

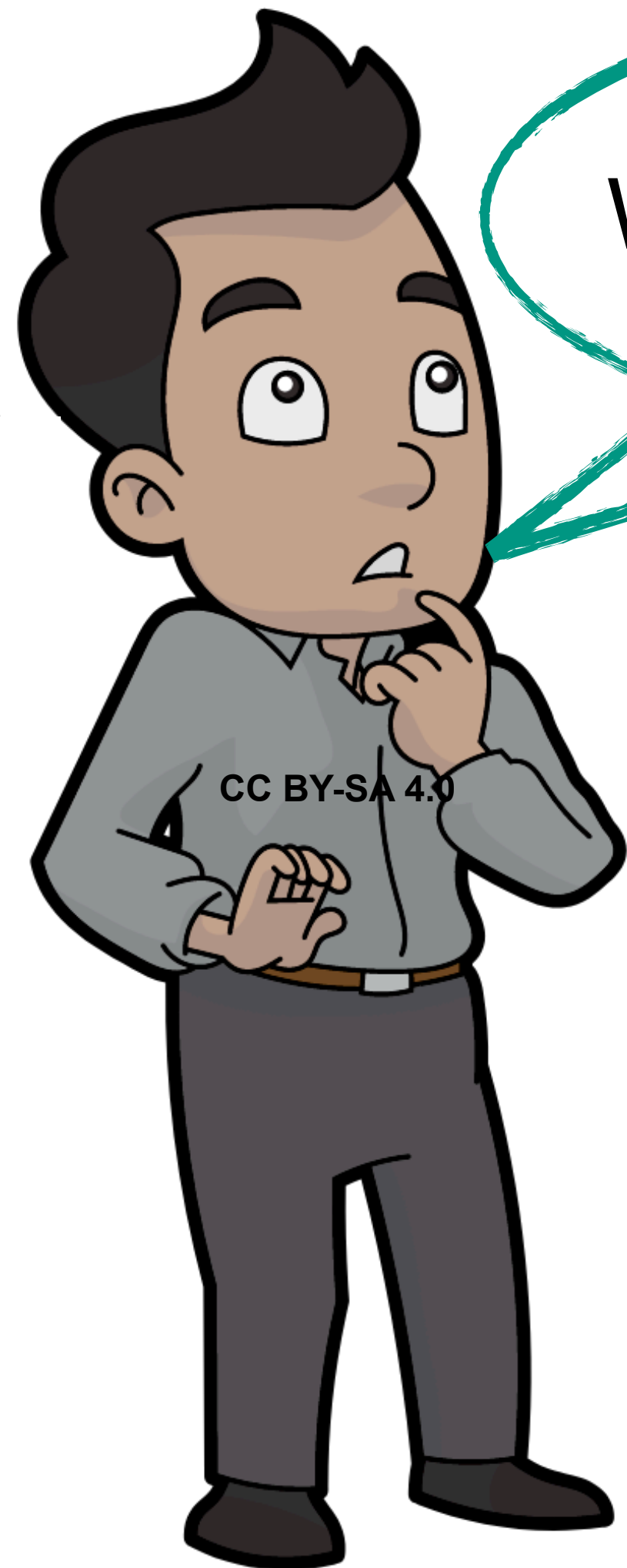
COBaID/TARDIS

The Past, Presence & Future

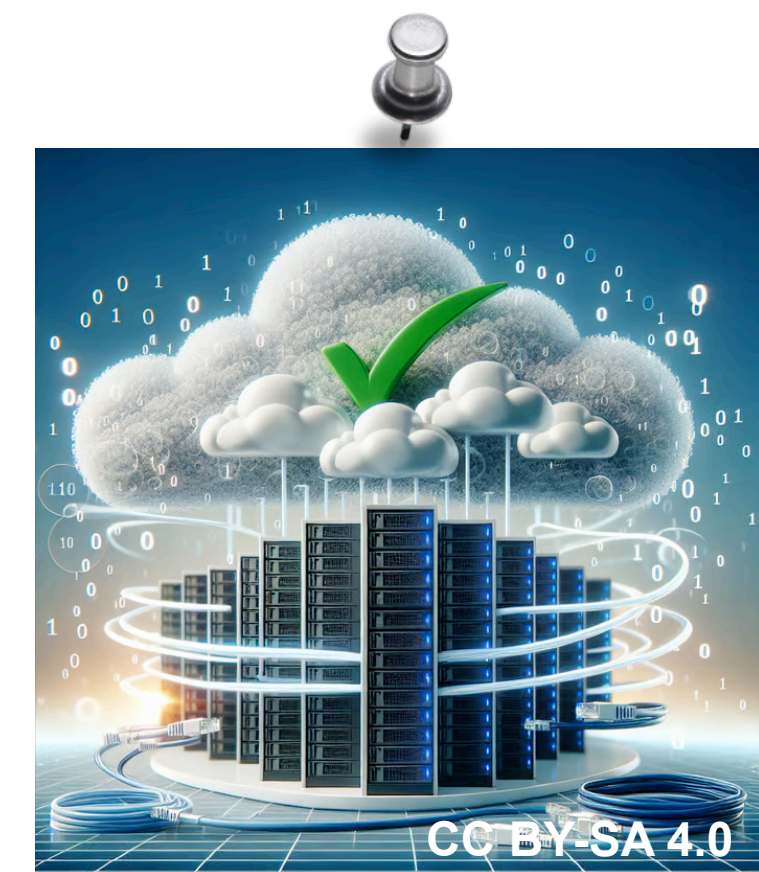
ATLAS/CMS Meeting, Freiburg, 27.03.2024
Manuel Giffels



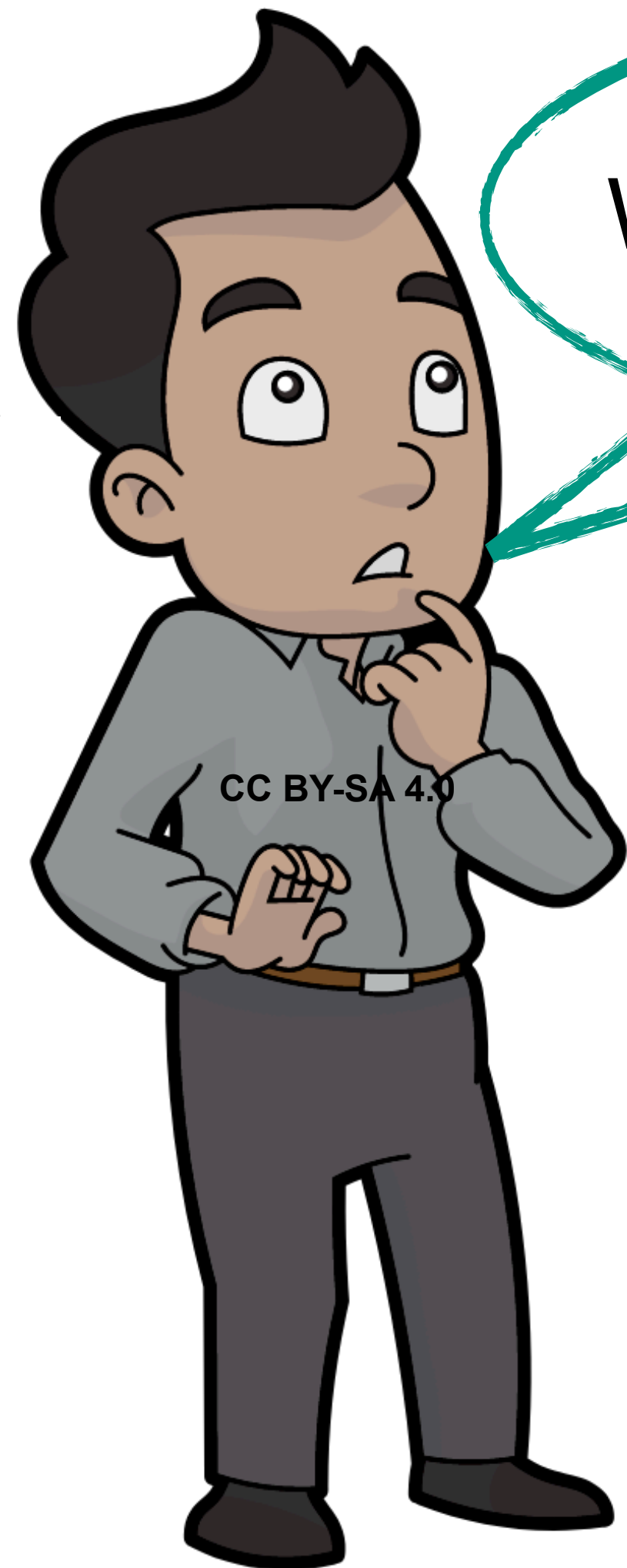
Once Upon The Time ...



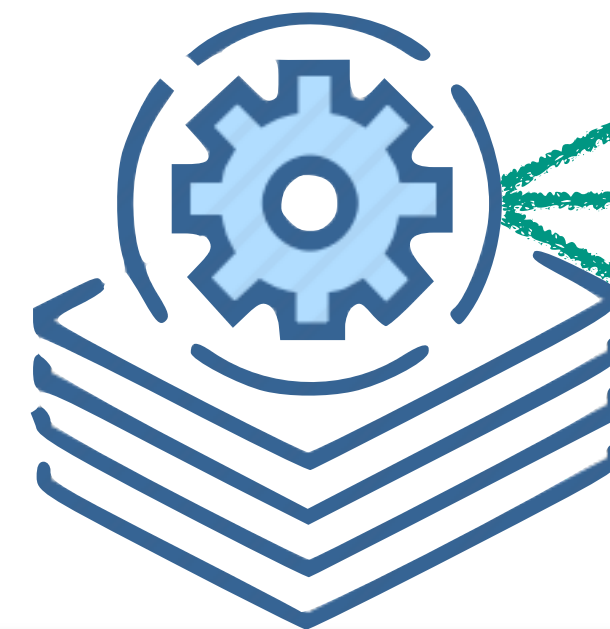
Where to send my jobs?



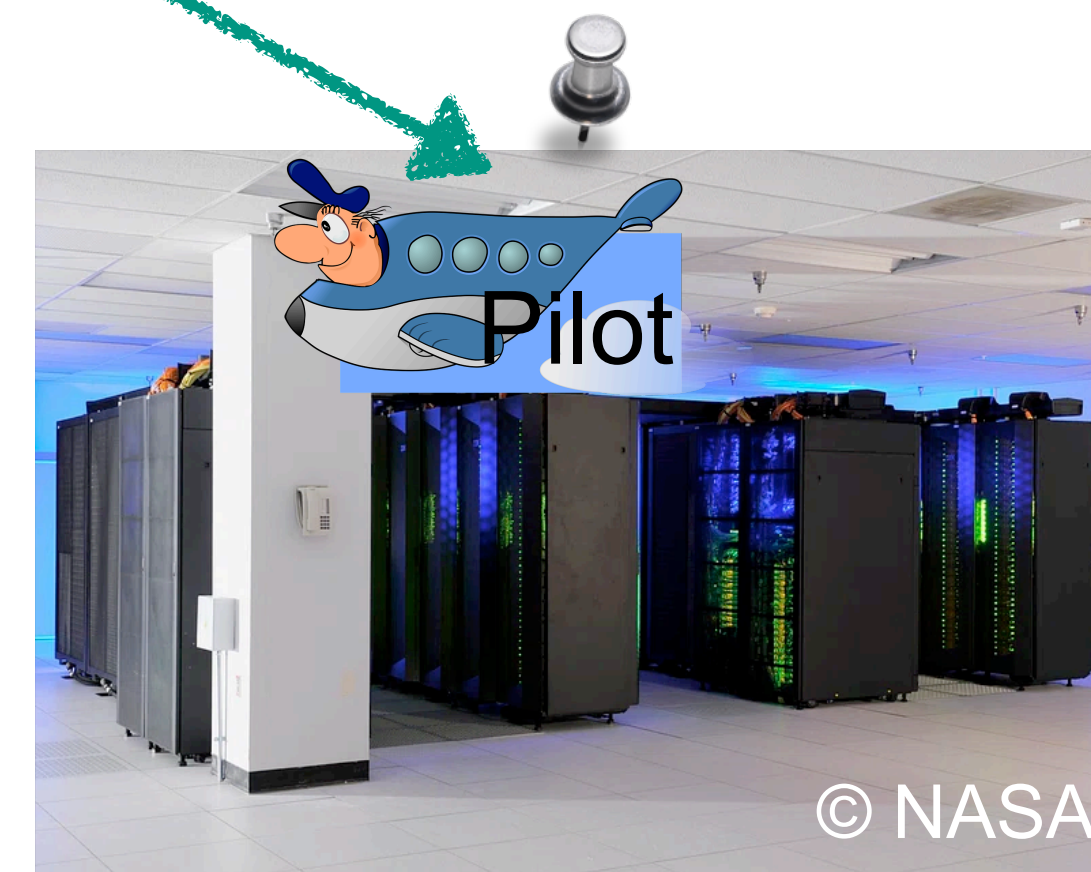
Once Upon The Time ...



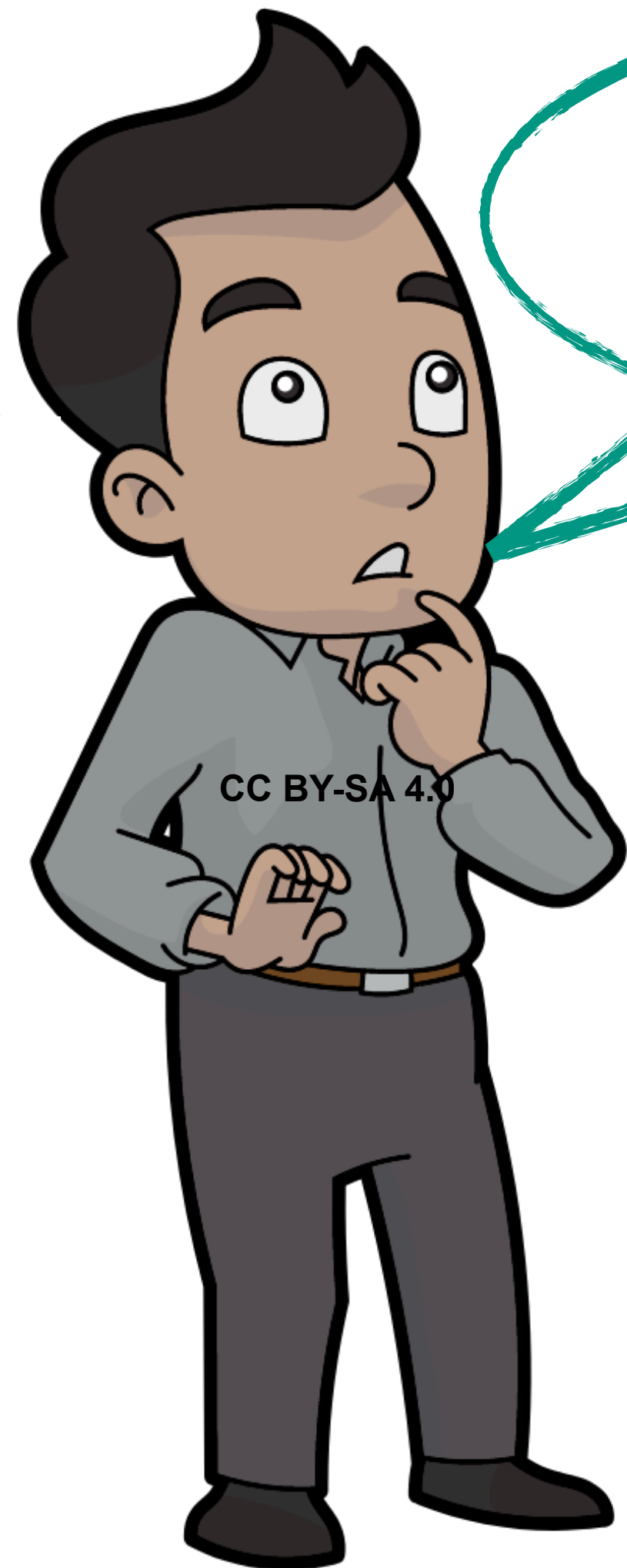
Where to send my jobs?



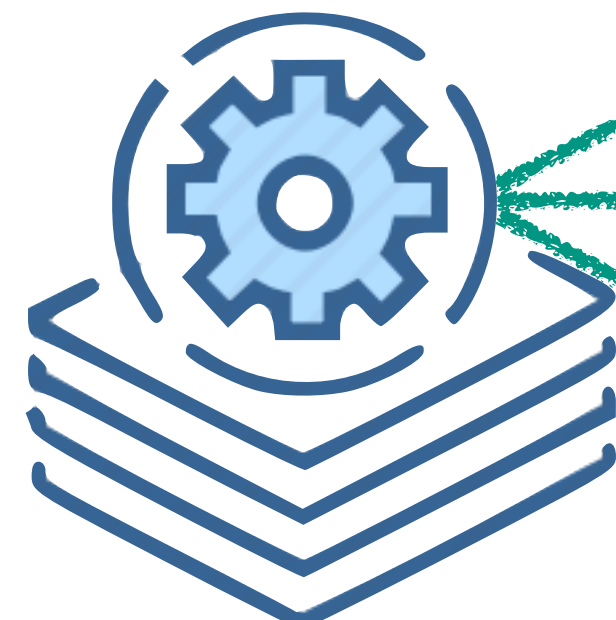
Overlay
Batch System



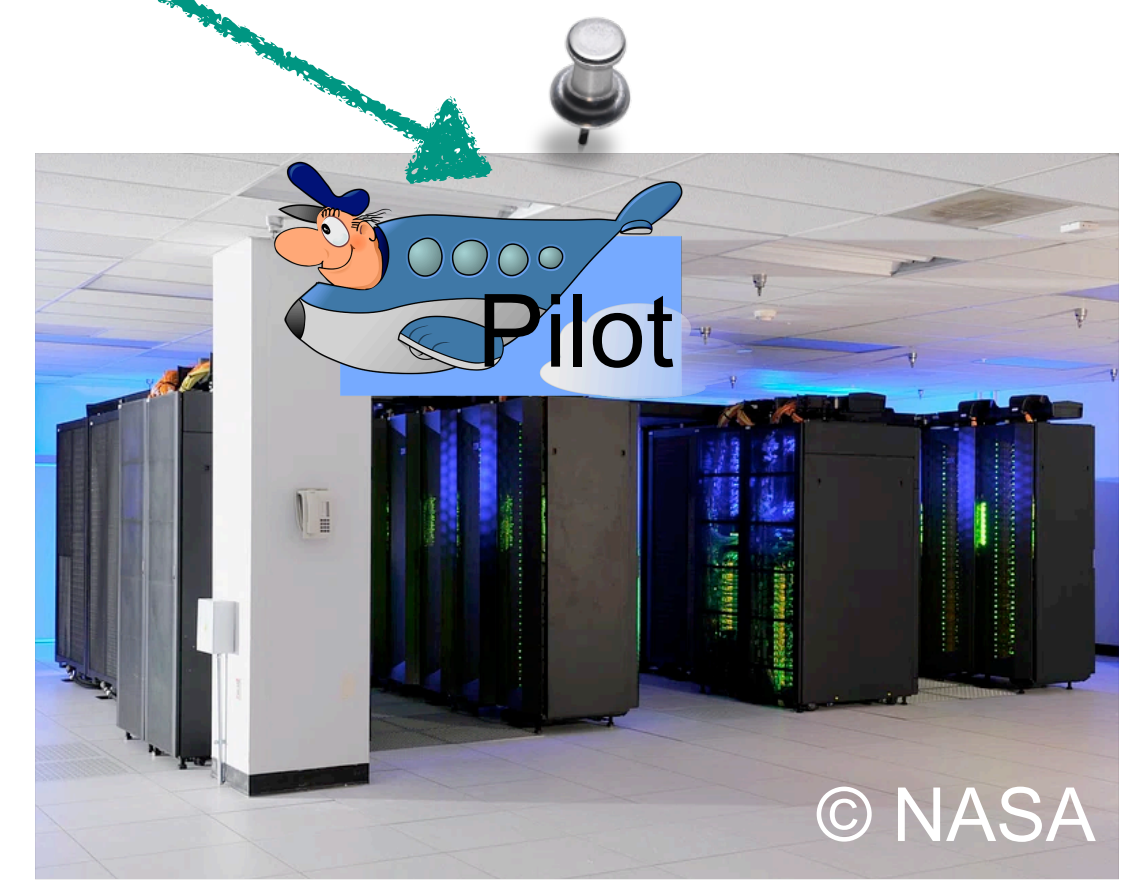
Once Upon The Time ...



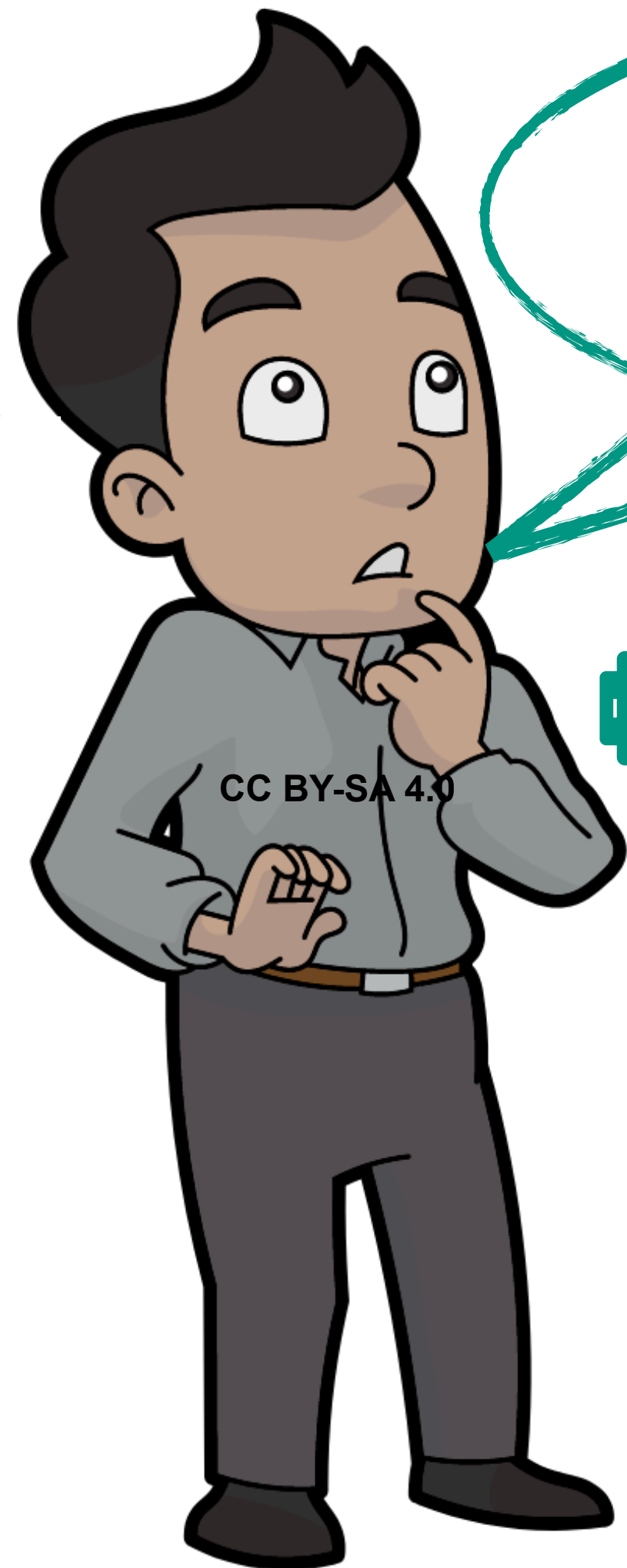
How to access it?



Overlay
Batch System



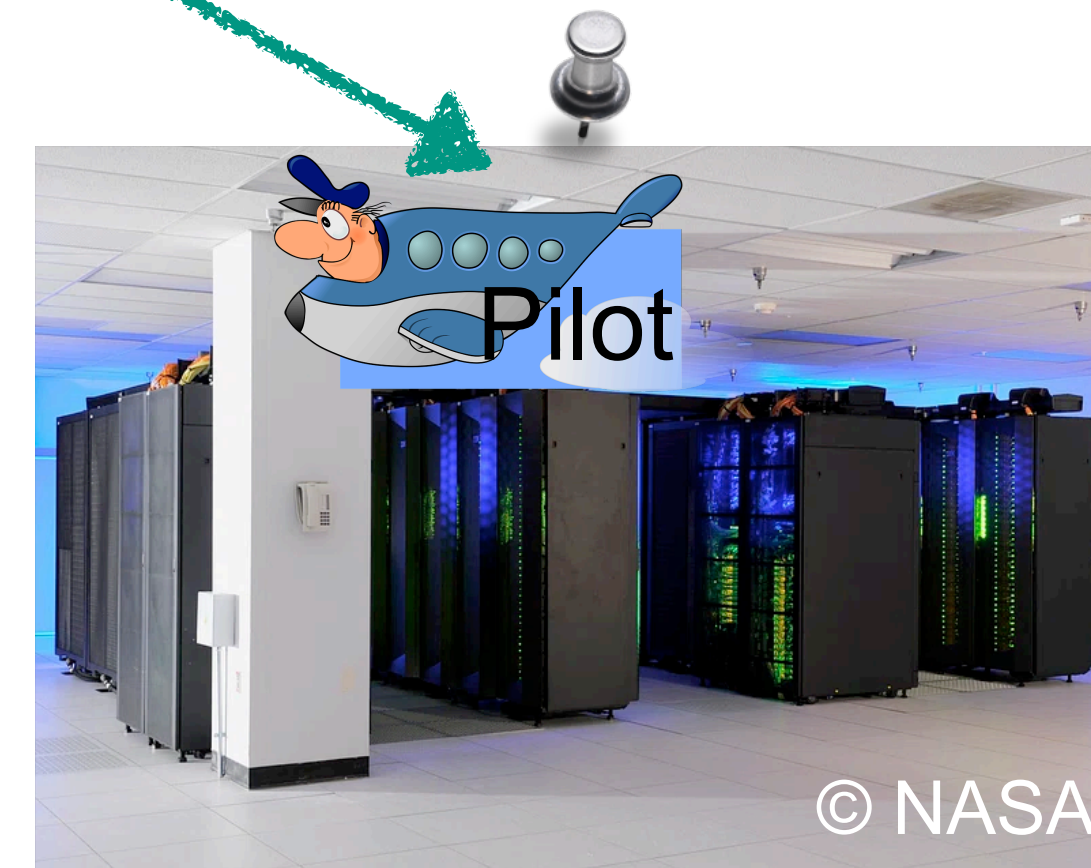
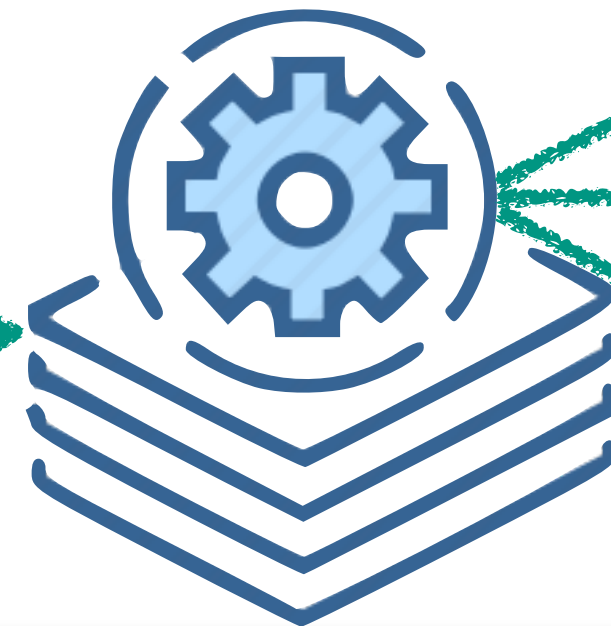
Once Upon The Time ...



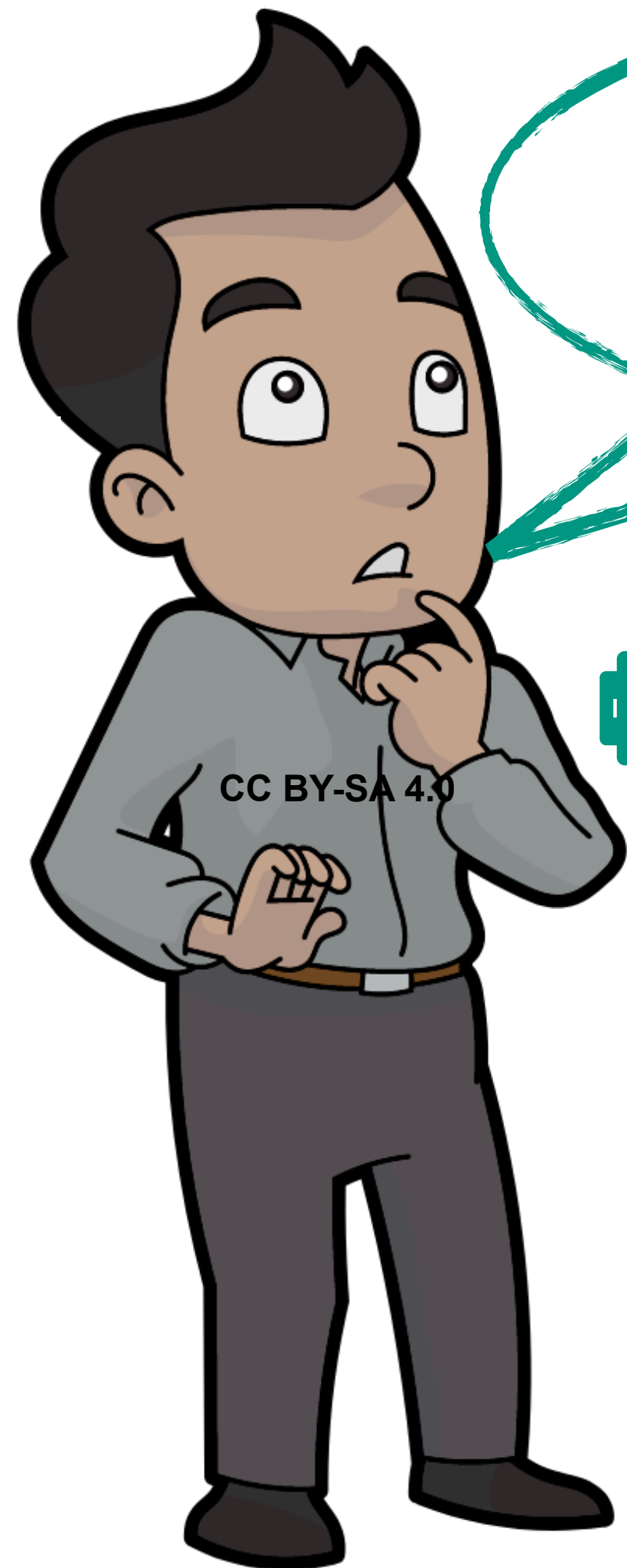
How to access it?



Overlay
Batch System



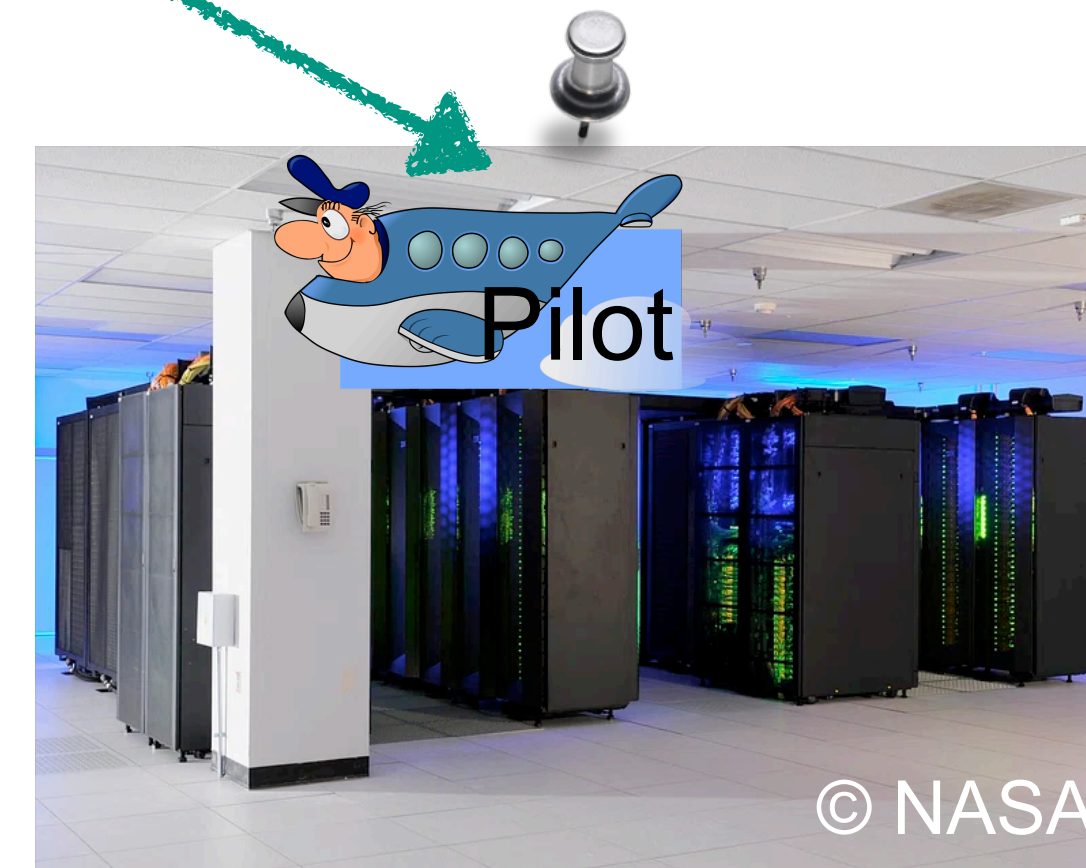
Once Upon The Time ...



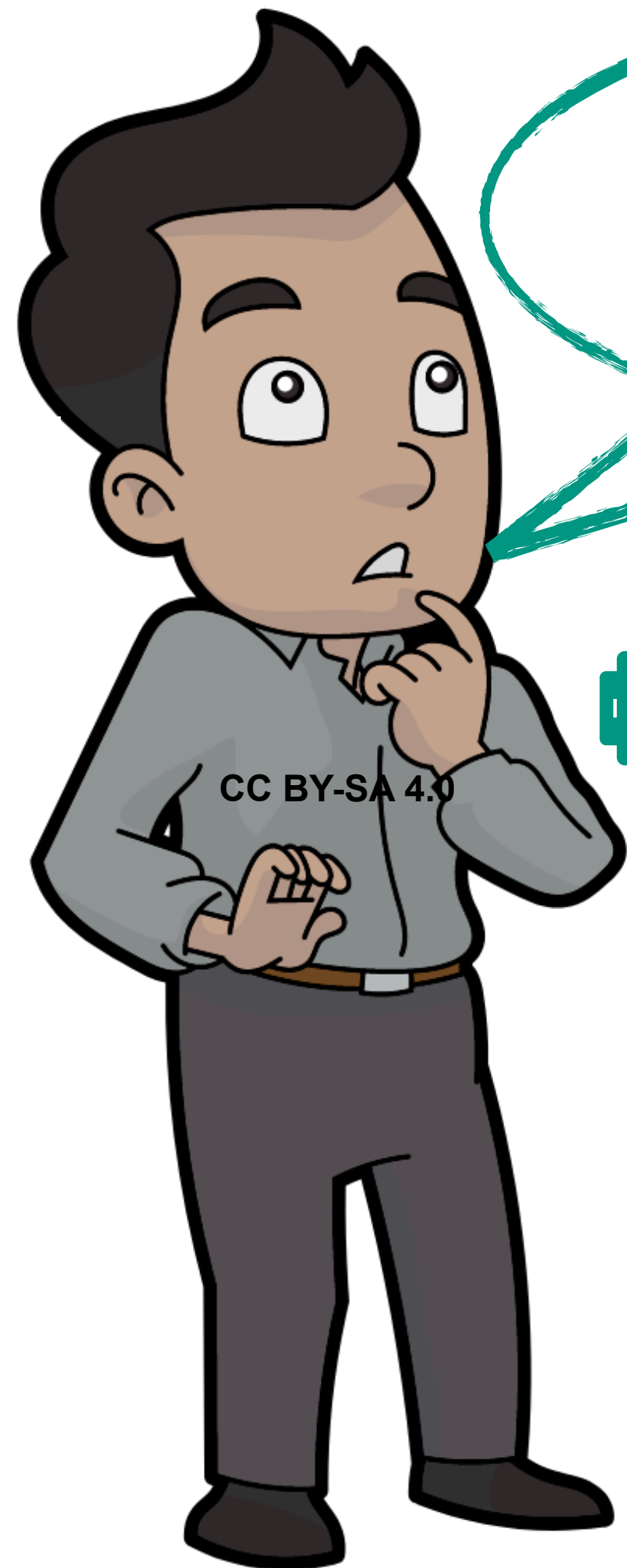
How to access my software?



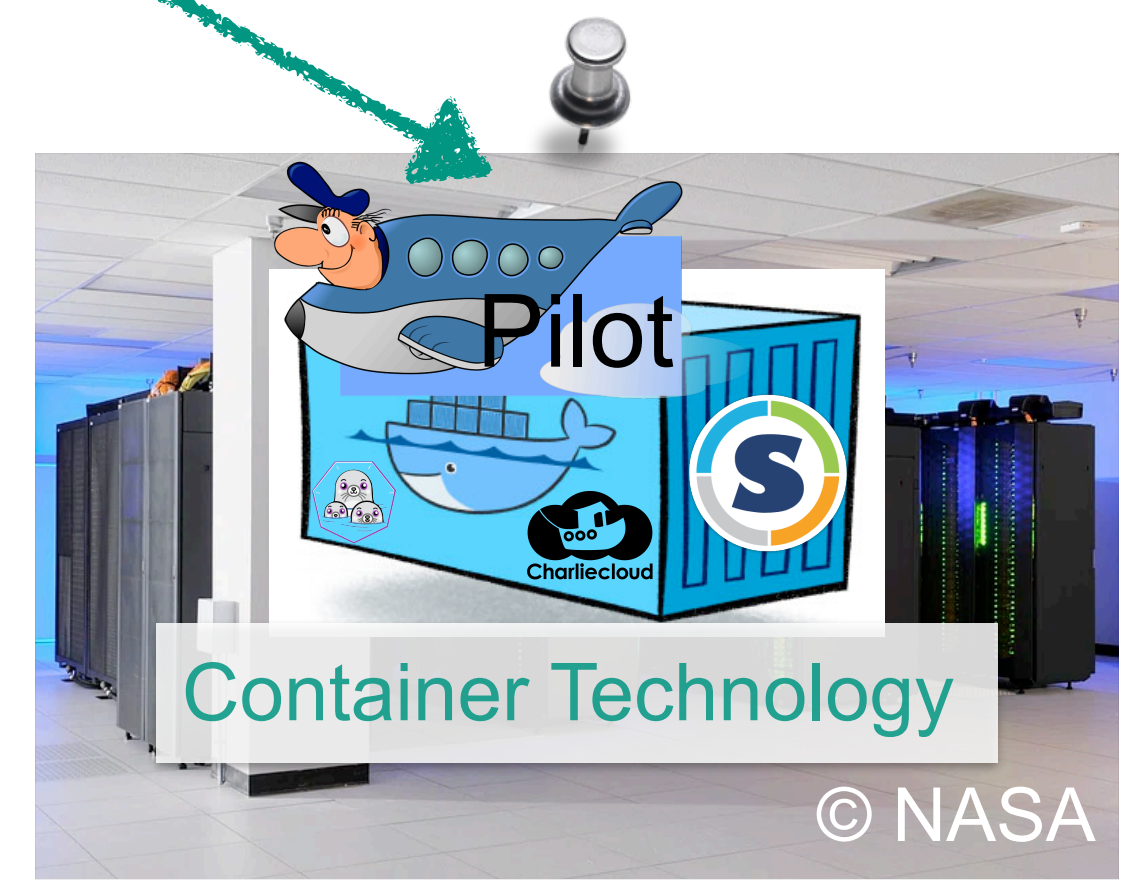
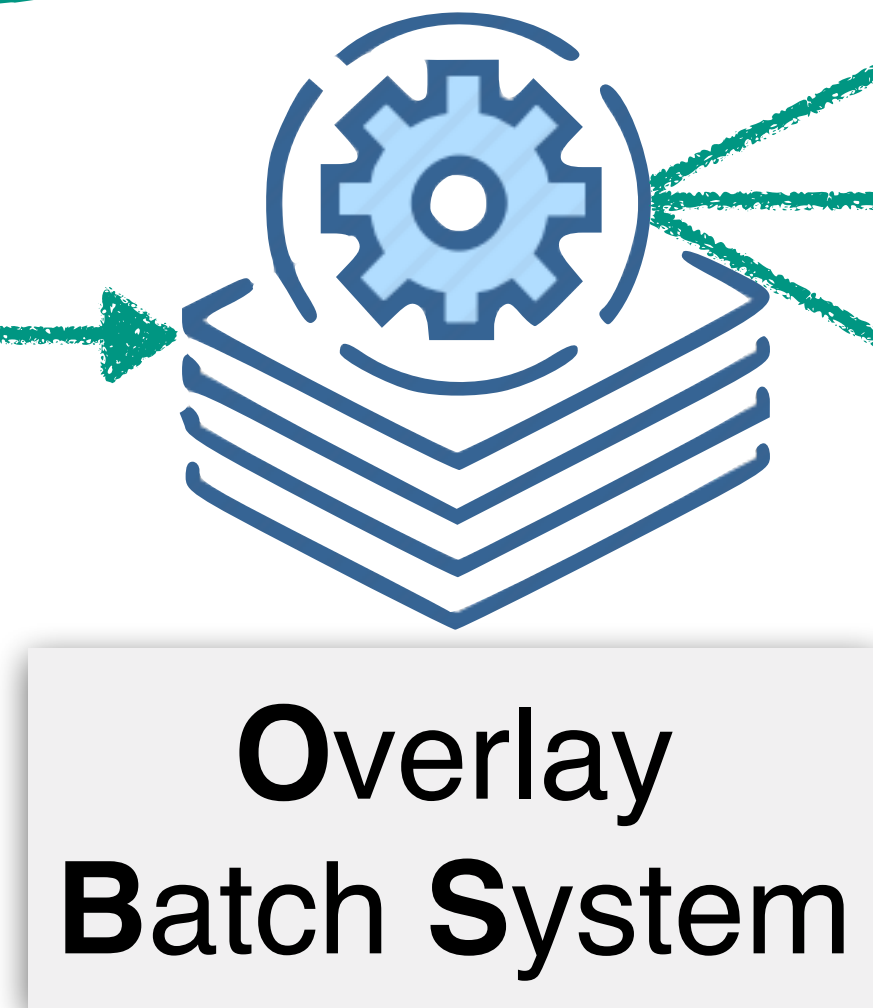
Overlay
Batch System



Once Upon The Time ...

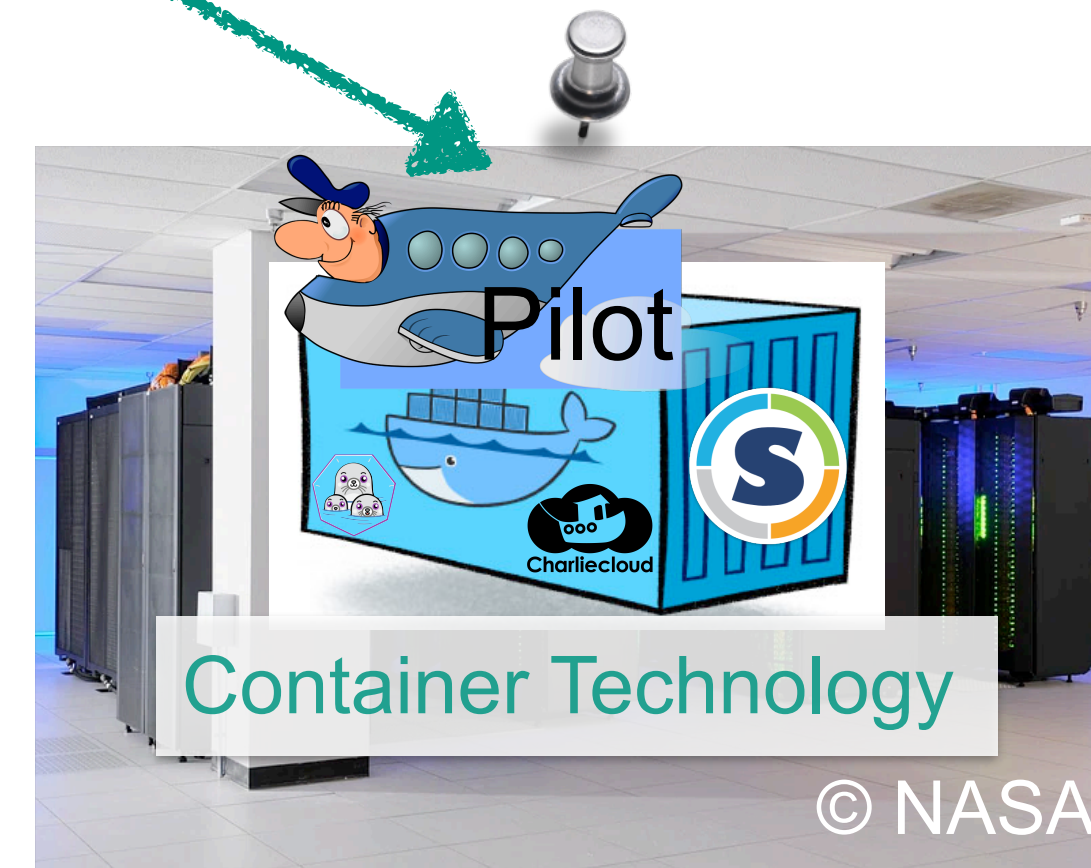
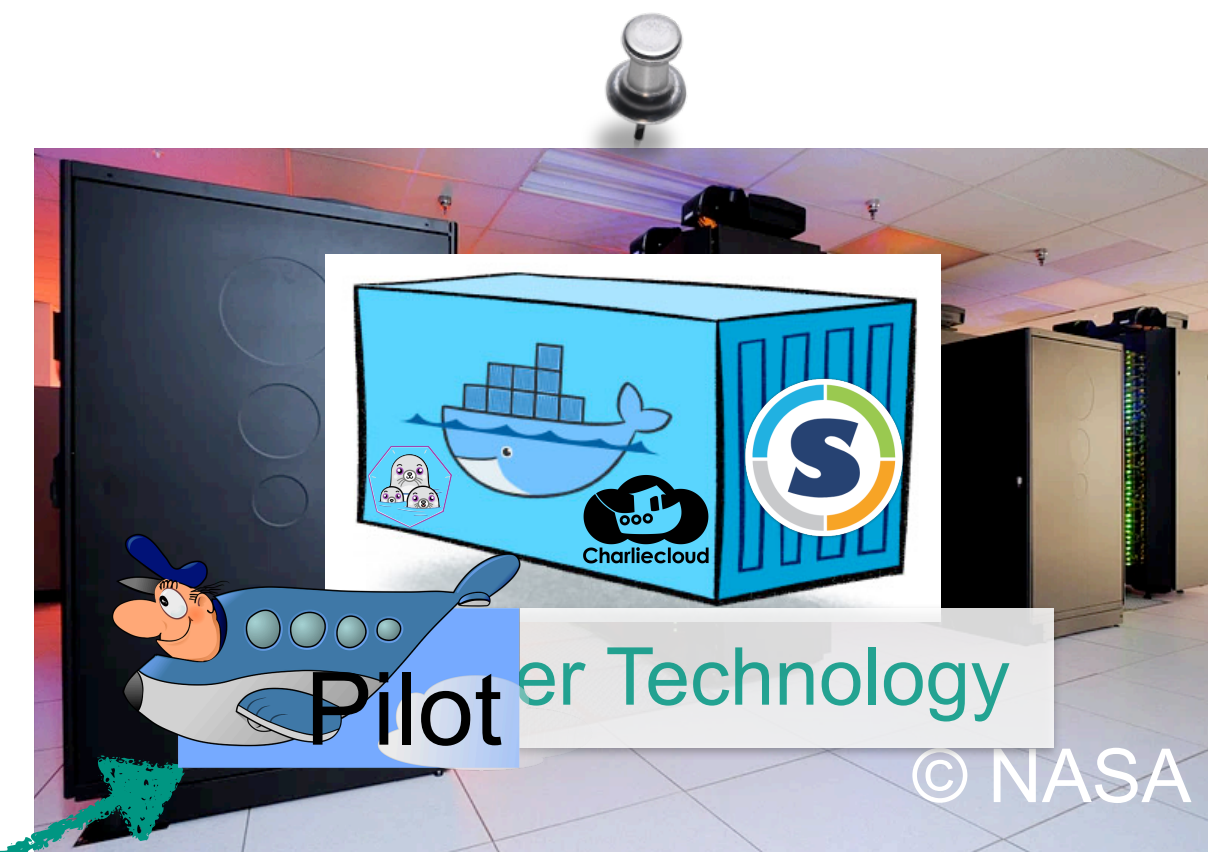
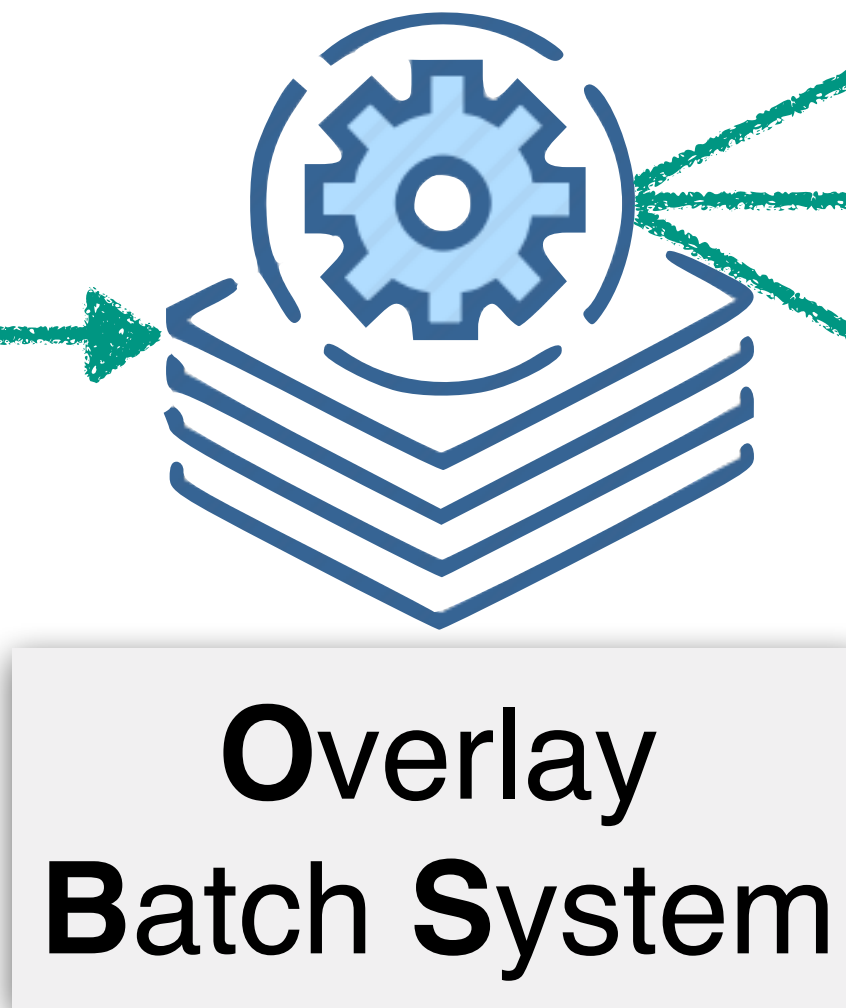
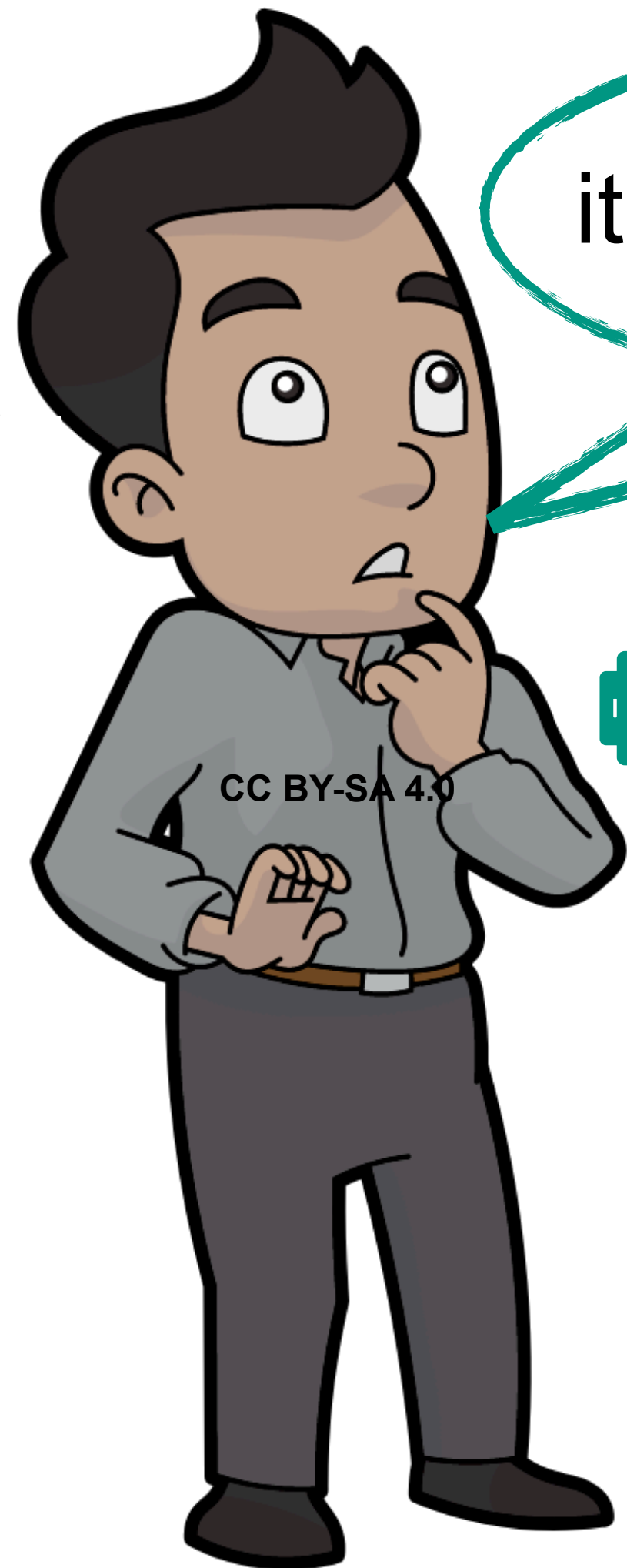


How to access my software?



Once Upon The Time ...

How does it know which resource fits to my job?



The COBaID View of Resource Meta-Scheduling

[COBaID - the Opportunistic Balancing Daemon]

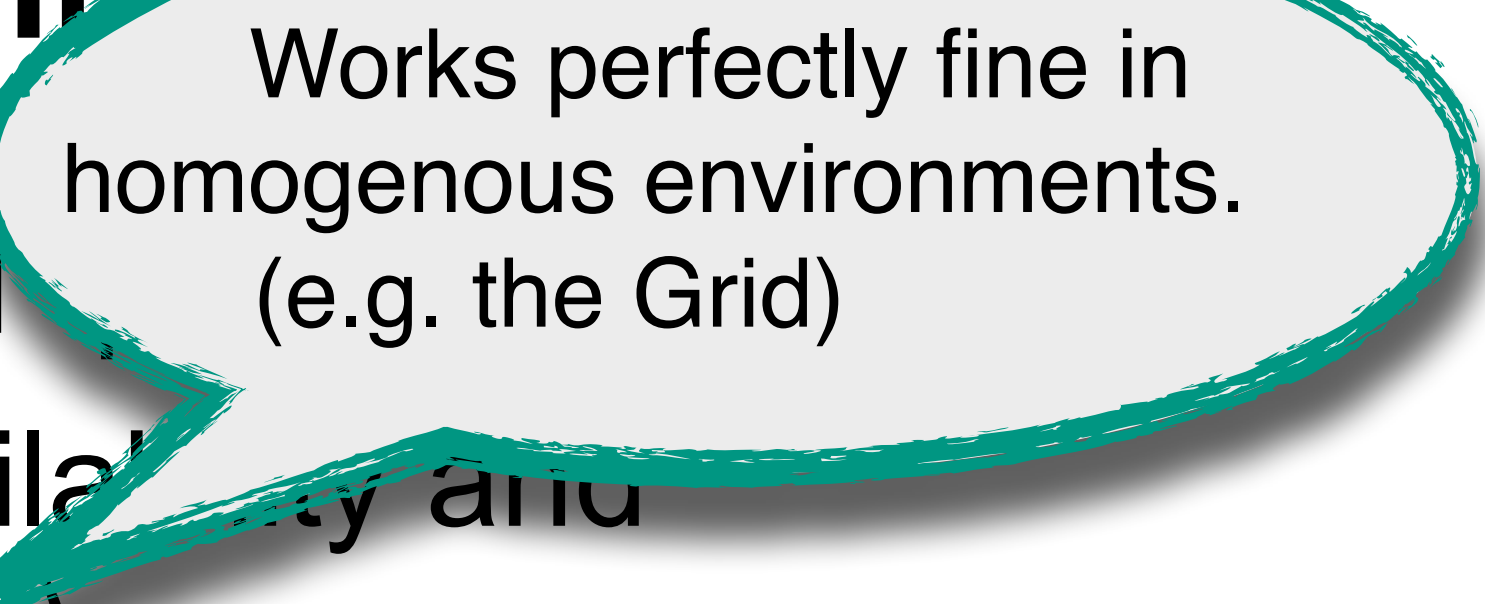
- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and mixture of job classes (e.g. CPU intense, I/O intense, ...)

Based on a slide by Max Fischer

The COBaID View of Resource Meta-Scheduling

[COBaID - the Opportunistic **B**alancing **D**aemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and a mixture of job classes (e.g. CPU intense, I/O intense, ...)



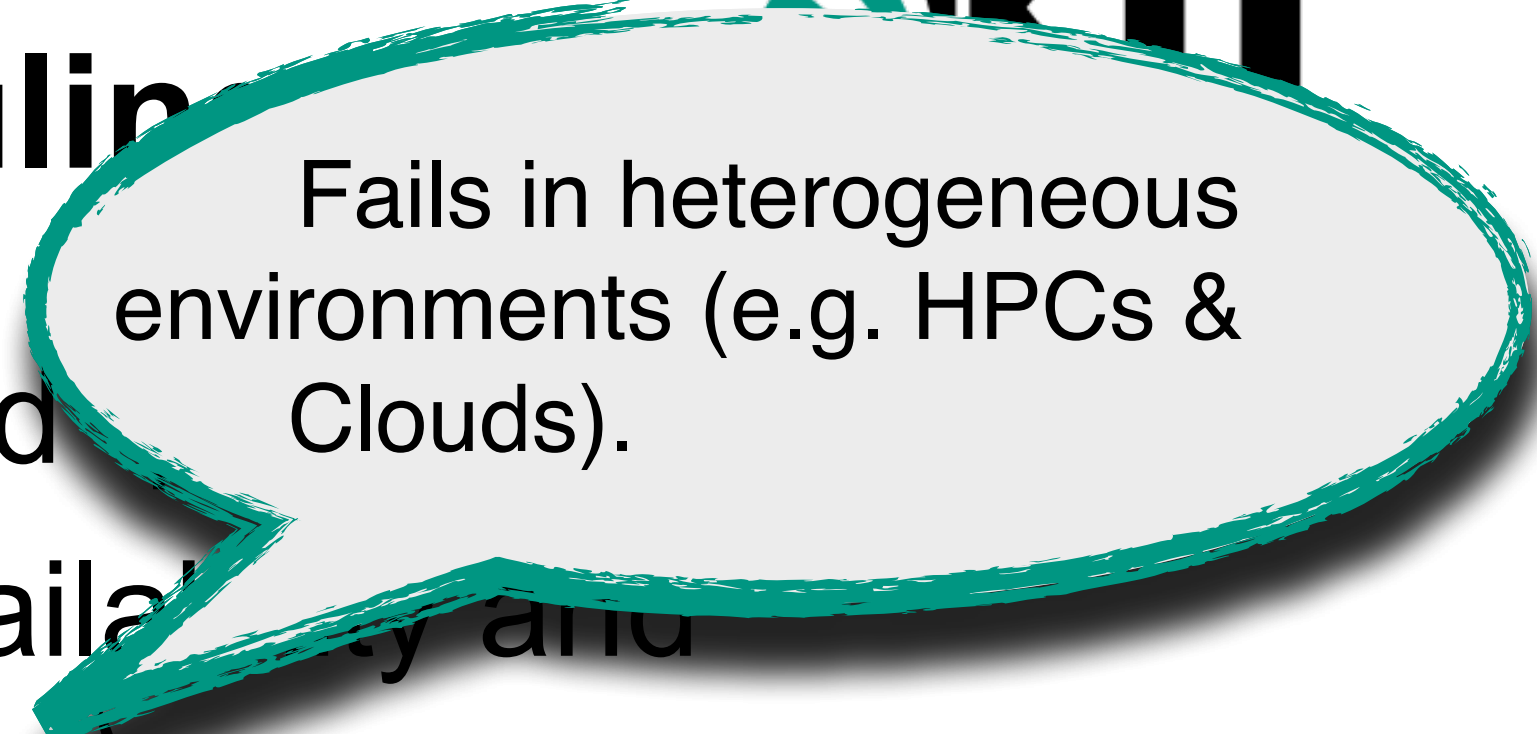
Works perfectly fine in homogenous environments. (e.g. the Grid)

Based on a slide by Max Fischer

The COBaID View of Resource Meta-Scheduling

[COBaID - the Opportunistic **B**alancing **D**aemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and a mixture of job classes (e.g. CPU intense, I/O intense, ...)



Fails in heterogeneous environments (e.g. HPCs & Clouds).

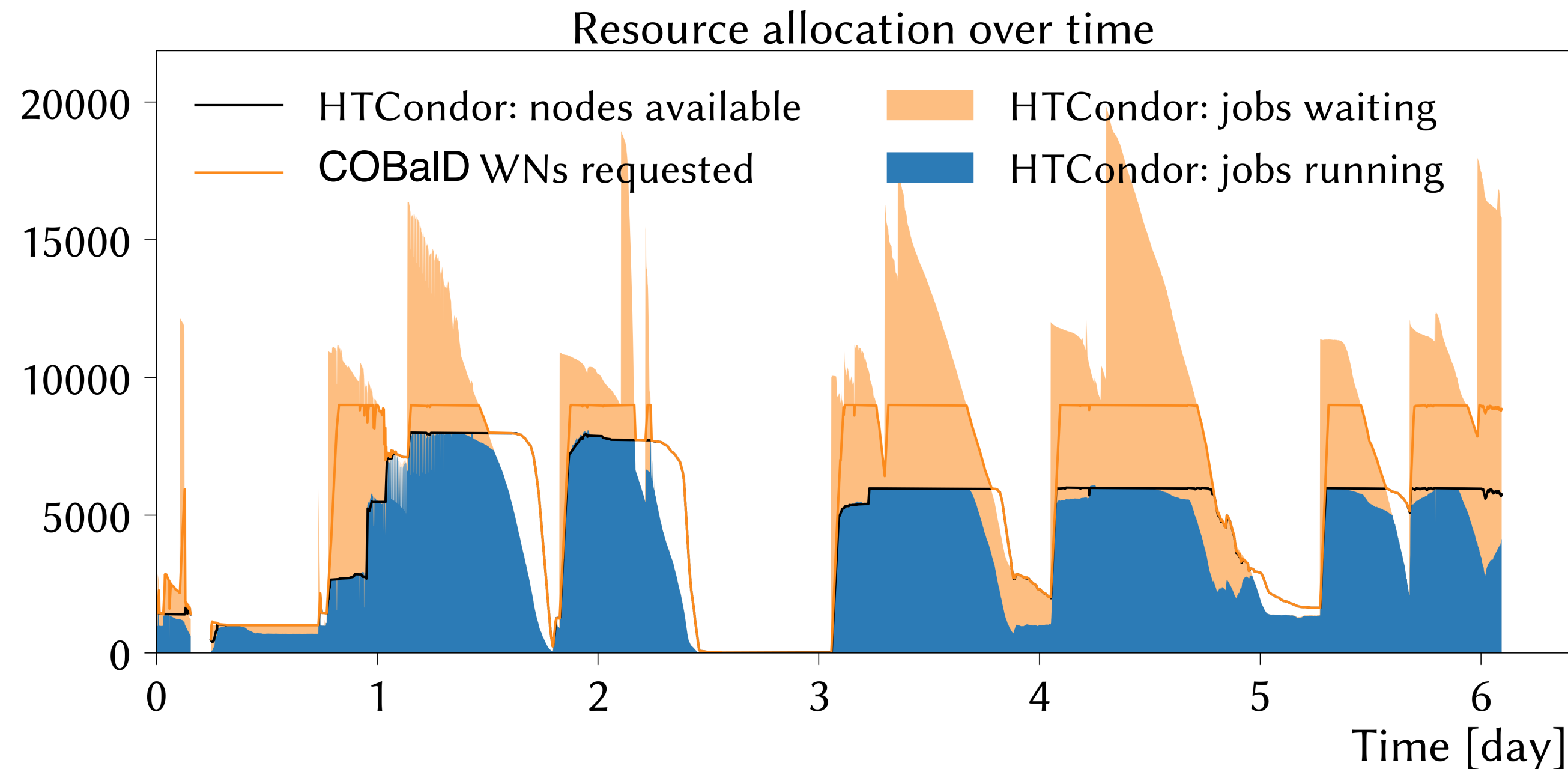
Based on a slide by Max Fischer

The COBaID View of Resource Meta-Scheduling

[COBaID - the Opportunistic Balancing Daemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and a mixture of job classes (e.g. CPU intense, I/O intense, ...)
- However: We usually care only about a simpler problem!

Fails in heterogeneous environments (e.g. HPCs & Clouds).



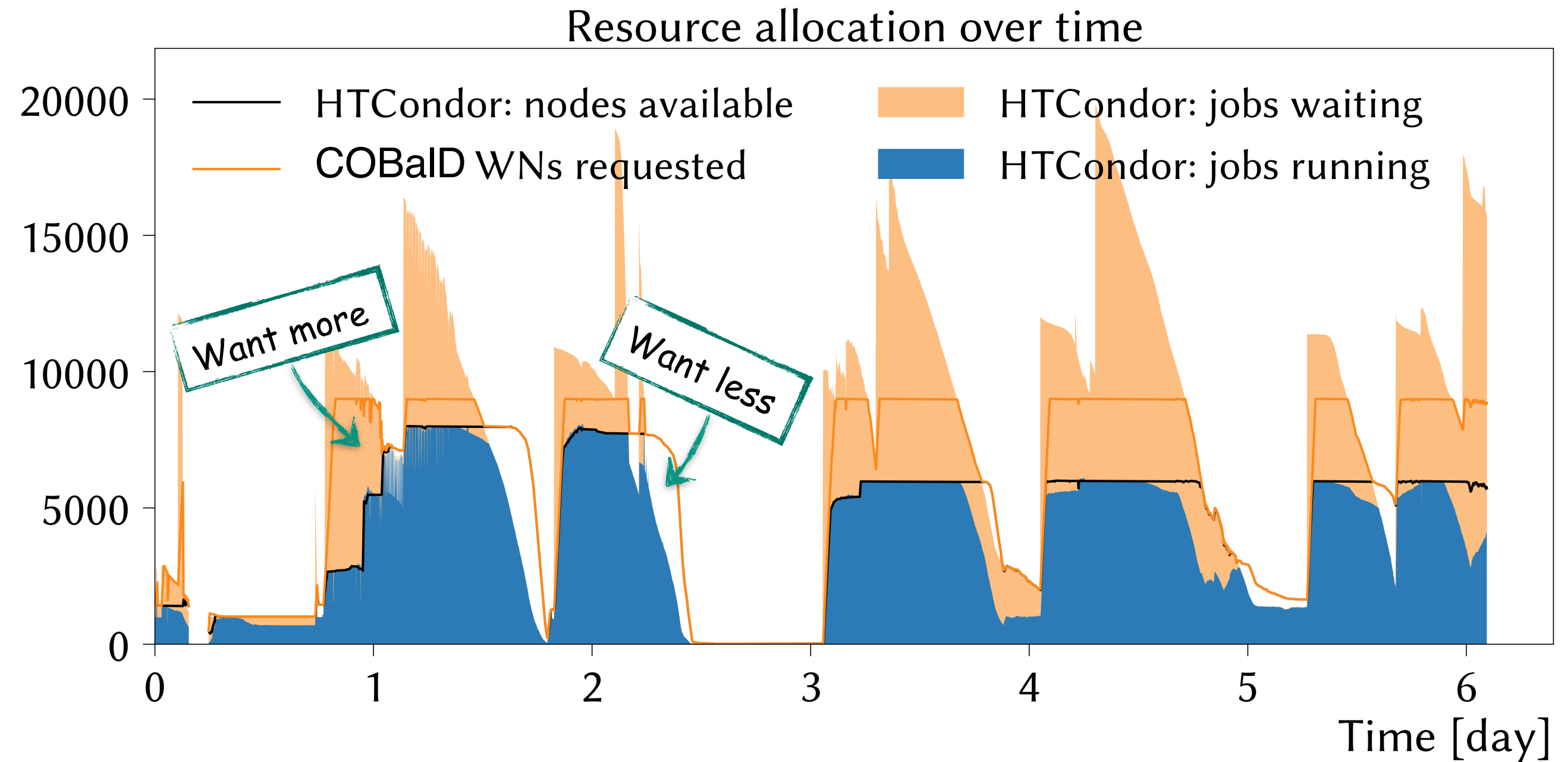
Based on a slide by Max Fischer

The COBaID View of Resource Meta-Scheduling

[COBaID - the Opportunistic Balancing Daemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and a mixture of job classes (e.g. CPU intense, I/O intense, ...)
- However: We usually care only about a simpler problem!

Fails in heterogeneous environments (e.g. HPCs & Clouds).

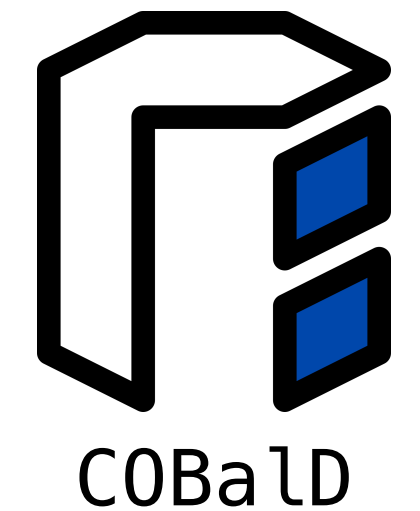


Based on a slide by Max Fischer

The COBaID View of Resource Scheduling

[COBaID - the Opportunistic Balancing Daemon]

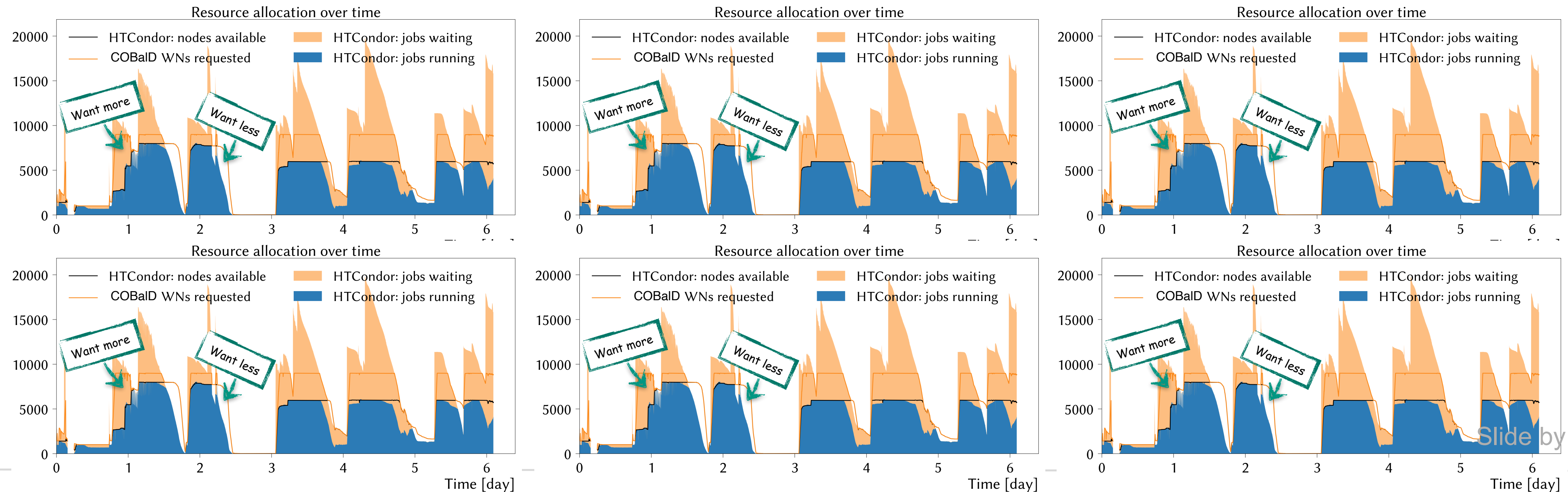
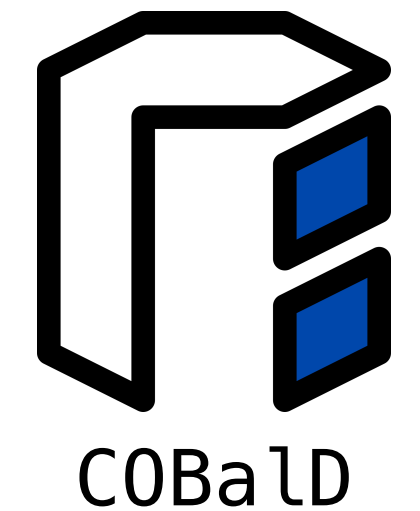
- Resource Meta-Scheduling for Job Scheduler is a „hard“
- COBaID cares only about resources, not jobs
 - Observe how much and how well each resource is used
 - Increase well-utilized resources, reduce poor utilized resources



The COBaID View of Resource Scheduling

[COBaID - the Opportunistic Balancing Daemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“
- COBaID cares only about resources, not jobs
 - Observe how much and how well each resource is used
 - Increase well-utilized resources, reduce poor utilized resources

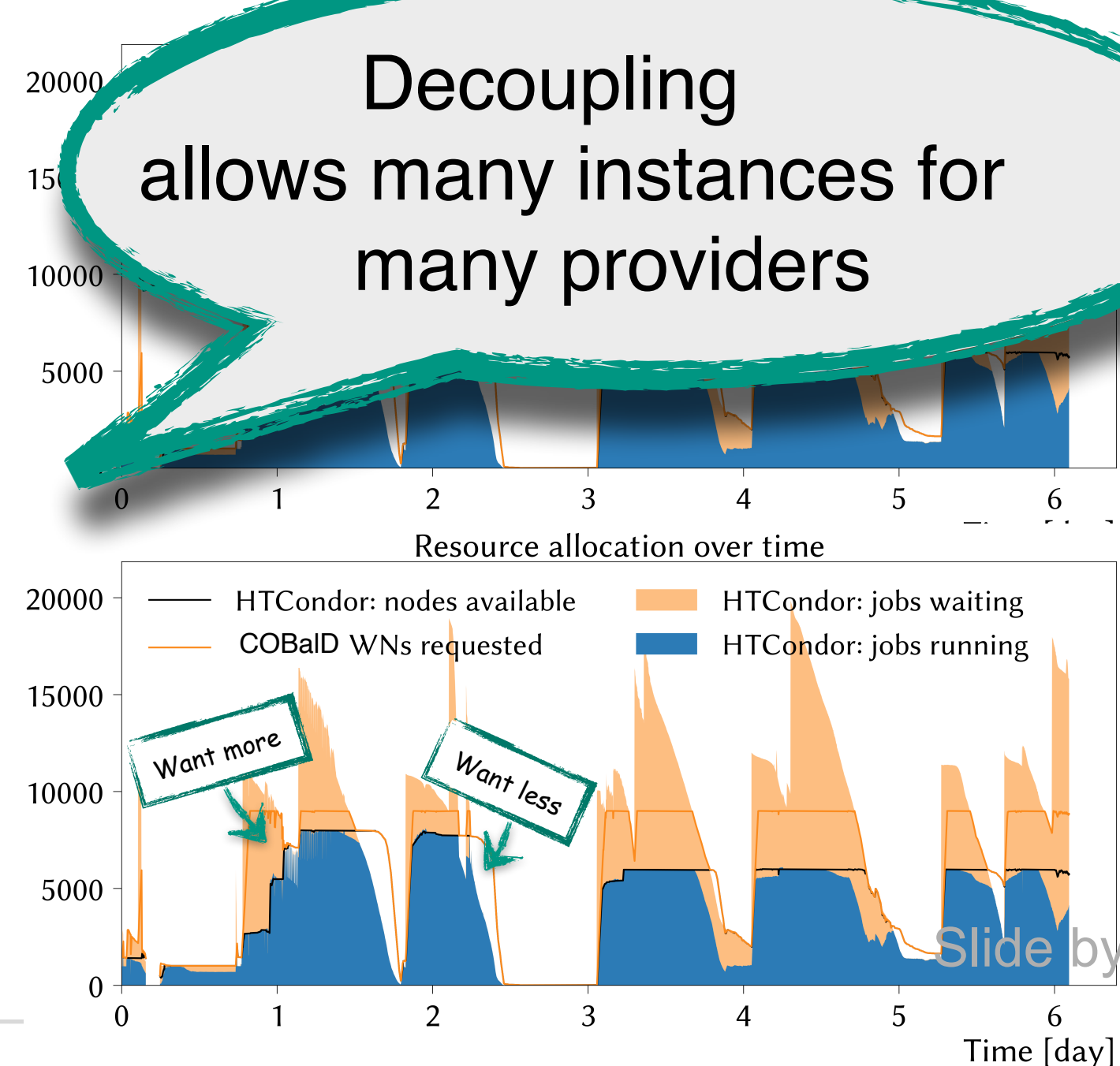
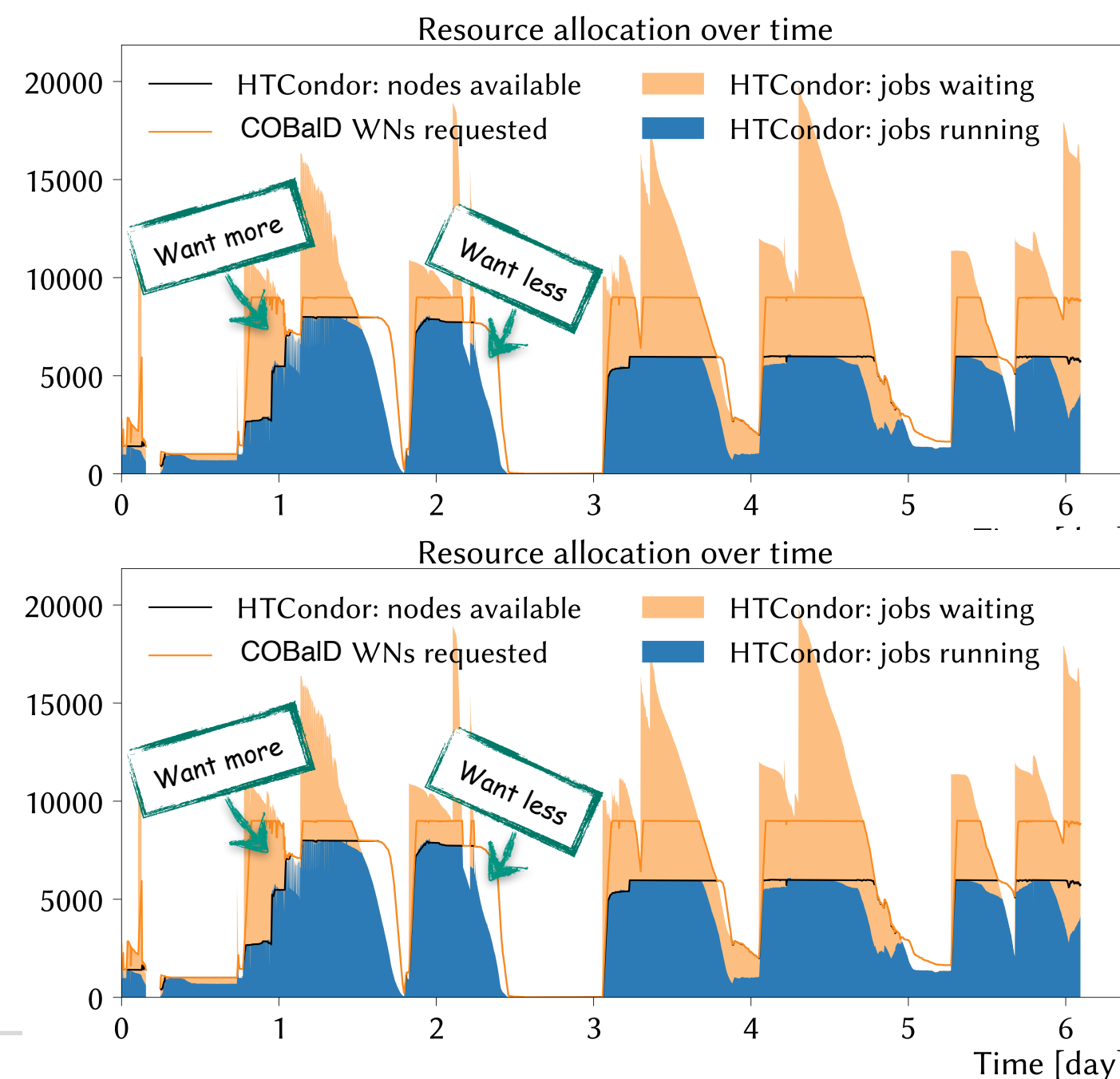
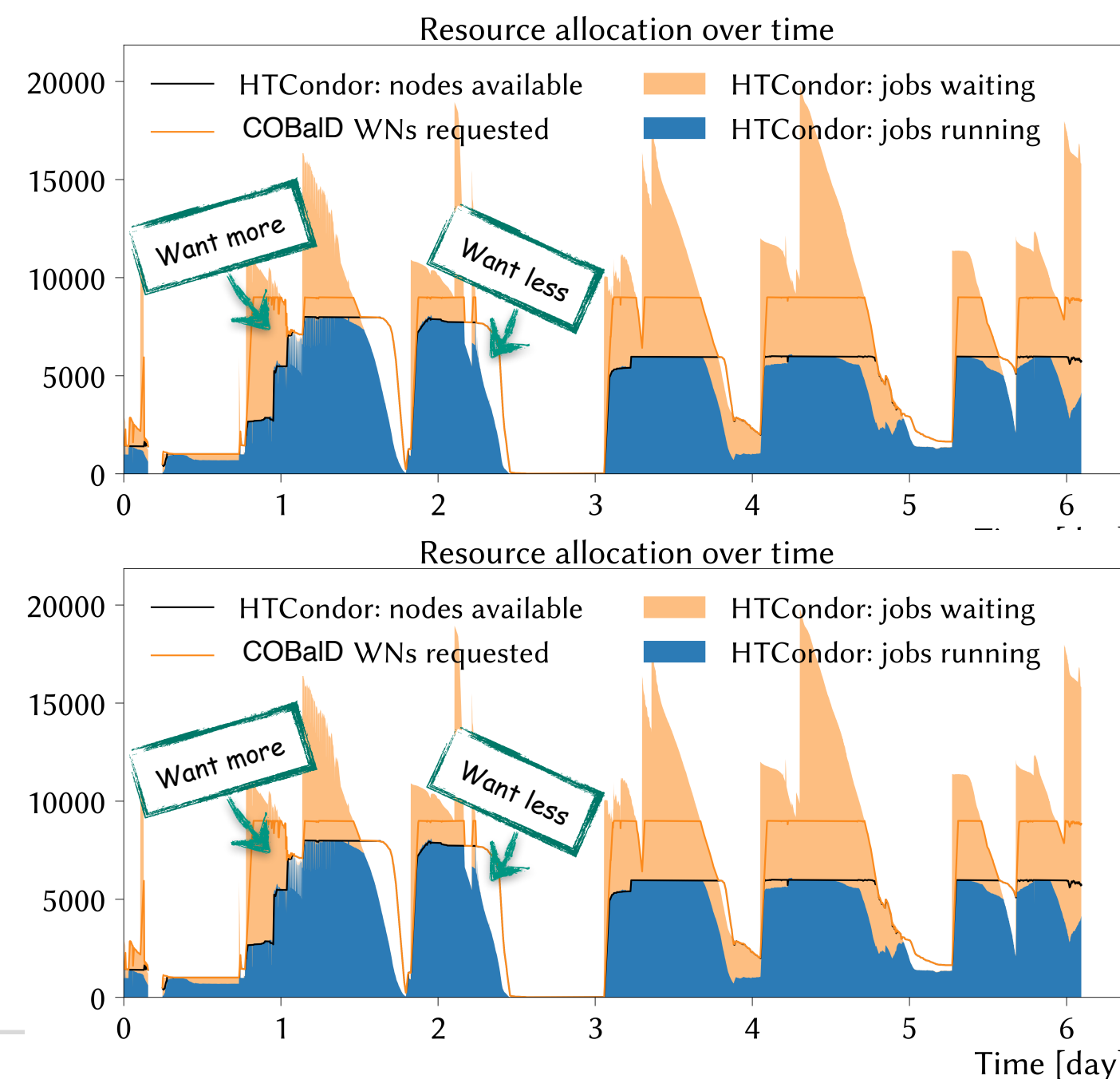
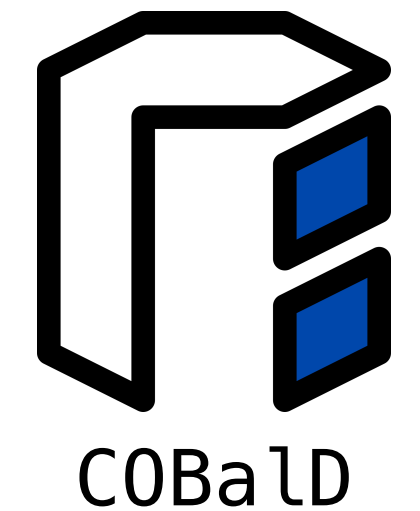


Slide by Max Fischer

The COBaID View of Resource Scheduling

[COBaID - the Opportunistic Balancing Daemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard“
- COBaID cares only about resources, not jobs
 - Observe how much and how well each resource is used
 - Increase well-utilized resources, reduce poor utilized resources



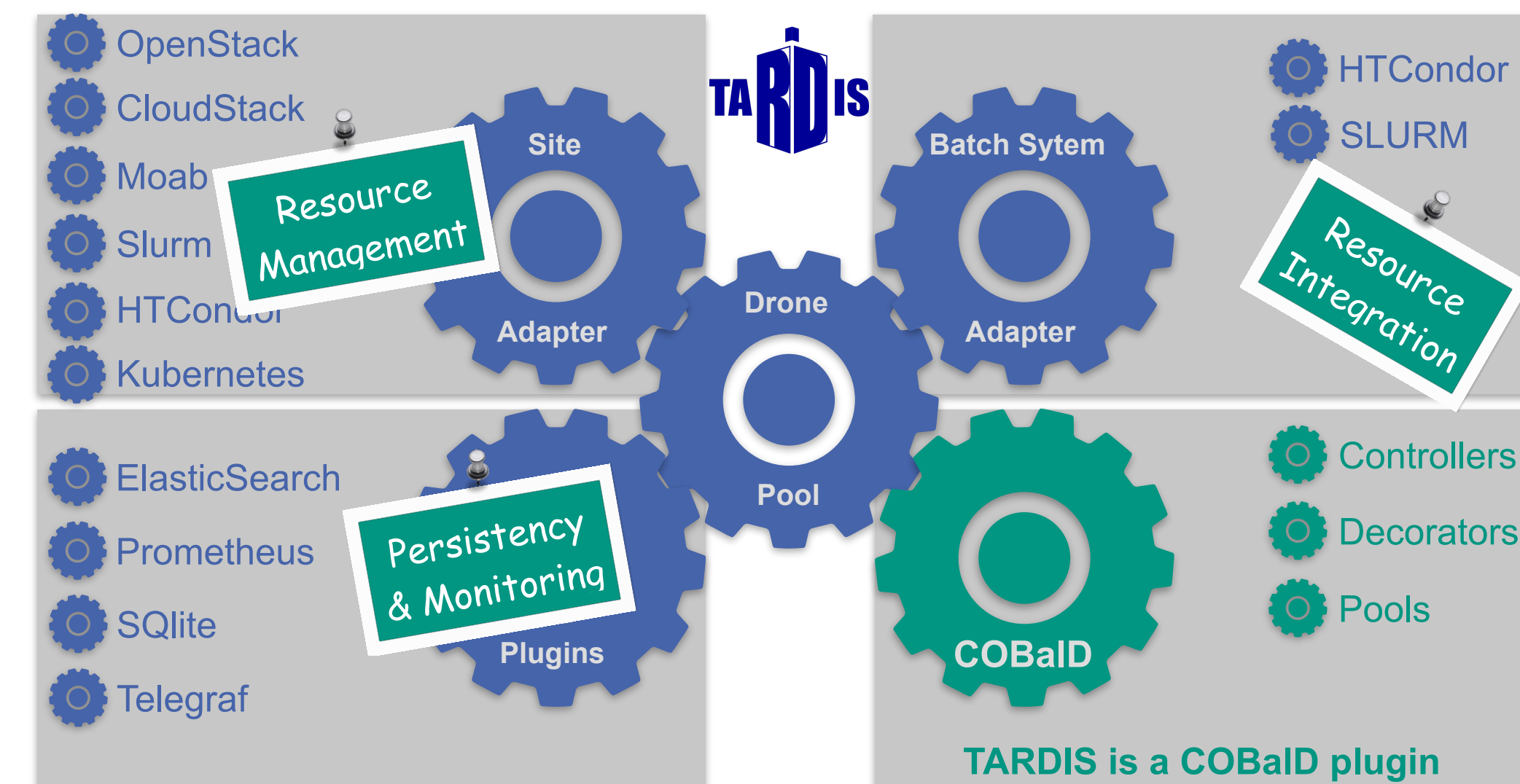
Slide by Max Fischer

TARDIS - Out-of-the-Box Resource Adapters

[Transparent Adaptive Resource Dynamic Integration System]



- Combine resource provider APIs with COBaID
 - Request, monitor, decommission individual resources (Manages resource life cycle)
 - Automatically matches resource demand via COBaID approach
 - Basically a „use-case agnostic autonomous Pilot factory“
- Support for common HPC batch systems, Cloud APIs,
 - ...
 - Behave like „regular users“ as much as possible
 - Customizable pilot for each centre’s peculiarities
 - HEP: Insert HTCondor+CVMFS as available



The Entire COBaID/TARDIS Ecosystem

container-stacks [↗](#)

Container images to provide dedicated job environments

Available containers [↗](#)

Container	Environment provided
wlwg-wn	Provides a standard environment to run all jobs of VOs supported by WLCG
htcondor-wn	Provides a standard htcondor enabled workernode configurable using ansible

 **cobald** Public

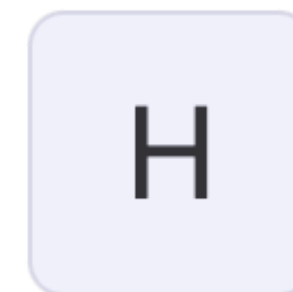
Cobald is an Opportunistic Balancing Daemon

● Python ☆ 11 🍴 9

 **tardis** Public

Transparent Adaptive Resource Dynamic Integration System

● Python ☆ 15 🍴 17



HTCondor_configs 🔒

Project ID: 3523 [📄](#) [Leave project](#)



🔗 236 Commits 🍴 3 Branches 📄 0 Tags 📦 278 KiB Project Storage

HTCondor configs for each site

condor-git-config 0.1.5

```
pip install condor-git-config
```

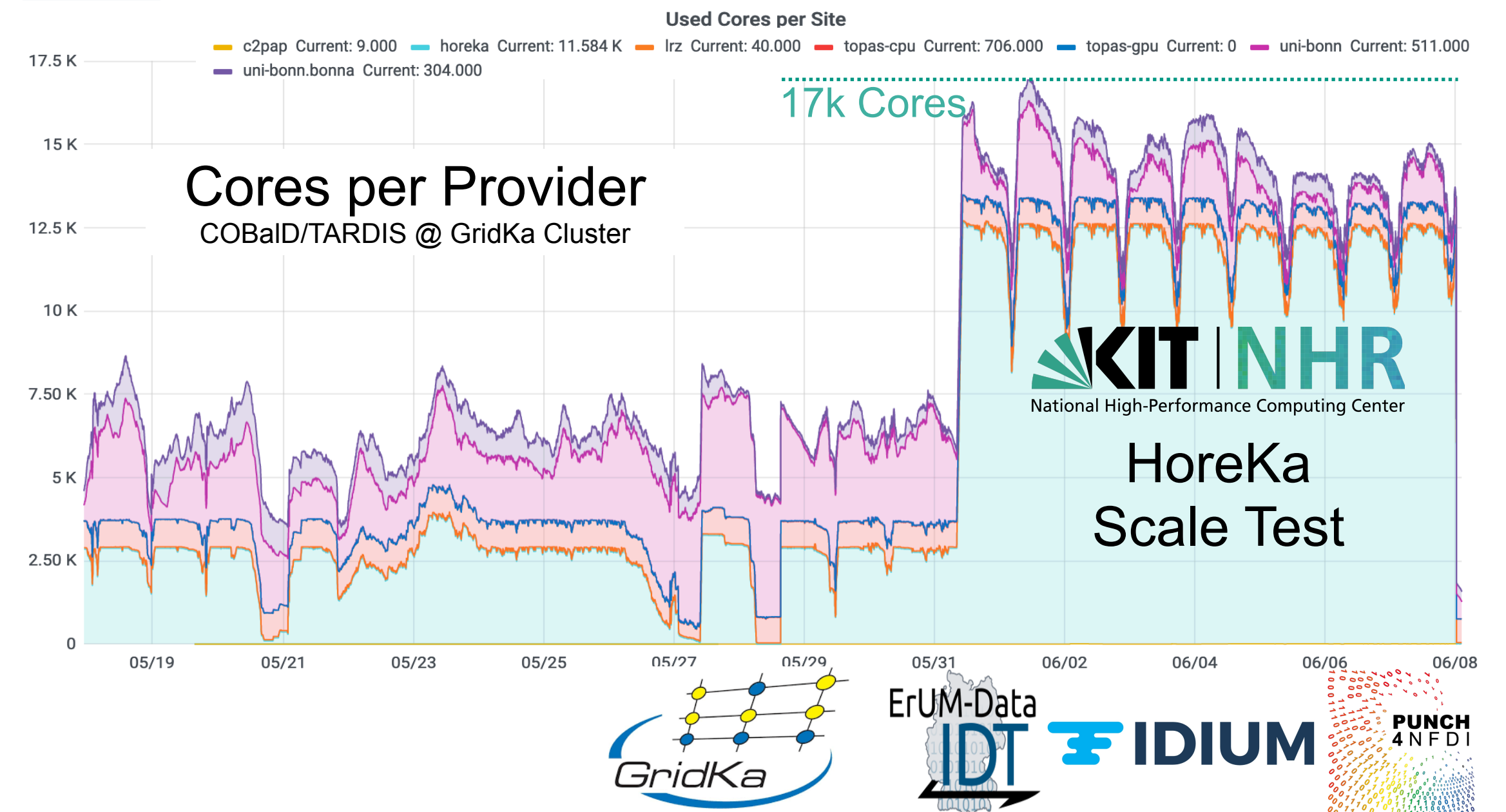
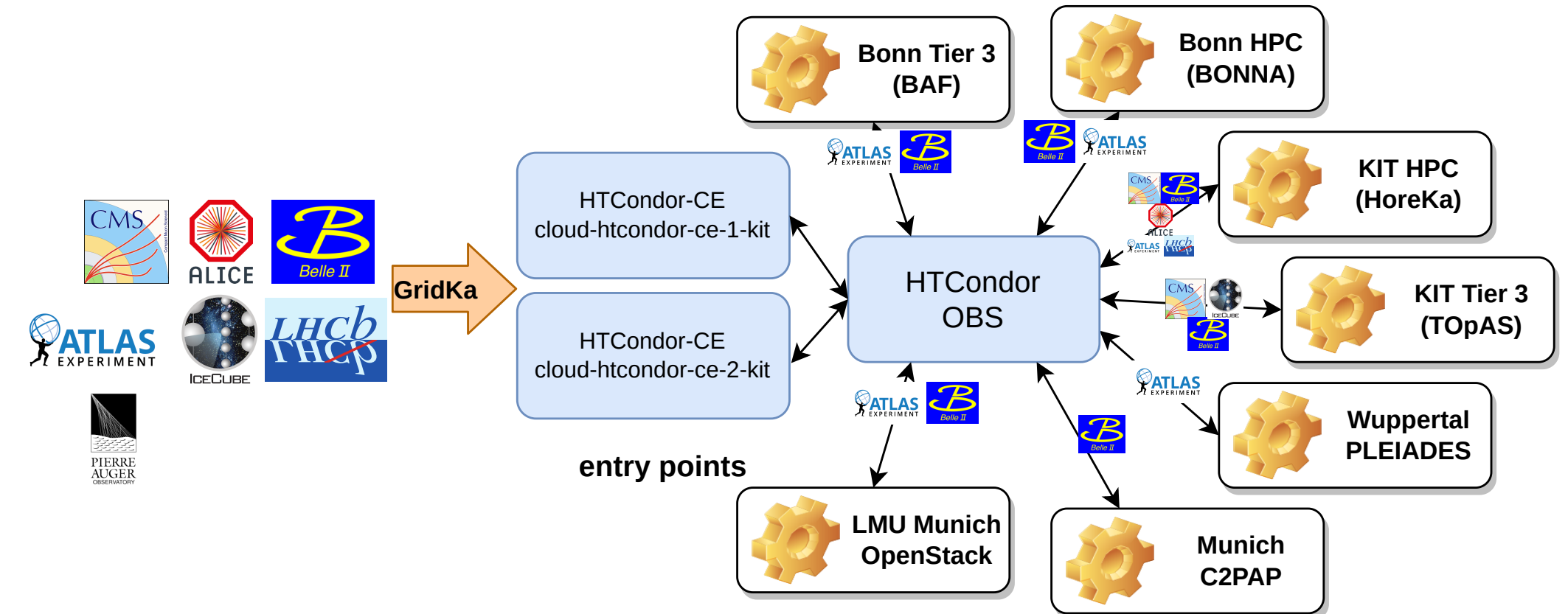
Please, re-use and improve existing toolset,
if you plan to integrate your resources!

Use-cases so far ...

Opportunistic Resources & WLCG in Practice

Simplify provisioning and utilization of third-party compute resources for the GridKa communities:

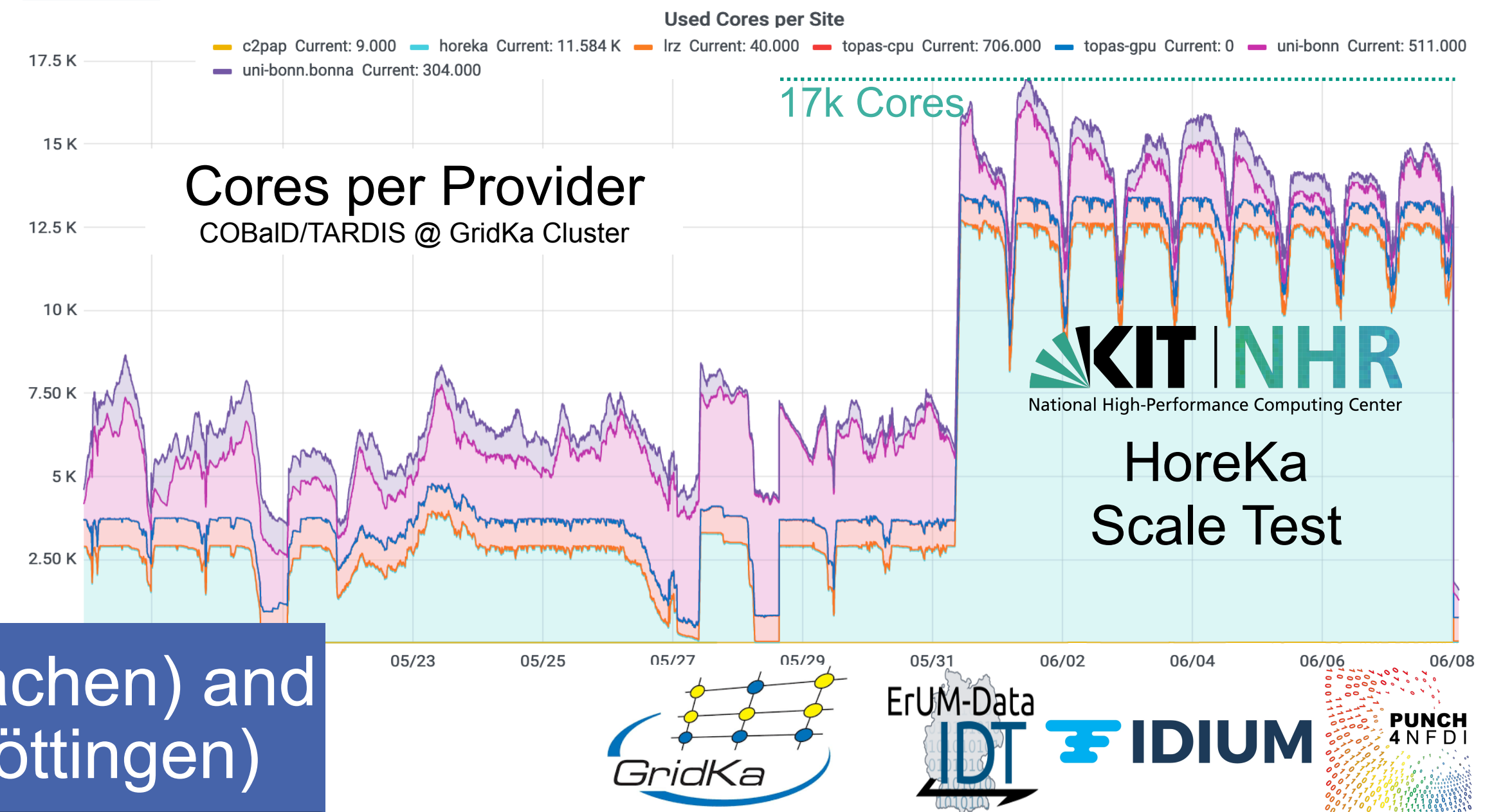
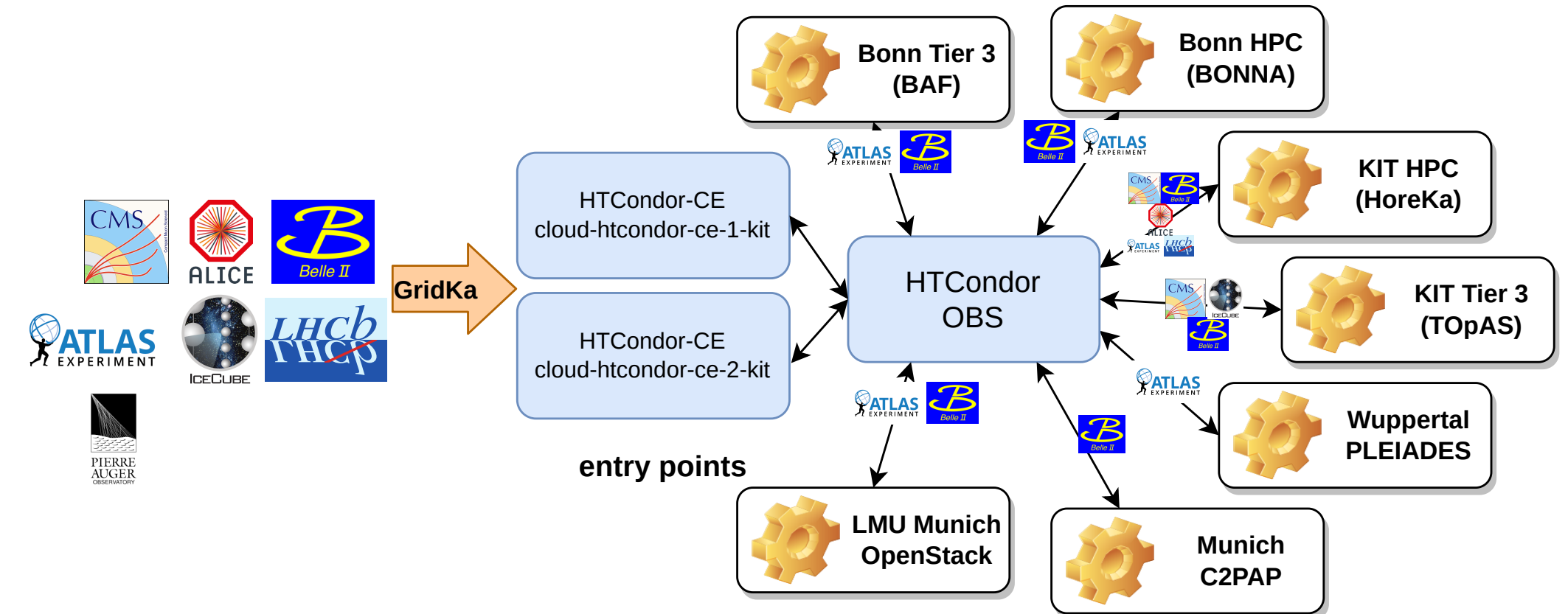
- Dynamic, transparent and on-demand integration via COBaID/TARDIS (in-house development)
- Provide community-overarching unified entry points to a variety of resources (HPCs, Clouds, ...)
- Demonstrated production scale operation during scale test together with HoreKa (KIT HPC cluster)
- Production deployments across HEP institutes & HPC resources coordinated by KIT/GridKa
- Site specific accounting is now also possible with AUDITOR (see talk today)



Opportunistic Resources & WLCG in Practice

Simplify provisioning and utilization of third-party compute resources for the GridKa communities:

- Dynamic, transparent and on-demand integration via COBaID/TARDIS (in-house development)
- Provide community-overarching unified entry points to a variety of resources (HPCs, Clouds, ...)
- Demonstrated production scale operation during scale test together with HoreKa (KIT HPC cluster)
- Production deployments across HEP institutes & HPC resources coordinated by KIT/GridKa
- Site specific accounting is now also possible with AUDITOR (see talk today)

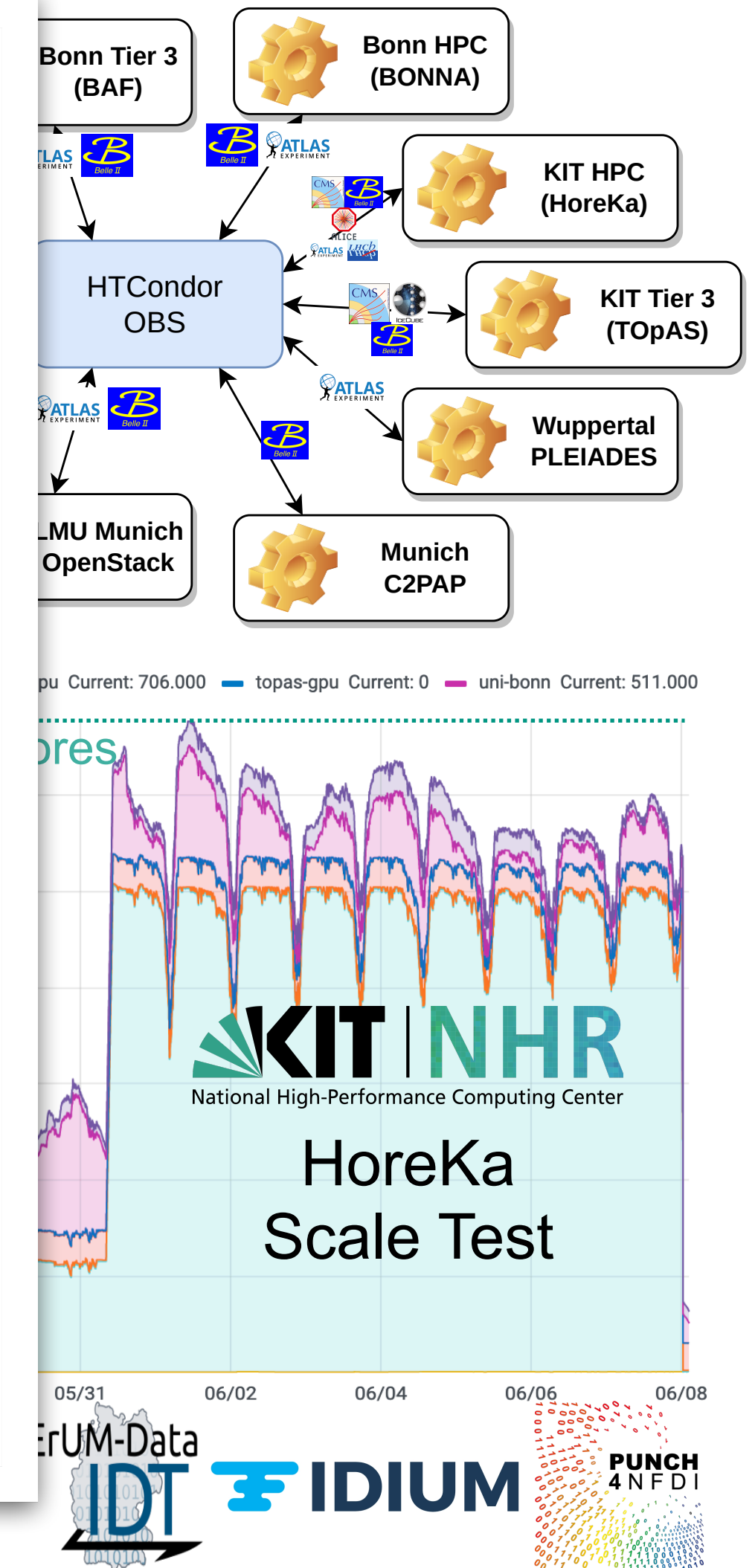
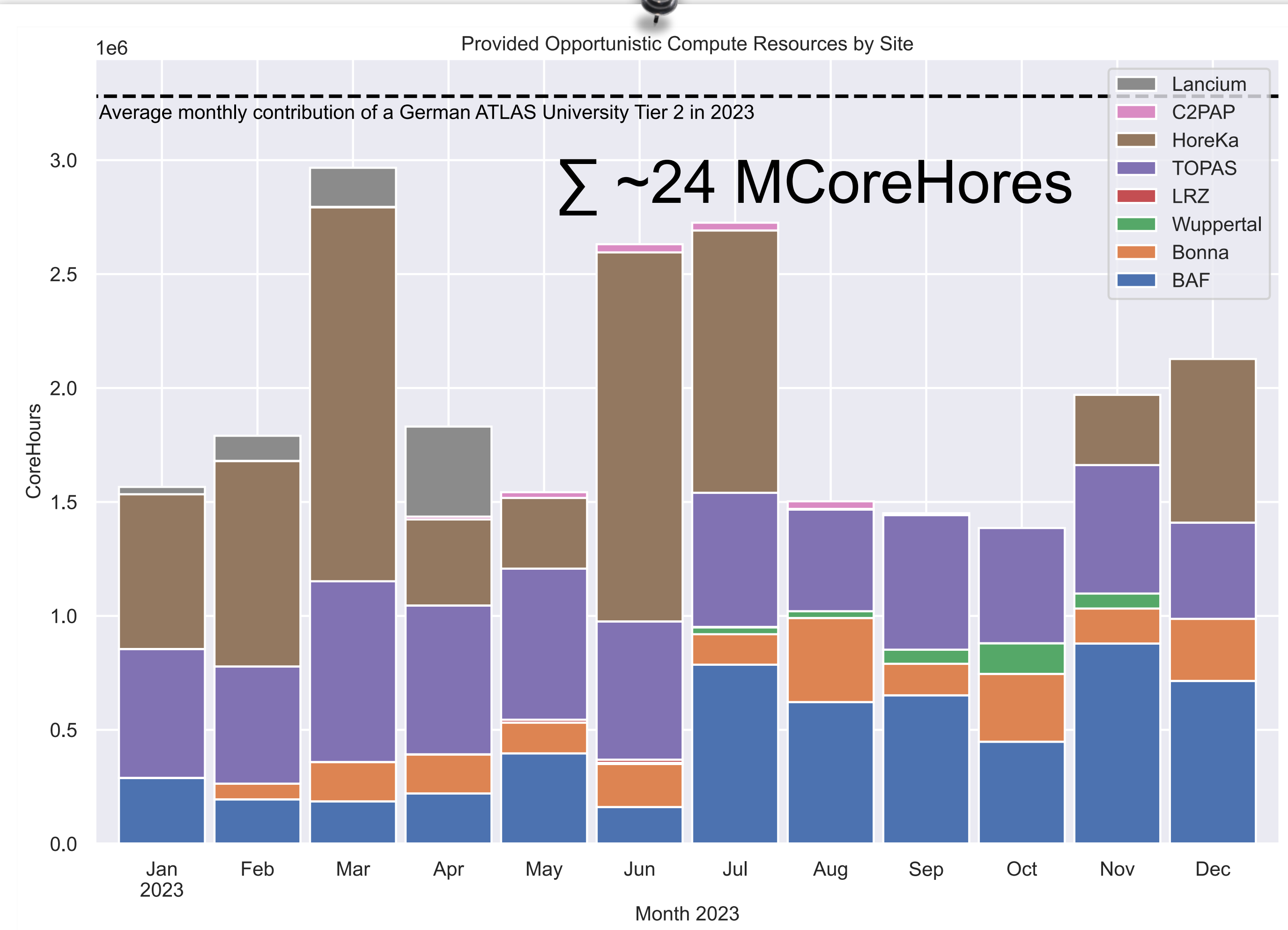


Similar setup deployed at CLAIX HPC (RWTH Aachen) and on-going deployment at Emmy (University of Göttingen)

Opportunistic Resources & WLCG in Practice

Simplify provisioning compute resources for

- Dynamic, transparent via COBaID/TARD
- Provide community points to a variety
- Demonstrated production scale test together
- Production deployment HPC resources
- Site specific accounts AUDITOR (see table)



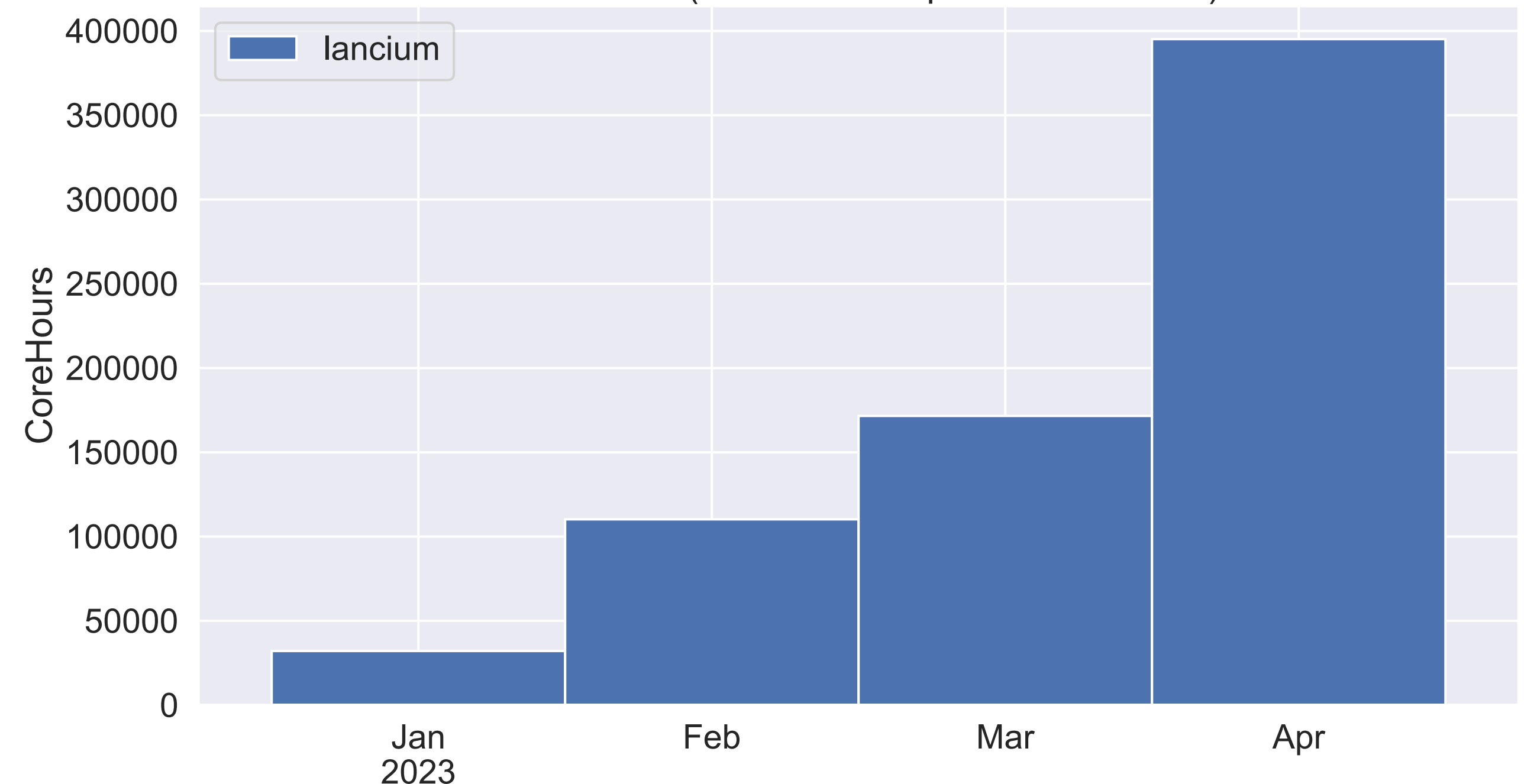
Similar setup deployment on-going deployment at Emmy (University of Göttingen)



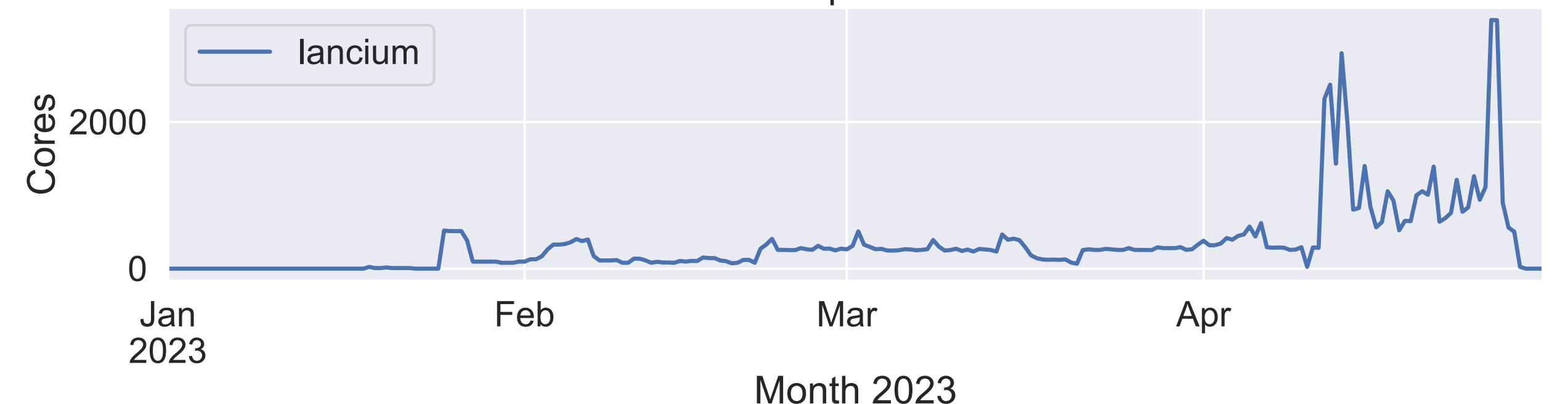
Enabling Access to Sustainable Compute Resources

- Lancium (US company) balancing the power grid by operating compute facilities close to renewables (wind & solar) - CO₂ neutral operation
- Dynamic, transparent and on-demand integration via COBaID/TARDIS
- Used for ATLAS/CMS MC generation (~700,000 CoreHours during PoC)
- Very smooth „Proof of Concept“ project, experiments did not even noticed that the jobs ran in the US
- Unfortunately, Lancium decided to get out of the PaaS business in April 2023

CoreHours (Lancium Compute Contribution)

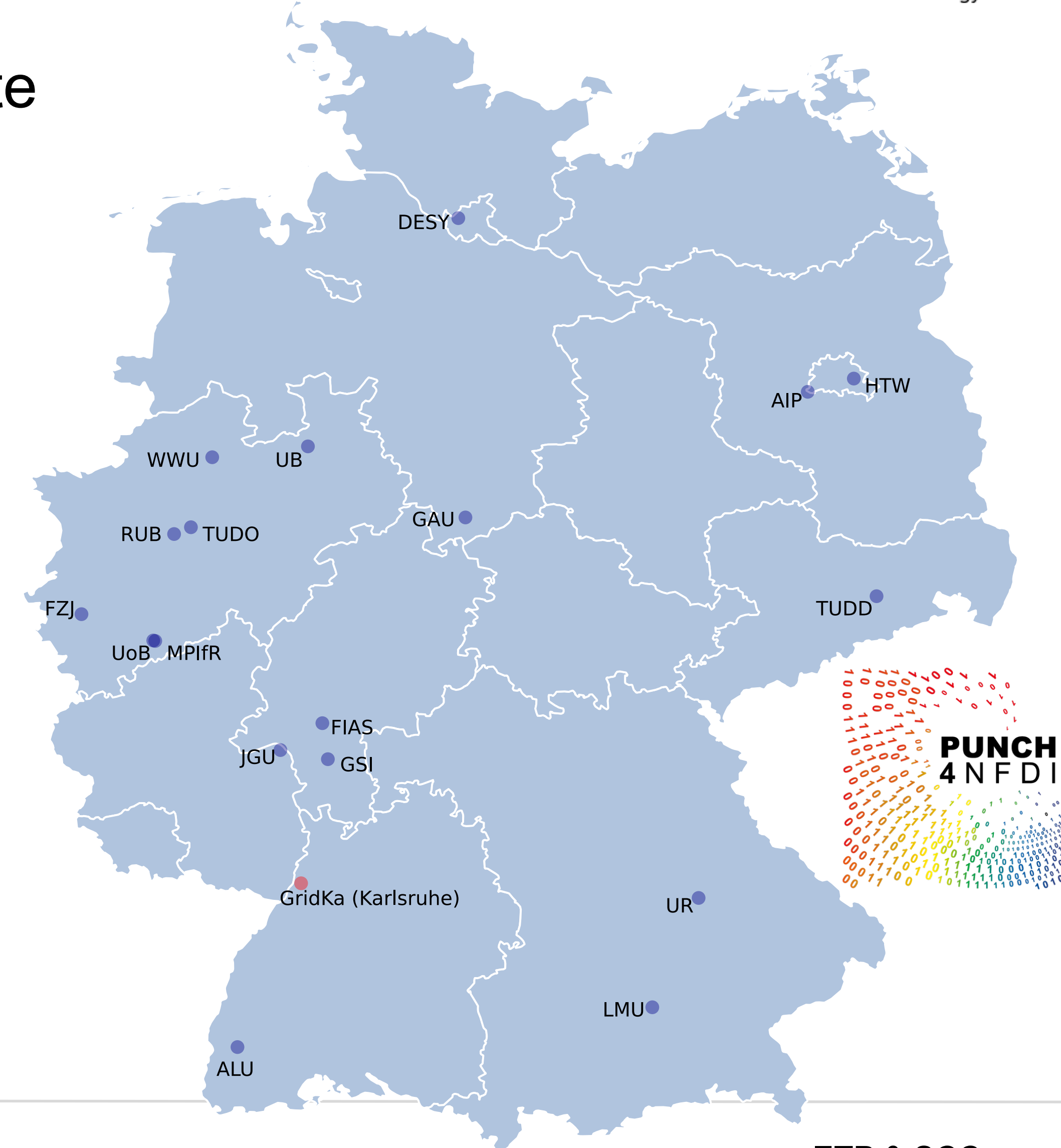


Lancium Compute Contribution



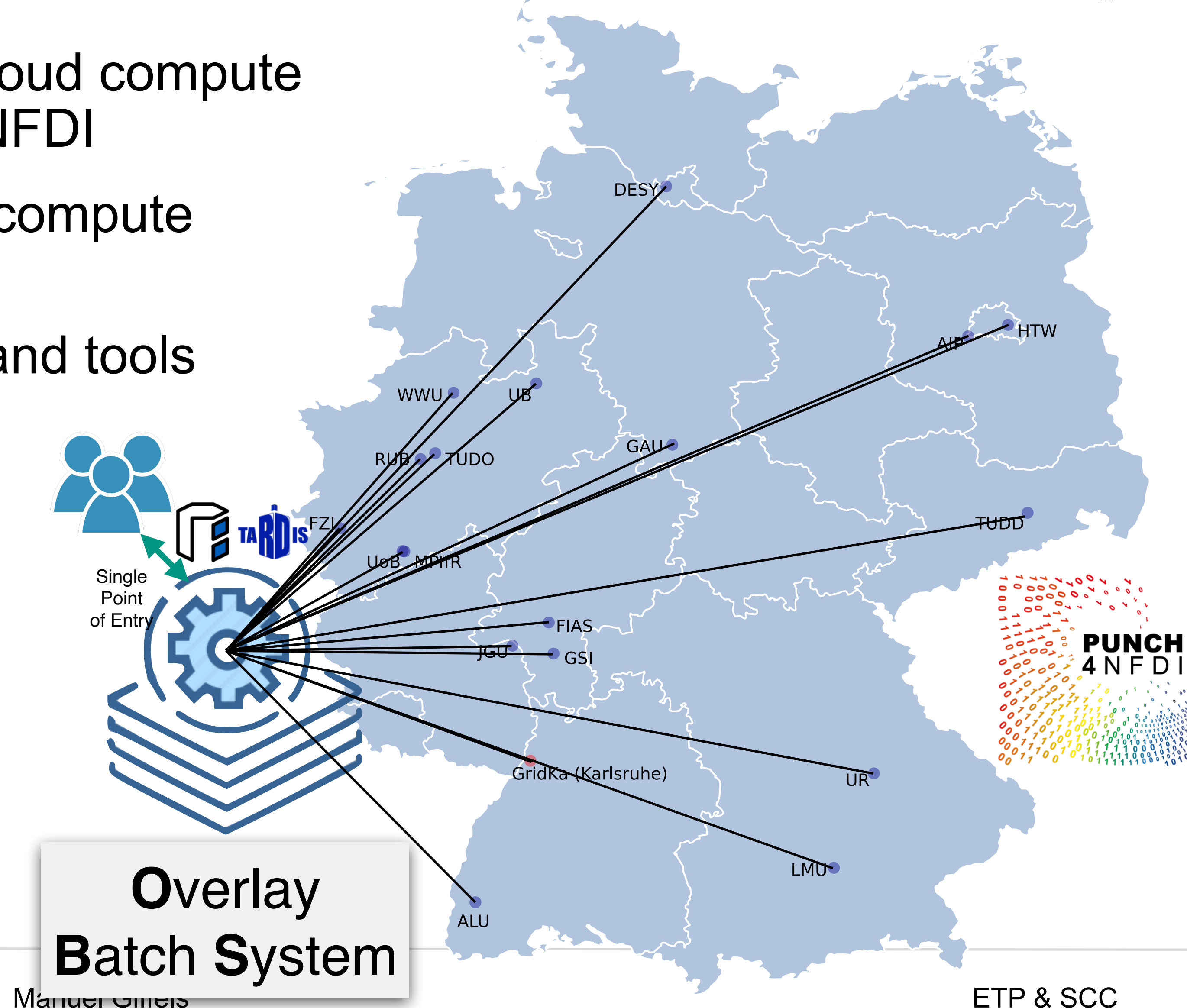
Towards the Compute4PUNCH Infrastructure

- Substantial amount of HTC, HPC, Cloud compute resources are provided to PUNCH4NFDI



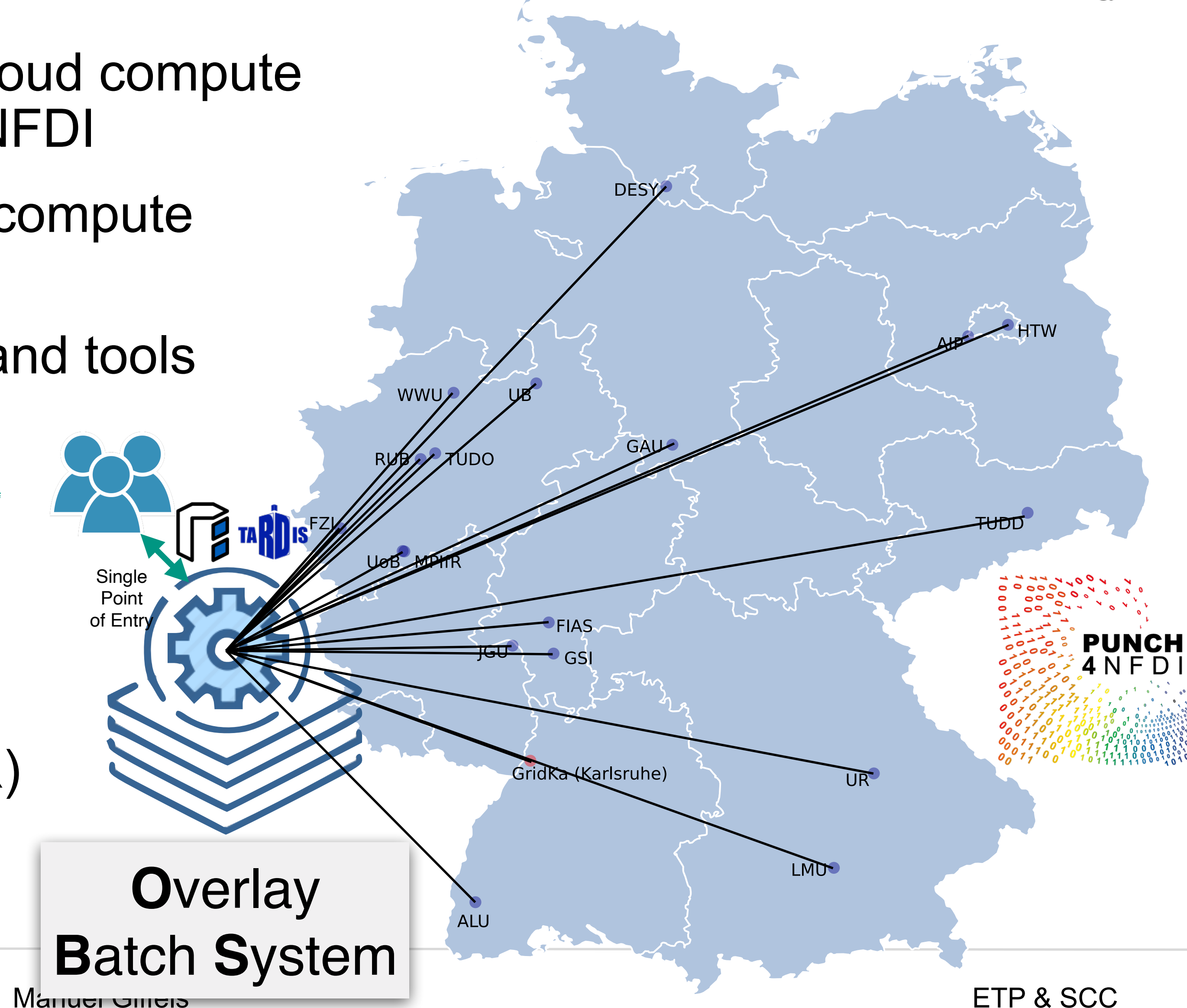
Towards the Compute4PUNCH Infrastructure

- Substantial amount of HTC, HPC, Cloud compute resources are provided to PUNCH4NFDI
- Establish a federated heterogeneous compute infrastructure for PUNCH4NFDI
- Benefit from experiences, concepts and tools available in HEP community



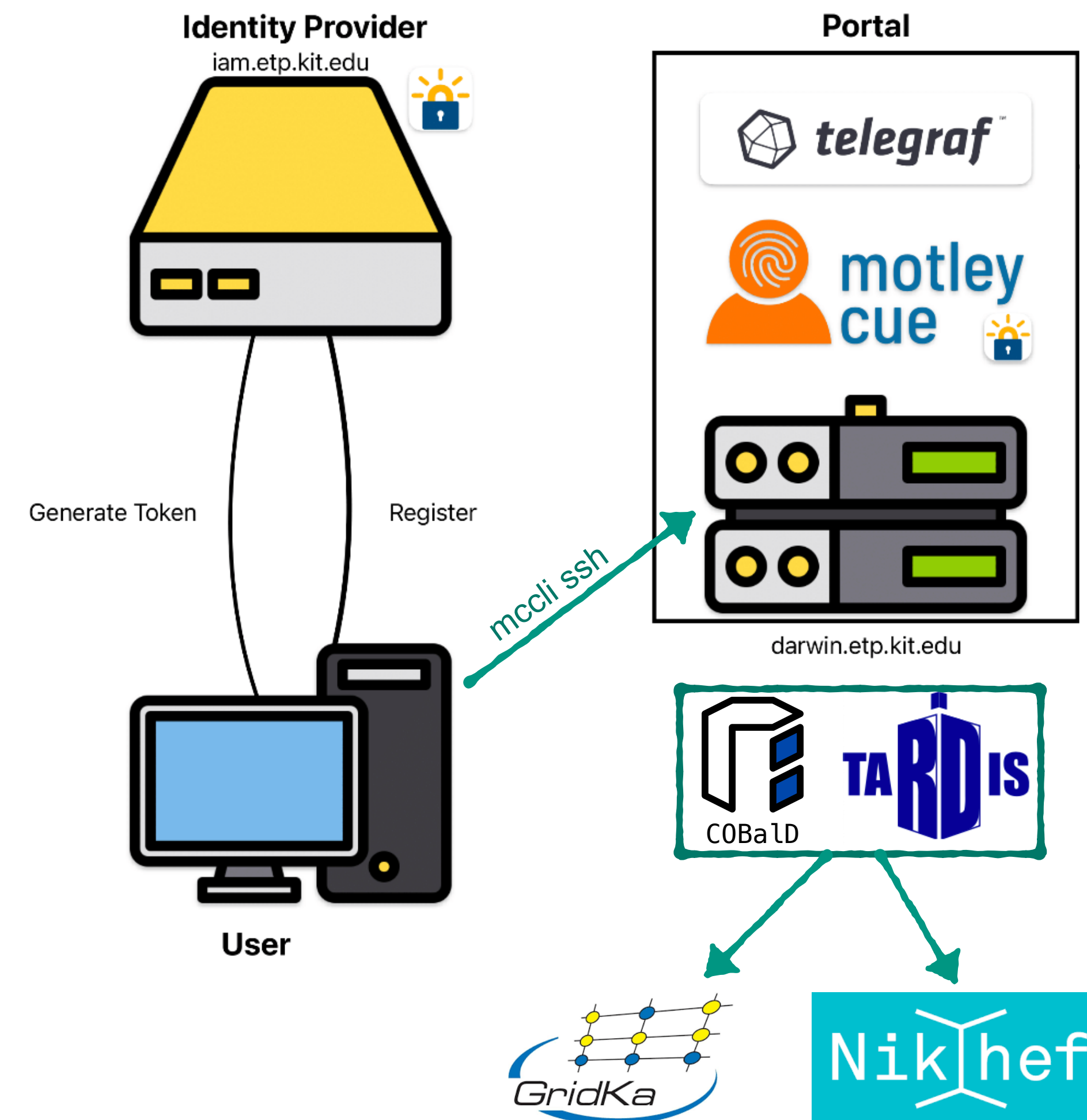
Towards the Compute4PUNCH Infrastructure

- Substantial amount of HTC, HPC, Cloud compute resources are provided to PUNCH4NFDI
 - Establish a federated heterogeneous compute infrastructure for PUNCH4NFDI
 - Benefit from experiences, concepts and tools available in HEP community
-
- Compute4PUNCH demonstrator is available
 - Demonstration workflows of HEP (ATLAS/CMS), Astrophysics (LOFAR) and Lattice QCD have been successfully performed



Building a Computing Infrastructure for DARWIN

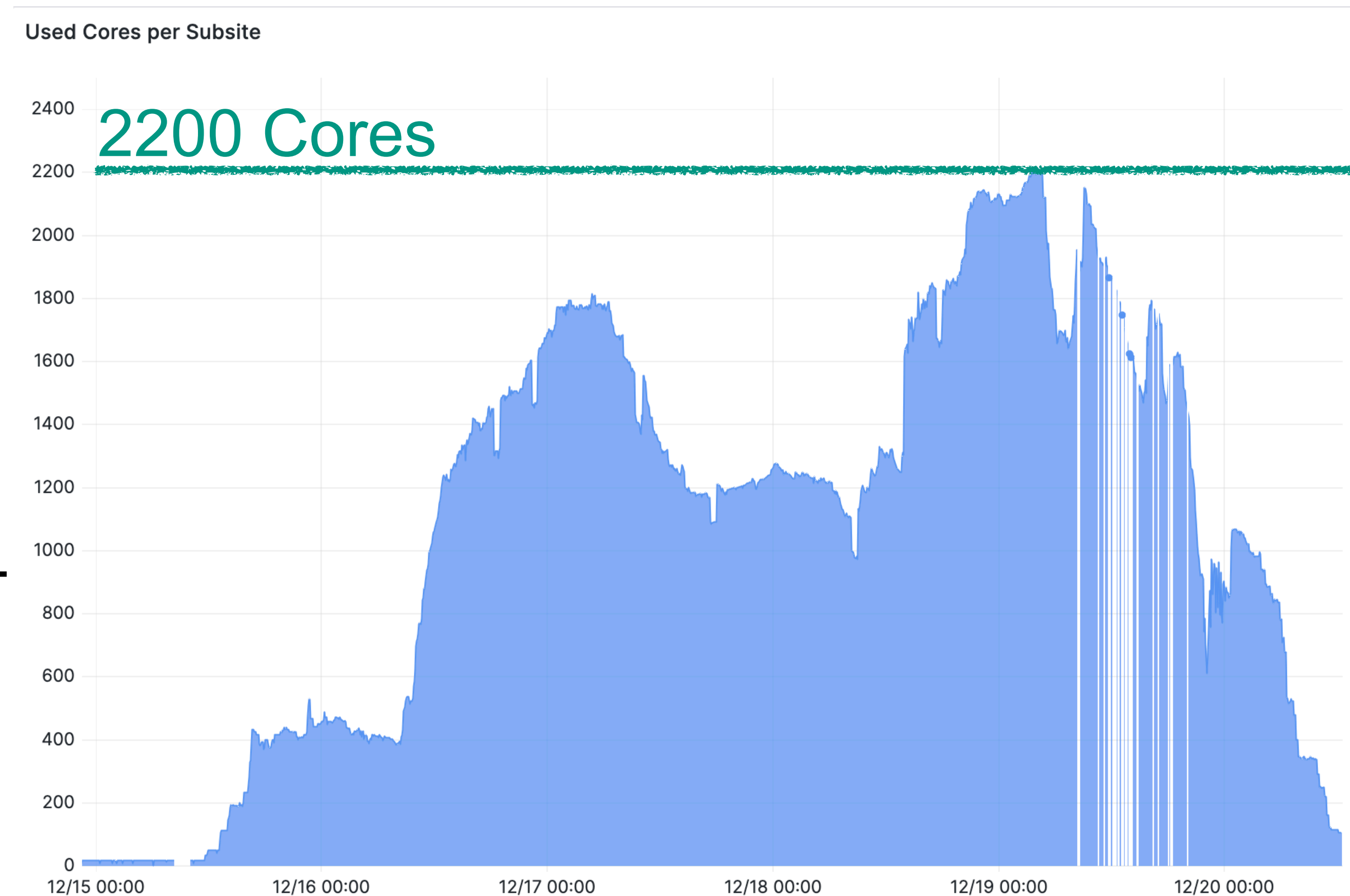
- Manage VO via Indigo Identity and access management service (IAM)
- Provide token based access via ssh to a login node (using Motley Cue)
- Provide dedicated JupyterHub for interactive data analysis
- Integrate external Grid resources using C/T as a pilot factory [GridKa & Nikhef (ongoing)]
- All resources available via an HTCondor OBS on the login node



work by Sebastian Brommer (KIT)

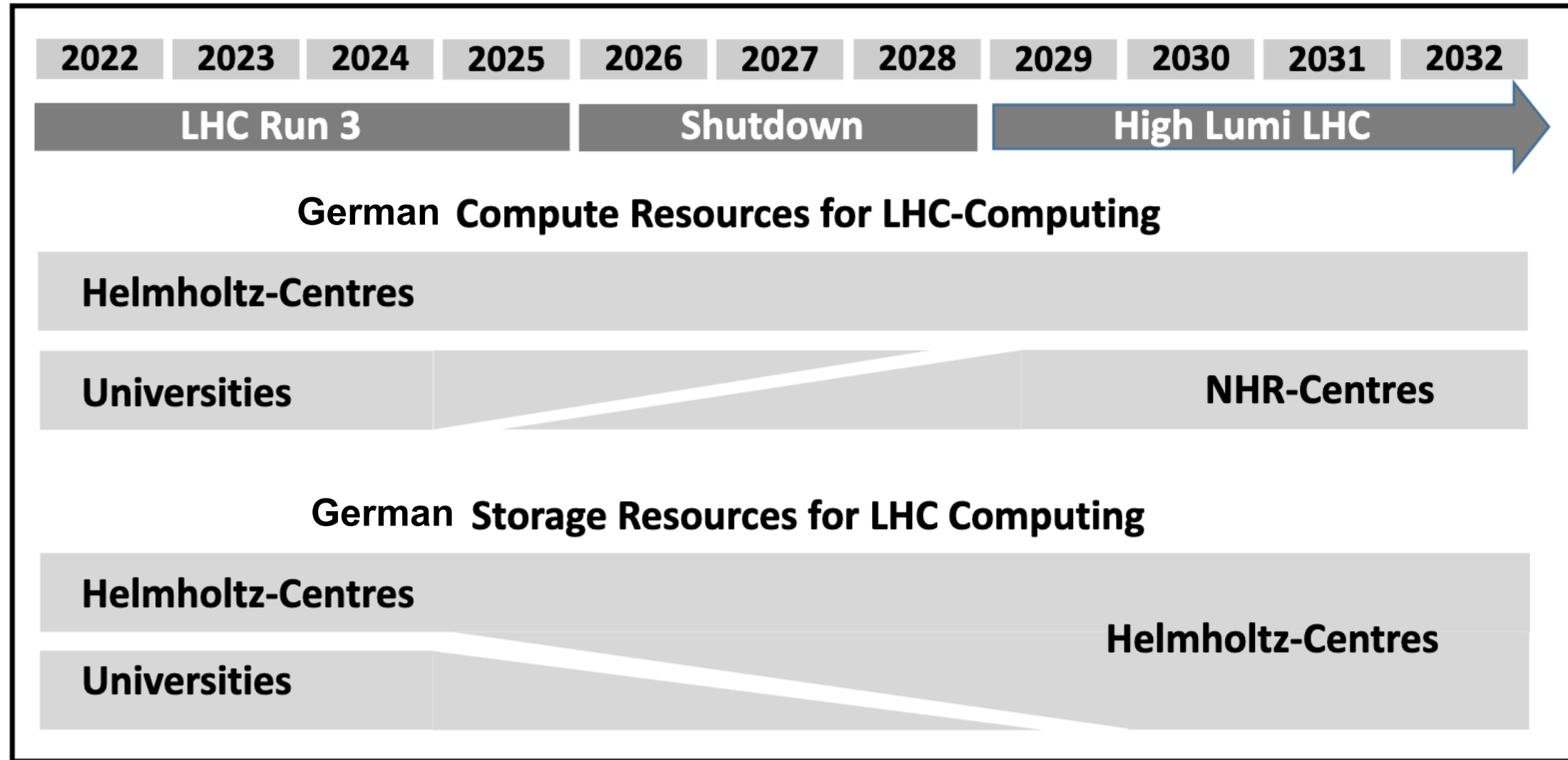
Remote Operation of the LMU Tier-2

- In December 2023 Rod contacted us about a nice PoC idea (12.12.2023)
- There was a week long scheduled storage downtime at the LMU Tier-2
- So, how about integrating the LMU Tier-2 workers into the opportunistic compute cloud operated at GridKa?
- Rod was able to quickly set-up the C/T ecosystem at LMU supported by KIT
- During the downtime the LMU Tier-2 was fed with ATLAS jobs via GridKa (incl. remote data access)

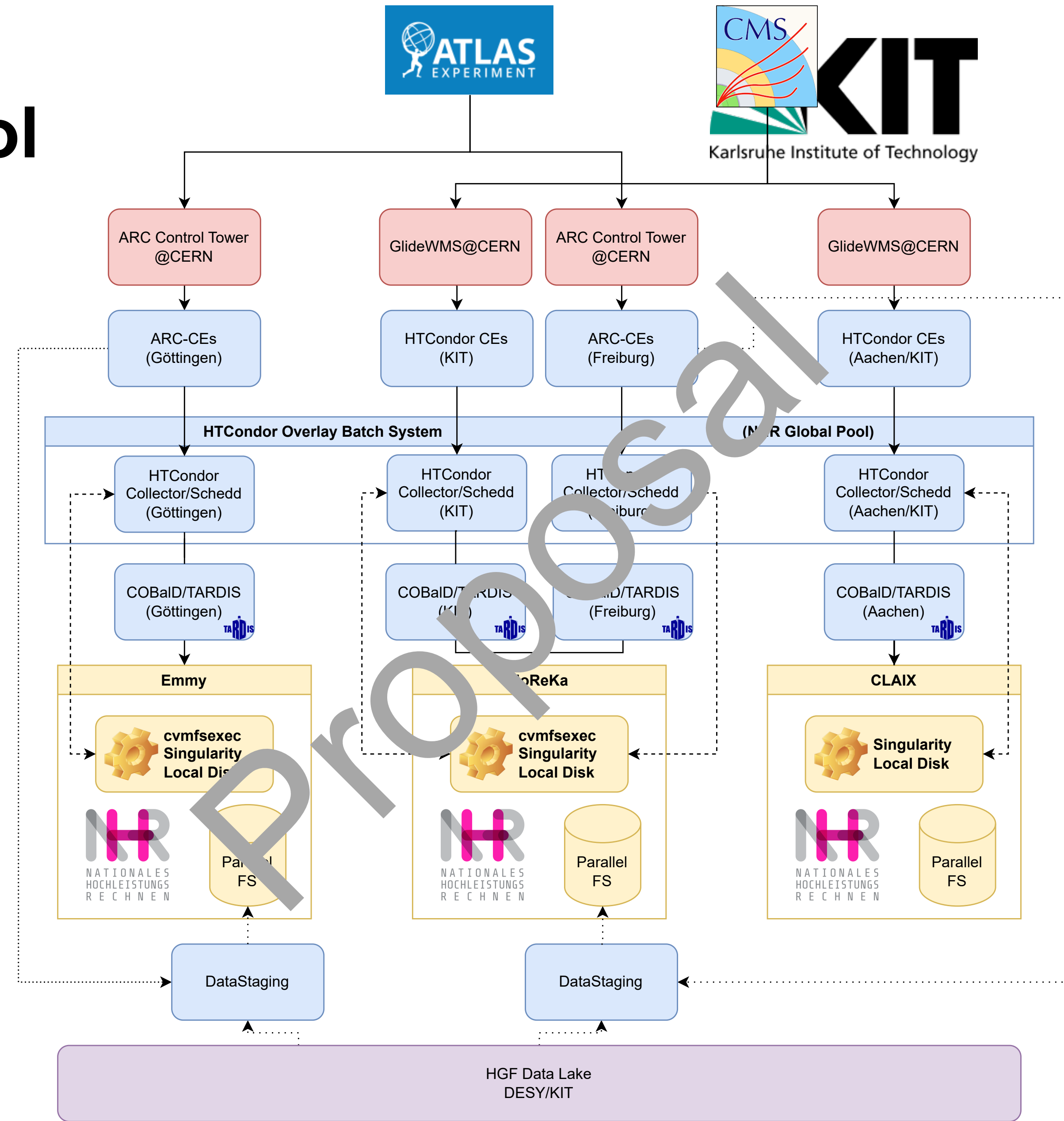


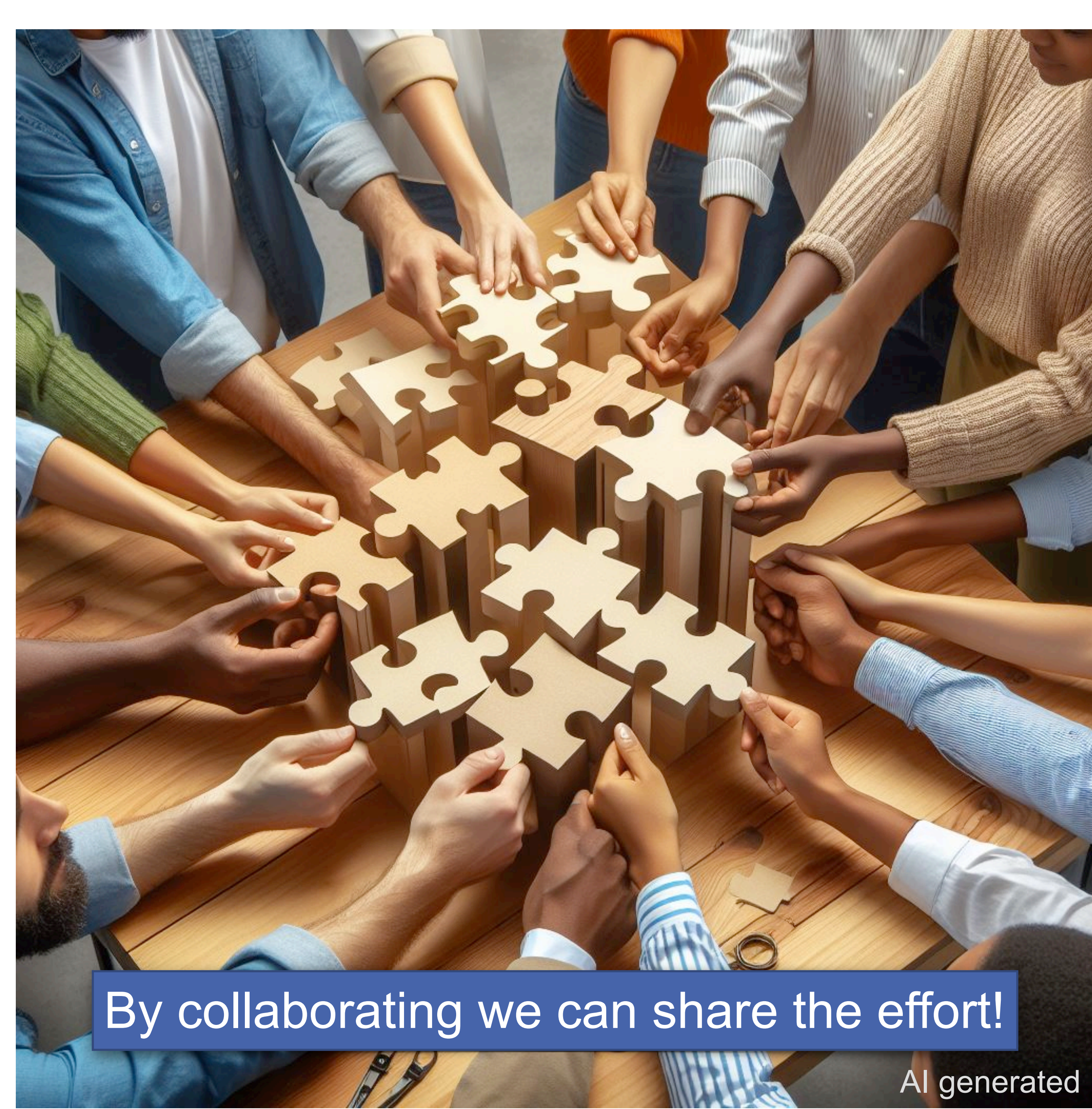
The future ...

Towards NHR German Tier-2 Pool



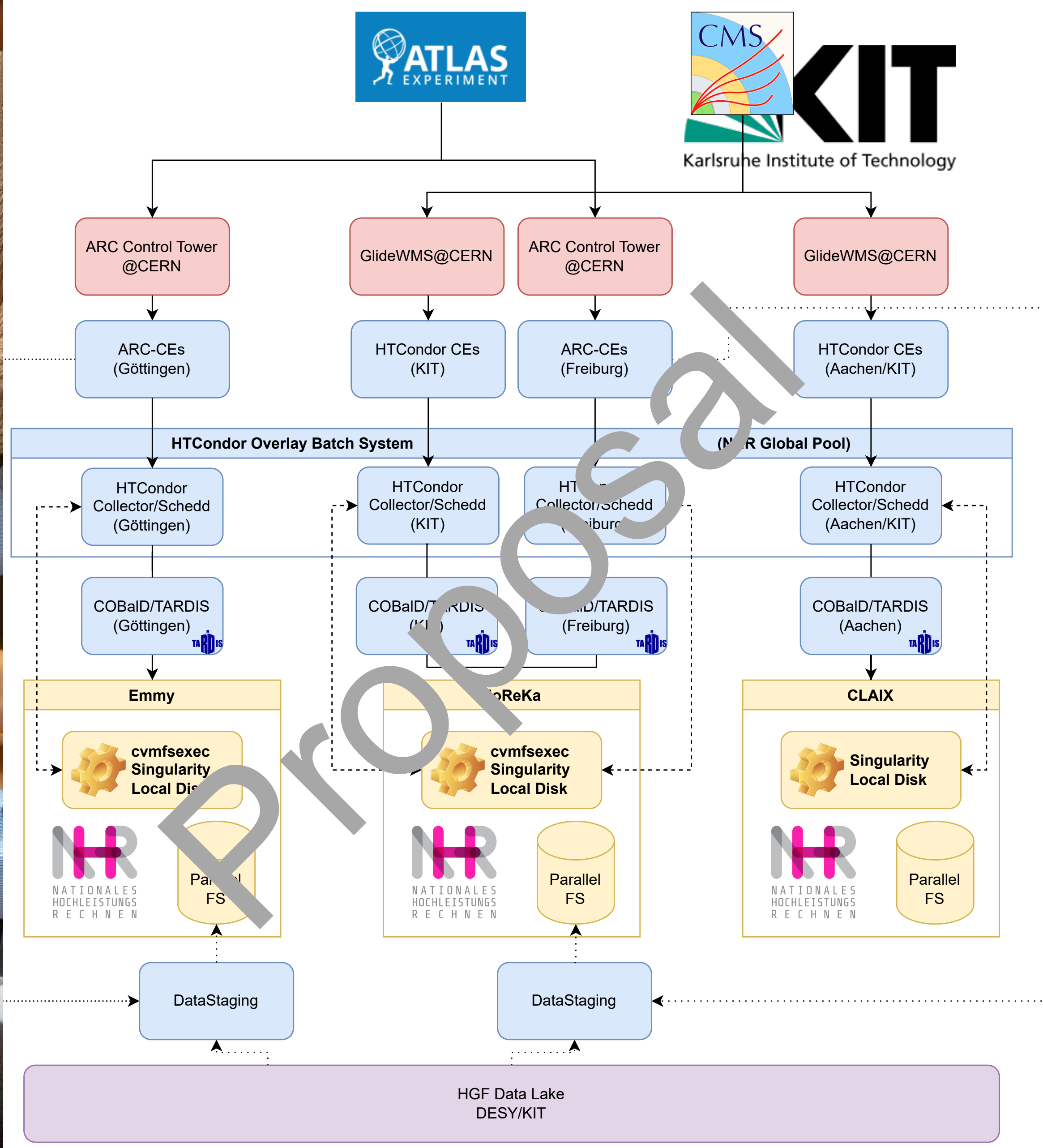
- University Tier-2 compute will be gradually transitioned towards NHR HPCs (starting 2025)
- We are well-prepared thanks to the comprehensive toolset developed within IDT-UM & FIDIUM





By collaborating we can share the effort!

AI generated

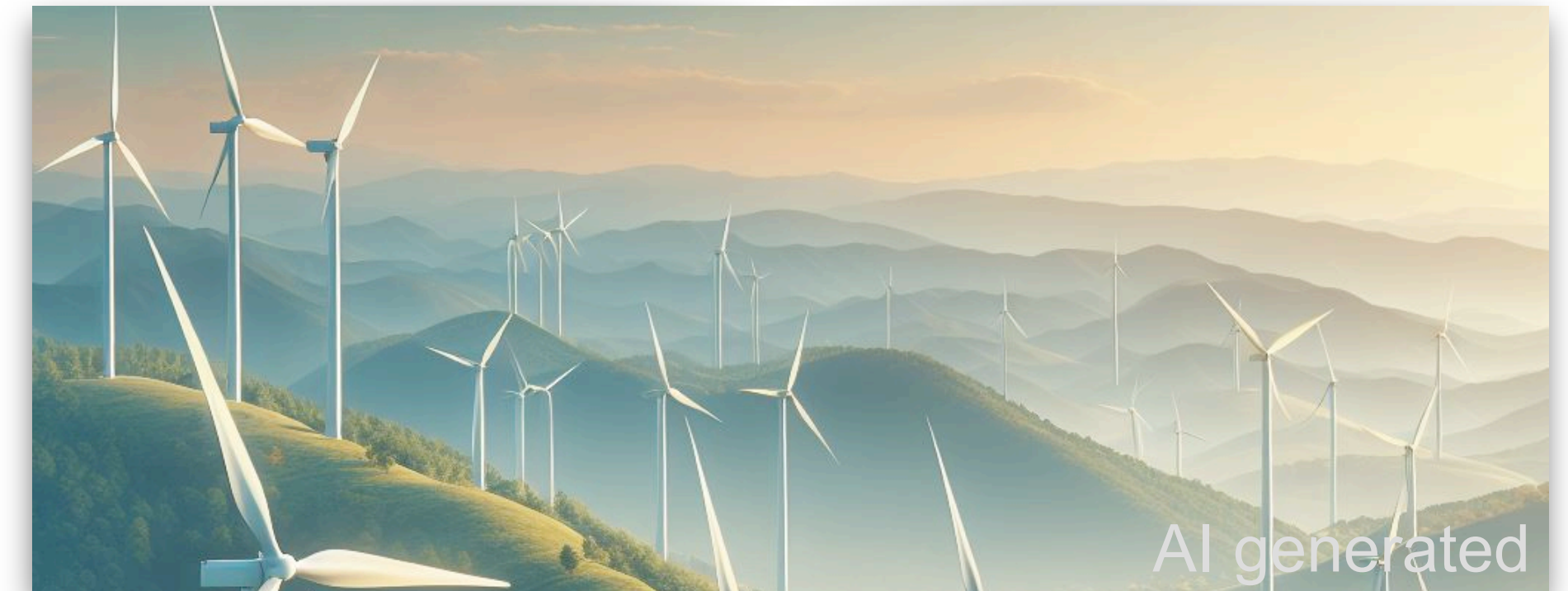


ETP & SCC

Sustainability & Energy cost aware scheduling

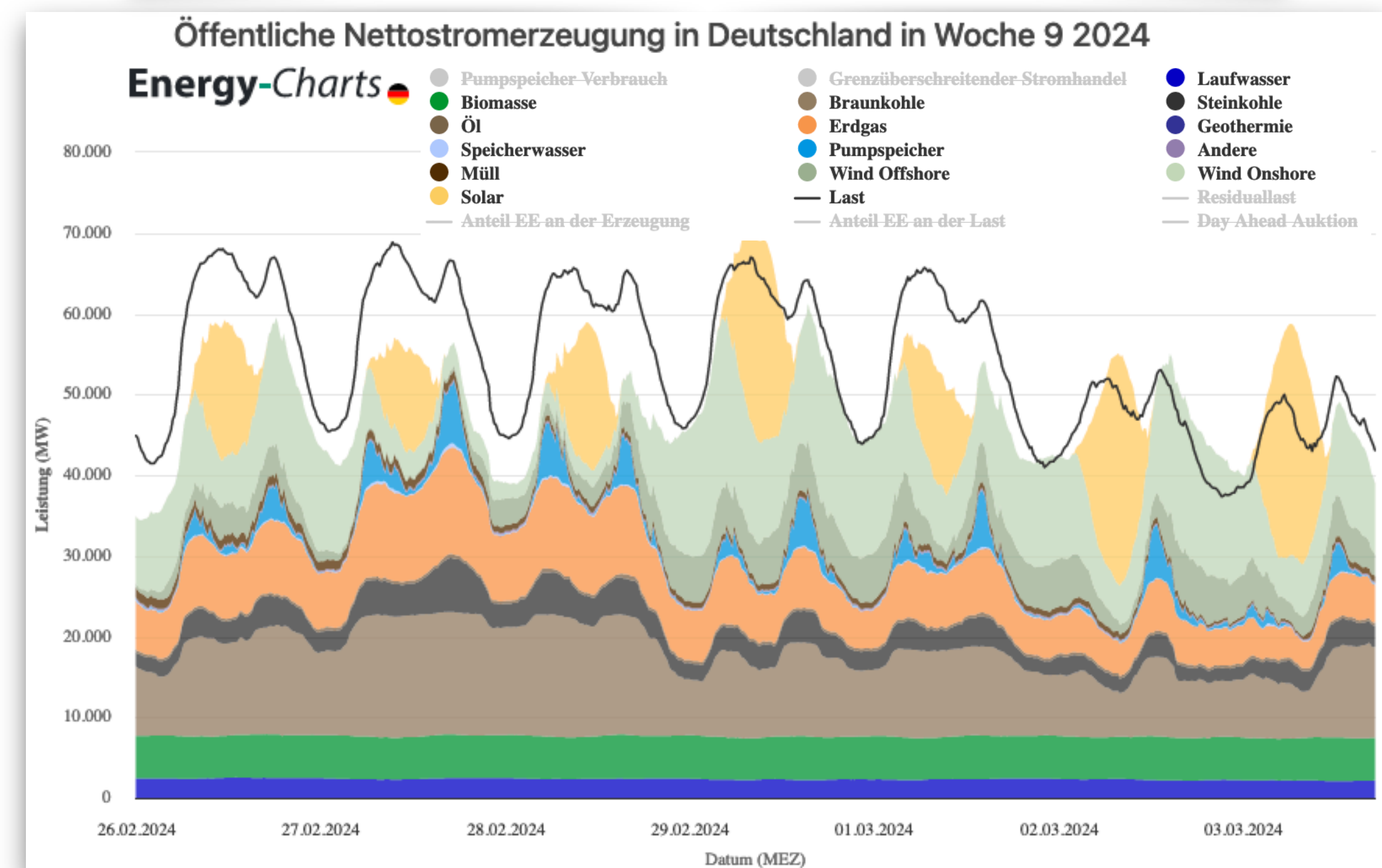
Sustainability & Climate Neutral Compute (Wind energy)

- Looking for industry cooperations like windCores (Westfalen Wind)
- Include them as opportunistic resource



Energy cost aware resource scheduling

- Include current energy mix or energy price into the COBalD meta-scheduler
- Increase resource utilisation when prices are low or energy is clean
- Decrease it otherwise



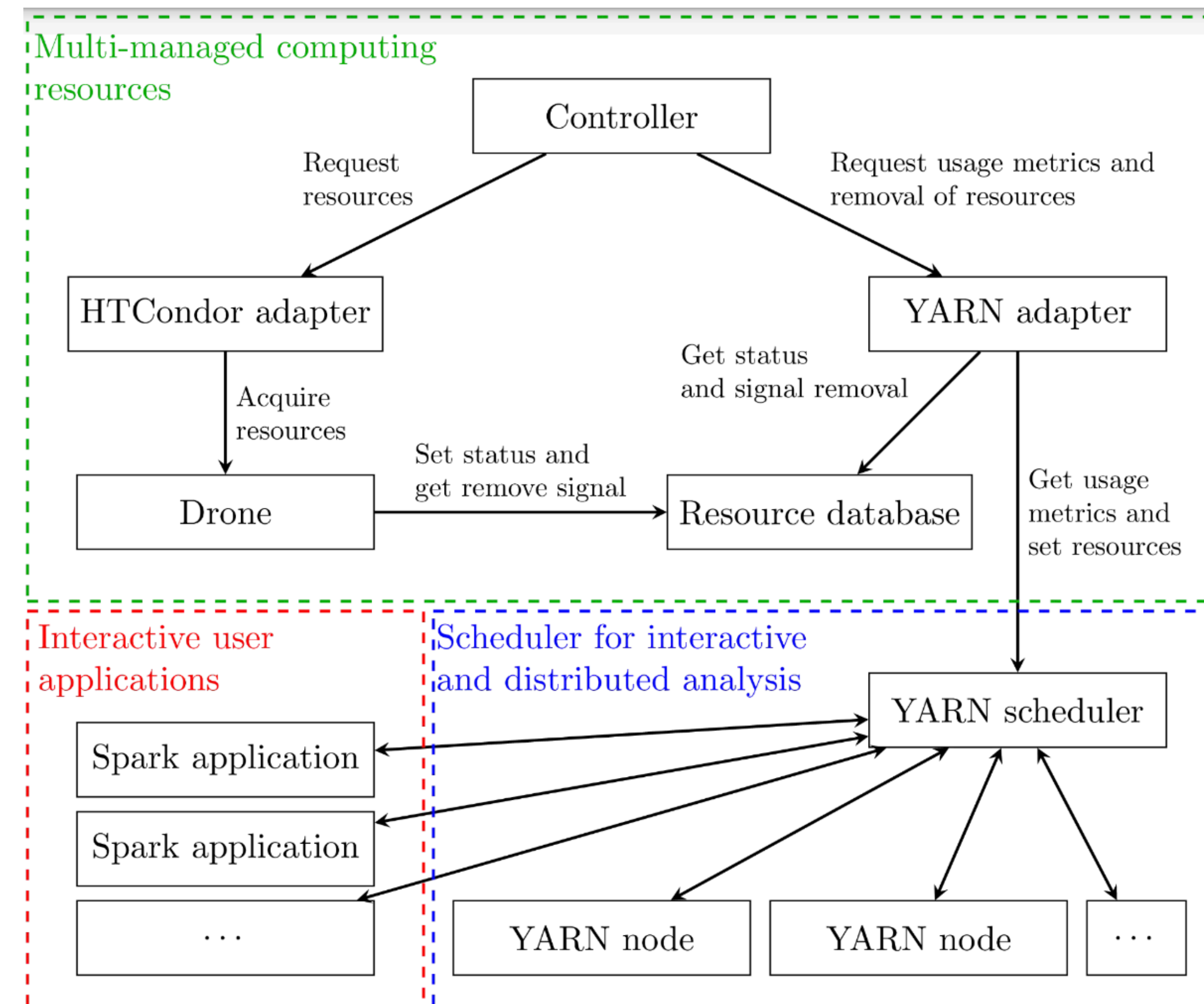
Dynamic Interactive Analysis Cluster (Tier-3)

- Find the balance between interactive and batch utilization at Tier-3 clusters
- Idea: Deploy on-demand Apache Spark analysis cluster on HTCondor or K8S
- COBaID/TARDIS perfectly suited to do the balancing depending on the utilization/demand
- Why Apache Spark? Cluster can be shared among different users.
- Why not Dask? Each user need its own cluster, no multi-user scheduling possible.



Dynamic Interactive Analysis Cluster (Tier-3)

- Find the balance between interactive and batch utilization at Tier-3 clusters
- Idea: Deploy on-demand Apache Spark analysis cluster on HTCondor or K8S
- COBaID/TARDIS perfectly suited to do the balancing depending on the utilization/demand
- Why Apache Spark? Cluster can be shared among different users.
- Why not Dask? Each user need its own cluster, no multi-user scheduling possible.



Conclusion

Developed enabling toolset for a dynamic federation of heterogeneous compute resources:

- Result of a fruitful experiment overarching collaboration in the ErUM-Data Pilot Projects IDT-UM and FIDIUM (→ should be intensified in FCR 3)
- Impressively shown it is production ready in many different use-cases
- Actively used in WLCG computing, FIDIUM & PUNCH4NFDI for workflows from HEP, Astronomy and Lattice QCD
- Still, the full potential of our toolset has not been exploited yet

Conclusion

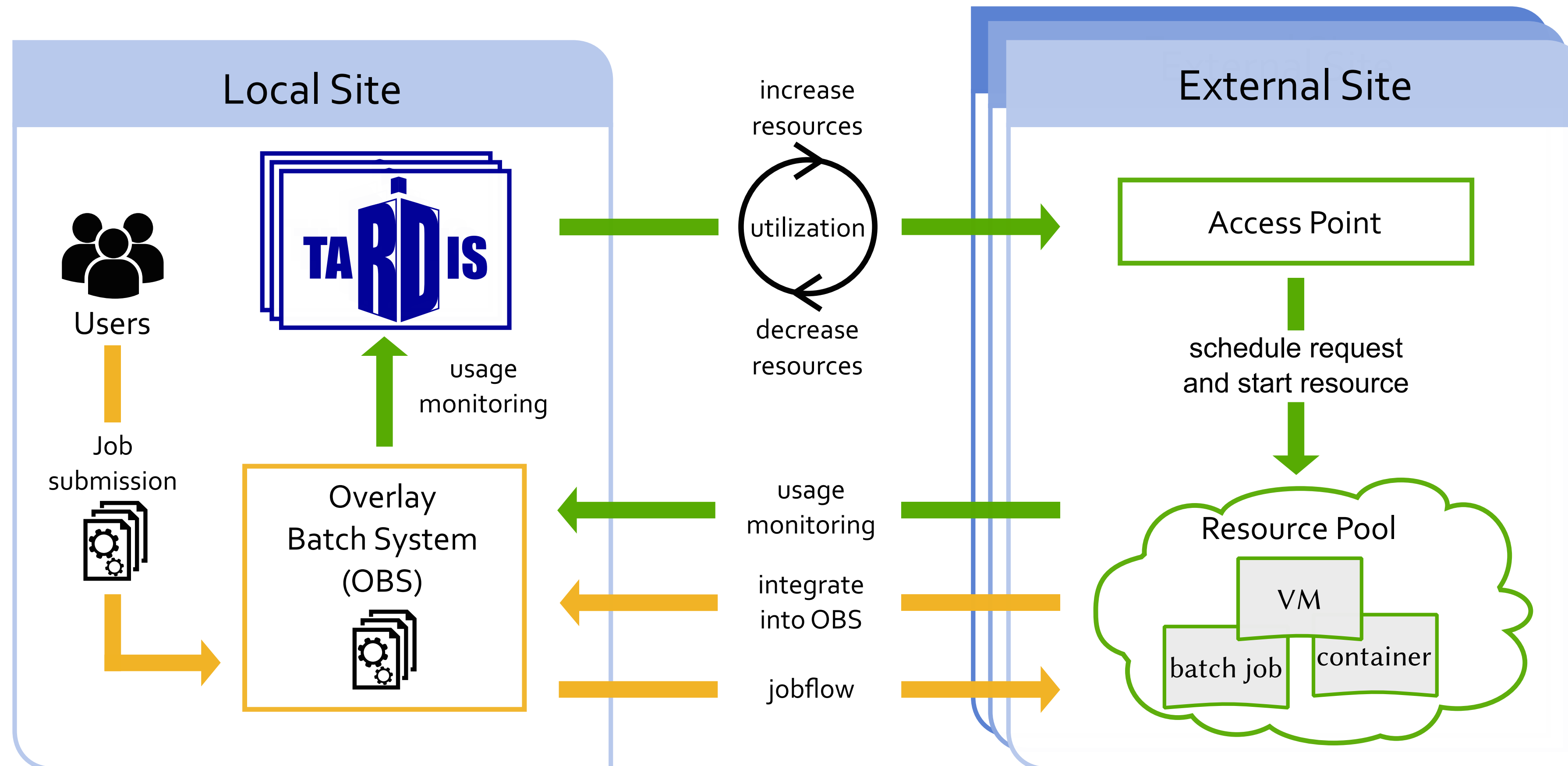
Developed enabling toolset for a dynamic federation of heterogeneous compute resources:

- Result of a fruitful experiment overarching collaboration in the ErUM-Data Pilot Projects IDT-UM and FIDIUM (→ should be intensified in FCR 3)
- Impressively shown it is production ready in many different use-cases
- Actively used in WLCG computing, FIDIUM & PUNCH4NFDI for workflows from HEP, Astronomy and Lattice QCD
- Still, the full potential of our toolset has not been exploited yet

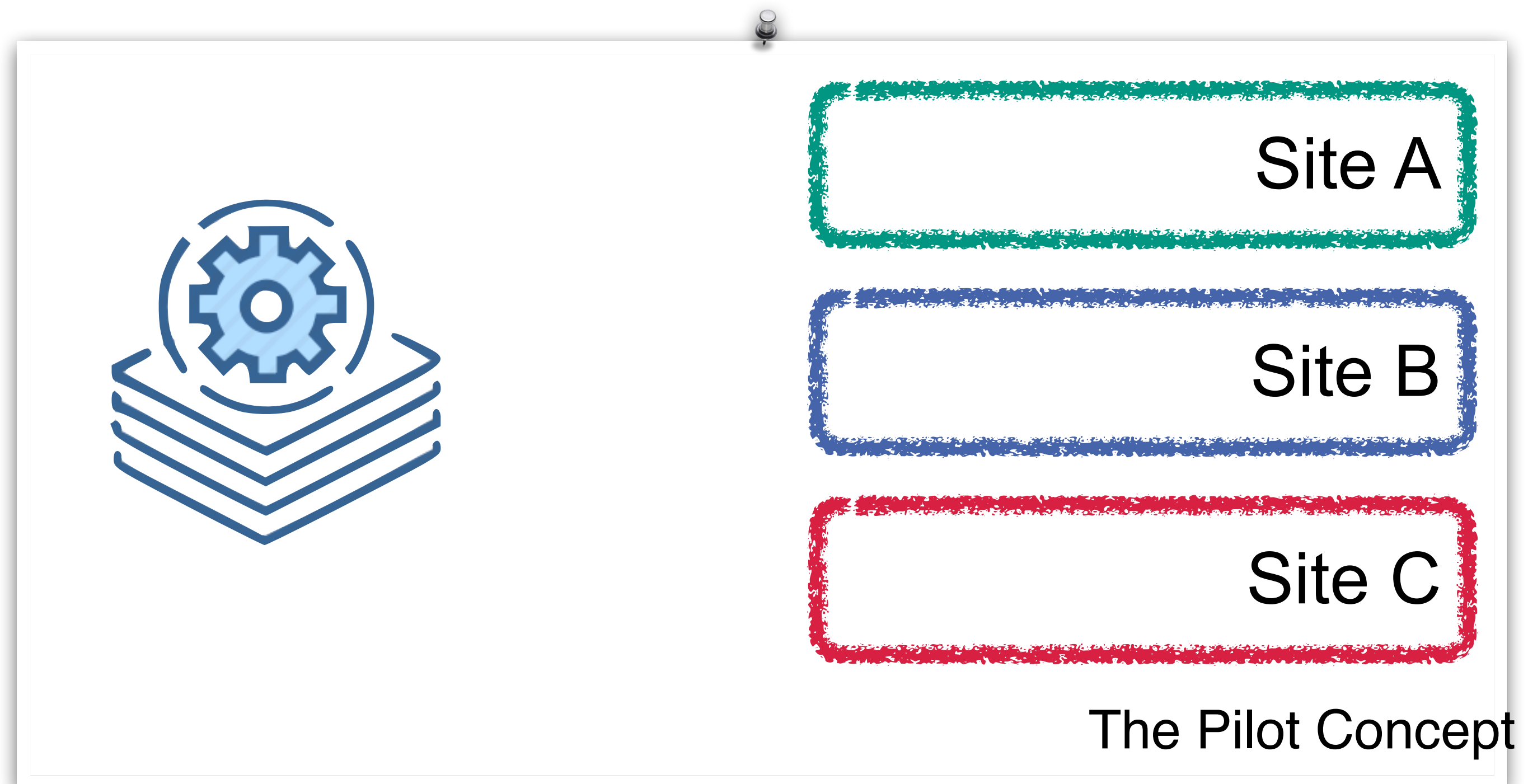
We have promising future plans, that require to get sufficient funding!

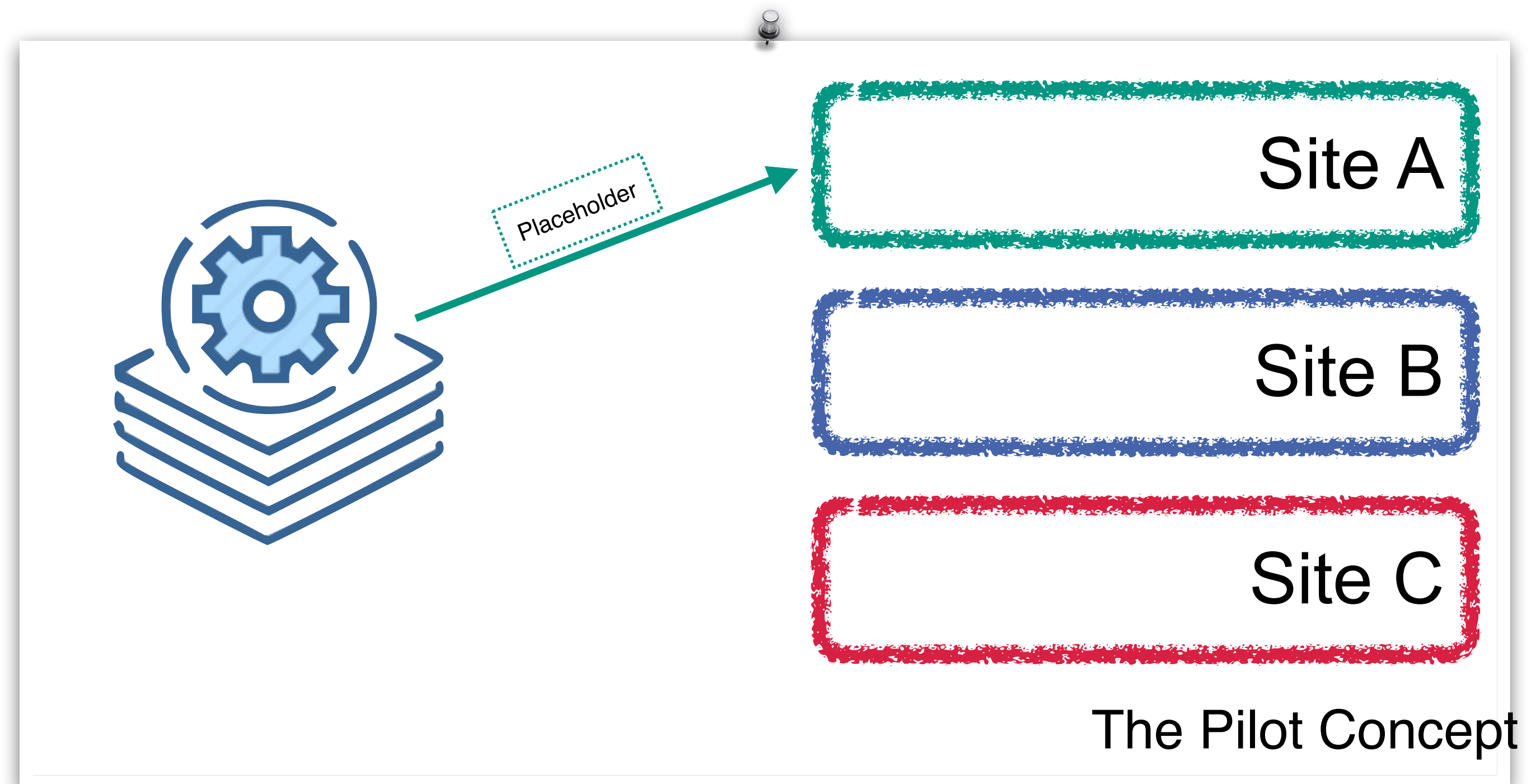
Backup

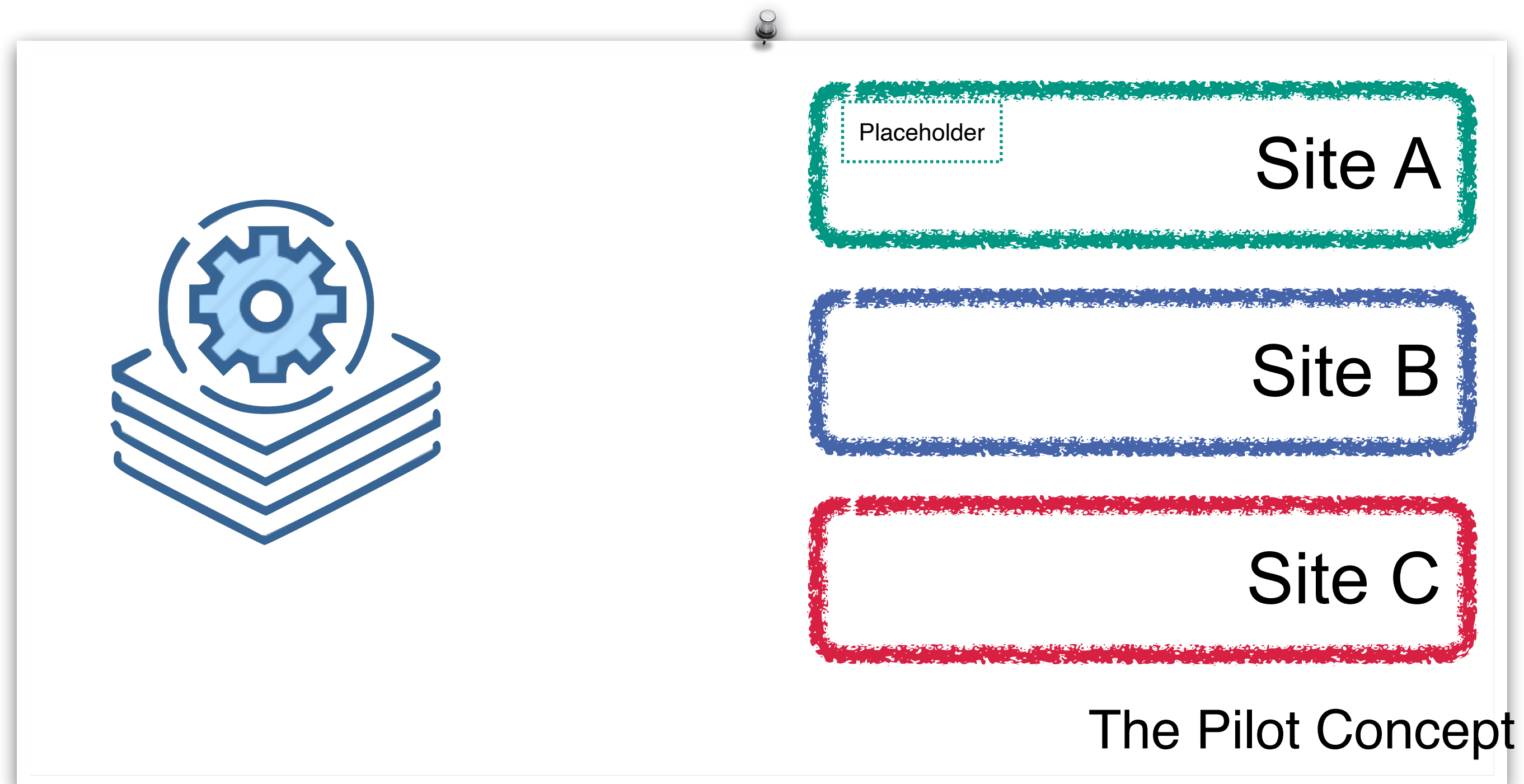
COBaID/TARDIS & Opportunistic Resources in Practice

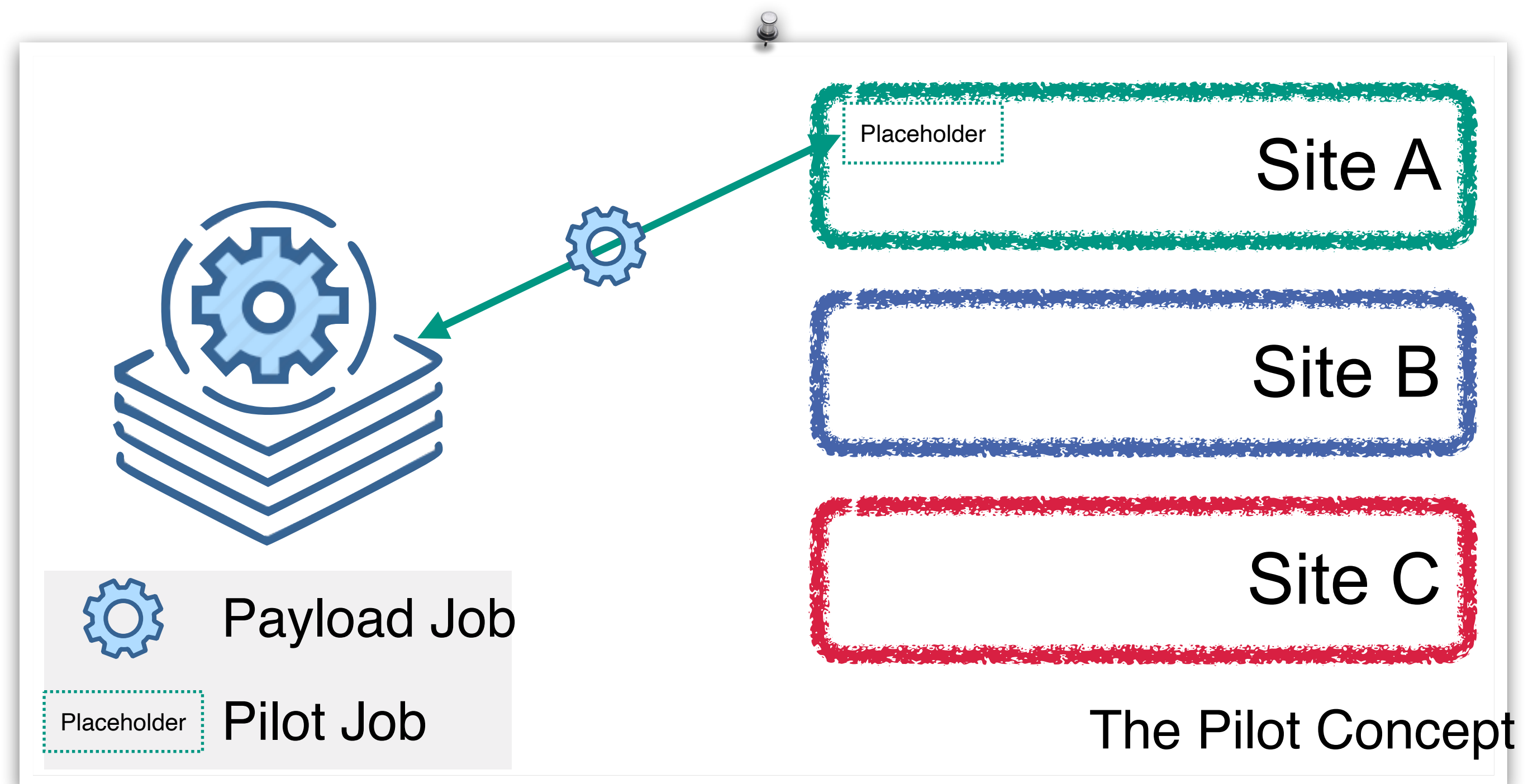


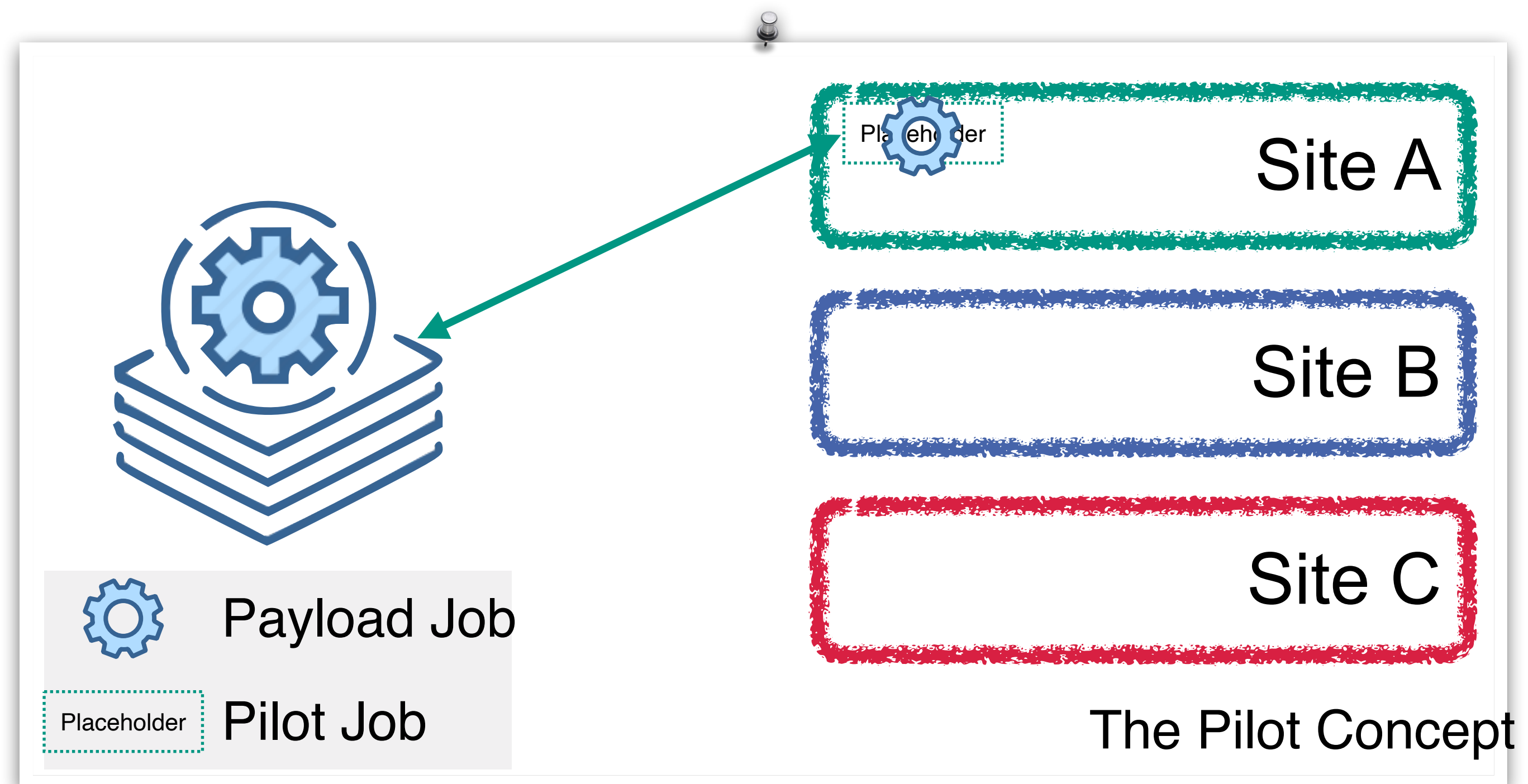


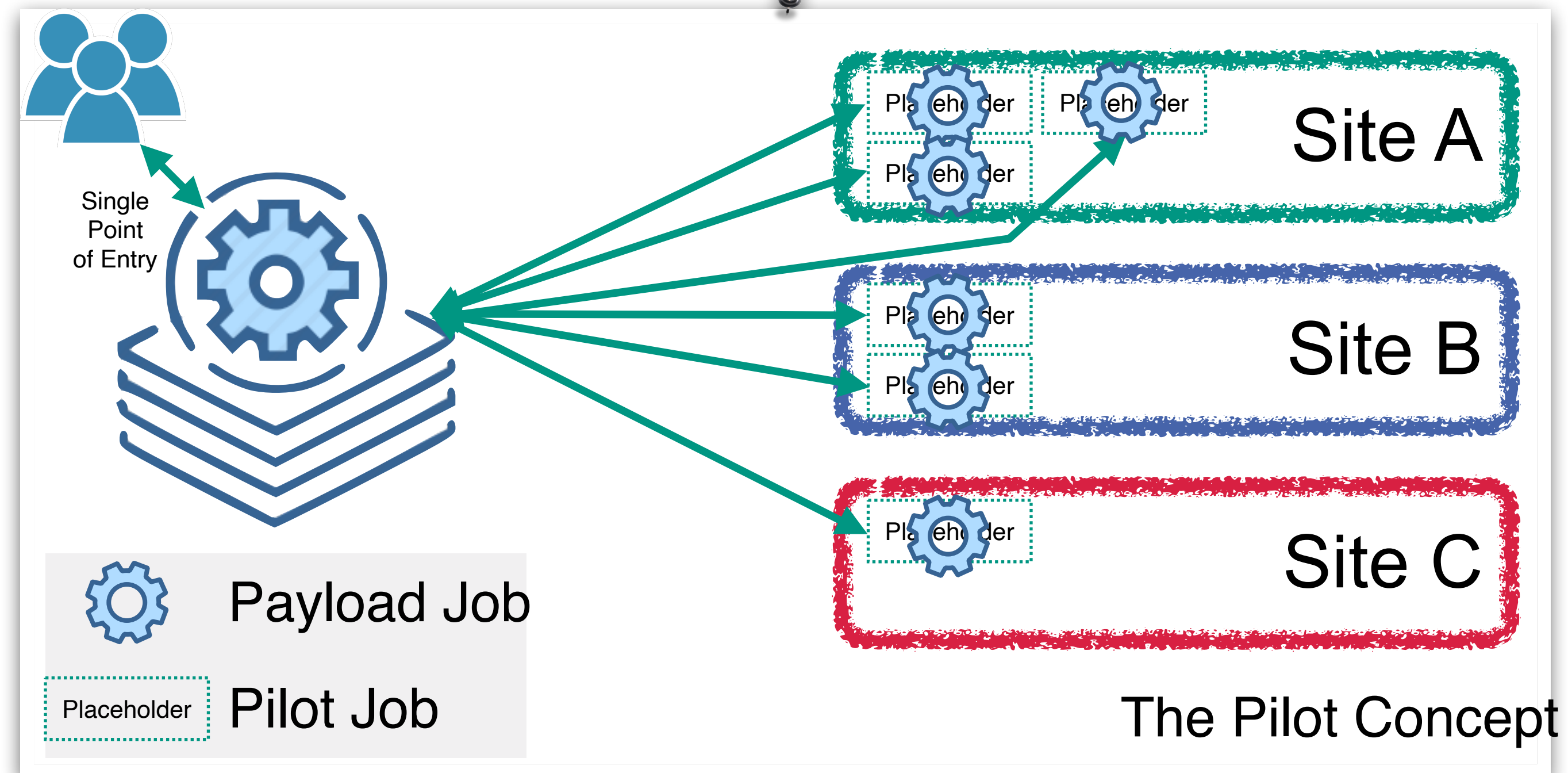






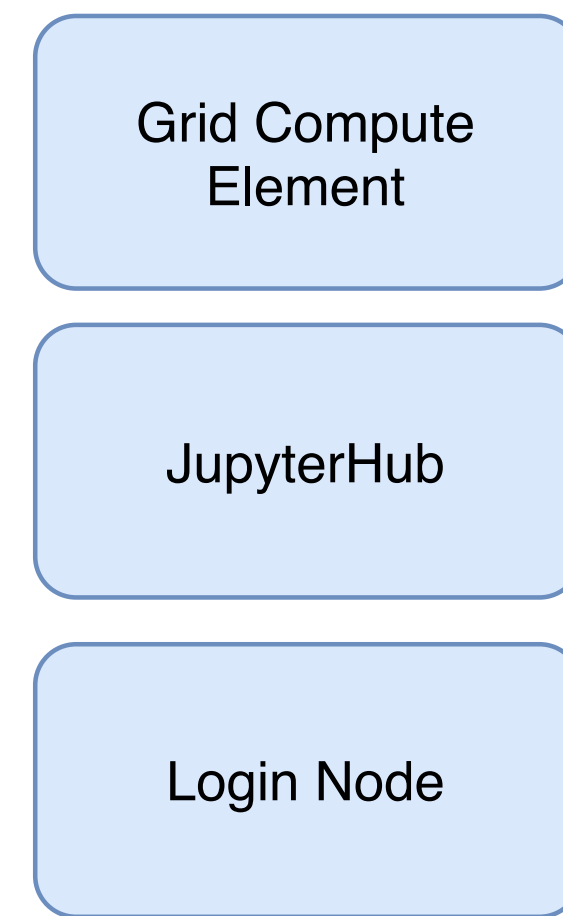




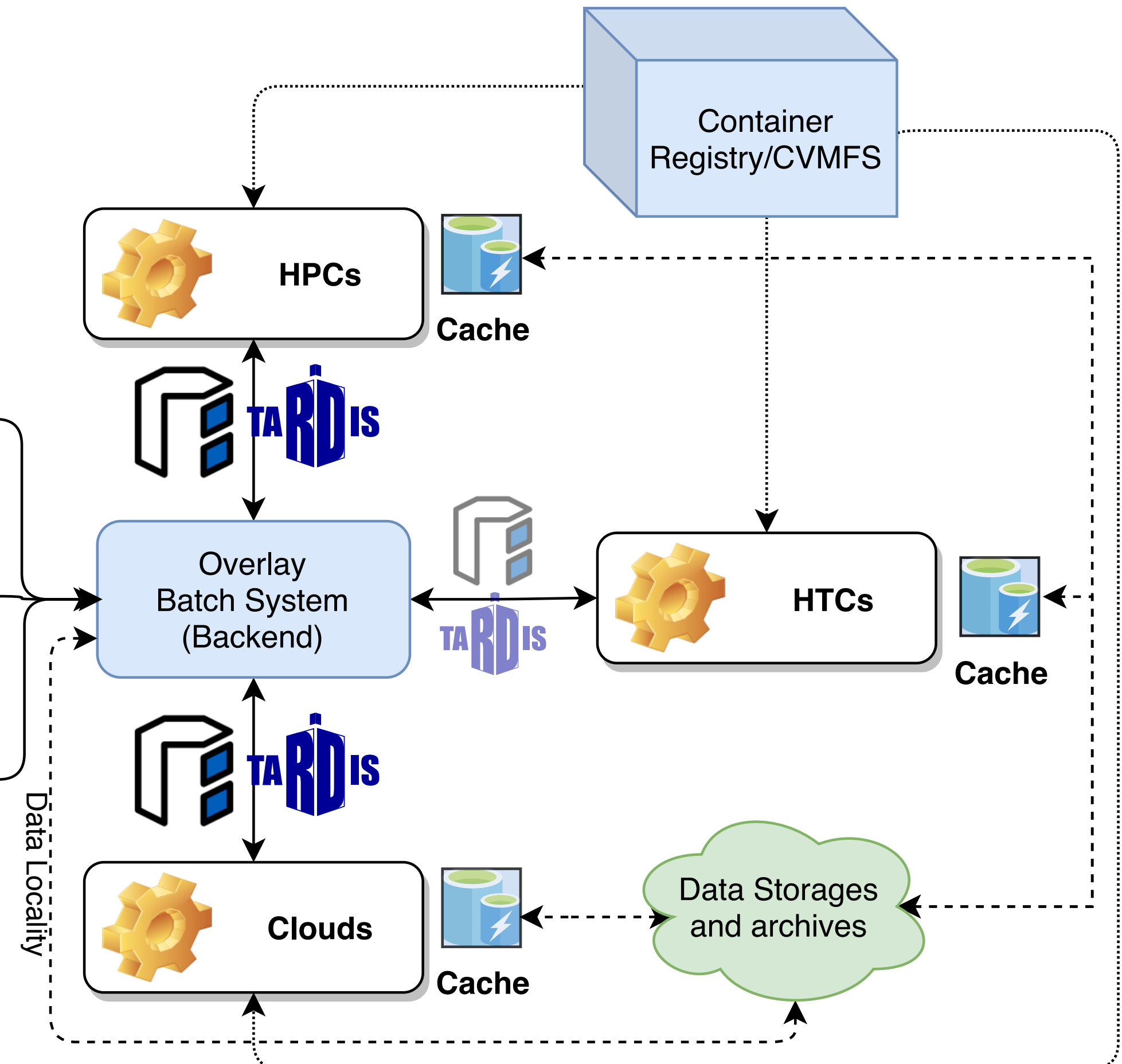


Towards the Compute4PUNCH Infrastructure

- Establish a federated heterogeneous compute infrastructure for PUNCH
- Integrate data storages, archives and opportunistic caches



Single Point(s) of Entry



- Introduce data-locality aware scheduling
- Benefit from experiences, concepts and tools available in HEP community

Workflows on Compute4PUNCH & Storage4PUNCH



Workflows on Compute4PUNCH & Storage4PUNCH

LOFAR Radio imaging workflow

■ Low Frequency Array (LOFAR)



Workflows on Compute4PUNCH & Storage4PUNCH

LOFAR Radio imaging workflow

- **Low Frequency Array (LOFAR)**
- Reconstruction of the sky brightness distribution from recorded interferometry data
- Software provided via apptainer container
- Data is available on Storage4PUNCH (~150 GB)

```
# HTCondor Job Description
#=====
# The name of the executable
executable = wsclean.sh

# where to store log files
output = logs/$(cluster).$(process).out
error = logs/$(cluster).$(process).err
log = logs/cluster.log

# The requirements of your job. Memory is in MBytes
request_cpus = 8
request_memory = 20480

# In which container your job should be executed.
+SINGULARITY_JOB_CONTAINER = "linc-wn:latest"

# and we would like to submit it only once
queue 1
```

retrieving data from
Storage4PUNCH

running imager

download final image
from login node

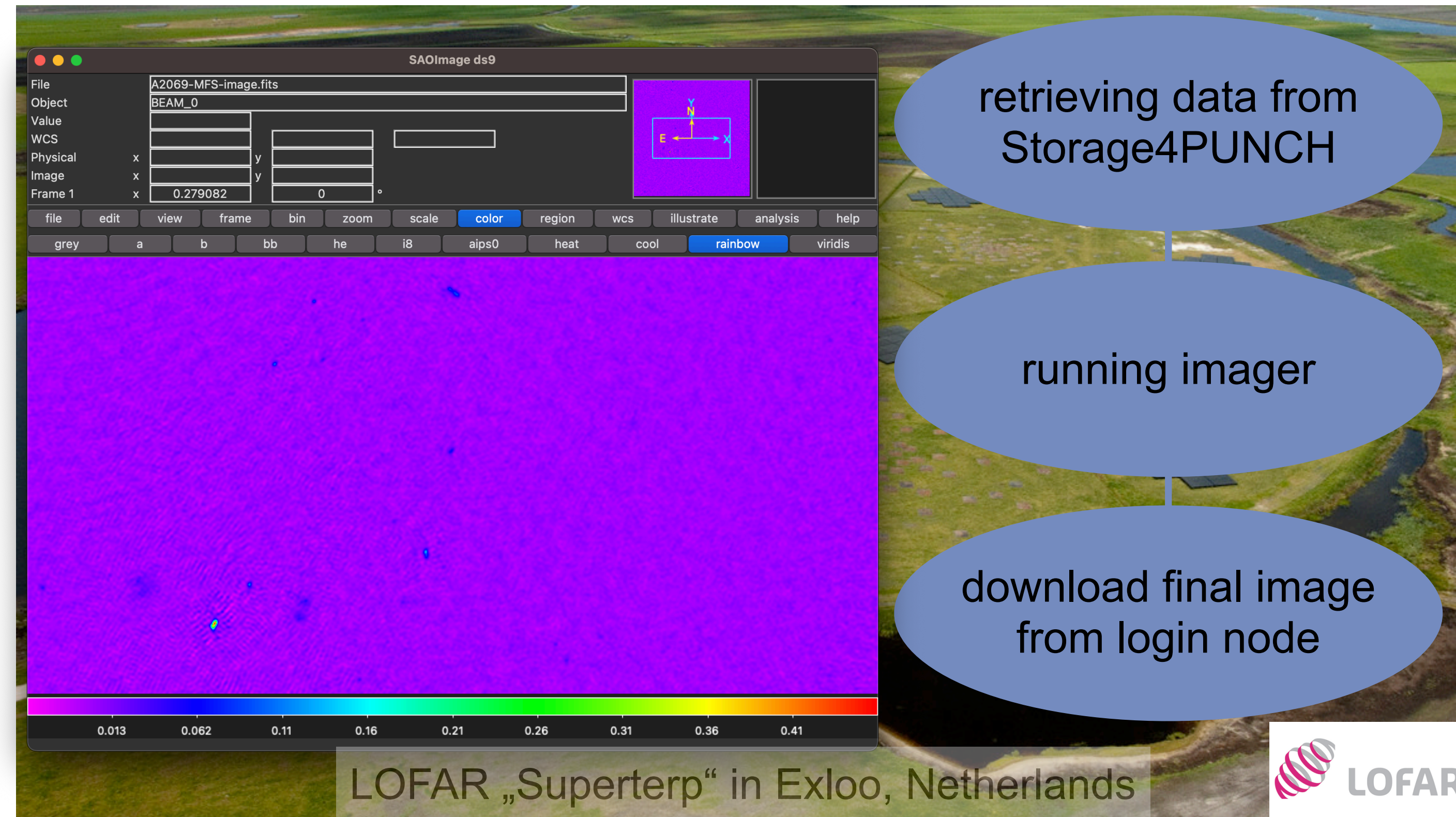
LOFAR „Superterp“ in Exloo, Netherlands



Workflows on Compute4PUNCH & Storage4PUNCH

LOFAR Radio imaging workflow

- **Low Frequency Array (LOFAR)**
- Reconstruction of the sky brightness distribution from recorded interferometry data
- Software provided via apptainer container
- Data is available on Storage4PUNCH (~150 GB)



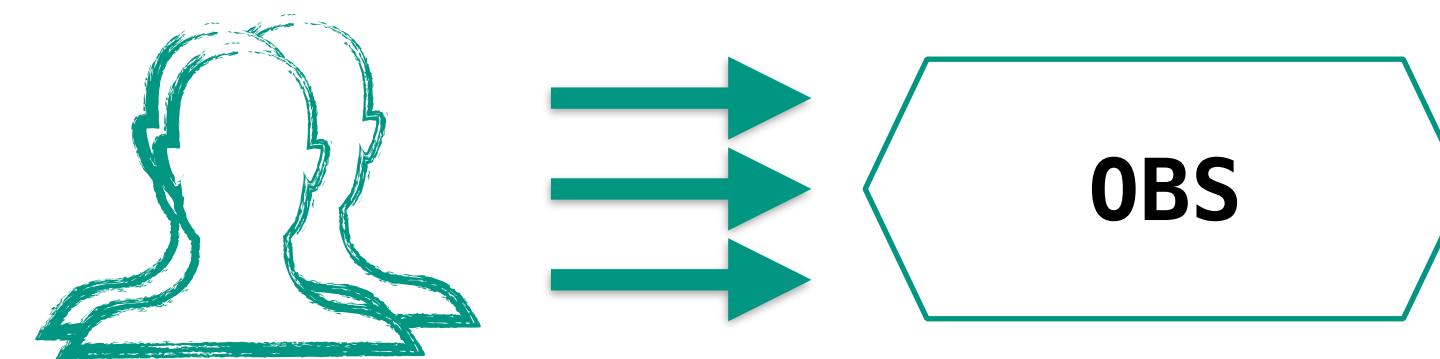
Resource Meta-Scheduler

Resource Meta-Scheduler

Classical [Job to Resource to Job](#) meta-scheduler:

Resource Meta-Scheduler

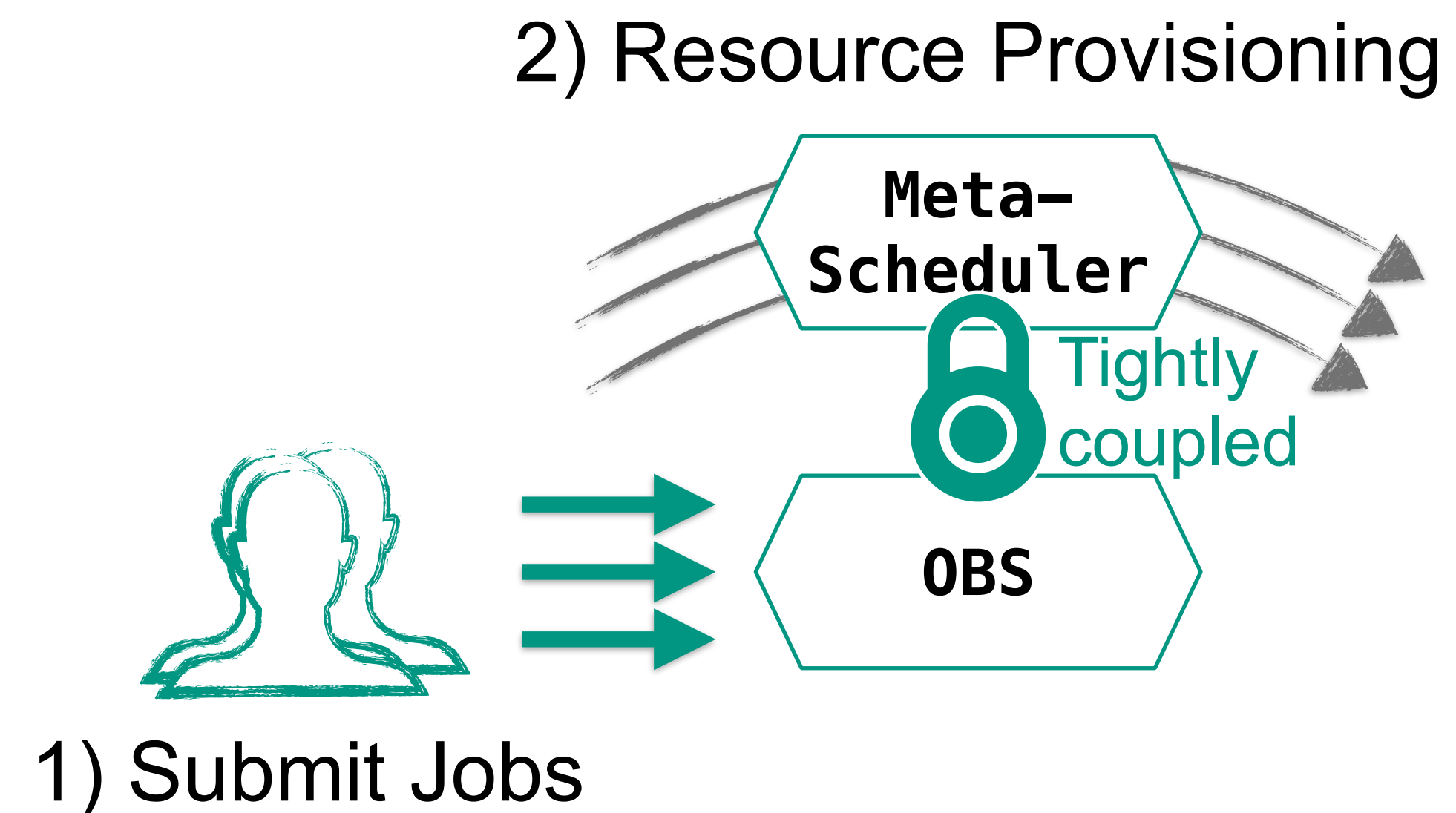
Classical **Job to Resource to Job** meta-scheduler:



1) Submit Jobs

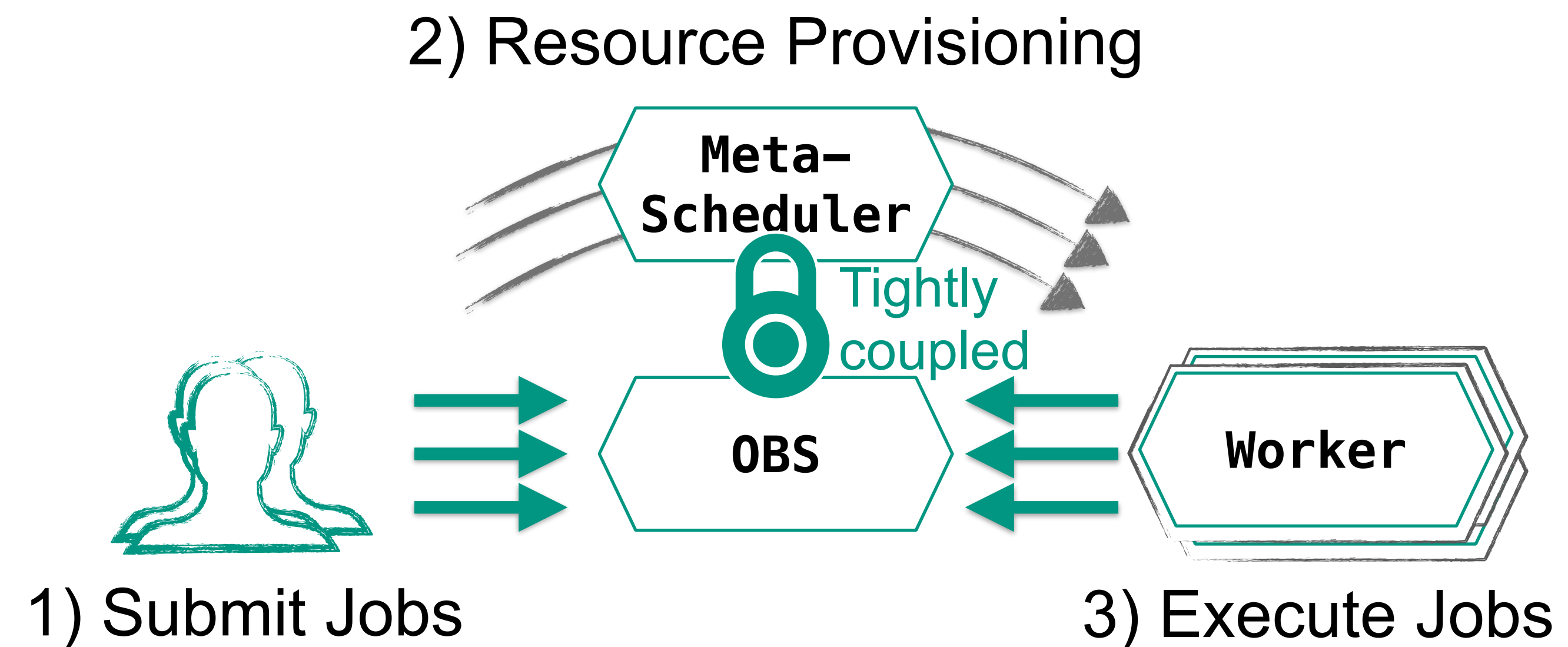
Resource Meta-Scheduler

Classical Job to Resource to Job meta-scheduler:



Resource Meta-Scheduler

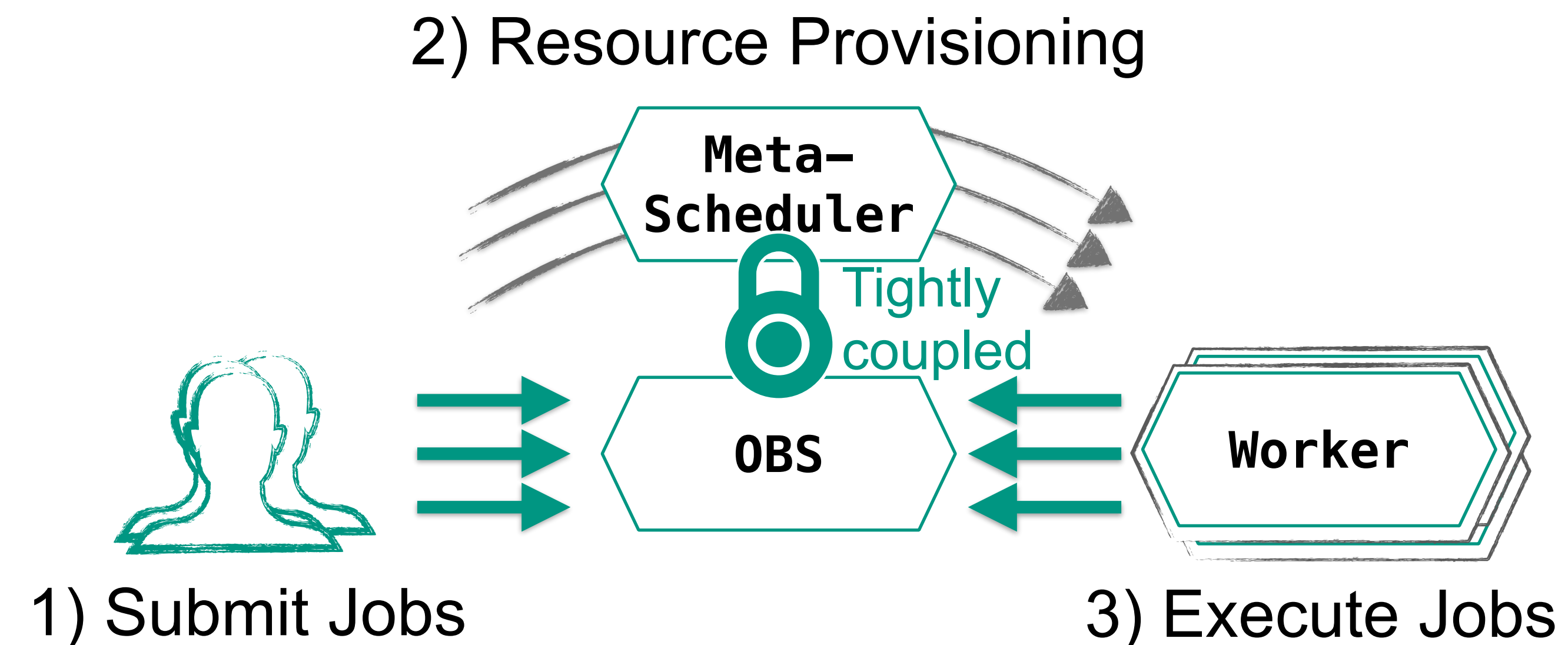
Classical Job to Resource to Job meta-scheduler:



Resource Meta-Scheduler

Classical **Job to Resource to Job** meta-scheduler:

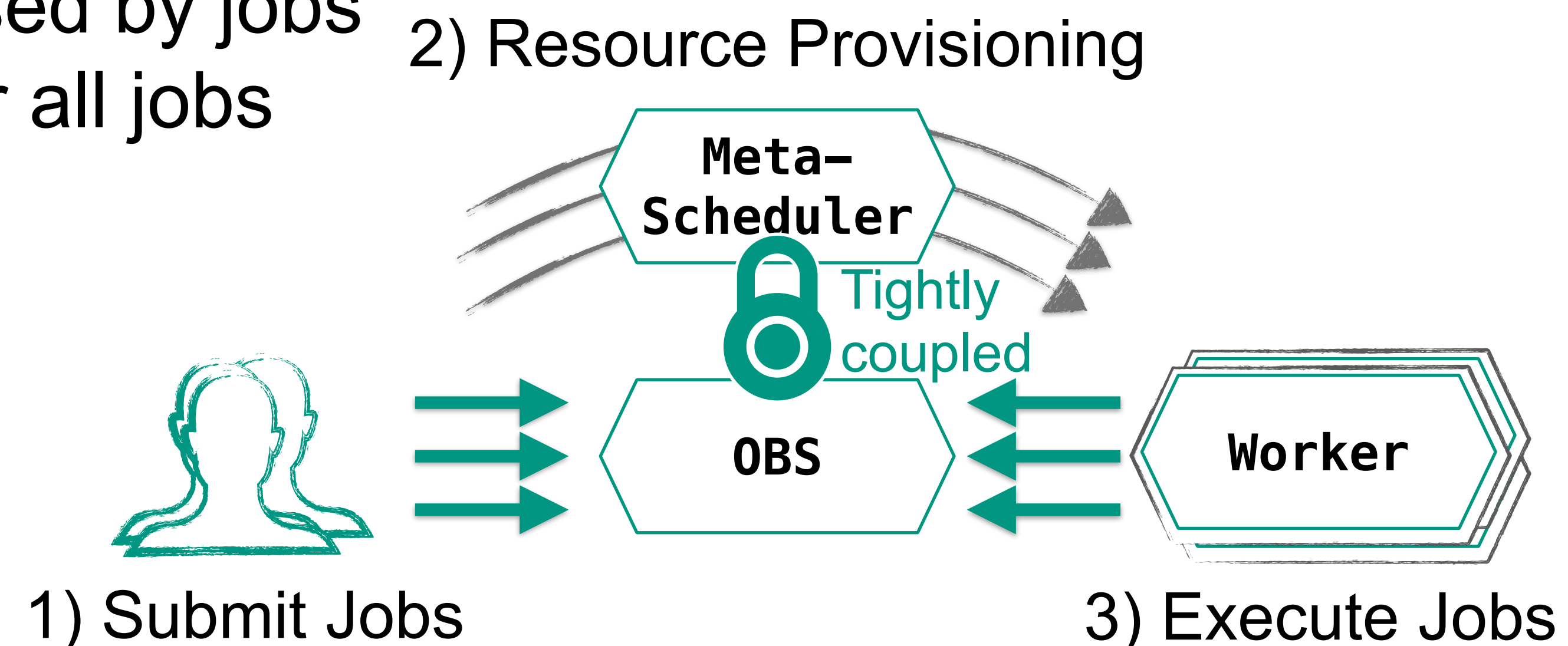
- Dynamic resource acquisition matching user demand
 - Trivial to support **new providers** for many users
 - Difficult to manage **several providers** for many users



Resource Meta-Scheduler

Classical **Job to Resource to Job** meta-scheduler:

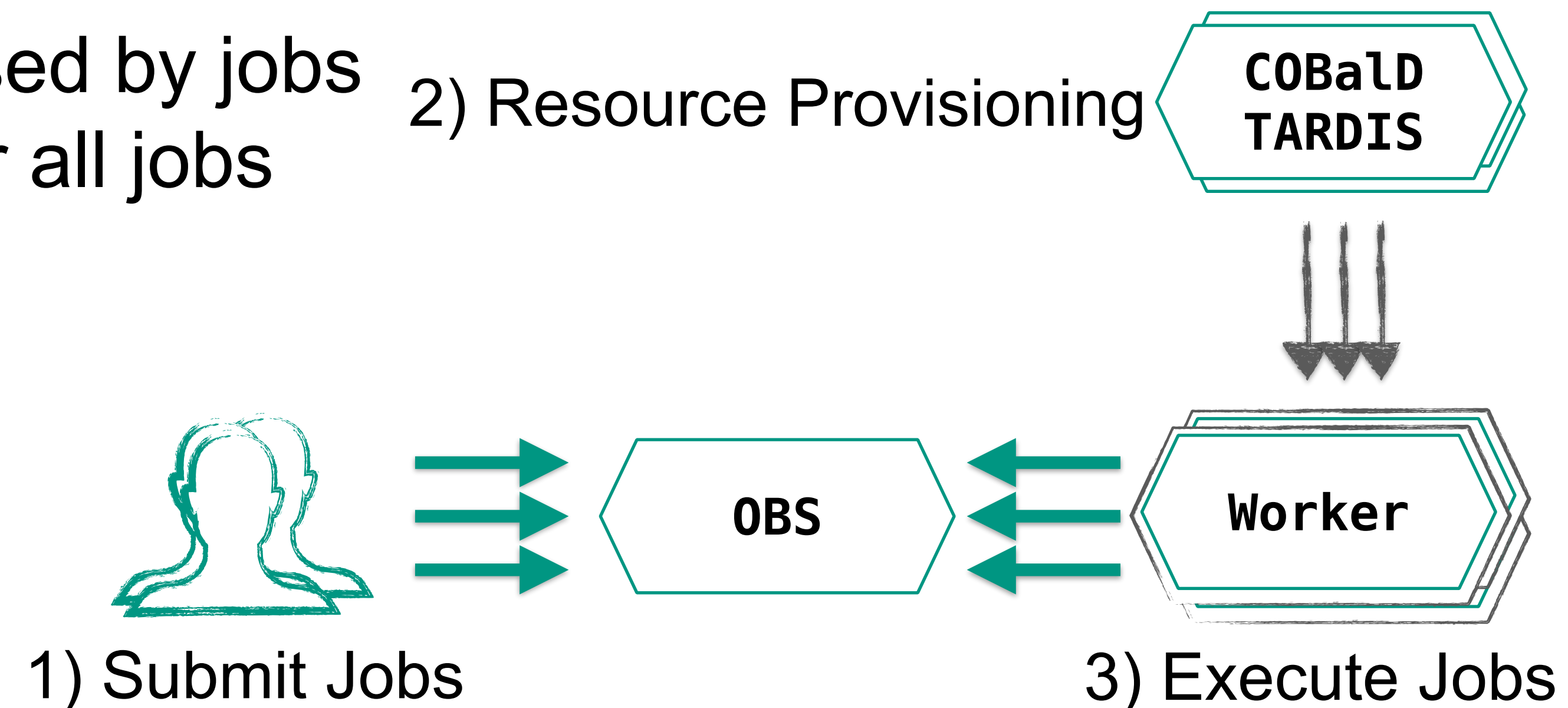
- Dynamic resource acquisition matching user demand
 - Trivial to support **new providers** for many users
 - Difficult to manage **several providers** for many users
- Job scheduling in overlay batch system
 - Unreliable to **predict resources** used by jobs
 - Efficient to **integrate resources** for all jobs



Resource Meta-Scheduler

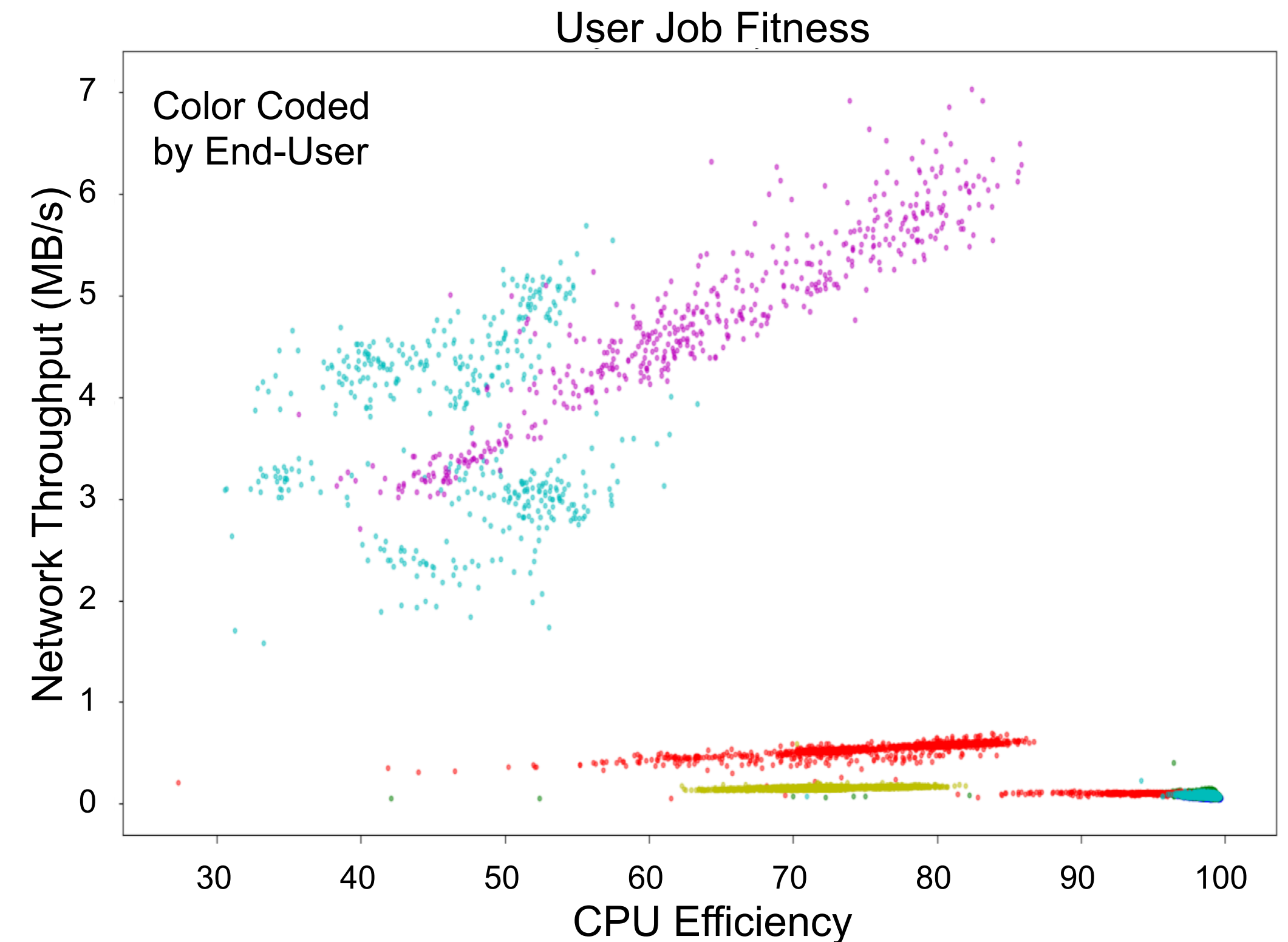
Classical **Job to Resource to Job** meta-scheduler:

- Dynamic resource acquisition matching user demand
 - Trivial to support **new providers** for many users
 - Difficult to manage **several providers** for many users
- Job scheduling in overlay batch system
 - Unreliable to **predict resources** used by jobs
 - Efficient to **integrate resources** for all jobs



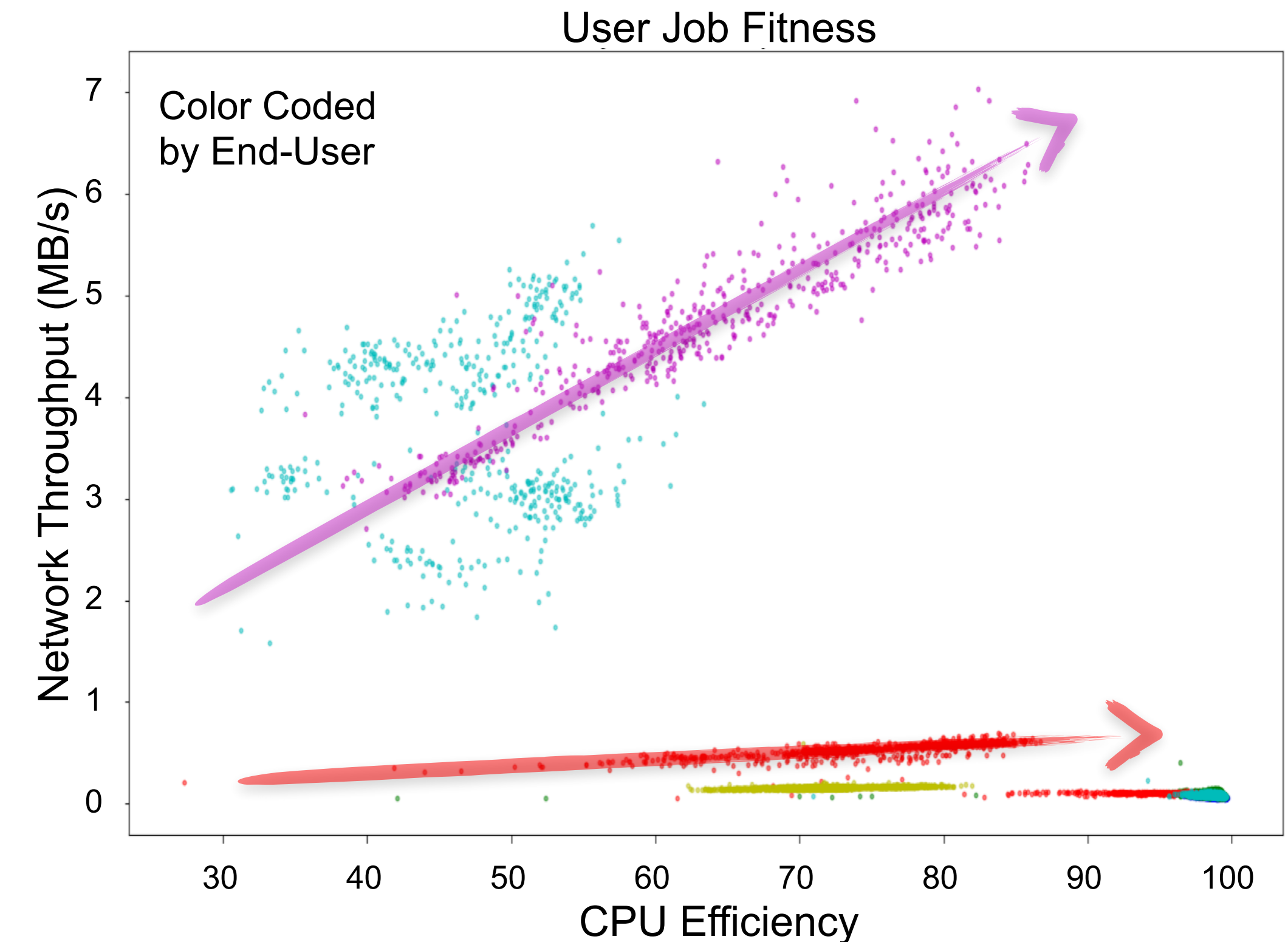
Implicit Resource Scheduling via Feedback

- Respect network availability and congestion for provisioning
 - Congested network is the bottleneck for opportunistic resources
 - Non-linear interference and noticeable measurement overhead

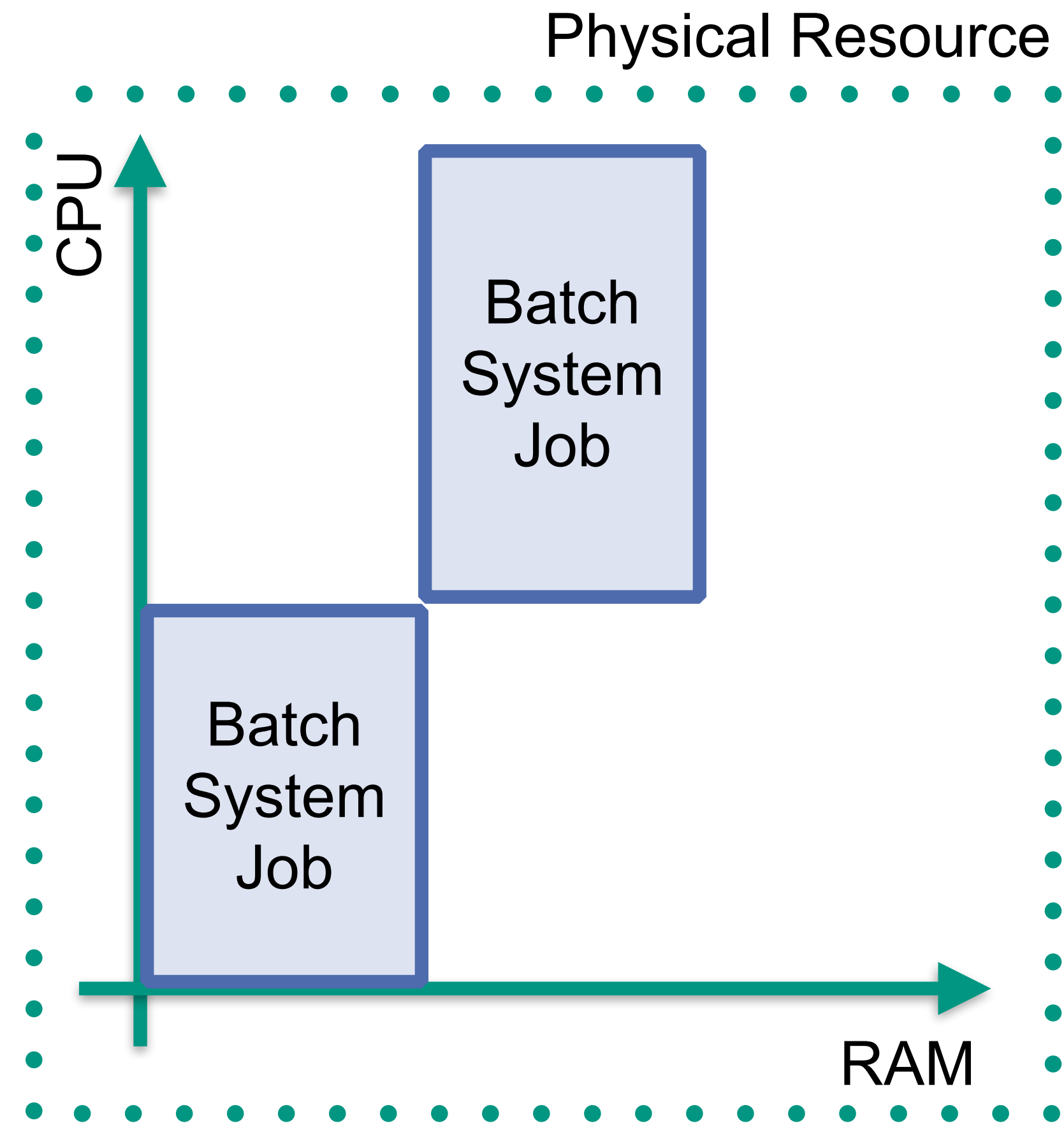


Implicit Resource Scheduling via Feedback

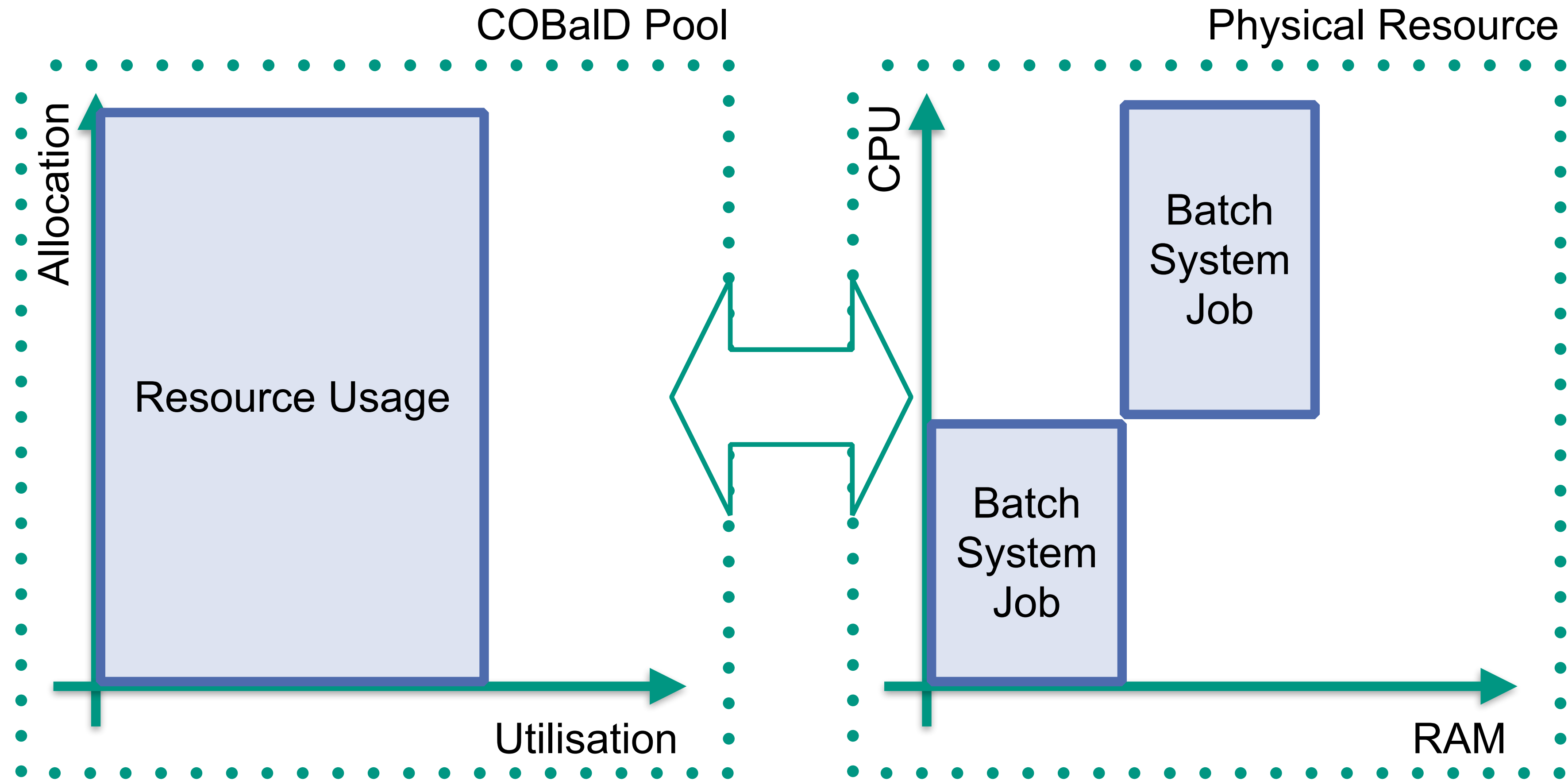
- Respect network availability and congestion for provisioning
 - Congested network is the bottleneck for opportunistic resources
 - Non-linear interference and noticeable measurement overhead
- Research: Implicitly schedule network capacity via side-effects
 - Cheap CPU efficiency query as boundary for network efficiency (and other resources)
 - CPU efficiency implies general fitness
- Safeguard to push the maximum possible data analysis jobs to opportunistic resources



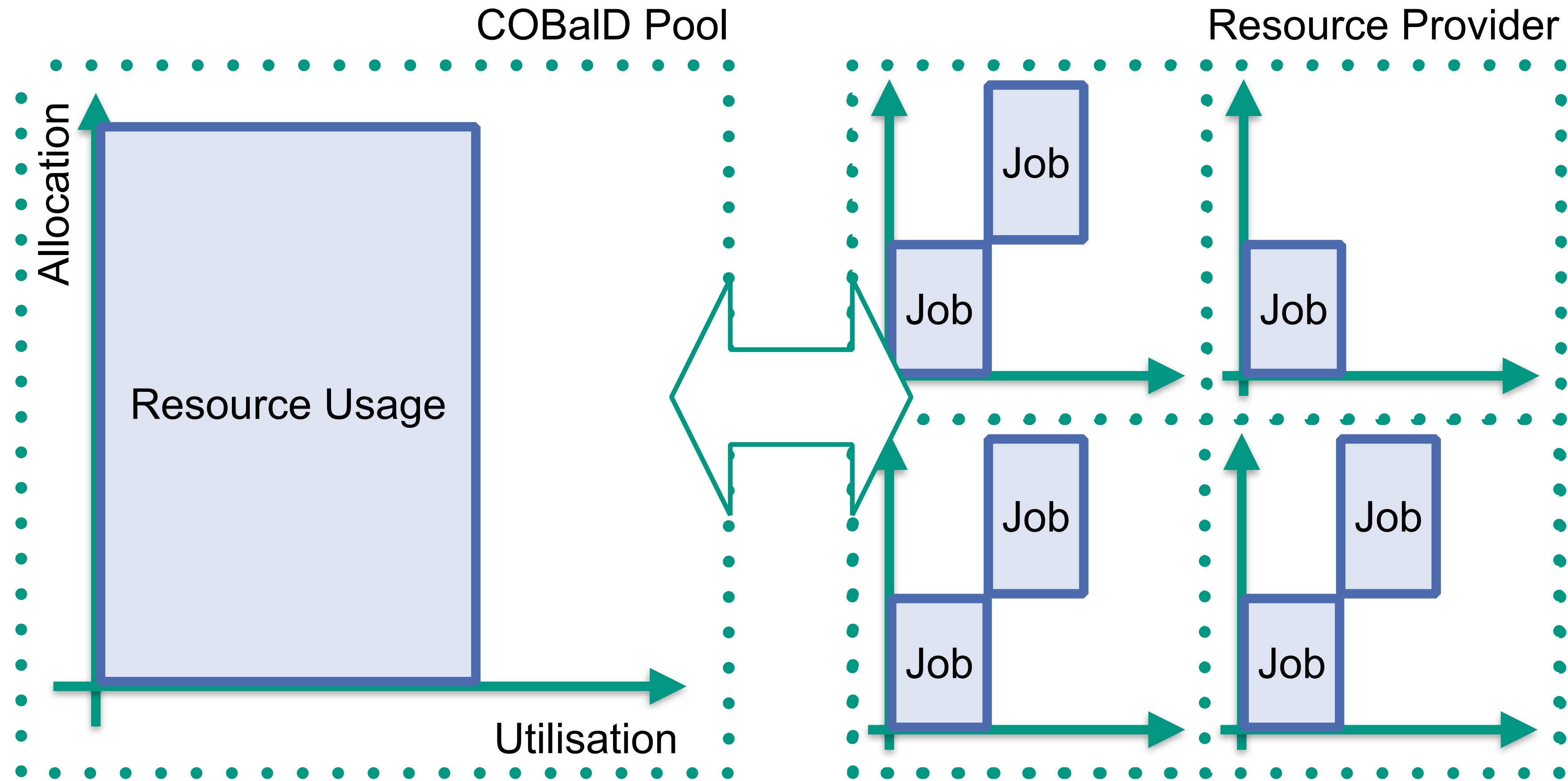
COBaID Resource Pool Model



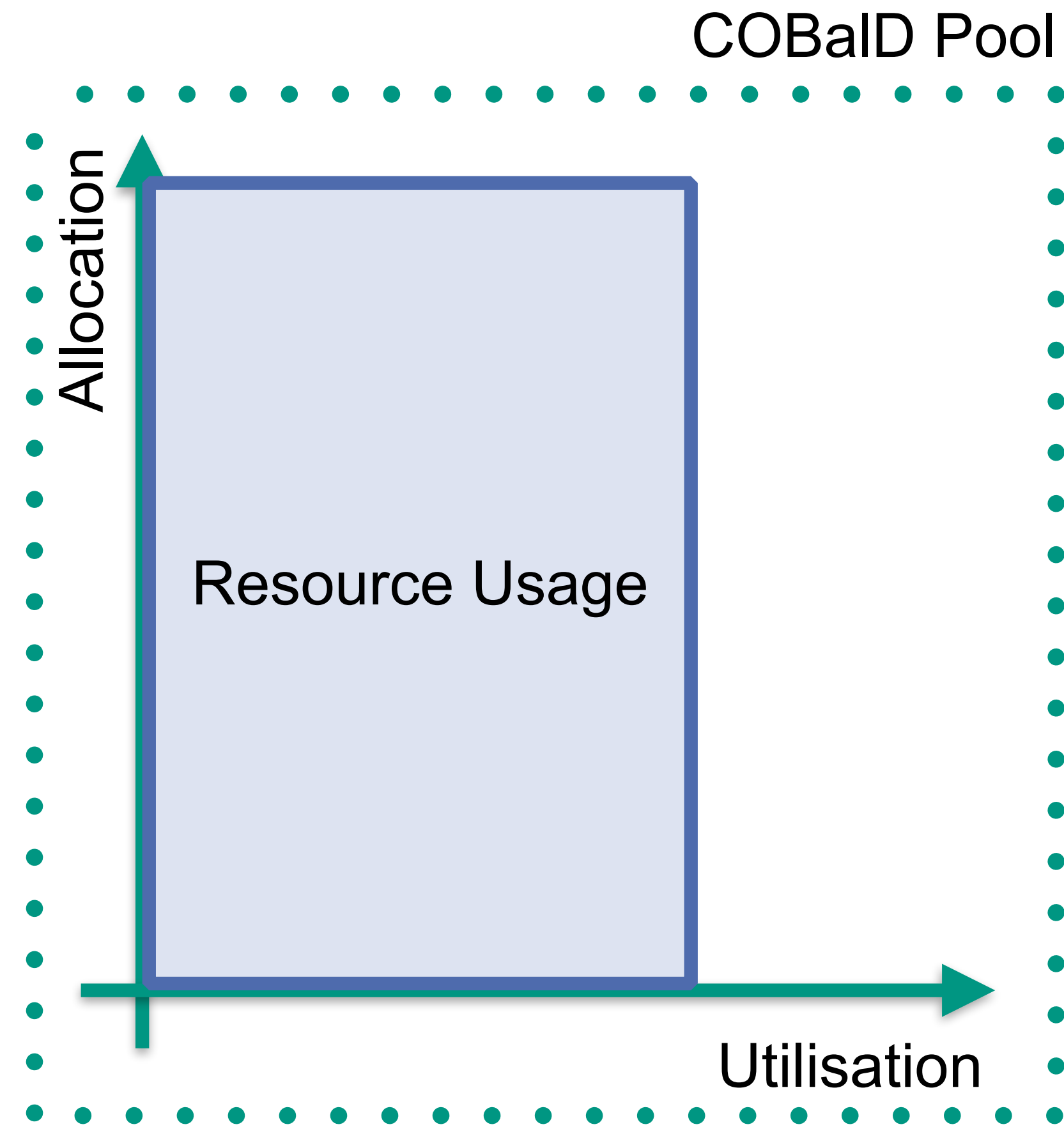
COBaID Resource Pool Model



COBaID Resource Pool Model



COBaID Resource Pool Model



COBaID Resource Pool Model

```
if utilisation < self.low_utilisation:  
    return supply * self.low_scale  
elif allocation > self.high_allocation:  
    return supply * self.high_scale
```

