Karlsruhe Institute of Technology

Dynamic Federation of Heterogeneous Compute Resources in High Energy Physics & Beyond

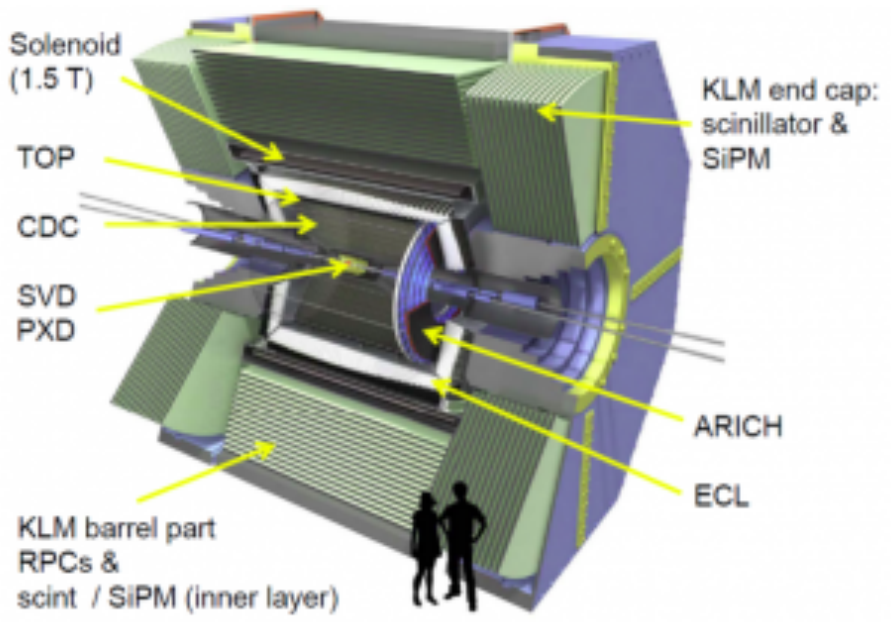
MU Days @ KIT, 14.09.2023

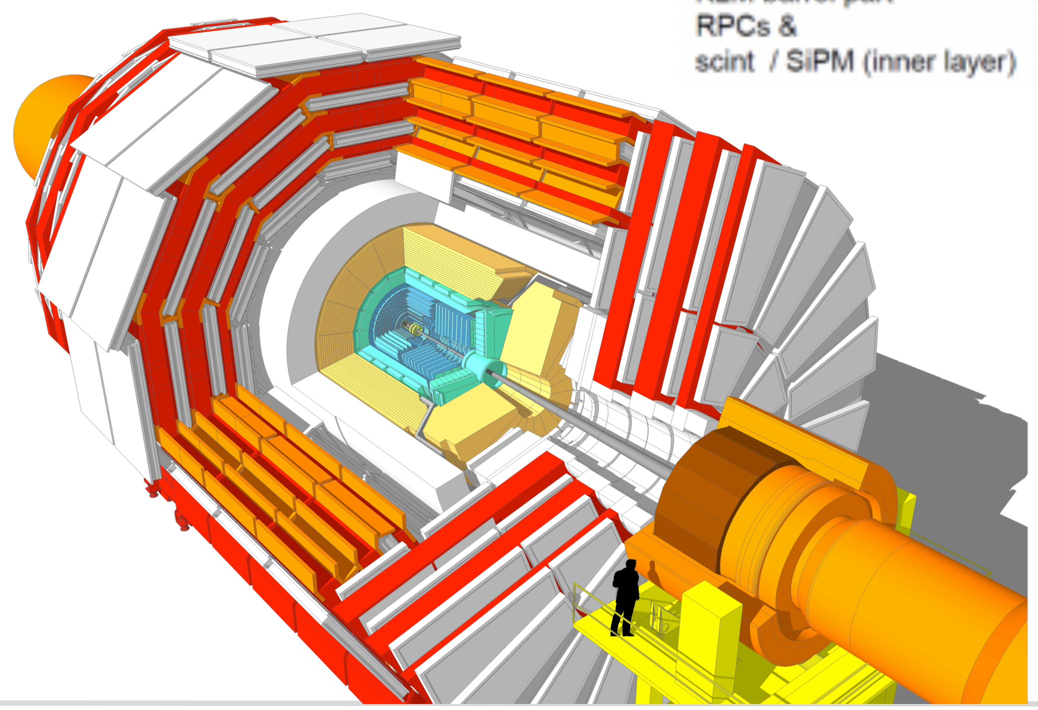
Manuel Giffels

The High Energy Physics Workflow

- Particle detectors record physics event data
- Each detector used by a collaboration of scientists

+ MC Simulation

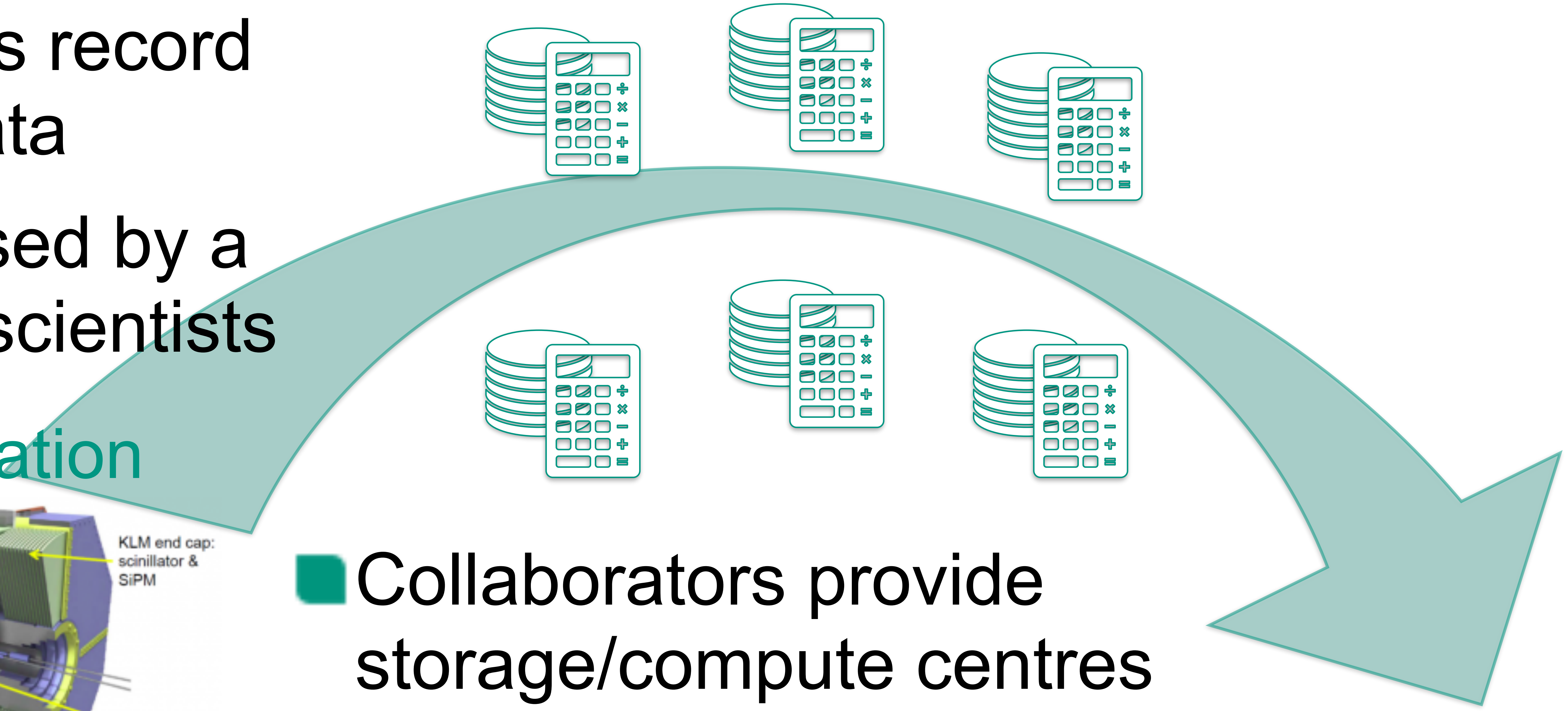


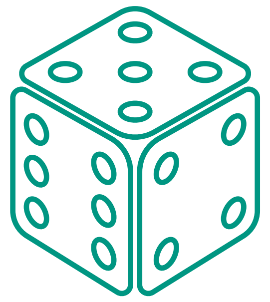


Base on a slide by Max Fischer

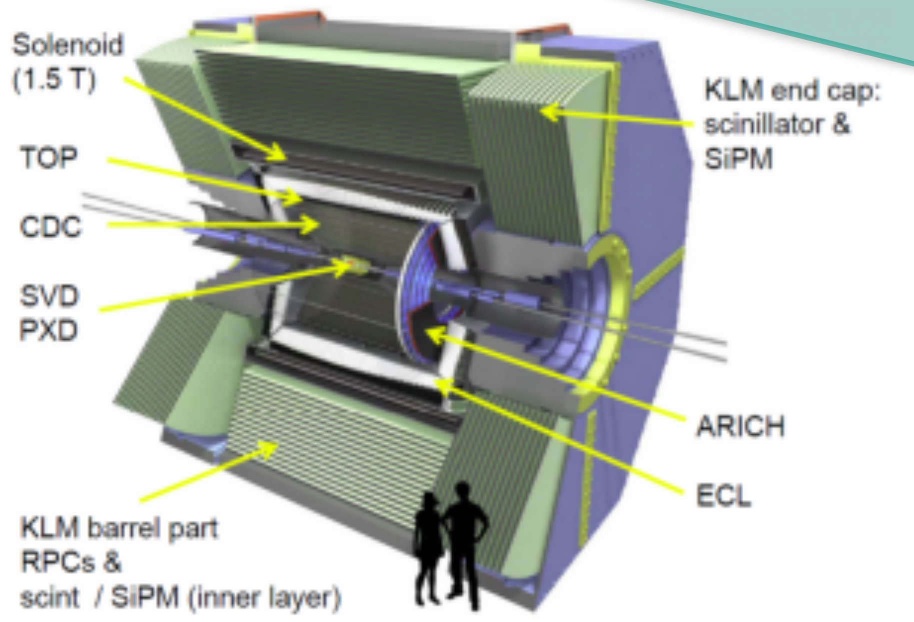
The High Energy Physics Workflow

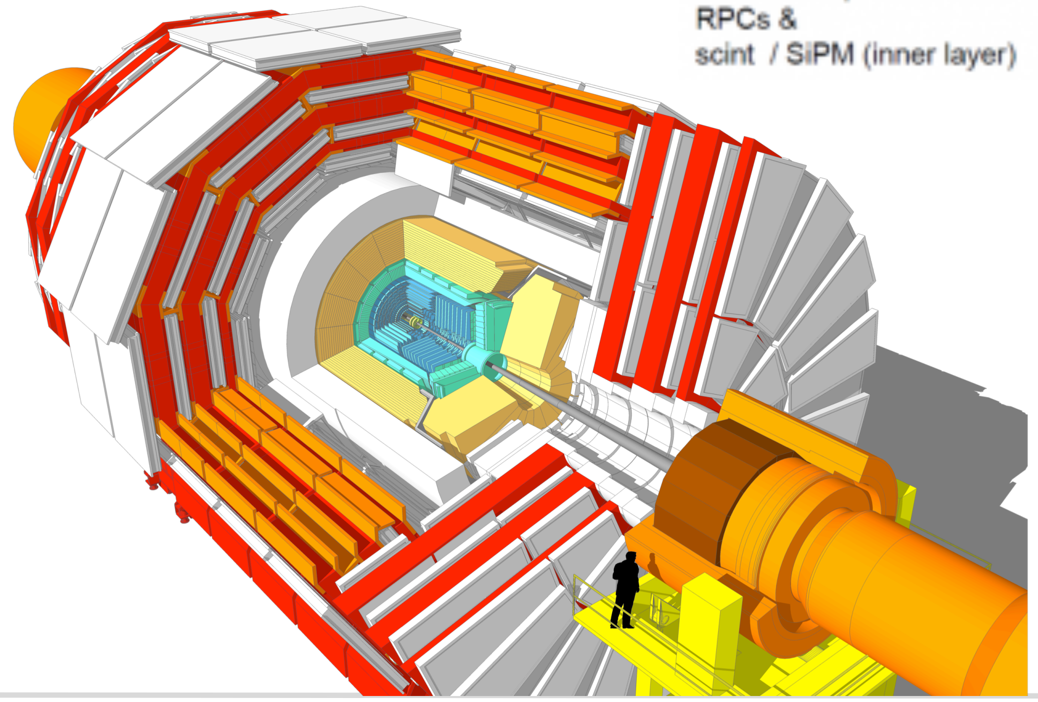
- Particle detectors record physics event data
- Each detector used by a collaboration of scientists





+ MC Simulation





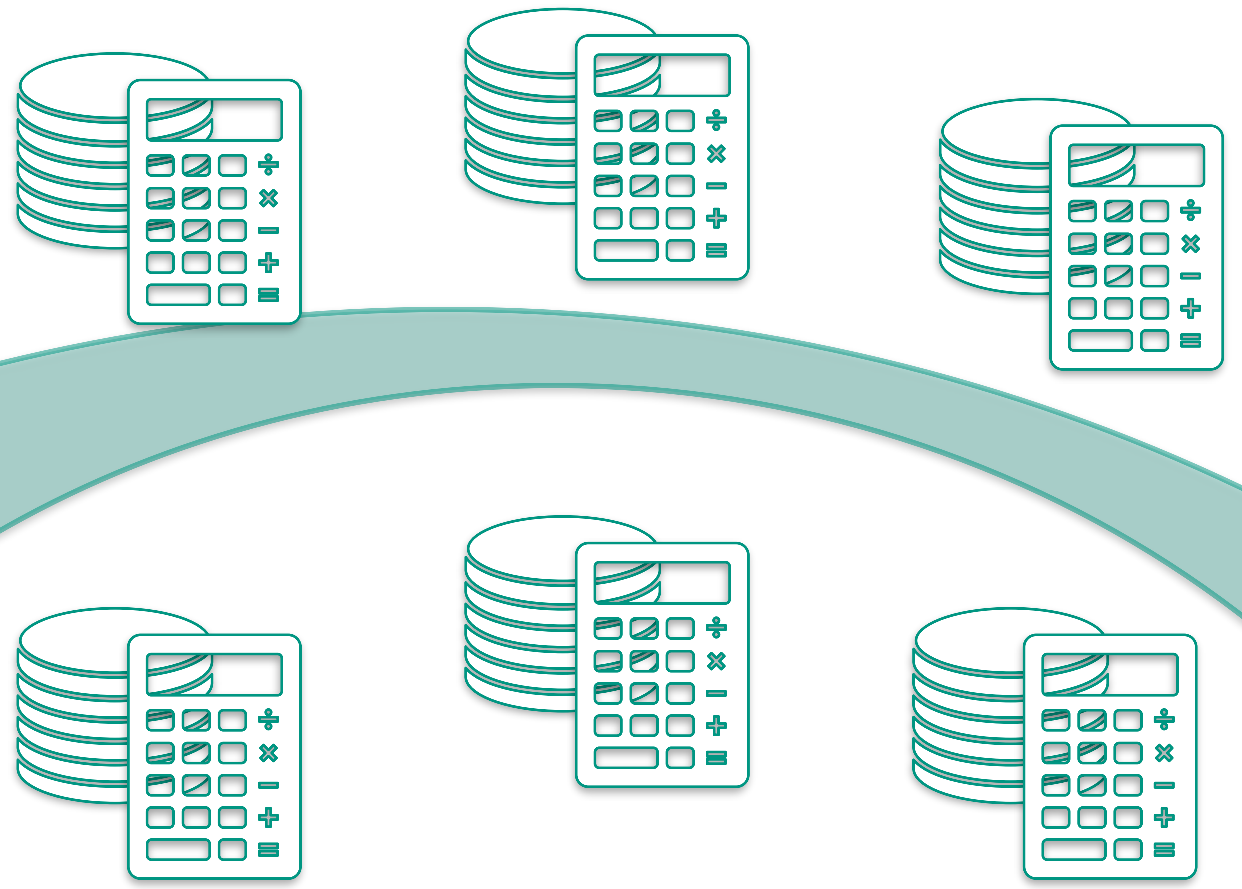
- Collaborators provide storage/compute centres
- Resources shared via a worldwide computing Grid

Base on a slide by Max Fischer

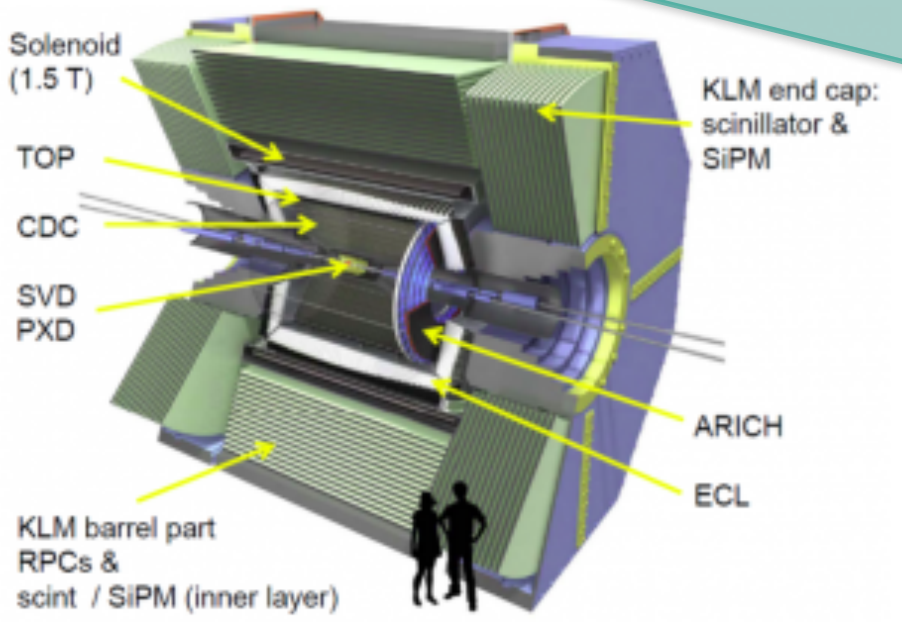
The High Energy Physics Workflow

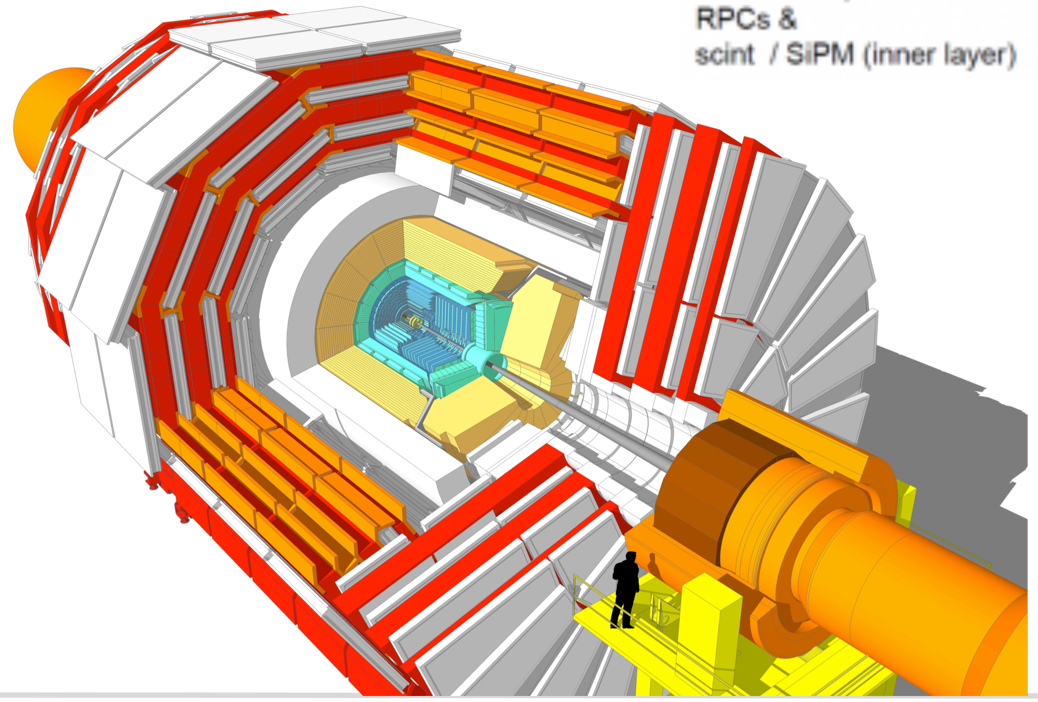
- Particle detectors record physics event data
- Each detector used by a collaboration of scientists

\+ MC Simulation

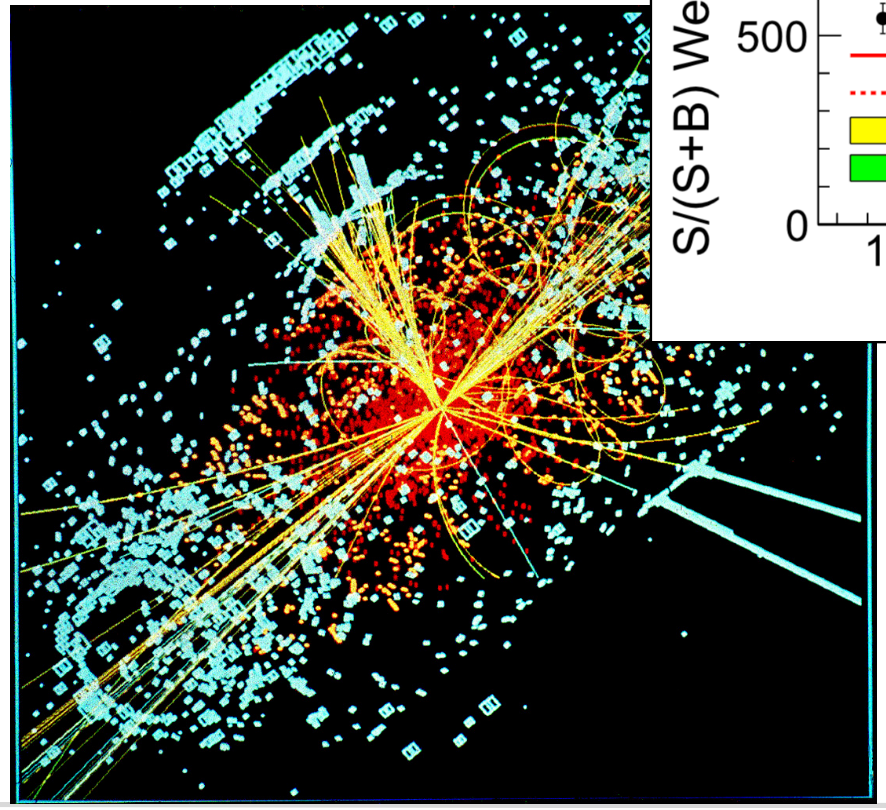


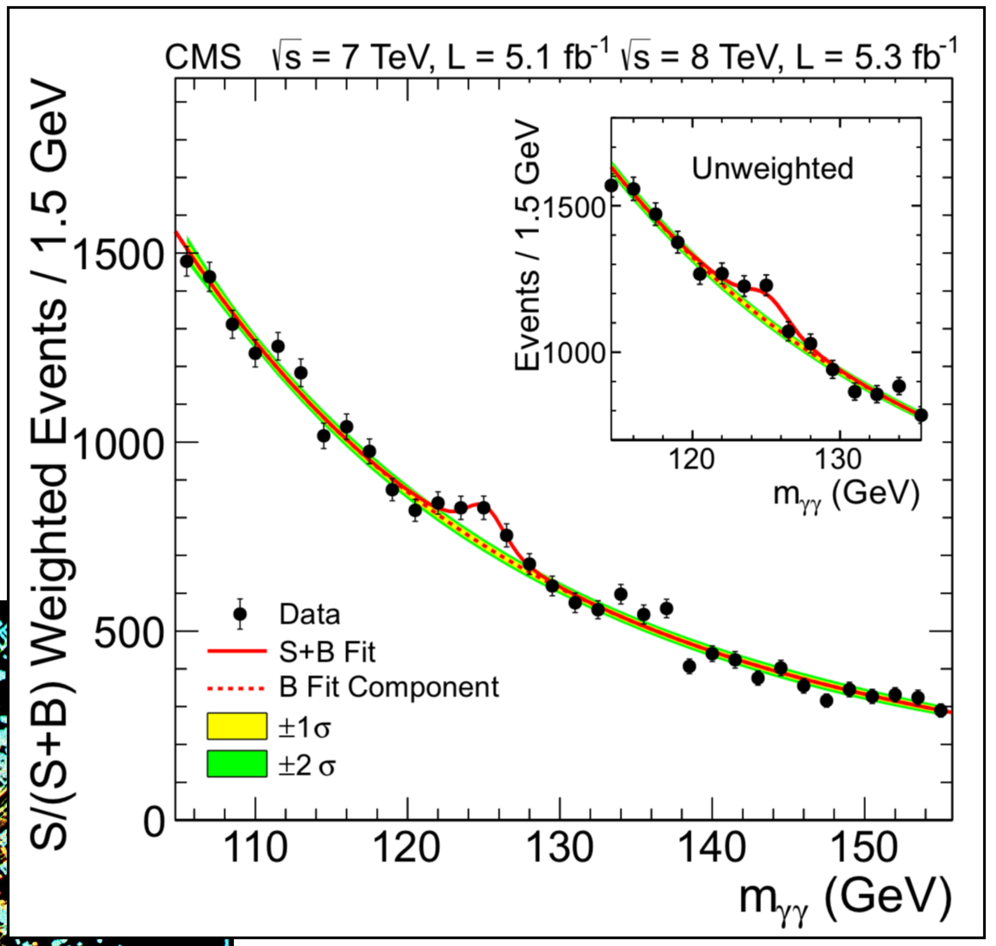
- Collaborations automate common pre-processing
- Scientists run individual end-user analyses





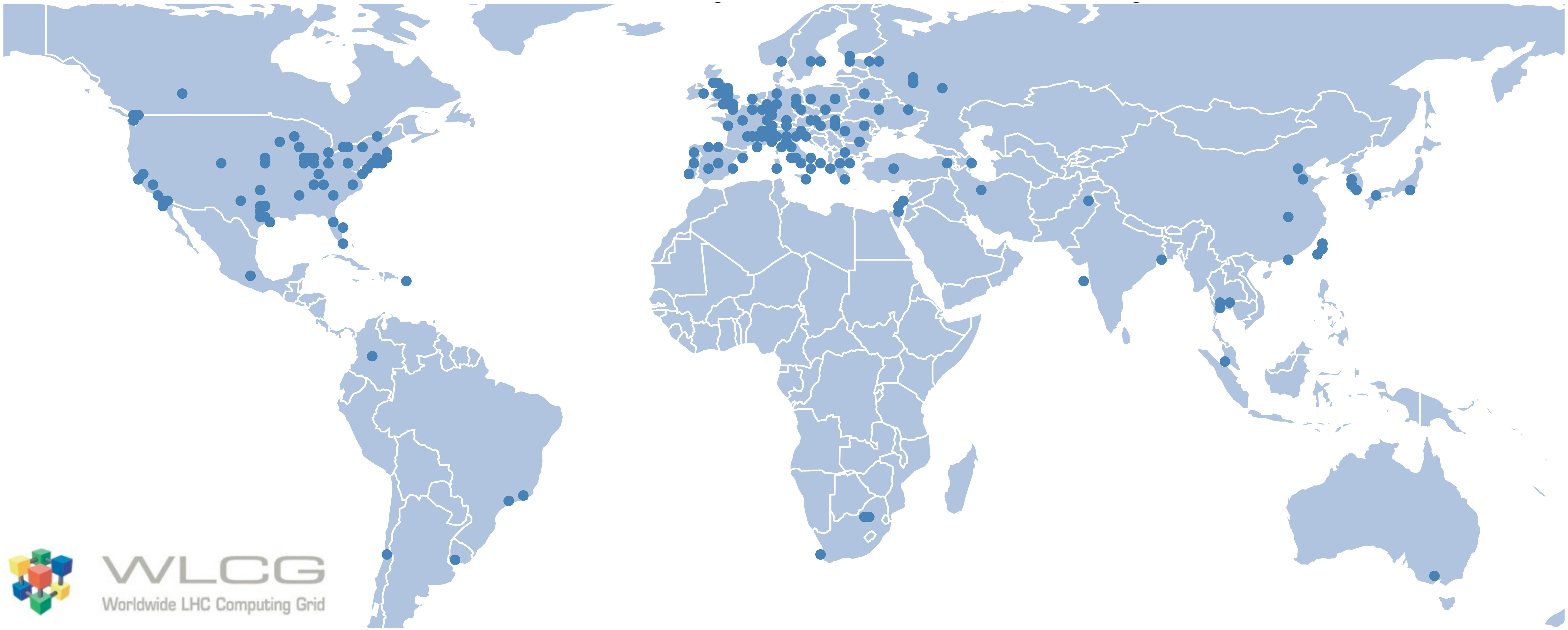
- Collaborators provide storage/compute centres
- Resources shared via a worldwide computing Grid





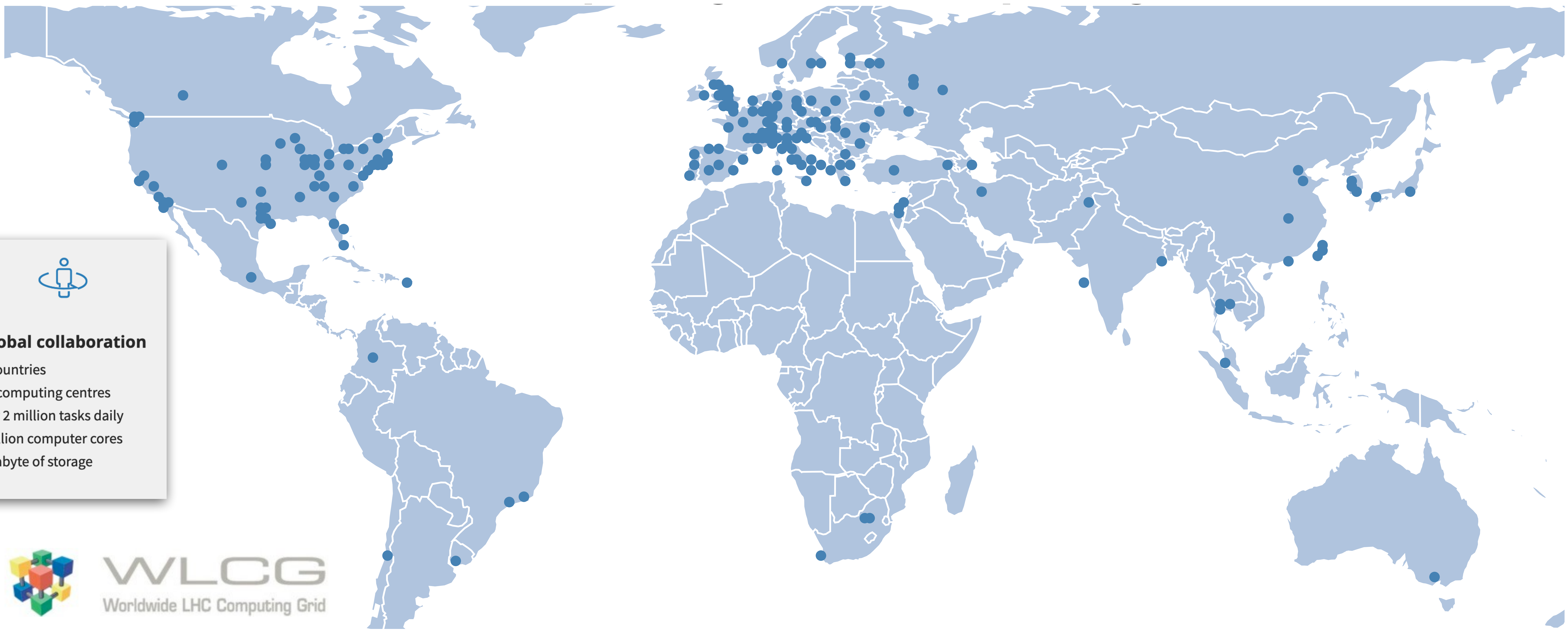
Base on a slide by Max Fischer

Computing in High Energy Physics (HEP)



20+ years experience

Computing in High Energy Physics (HEP)



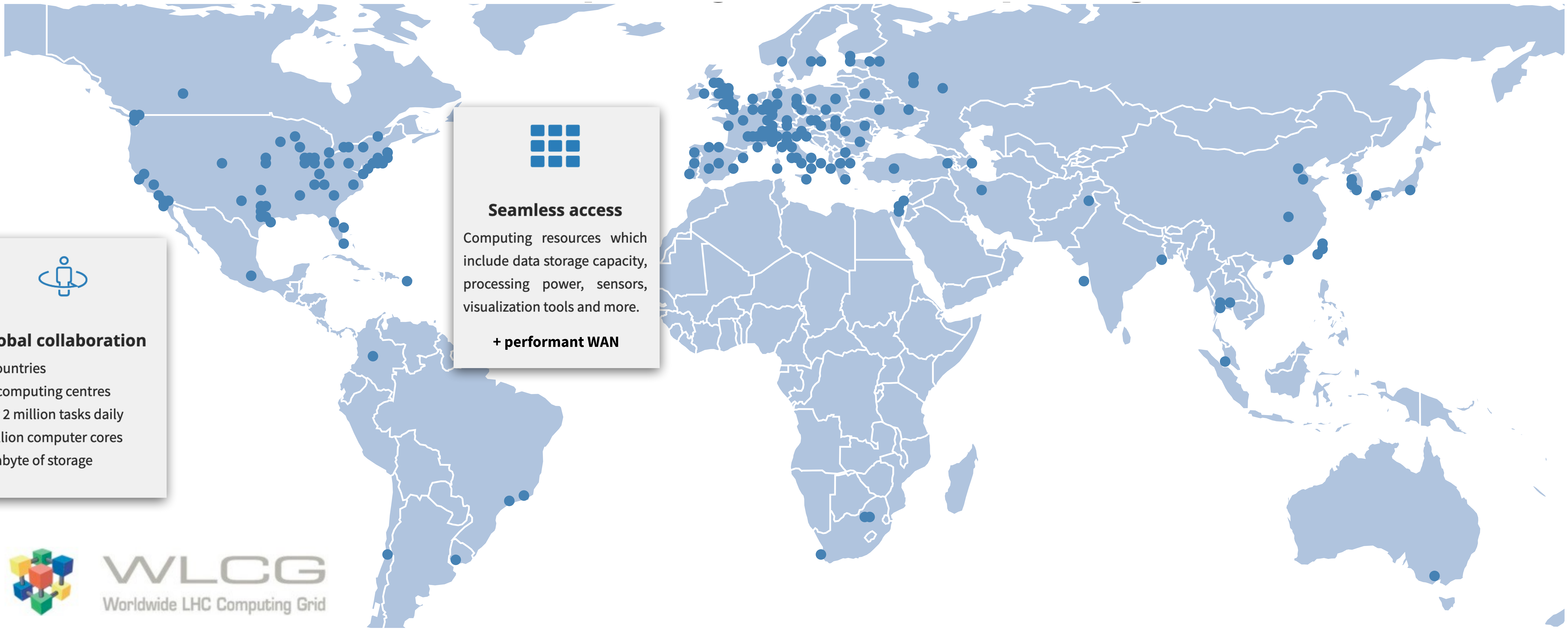
Global collaboration

42 countries
170 computing centres
Over 2 million tasks daily
1 million computer cores
1 exabyte of storage

WLCG
Worldwide LHC Computing Grid

20+ years experience

Computing in High Energy Physics (HEP)



Seamless access

Computing resources which include data storage capacity, processing power, sensors, visualization tools and more.

+ performant WAN

Global collaboration

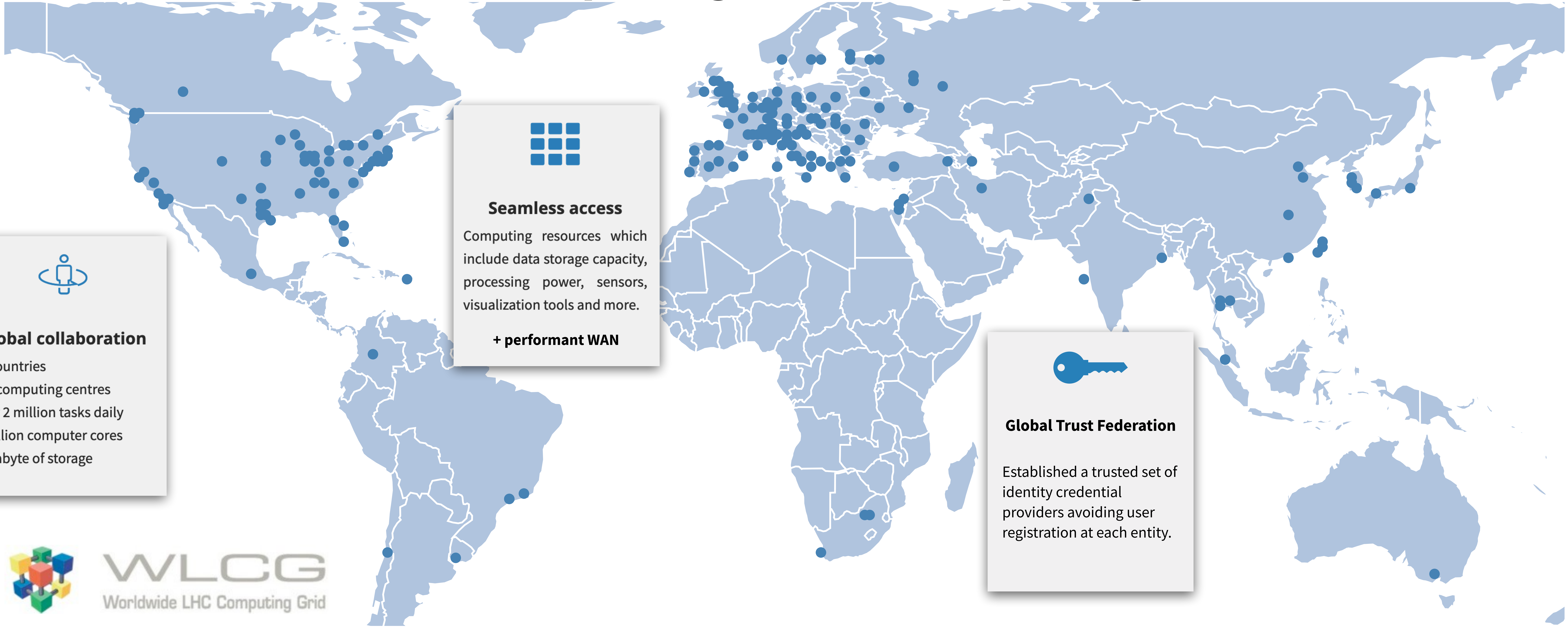
42 countries
170 computing centres
Over 2 million tasks daily
1 million computer cores
1 exabyte of storage

WLCG
Worldwide LHC Computing Grid

20+ years experience

Computing in High Energy Physics (HEP)

Karlsruhe Institute of Technology



Global collaboration

42 countries
170 computing centres
Over 2 million tasks daily
1 million computer cores
1 exabyte of storage

Seamless access

Computing resources which include data storage capacity, processing power, sensors, visualization tools and more.

+ performant WAN

Global Trust Federation

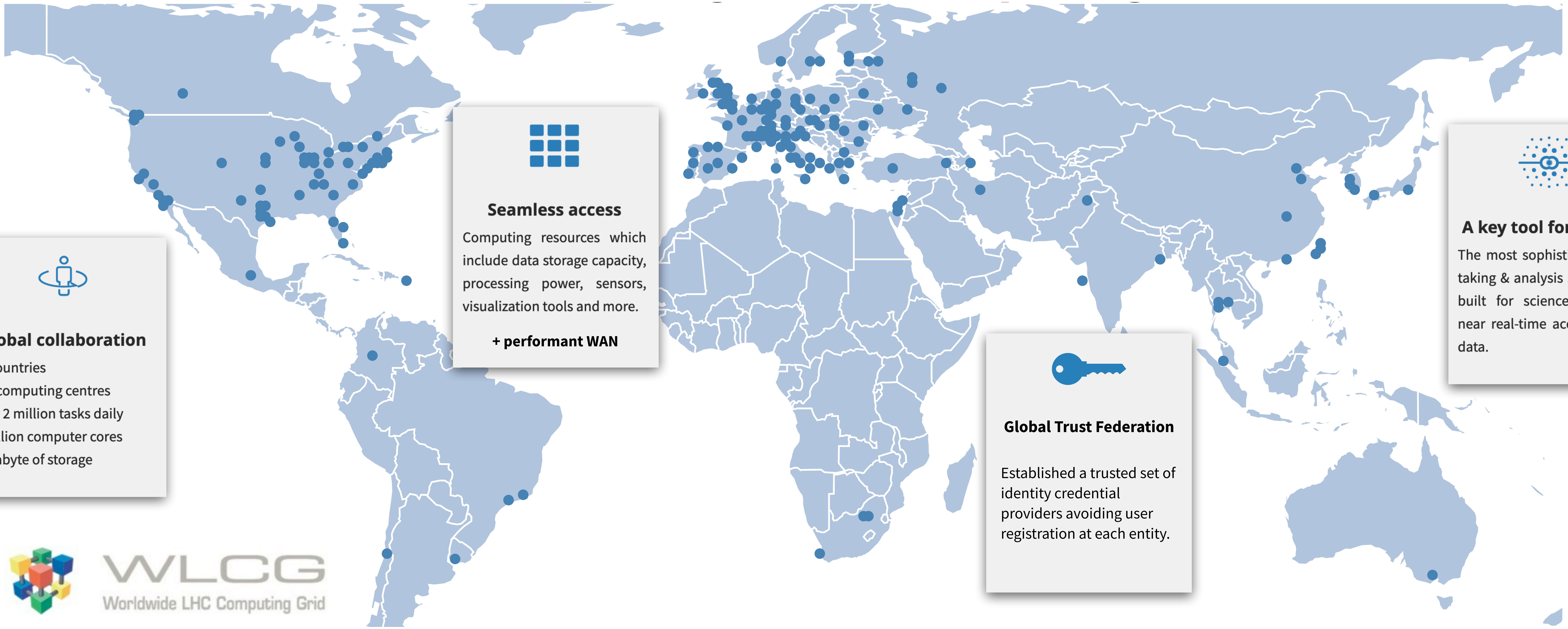
Established a trusted set of identity credential providers avoiding user registration at each entity.

WLCG
Worldwide LHC Computing Grid

20+ years experience

Computing in High Energy Physics (HEP)

Karlsruhe Institute of Technology



Global collaboration

42 countries
170 computing centres
Over 2 million tasks daily
1 million computer cores
1 exabyte of storage

Seamless access

Computing resources which include data storage capacity, processing power, sensors, visualization tools and more.

+ performant WAN

Global Trust Federation

Established a trusted set of identity credential providers avoiding user registration at each entity.

A key tool for physics

The most sophisticated data-taking & analysis system ever built for science, providing near real-time access to LHC data.

WLCG
Worldwide LHC Computing Grid

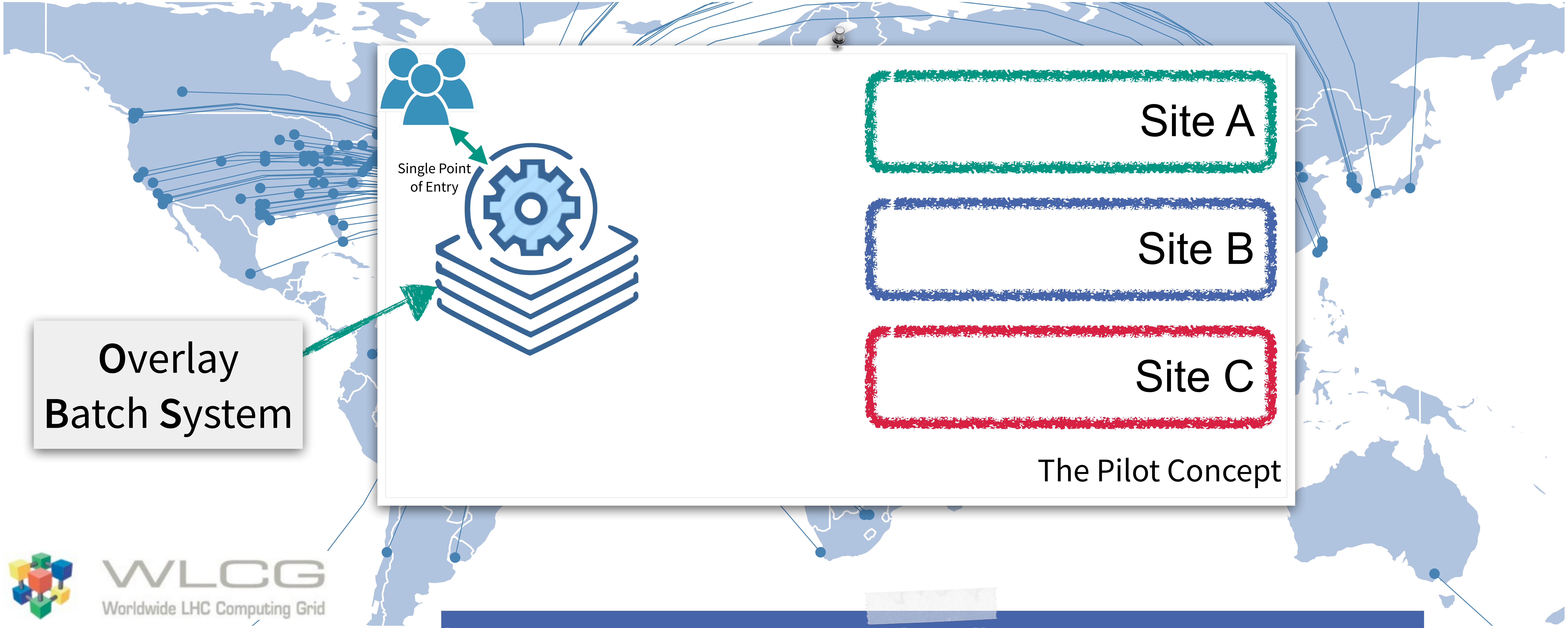
20+ years experience

From Grid towards Global Distributed Computing

Karlsruhe Institute of Technology

Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

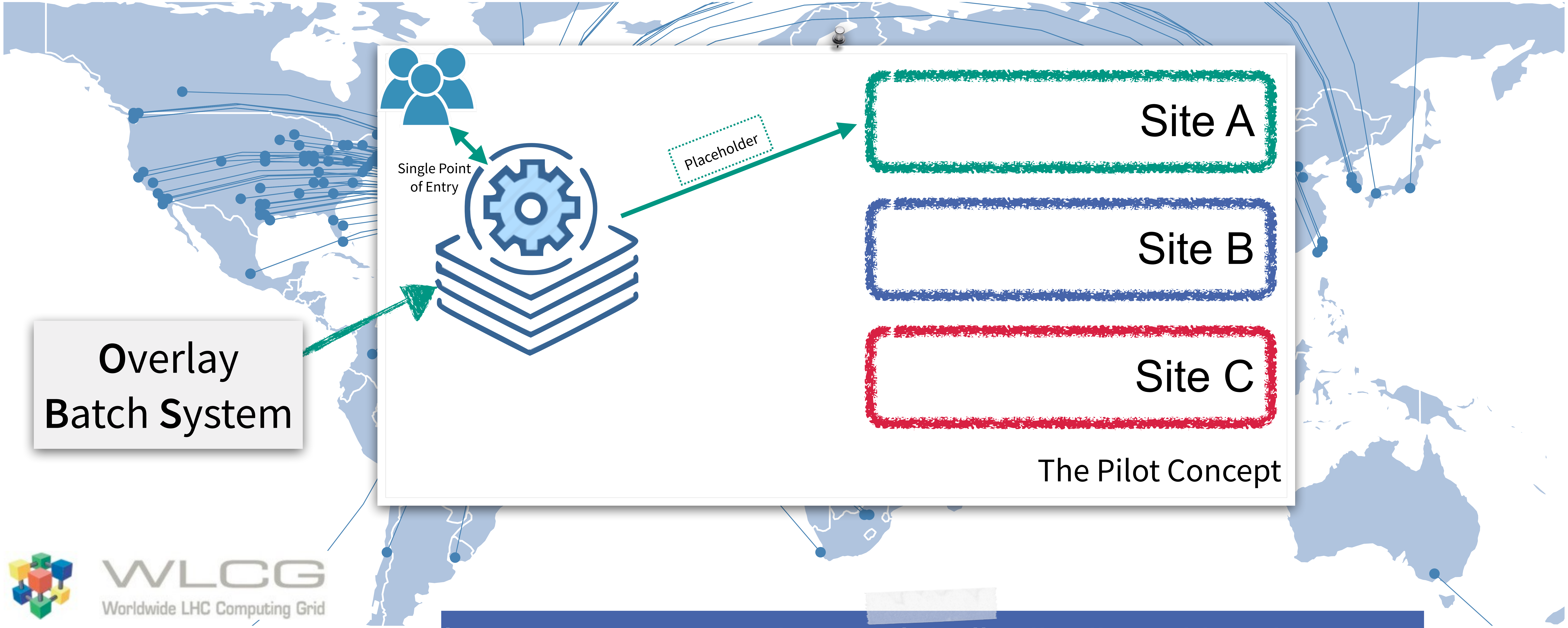
From Grid towards Global Distributed Computing



Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

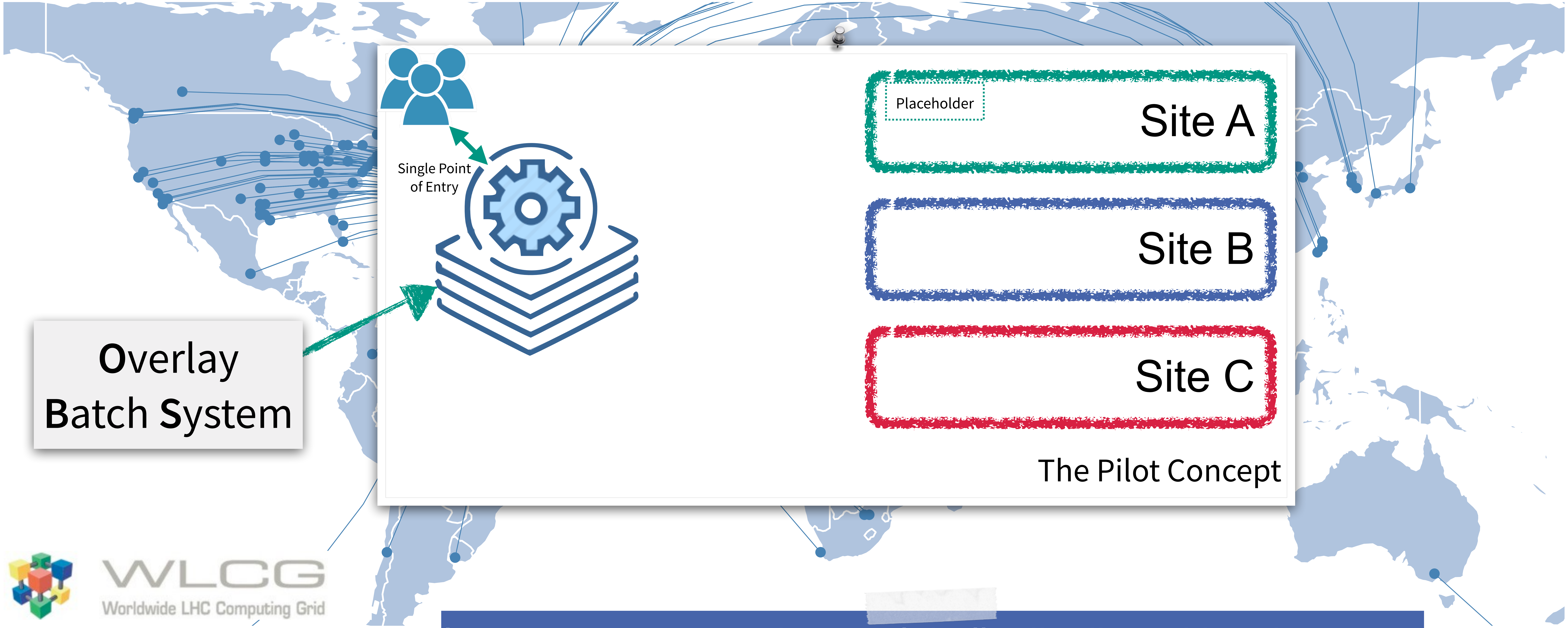
From Grid towards Global Distributed Computing

Karlsruhe Institute of Technology



Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

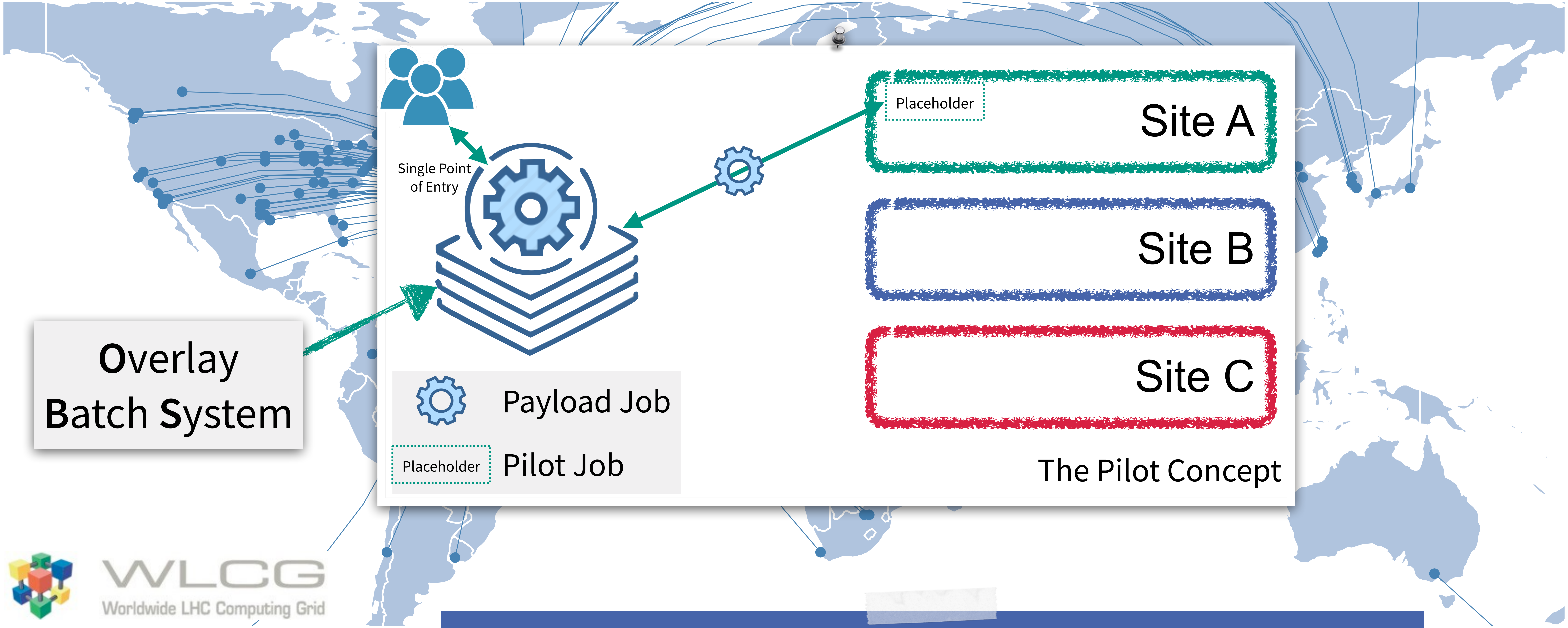
From Grid towards Global Distributed Computing



Worldwide LHC Computing Grid

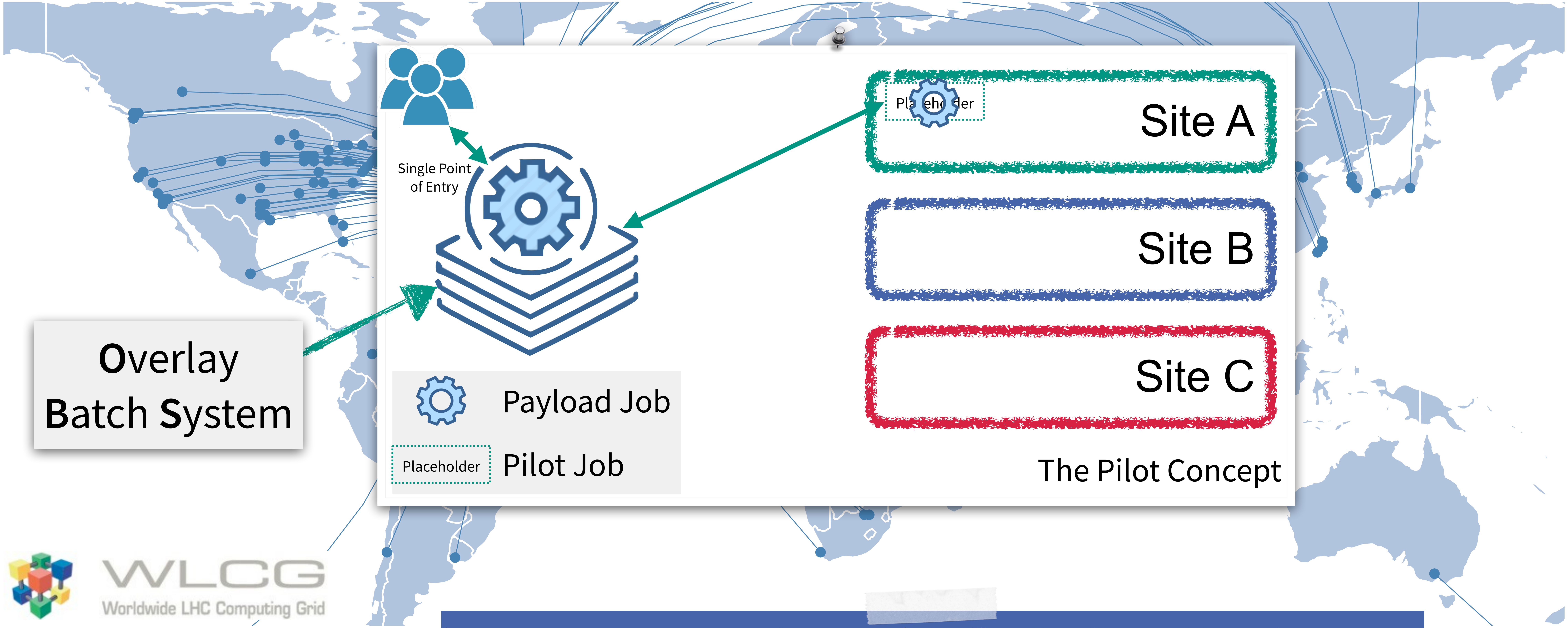
Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

From Grid towards Global Distributed Computing



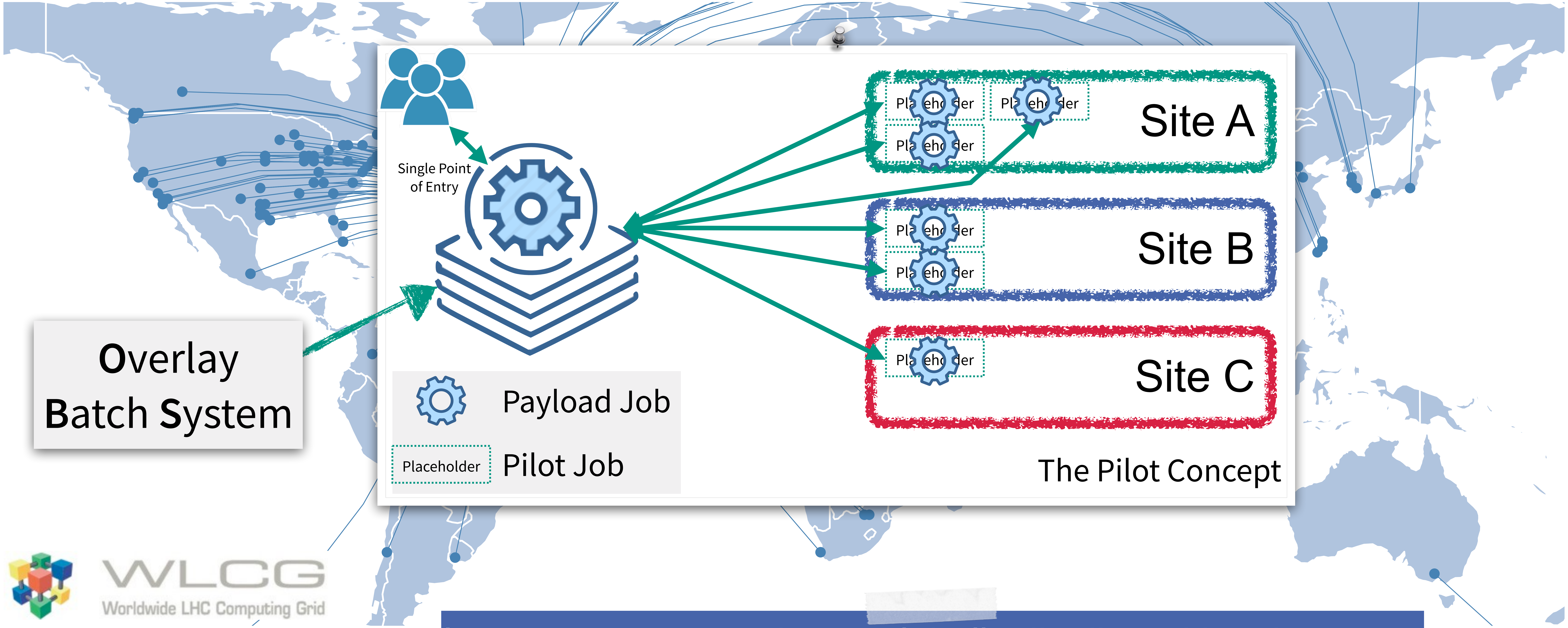
Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

From Grid towards Global Distributed Computing



Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

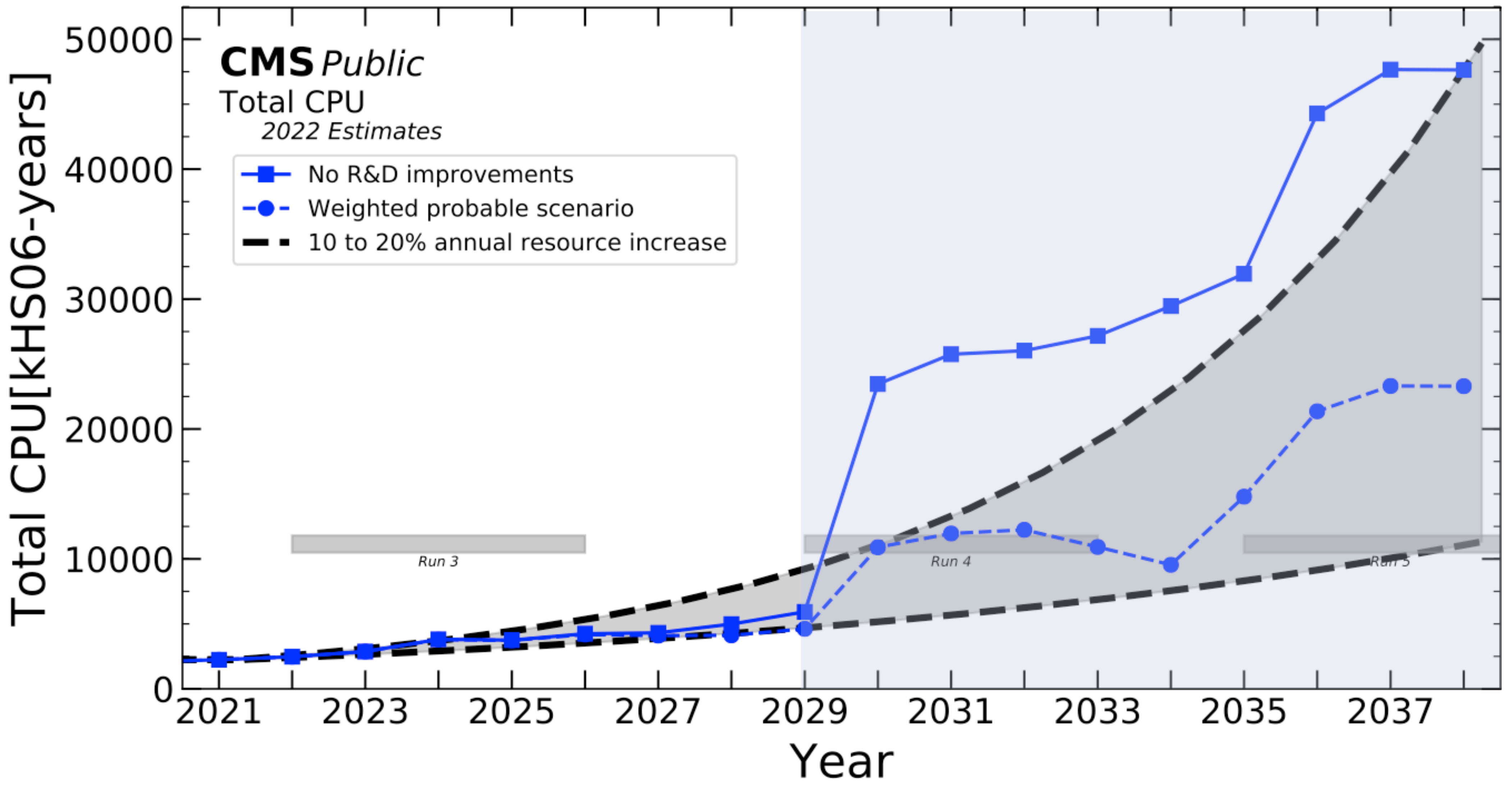
From Grid towards Global Distributed Computing

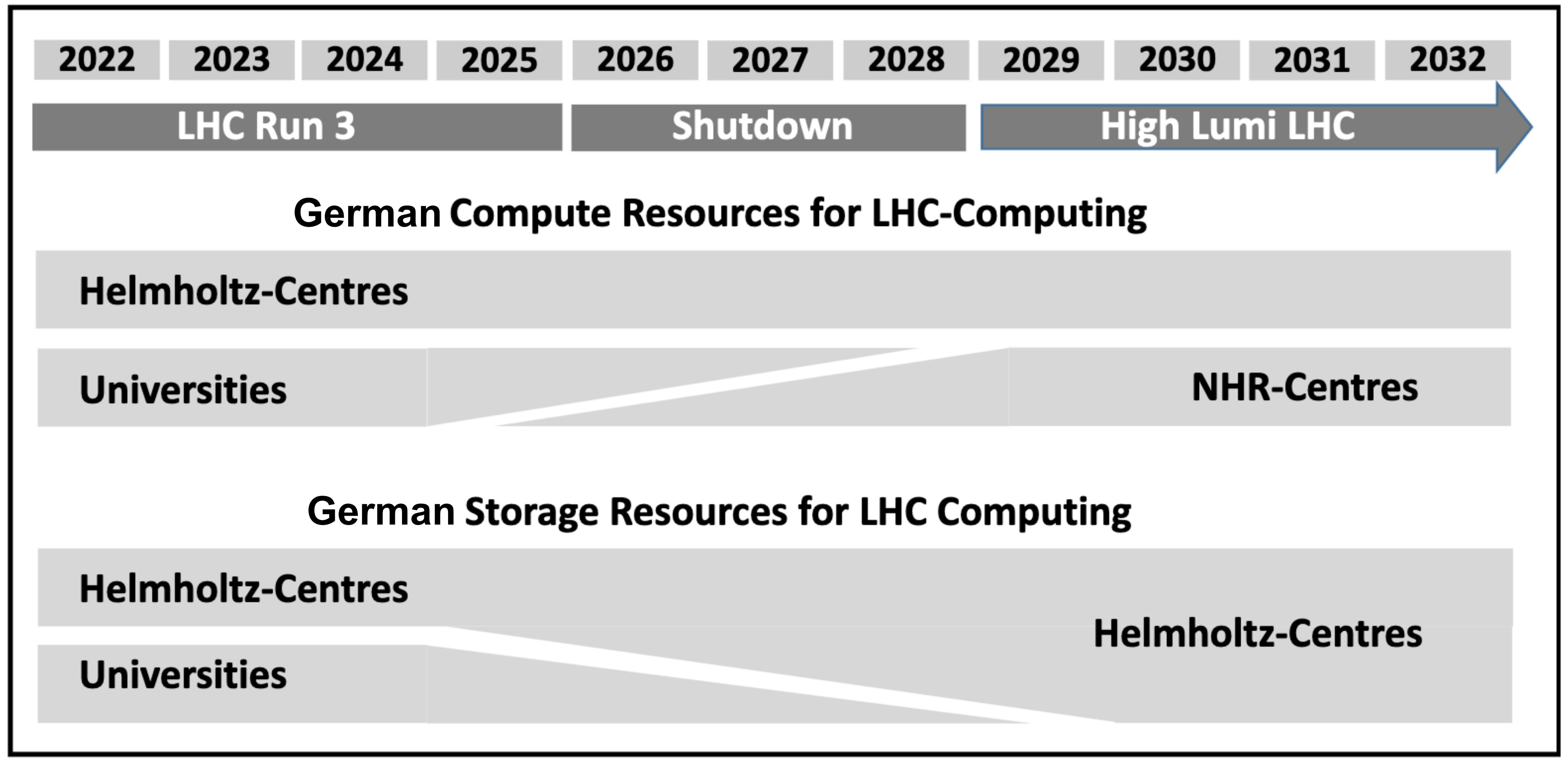


Integrate resources into a globally distributed batch system and remove some parts of the initial Grid middleware

Upcoming Computing Challenges in HEP & Beyond

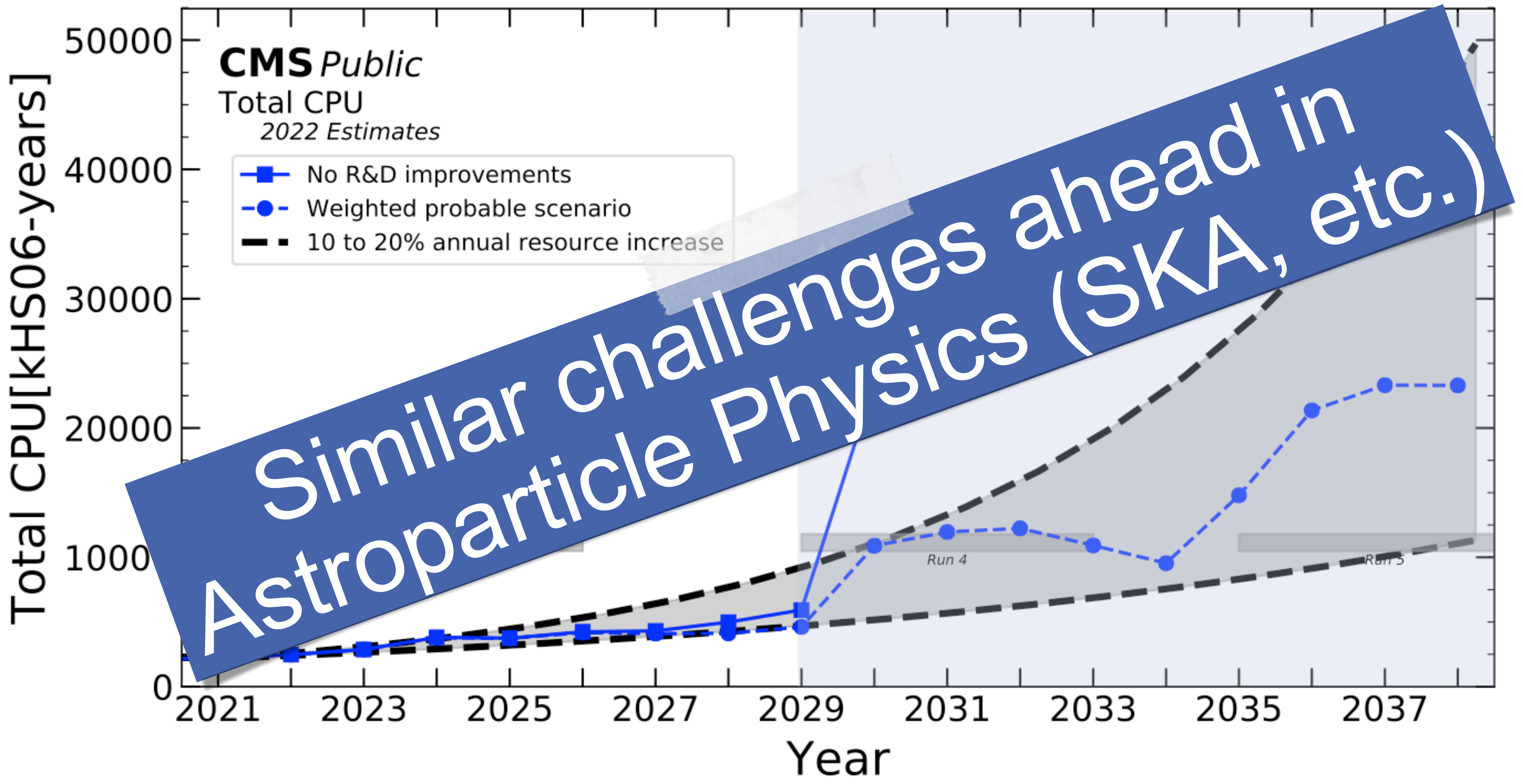
- HL-LHC poses unprecedented challenges to HEP computing
- Assuming flat budget and 10-20% technology advance per year
- Needs major invests in Software & Computing Model Evolution (R&D)
  - Utilize non HEP-dedicated and non Grid-enabled (opportunistic) compute resources (Institute clusters, HPCs, Clouds, etc.)
  - Transition of German University WLCG resource provisioning towards NHR HPCs (Compute) and Helmholtz (Storage)

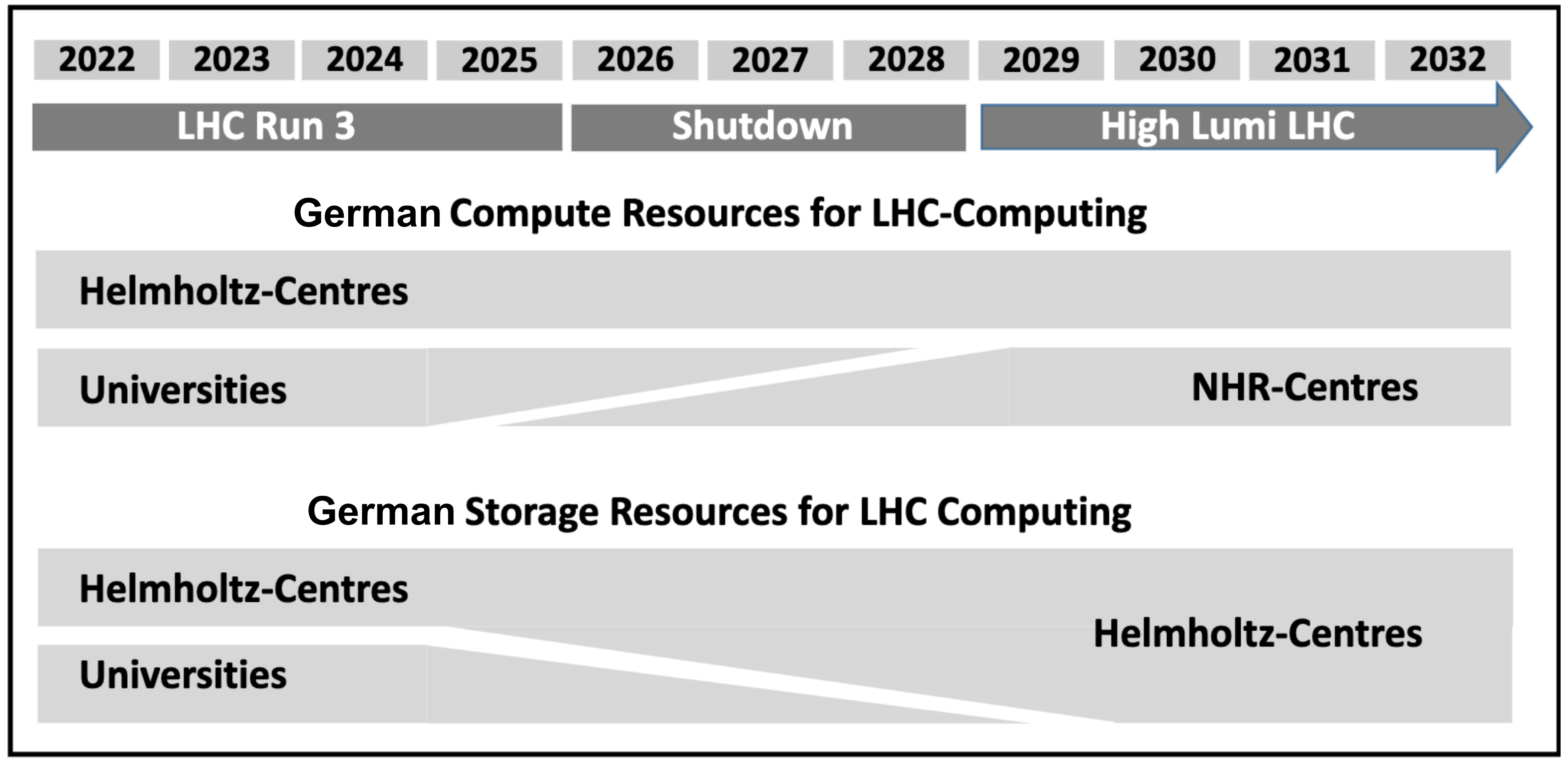




Upcoming Computing Challenges in HEP & Beyond

- HL-LHC poses unprecedented challenges to HEP computing
- Assuming flat budget and 10-20% technology advance per year
- Needs major invests in Software & Computing Model Evolution (R&D)
  - Utilize non HEP-dedicated and non Grid-enabled (opportunistic) compute resources (Institute clusters, HPCs, Clouds, etc.)
  - Transition of German University WLCG resource provisioning towards NHR HPCs (Compute) and Helmholtz (Storage)

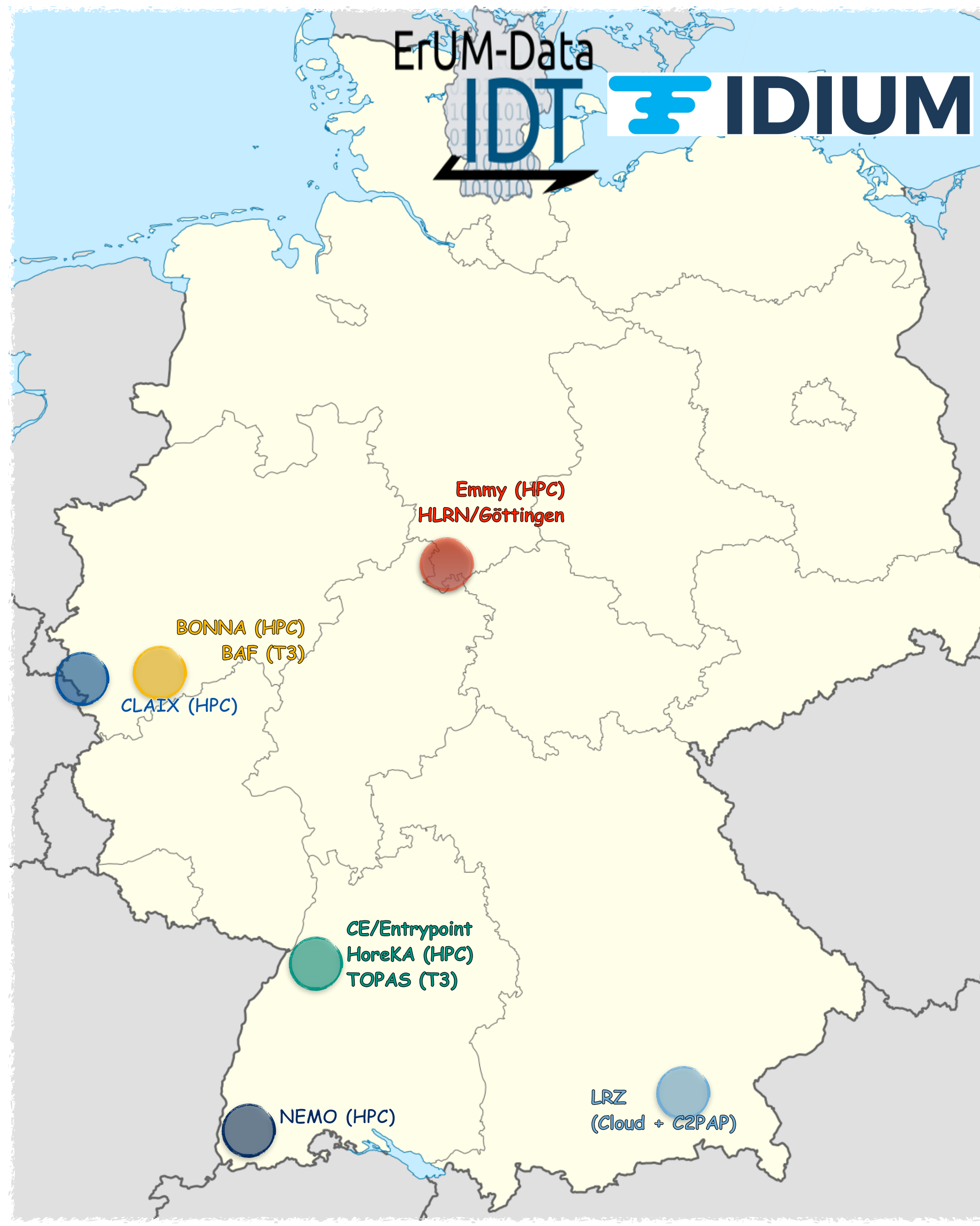




Opportunistic Resources & WLCG

Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.

+ Clouds



Resource providers distributed all-over Germany

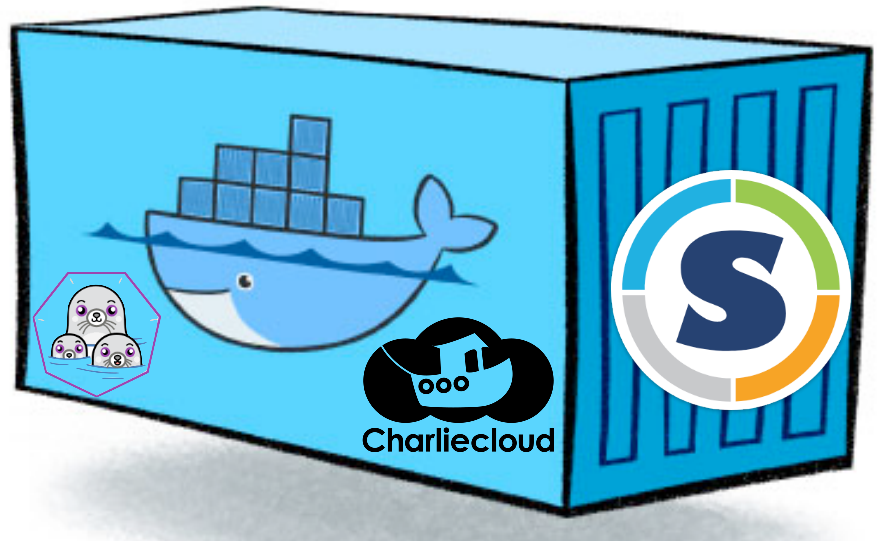
Opportunistic Resources & WLCG

Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.

+ Clouds

Challenges:

- Different OS & Software availability



Container Technology

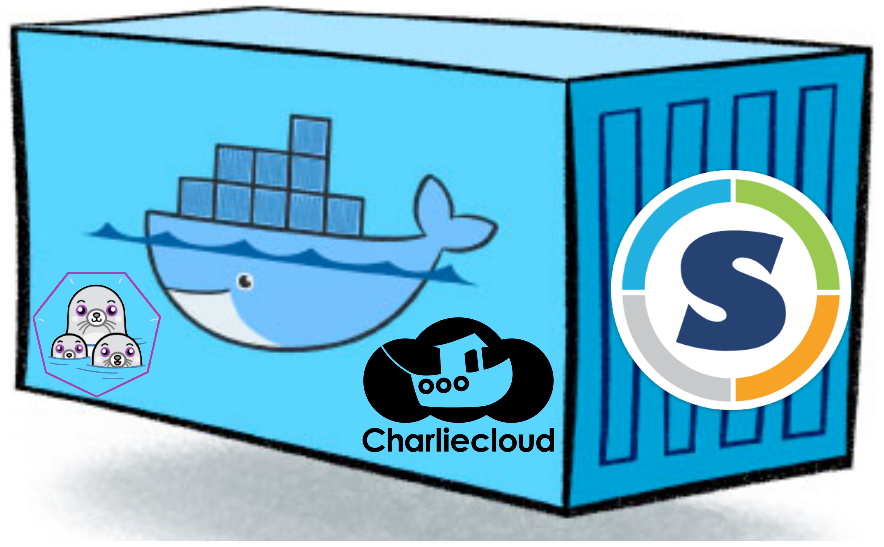
Opportunistic Resources & WLCG

Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.

+ Clouds

Challenges:

- Different OS & Software availability
- Very heterogenous systems, not all resources are suited for all tasks
- Varying availability of and demand for those resources



Container Technology

COBalD
TARDIS
in-house development

Resource Scheduler

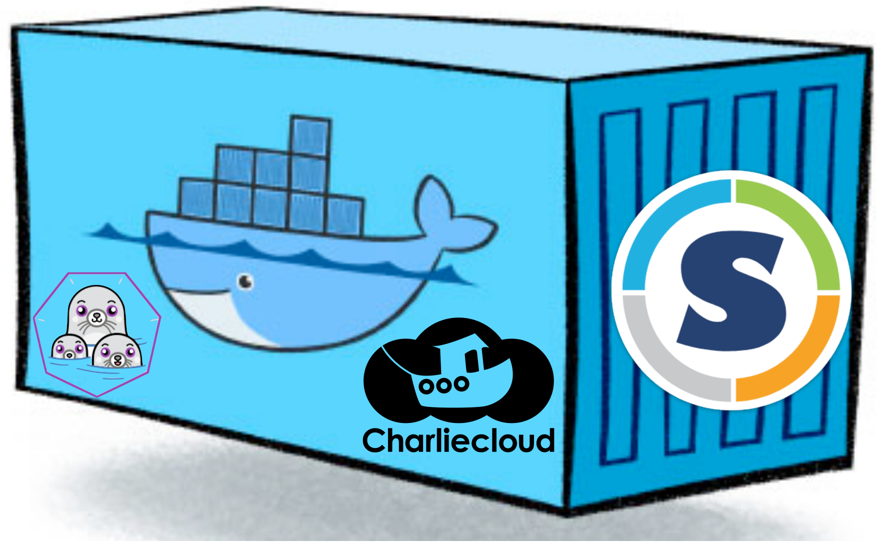
Opportunistic Resources & WLCG

Karlsruhe Institute of Technology

Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.

Challenges:

- Different OS & Software availability
- Very heterogenous systems, not all resources are suited for all tasks
- Varying availability of and demand for those resources
- No global trust federation/Grid entry point available



Container Technology

+

Resource Scheduler

COBalD

TARDIS

in-house development

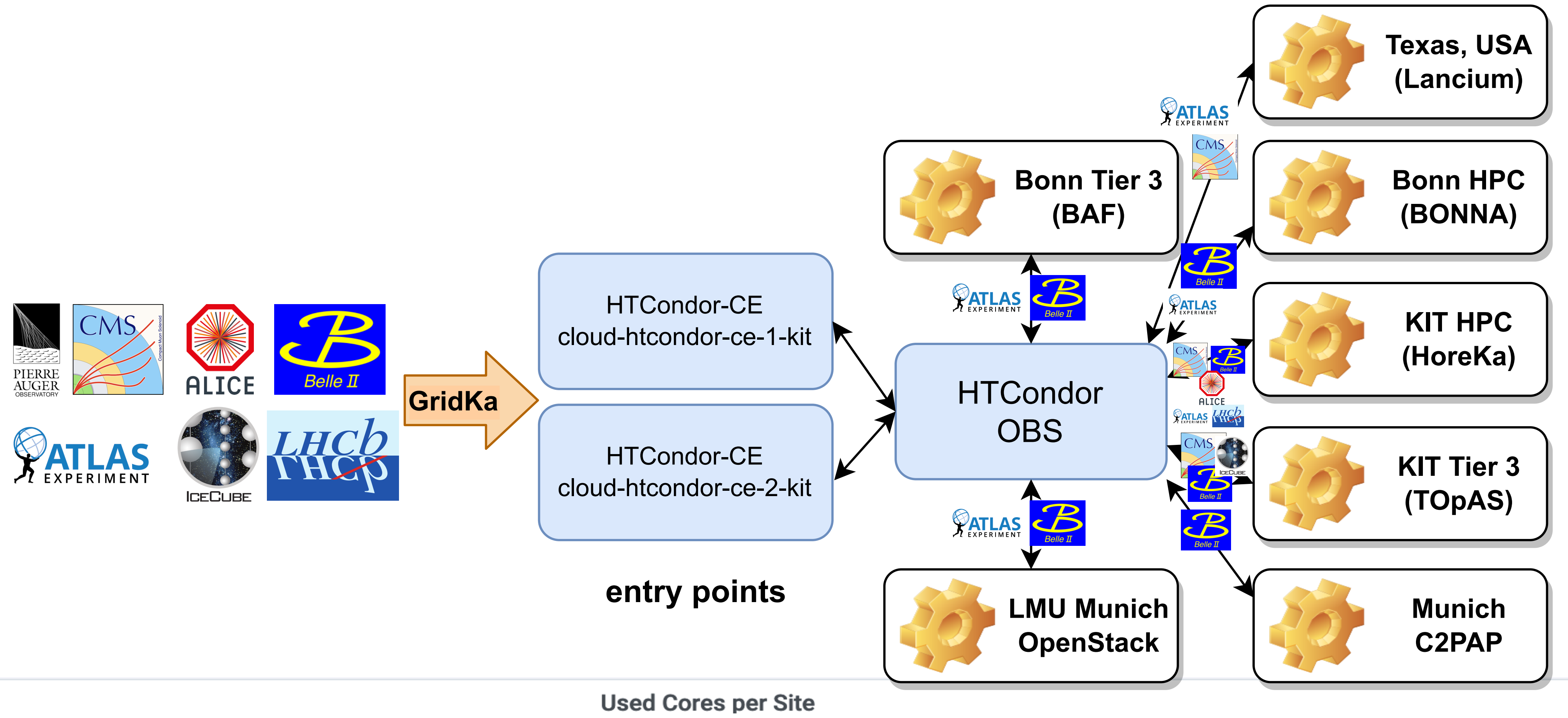
+

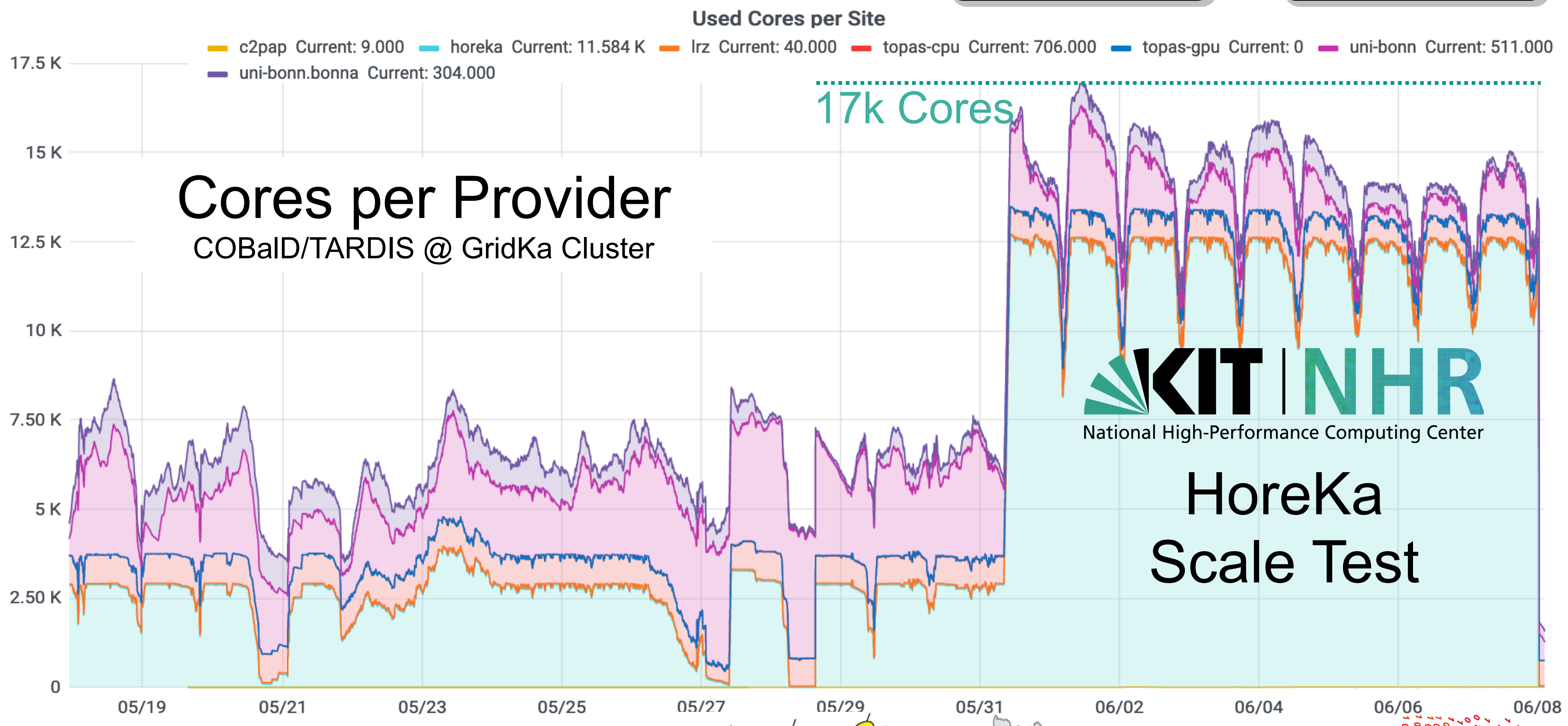
OBS + Pilots
in more generalized way

Opportunistic Compute @ GridKa in a Nutshell

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

- Dynamic, transparent and on-demand integration via COBalD/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 deployment across HEP institutes & HPC resources coordinated by KIT/GridKa
- Central building block of the Compute4PUNCH infrastructure within PUNCH4NFDI

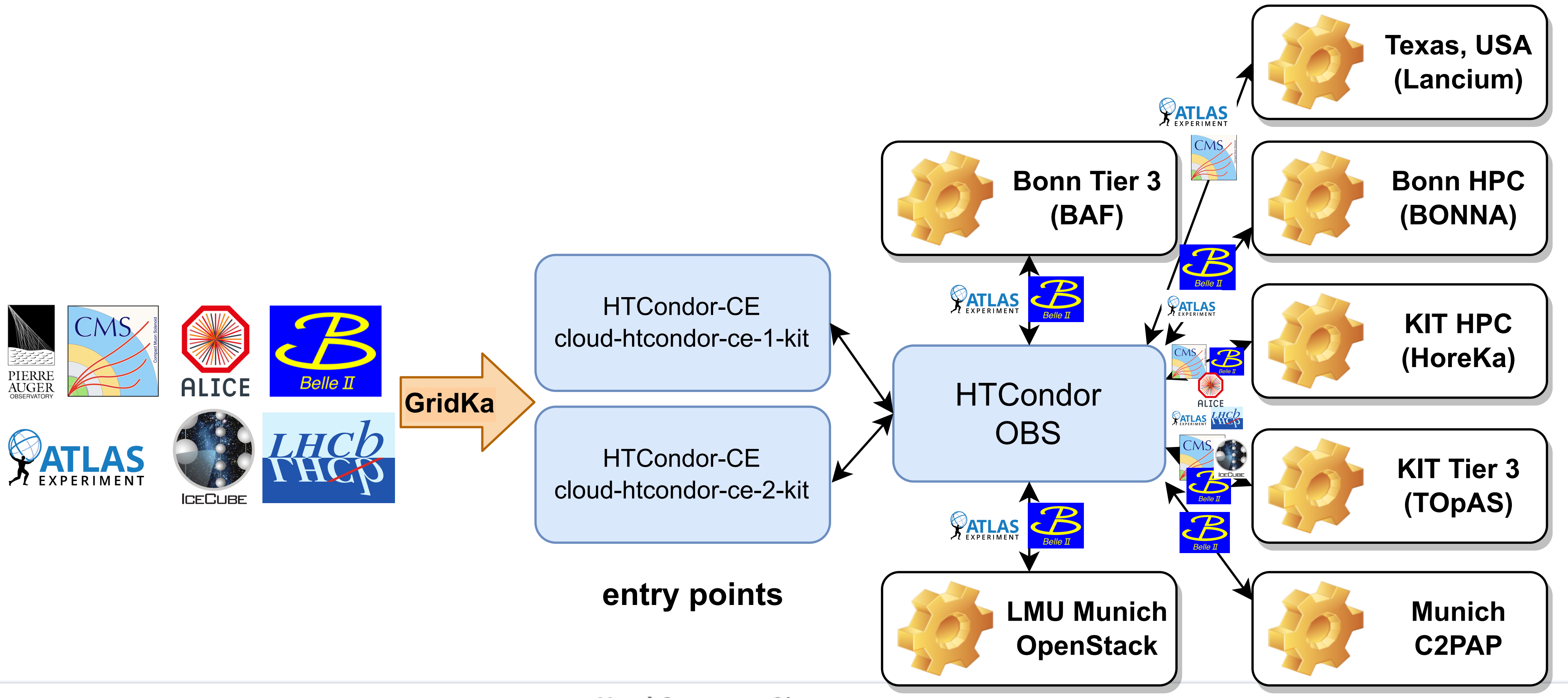


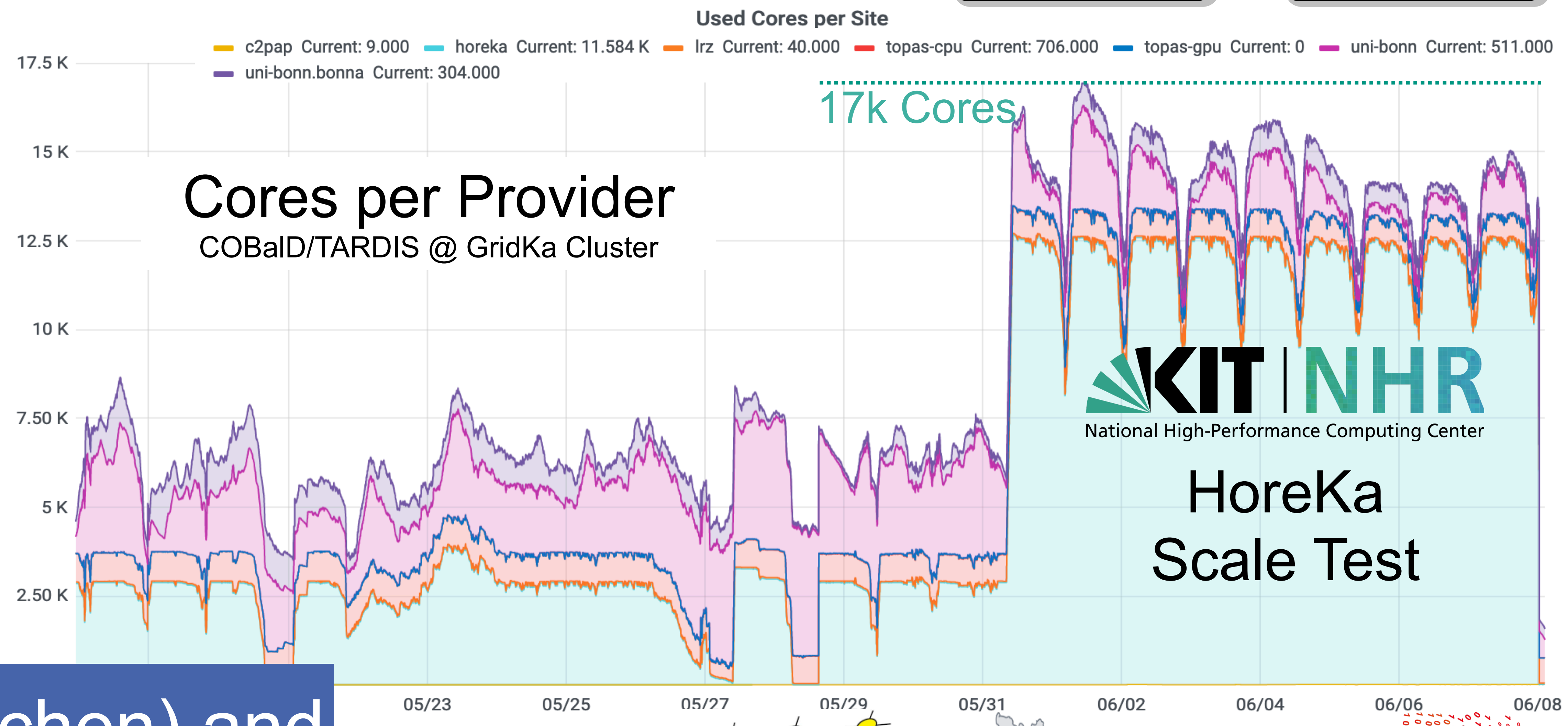


Opportunistic Compute @ GridKa in a Nutshell

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

- Dynamic, transparent and on-demand integration via COBalD/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 deployment across HEP institutes & HPC resources coordinated by KIT/GridKa
- Central building block of the Compute4PUNCH infrastructure within PUNCH4NFDI



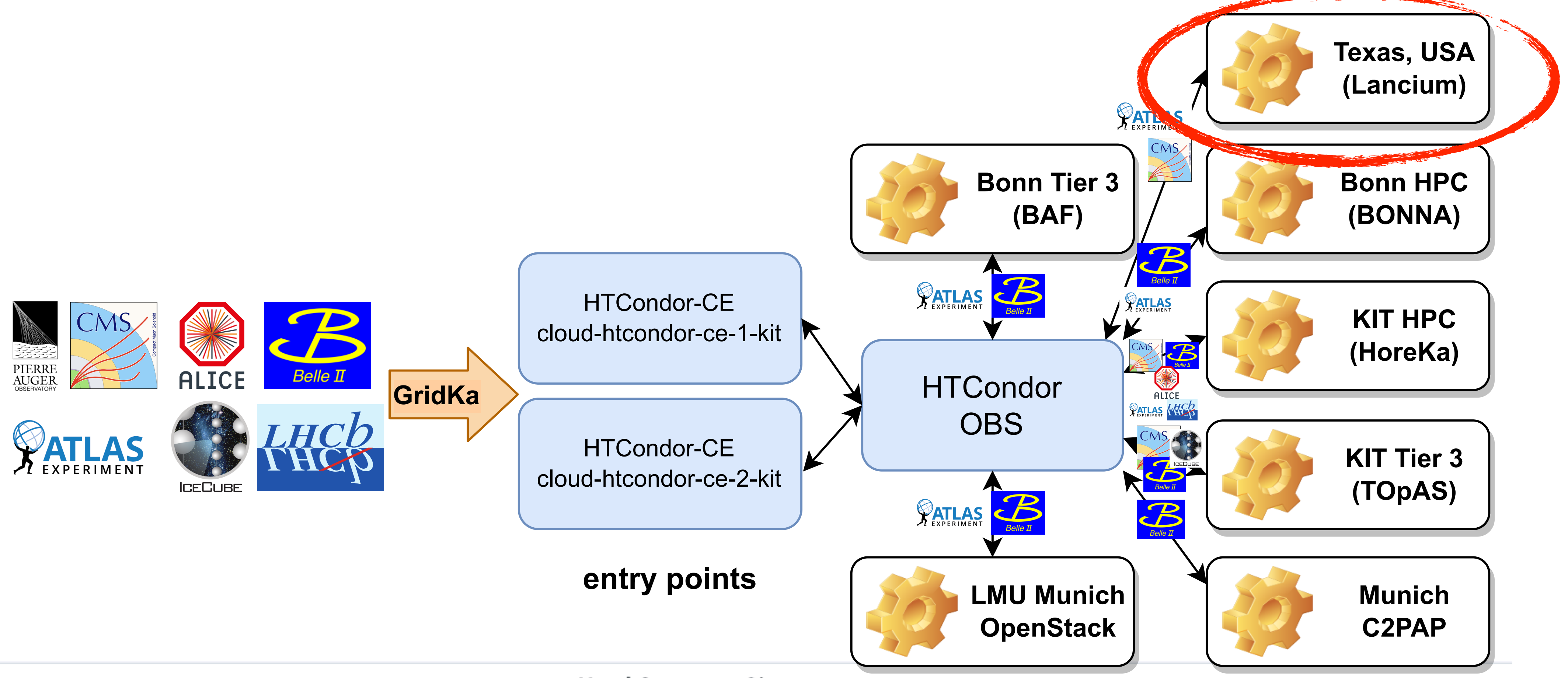


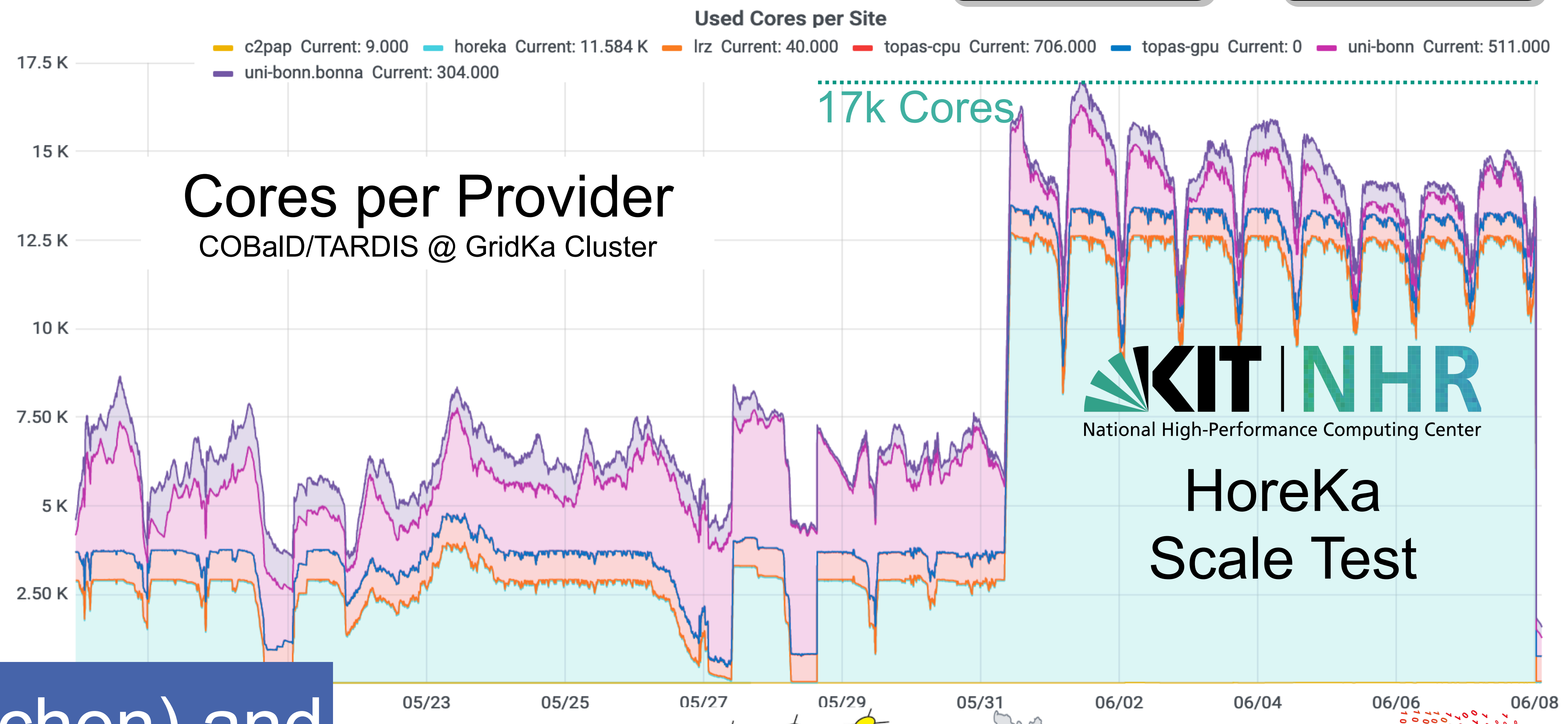
Similar setup deployed at CLAIX HPC (RWTH Aachen) and on-going deployment at Emmy (University of Göttingen)

Opportunistic Compute @ GridKa in a Nutshell

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

- Dynamic, transparent and on-demand integration via COBalD/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 deployment across HEP institutes & HPC resources coordinated by KIT/GridKa
- Central building block of the Compute4PUNCH infrastructure within PUNCH4NFDI

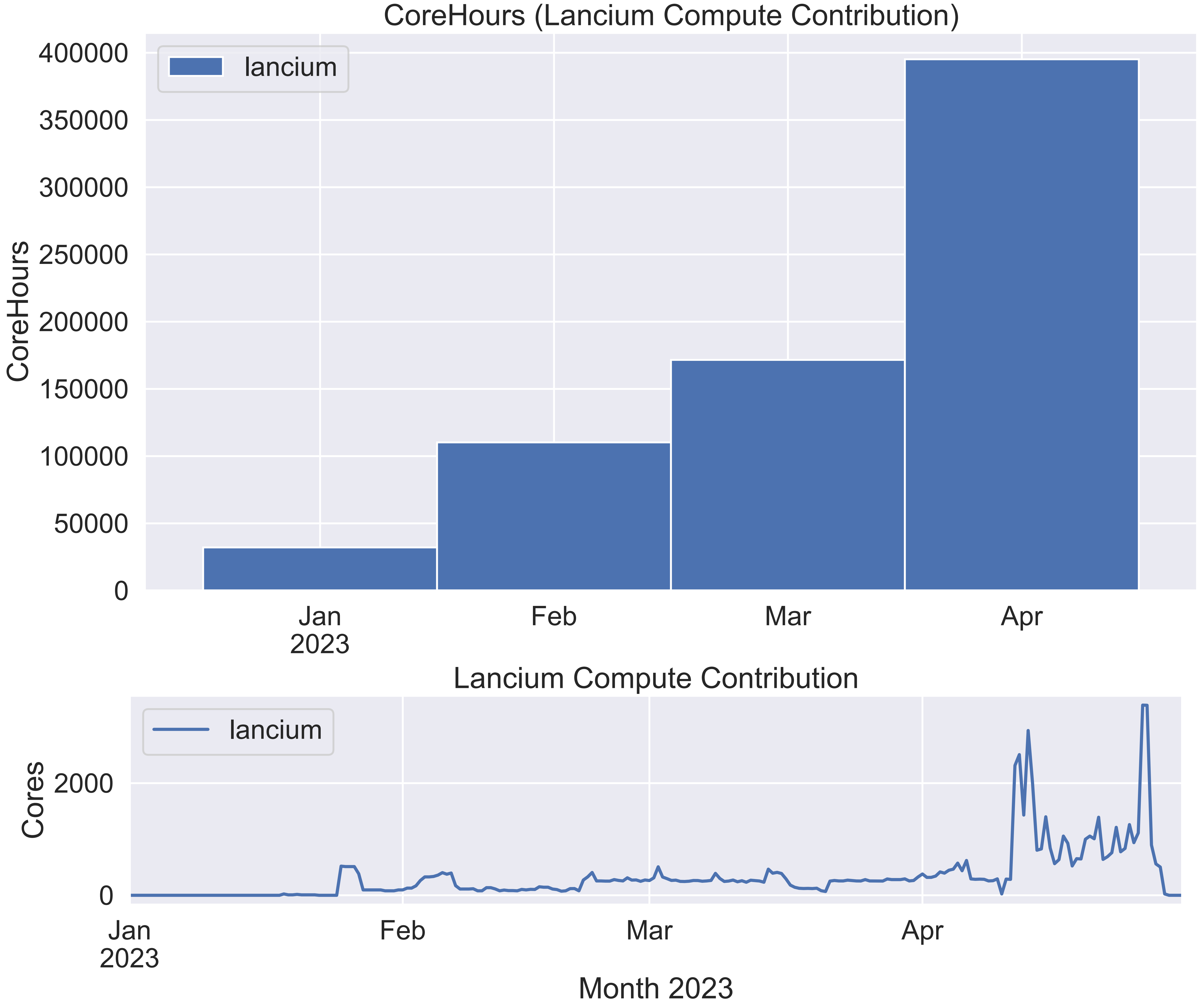




Similar setup deployed at CLAIX HPC (RWTH Aachen) and on-going deployment at Emmy (University of Göttingen)

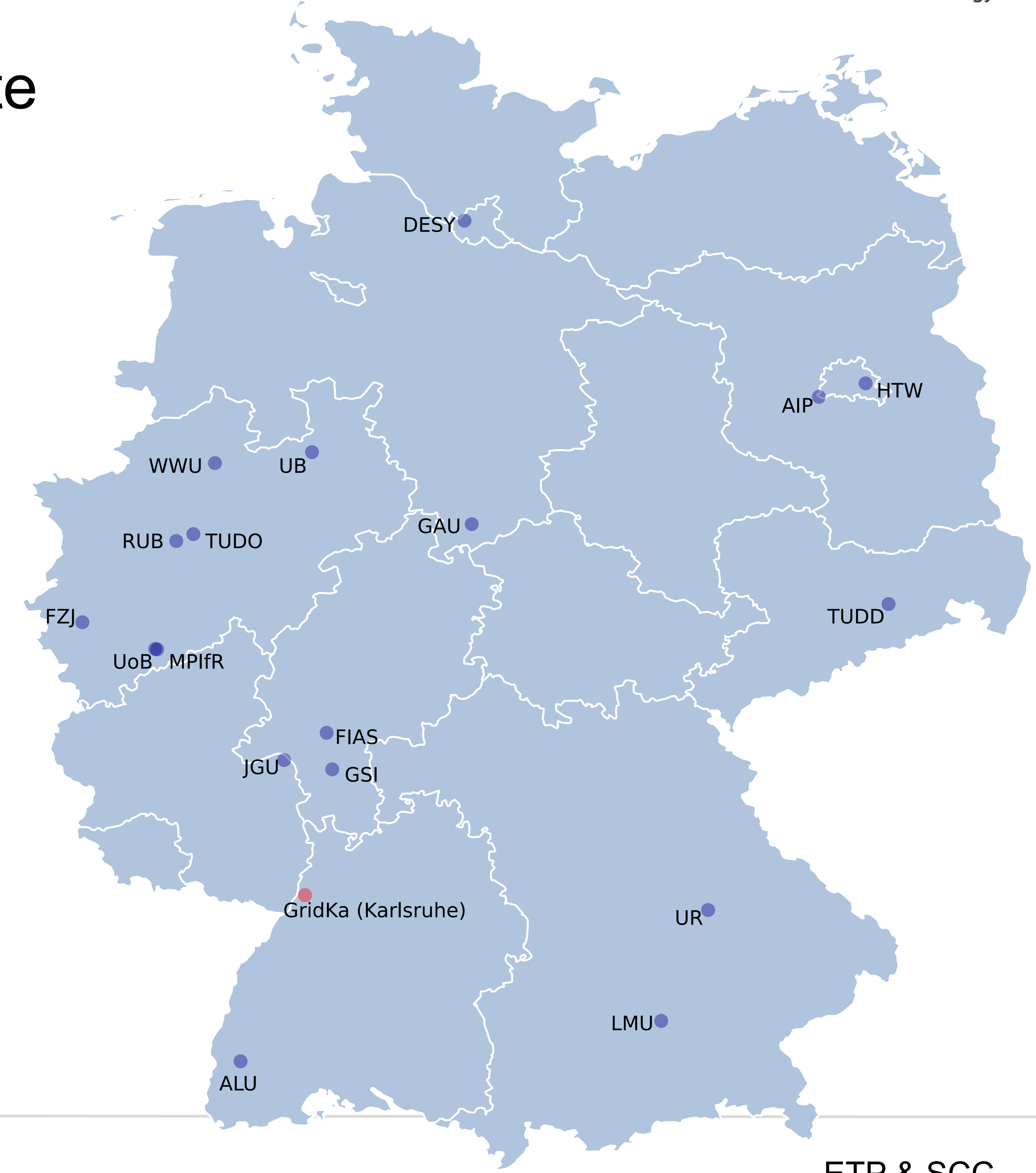
Enabling Access to Sustainable Compute Resources

- Lancium (US company) balancing the power grid by operating compute facilities close to renewables (wind & solar) - CO_2 neutral operation
- Dynamic, transparent and on-demand integration via COBalD/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



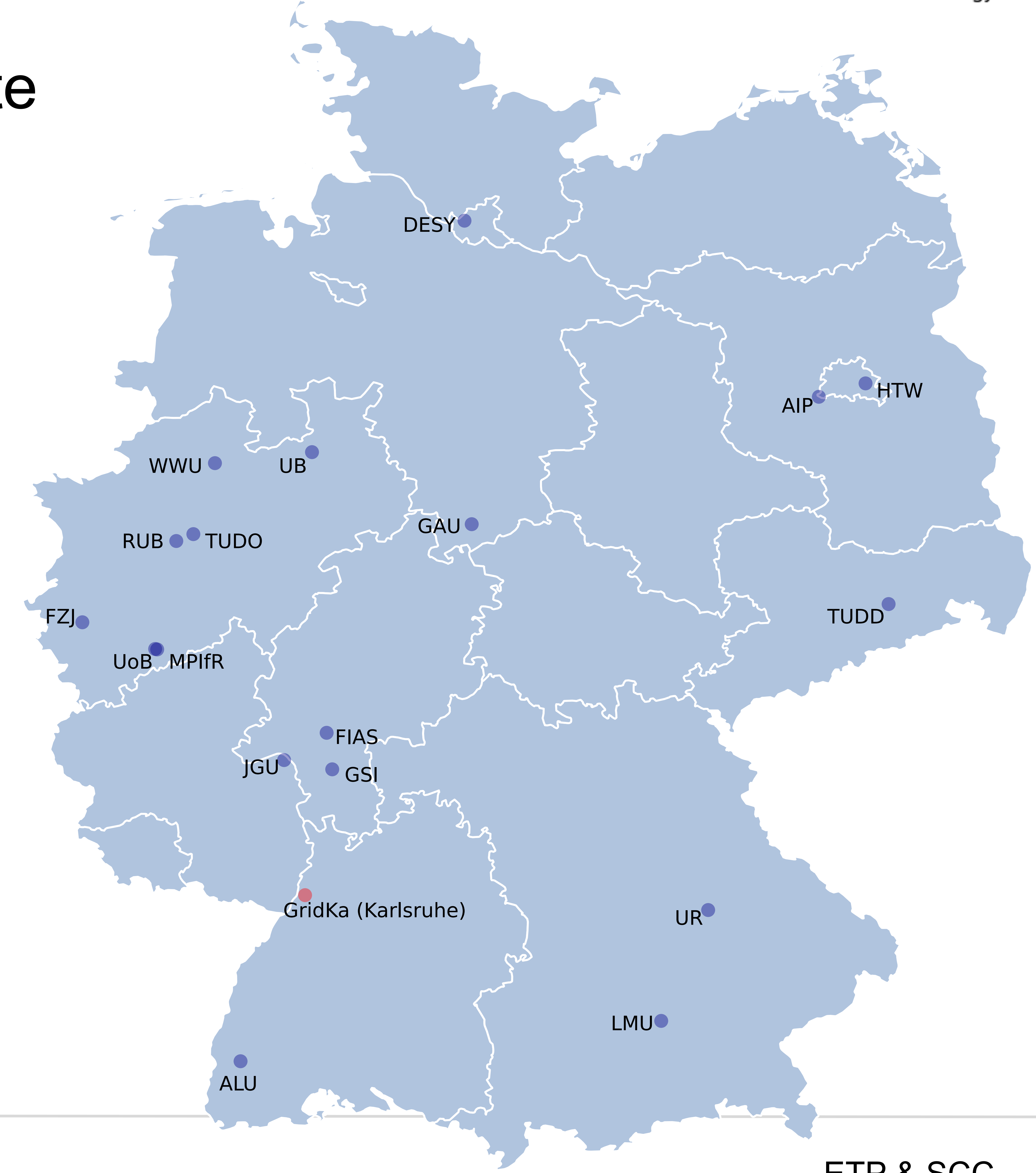
Towards the Compute4PUNCH Infrastructure

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



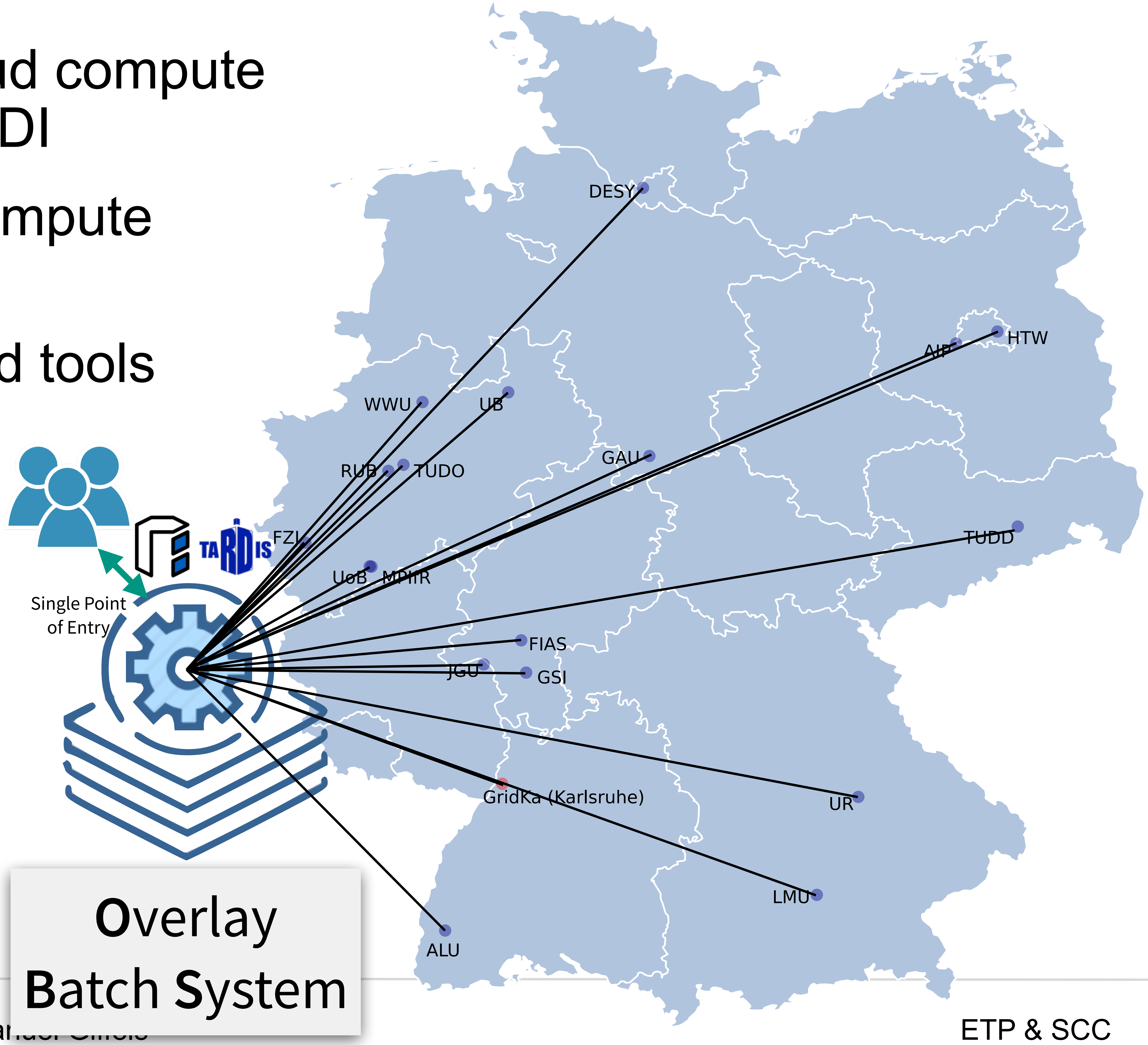
Towards the Compute4PUNCH Infrastructure

- Substantial amount of HTC, HPC, Cloud compute resources are provided to PUNCH4NFDI
- Establish a federated heterogenous compute infrastructure for PUNCH4NFDI



Towards the Compute4PUNCH Infrastructure

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

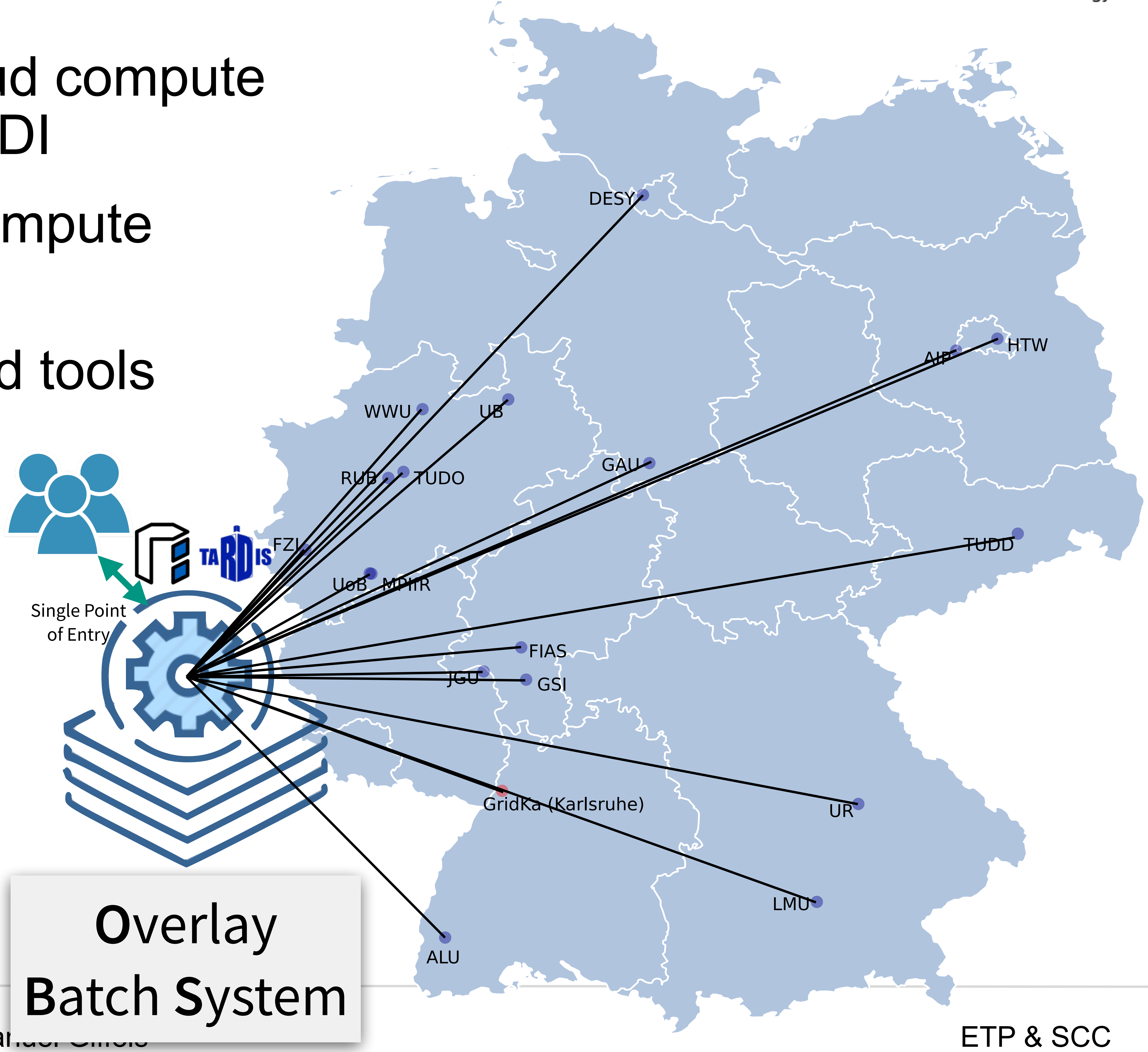


Towards the Compute4PUNCH Infrastructure

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

---

- Compute4PUNCH demonstrator is available

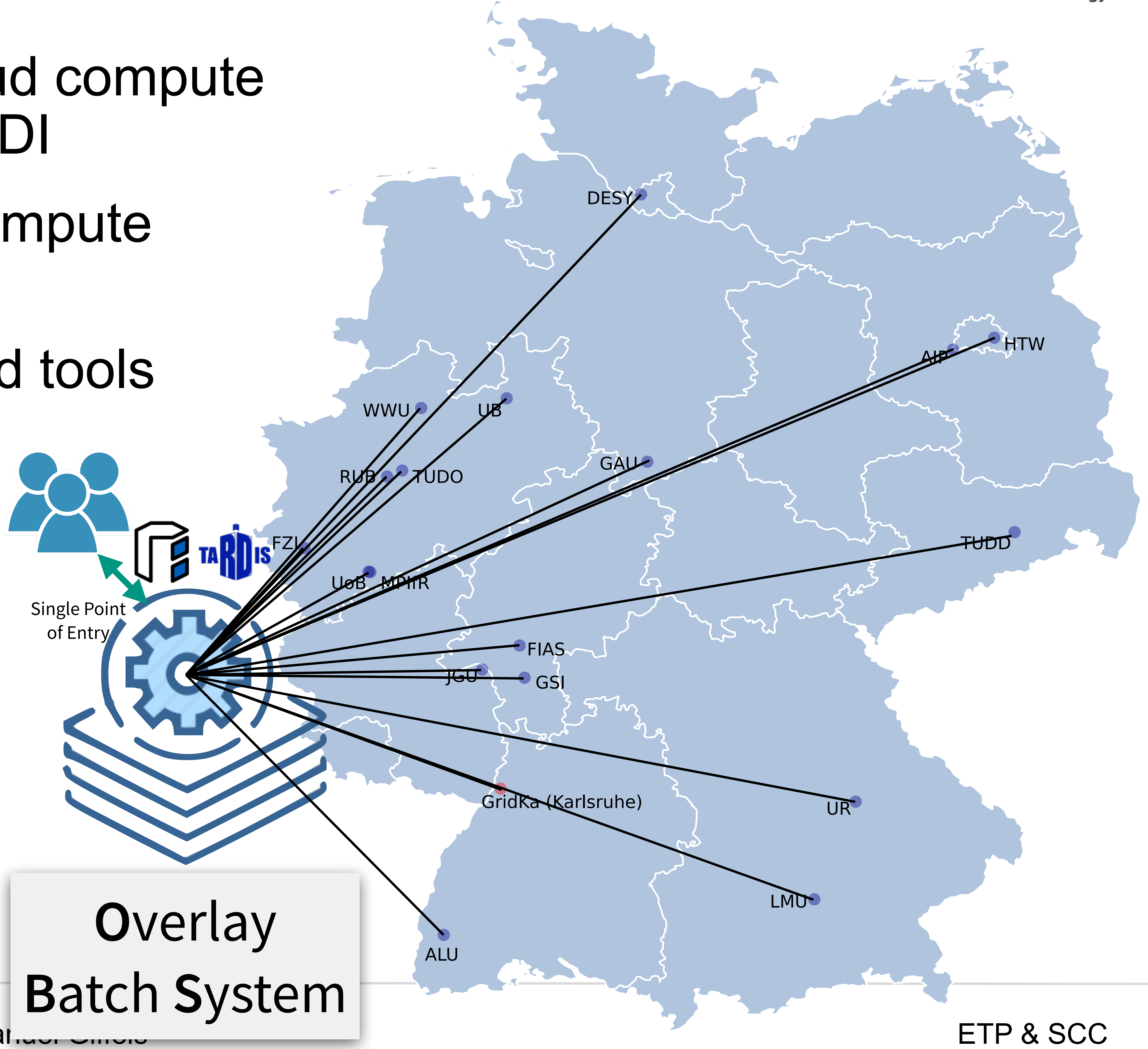


Towards the Compute4PUNCH Infrastructure

- Substantial amount of HTC, HPC, Cloud compute resources are provided to PUNCH4NFDI
- Establish a federated heterogenous 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



Conclusion

<u>Enabling toolset for dynamic federation of heterogeneous compute resources:</u>

- Modern container technology (OS & Software provisioning)
- `COBalD/TARDIS` resource scheduler developed at KIT
- HTCondor overlay batchsystem as federated resource pool
- Single point of entry for users/experiment (e.g. Grid Compute Elements)

- Enables transparent and dynamic on-demand provisioning of heterogeneous compute resources
- Production ready software at scale

Actively used in WLCG computing, FIDIUM & PUNCH4NFDI

Conclusion

Enabling toolset for dynamic federation of heterogeneous compute resources:

- Modern container technology (OS & Software provisioning)
- COBalD/TARDIS resource scheduler developed at KIT
- HTCondor overlay batchsystem as federated resource pool
- Single point of entry for users/experiment
  (e.g. Grid Compute Elements)

- Enables transparent and dynamic on-demand provisioning of heterogeneous compute resources
- Production ready software at scale

for workflows from HEP, Astronomy and Lattice QCD!

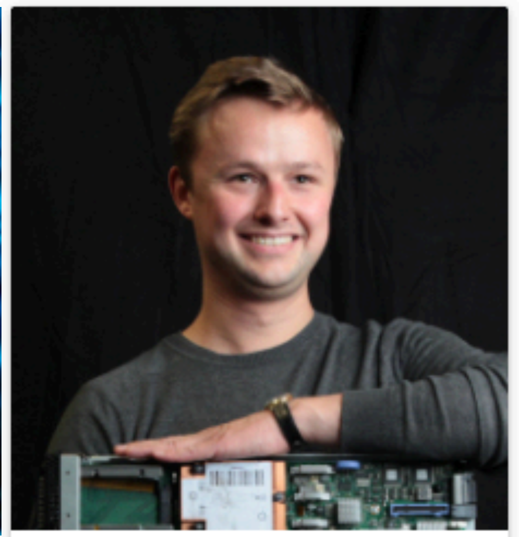
The awesome team behind these success stories

https://matterminers.github.io/

@matterminers

matterminers@lists.kit.edu

Thank you!



R. Florian von Cube
Researcher

Max Fischer
Researcher

Manuel Giffels
Researcher

Robin Hofsaess
Doctoral Researcher

Maximilian Horzela
Doctoral Researcher

Eileen Kuehn
Researcher

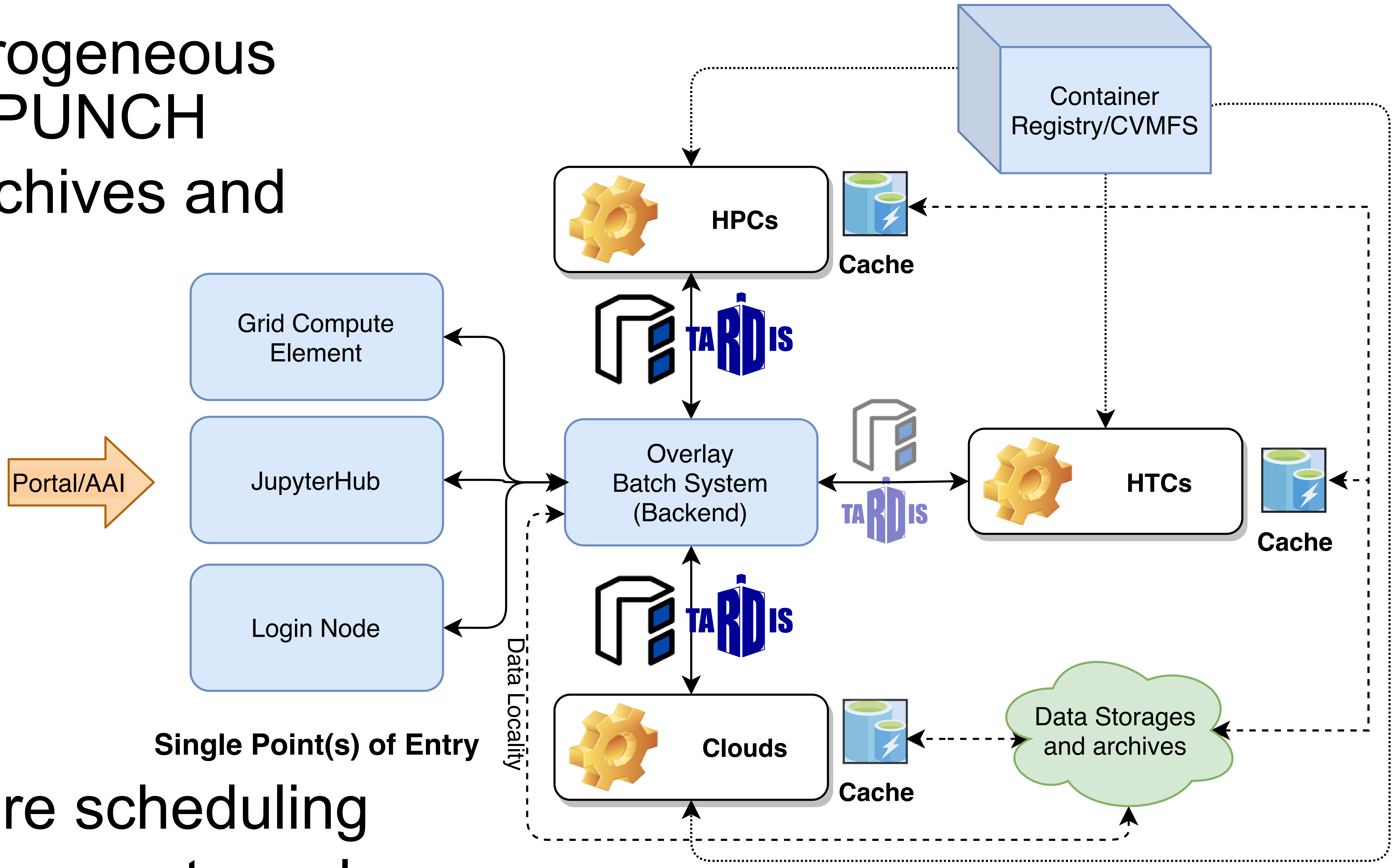
Benoit Roland
Researcher

Matthias Schnepf
Researcher

Backup

Towards the Compute4PUNCH Infrastructure

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



- 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

- **Lo**w **F**requency **Ar**ray (LOFAR)

LOFAR „Superterp“ in Exloo, Netherlands

Workflows on Compute4PUNCH & Storage4PUNCH

LOFAR Radio imaging workflow

- **L**ow **F**requency **Ar**ray (LOFAR)
- Reconstruction of the sky brightness distribution from recorded interferometry data
- Software provided via apptainer container
- Data is available on Storage4PUNCH (~150 GB)

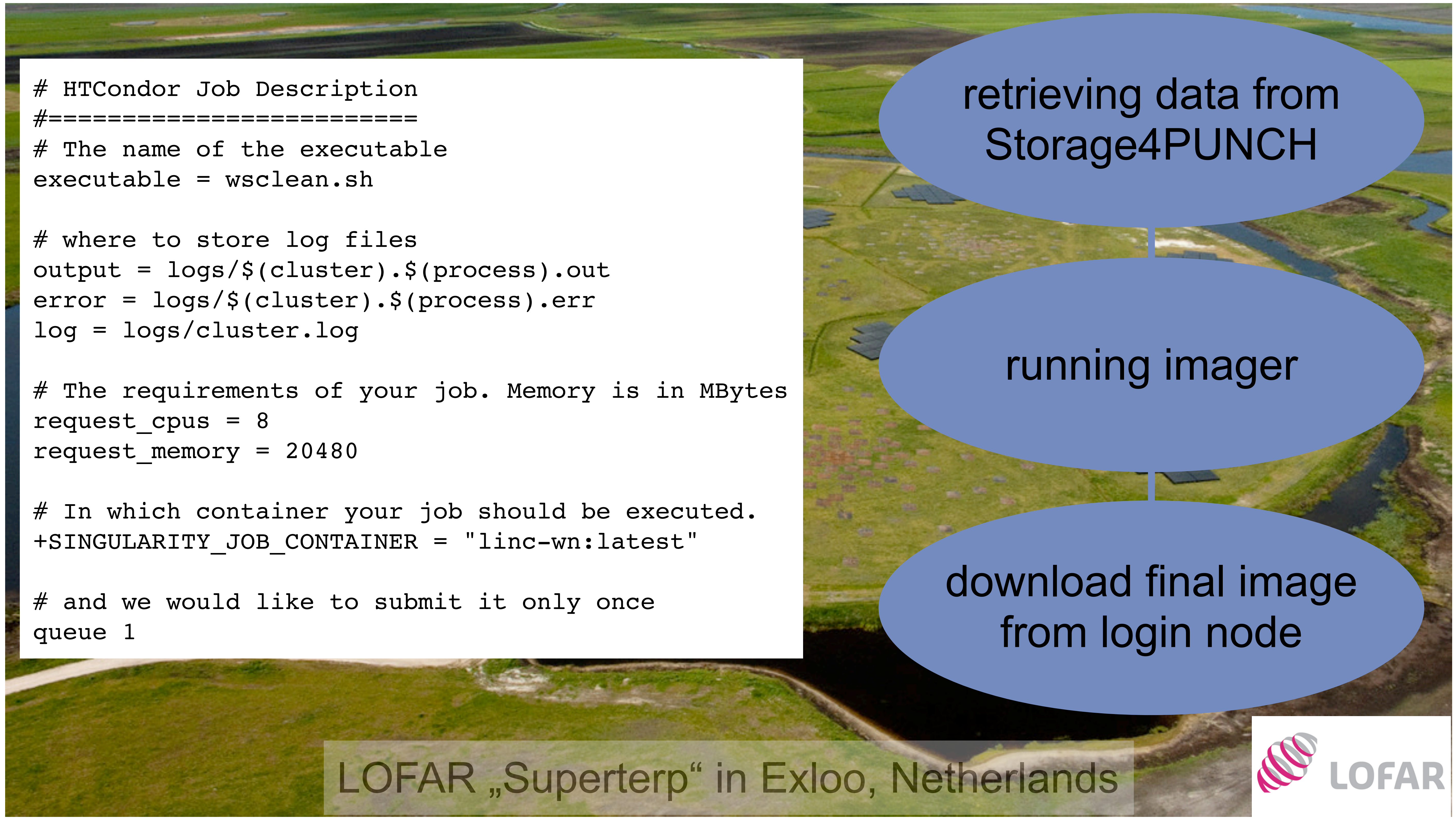
```
# HTCondor Job Description
#=========================
# The name of the executable
executable = wsclean.sh

# 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
```



Workflows on Compute4PUNCH & Storage4PUNCH

Karlsruhe Institute of Technology

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)

LOFAR „Superterp“ in Exloo, Netherlands

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

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

Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

- Resource Meta-Scheduling for Job Scheduler is a „hard
- Usually based on predictions of the future resource availa ity and future mixture of job classes (CPU intense, I/O intense, …)

Works perfectly fine in homogenous environments.

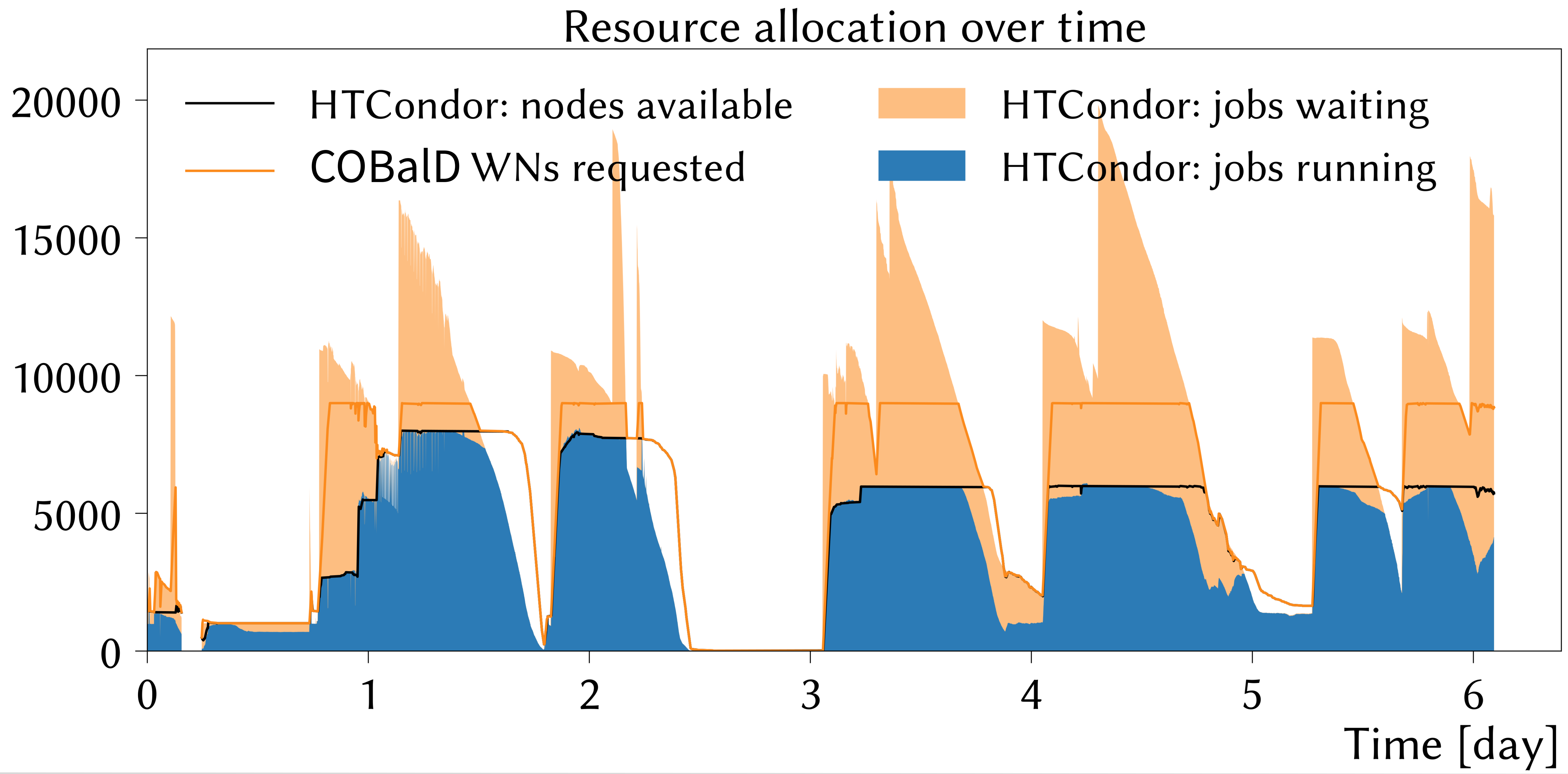
Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

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

Works perfectly fine in homogenous environments.



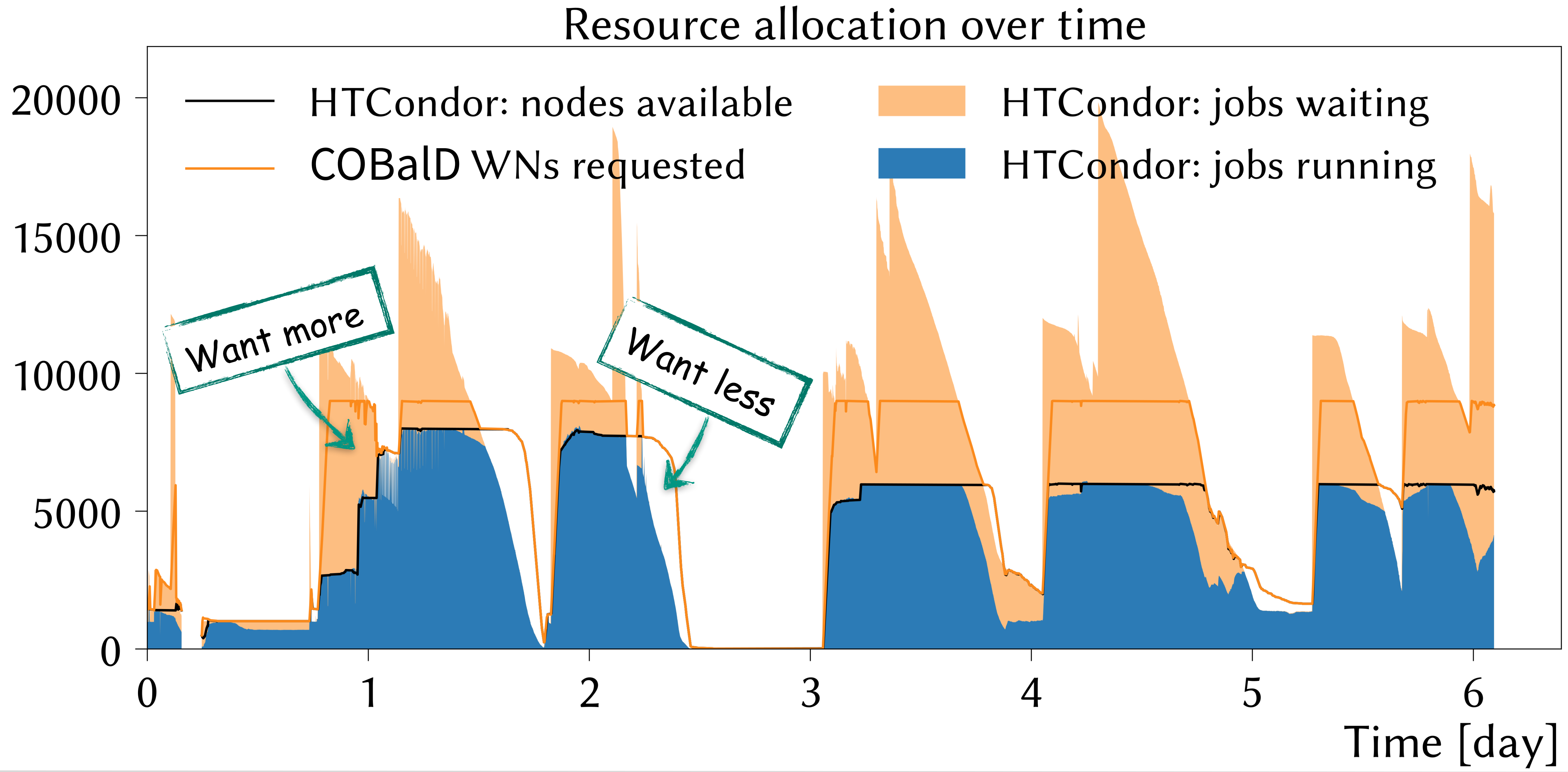
Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

Works perfectly fine in homogenous environments.

- Resource Meta-Scheduling for Job Scheduler is a „hard
- Usually based on predictions of the future resource availa ty and future mixture of job classes (CPU intense, I/O intense, …)
- **<u>However:</u>** We usually care only about a simpler problem!



Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

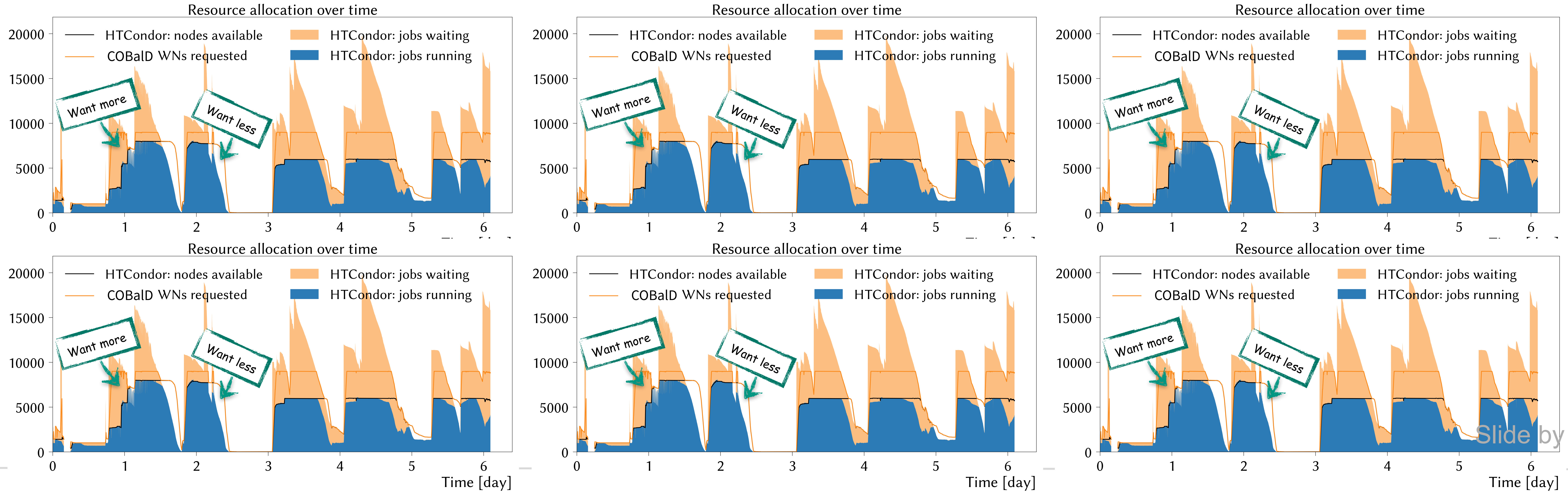
- Resource Meta-Scheduling for Job Scheduler is a „hard“
- COBalD cares only about resources, not jobs
  - Observe how much and how well each resource is used
  - Increase well-used resources, decrease unused resources

Slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

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



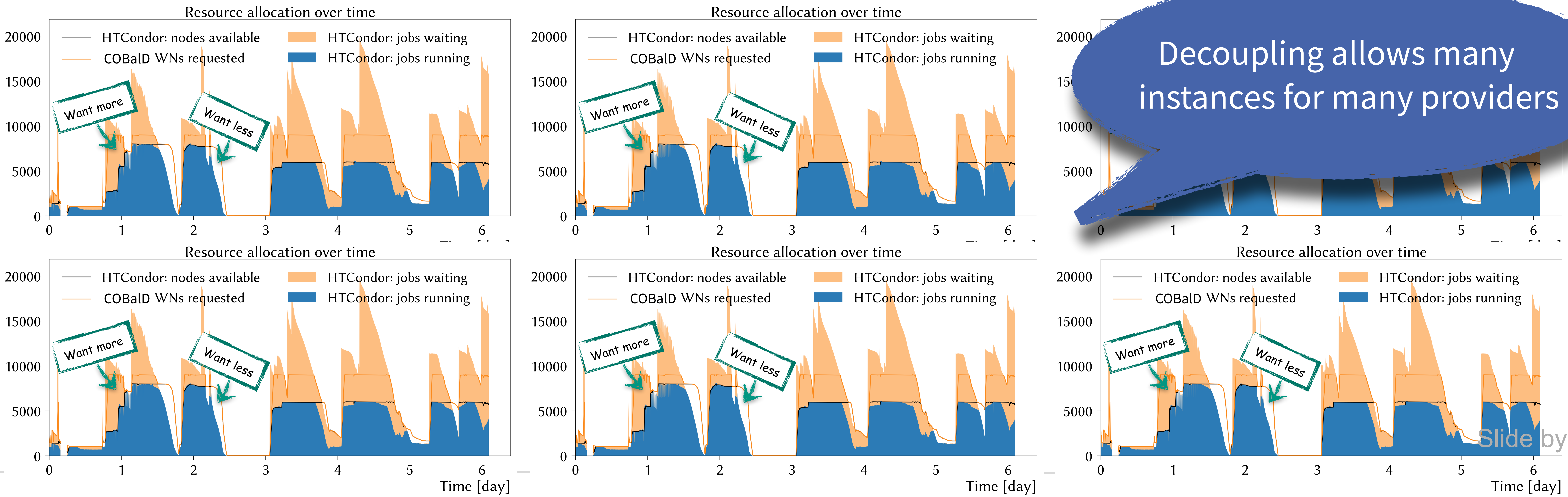
Slide by Max Fischer

The COBalD View of Resource Scheduling

[COBalD - the **O**pportunistic **Bal**ancing **D**aemon]

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

COBalD

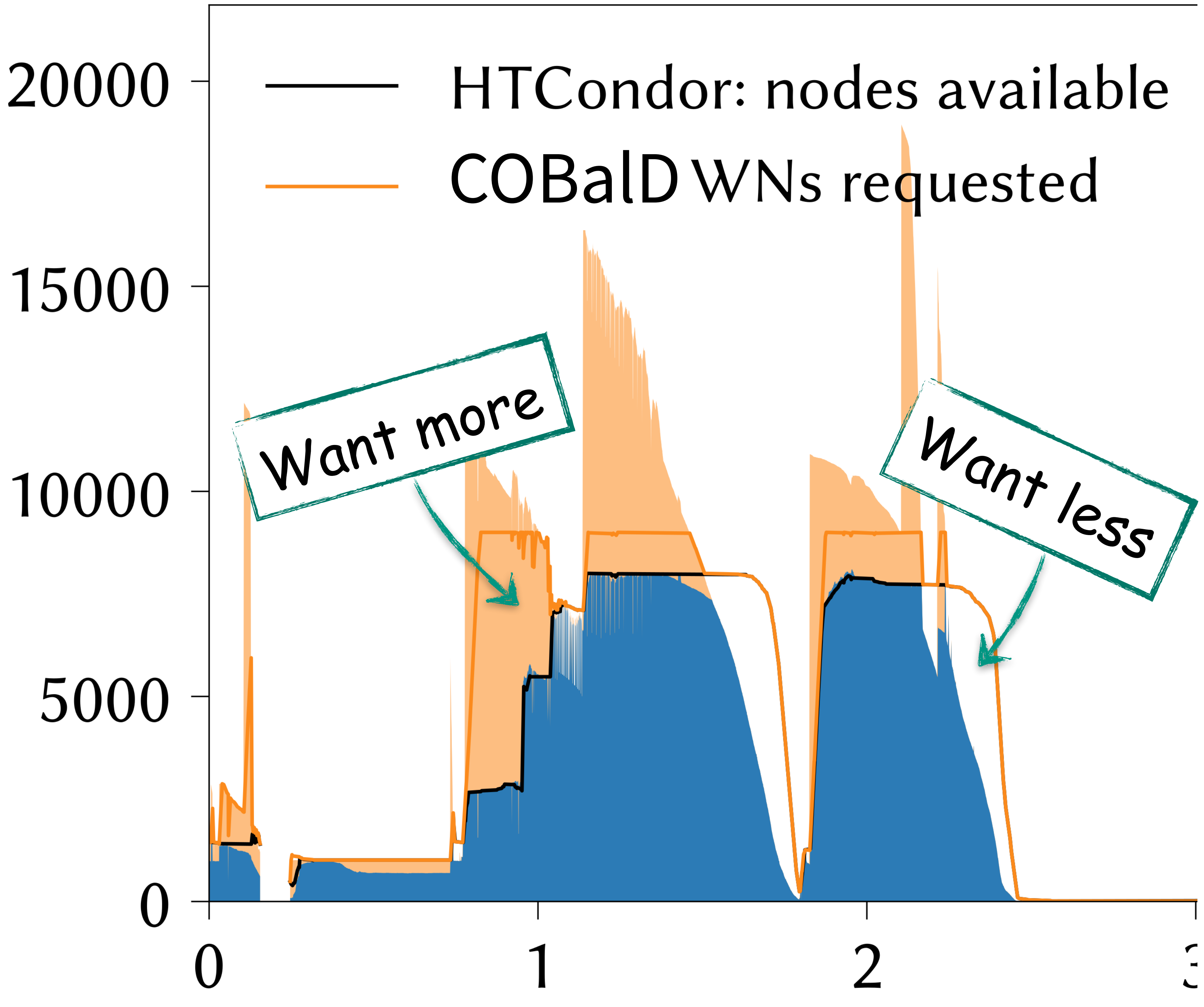


Decoupling allows many instances for many providers

Slide by Max Fischer

COBalD - The Opportunistic Balancing Daemon

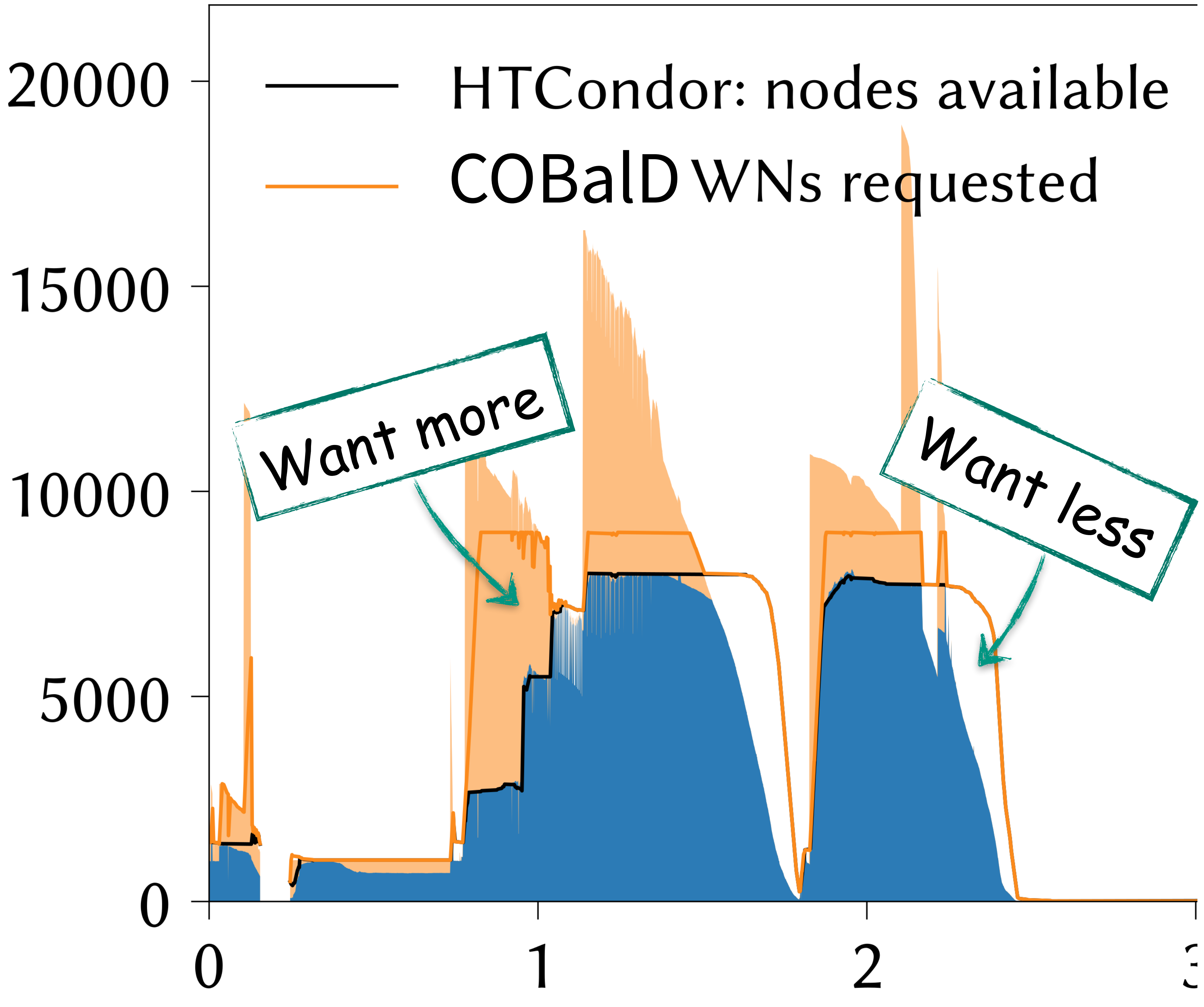
- Look at what is used, not what is requested
  - Simple logic: more used, less unused resources
  - COBalD acquires/releases resources
  - Batch system scheduler handles jobs
- Generic design for any resources
  - COBalD just knows (un-)used resources
  - CPU, CPU+RAM, GPU, VM, …
- HTC integration via COBalD/TARDIS
  - Define VM/Container/Job as resource
  - Supports any use-case that can be put into a VM/container/script!



COBalD - The Opportunistic Balancing Daemon

- Look at what is used, not what is requested
  - Simple logic: more used, less unused resources
  - COBalD acquires/releases resources
  - Batch system scheduler handles jobs
- Generic design for any resources
  - COBalD just knows (un-)used resources
  - CPU, CPU+RAM, GPU, VM, …
- HTC integration via COBalD/TARDIS
  - Define VM/Container/Job as resource
  - Supports any use-case that can be put into a VM/container/script!

Mainly developed at KIT



TARDIS - Out-of-the-Box Resource Adapters

[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

TARDIS - Out-of-the-Box Resource Adapters

[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

- Combine resource provider APIs with COBalD
  - Request, monitor, decommission individual resources (resource life cycle)
  - Automatically match demand via COBalD approach
  - Basically a „use-case agnostic autonomous Pilot factory“

TARDIS - Out-of-the-Box Resource Adapters

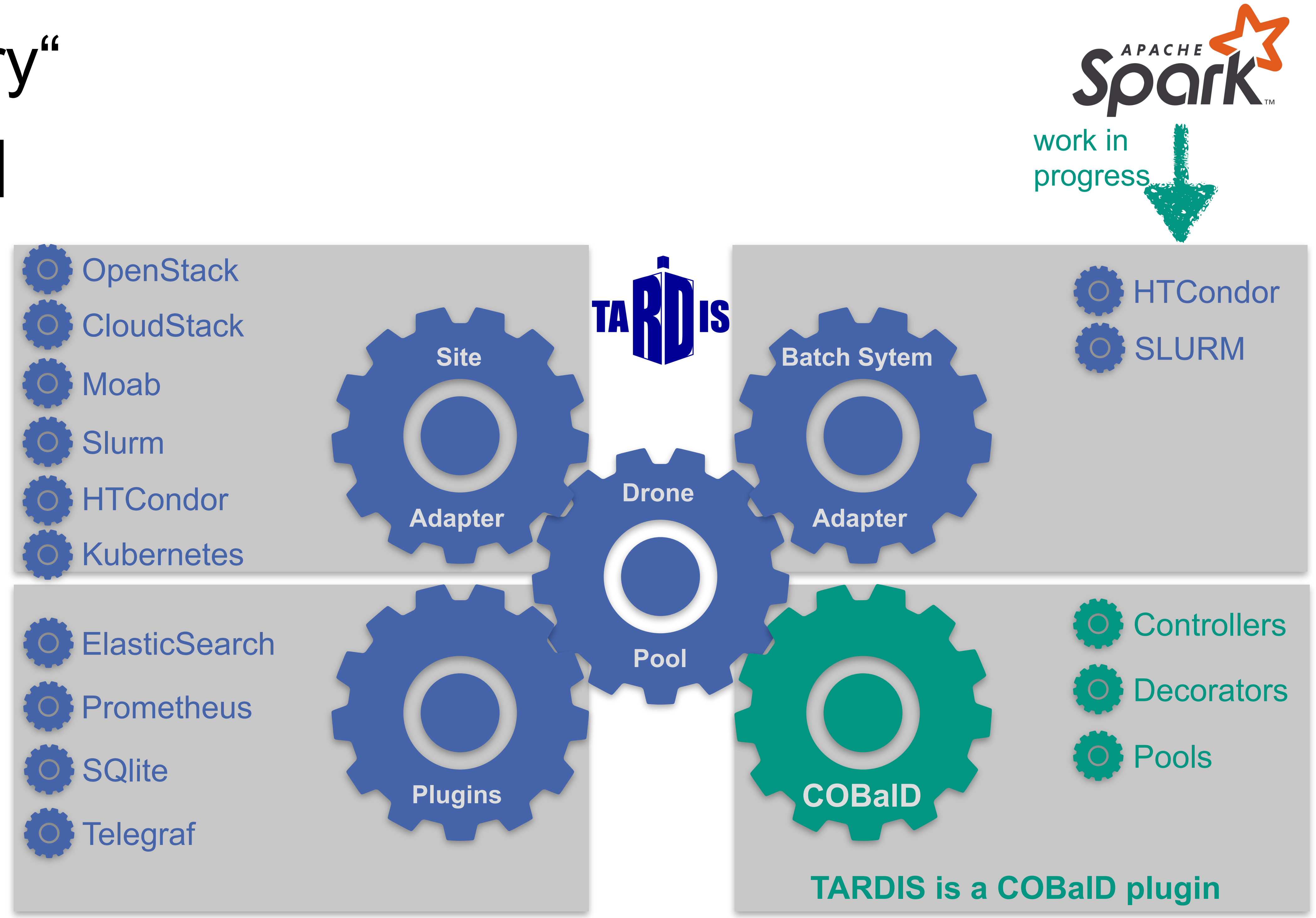
[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

- Combine resource provider APIs with COBalD
  - Request, monitor, decommission individual resources (resource life cycle)
  - Automatically match demand via COBalD 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
    - Need user (PID) namespaces to be enabled
    - Decent WAN connection to WLCG Grid storages
  - Customizable payload for each centre’s peculiarities
  - HEP: Insert HTCondor+CVMFS as available

TARDIS - Out-of-the-Box Resource Adapters

[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

- Combine resource provider APIs with COBalD
  - Request, monitor, decommission individual resources (resource life cycle)
  - Automatically match demand via COBalD 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
    - Need user (PID) namespaces to be enabled
    - Decent WAN connection to WLCG Grid storages
  - Customizable payload for each centre’s peculiarities
  - HEP: Insert HTCondor+CVMFS as available



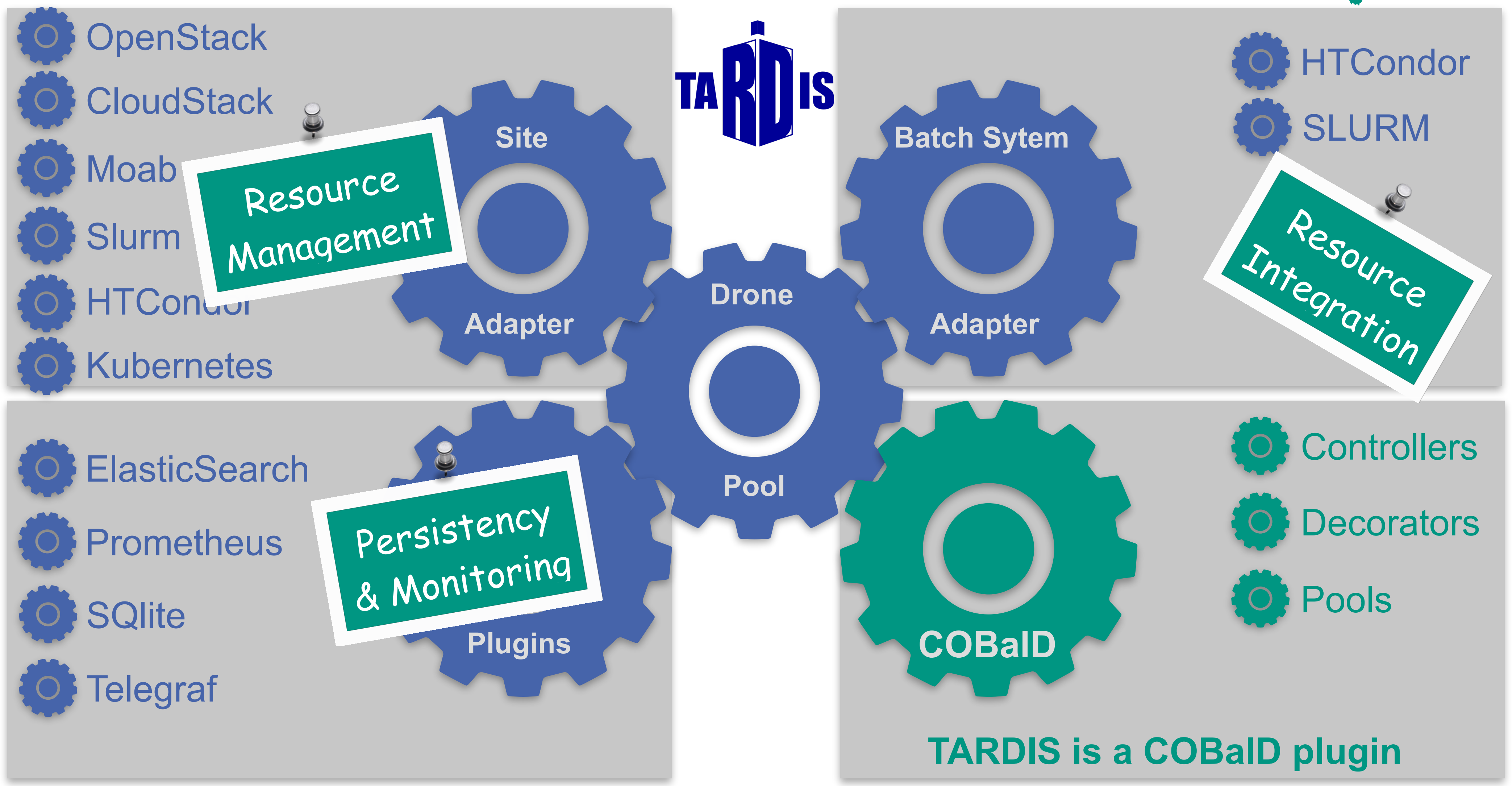
TARDIS - Out-of-the-Box Resource Adapters

[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

- Combine resource provider APIs with COBalD
  - Request, monitor, decommission individual resources (resource life cycle)
  - Automatically match demand via COBalD 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
    - Need user (PID) namespaces to be enabled
    - Decent WAN connection to WLCG Grid storages
  - Customizable payload for each centre’s peculiarities
  - HEP: Insert HTCondor+CVMFS as available

Apache Spark

work in progress



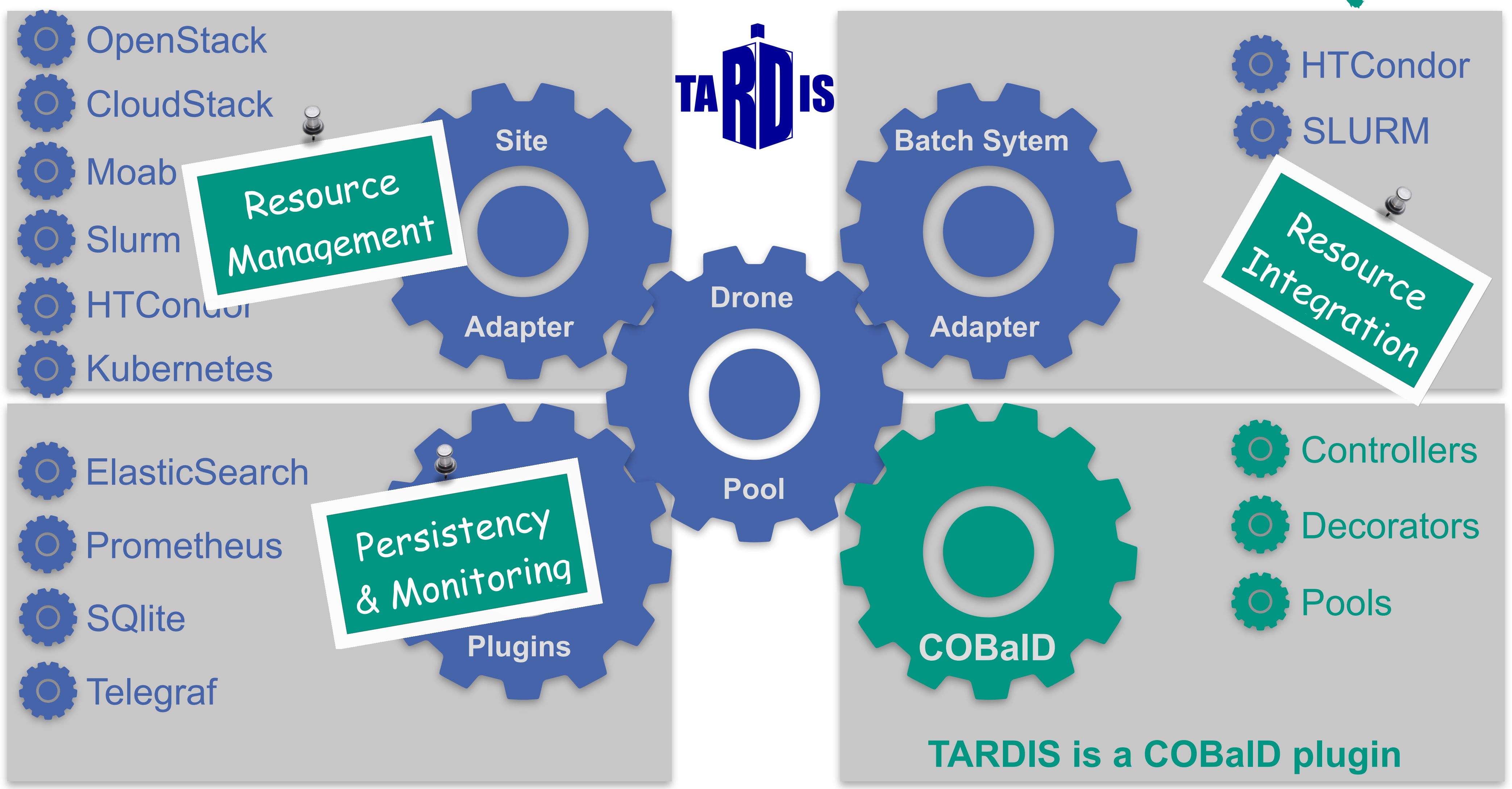
TARDIS - Out-of-the-Box Resource Adapters

[**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

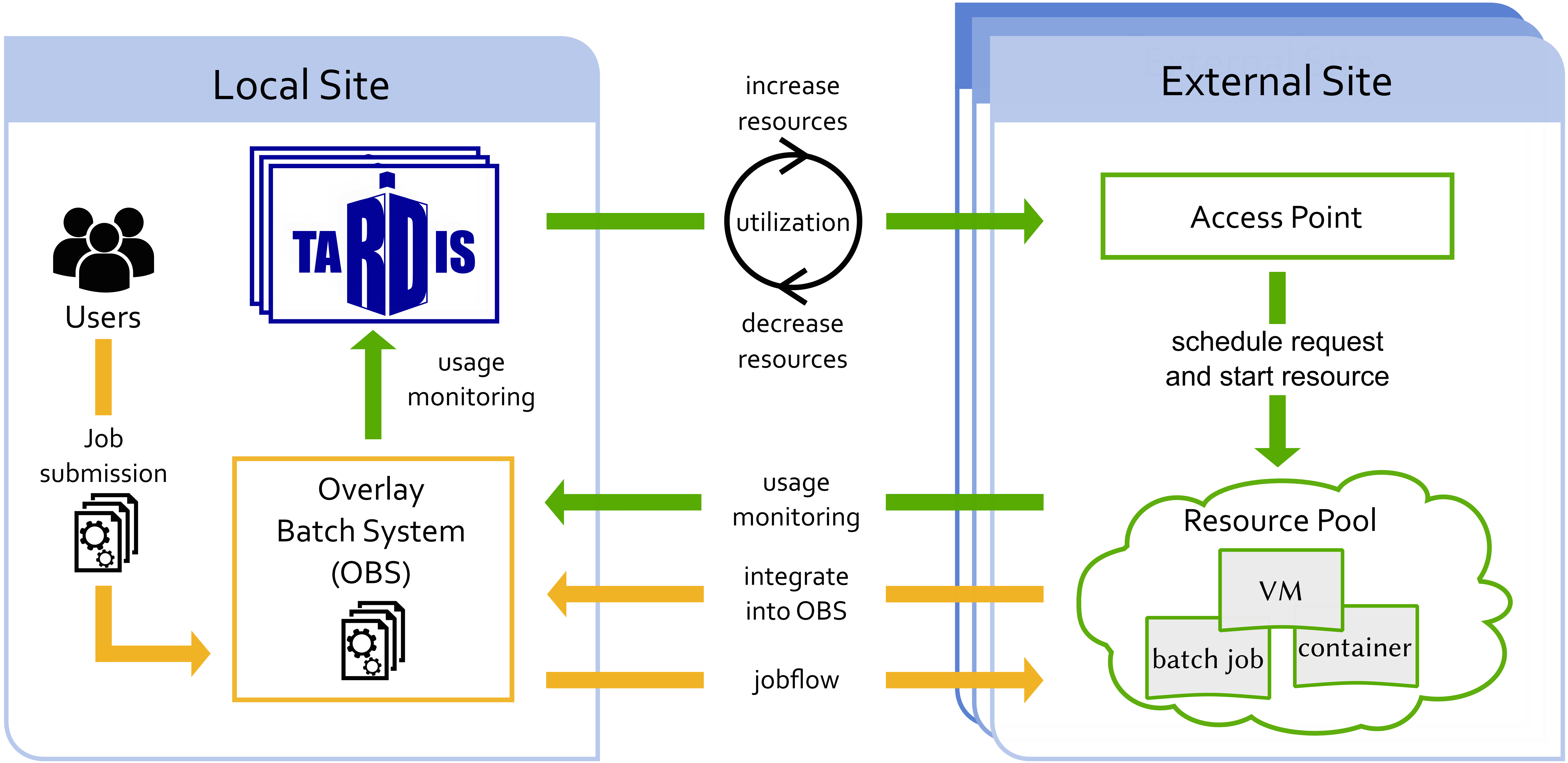
- Combine resource provider APIs with COBalD
  - Request, monitor, decommission individual resources (resource life cycle)
  - Automatically match demand via COBalD approach
  - Basically a „use-case agnostic autonomous Pilot factory“
- Support for common HPC batch systems, Cloud APIs, …
  - Behave like „regular users“ as much as possible
    - Need user (PID) namespaces to be enabled
    - Decent WAN connection to WLCG Grid storages
  - Customizable payload for each centre’s peculiarities
  - HEP: Insert HTCondor+CVMFS as available

Mainly developed at KIT

work in progress



COBalD/TARDIS & Opportunistic Resources in Practice



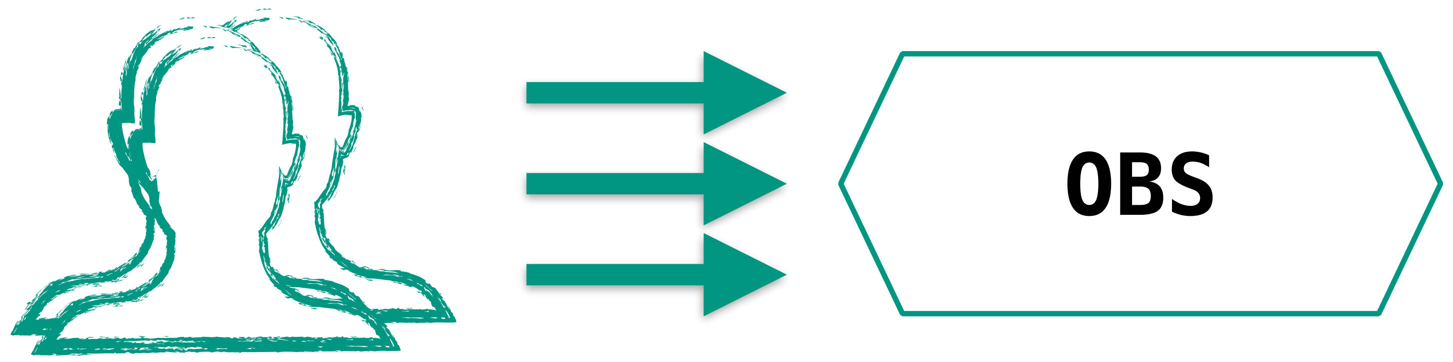
Resource Meta-Scheduler

Resource Meta-Scheduler

Classical Job to Resource to Job meta-scheduler:

Resource Meta-Scheduler

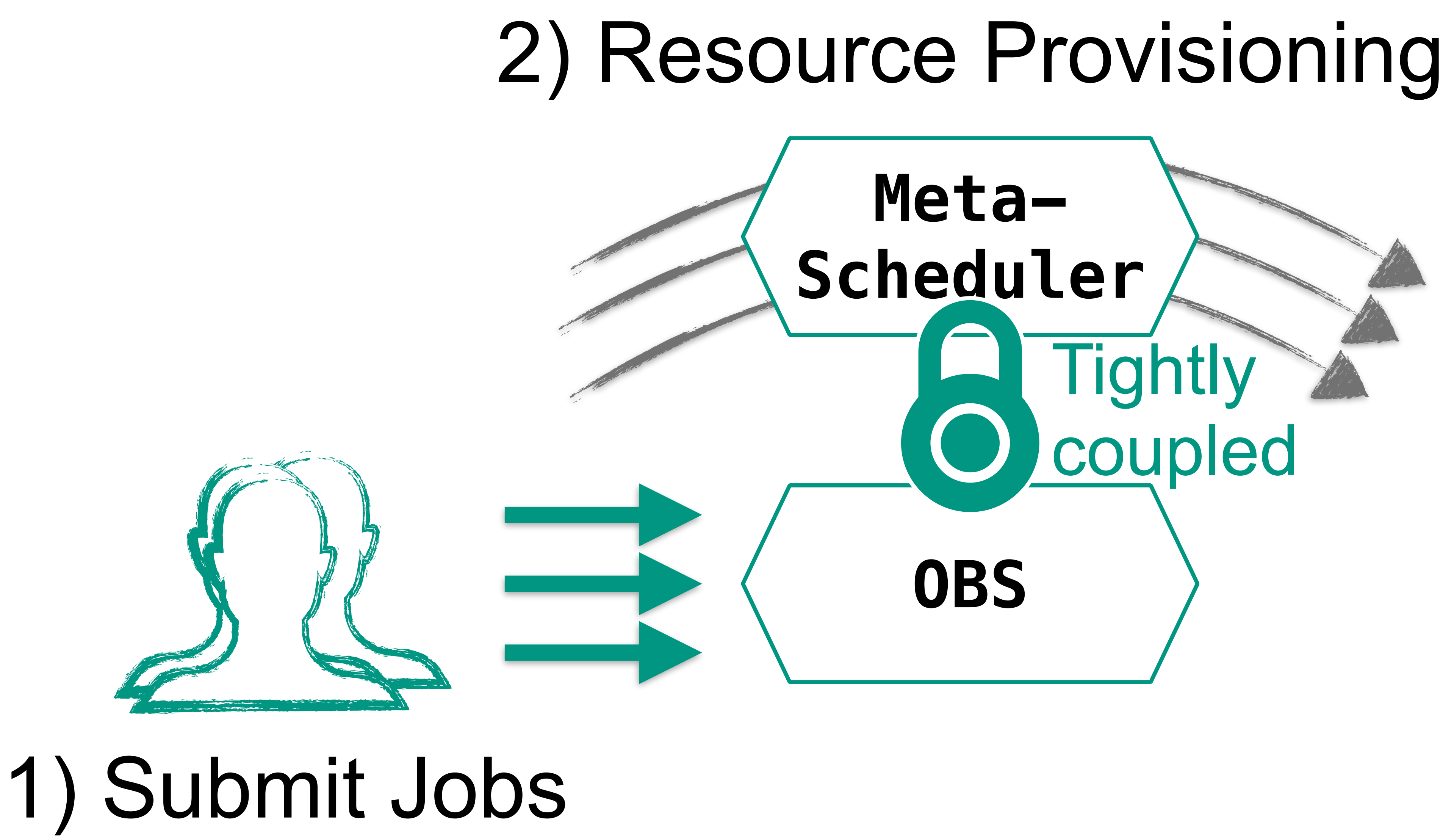
Classical Job to Resource to Job meta-scheduler:



1) Submit Jobs

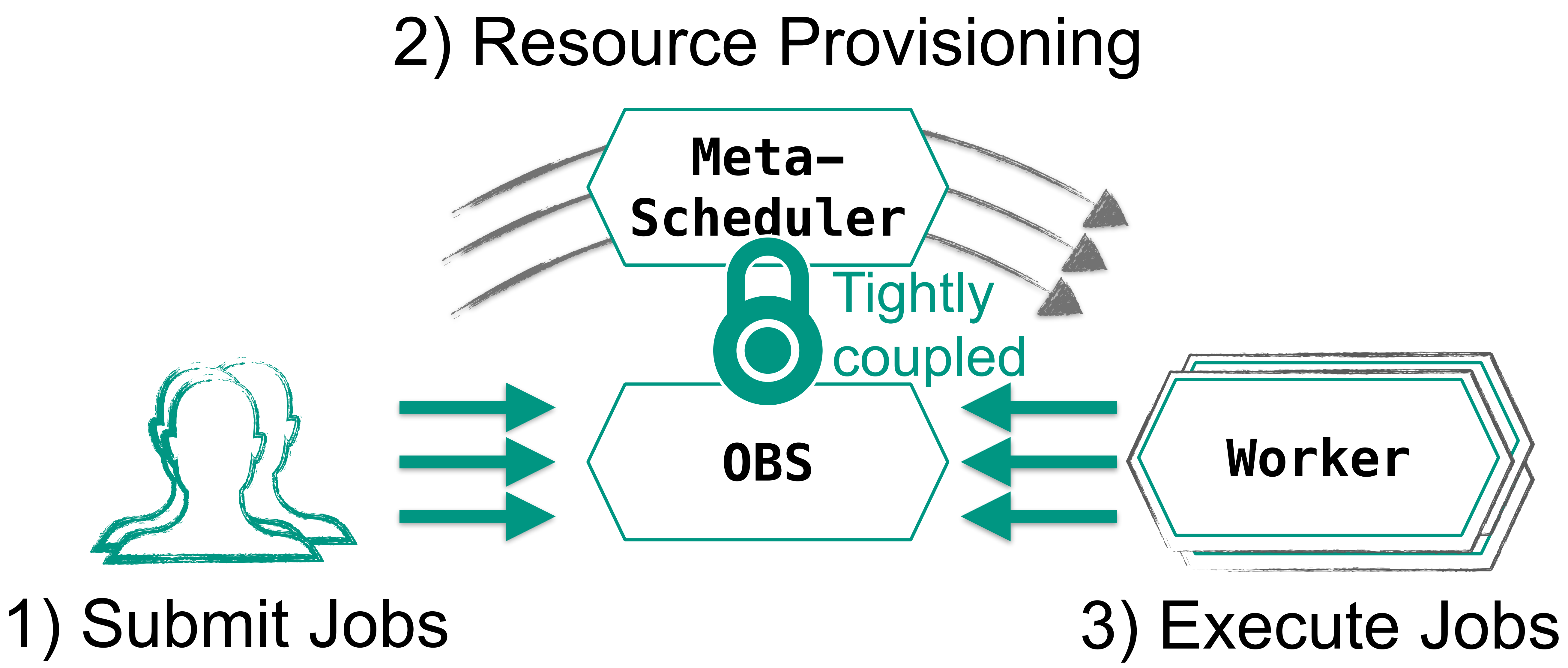
Resource Meta-Scheduler

Classical Job to Resource to Job meta-scheduler:



Resource Meta-Scheduler

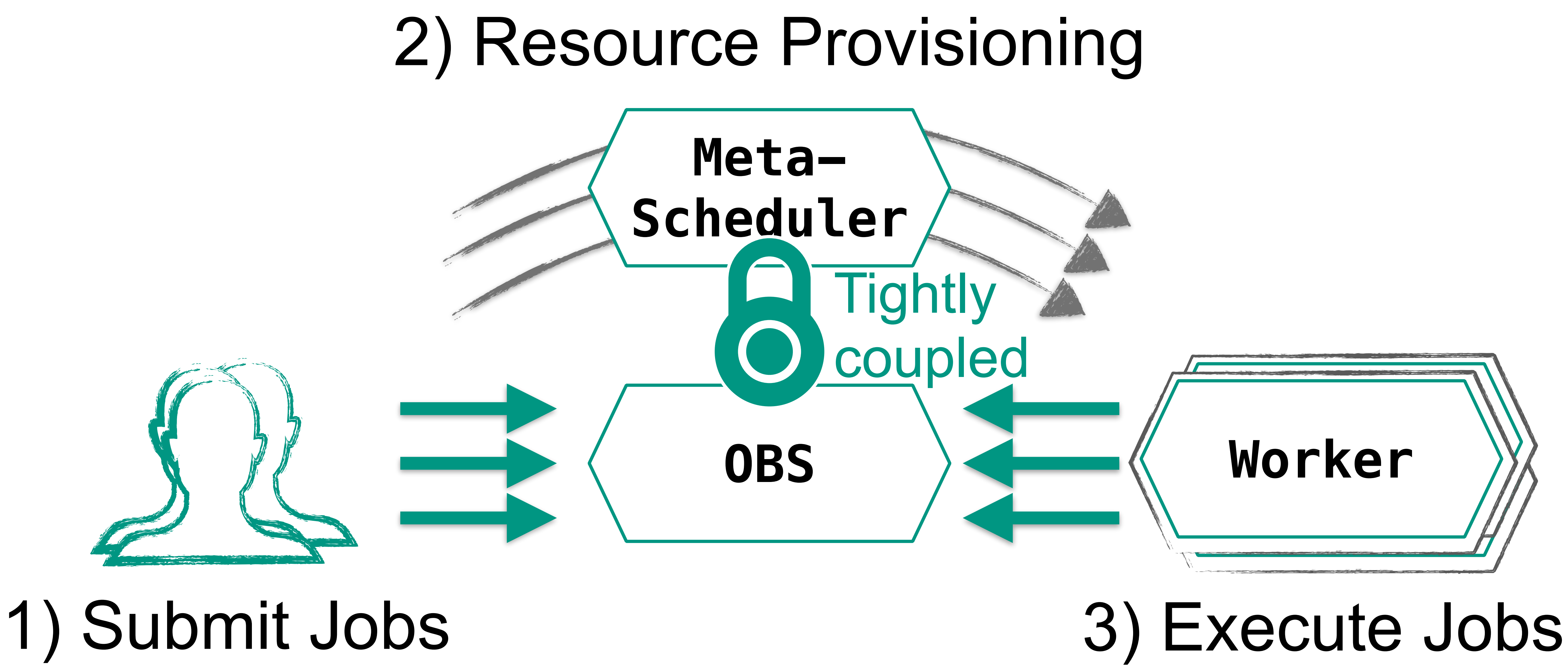
Classical Job to Resource to Job meta-scheduler:



Resource Meta-Scheduler

Classical Job to Resource to Job meta-scheduler:

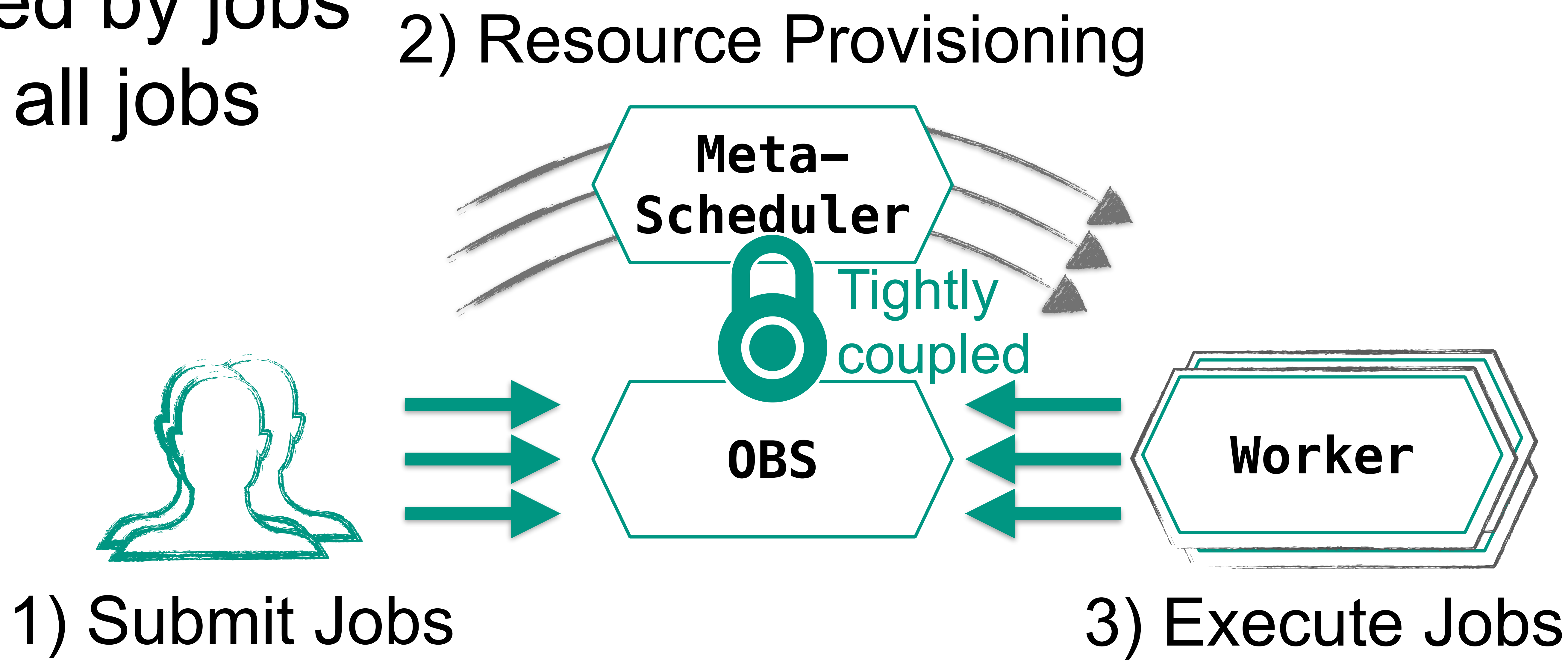
- Dynamic resource acquisition matching user demand
  - Trivial to support new providers for many users
  - Difficult to manage several providers for many users



Resource Meta-Scheduler

Classical Job to Resource to Job meta-scheduler:

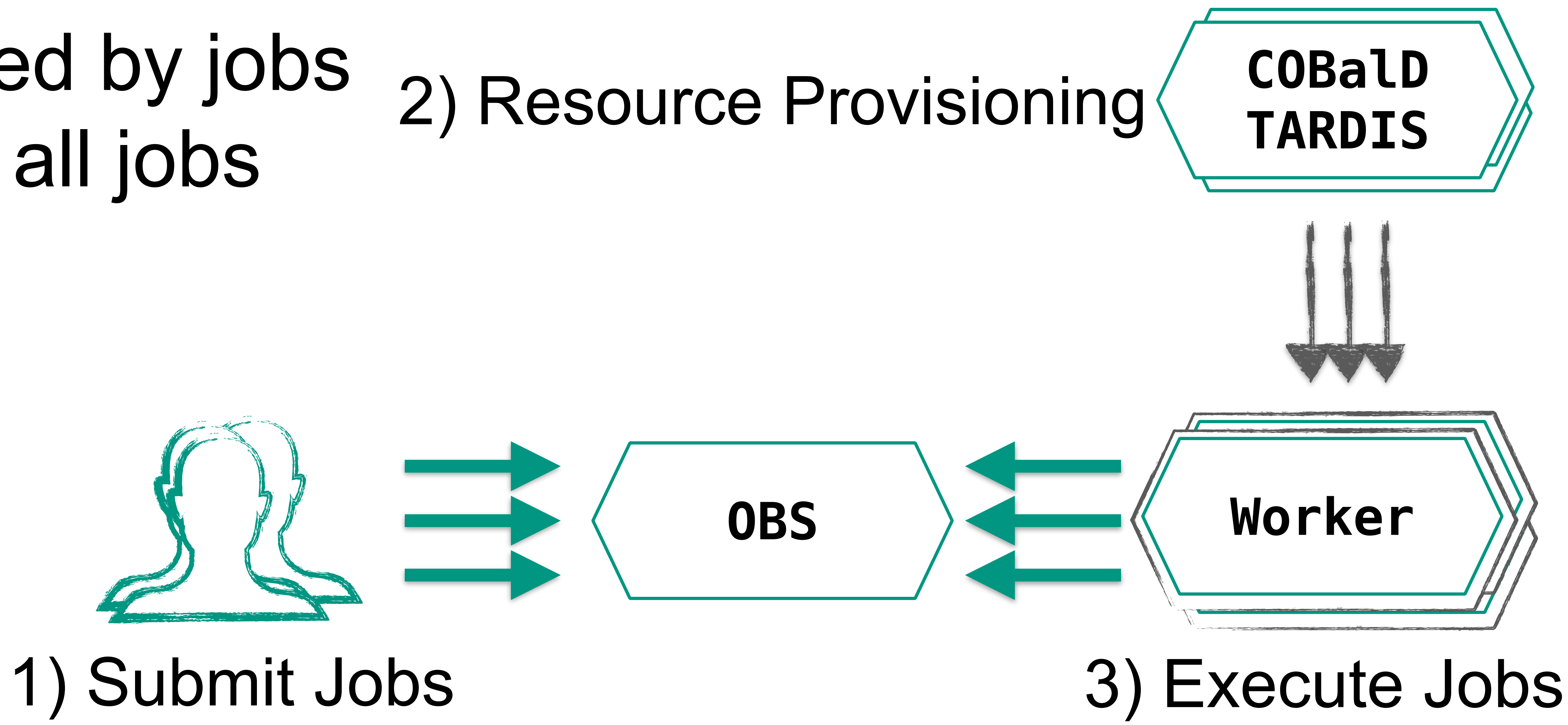
- Dynamic resource acquisition matching user demand
  - Trivial to support new providers for many users
  - Difficult to manage several providers for many users
- Job scheduling in overlay batch system
  - Unreliable to predict resources used by jobs
  - Efficient to integrate resources for all jobs



Resource Meta-Scheduler

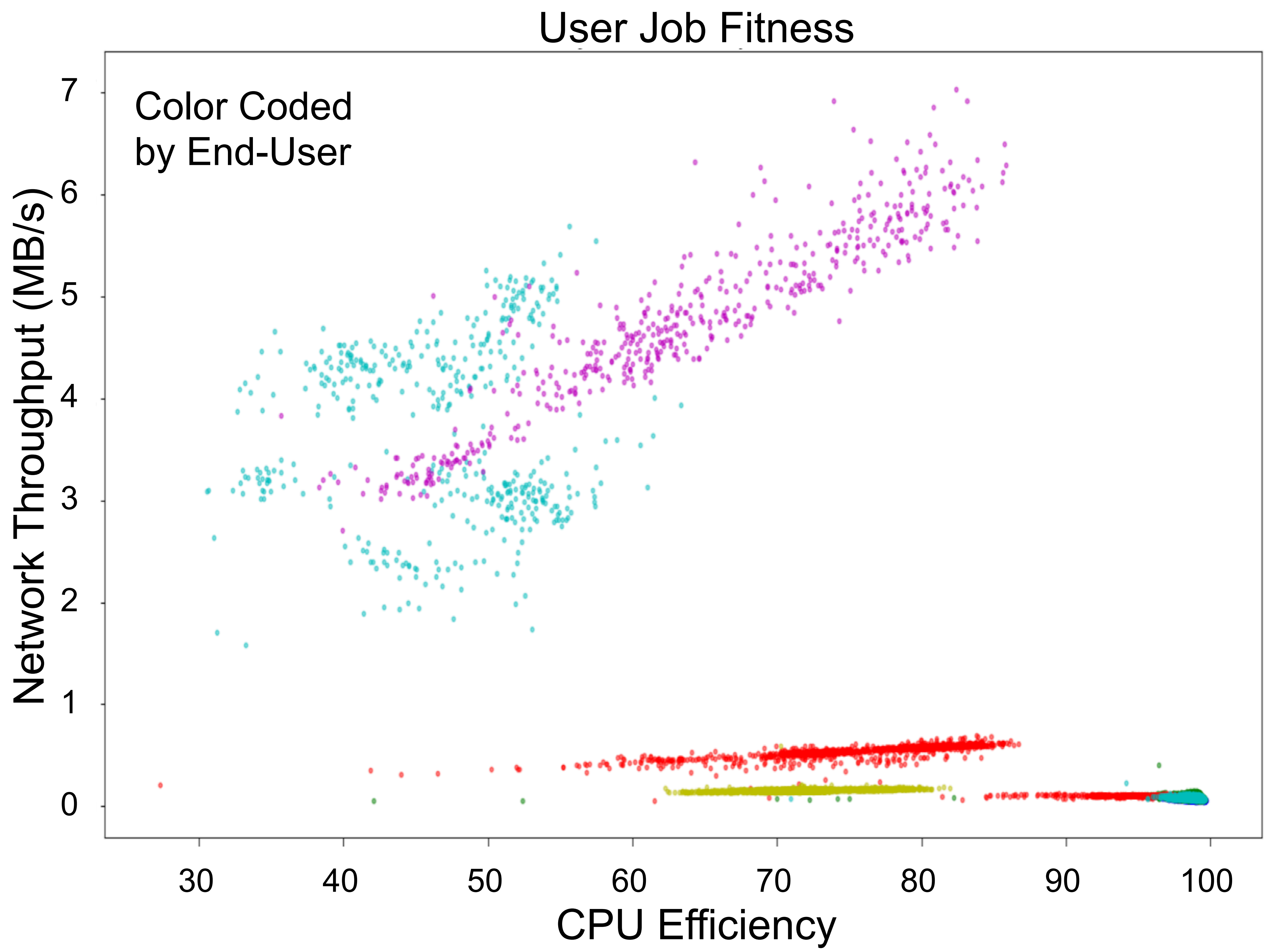
Classical Job to Resource to Job meta-scheduler:

- 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

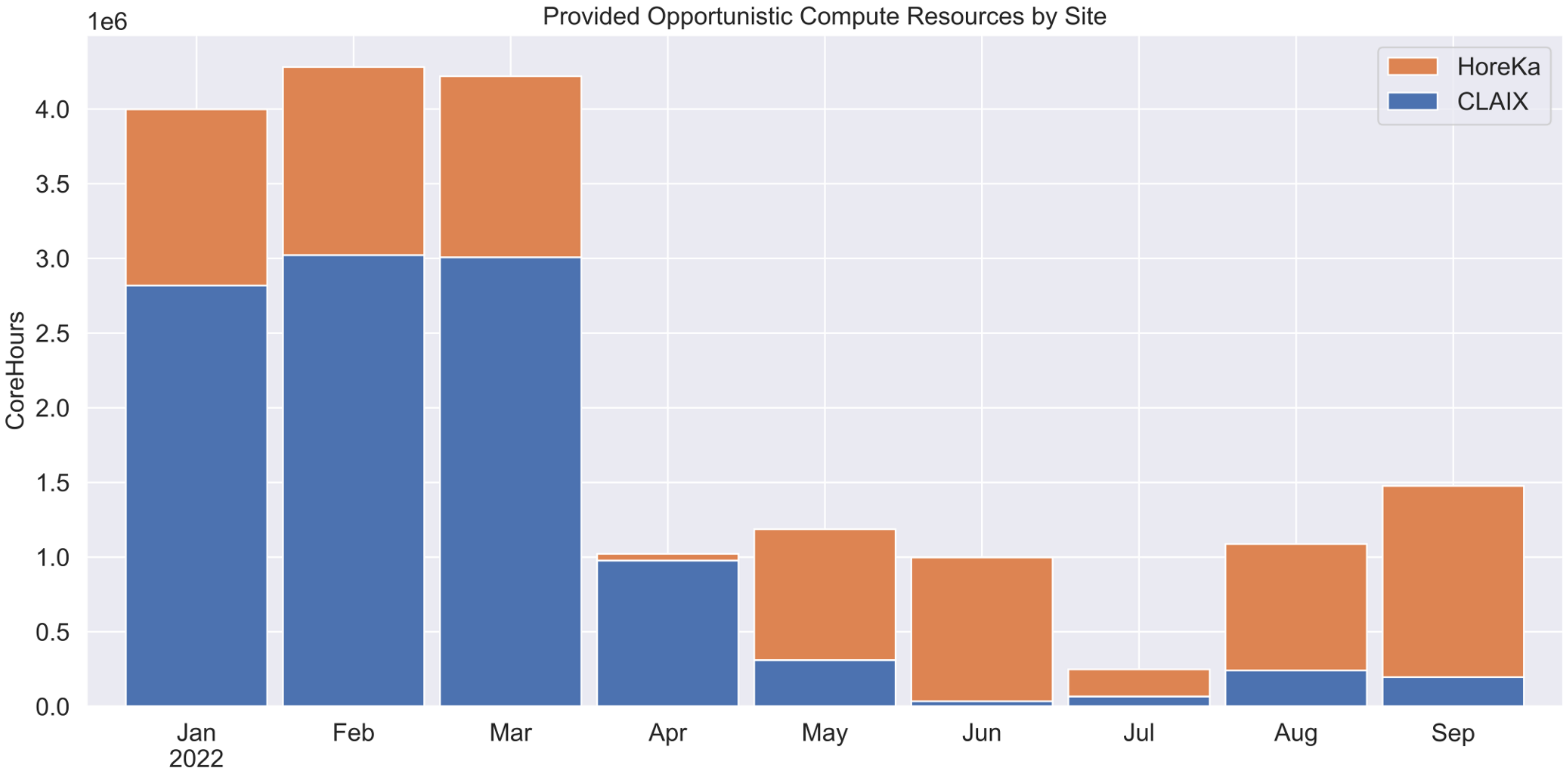
- Respect network availability and congestion for provisioning
  - Congested network is the bottleneck for opportunistic resources
  - Non-linear interference and noticeable measurement overhead



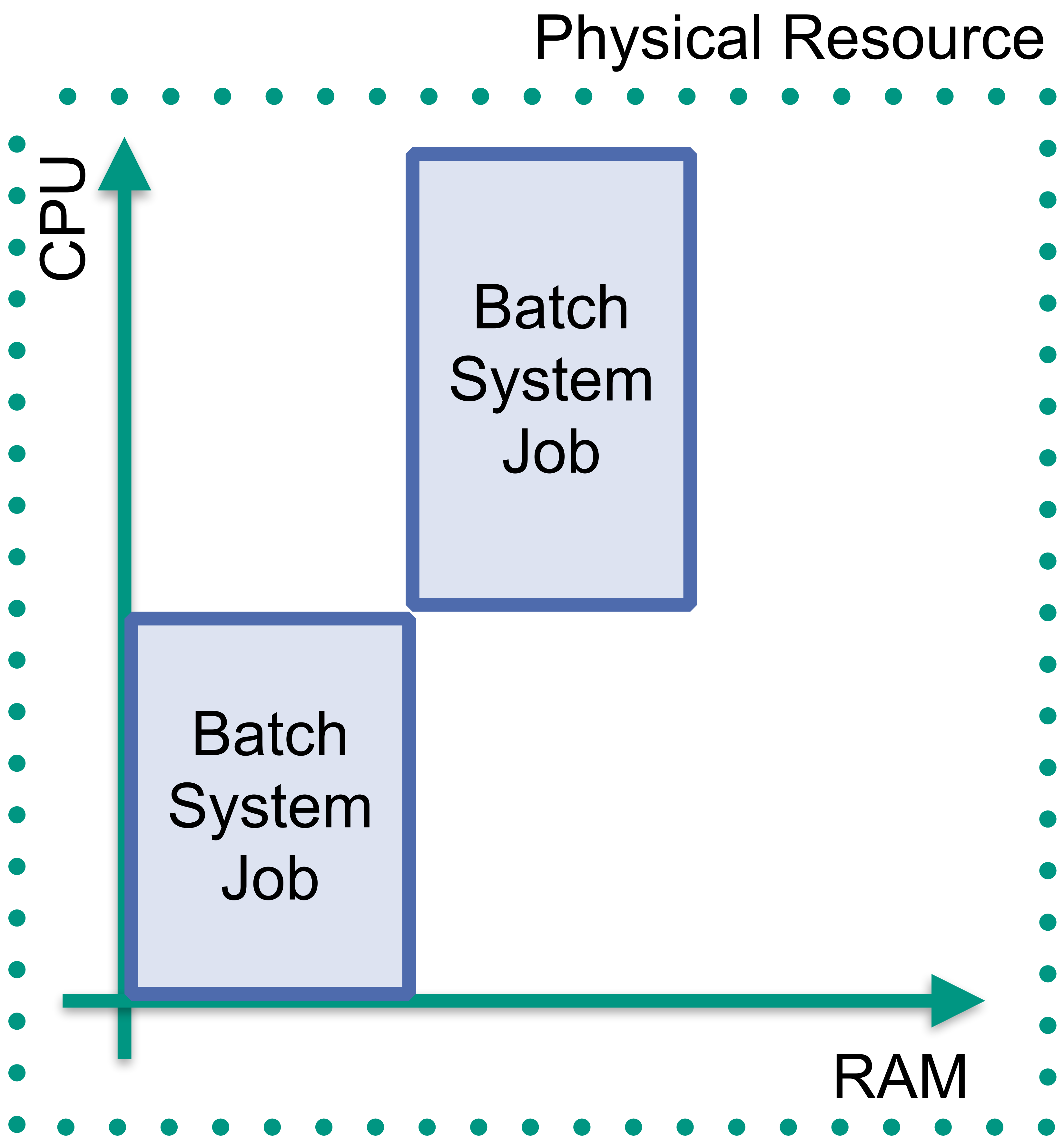
Implicit Resource Scheduling via Feedback

- 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

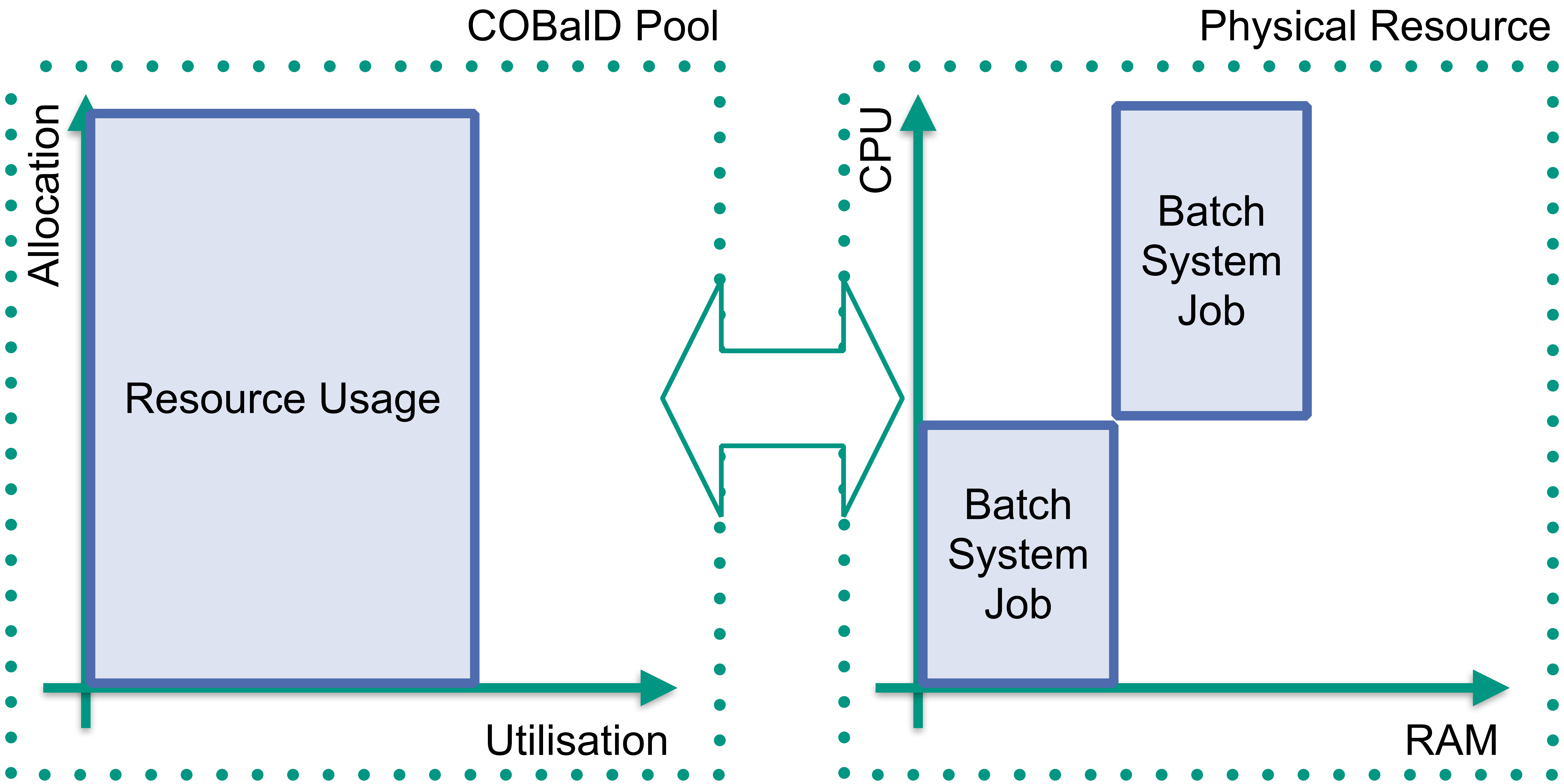
NHR WLCG Contributions



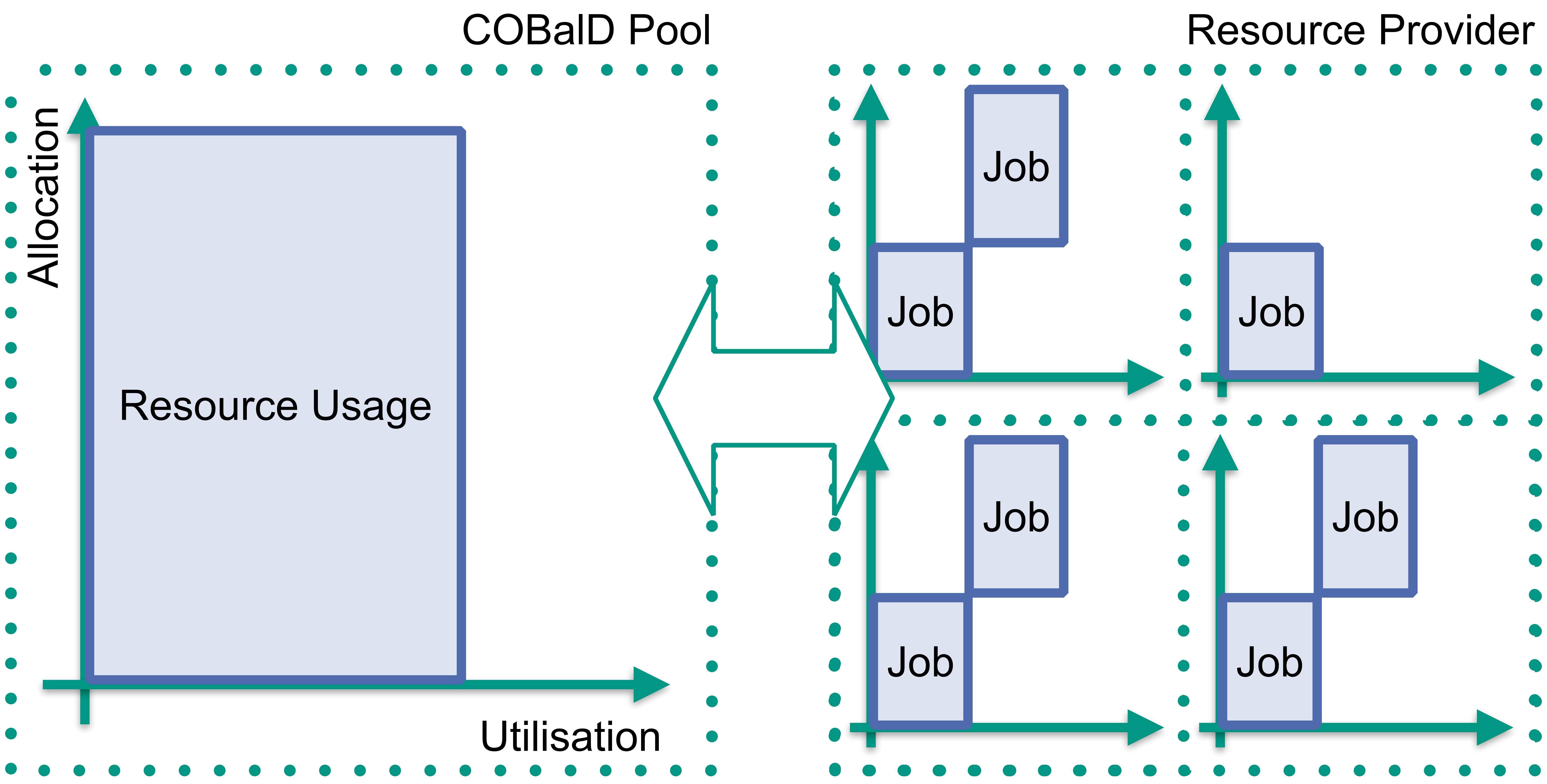
COBalD Resource Pool Model



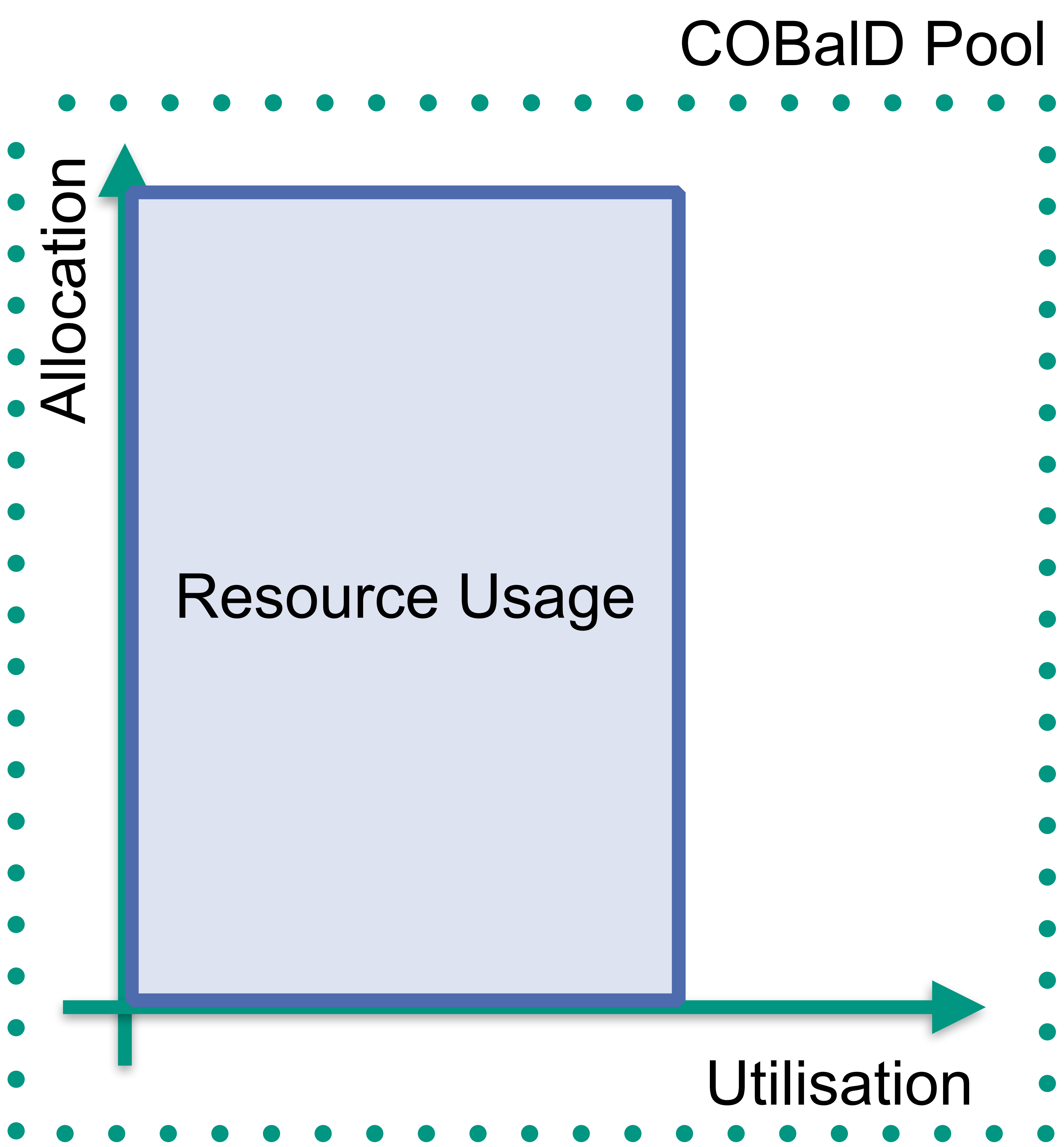
COBalD Resource Pool Model



COBalD Resource Pool Model



COBalD 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
```