Karlsruhe Institute of Technology

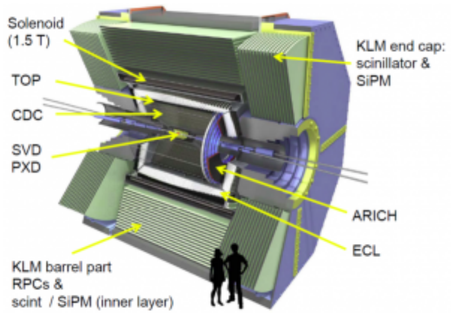
Dynamic Federation of Heterogeneous Compute Resources in High Energy Physics & Beyond

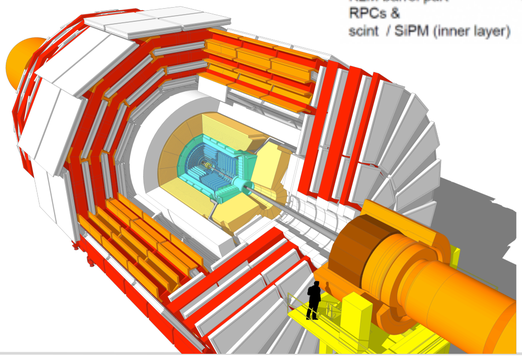
PUNCH-Lunch, 16.09.2021

Manuel Giffels & Kilian Schwarz

The High Energy Physics Use-case

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

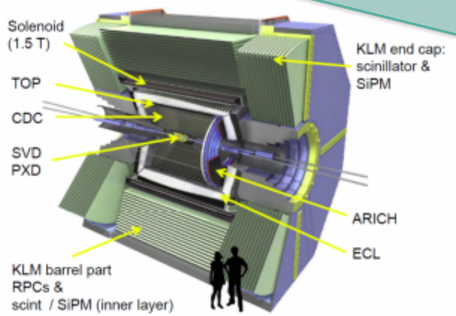


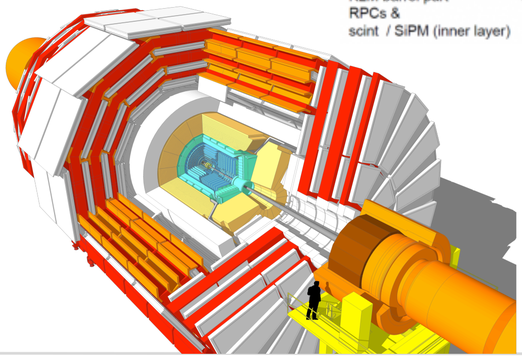


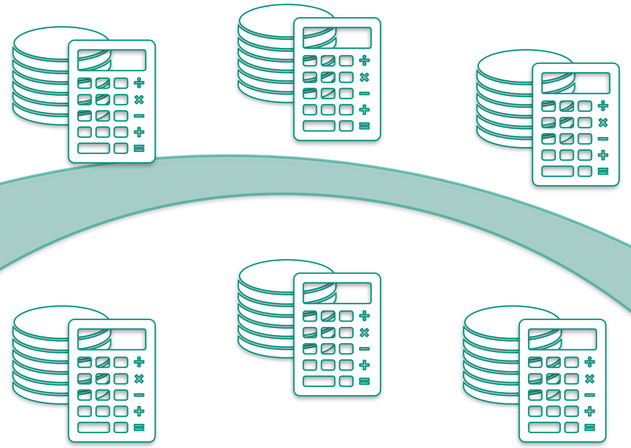
Slide by Max Fischer

The High Energy Physics Use-case

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





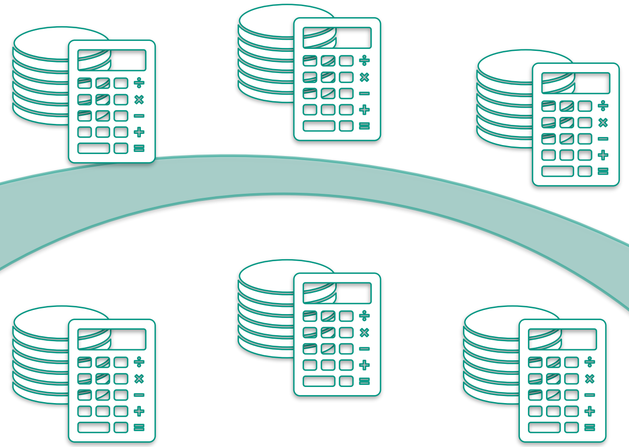


- Collaborators provide storage/compute centres
- Resources shared via a worldwide computing Grid

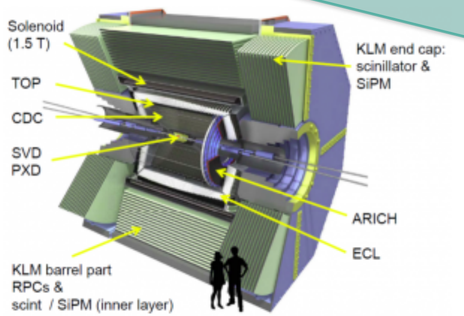
Slide by Max Fischer

The High Energy Physics Use-case

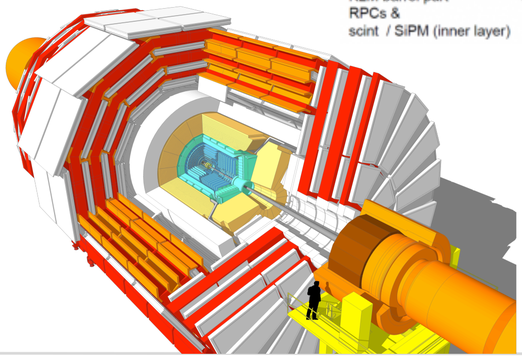
- Particle detectors record physics event data
- Each detector used by a collaboration of scientists

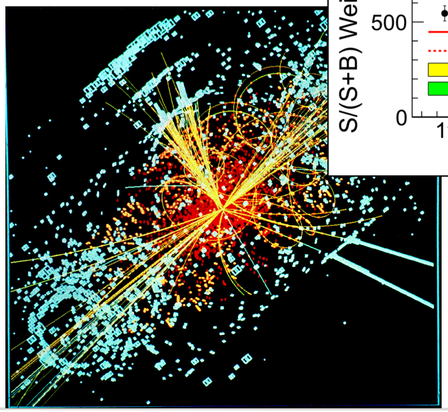


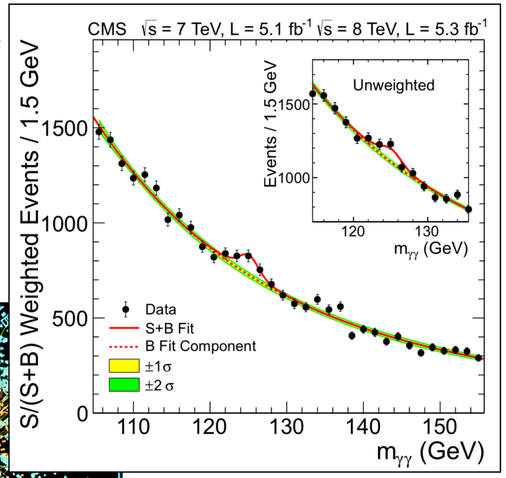
- Collaborations automate common pre-processing
- Scientists run individual end-user analyses



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

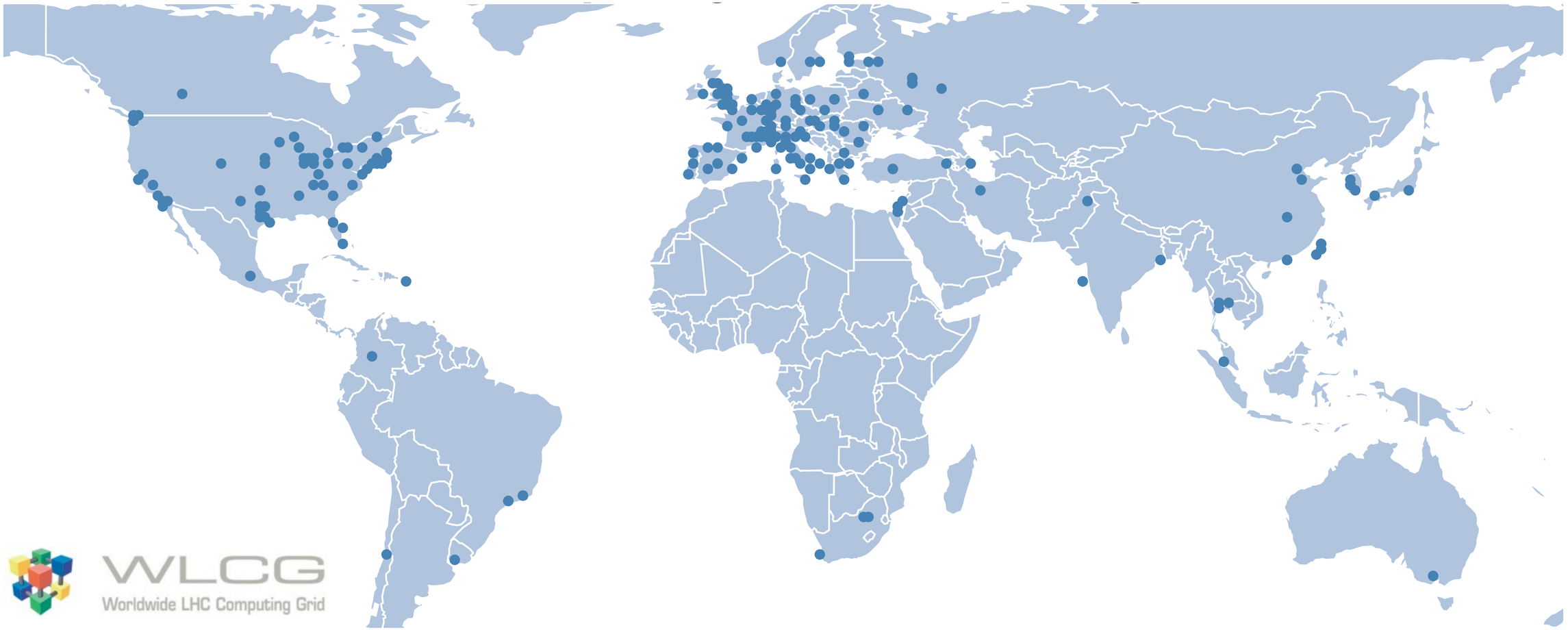






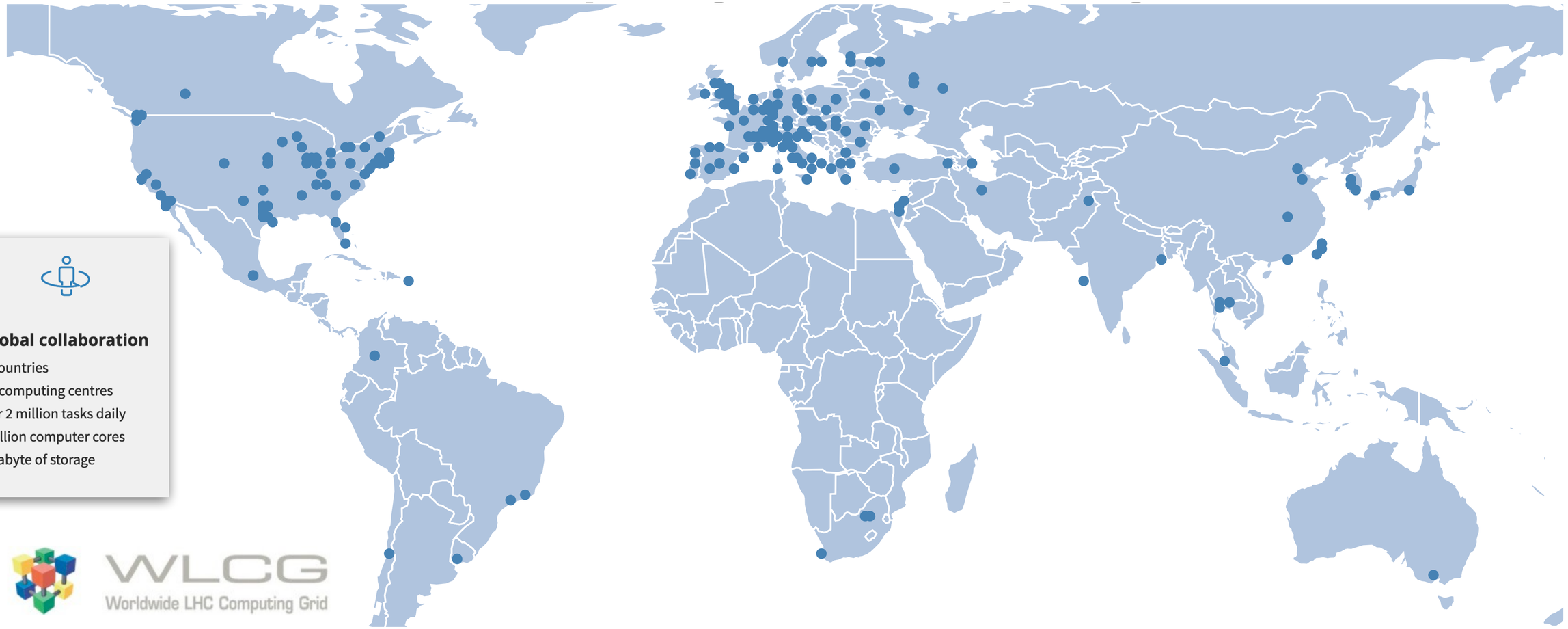
Slide by Max Fischer

Computing in High Energy Physics (HEP)



~20 years experience

Computing in High Energy Physics (HEP)



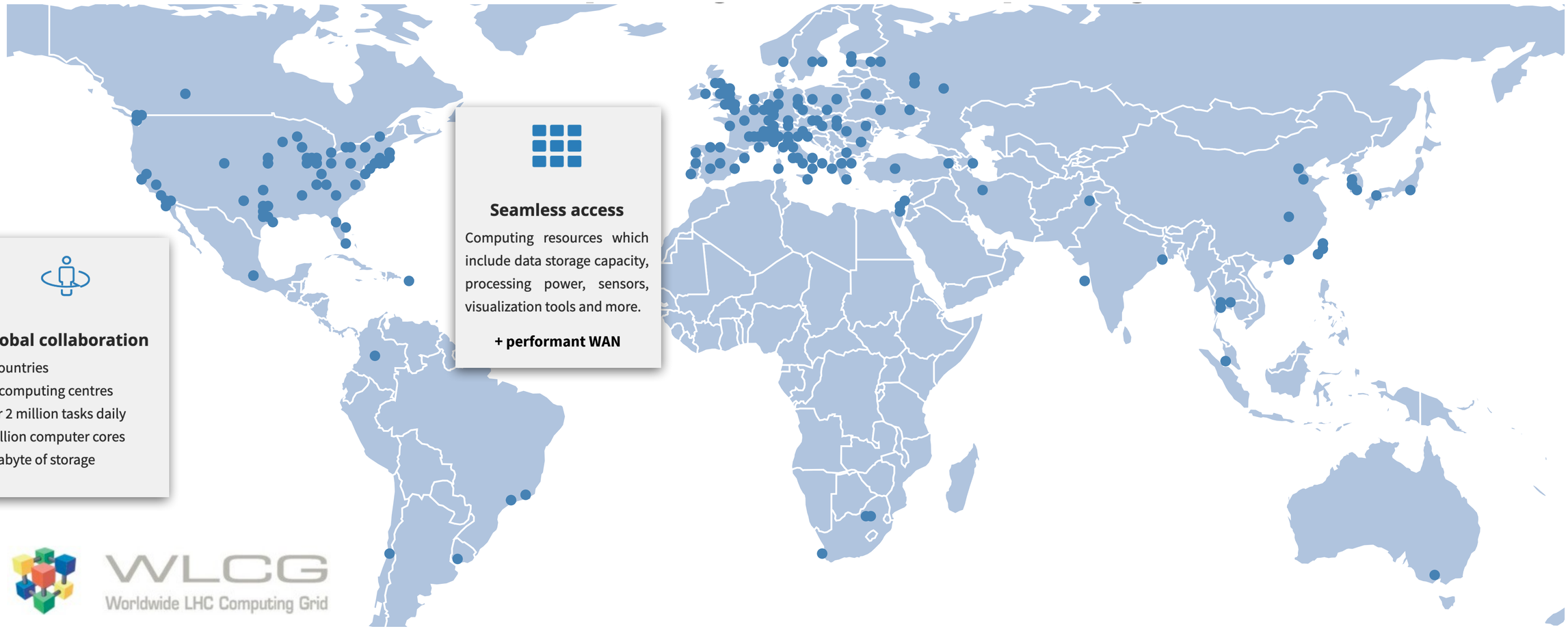
Global collaboration

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)



Global collaboration

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

Seamless access

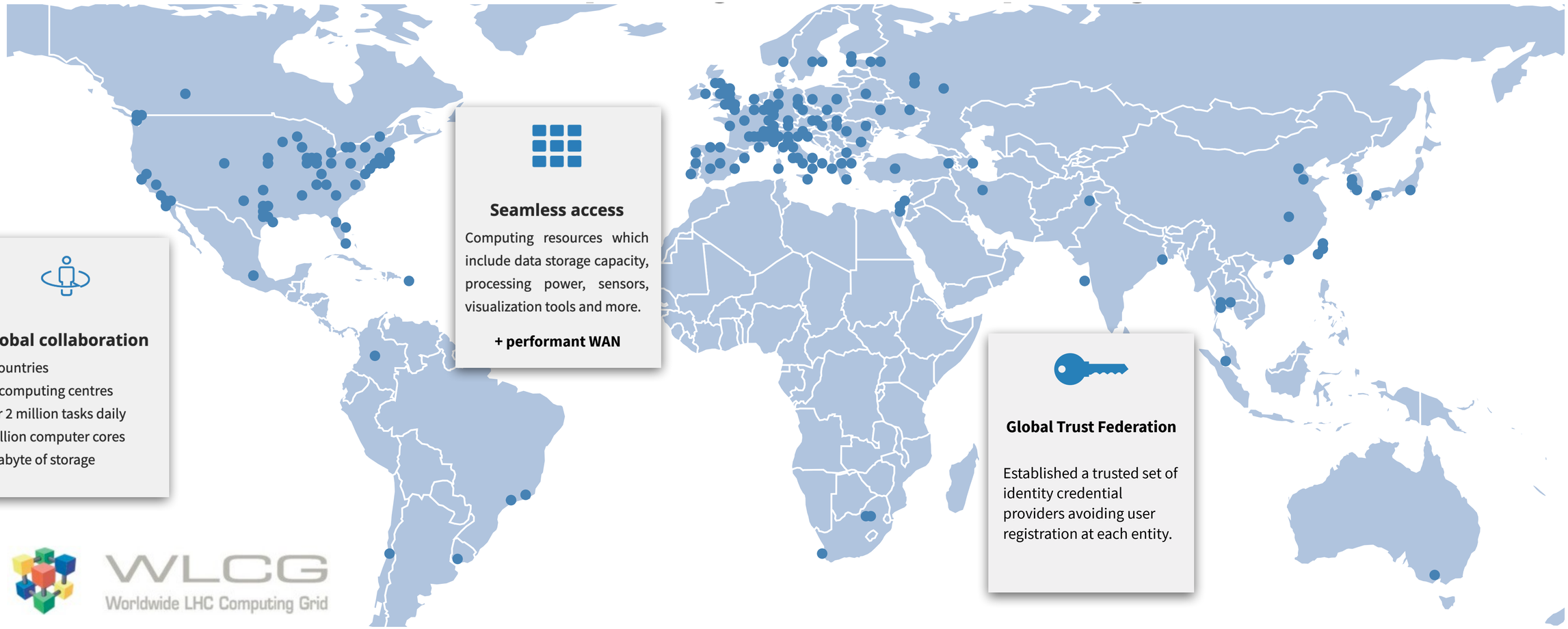
Computing resources which include data storage capacity, processing power, sensors, visualization tools and more.

+ performant WAN

WLCG
Worldwide LHC Computing Grid

~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

Seamless access

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

+ performant WAN

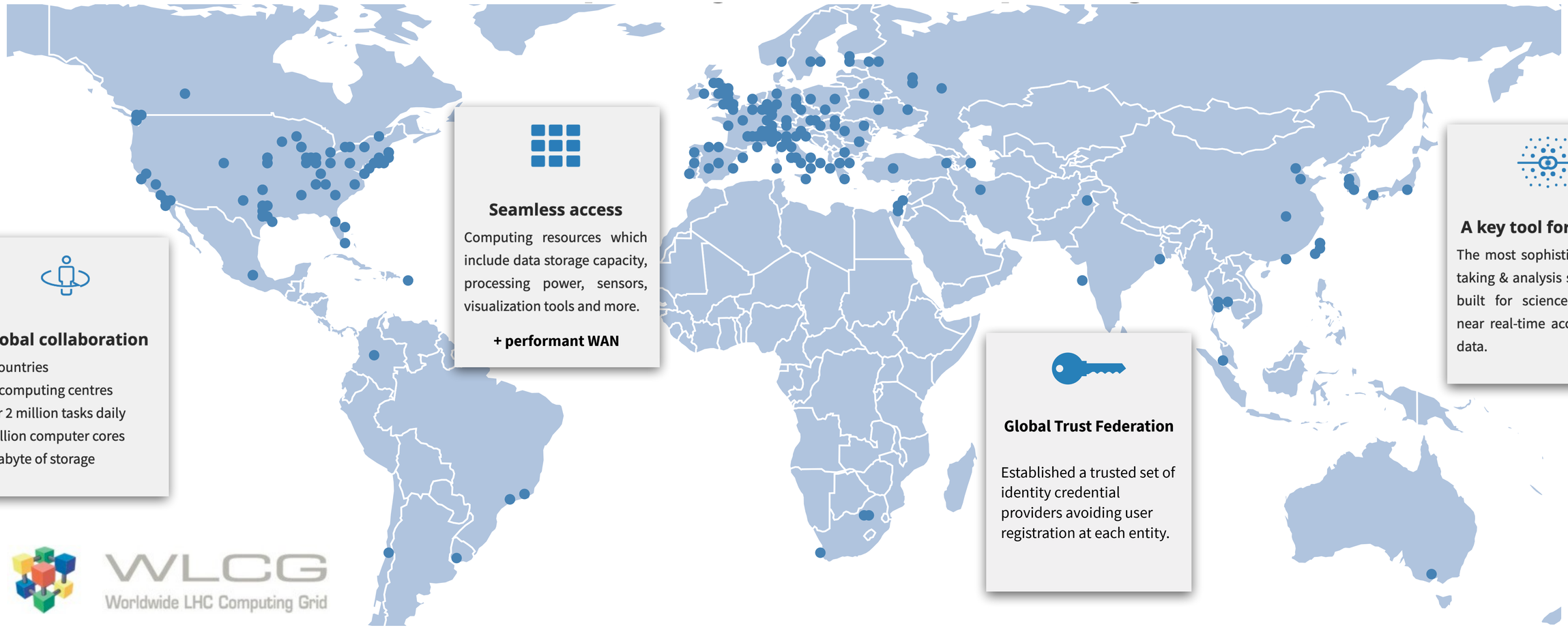
Global Trust Federation

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)



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.

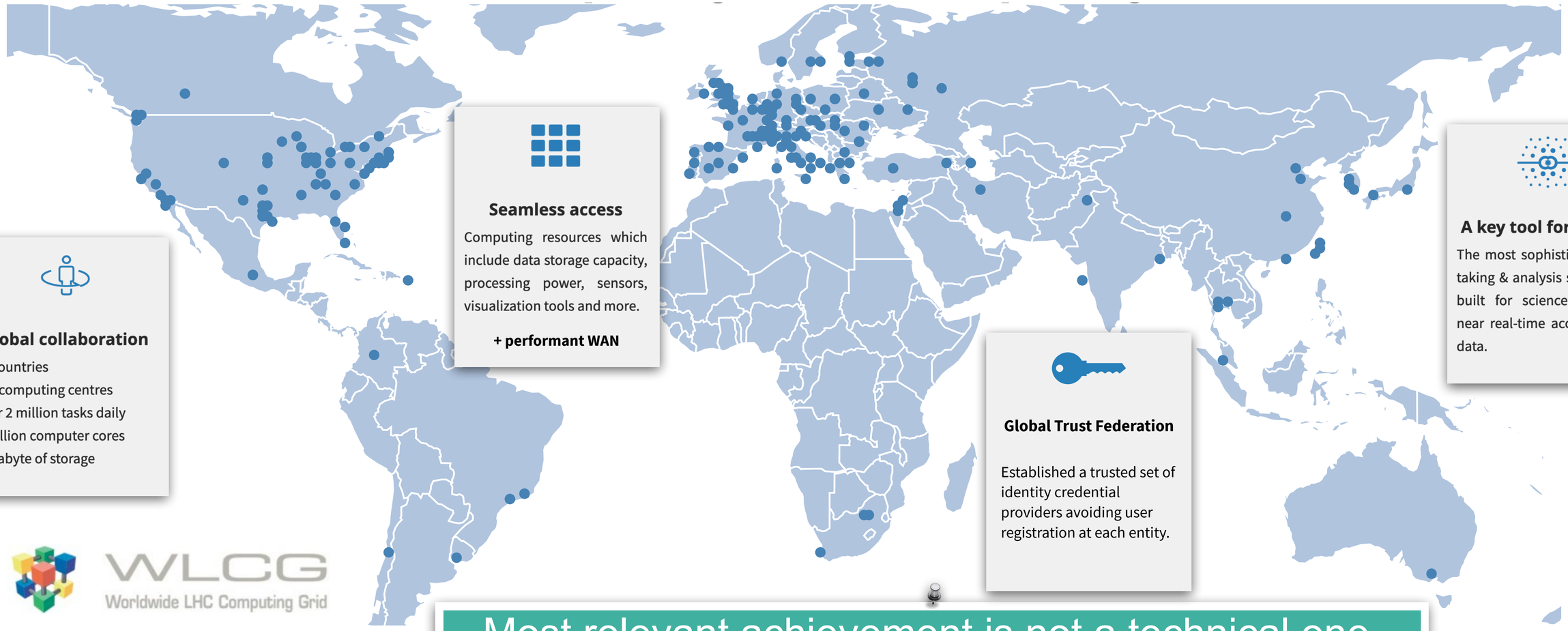
A key tool for physics

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

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

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

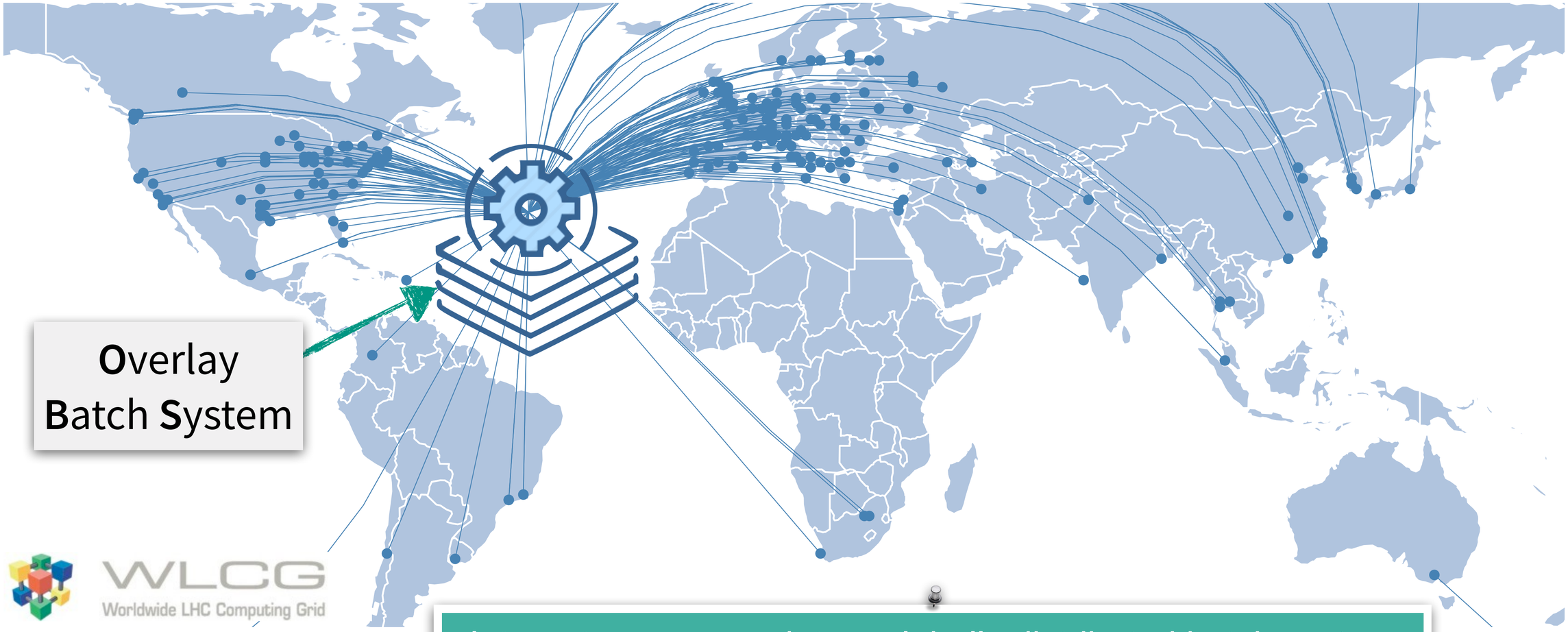
The most sophisticated data-taking & analysis system ever built for science, providing near real-time access to LHC data.

WLCG
Worldwide LHC Computing Grid

~20 years experience

Most relevant achievement is not a technical one, it is the establishment of the Global Trust Federation.

From Grid towards Global Distributed Computing



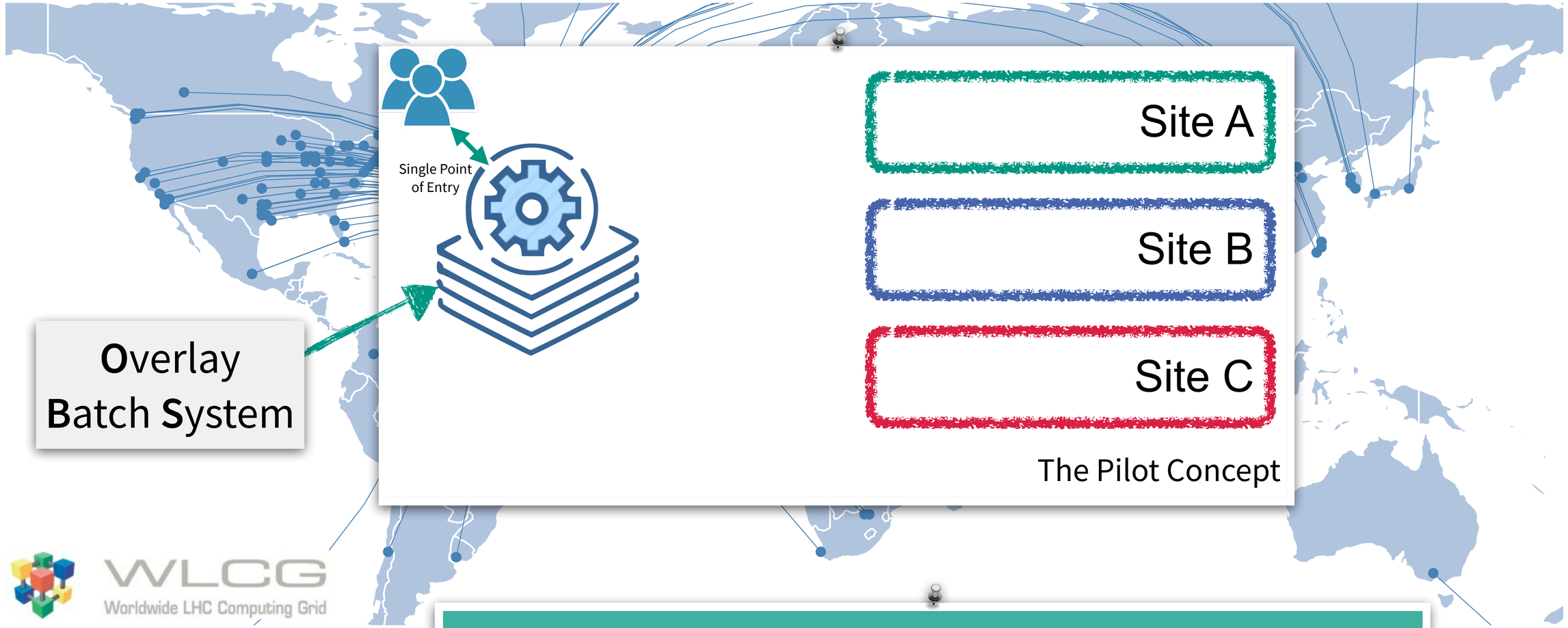
Overlay
Batch **S**ystem

WLCG
Worldwide LHC Computing Grid

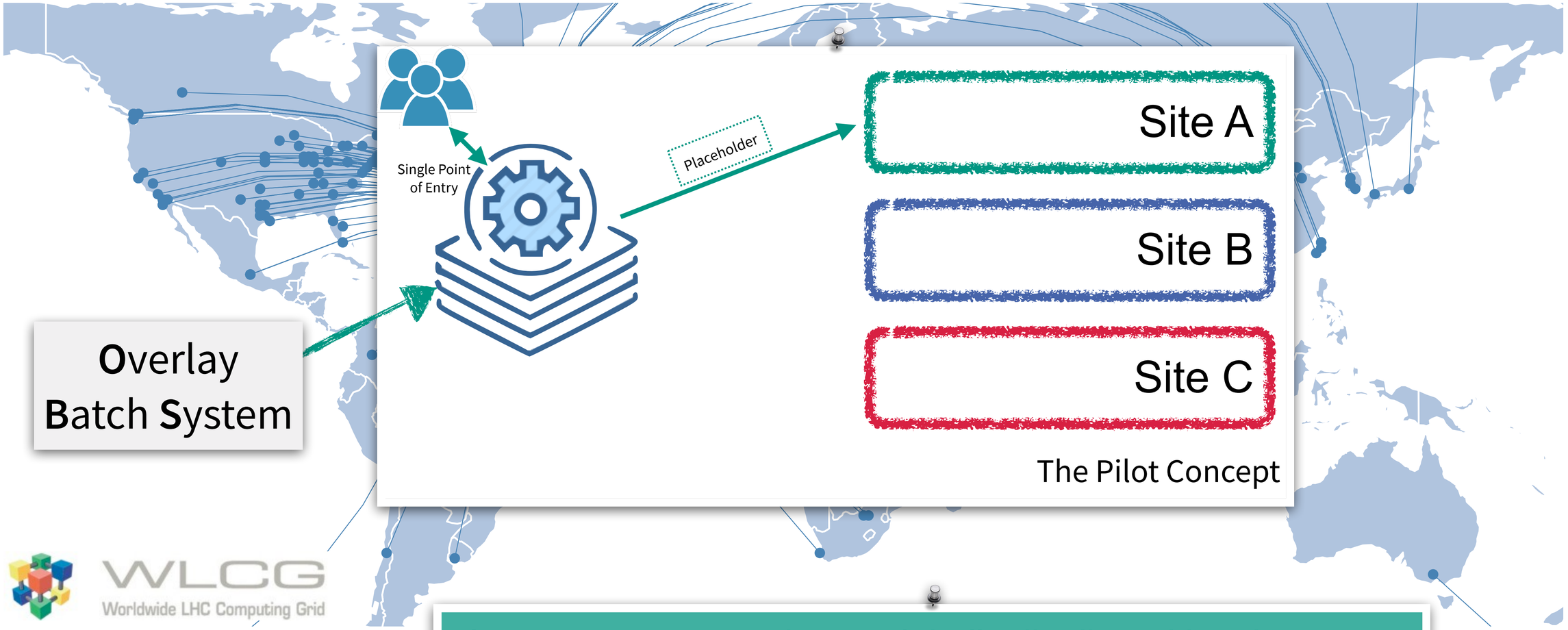
Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

Simplified view: Each experiment has one overlay batch system

From Grid towards Global Distributed Computing

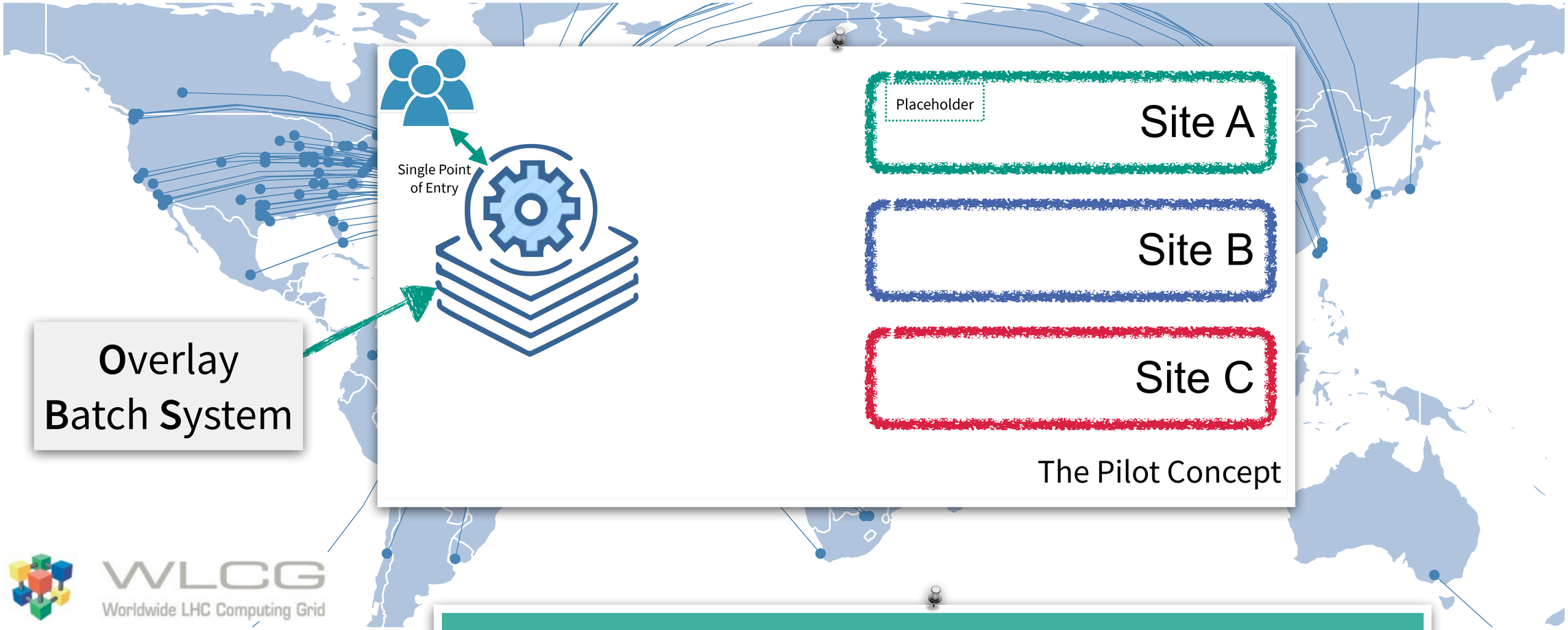


From Grid towards Global Distributed Computing



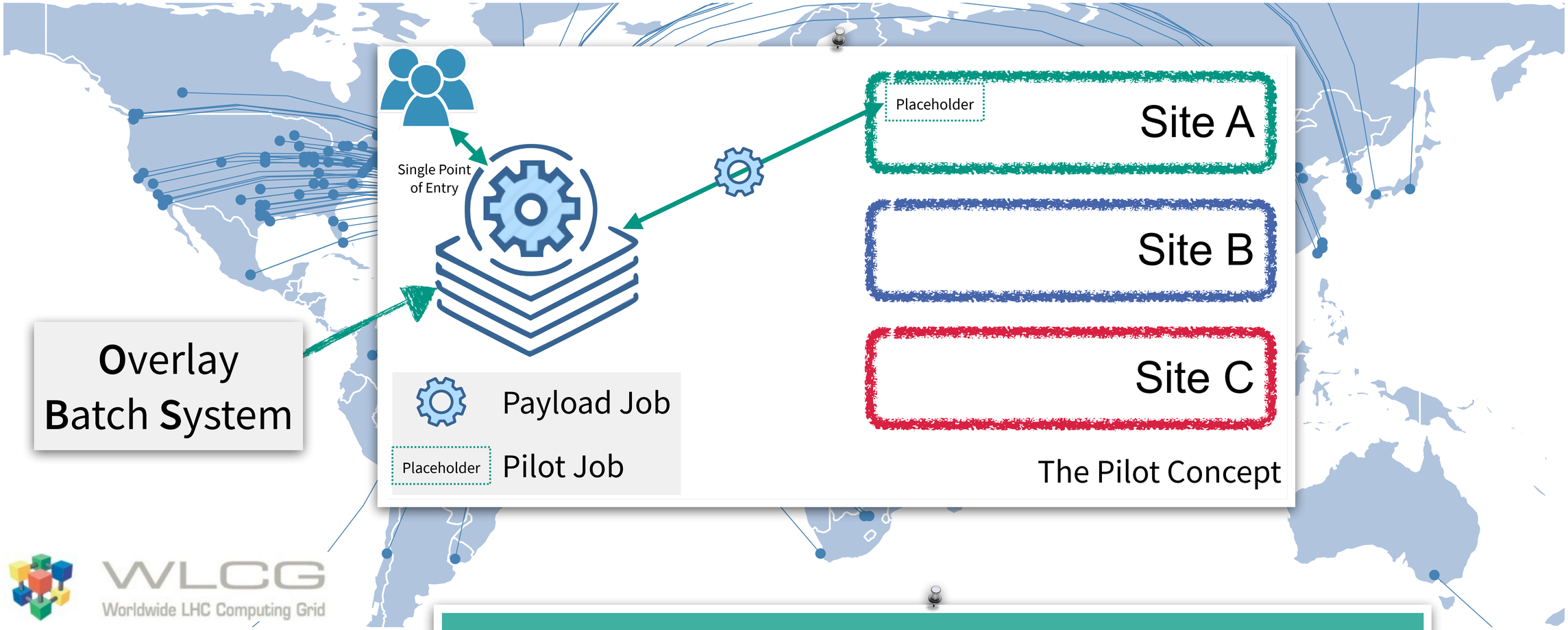
Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

From Grid towards Global Distributed Computing



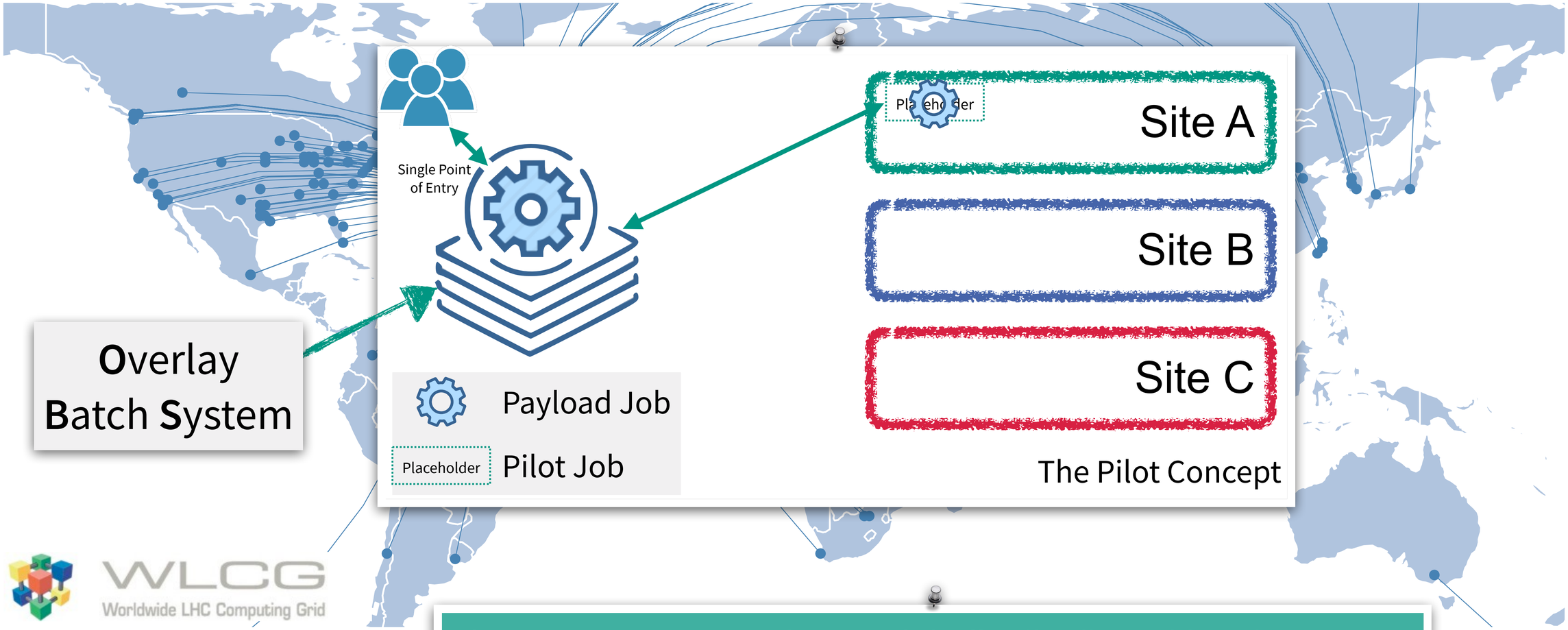
Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

From Grid towards Global Distributed Computing



Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

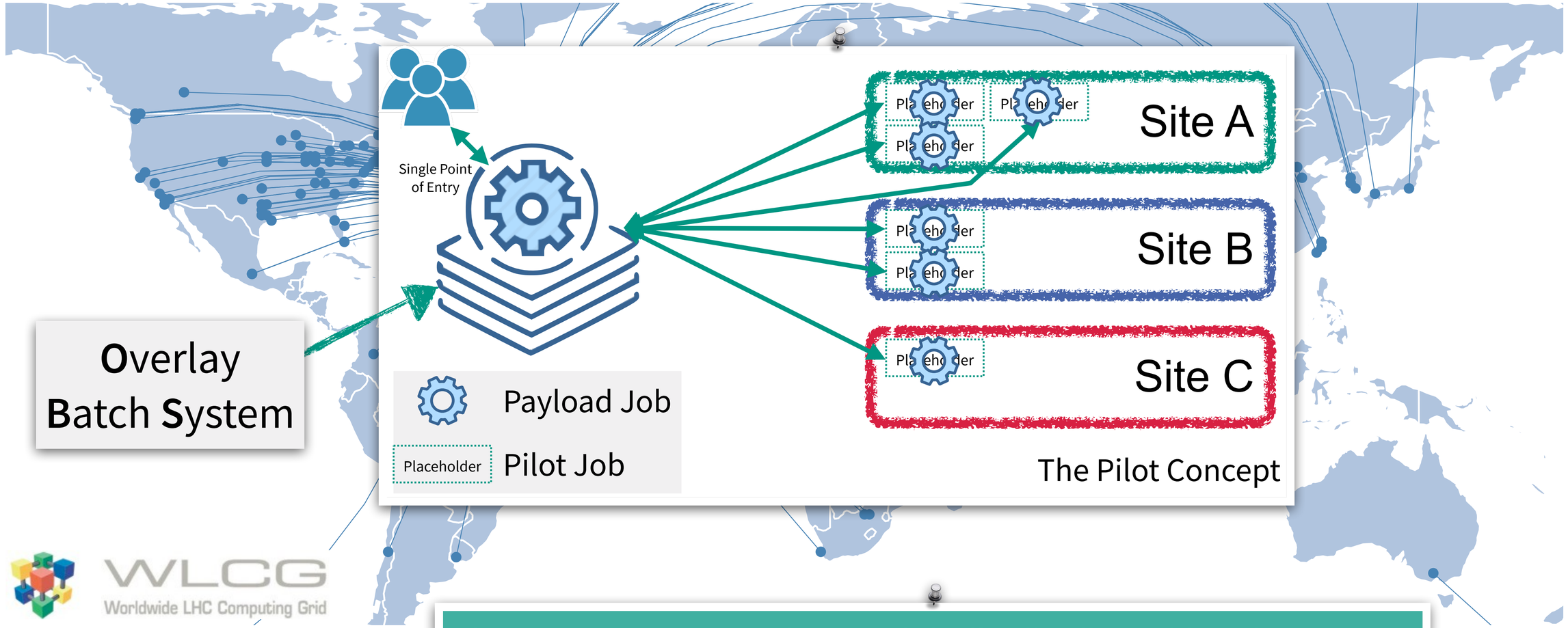
From Grid towards Global Distributed Computing



Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

Simplified view: Each experiment has one overlay batch system

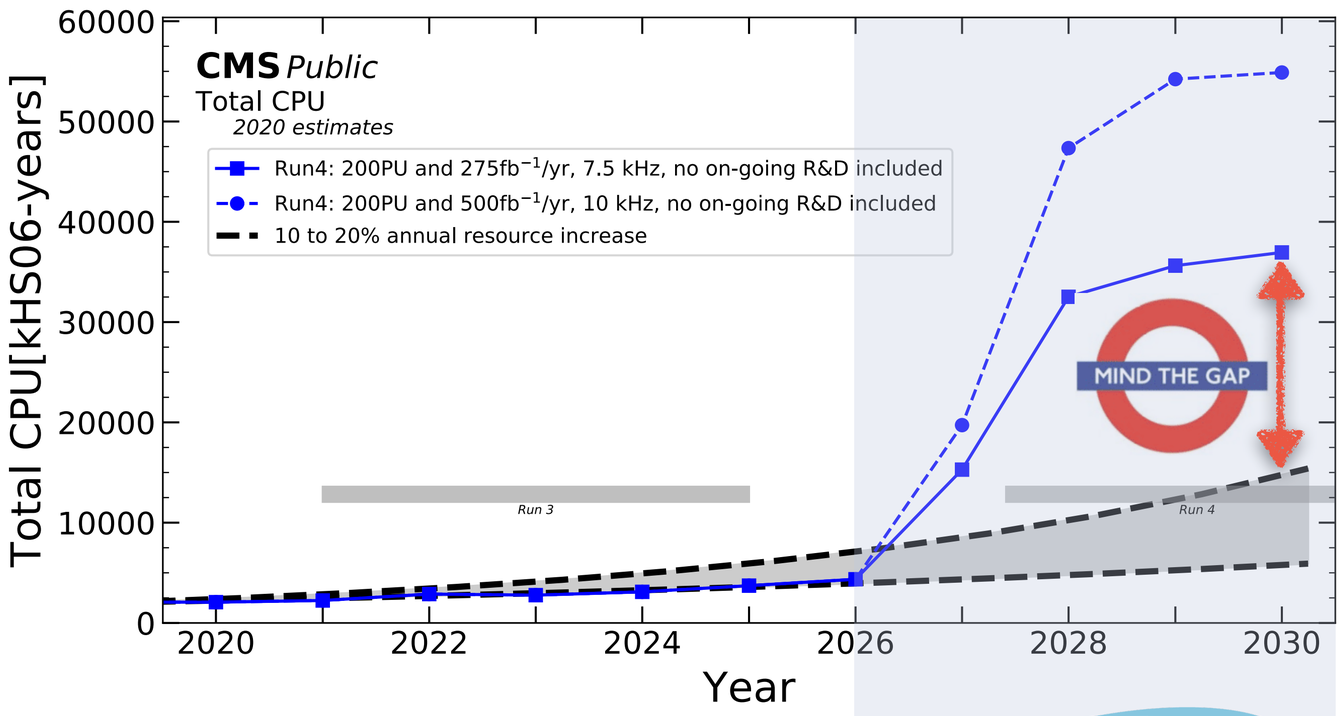
From Grid towards Global Distributed Computing



Integrate resources into a globally distributed batch system and remove a significant fraction of the initial Grid middleware

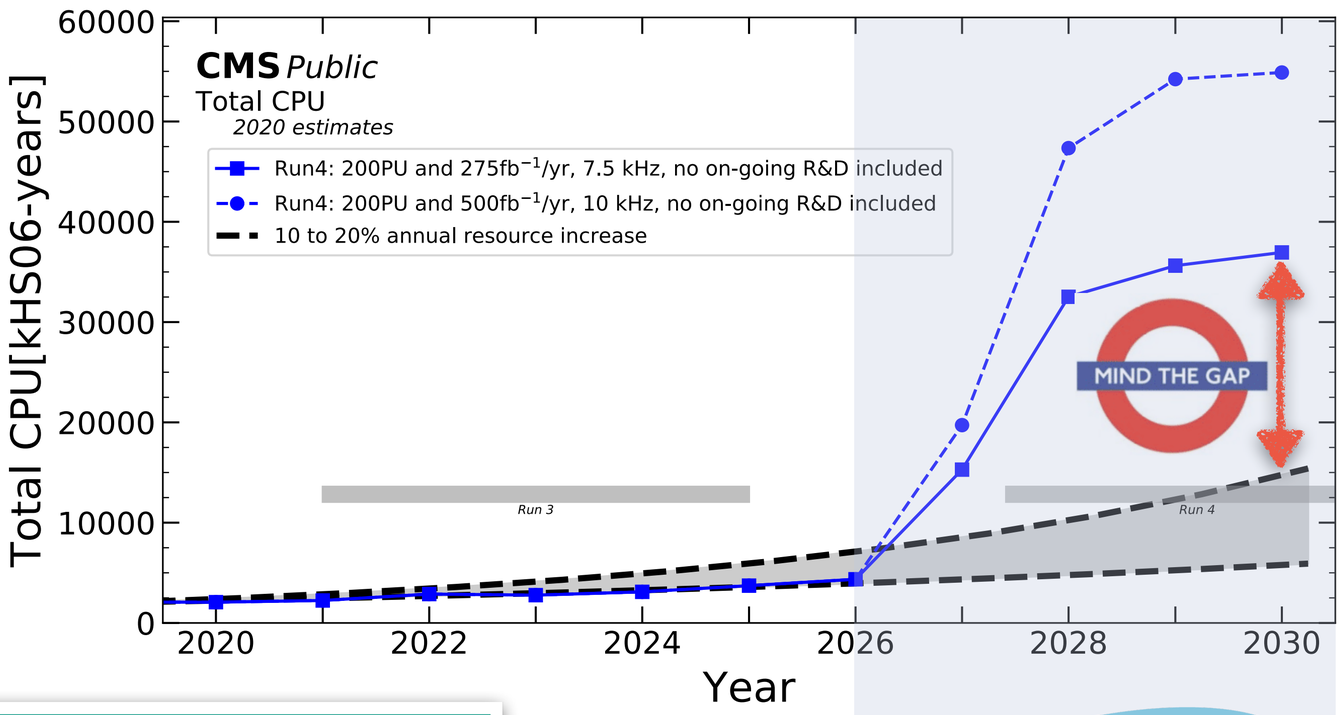
Upcoming Computing Challenges in HEP

- HL-LHC poses unprecedented challenges to HEP computing
- Assuming flat budget and 10-20% technology advance per year
- CPU shortcoming of factor 2 - 3 estimated in 2027
- ➡ Critical conditions for HL-LHC (Run 4)



Upcoming Computing Challenges in HEP

- HL-LHC poses unprecedented challenges to HEP computing
- Assuming flat budget and 10-20% technology advance per year
- CPU shortcoming of factor 2 - 3 estimated in 2027
- ➡ Critical conditions for HL-LHC (Run 4)

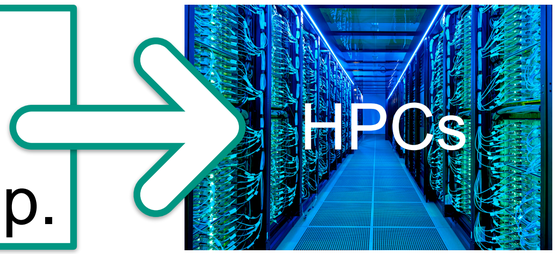


Potential Measures:

- Software & Computing Evolution (R&D)
- Add additional resources (opportunistic resources)

Opportunistic Resources & WLCG

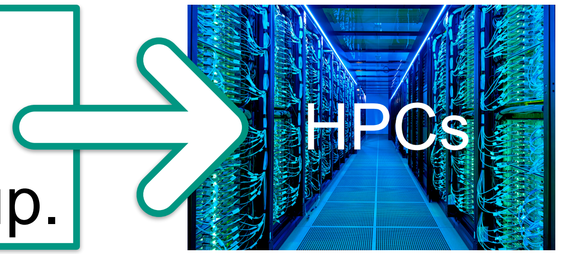
Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.



+ Clouds

Opportunistic Resources & WLCG

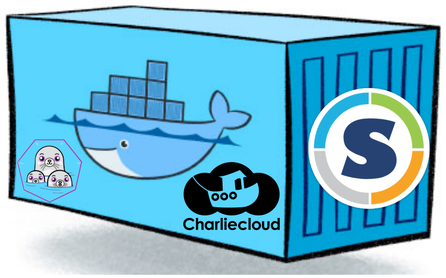
Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.



+ Clouds

Challenges:

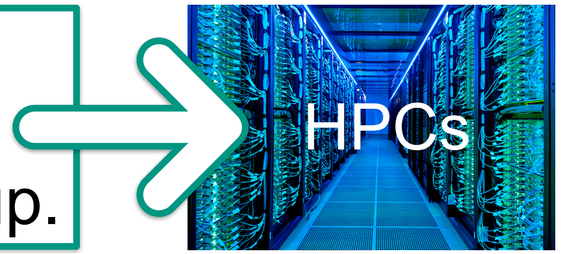
- Different OS & Software availability



Container Technology

Opportunistic Resources & WLCG

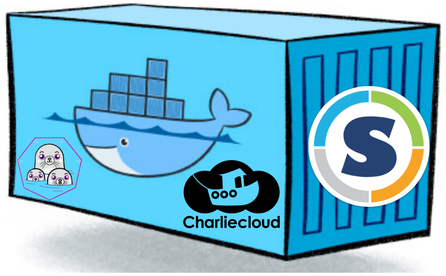
Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.



+ 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

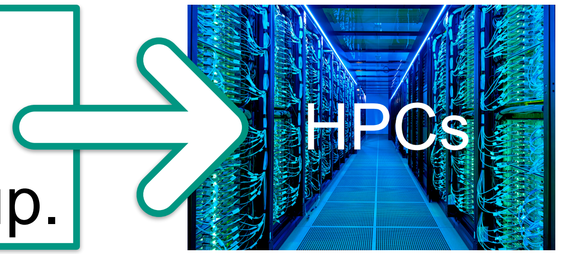


Container Technology

Resource Scheduler

Opportunistic Resources & WLCG

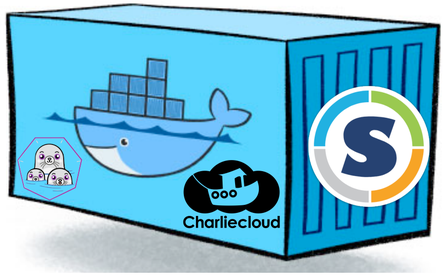
Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.



+ 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
- No global trust federation/Grid entry point available



Container Technology

Resource Scheduler

OBS + Pilots
in more generalized way

Opportunistic Resources & WLCG

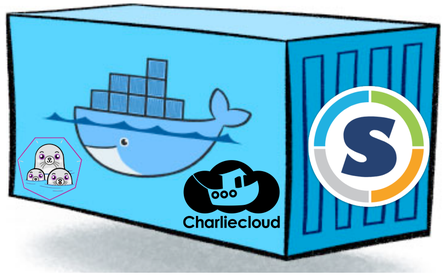
Opportunistic Resources
Any resources not permanently dedicated to but temporarily available for a specific task, user or group.

+ Clouds

Challenges:

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

What about storage?
➜ See Kilian's part



Container Technology

Resource Scheduler

OBS + Pilots
in more generalized way

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]

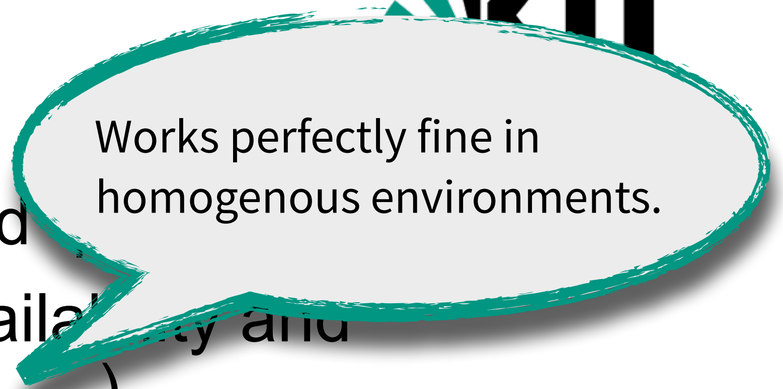
- Resource Meta-Scheduling for Job Scheduler is a „hard“ problem
- Usually based on predictions of the future resource availability and future mixture of job classes (CPU intense, I/O intense, …)

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 and future mixture of job classes (CPU intense, I/O intense, …)



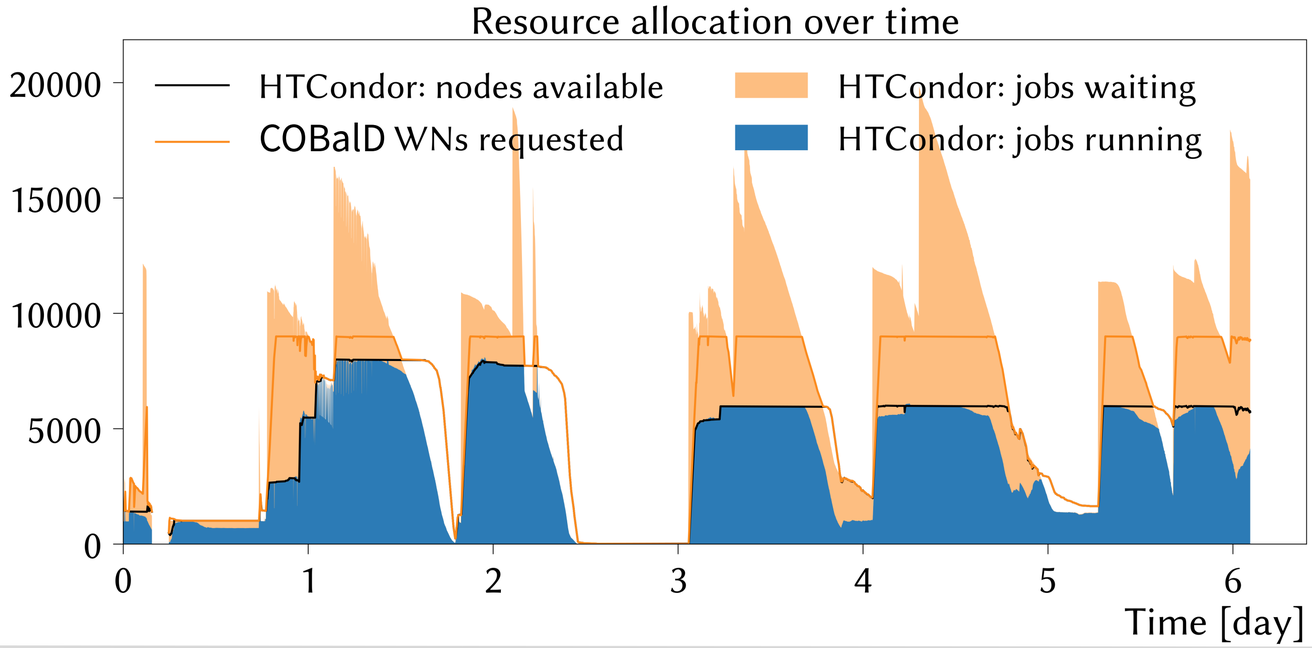
Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

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

Works perfectly fine in homogenous environments.

- 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, …)
- **However:** We usually care only about a simpler problem!



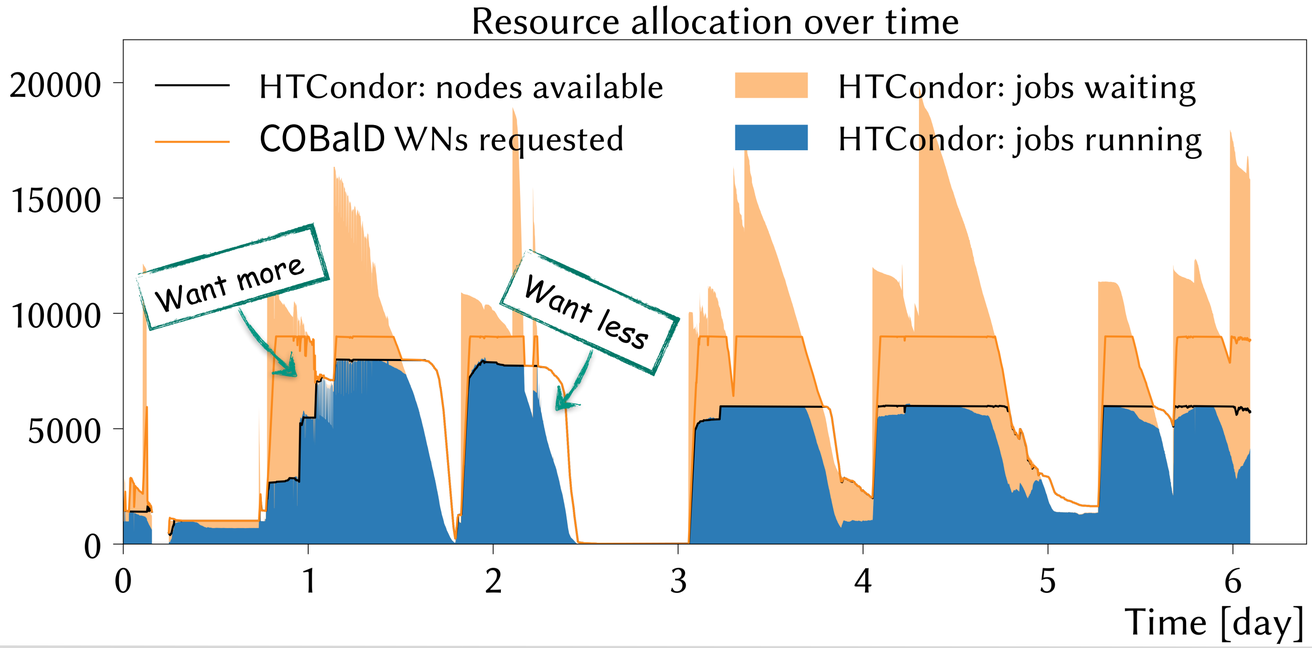
Based on a slide by Max Fischer

The COBalD View of Resource Scheduling

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

Works perfectly fine in homogenous environments.

- 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, …)
- **However:** We usually care only about a simpler problem!

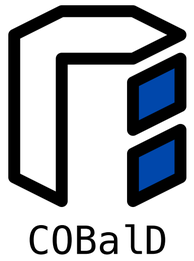


Based on a slide by Max Fischer

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

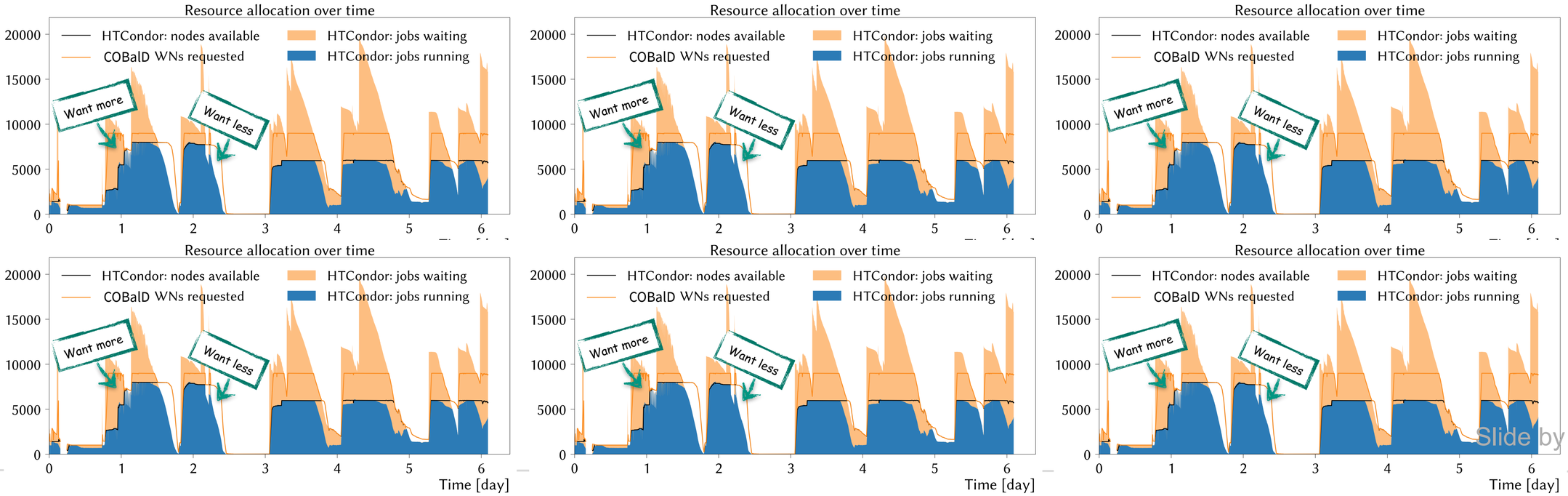


Slide by Max Fischer

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

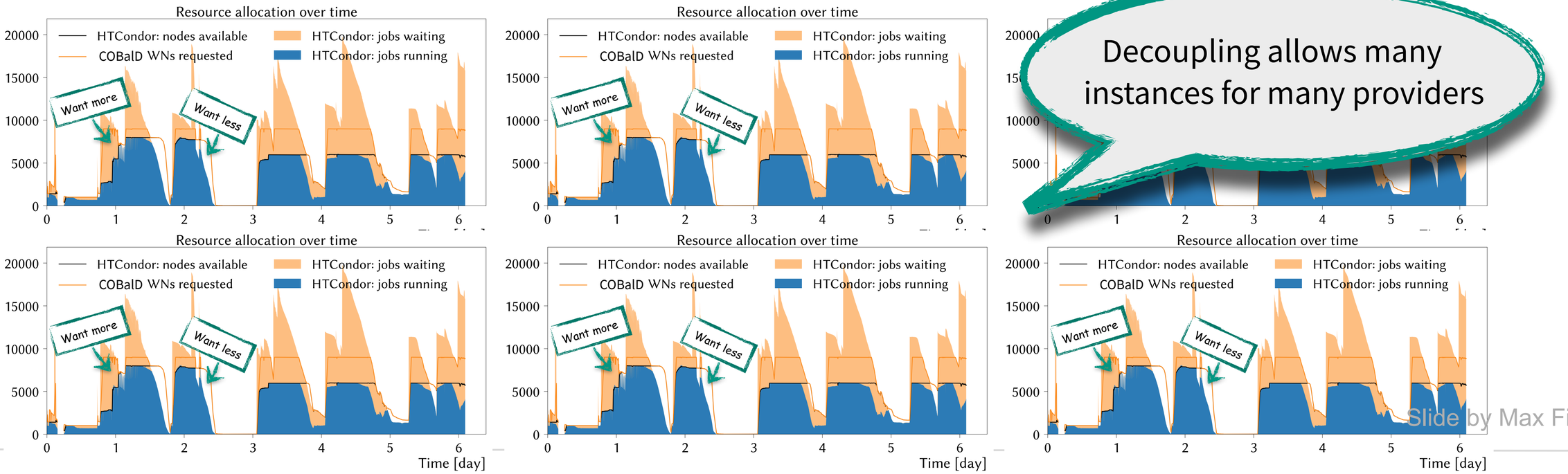


Slide by Max Fischer

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



Slide by Max Fischer

The COBalD/TARDIS - Resource Scheduler

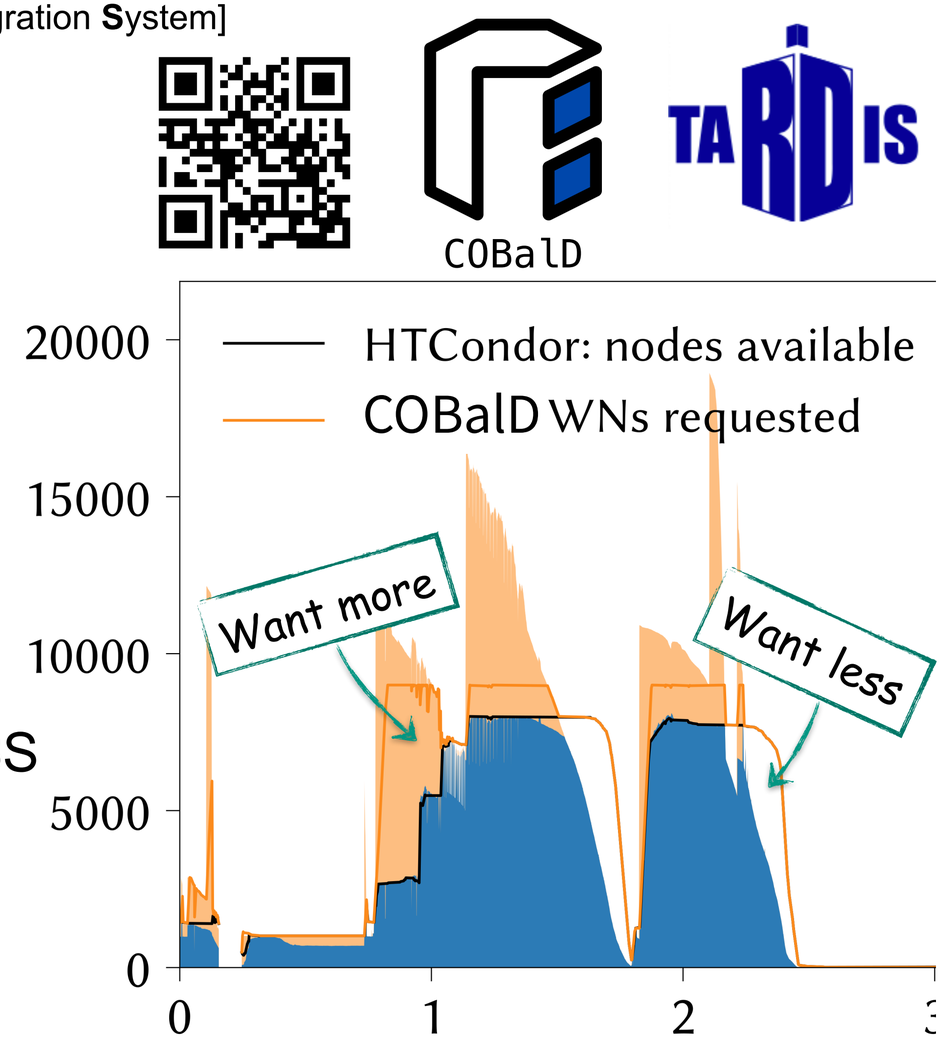
[COBalD - the **O**pportunistic **Bal**ancing **D**aemon] [**T**ransparent **A**daptive **R**esource **D**ynamic **I**ntegration **S**ystem]

COBalD:

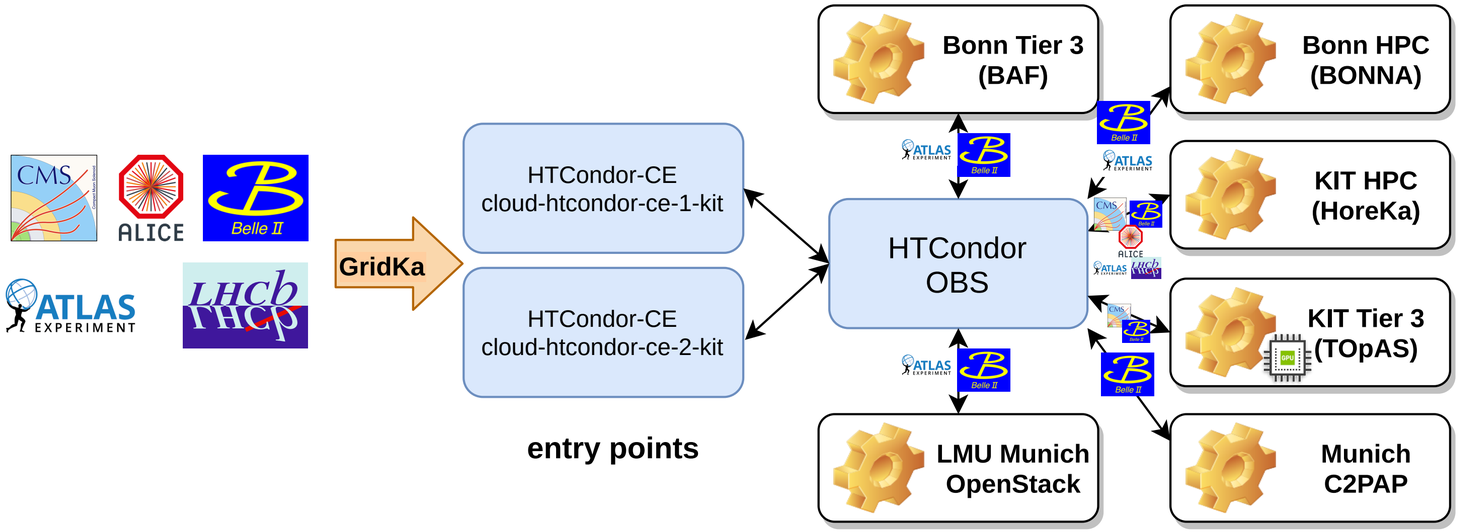
- Look at what is used, not what is requested
  - Simple logic: more used, less unused resources
  - COBalD acquires/releases resources
  - Batch system scheduler handles jobs

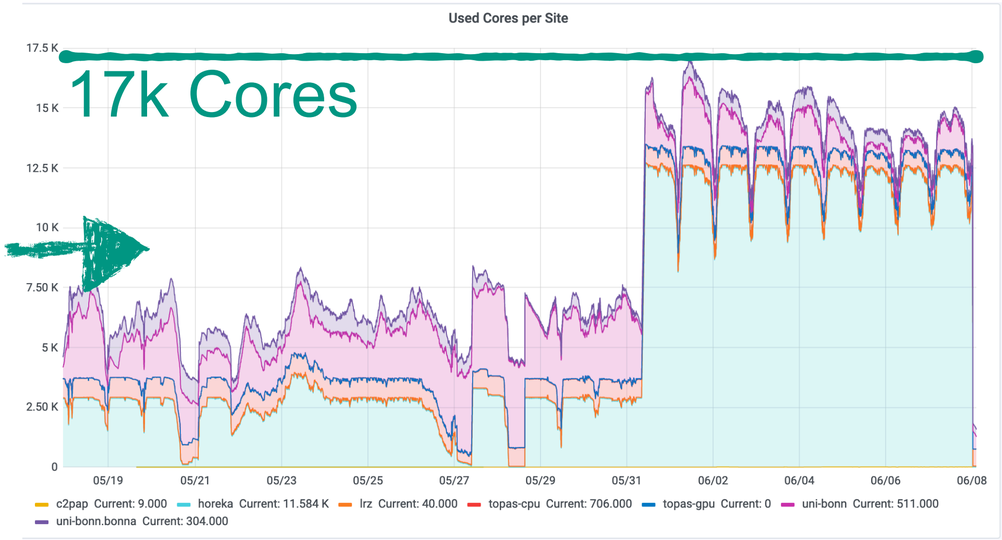
TARDIS (a COBalD plugin):

- Defines VM/Container/Job(Script) as resource
- Provides access to resource provider APIs
  - OpenStack, CloudStack, HTCondor, Slurm, Moab and K8S
- Integrates resources into Overlay Batch System
  - HTCondor and Slurm are supported
- Manages resource life cycle

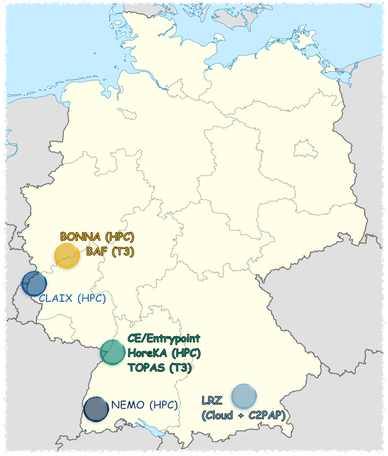


Opportunistic Resources & WLCG in Practice



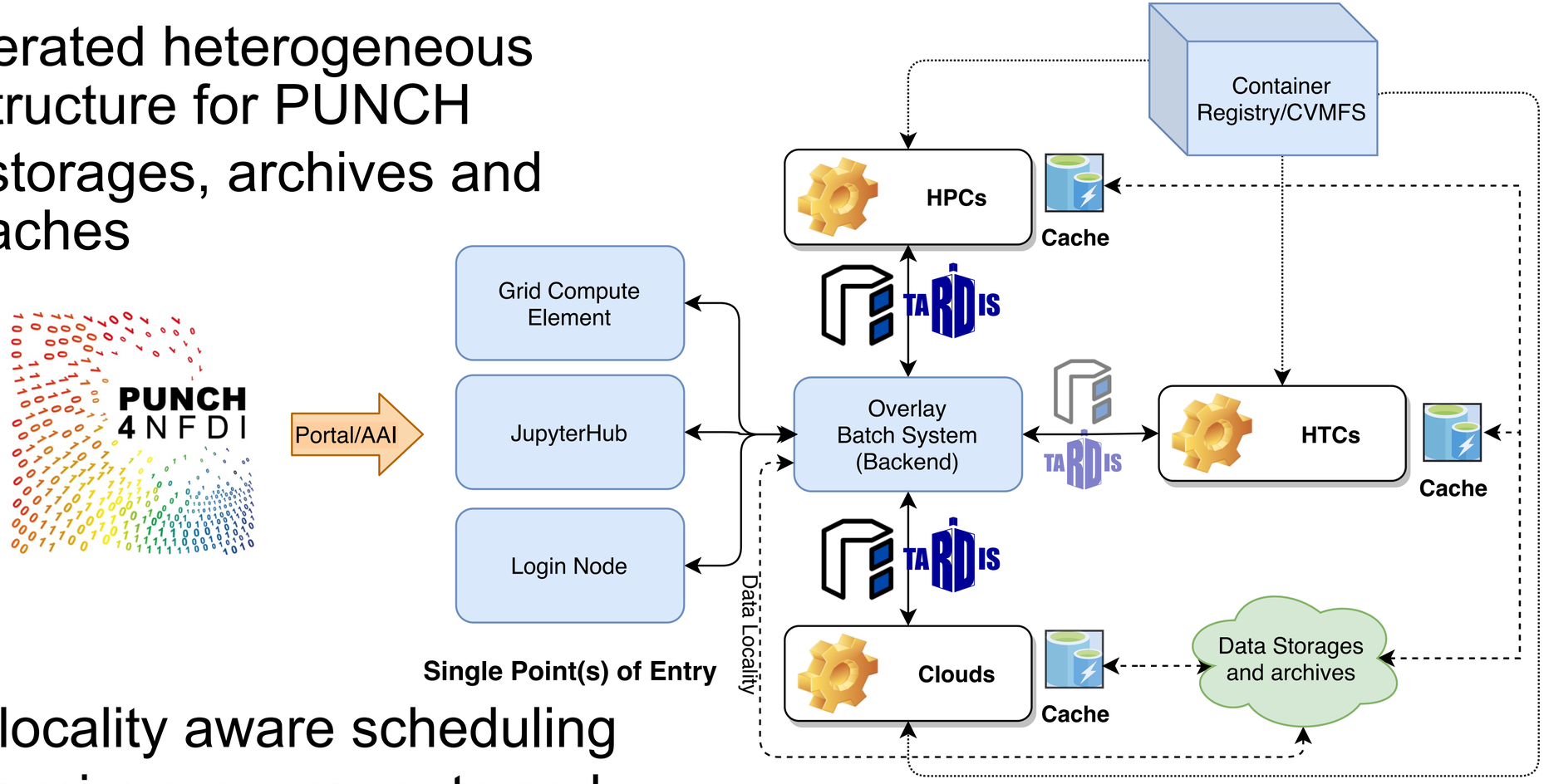


- Dynamic, transparent and on-demand integration using COBalD/TARDIS resource manager developed at KIT
- Single point(s) of entry to opportunistic resources (HPCs, local clusters and Clouds) using traditional Grid Compute Elements
- HTCondor overlay batch system to federate a variety of compute resources



Outlook: Towards the Compute4PUNCH Infrastructure

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

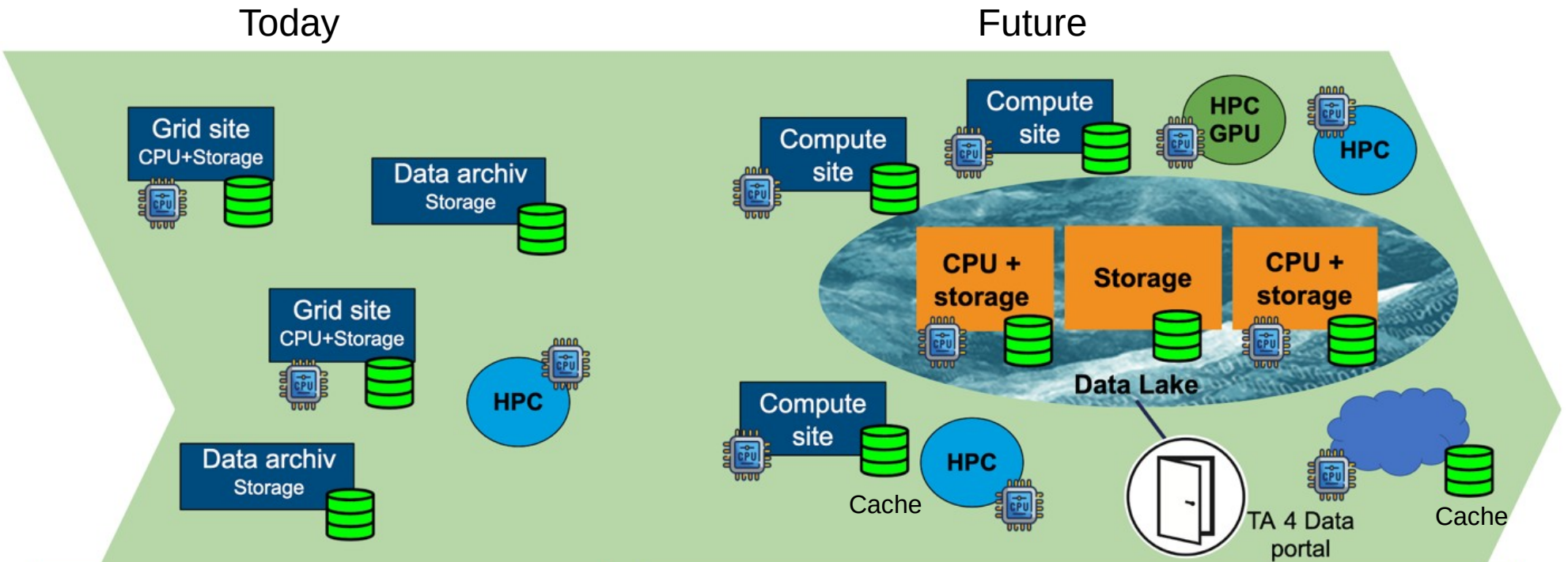


- Introduce data-locality aware scheduling
- Benefit from experiences, concepts and tools available in HEP community

Opportunistic Resources

Kilian Schwarz

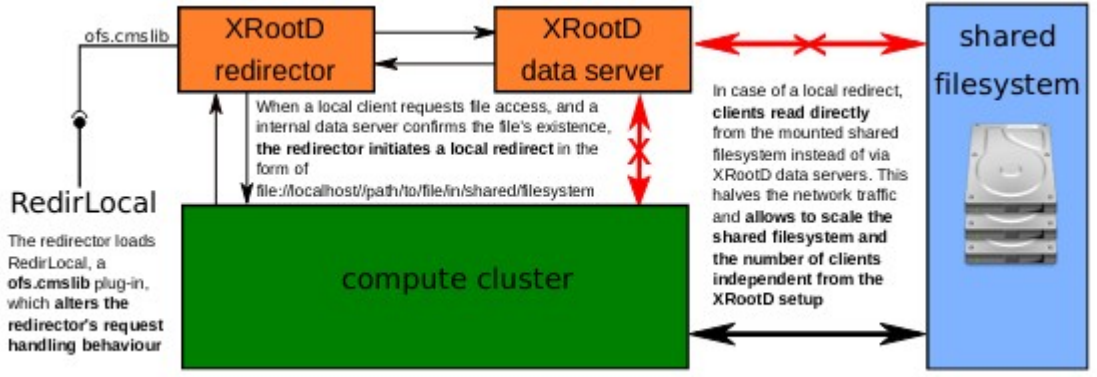
Infrastructure Evolution



Caches require less operational cost while minimising external I/O, and providing high data locality

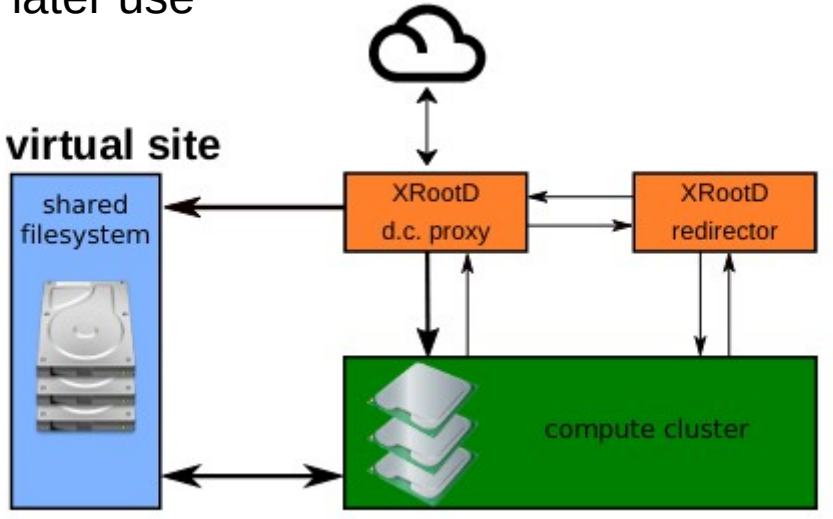
Dynamic disk caches have been developed in ErUM-Data-Pilot

- XrootD-plug-in based ingredients
  - ProxyPrefix (local clients are redirected to the local XrootD forward proxy)
  - RedirLocal (local clients read from local file systems if data is available)
  - In a classic XrootD based Grid SE clients would always read via XrootD redirector and data server



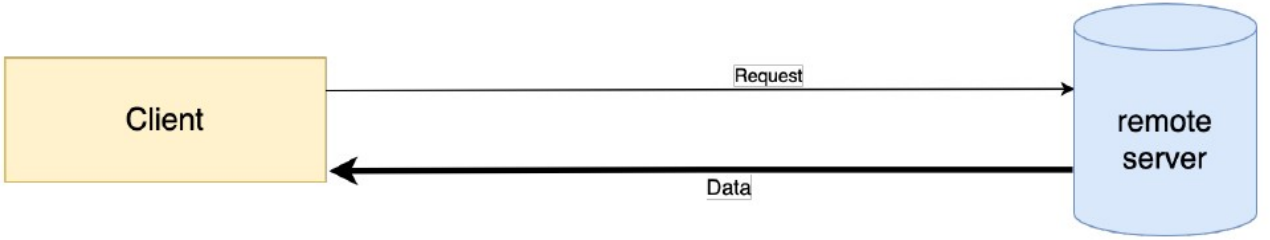
Disk Caching Proxy Setup (Disk Cache on the fly)

- Clients access data through the redirector
- If data exists redirector redirects clients to local file system
- Otherwise redirector redirects to disk caching forward proxy
- Disk caching proxy forwards request to external site and retrieves the data
- Data are being cached on local file system for later use

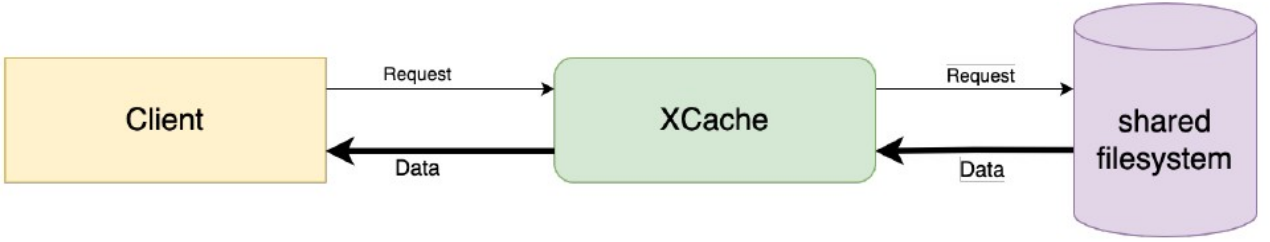


Cache Scenarios for Benchmarks

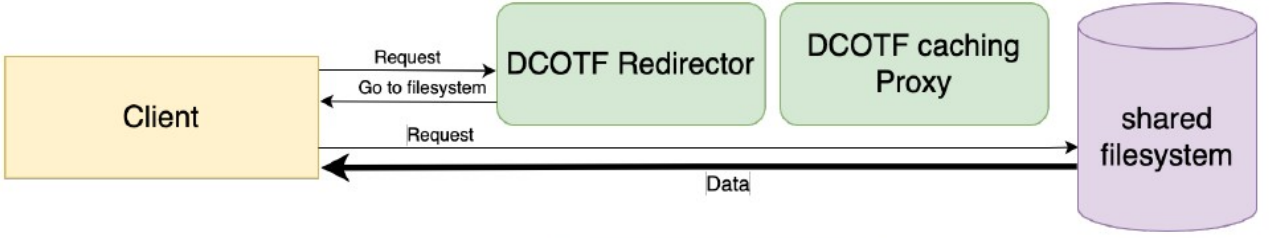
- No cache



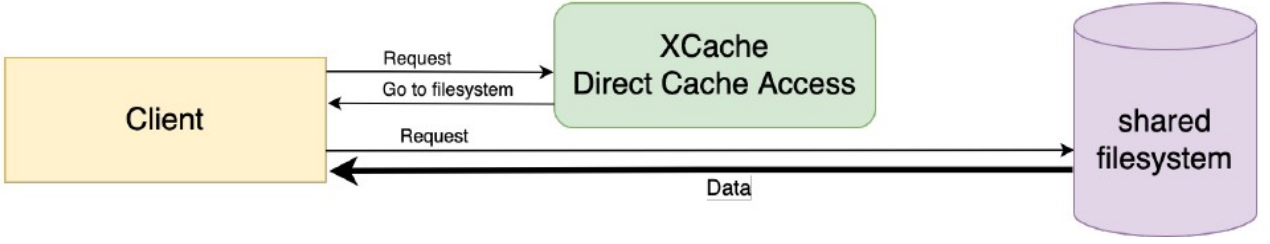
- XCache, data in cache



- DCOTF, data in cache



- XCache DCA, data in cache

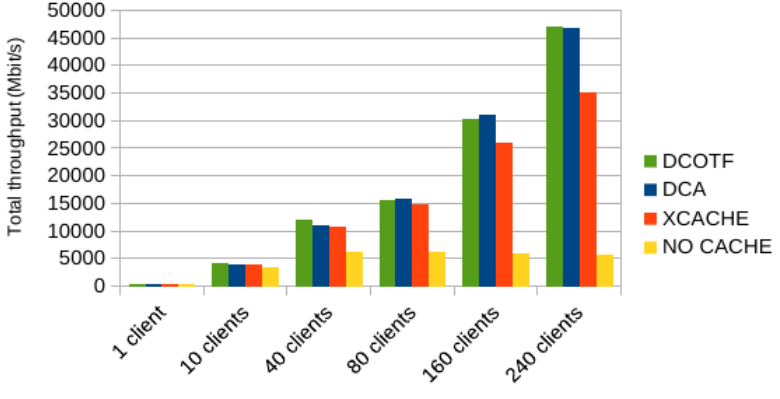


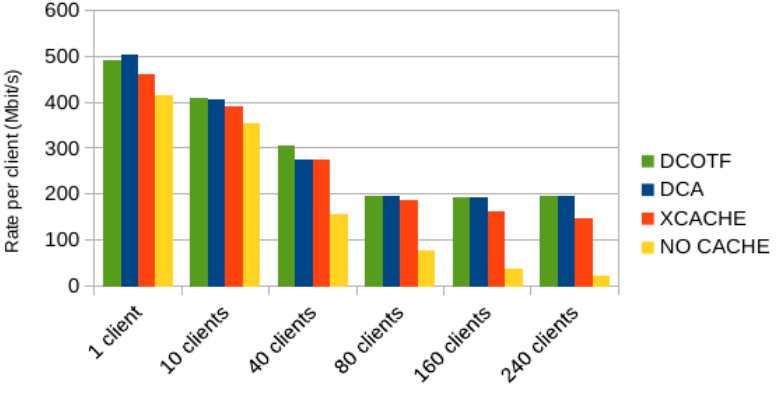
Cache performance test setup

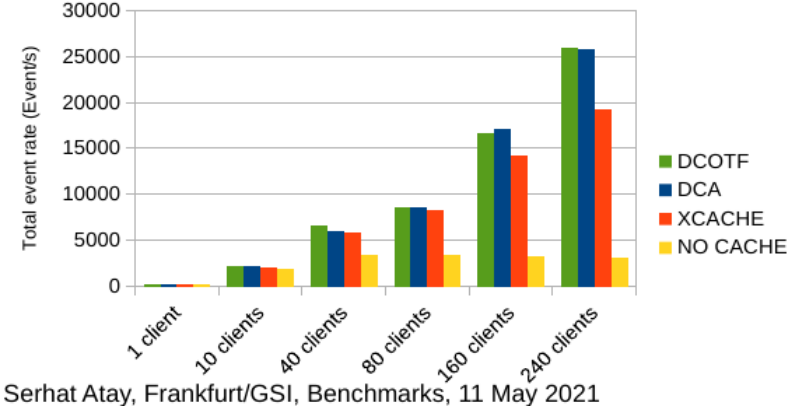
- Jobs submitted via Slurm to LOEWE CSC (Frankfurt) running in Singularity container
- Sample analysis reads events from branch and does nothing further
- Dynamic Disk Cache setup at LOEWE CSC
- Data located at GSI Lustre Cluster served via XrootD data server
- Bandwidth connection GSI – LOEWE CSC 10 Gb/s

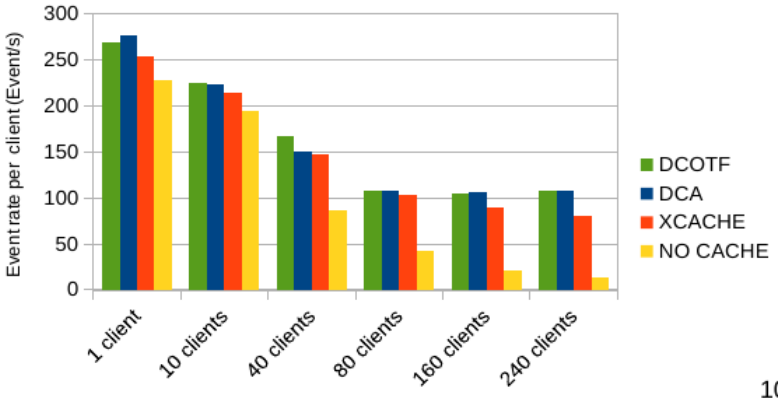
Cache performance results

ALICE ESD Data in Singularity



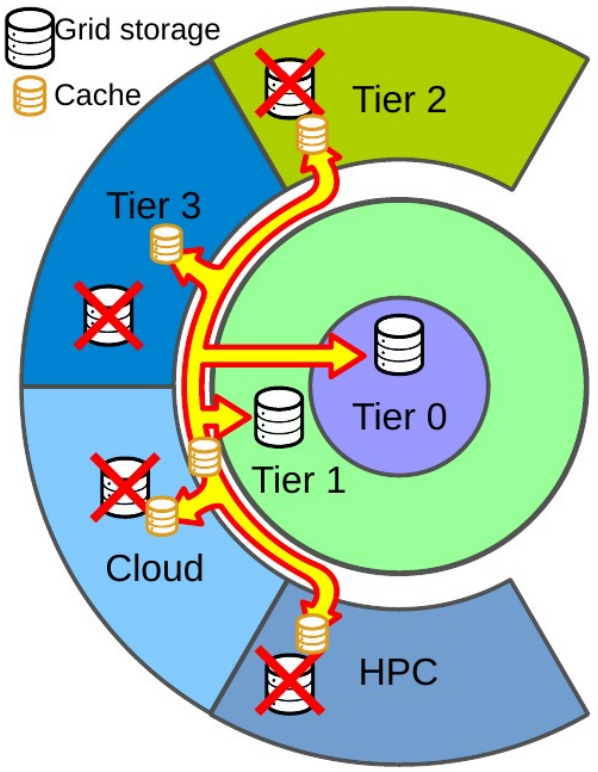






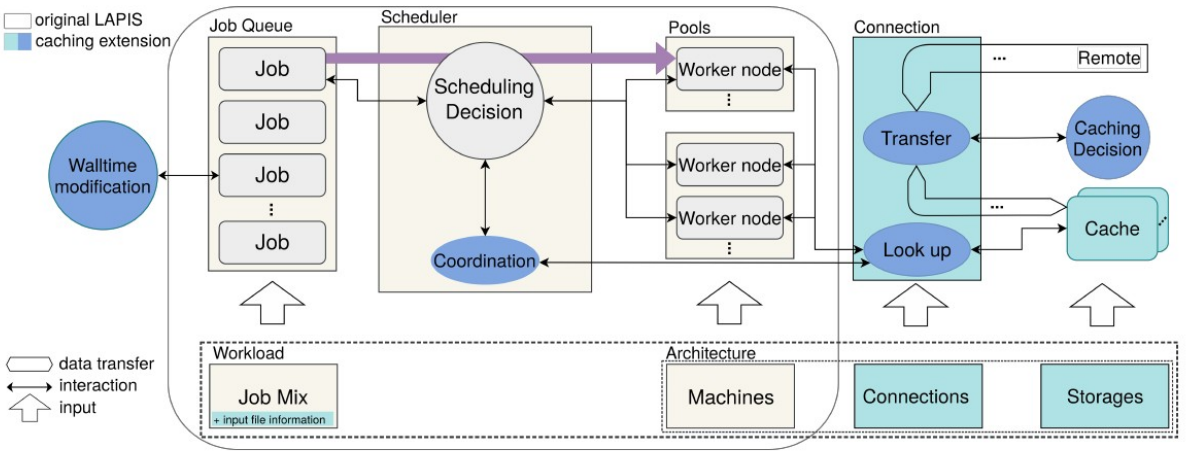
Modelling of heterogeneous distributed systems with coordinated caches

- **Fixing workload and scheduler: How do we achieve the most efficient data processing?**
  - How do we design the network?
  - How many caches do we need to place? Where?
  - What is the optimal size and caching logic of these caches?
  - Can we replace managed storage with caches without losing momentum?
  - Which figure of merit do we want to optimize? User walltime? Monetary costs? WAN bandwidth?
  - Do we cache for all workflow types? If not, for which?



Questions can be answered by simulation

Simulator LAPIS



T. Feßenbecker, *Modeling of distributed coordinated caching for LHC data analyses*

- Central metric for job characterization: job walltime

$$t_{\text{wall}} = \max\left(t_{\text{calculation}}, t_{\text{transfer}}\right) = \max\left(\frac{t_{\text{CPU}}}{\varepsilon_{\text{instr}}}, \frac{V \cdot (1-h)}{b_{\text{remote}}}, \frac{V \cdot h}{b_{\text{cache}}}\right)$$

Summary and outlook

- Dynamic disk caches have the potential to increase throughput significantly
- Dynamic disk caches are an efficient tool for integrating opportunistic resources
- DCOTF and XCache DCA have a similar performance
- XCache creates a bottleneck in terms of CPU and bandwidth (number of threads on caching node)
- No cache is limited by intersite network bandwidth
- Containerisation does not affect performance
- Strategic placement of dynamic disk caches can be optimised via KIT cache simulation