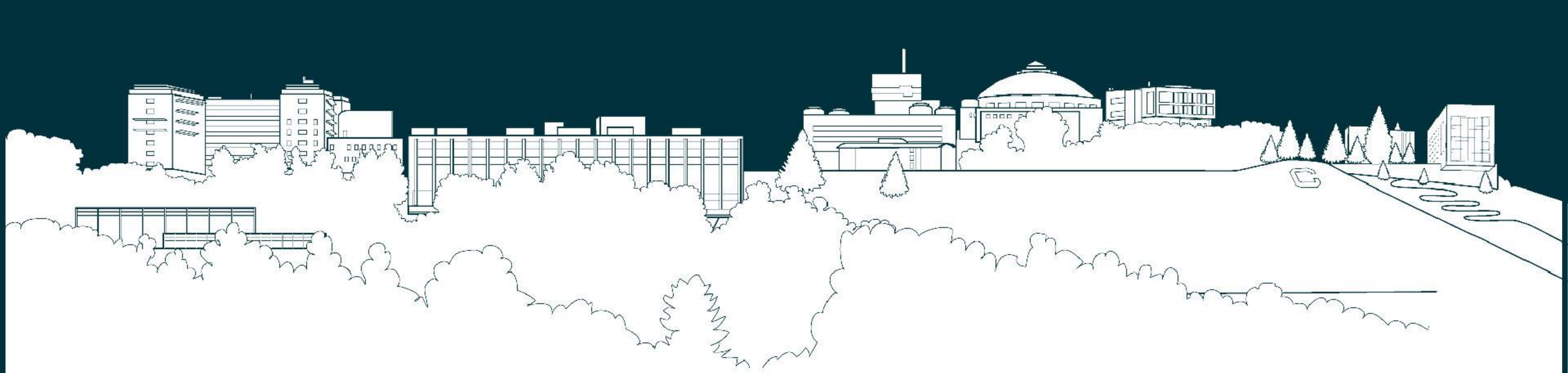
- **Stage 1:** Get ready to certify Raythena for production (0.5 FTE for 3 months)
  - Validate Raythena pilot plugin in real Harvester tests, make sure that Harvester can stage-out output,
  - Once ready, adopt the new Pilot version where instead of running in a container, Pilot launches the container,
  - Error handling in Raythena: make sure that Ray driver application is aware of any failing nodes and develop a mechanism to stop the job if certain number of nodes fail,
  - Validate error propagation to Harvester,
  - Validate production of merged output and run through standard Reco,
- **Stage 2**: certification, no duplicated and no events are lost (start in summer?),
  - Infrastructure for certification comes for free (e.g. Event Index),
- **Bonus** developments that would be nice:
  - Run an AthenaMP instance with fewer processes on head node if feasible,
  - Develop merging on-the-fly in Raythena (could use head node for merge jobs).

16 April

Summary

- Successfully running Event Service jobs on Cori with the new Pilot2 setup where container is started by Pilot2 through ALRB,
- Stage-out and merge jobs work well,
- Setting up a Raythena Event Service queue in BNL,
- On track to start with certification in few months if no show-stoppers.

Backup



Machine Learning and Hyperparameter scan with Ray Tune

- We already have a workflow where we can connect a ray cluster on an HPC (Raythena) to Panda through harvester,
  - Ray cluster across multiple sites not feasible because of security issues with Redis,
- We could use this machinery also for other workflows suitable for HPC,
- One interesting example is Ray Tune, a multi-node distributed hyperparameter scan,
  - Supports any machine learning framework, e.g.: PyTorch, XGBoost, MXNet, Keras,
- Enable users to submit very large hyper parameter scan jobs to HPCs via Panda.