Merlin

Machine Learning Workflow for HPC

Joe Koning

Luc Peterson

The LBPM WF Team

LLNL

Aug 10-12, 2021

LLNL-PRES-825612

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Outline

Technical Overview

Current Direction

Future Direction

Merlin is the work of many amazing people from across LLNL

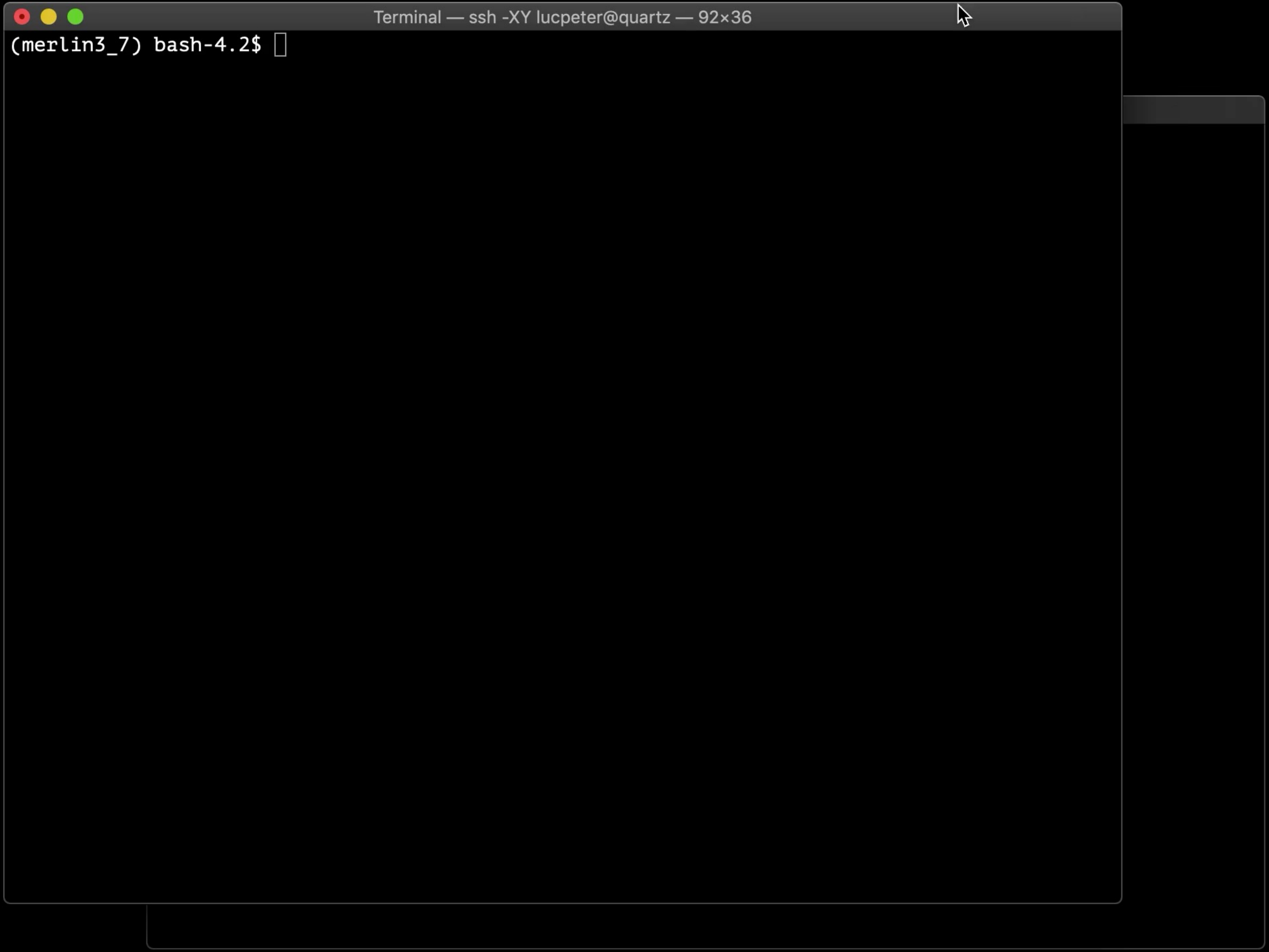
- Ben Bay
- Joe Koning
- Jeremy White
- Jessica Semler
- Peter Robinson
- Frank Di Natale
- Bogdan Kustowski
- Vic Castillo
- Yamen Mubarak
- Kevin Athey

- Aiden Keogh
- Brian Spears
- Timo Bremer
- Rushil Anirudh
- Jay Thiagarajan
- Jim Gaffney
- Gemma Anderson
- John Field
- Scott Brandon

- Dave Fox & Livermore Computing (LC)
- Dong Ahn, Steve Herbein & the Flux Team
- Brian Van Essen & the LBANN Team
- Dan Laney, Becky Haluska & the Workflow Team
- Cyrus Harrison & the Conduit team
- Steve Langer & many friendly and patient users

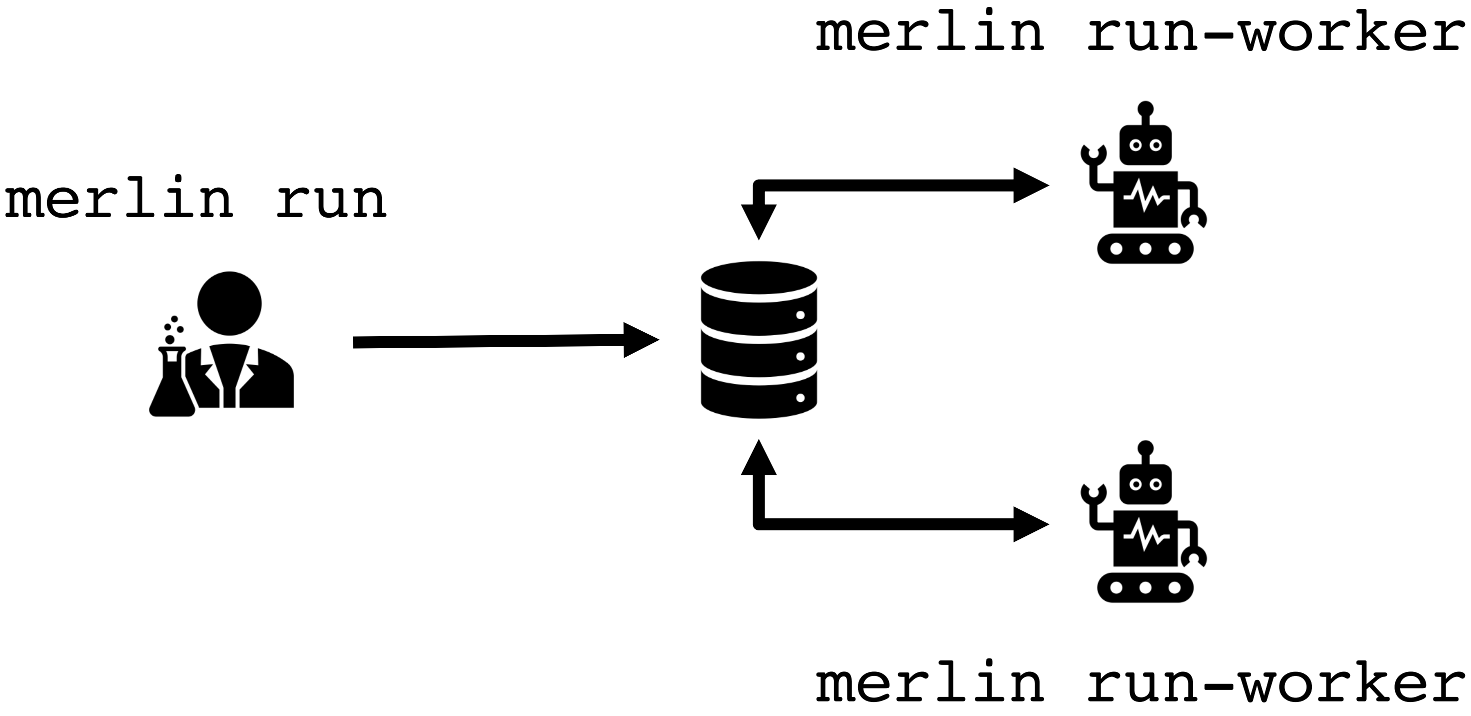
Funding: Learning Based Predicted Models LDRD-SI (Spears PI)

technical overview



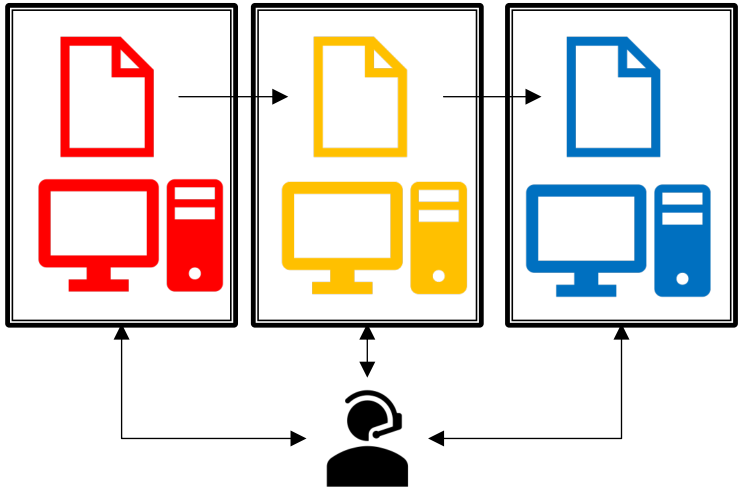
```
(merlin3_7) bash-4.2$ 
```

What just happened? A producer-consumer workflow



Traditional HPC workflow systems: External or Internal Coordination

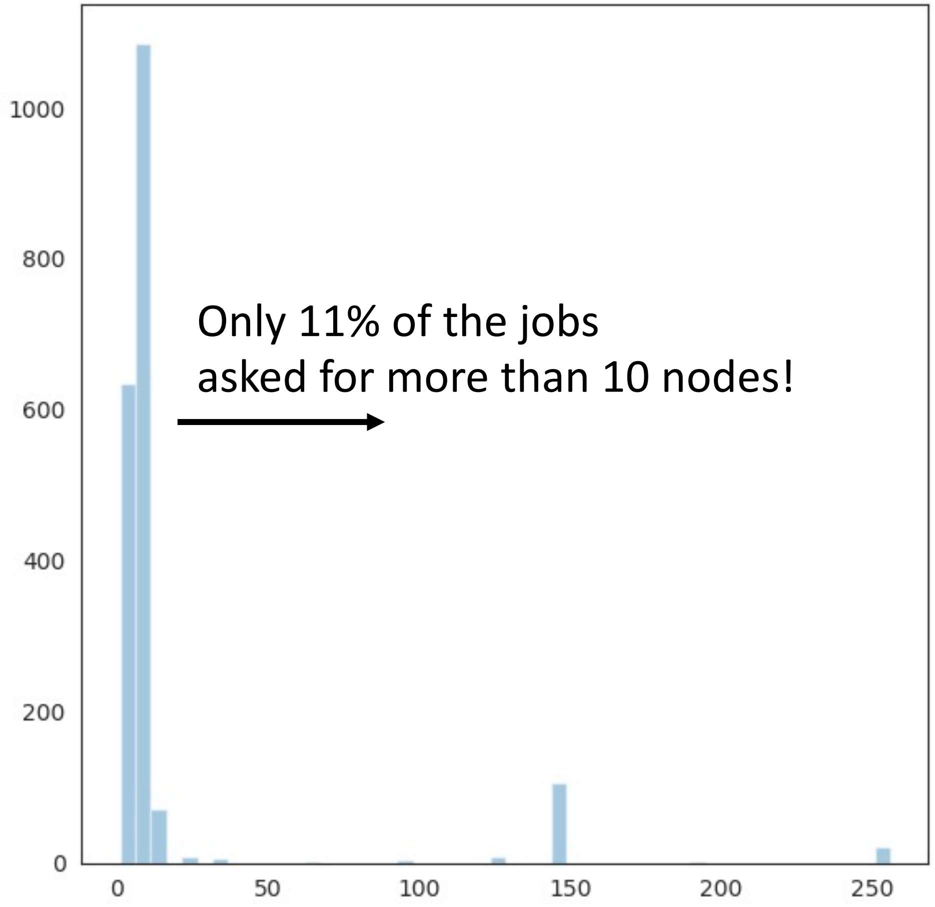
External coordination
Independent batch jobs tied to tasks



e.g. maestro

There is a large demand for small jobs

Lassen Queue 5/14/2020



Number of Queued or Running Jobs

Only 11% of the jobs asked for more than 10 nodes!

Requested Node Count

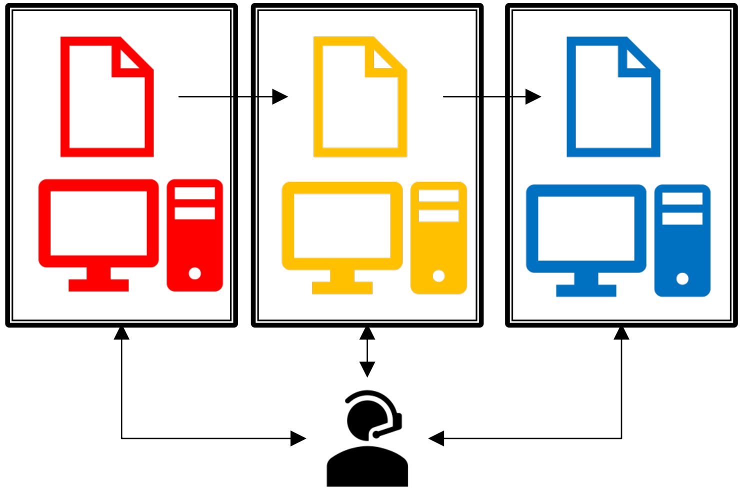
Problem:

HPC schedulers are more efficient with fewer large jobs than with many small jobs

As our systems get better, we can do more with fewer nodes. El Capitan…

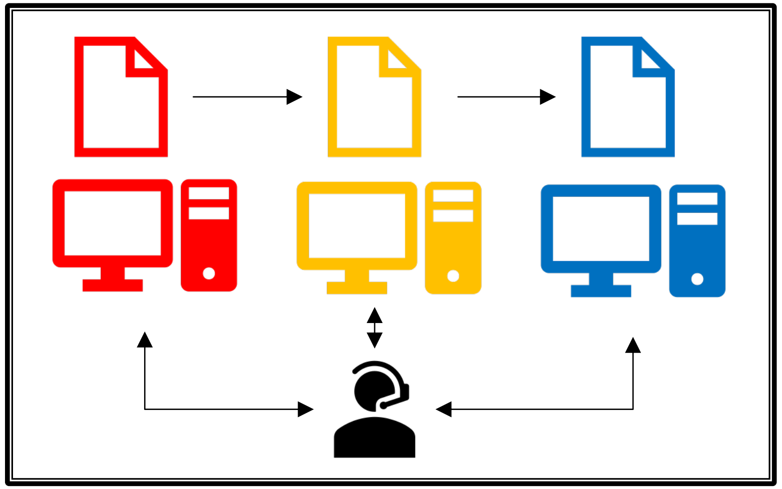
Traditional HPC workflow systems: External or Internal Coordination

External coordination
Independent batch jobs tied to tasks



e.g. maestro

Internal coordination
Large batch jobs for multiple tasks

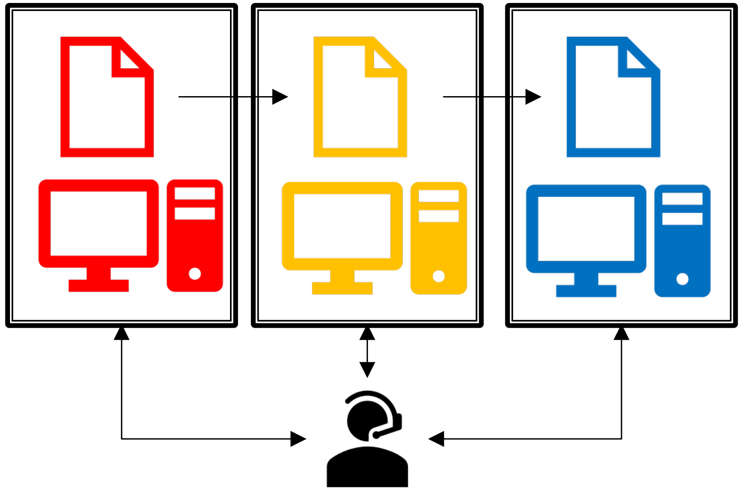


e.g. UQPipeline

Improves scheduler performance via packing.

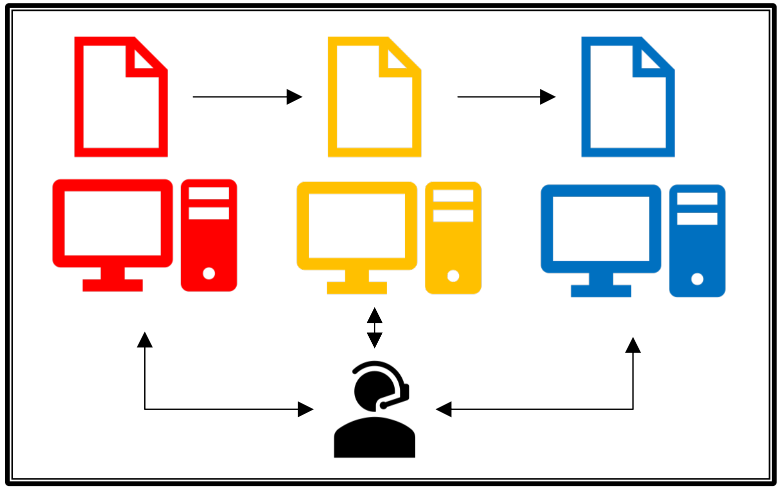
Merlin is a producer-consumer workflow system

External coordination
Independent batch jobs tied to tasks



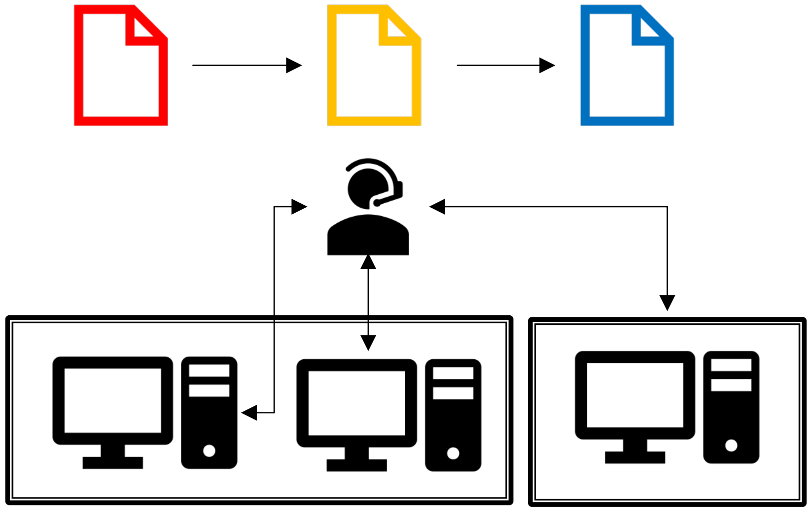
e.g. maestro
https://github.com/LLNL/maestrowf

Internal coordination
Large batch jobs for multiple tasks

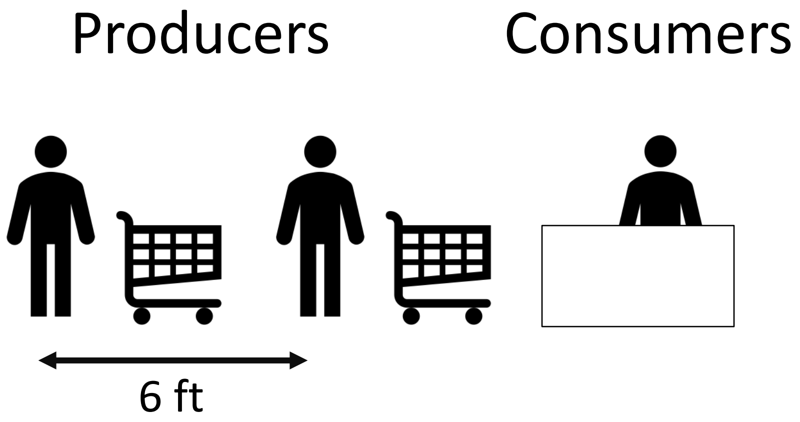


e.g. UQPipeline

Centralized task coordination
Distributed task-agnostic resources

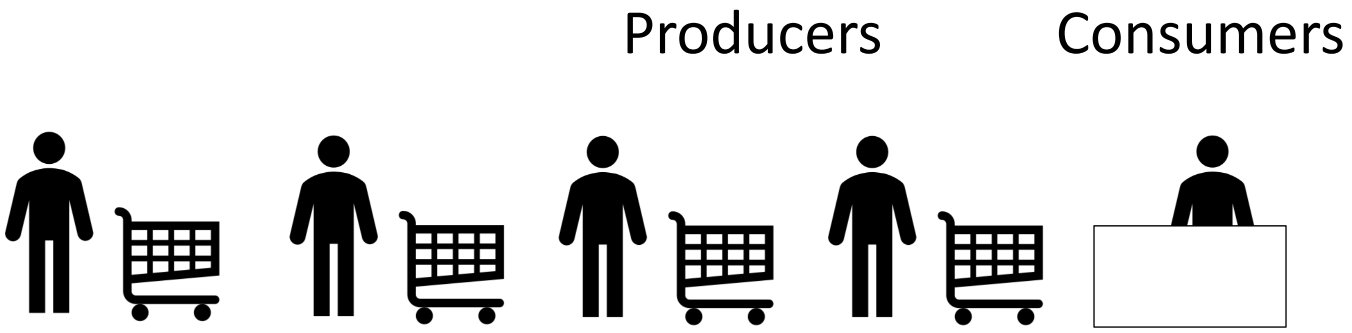


A (useful?) analogy for a producer-consumer workflow model: grocery shopping



“Much of education is cleverly contrived not to produce understanding, but merely the ILLUSION of understanding.”
- Donald Simanek, *The Dangers of Analogies*

A (useful?) analogy for a producer-consumer workflow model: grocery shopping

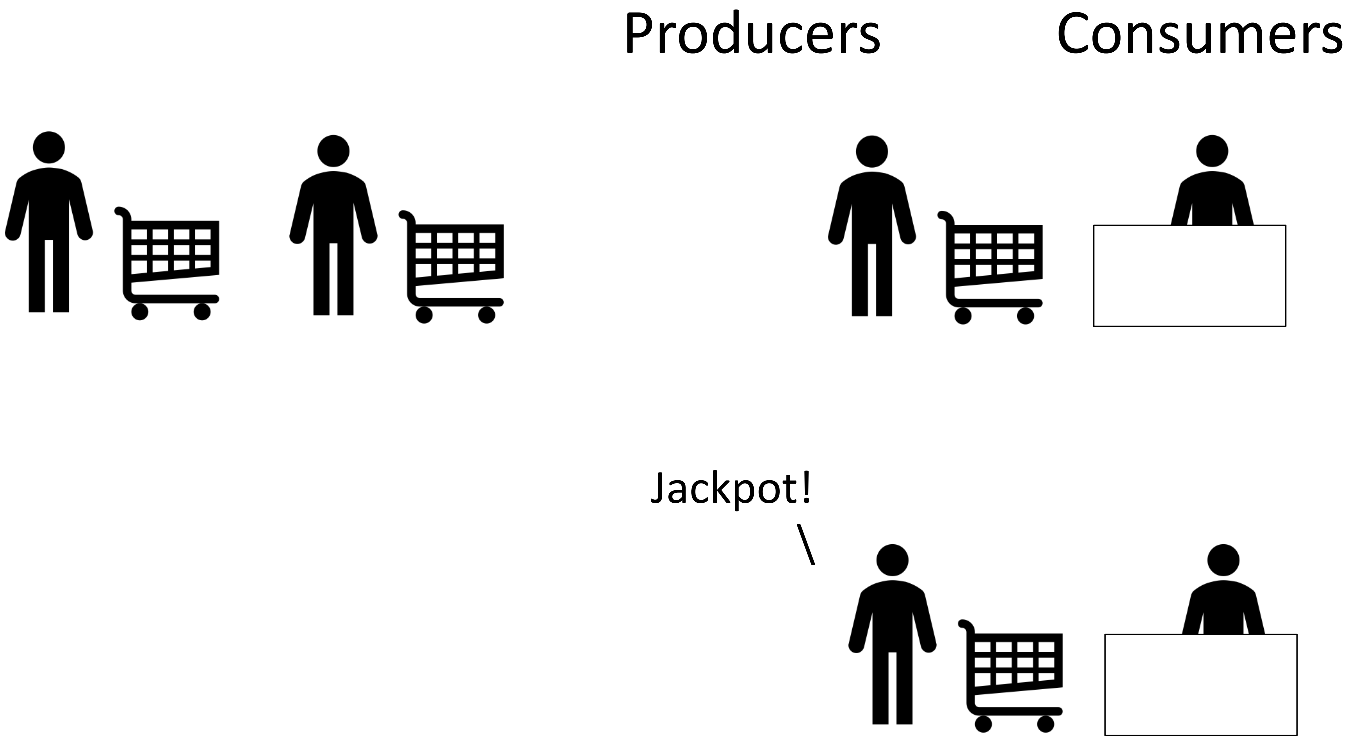


A (useful?) analogy for a producer-consumer workflow model: grocery shopping

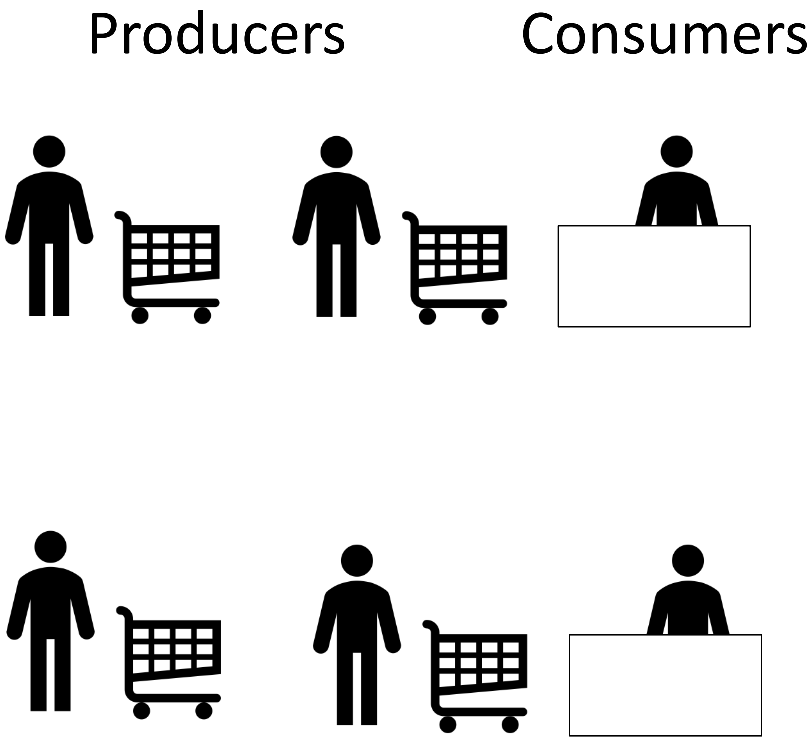
Producers Consumers

I can help the next customer

A (useful?) analogy for a producer-consumer workflow model: grocery shopping

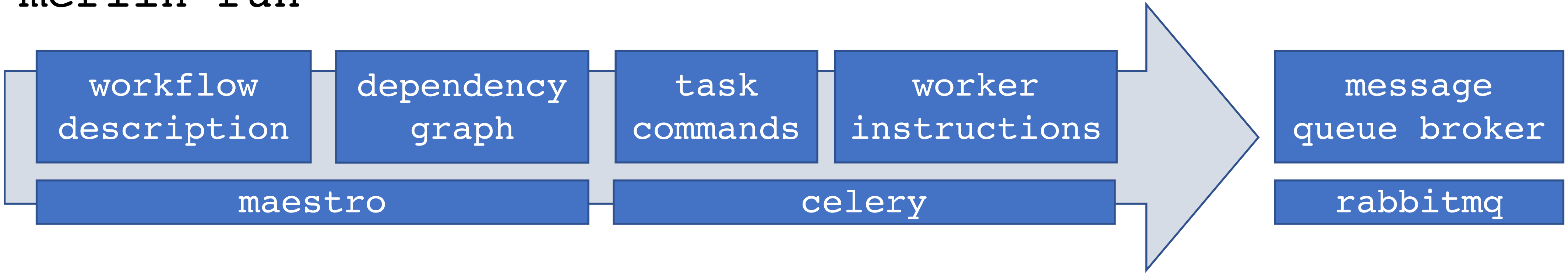


A (useful?) analogy for a producer-consumer workflow model: grocery shopping

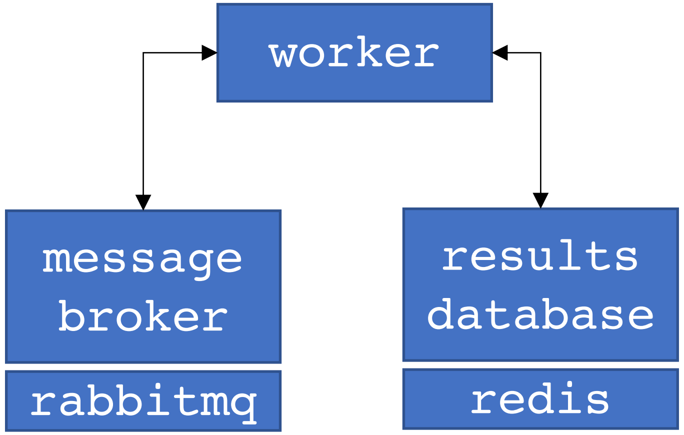


Merlin builds off and leverages 3rd party software

Producer: `merlin run`



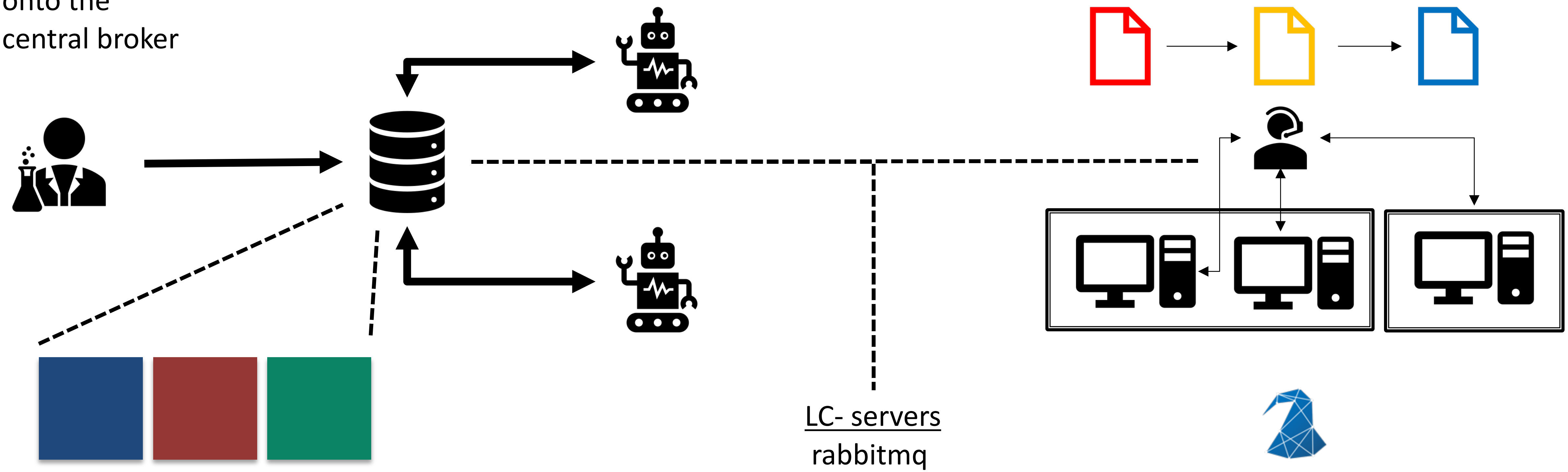
Consumer: `merlin run-worker`



Airflow
DoorDash
Instagram
Lyft
Mozilla
OpenGov

Nextdoor
Red Hat
Read the Docs
Uber
Venmo
…

At LLNL we have centralized servers as brokers (thanks, LC!)



People can push work onto the central broker

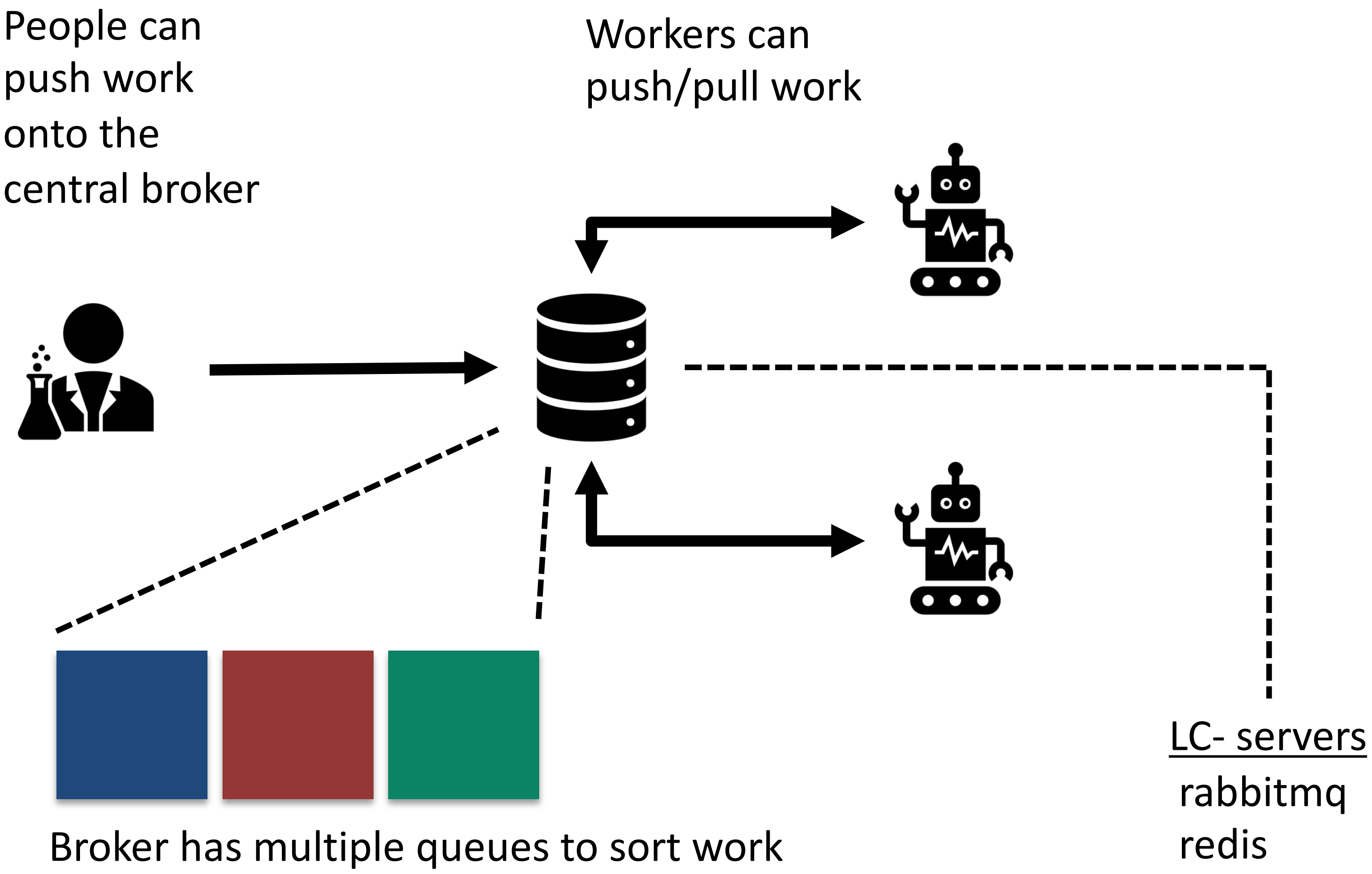
Workers can push/pull work

Centralized task coordination
Distributed task-agnostic resources

LC- servers
rabbitmq
redis

Broker has multiple queues to sort work

At LLNL we have centralized servers as brokers (thanks, LC!)


People can push work onto the central broker
Workers can push/pull work
Broker has multiple queues to sort work
LC- servers
rabbitmq
redis

The central broker lives outside of batch jobs and is visible by all compute nodes on the respective network.

Takeaway: Multiple batch jobs and machines can work on the same work.

We worked hard w/ LC to conform to security requirements, including auto encryption of traffic and results.

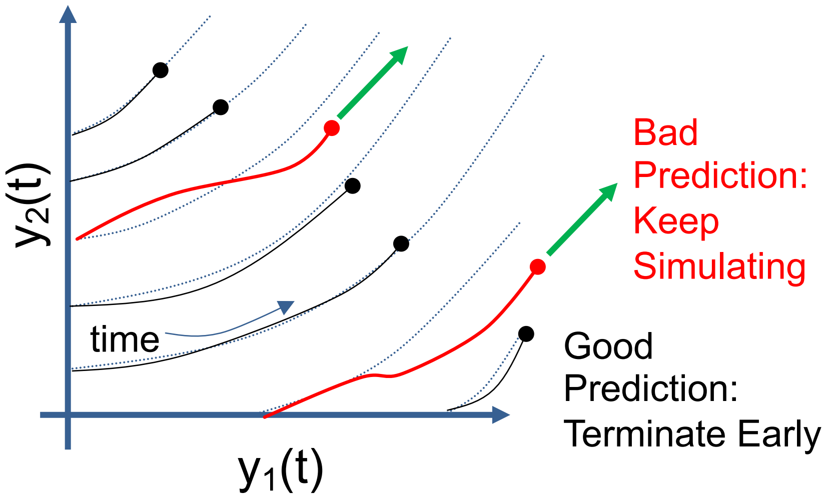
Bigger takeaway: this wouldn't have been possible without LC!

technical overview

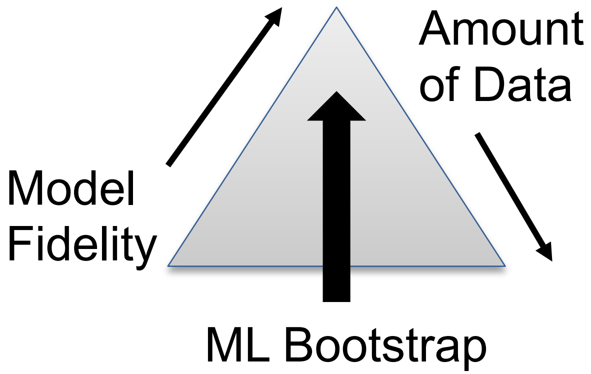
the story

Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies

Intelligent Sampling via Forecasting



Multi-Fidelity Modeling via Transfer Learning



LBPM SI LDRD Begins: Is speculative sampling possible?

2015 — 2016 — 2017 — 2018 — 2019 — 2020 — Today

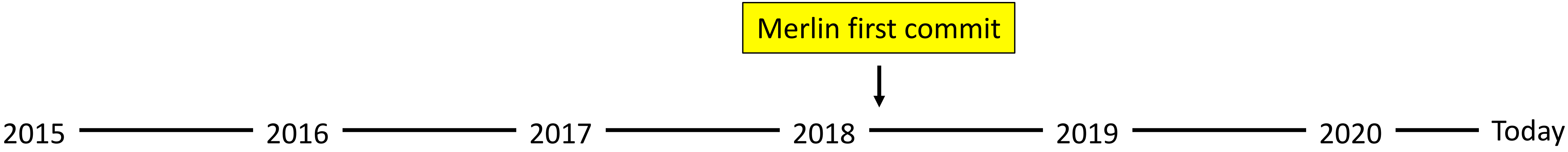
Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies

The problem: we would need to be able to coordinate

- Simulations Starting, Stopping and Continuing
- Machine Learning Training
- Machine Learning Inference
- Database Injections
- Database Queries
- Launching New Simulations
- And Operating at Scale b/c deep learning eats data

Team: Could celery work?

We'd need a lot of new infrastructure at LC, but let's try! Why not?


Merlin first commit

2015 — 2016 — 2017 — 2018 — 2019 — 2020 — Today

Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies

Sierra Open Science: Merlin runs 100 million simulations

- Tested Infrastructure at Scale
  - Hierarchical Ensemble Tasking
  - Flux for HPC launch
  - Data broker and backend
  - Error recovery
  - Auto cross-machine data-bounce
- 100 Million ICF simulations
- 4.8 Billion Images
- Massive Scientific ML Training Set

Sierra 100M Run

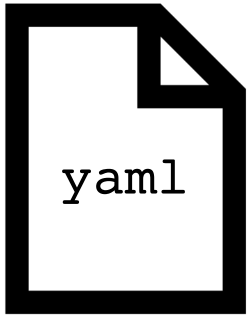
↓

2015 —— 2016 —— 2017 —— 2018 —— 2019 —— 2020 —— Today

10k sims released to public: https://data-science.llnl.gov/open-data-initiative

Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies

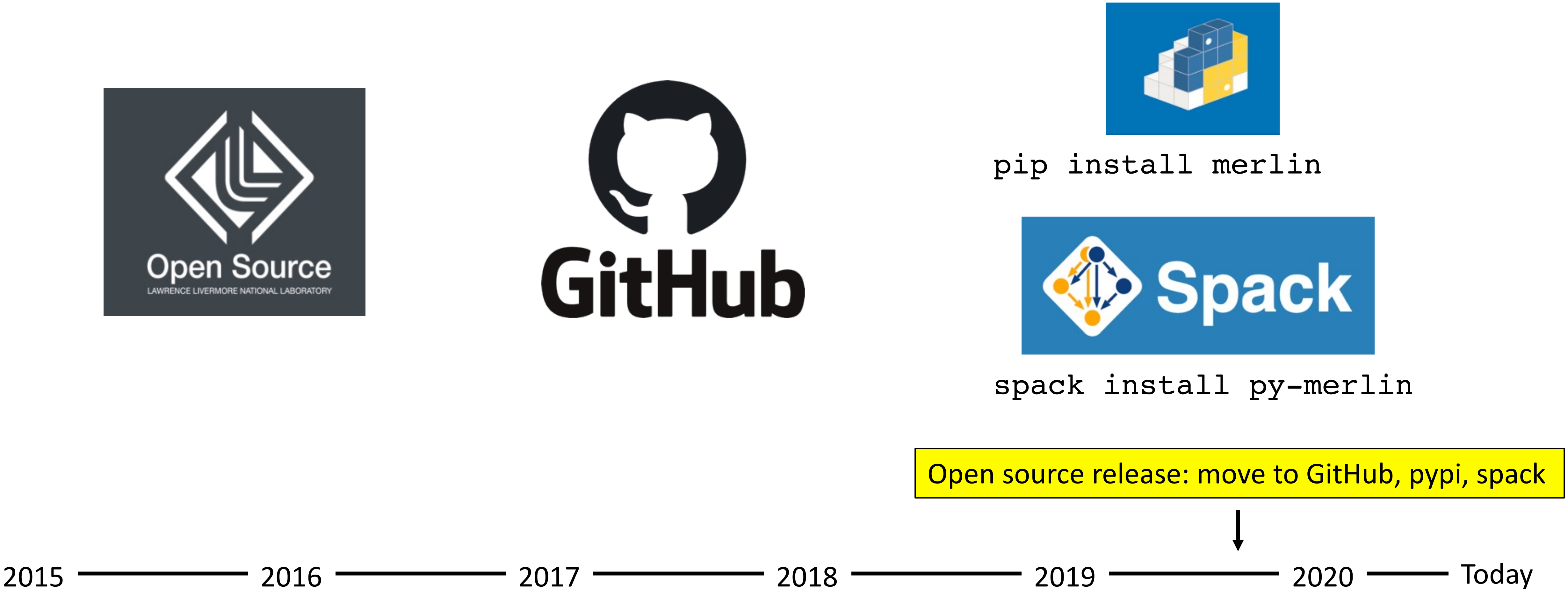
- Redesign to use maestro as a library
- Can read and launch any workflow described by maestro
- Opened space of supported workflows
- Expanded upon syntax: control logic, variables, broker-queue



System Redesign: maestro front end

2015 —— 2016 —— 2017 —— 2018 —— 2019 —— 2020 —— Today

Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies



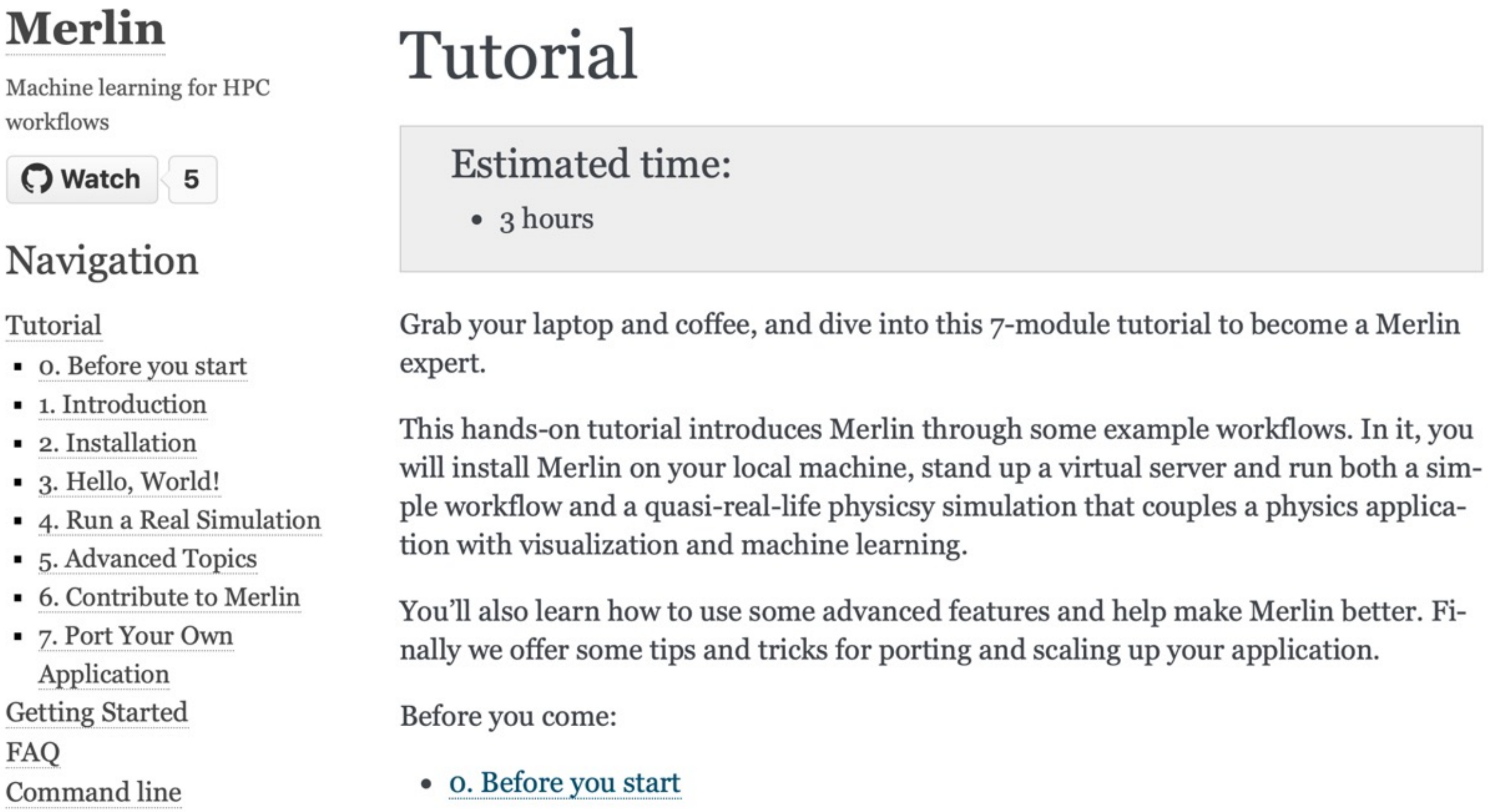
```
pip install merlin
```

```
spack install py-merlin
```

Open source release: move to GitHub, pypi, spack

2015 —— 2016 —— 2017 —— 2018 —— 2019 —— 2020 —— Today

Merlin builds off 5 years of exploring how to augment HPC simulation ensembles with cloud compute technologies



Merlin

Machine learning for HPC workflows

Watch 5

Navigation

Tutorial

- 0. Before you start
- 1. Introduction
- 2. Installation
- 3. Hello, World!
- 4. Run a Real Simulation
- 5. Advanced Topics
- 6. Contribute to Merlin
- 7. Port Your Own Application

Getting Started

FAQ

Command line

Tutorial

Estimated time:

- 3 hours

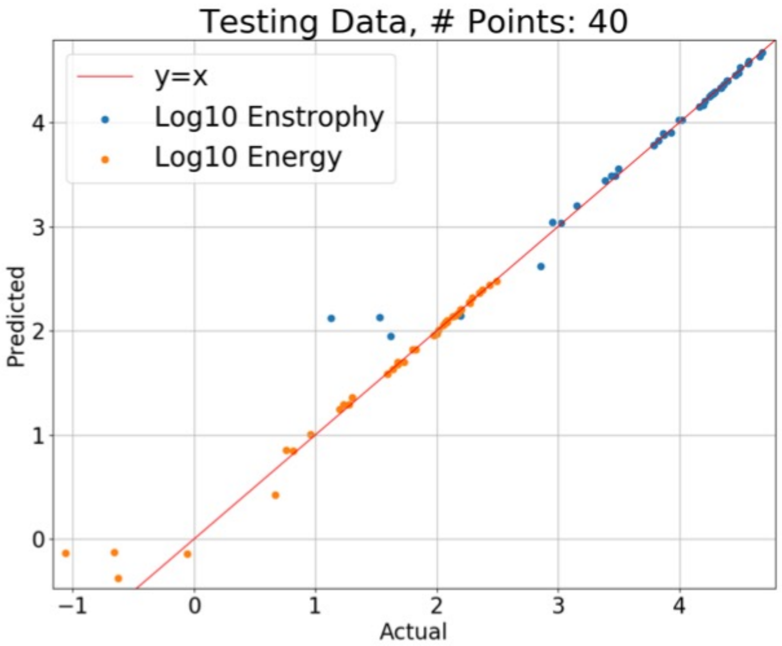
Grab your laptop and coffee, and dive into this 7-module tutorial to become a Merlin expert.

This hands-on tutorial introduces Merlin through some example workflows. In it, you will install Merlin on your local machine, stand up a virtual server and run both a simple workflow and a quasi-real-life physicsy simulation that couples a physics application with visualization and machine learning.

You'll also learn how to use some advanced features and help make Merlin better. Finally we offer some tips and tricks for porting and scaling up your application.

Before you come:

- 0. Before you start



~~WoWoHa~~ Tutorial

2015 ——— 2016 ——— 2017 ——— 2018 ——— 2019 ——— 2020 ——— Today

https://merlin.readthedocs.io/en/latest/tutorial.html

technical overview

tech details

the algorithm

performance

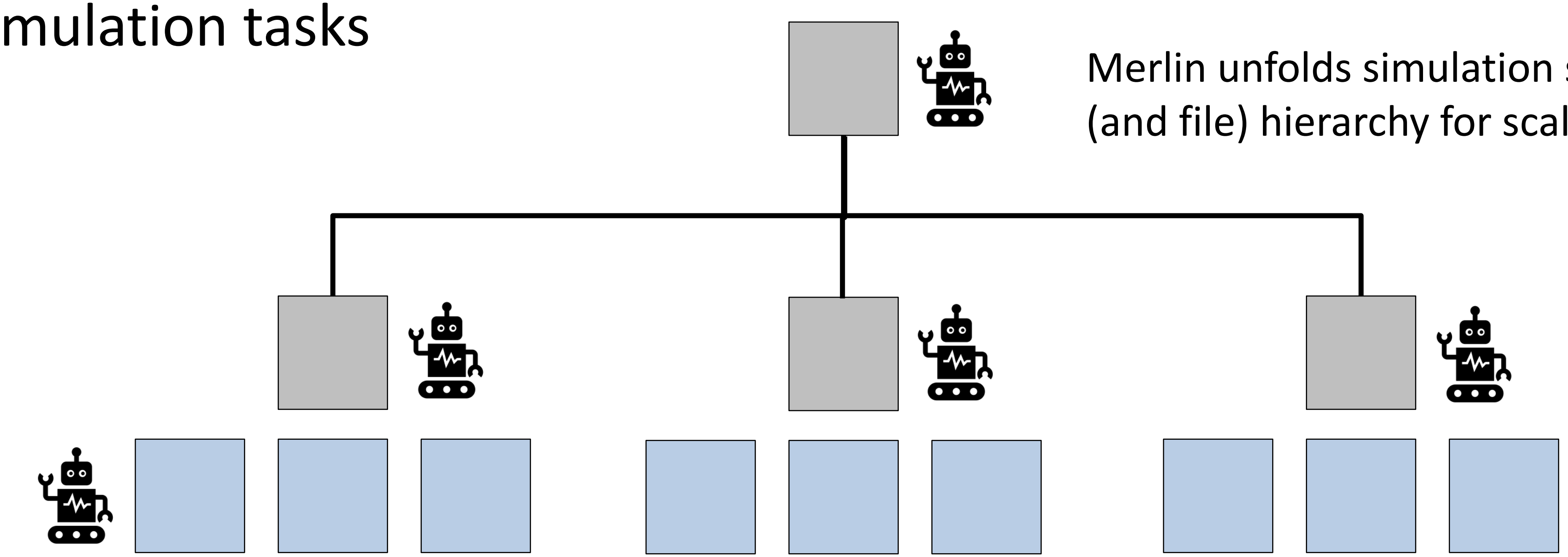
The heart of Merlin is the hierarchical task creation algorithm

⬛ task-creation tasks

🟦 simulation tasks

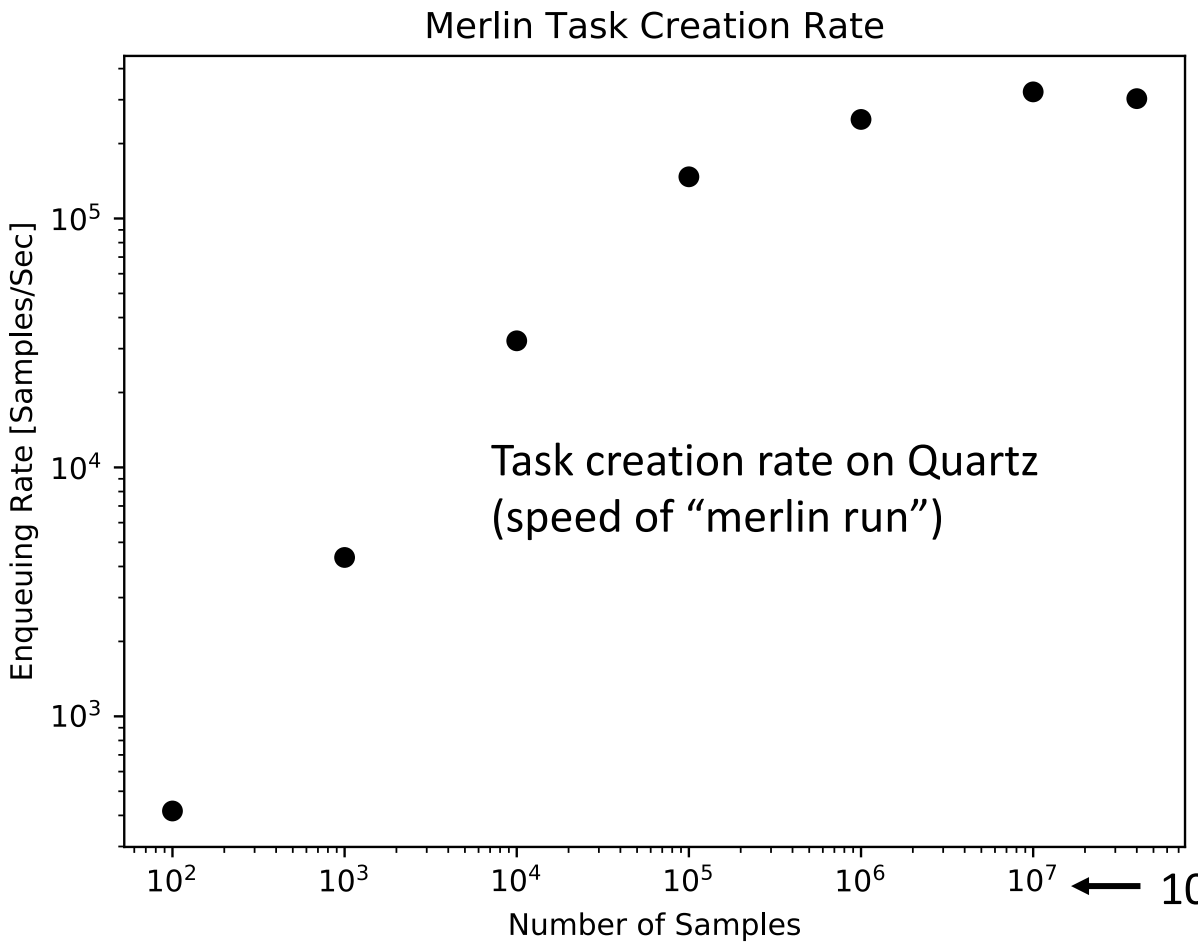
In a producer-consumer model, workers can create tasks that create tasks

Merlin unfolds simulation samples into a task (and file) hierarchy for scalability



Workflow creation scales with worker count, significantly reducing time-to-first simulation

Launching new simulations is very fast, essentially non-blocking



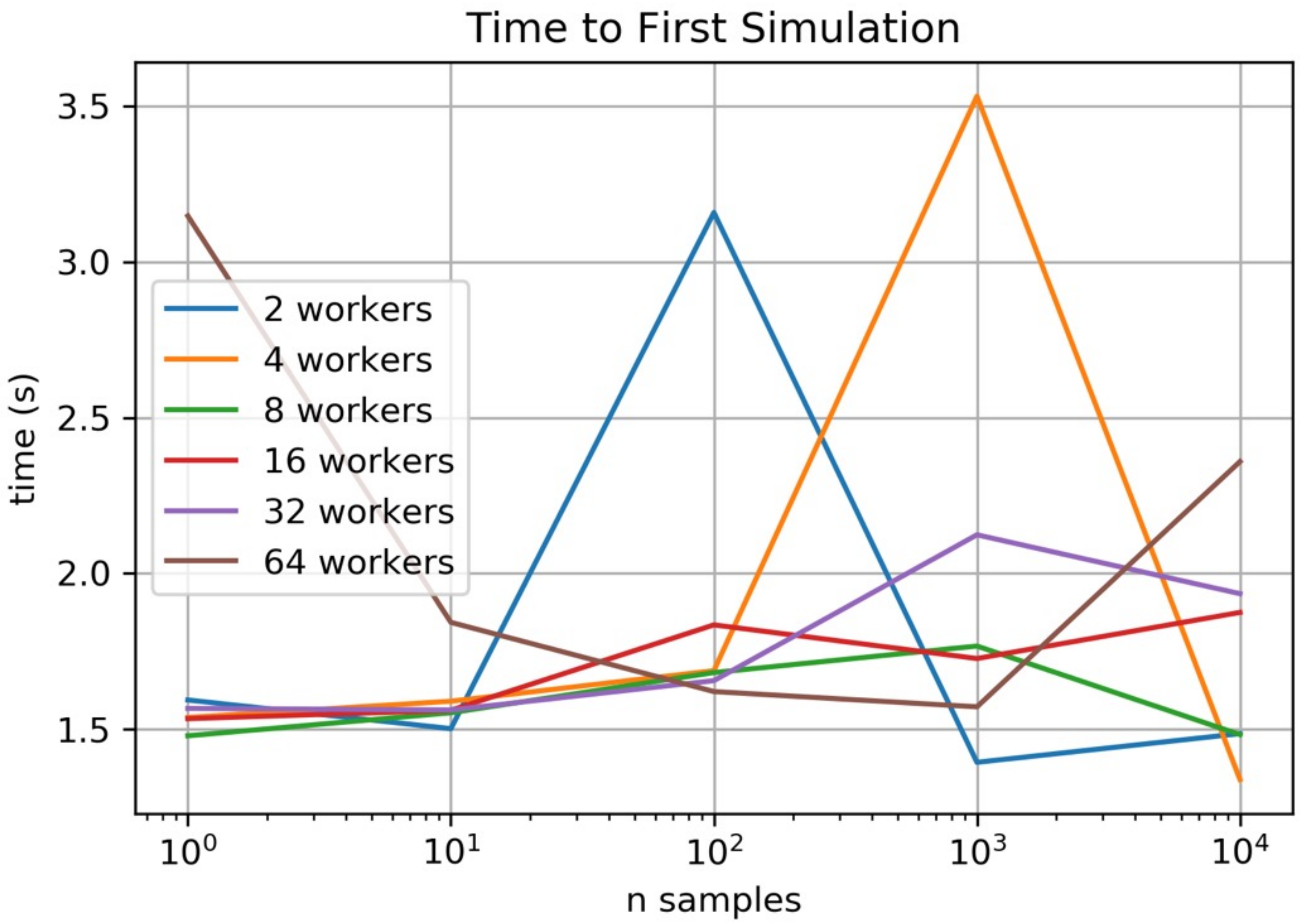
Task creation rate on Quartz (speed of “merlin run”)

← 300,000 tasks / second

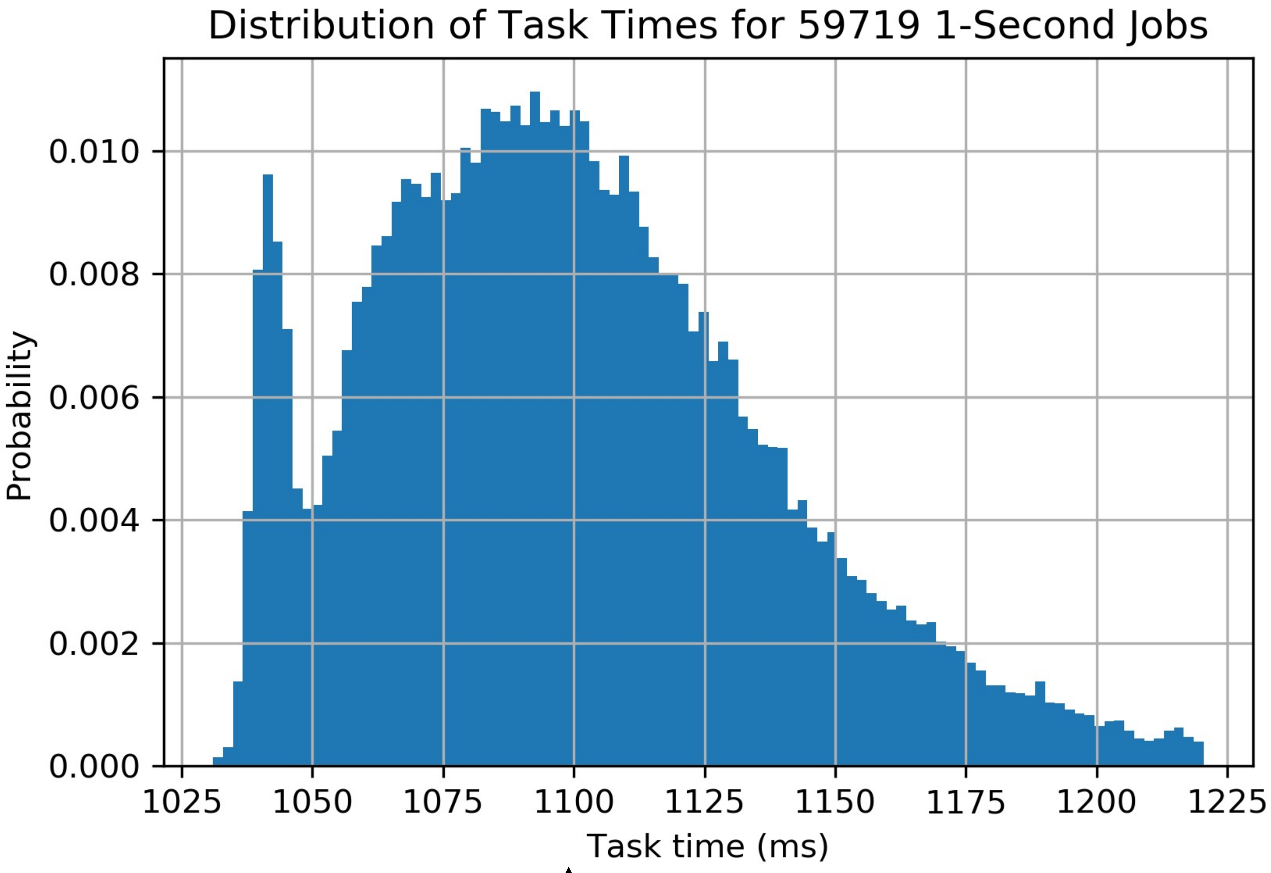
← 10 million samples

Why is it so fast? We only encode the metadata needed to unroll the task hierarchy by the workers

Time-to-first simulation is a few seconds

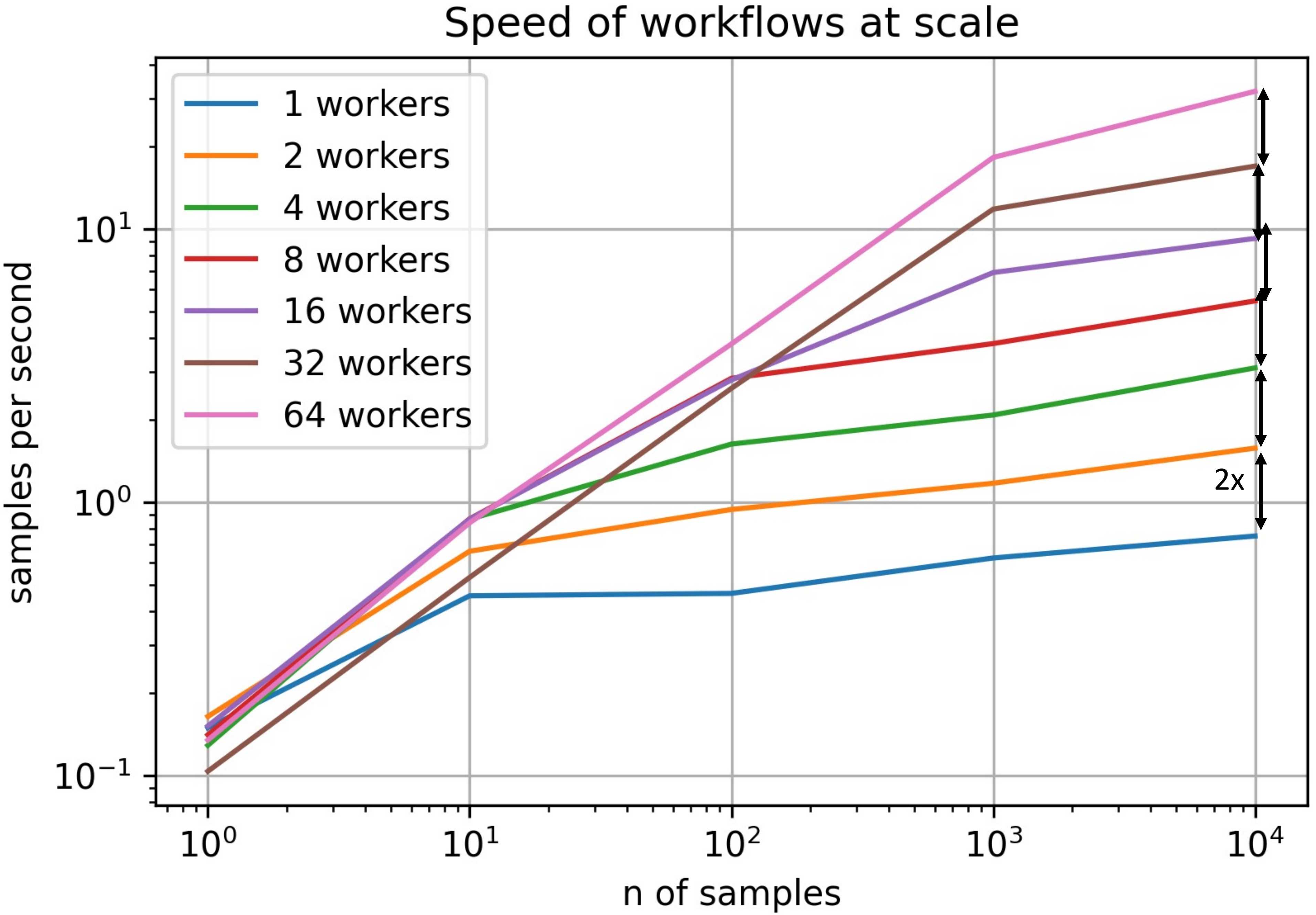


Per-simulation overhead is ~ 100ms



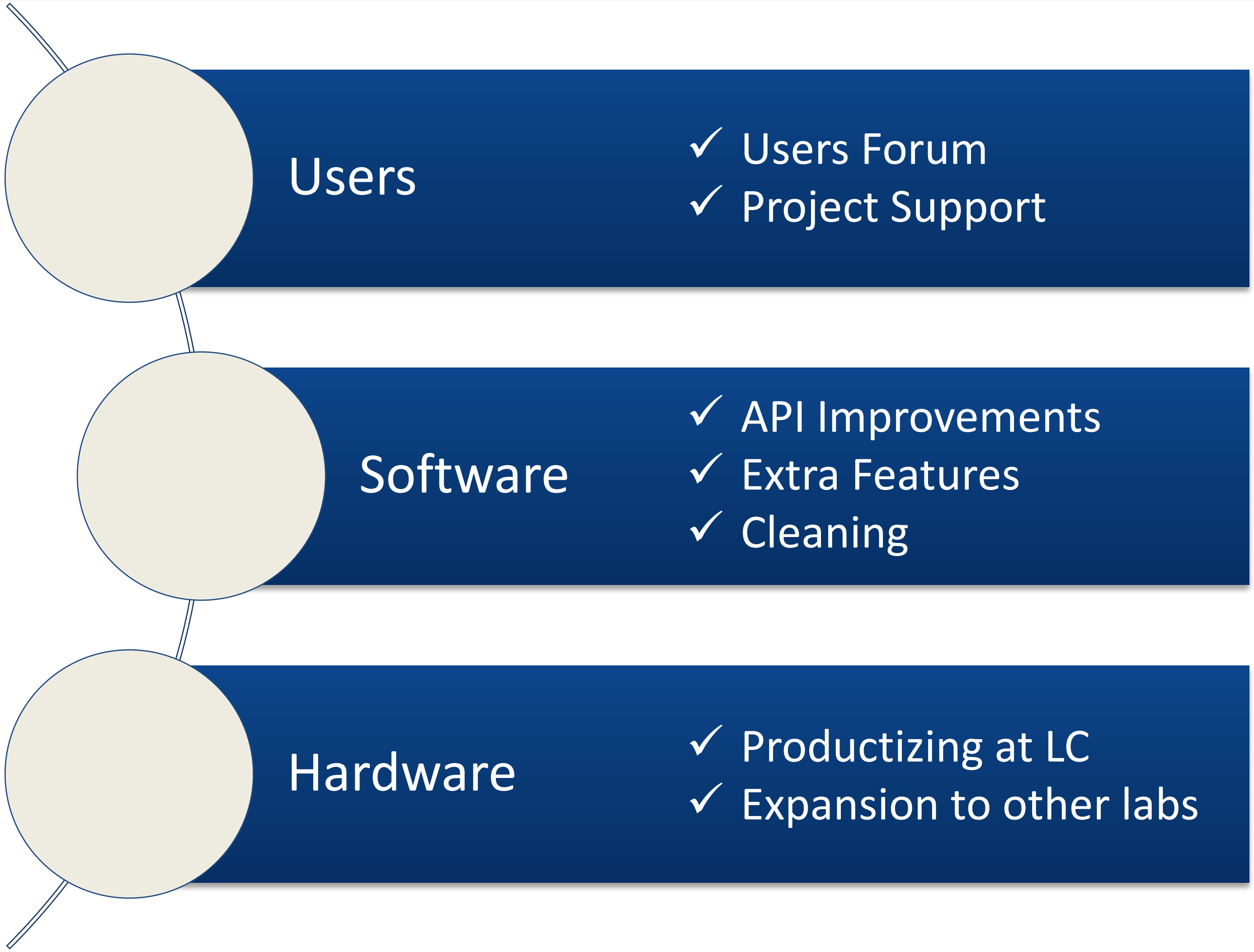
60k “sleep 1” tasks, 1000 ms/task

Overhead is ~100 ms / task


Speed of workflows at scale
1 workers
2 workers
4 workers
8 workers
16 workers
32 workers
64 workers
samples per second
10^{1}
10^{0}
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}
10^{4}
n of samples
2x

Current Direction

Development Endgame



Users

- ✓ Users Forum
- ✓ Project Support

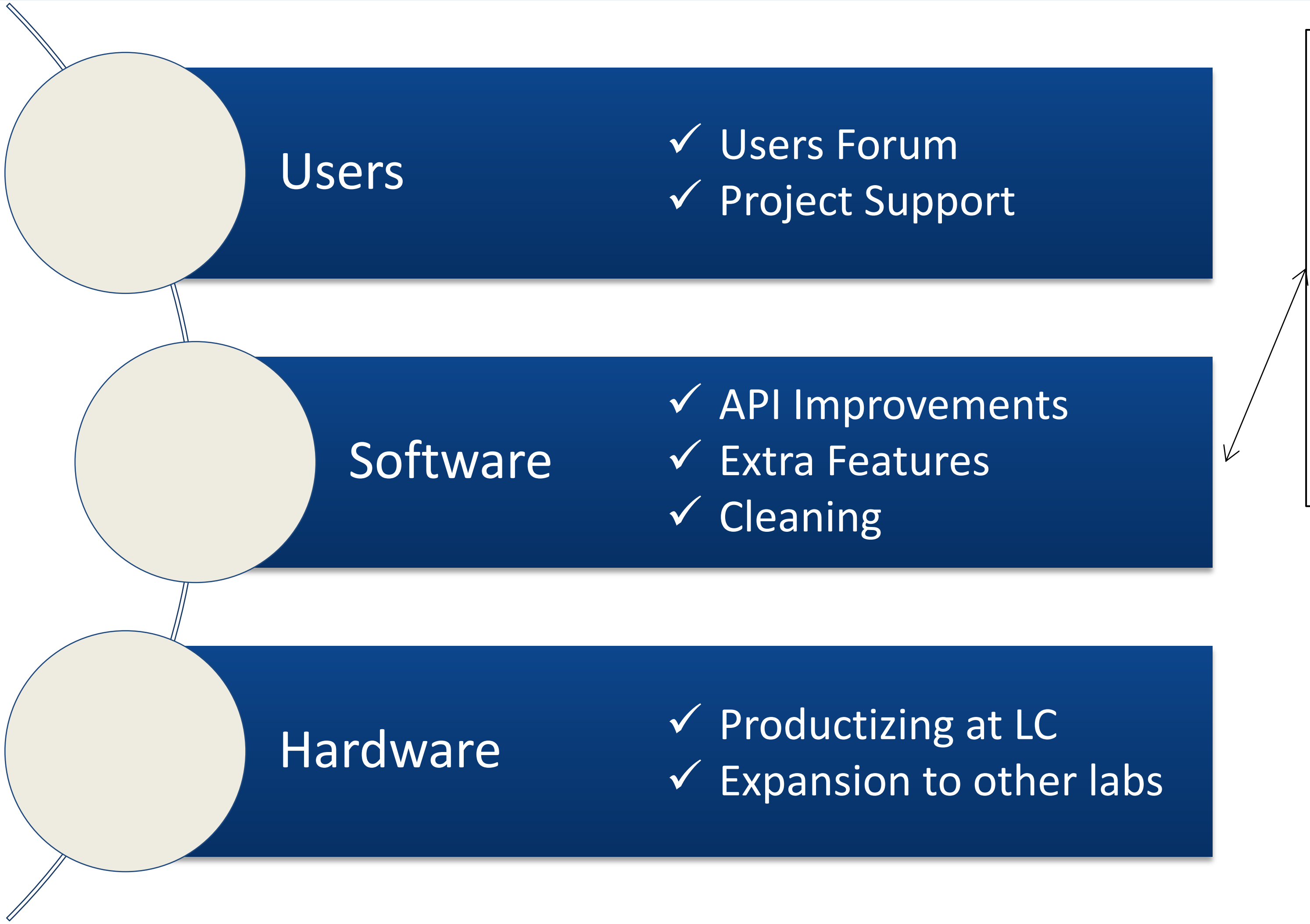
Software

- ✓ API Improvements
- ✓ Extra Features
- ✓ Cleaning

Hardware

- ✓ Productizing at LC
- ✓ Expansion to other labs

Development Endgame



Users

- ✓ Users Forum
- ✓ Project Support

Software

- ✓ API Improvements
- ✓ Extra Features
- ✓ Cleaning

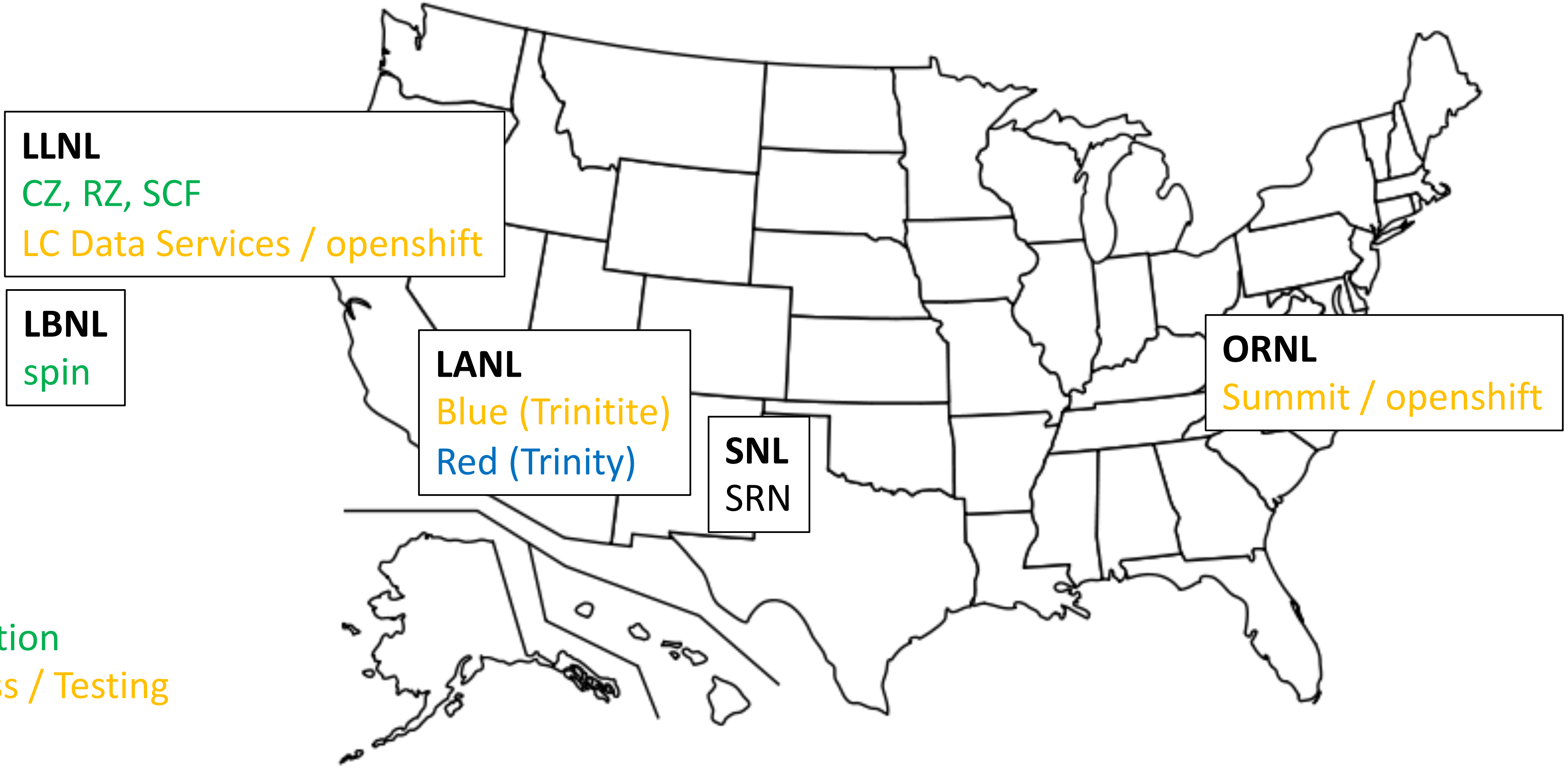
Hardware

- ✓ Productizing at LC
- ✓ Expansion to other labs

1. Support for latest version of Maestro
2. Upgrade status command with task results
3. Add auto worker configuration

Future Direction

Status of Merlin Infrastructure @ Other DOE Sites



LLNL
CZ, RZ, SCF
LC Data Services / openshift

LBNL
spin

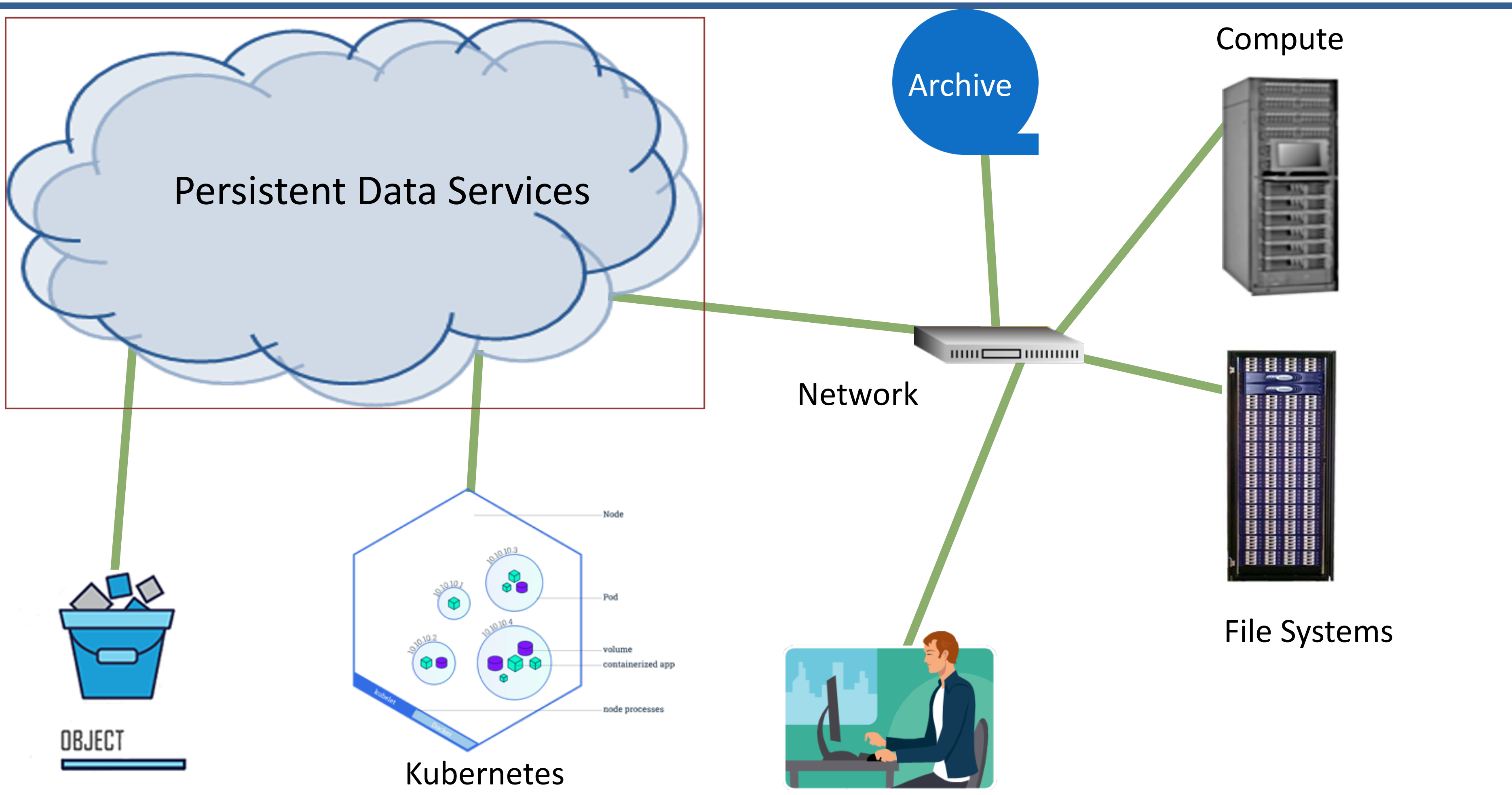
LANL
Blue (Trinitite)
Red (Trinity)

SNL
SRN

ORNL
Summit / openshift

In Production
In Progress / Testing
Planned
Expressed Interest

LC plans to productize this: A Persistent Data Services Platform

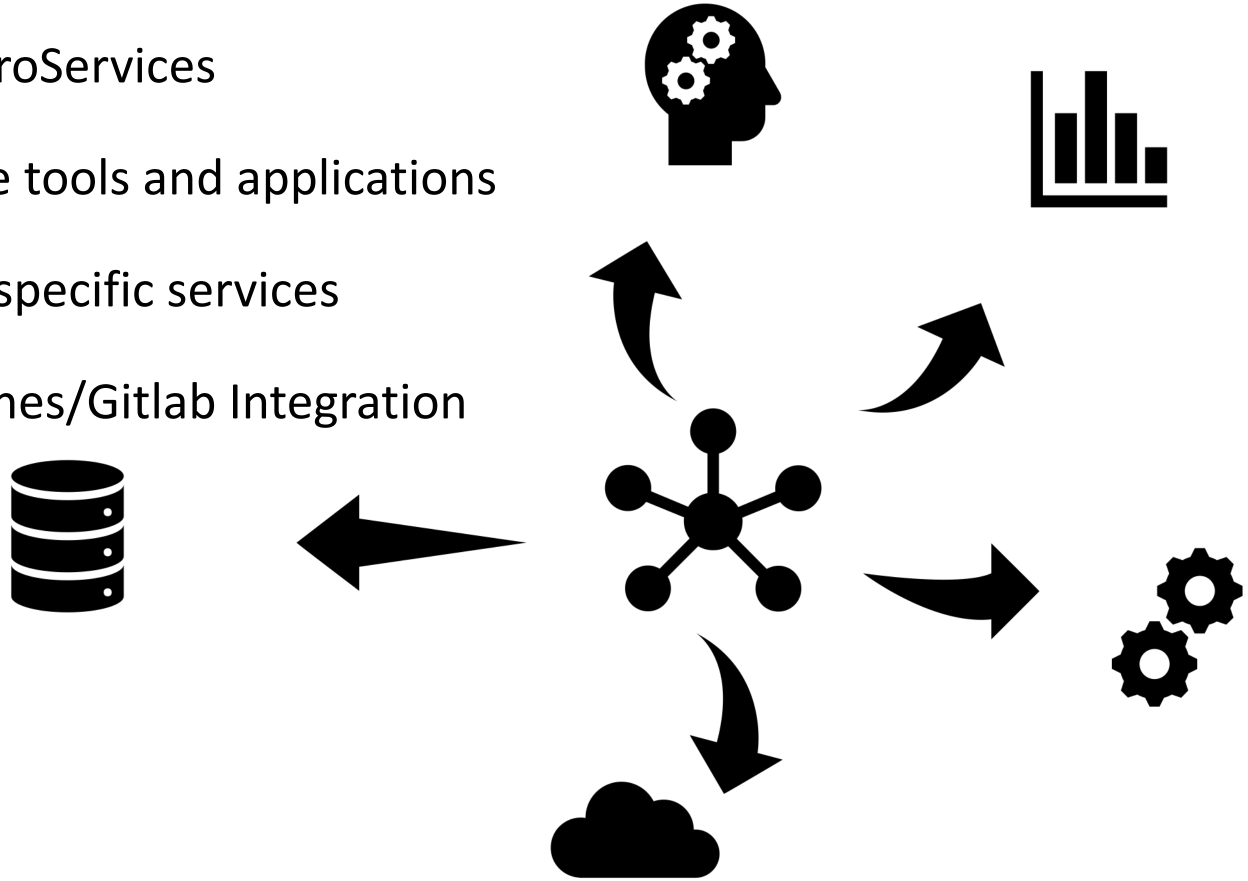


Coming Soon @ LC

Slide from Dave Fox (LC)

A Persistent Data Services Platform: Advanced Deployments

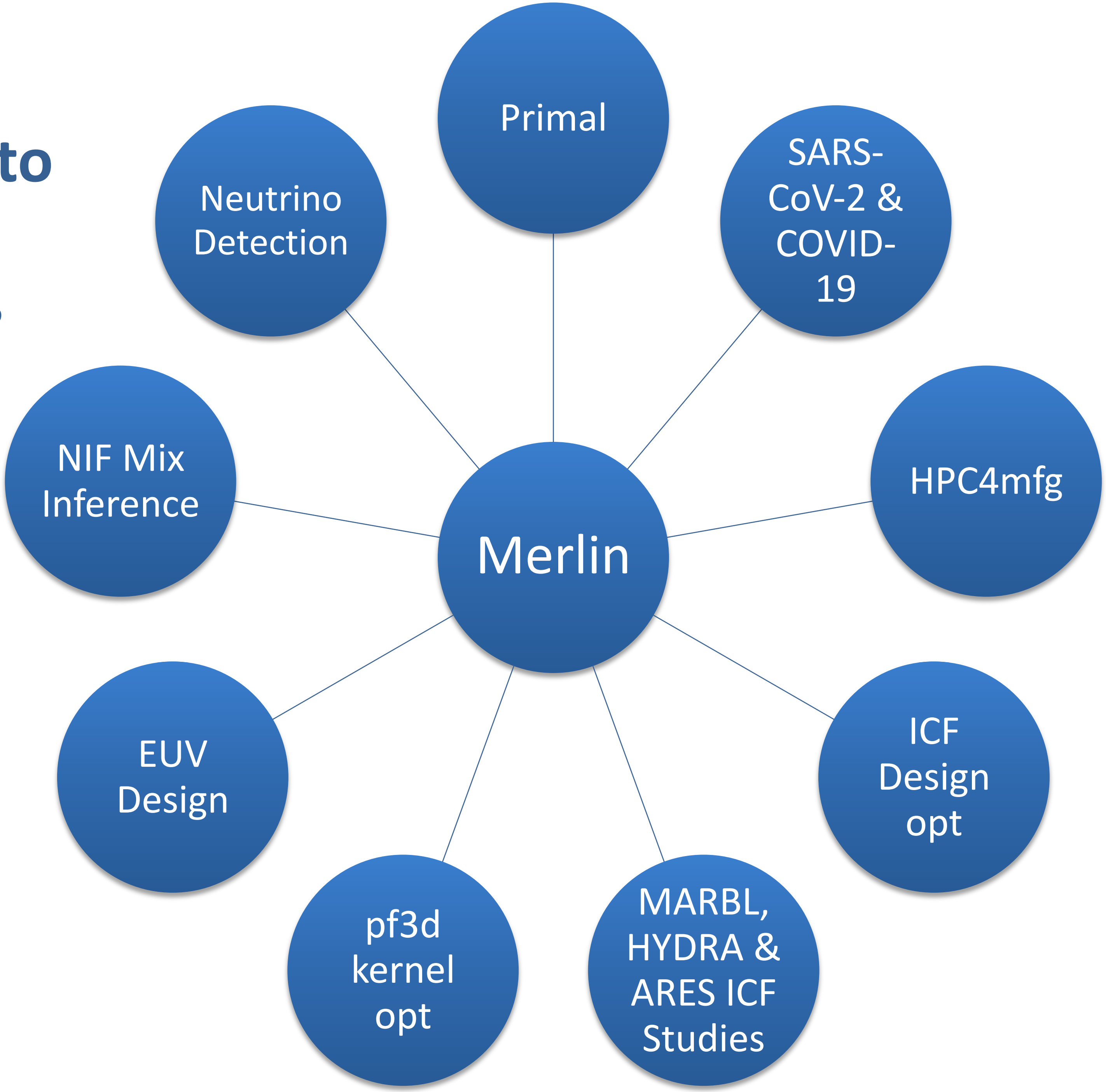
- Suite of MicroServices
- Data Science tools and applications
- Application specific services
- CI/CD pipelines/Gitlab Integration



Coming Soon @ LC

Slide from Dave Fox (LC)

From LDRD investment to program deliverables



Merlin

Primal

SARS-CoV-2 & COVID-19

HPC4mfg

ICF Design opt

MARBL, HYDRA & ARES ICF Studies

pf3d kernel opt

EUV Design

NIF Mix Inference

Neutrino Detection

Want to learn more?

- Full Documentation: https://merlin.readthedocs.io/
- ArXiv article: **Enabling Machine Learning-Ready HPC Ensembles with Merlin**
- Tutorial: https://merlin.readthedocs.io/en/latest/tutorial.html
- GitHub: https://github.com/llnl/merlin
- `pip install merlin`
- `spack install py-merlin`

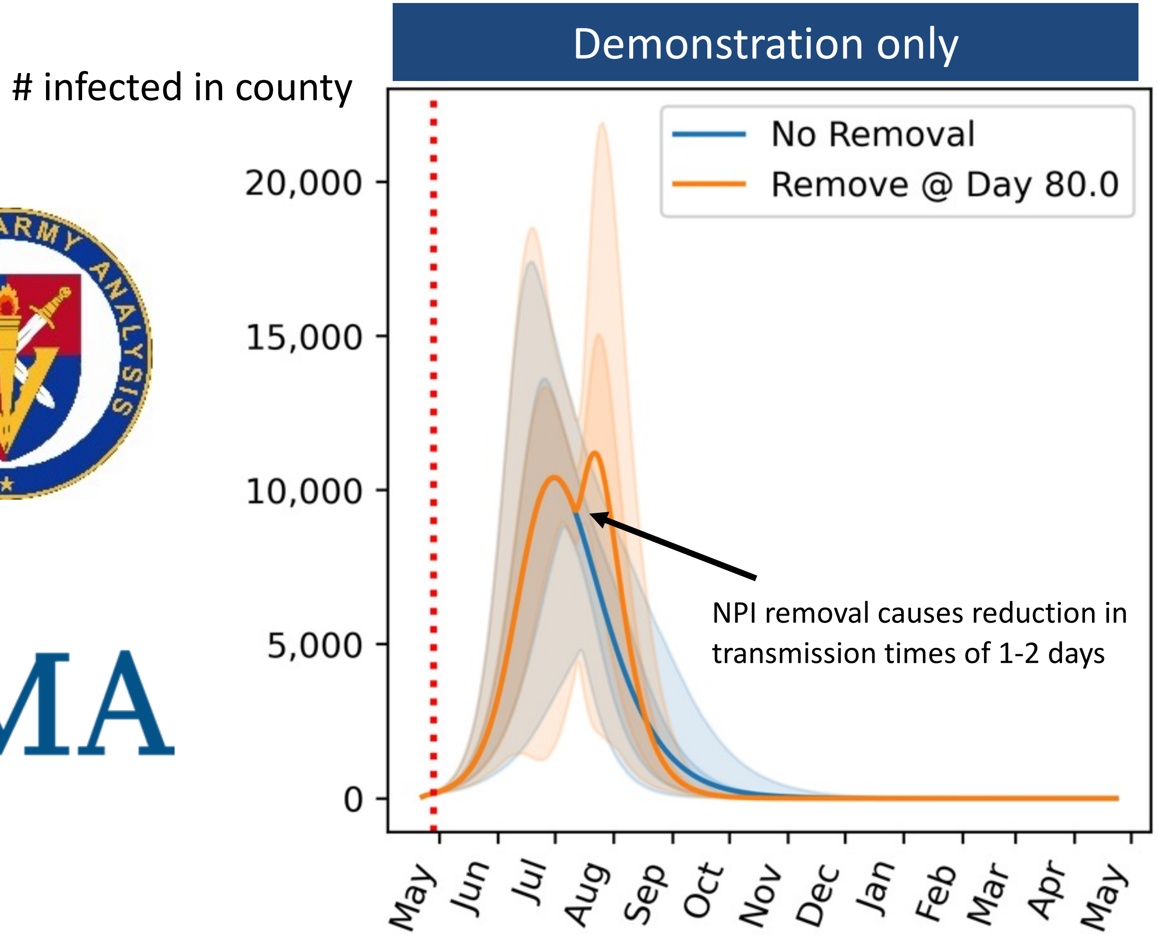
Thank you!

github.com/llnl/merlin

Lawrence Livermore
National Laboratory

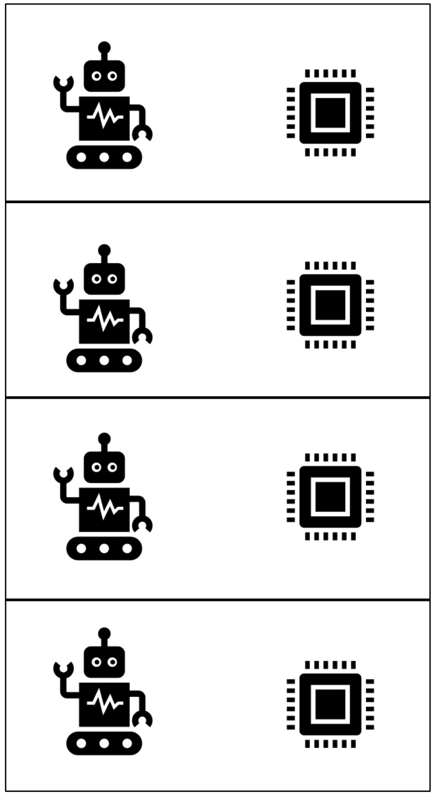
Disclaimer
This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

We are using Merlin for large-scale COVID19 Scenario Modeling



NPI removal causes reduction in transmission times of 1-2 days

4 workers per Lassen Node
1/GPU, each running tensorflow



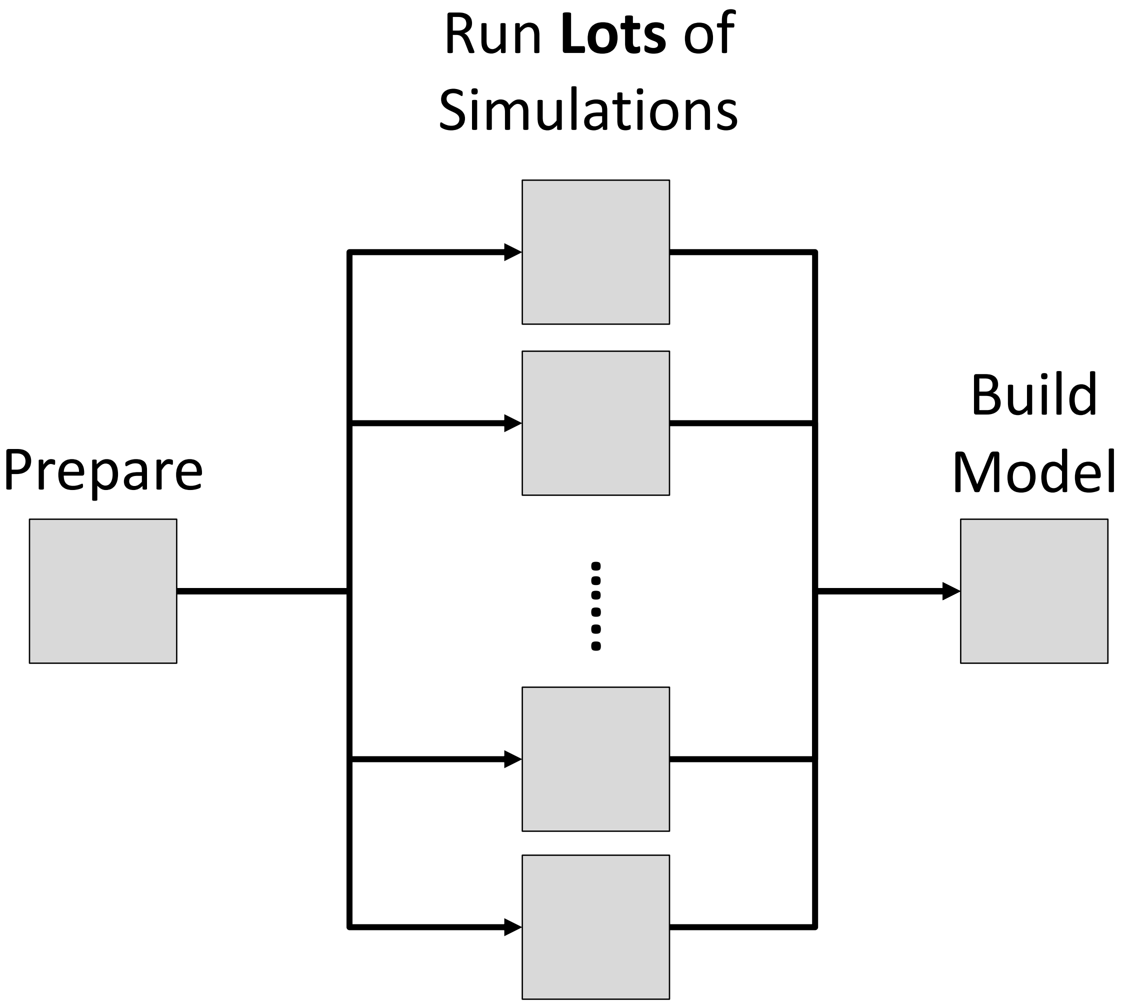
With flux we can model one scenario w/ UQ for the entire country in ~5 minutes on a few nodes: near real-time feedback

With J. Gaffney, B. Spears, A. Cunningham, T. Bremer, J. Moon, J. Thiagarajan, S. Mukhopadhyay

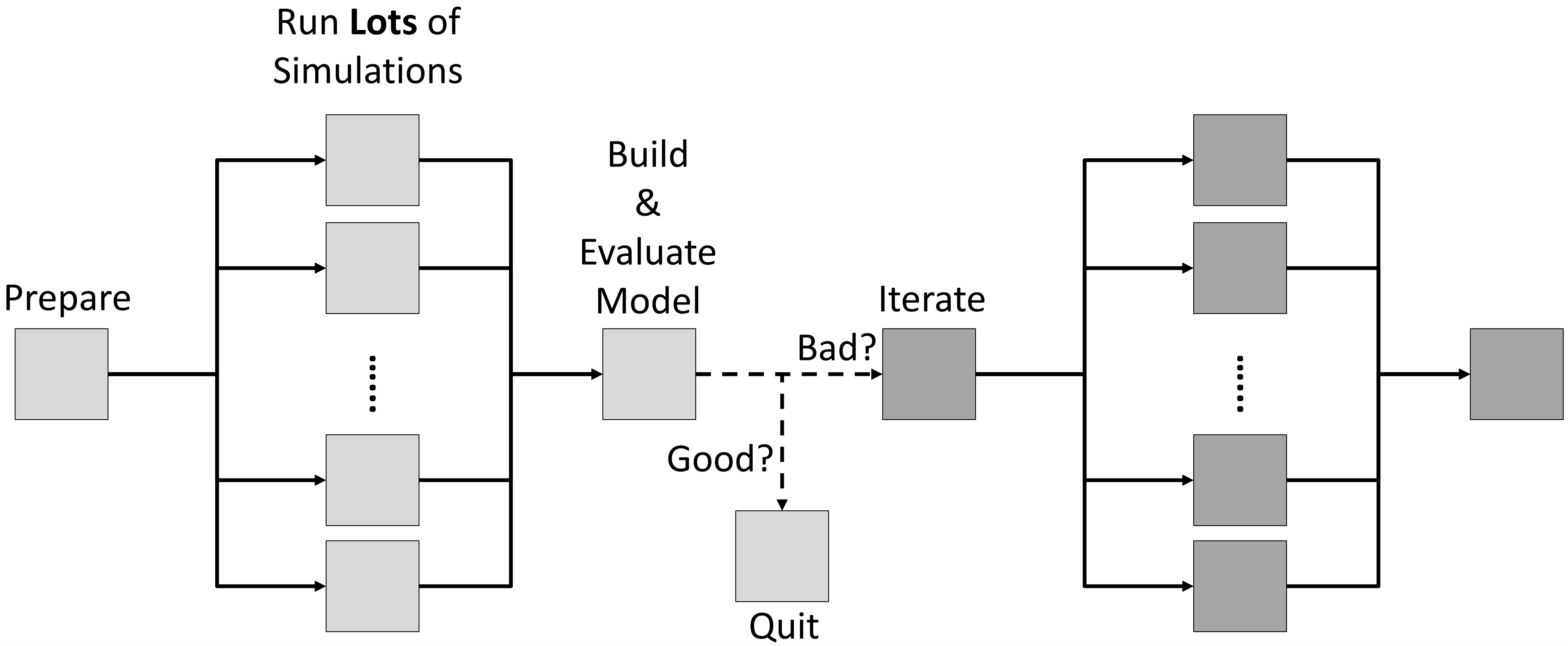
What can you do with Merlin?

What use cases did we target?

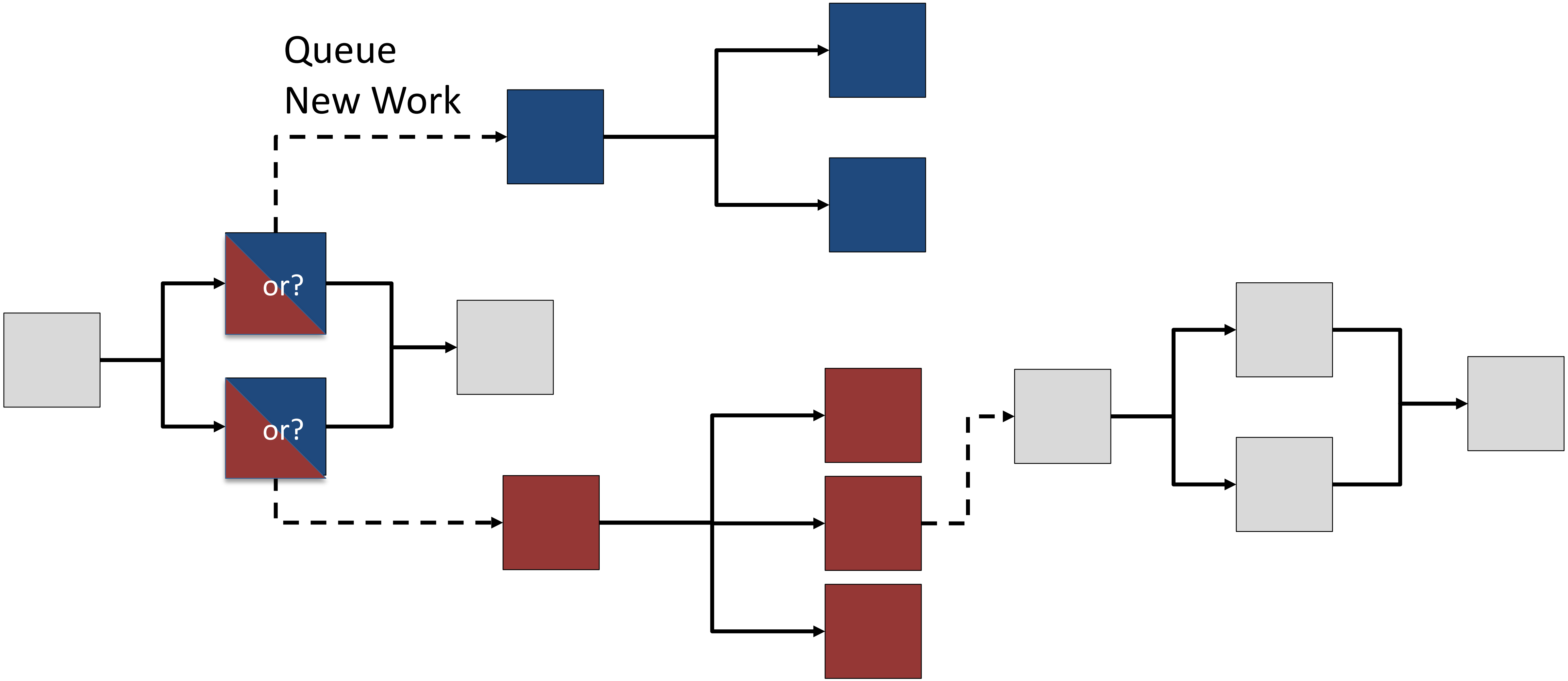
Surrogate Model Building



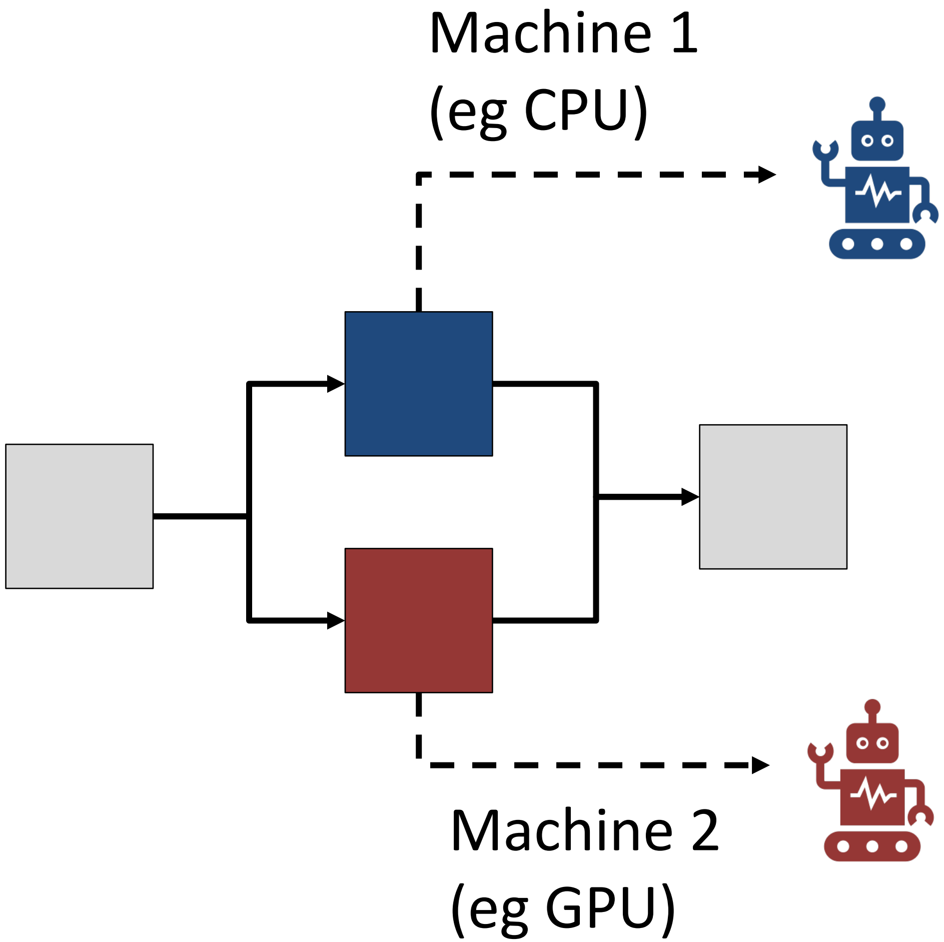
Active Learning



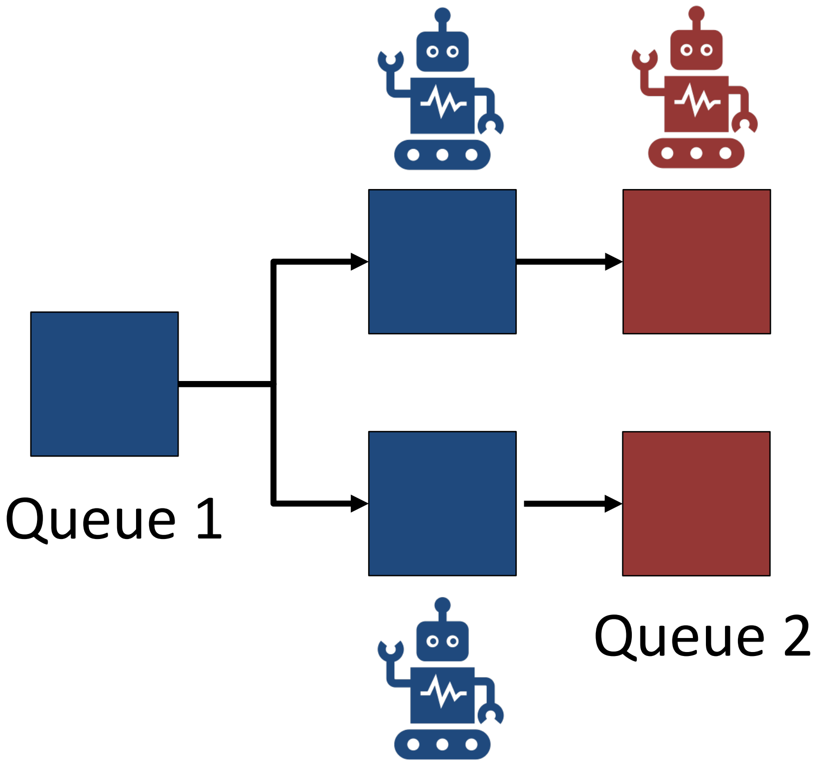
Dynamic Workflow Branching and Launching



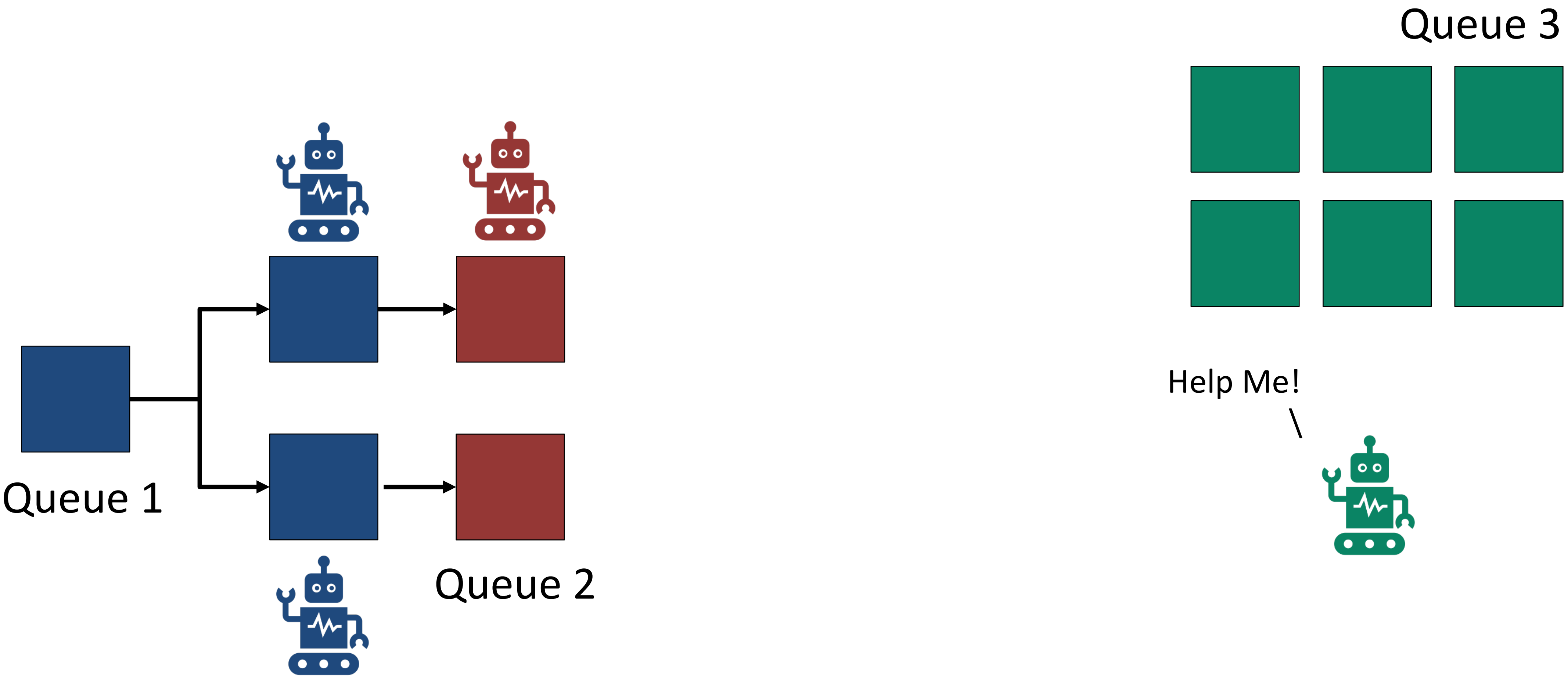
Heterogenous Workflows



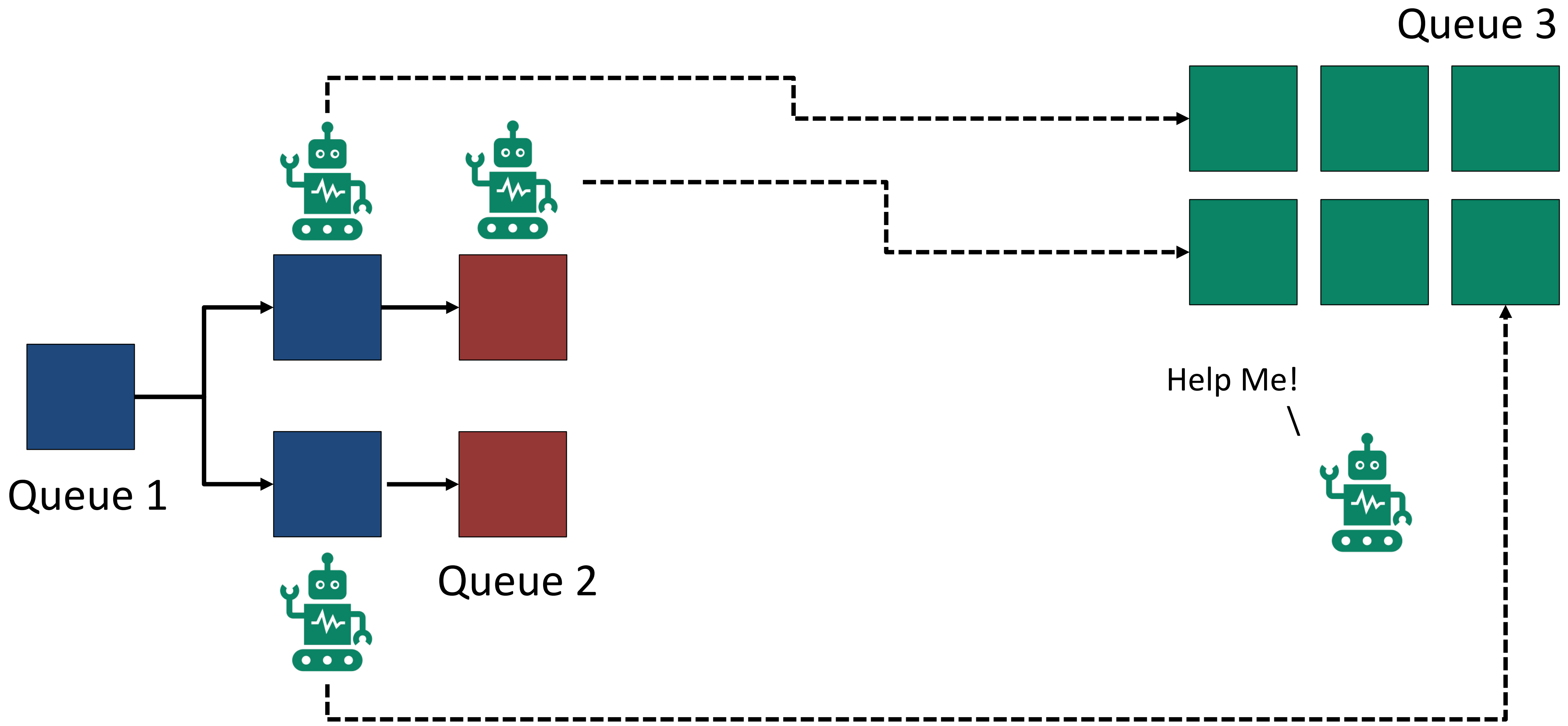
Task Prioritization



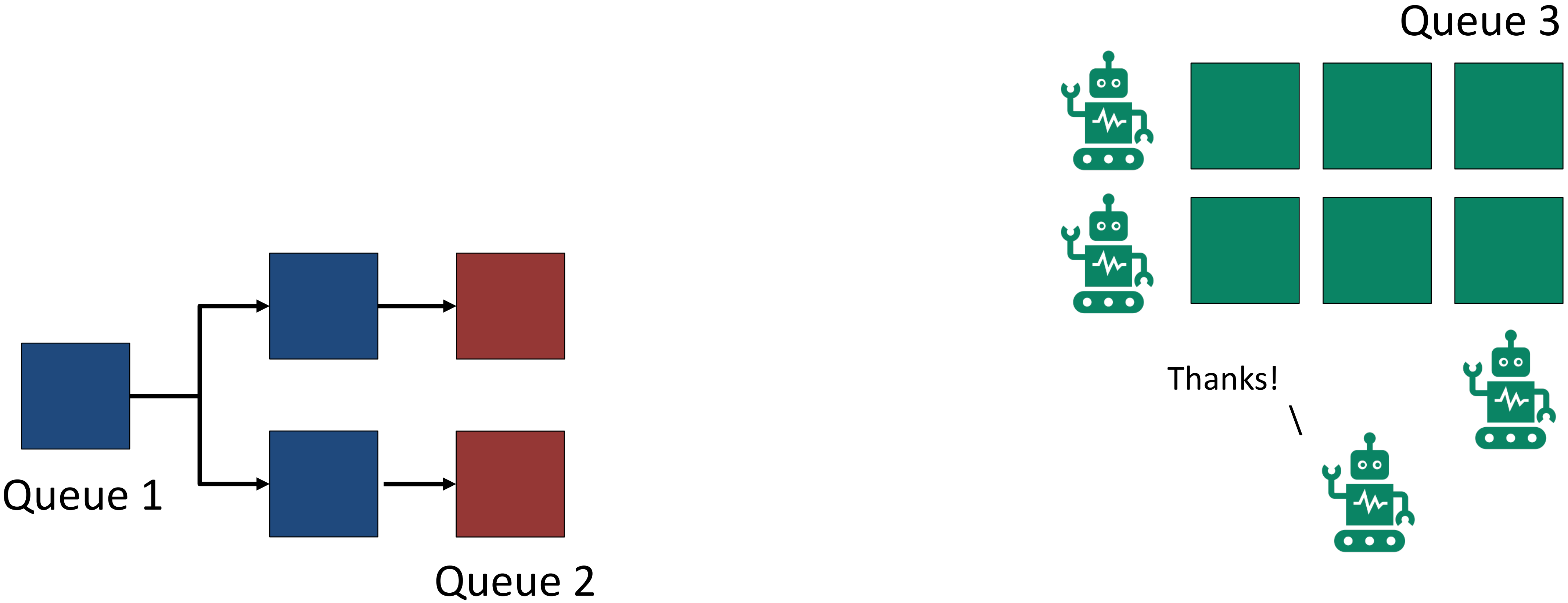
Task (Re)Prioritization



Task (Re)Prioritization



Task (Re)Prioritization


Queue 3
Thanks!
Queue 1
Queue 2