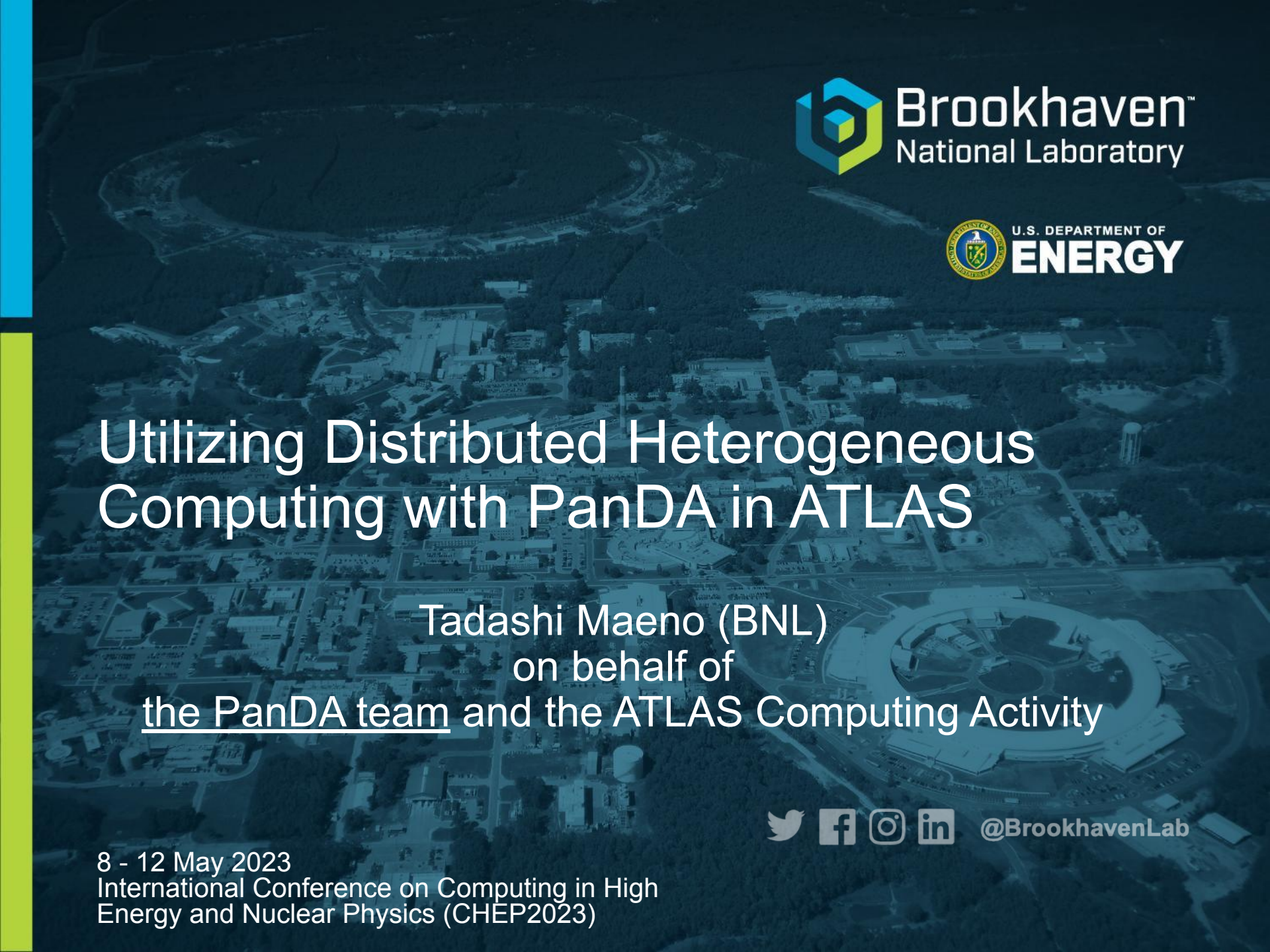


Utilizing Distributed Heterogeneous Computing with PanDA in ATLAS

Tadashi Maeno (BNL)
on behalf of
the PanDA team and the ATLAS Computing Activity

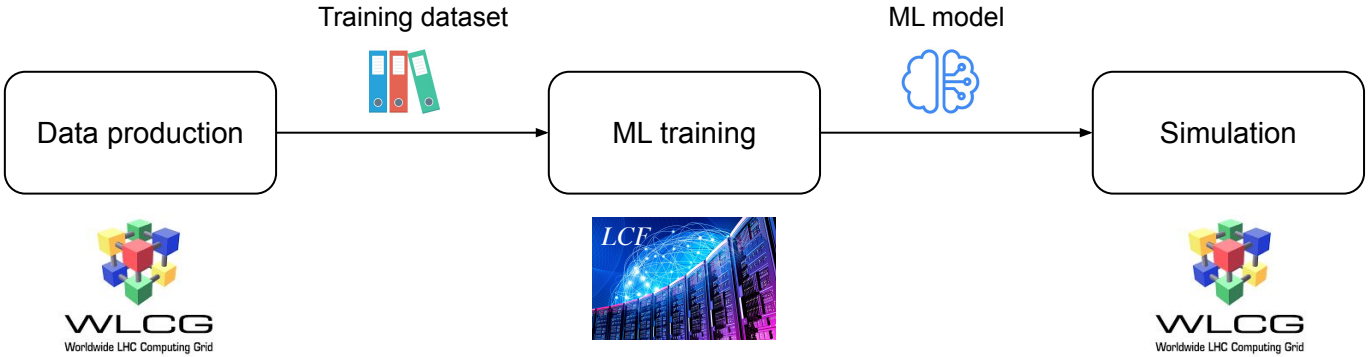
8 - 12 May 2023
International Conference on Computing in High Energy and Nuclear Physics (CHEP2023)

Needs for Distributed Heterogeneous Computing

- Distributed = Geographically distributed != Multi-node/process
- Most users love local resources, but have to go to remote providers when enough or suitable resources/services are locally unavailable
- A zoo of resource/service providers distributed worldwide with various benefits and constraints
  - The grid, commercial cloud service providers, High-performance computing (HPC) and Leadership Computing Facilities (LCFs), volunteer computing, Platform-as-a-Service (PaaS), Function-as-a-Service (FaaS), …
- Increasing importance of complex and emerging workflows
  - Various resource/service requirements even in a single workflow, to leverage optimal services for a part of the workflow
  - New challenges not foreseen in traditional HEP workflows

A Simple Usecase with Multiple Remote Providers

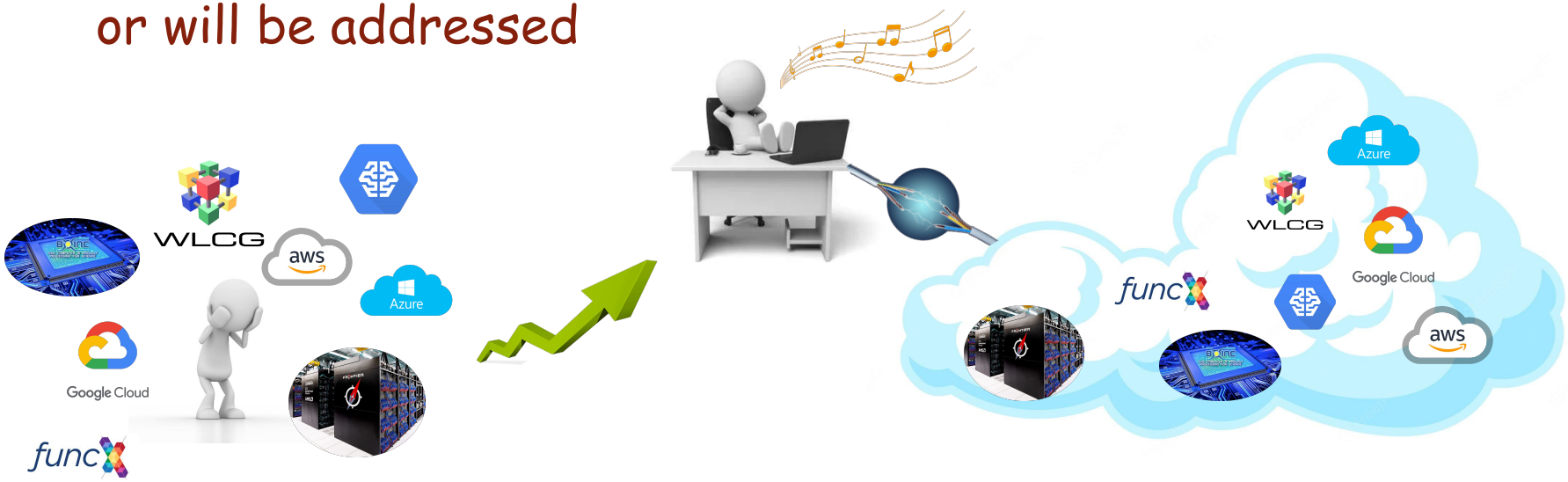
- To produce Monte-Carlo (MC) samples with a Machine Learning-based (ML-based) detector simulation
  - Three tasks in the workflow
    - Training data production, ML training, MC simulation
- Different resource/service requirements for each task
  - Training data production and MC simulation
    - Traditional High Energy Physics (HEP) workloads
    - CPU/IO intensive → WLCG
  - ML training
    - GPU intensive → LCF or PaaS/FaaS
- New resource providers (e.g., LCF) or service providers (e.g., PaaS/FaaS) for emerging workloads
  - PaaS/FaaS not versatile but powerful for specific workloads (e.g. ML workloads)



Goals of Distributed Heterogeneous Computing

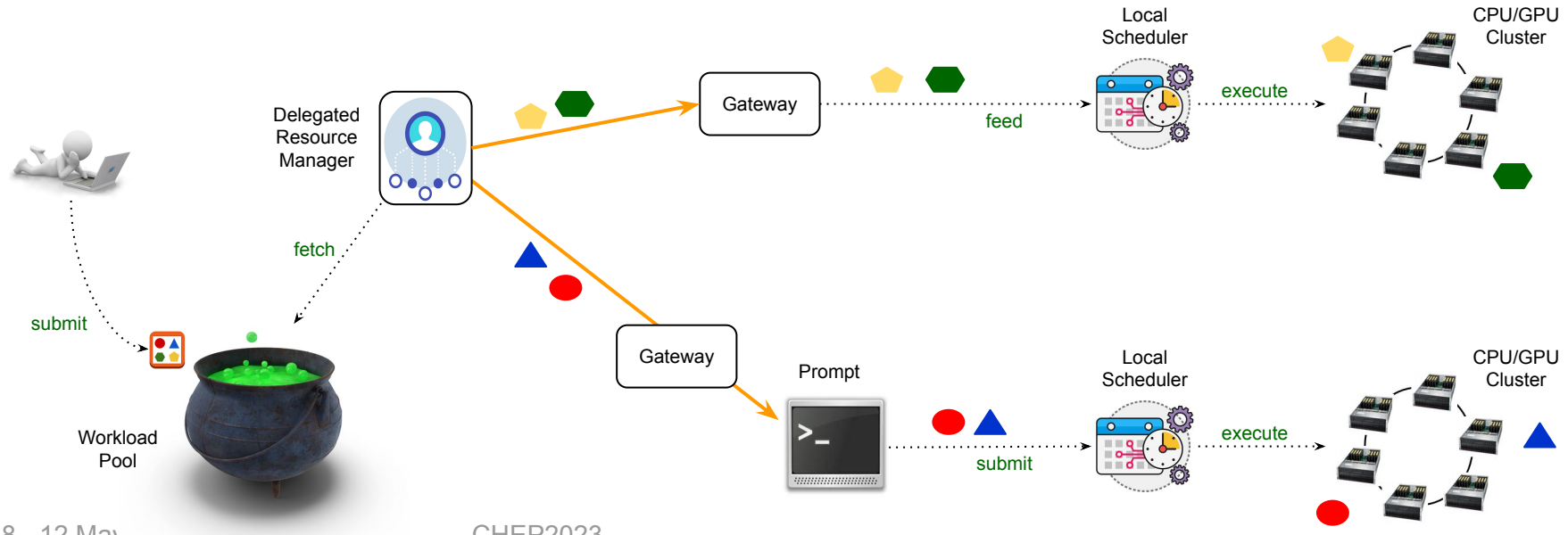
- The ultimate goal is to hide details in distributed remote resources/services from users, but also drill-down transparency and diagnostics, and bring an easy and quasi-interactive interface with quick turnaround
  - Isolation of users from resources/services
  - Smart workload routing
  - Dynamics resource provisioning and cost saving
  - Automatic data placement and software deployment
  - Seamless complex workflow execution

Will showcase in next slides how those issues have been or will be addressed



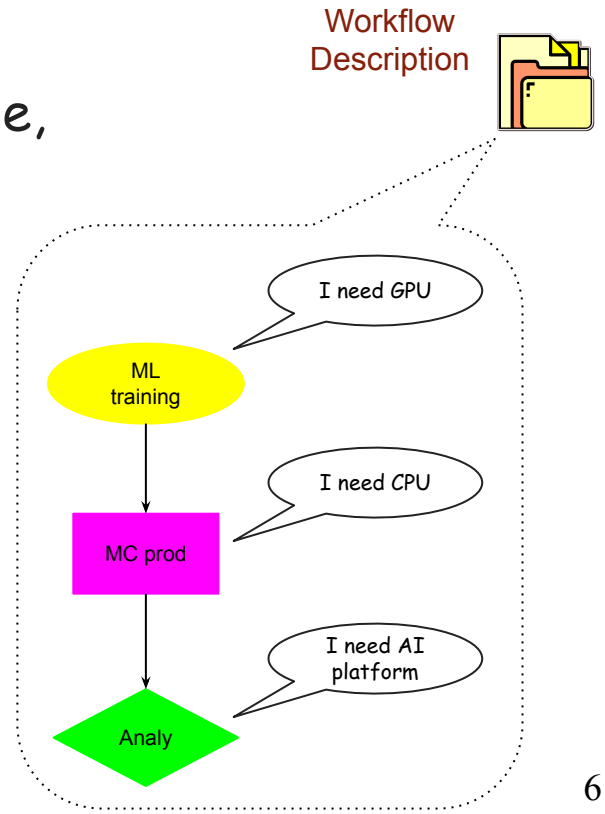
Hiding Remote Resource Access

- Central workload pool + delegated resource access
  - The user submits workflows to the central workload pool where workflows are decomposed to tasks and jobs
  - The delegated resource manager accesses resources on behalf of users using common or user's credentials
  - Small agents, a.k.a. pilots, may be scheduled instead of jobs and they may fetch jobs from the pool if the scheme is applicable
- Advantages
  - Isolation between the user and resources
  - Centrally managed fair-share and priorities among multiple users
  - Workload routing based on fine-grained requirements and central knowledge of resource/service characteristics and availabilities

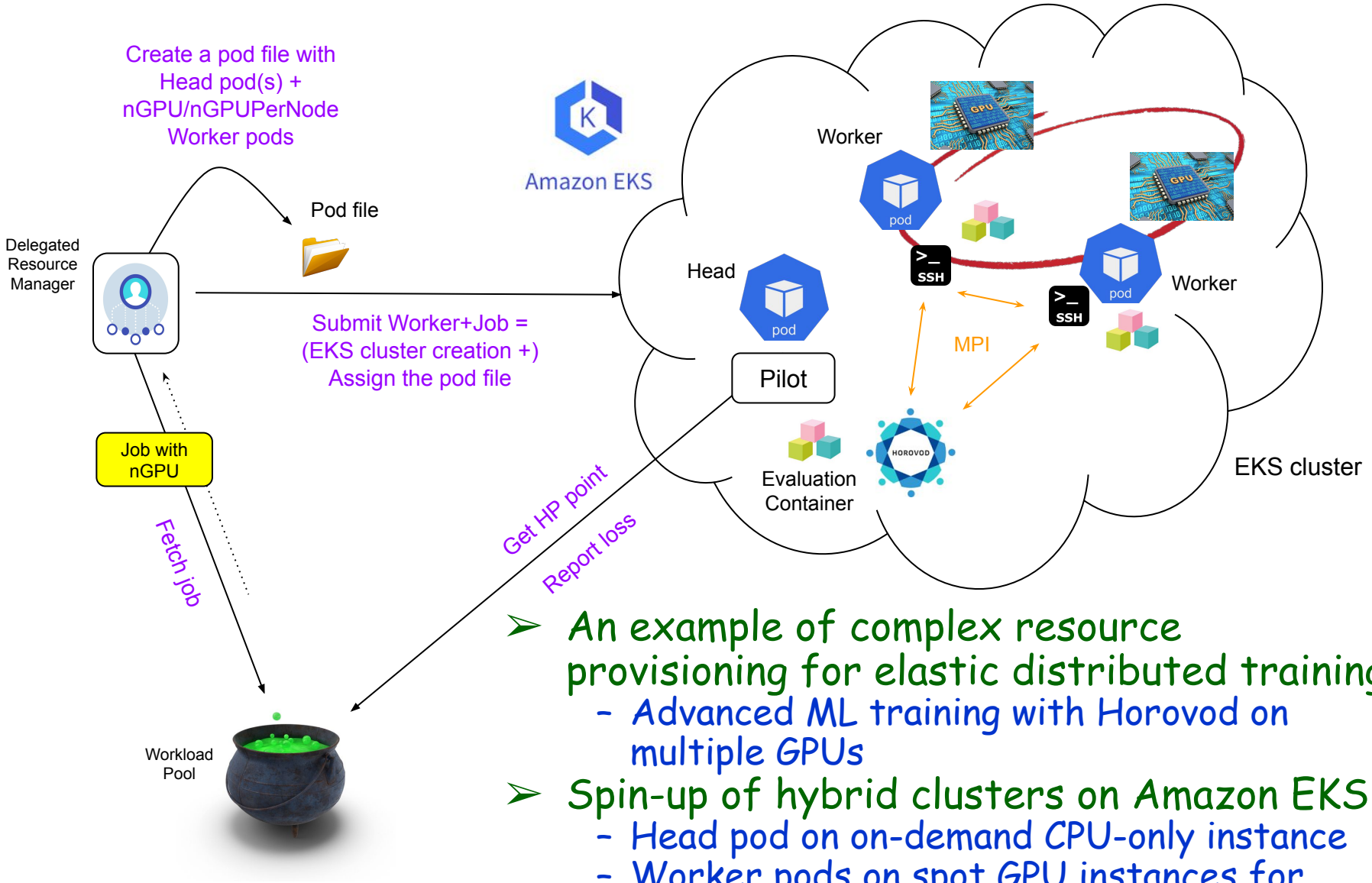


Workload Routing

- Resource/service requirements for each task in the workflow description
- Selection of providers based on matching between
  - Resource/service requirements
  - Published and real-time information of providers
    - Hardware (HW) spec
    - Service description
    - The number slots currently available
    - Data locality
    - Allocation
    - To consider also costs, electricity price, carbon footprint, …
- The user may wrongly specify resource requirements
  - E.g. too large/small RAM requirements
  - Dynamic correction of the requirements based on actual resource usage in the first few jobs before processing all the rest
    Scouts → Avalanche



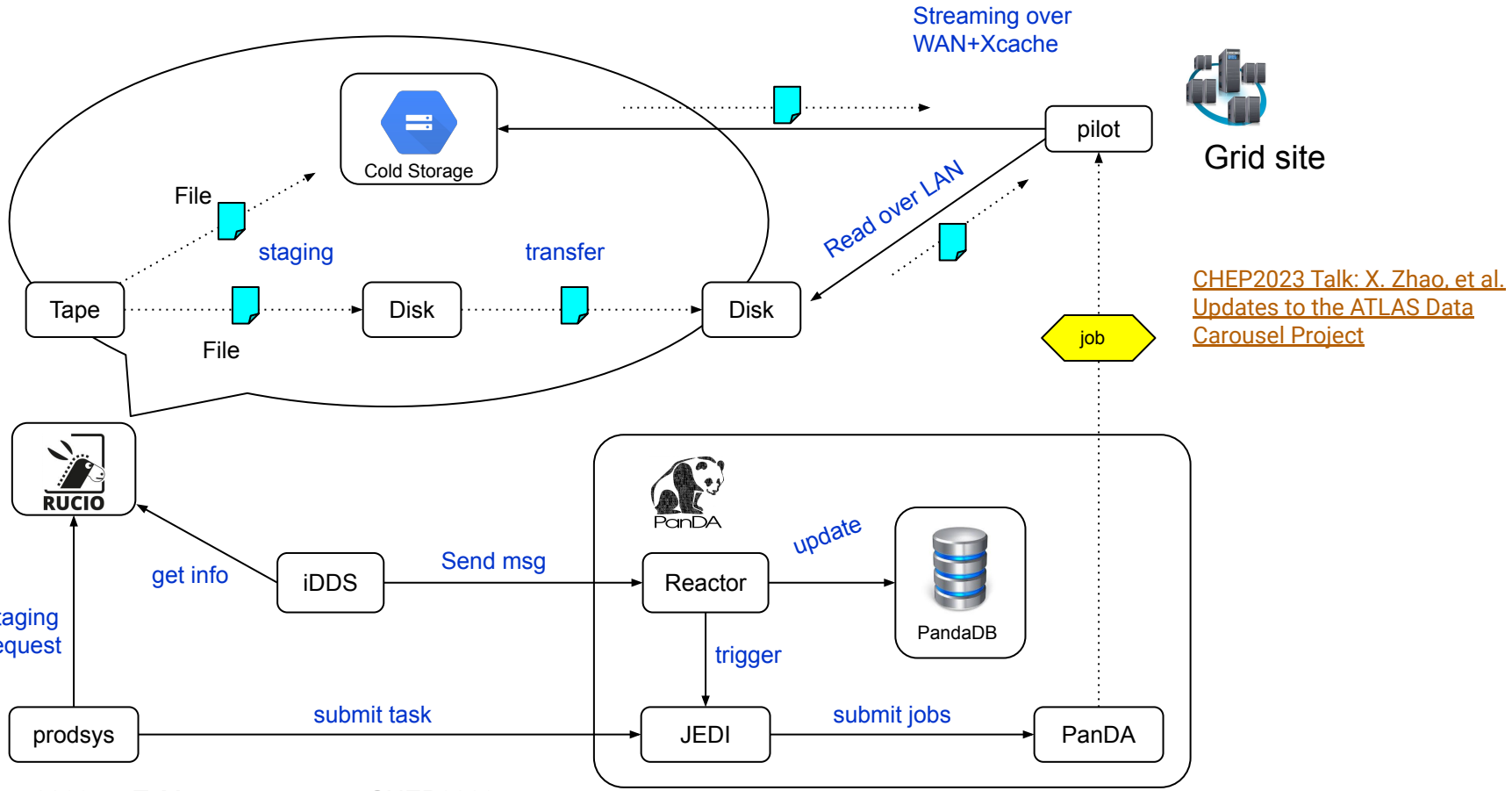
Complex Resource Provisioning



- An example of complex resource provisioning for elastic distributed training
  - Advanced ML training with Horovod on multiple GPUs
- Spin-up of hybrid clusters on Amazon EKS
  - Head pod on on-demand CPU-only instance
  - Worker pods on spot GPU instances for cost-saving + dynamic scaling-up/down of Horovod

Automatic Data Placement

- Data placement is vital and must be fully automated in complex workflows
- E.g. Data carousel
  - Fine-grained disk-efficient tape staging
  - On-demand disk replicas and their aggressive deletion
  - Mitigating the chronic shortage of disk in ATLAS



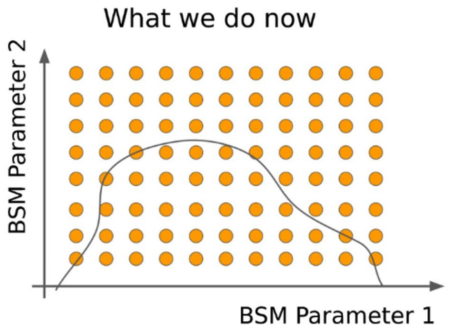
CHEP2023 Talk: X. Zhao, et al. Updates to the ATLAS Data Carousel Project

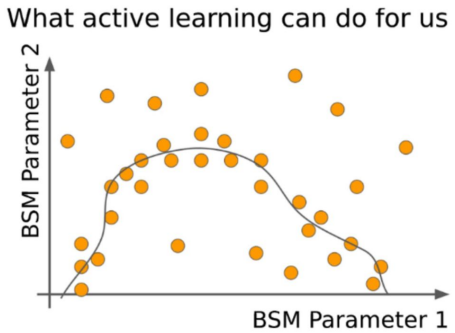
Workflow Description

- Many workflow description languages are available on the market, but most of them are tightly coupled with the workflow processing backends, or cannot be easily enhanced to work with heterogeneous resources/services
- Quite tricky to describe complex workflows
  - E.g., no common language across ML training, MC production, and Analysis on a cloud platform
  - Requires sophisticated syntax to specify resource or service requirements translated to HW specs or service descriptions in various providers
  - The user has to explicitly specify parallelizable or critical sections and relationship among tasks/jobs in the workflow, unless they are embarrassingly parallel and can be described declaratively
- Workflows described in DAGs with yaml+CWL or python+snakemake
- GUI to be developed to attract newbies
- iDDS: intelligent Data Delivery Service as workflow execution engine

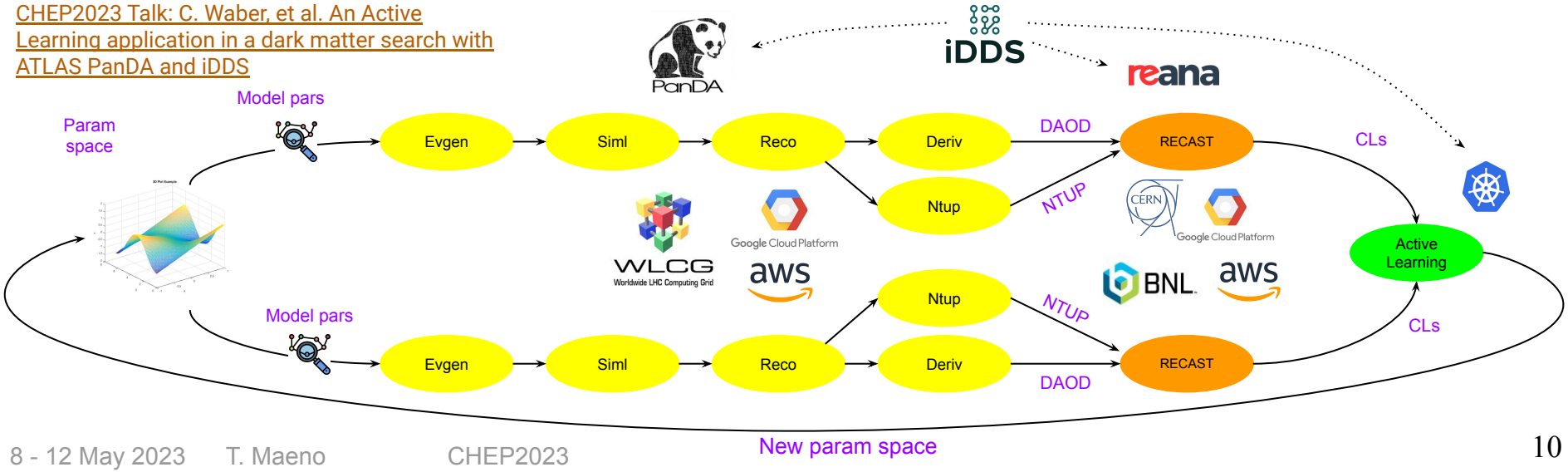
Seamless Execution of Advanced Workflows

- Active Learning (AL) using iterative regression on limit setting to increase analysis efficiency
  - Traditional: A brute-force grid/random search
  - New: A search session with multiple iterations
    - The search space for the next iteration is defined based on the results of old iterations
- A single workflow with multiple loops
  - Various tasks on different resource/service providers
  - No human intervention in transitions from one provider to another
- AL-based analysis for mono-Hbb exclusion limit, dark-Z search, and heavy higgs search in ATLAS





CHEP2023 Talk: C. Waber, et al. An Active Learning application in a dark matter search with ATLAS PanDA and iDDS

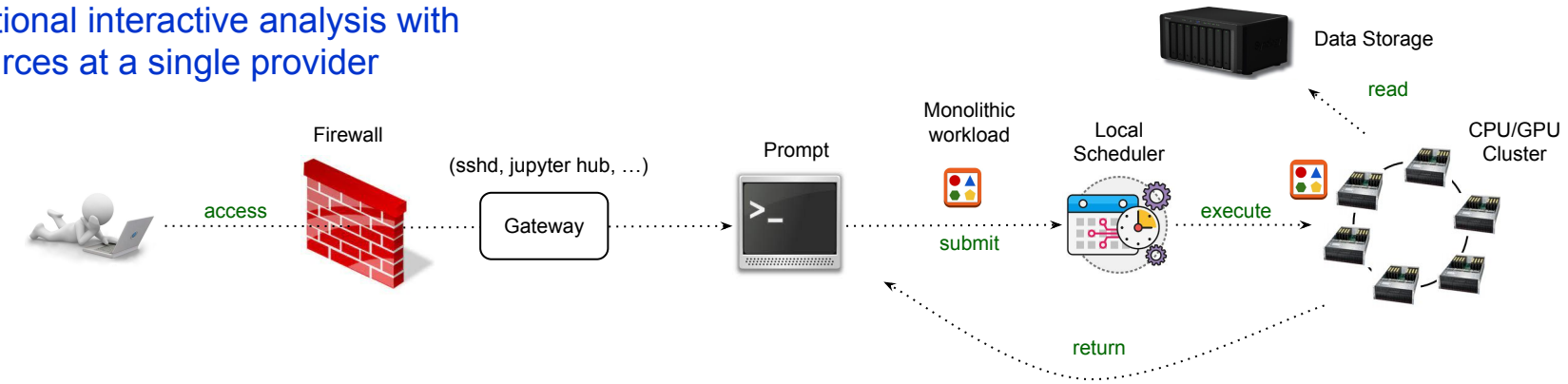


Pseudo Interactive Analysis

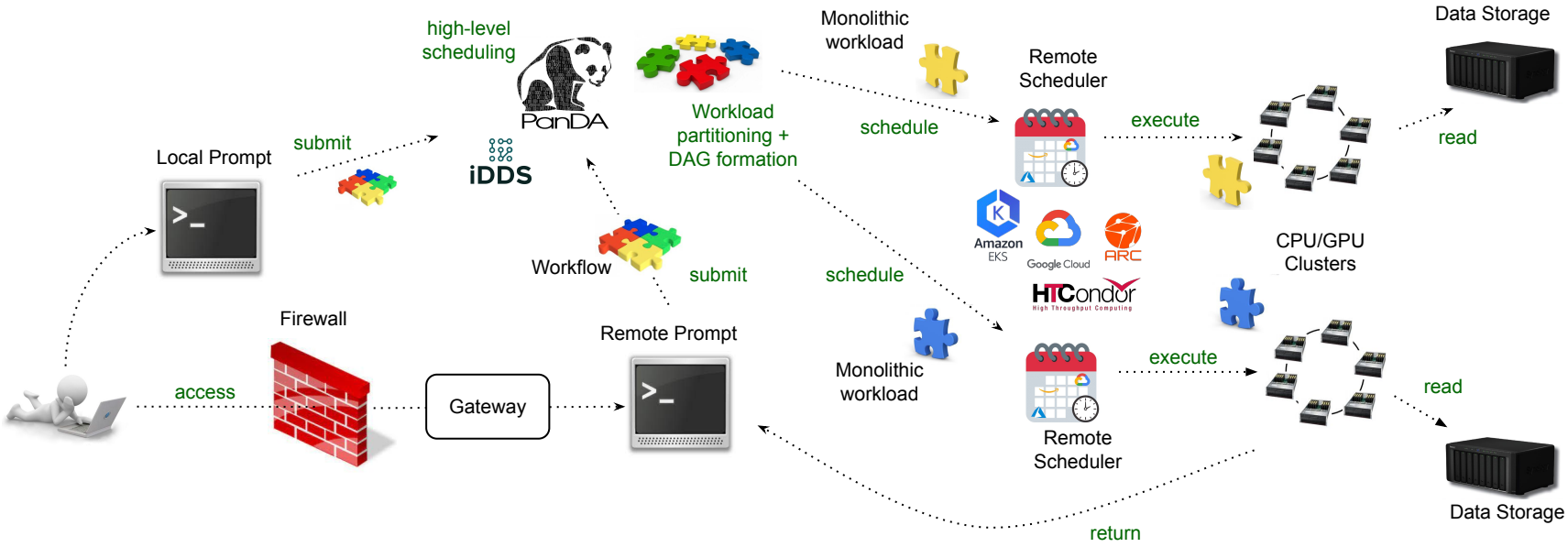
- Interactivity
  - One of hot topics in ATLAS
  - The ability of a system (user interface + processing backend) to respond quickly to the actions of the person using it
  - != Interactive user interface
    - End-users don't feel any interactivity in combination of interactive user interface and slow processing backend, e.g., when synchronously running a big payload on slow backend through Jupyter notebook
  - The response time in processing backend is more important
- Pseudo interactive system
  - Interactive system on top of asynchronous resources
    - Geographically-distributed asynchronous processing backend
    - Integrated monitoring with interactive user interface + realtime streaming of system and workload logging
    - Reasonable response time to process complex workflows
  - Advantages
    - One stop service isolating users from various processing backends and multiple resource types. Central accounting and fairshare. Further scale-up and support for complex workflows with more computing resources across various resource providers
  - Disadvantages
    - Latencies. More development efforts for sophisticated interface and monitoring due to lack of direct access to underlying computing resources

Traditional vs Pseudo Interactive

Traditional interactive analysis with resources at a single provider

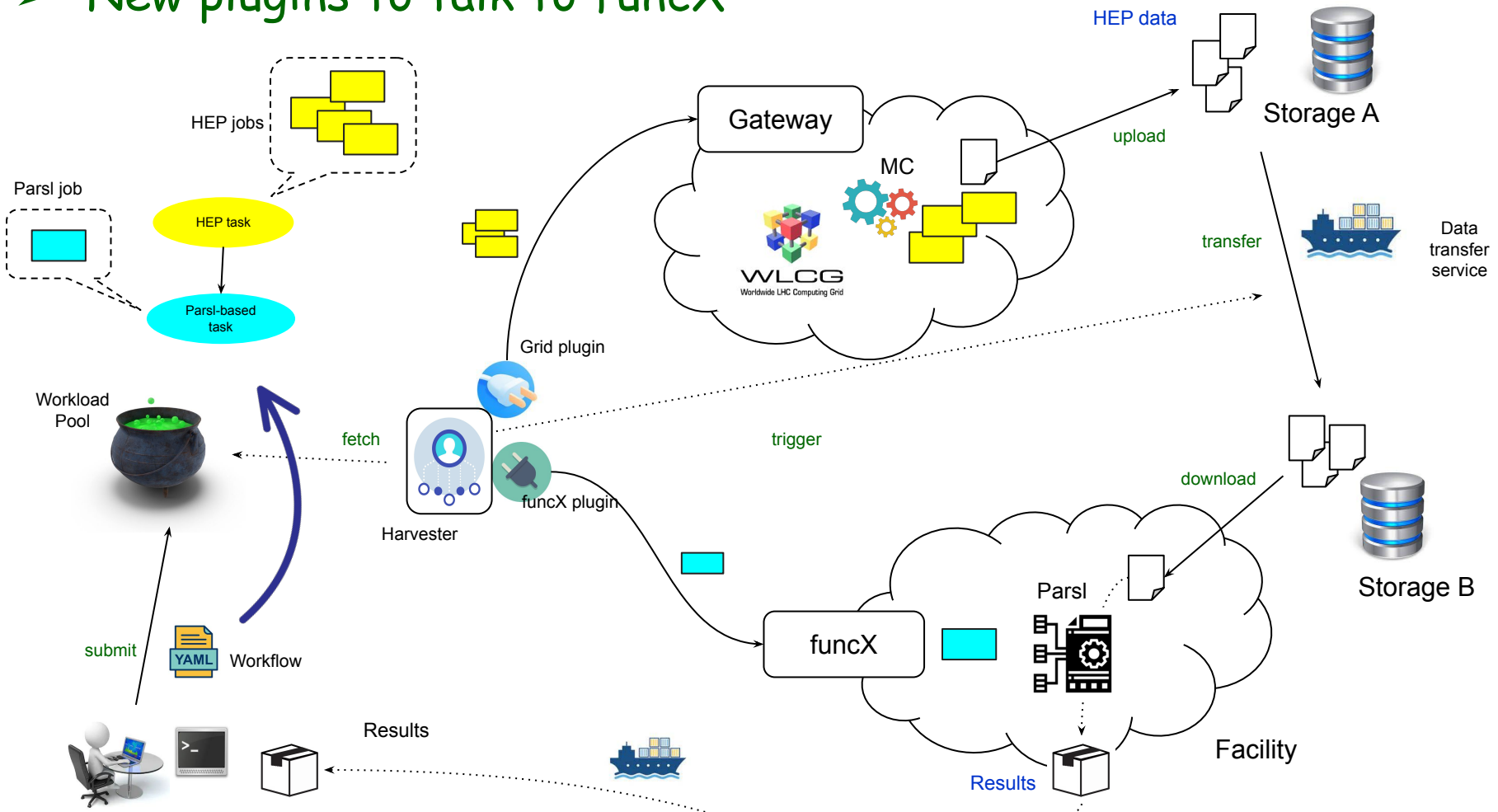


Pseudo interactive analysis with geographically distributed resources



Integration of PaaS/FaaS (e.g. funcX/Parsl)

- Parsl-based workload in a workflow
  - E.g. HEP data production → Parsl-based AI/ML
- funcX as a gateway to route parsl-based payloads to the resource at the facility such as LCF
- New plugins to talk to funcX



Conclusions

- The overall objective of distributed heterogeneous computing is to support users' workflows on available large scale resources
  - Hiding details in geographically distributed resources/services from users
  - Drill-down transparency and diagnostics
  - A quasi-interactive interface with quick turnaround
- Quite complicated due to a zoo of resource/service providers and complex and emerging workflows
  - Hard to support all possible combinations of resources, services, and workflows
  - Developments driven by real usecases
- PanDA and its work program for distributed heterogeneous computing
  - Many issues addressed, but yet many challenges to come
  - Versatility and flexibility for further evolutions
  - Ongoing efforts to grow both user applications and supported resources