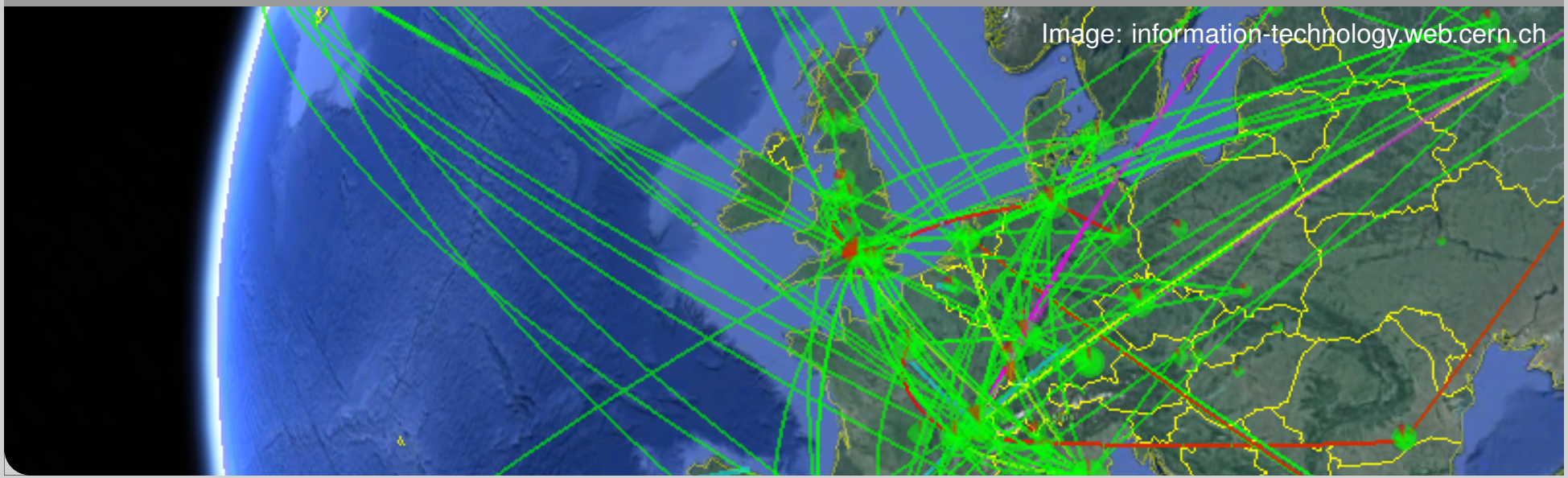
# Provisioning of Opportunistic Resources at KIT

R. F. von Cube, R. Caspart, T. Feßenbecker, M. Fischer, M. Giffels, C. Heidecker, E. Kühn, G. Quast, M. J. Schnepf, and the GridKa Support Team

IDT-UM Collaboration Meeting, Karlsruhe – September 30, 2019

INSTITUTE OF EXPERIMENTAL PARTICLE PHYSICS (ETP), STEINBUCH CENTRE FOR COMPUTING (SCC)



# Computing Resources at ETP

- Local storage and compute resources
- Access to multiple, shared resources } Resource Scheduling
  - HPC clusters, cloud providers
- Limited network to various sites } Coordinated Caching*
- Interdisciplinary collaboration of ETP, SCC, and the WLCG Tier 1 GridKa Team

→ Comparable to situation in HEP in the future

*See contribution by René Caspart later in this session

# Opportunistic Resources

# Opportunistic Resources

**Opportunistic resource in the HEP context**

Resource, not specifically designed, but temporarily made available for HEP workflows.

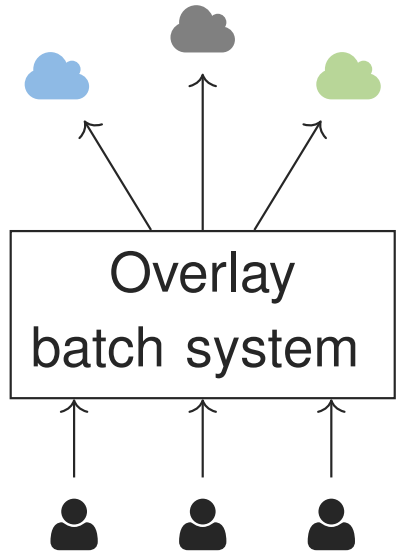
This may include

- HPC centers
- Resources at cloud providers

# Accessing Heterogeneous Resources

## Integration

- Resources are integrated into an overlay batch system
  - Transparent to the user
  - Single point of entry
  - Allows for hassle-free maintenance




## Provisioning

- Prepare resources to be able to run HEP jobs
  - OS (Centos 7, SLC6)
  - Protocols for file transfers (GridFTP, XRootD)
  - CVMFS

# Container technologies

- Containers allow for prepackaging images of exactly defined environments

**Docker**

- Widely used in industry
- Root-owned daemon for privilege escalation

**Singularity**

- Developed for HPC centers
- UIDs for privilege escalation
- User-id in container equals outside user-id

- Isolation of jobs via namespaces
- Host file system can be made available inside container
- For job scheduling: Each job can be placed in its own container

# TARDIS and COBALD

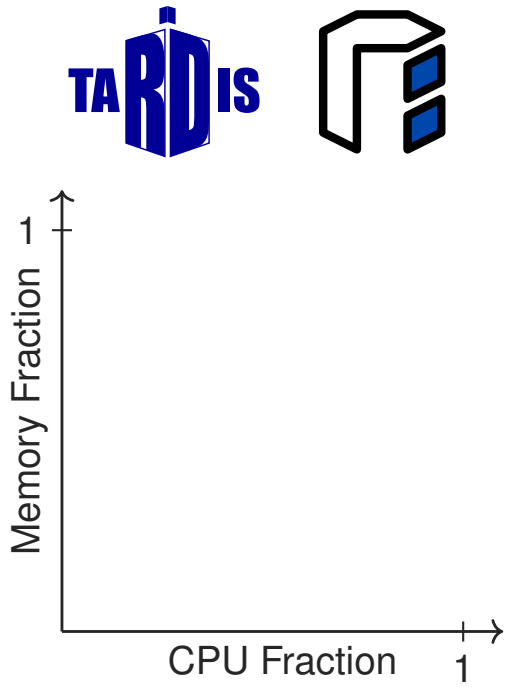
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

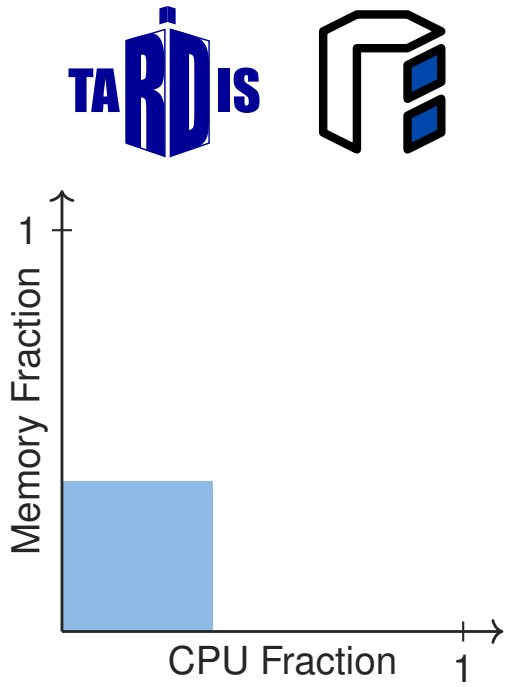
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

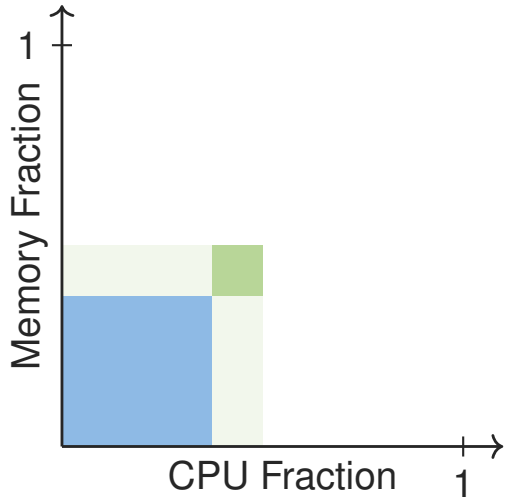
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

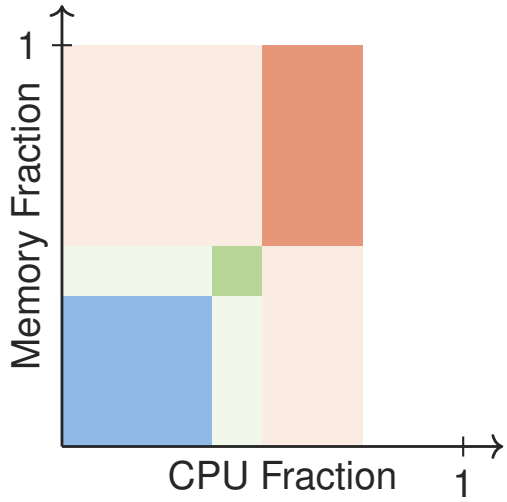
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

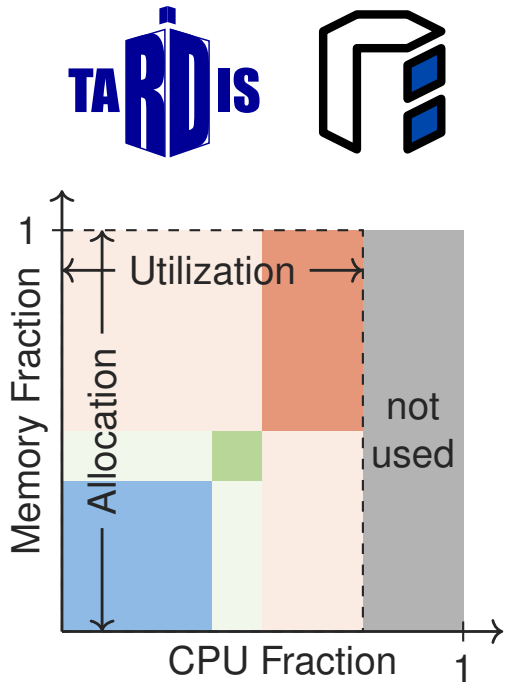
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

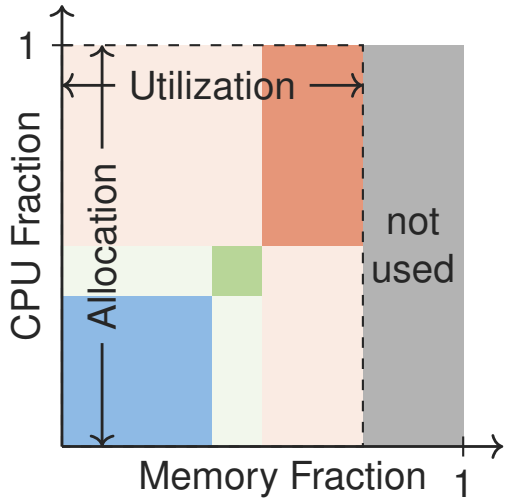
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



# TARDIS and COBALD

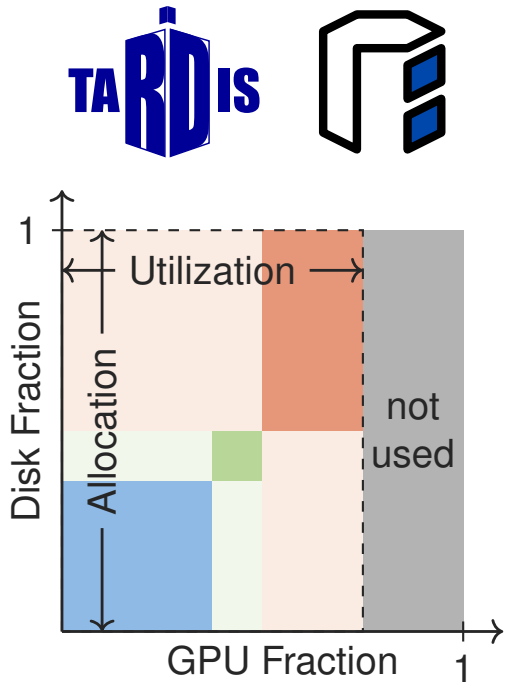
## Dynamic, Opportunistic Resource Scheduling

### TARDIS

Dynamically provisions and integrates resources into overlay batch system.

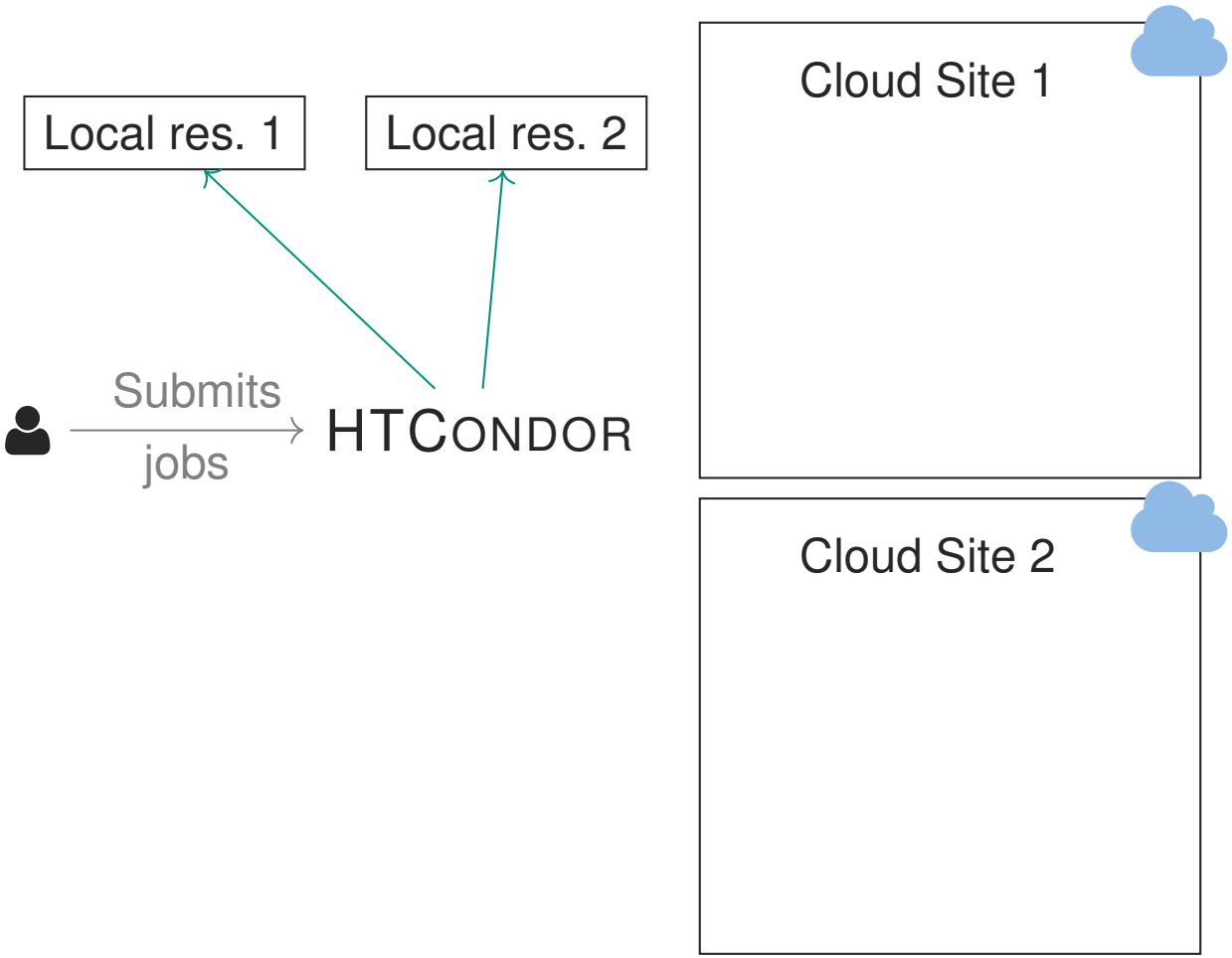
### COBALD

Assesses the fitness of resources to the current job mix with metrics *allocation* and *utilization*.



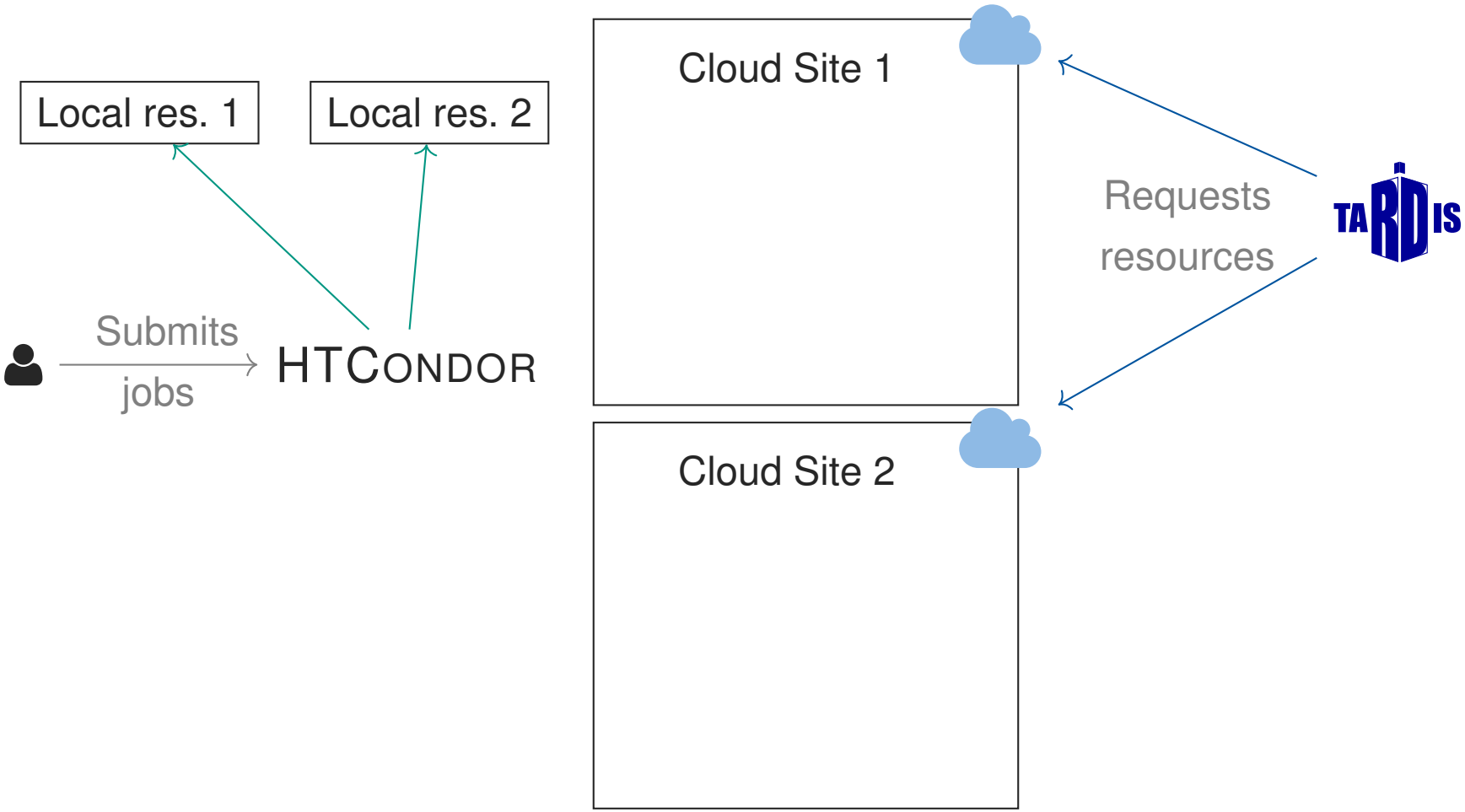
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



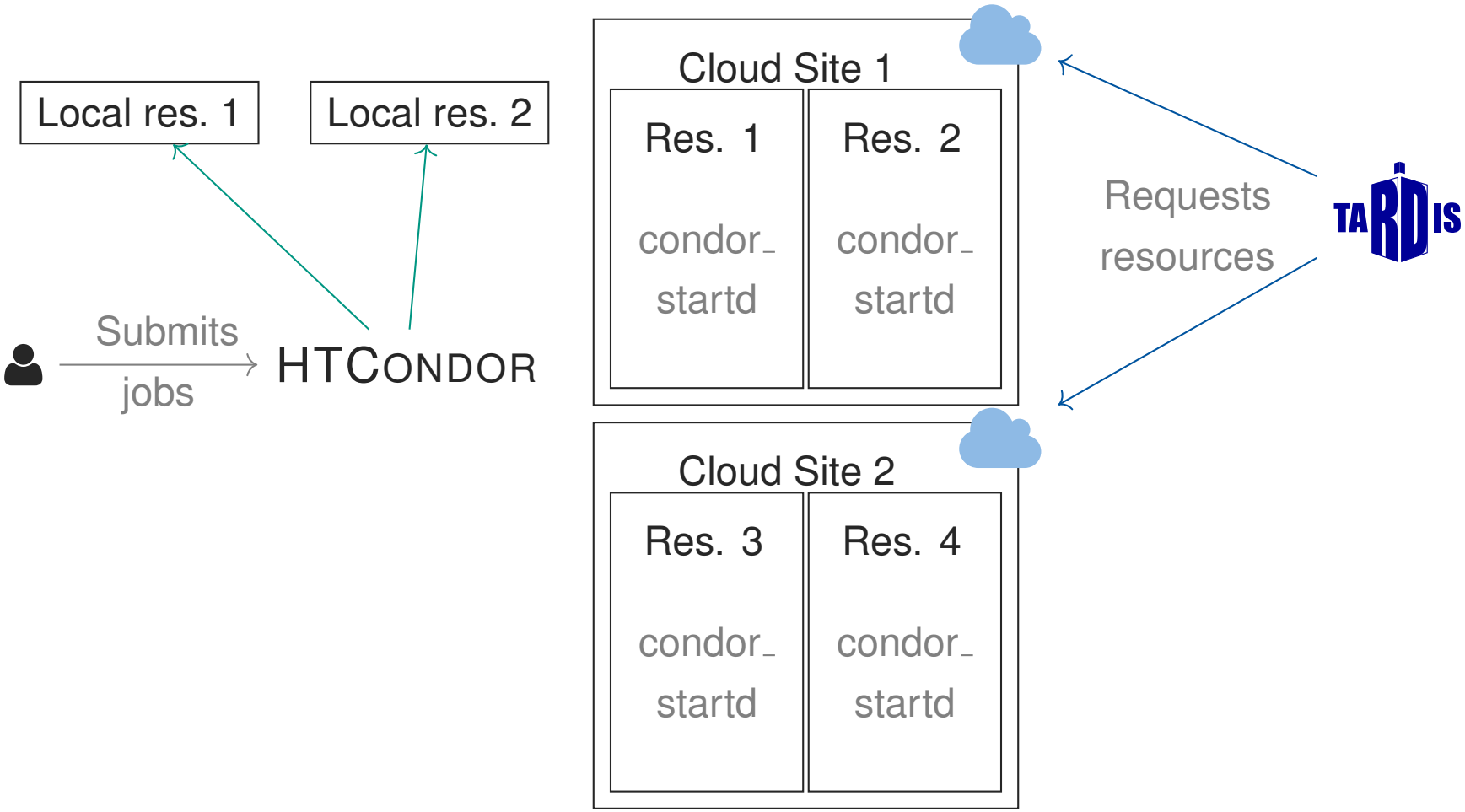
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



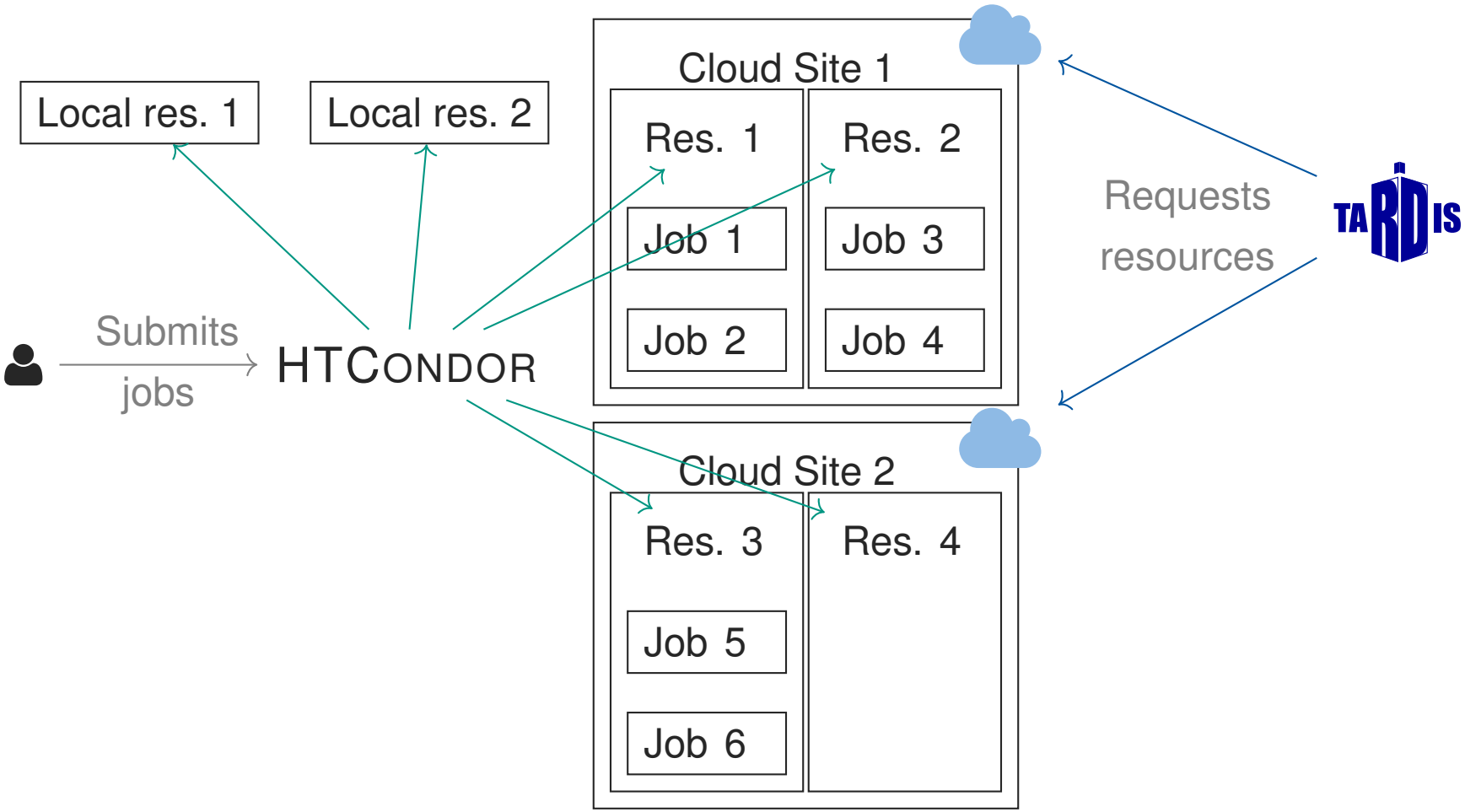
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



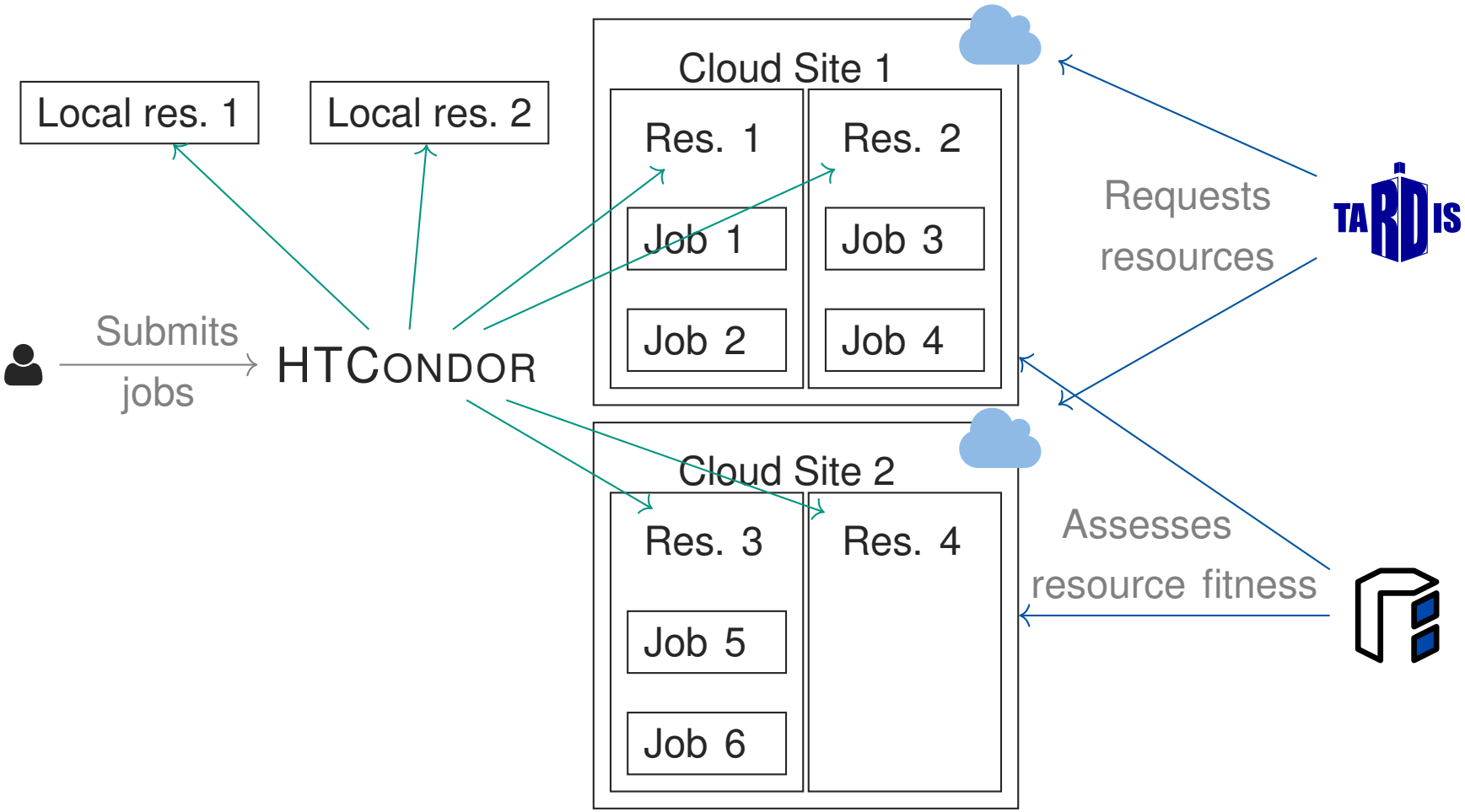
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



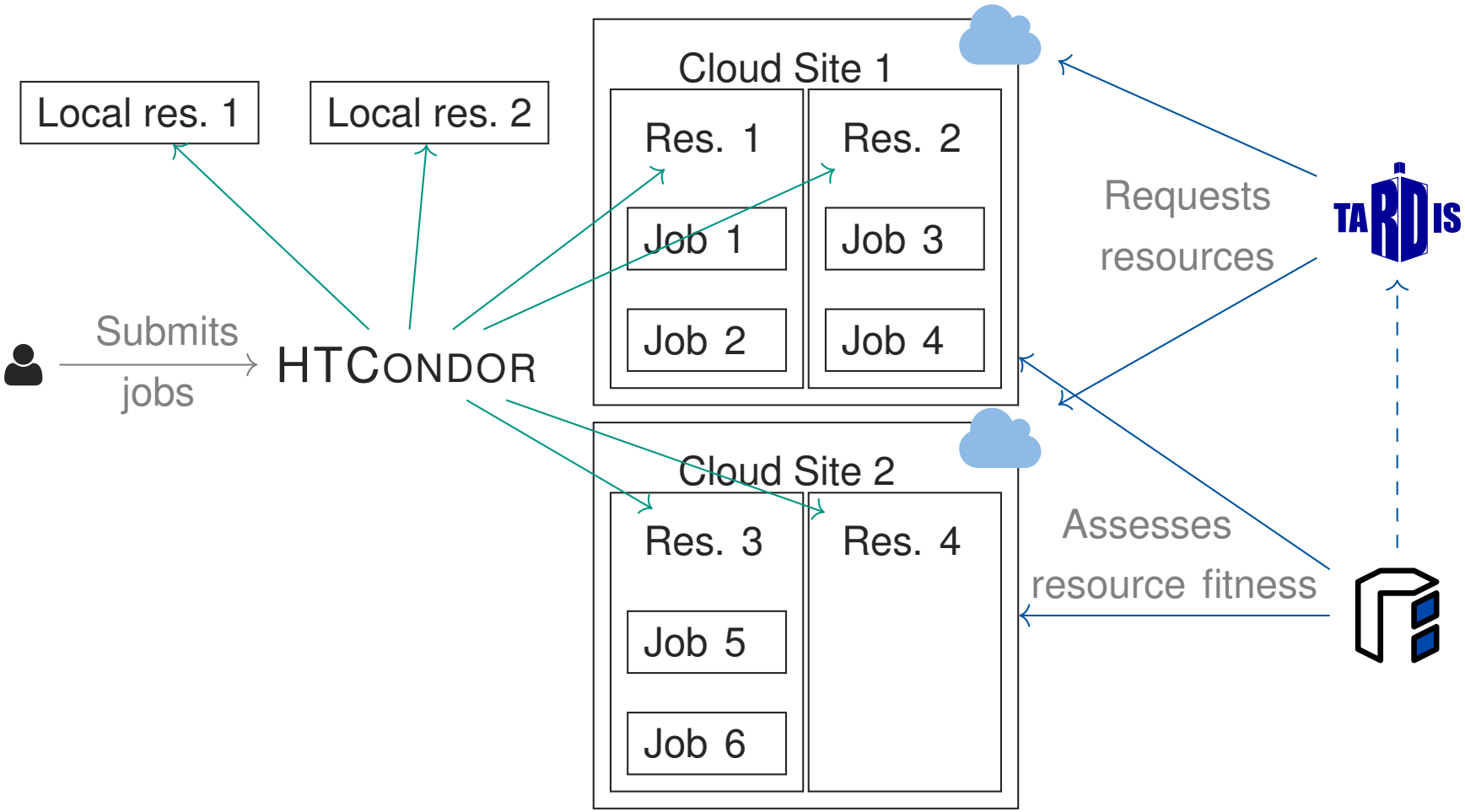
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



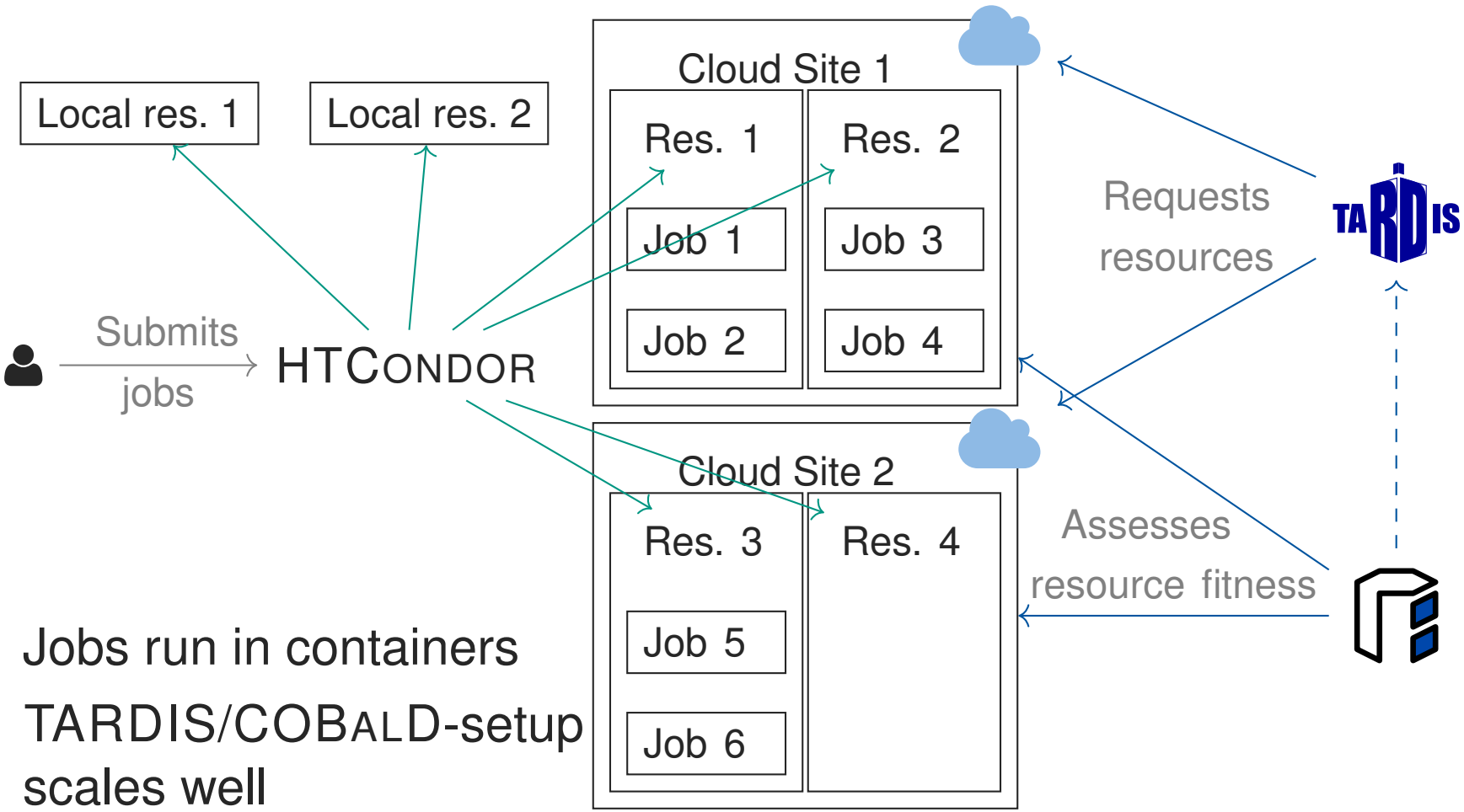
# Opportunistic Resources at ETP

- Resources are dynamically allocated on available cloud providers



# Opportunistic Resources at ETP

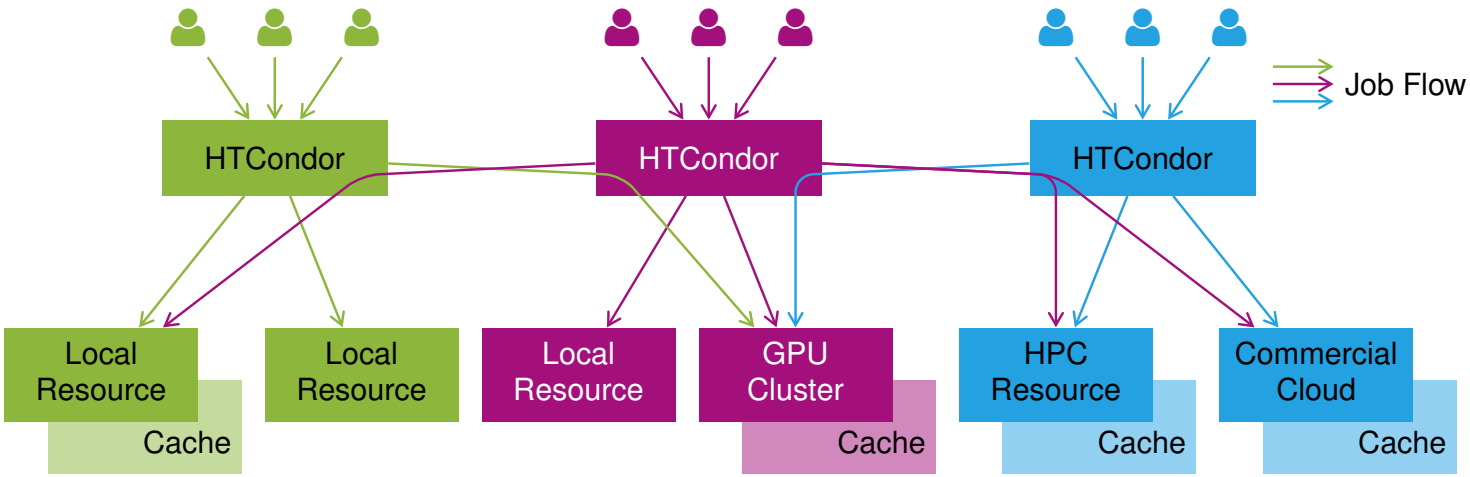
- Resources are dynamically allocated on available cloud providers




- Jobs run in containers
- TARDIS/COBALD-setup scales well

# Flocking

- Flocking is an internal HTCondor-mechanism for connecting HTCondor-resource pools
- Jobs are eligible for flocking if no match is found in local pool
- Less operational overhead than pilot-concept
- Would allow for connecting German resources (comparable to OSG)



# A Timeline of Opportunistic Resources at ETP

- Years of experience with opportunistic resources at ETP/SCC
- Experience taught us: Predicting decisions made by the batch system is hard
  $\Rightarrow$ react rather than predict
- Develop scalable, modular system for technologies to come

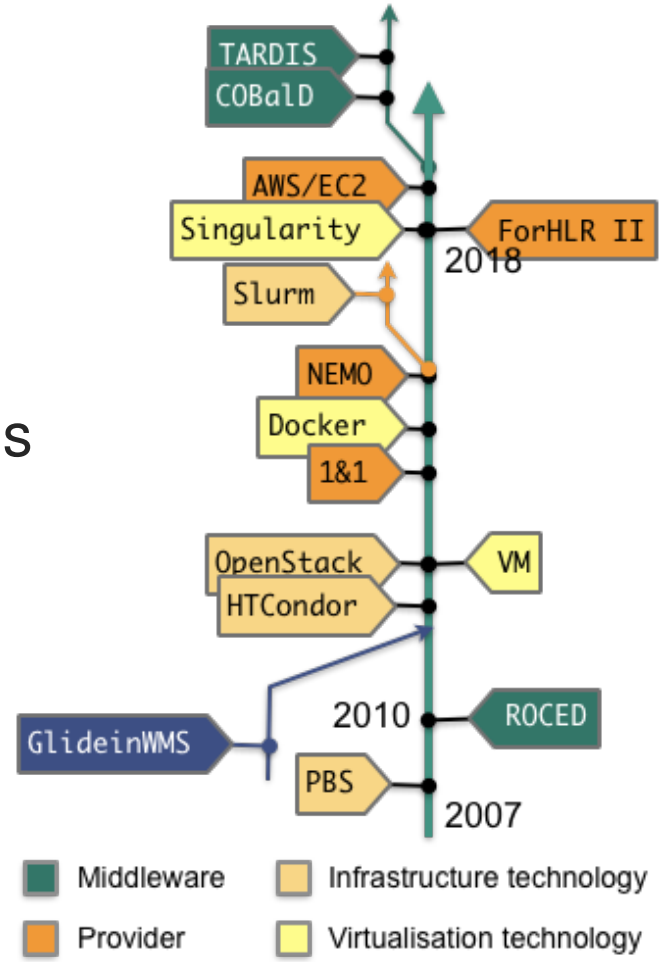



Figure by Eileen Kühn

# Current Setup

# TOPAS Cluster

**T**hroughput **Op**timized **A**nalysis **S**ystem

- Designed for 1PB/d throughput
- Located at GridKa, joined operation by ETP and GridKa
- Runs independent HTCONDOR instance

## Computing Operations

- ETP HTCONDOR flocks jobs to TOPAS
- TARDIS backfills for Tier1 extension
  - ALICE, ATLAS, Belle2, and LHCb jobs are run completely transparent to the experiments via extra entry point

# Current Situation at ETP

## FORHLR2

- BaWü Resource
- 25 000 cores, 97 TB memory (backfilling)
- Singularity on RHEL7

## TOPAS

- Located at GridKa
- $\sim$ 600 cores, 2.6 TB memory
- 1 PB cache
- Docker on RHEL7

## BWFORCLUSTER NEMO

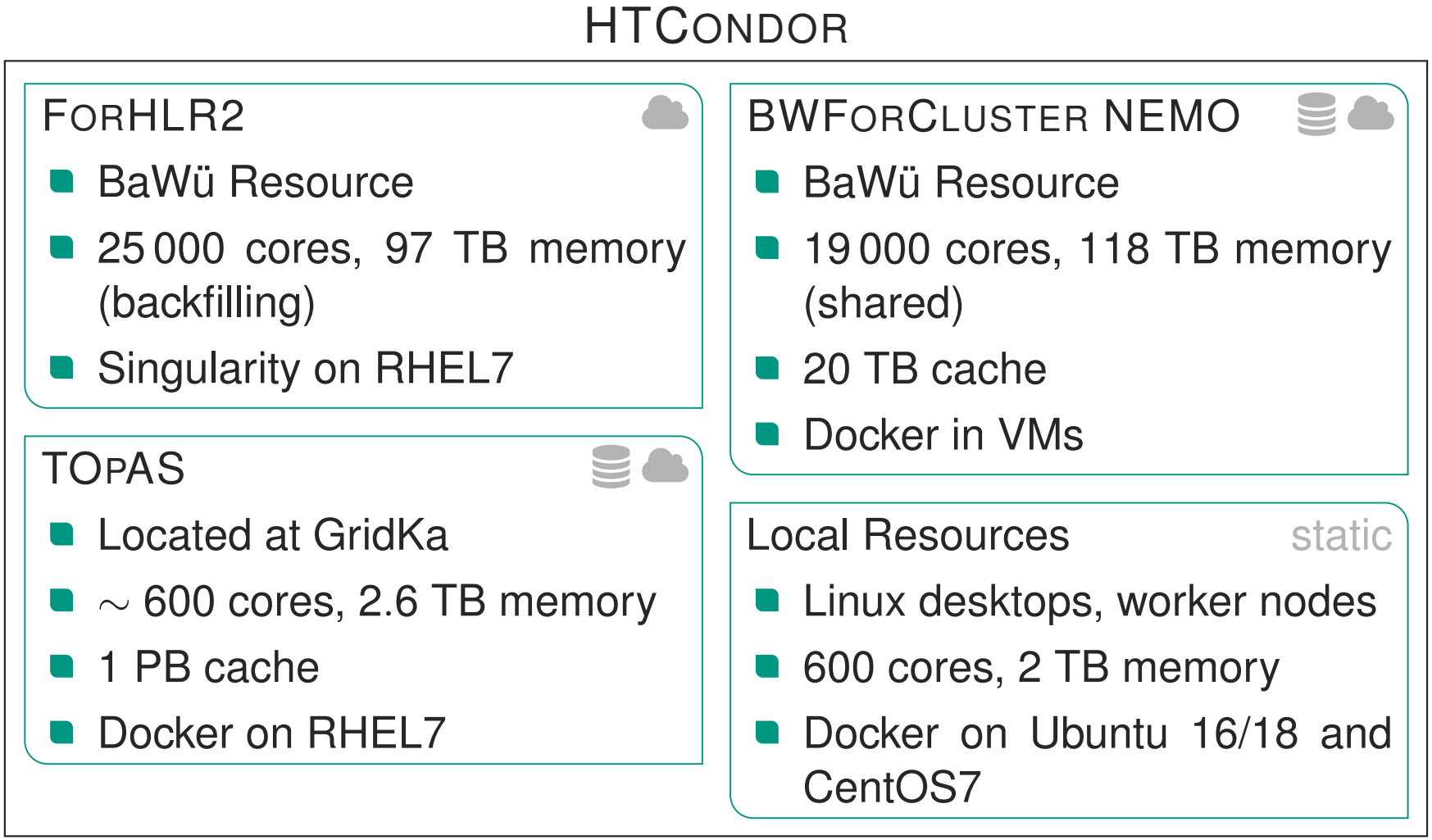
- BaWü Resource
- 19 000 cores, 118 TB memory (shared)
- 20 TB cache
- Docker in VMs

## Local Resources

static

- Linux desktops, worker nodes
- 600 cores, 2 TB memory
- Docker on Ubuntu 16/18 and CentOS7

# Current Situation at ETP



# Current Situation at ETP

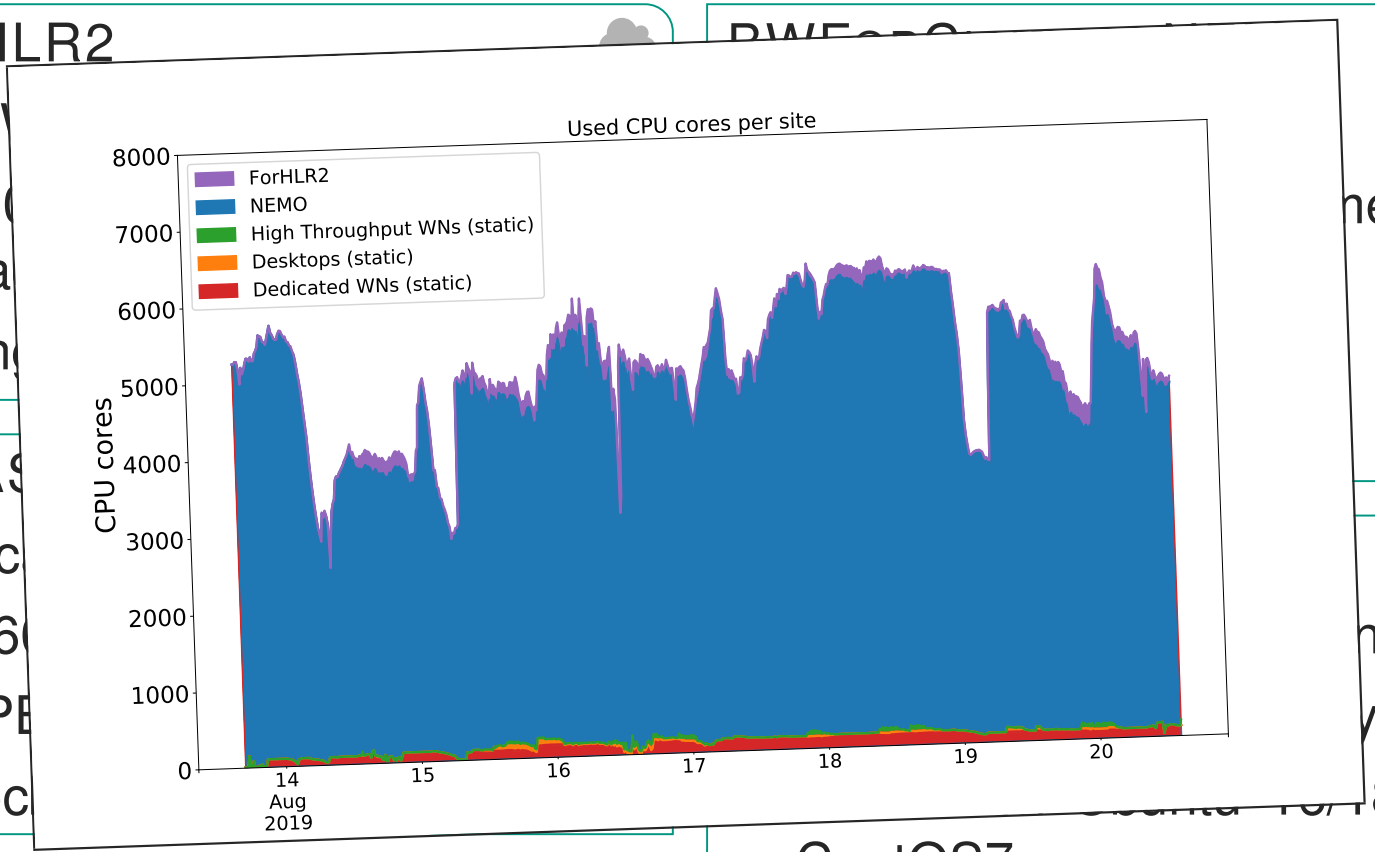
HTCondor

ForHLR2

- Ba
- 25 (ba
- Sin

TOpA

- Loc
- ~ 6
- 1 PB
- Doc




memory

static

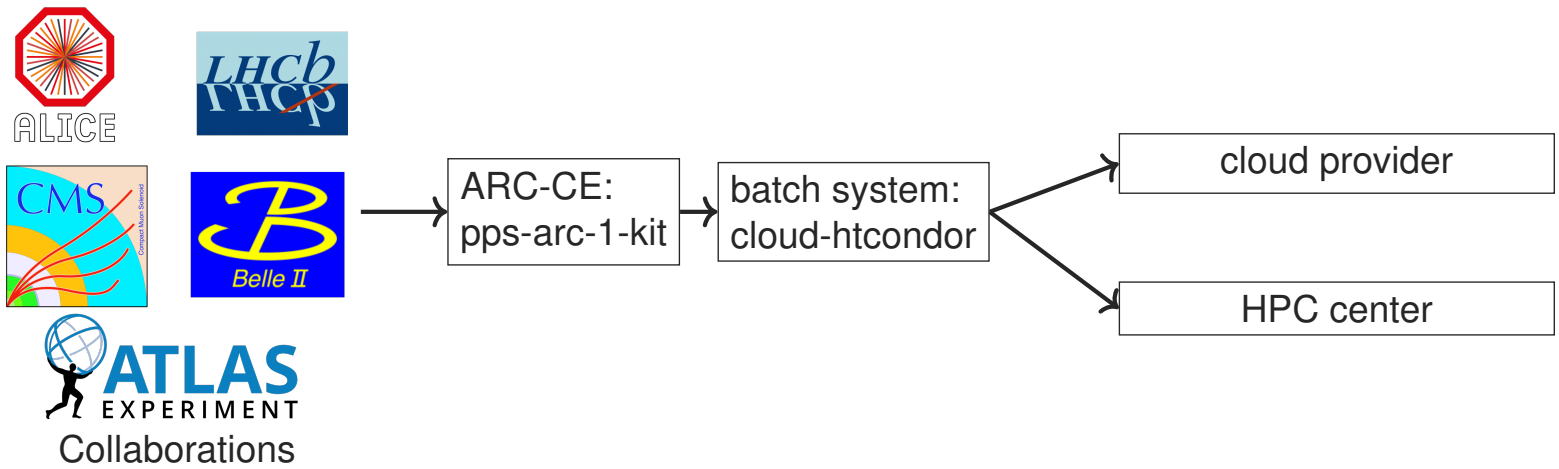
nodes

18 and

CentOS7

# TARDIS and COBALD at GridKa

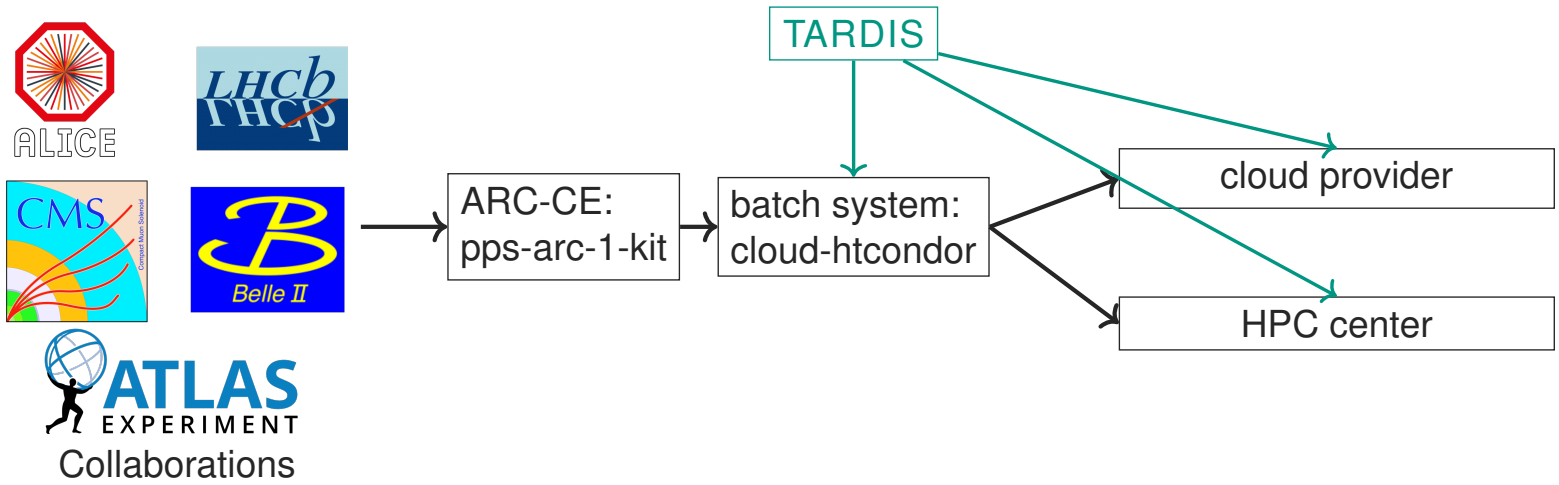
- entry point: `pps-arc-1-kit`
- HTCondor instance for opportunistic resources on `cloud-htcondor`




Slide adapted from Matthias Schnepf

# TARDIS and COBALD at GridKa

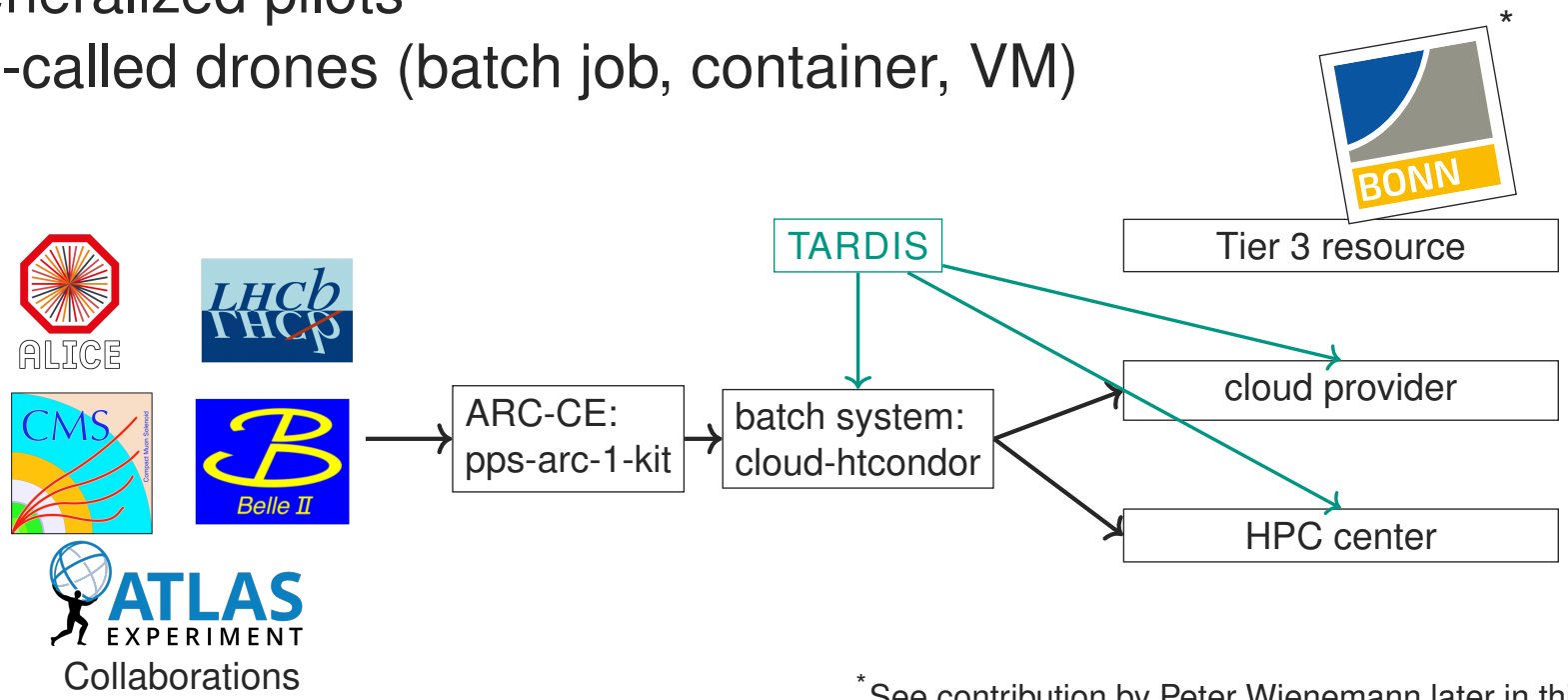
- entry point: `pps-arc-1-kit`
- HTCondor instance for opportunistic resources on `cloud-htcondor`
- resource manager TARDIS allocate and integrate resources via generalized pilots
  so-called drones (batch job, container, VM)




Slide adapted from Matthias Schnepf

# TARDIS and COBALD at GridKa

- entry point: `pps-arc-1-kit`
- HTCondor instance for opportunistic resources on `cloud-htcondor`
- resource manager TARDIS allocate and integrate resources via generalized pilots
  so-called drones (batch job, container, VM)

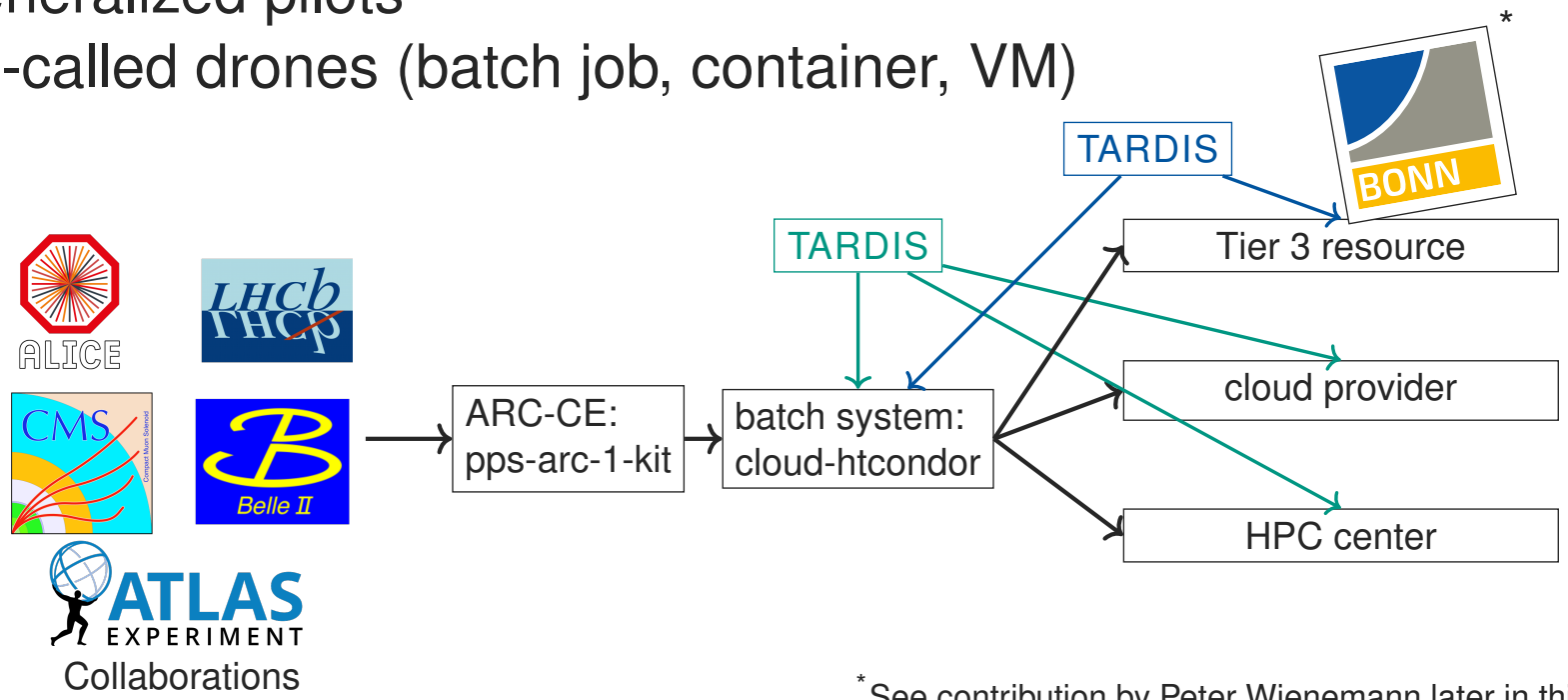



*See contribution by Peter Wienemann later in this session

Slide adapted from Matthias Schnepf

# TARDIS and COBALD at GridKa

- entry point: `pps-arc-1-kit`
- HTCondor instance for opportunistic resources on `cloud-htcondor`
- resource manager TARDIS allocate and integrate resources via generalized pilots
  so-called drones (batch job, container, VM)

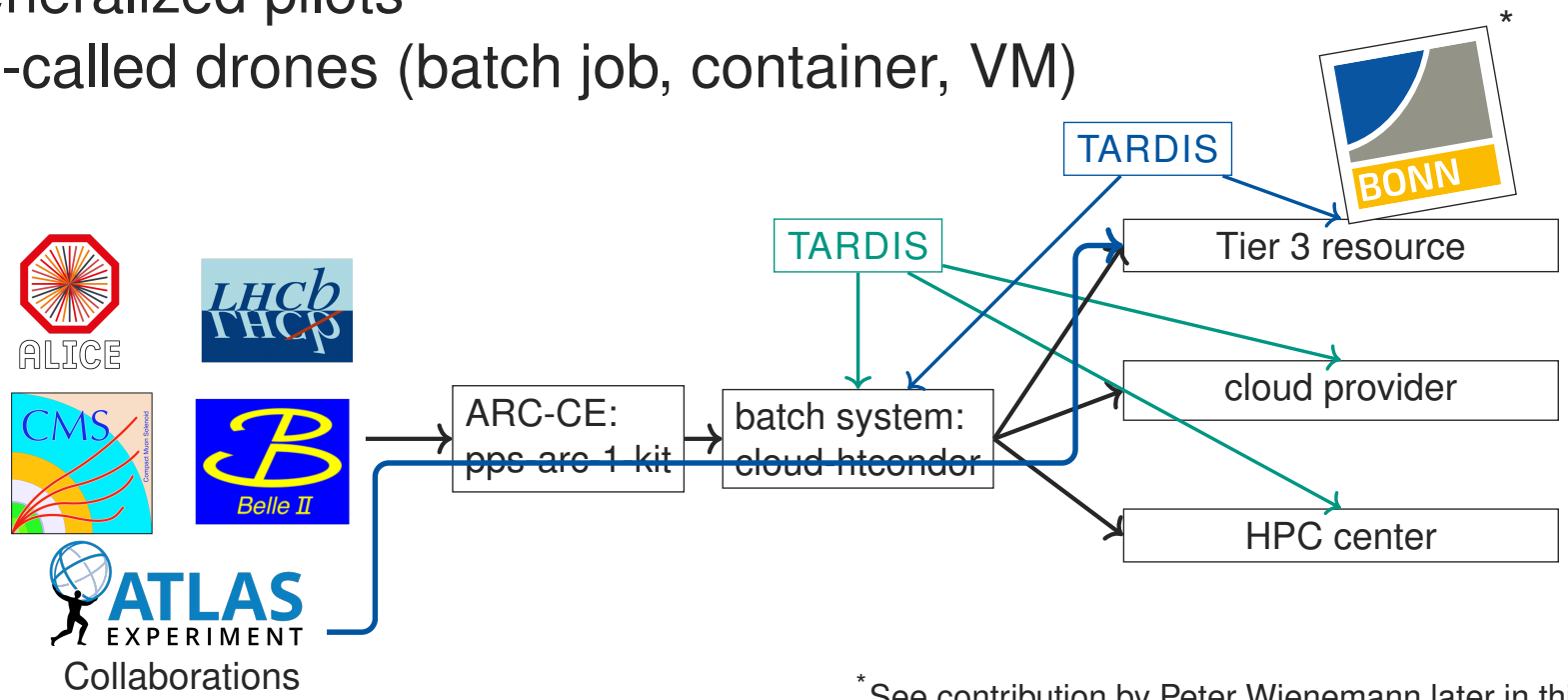



*See contribution by Peter Wienemann later in this session

Slide adapted from Matthias Schnepf

# TARDIS and COBALD at GridKa

- entry point: `pps-arc-1-kit`
- HTCondor instance for opportunistic resources on `cloud-htcondor`
- resource manager TARDIS allocate and integrate resources via generalized pilots
  so-called drones (batch job, container, VM)




*See contribution by Peter Wienemann later in this session

Slide adapted from Matthias Schnepf

# Placing ETP in the IDT-UM Context

## Area A

| | |
|---|---|
| **A1) Werkzeuge zur Einbindung**<br>• Scheduling von Cloud-Jobs<br>• Container-Technologien<br>• Checkpointing<br>• Zugang zu Experiment-Datenbanken | **A2) Effiziente Nutzung**<br>• Transiente Datencaches<br>• Transparenter Zugriff auf verteilte Daten |
| **A3) Workflow-Steuerung**<br>• Identifikation und Steuerung<br>• In-Pilot Job-Monitoring<br>• Accounting<br>• Optimierung durch Data-Mining | |

https://indico.desy.de/indico/event/21264/contribution/26/0/material/slides

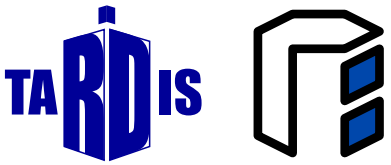
# Placing ETP in the IDT-UM Context

## Area B

| | |
|---|---|
| **B1) Tests der Technologiekomponenten**<br>Implementierung und Tests auf verschiedenen Plattformen von<br>• Speicher- und Cachinglösungen und<br>• virtualisierter Dienste (Datenbanken, Monitoring, Accounting). | **B2) Job- und Ressourcenmanagement**<br>Jobverteilung und Überwachung in der Umgebung heterogener Computingressourcen unter Einbeziehung von Containervirtualisierung.<br>  |
| **B3) Virtualisierung von Nutzerjobs**<br>• Erfassung der Anforderungen,<br>• Bestimmung und Erzeugung der Laufzeitumgebung,<br>• Erstellung des Containers und von Metadaten und<br>• Checkpointing von Containervirtualisierung. | **B4) Kombinierte Tests**<br>Testen von Gesamtsystemen (Speicher, Dienste, Ressourcemanagement) auf verschiedenen Plattformen in Bezug auf<br>• Installations- und Wartungsaufwand,<br>• Performance,<br>• Skalierbarkeit und<br>• Robustheit. |

https://indico.desy.de/indico/event/21264/contribution/26/0/material/slides

# Summary




- TARDIS and COBALD allow for dynamic integration of opportunistic resources
- Developed in close collaboration of ETP and SCC at KIT for production system

## Outlook

- Federation of national computing resources with single point of entry
- Single national submission infrastructure
- Nation-wide distribution of jobs on specific resources is done locally

# References

- TARDIS: `https://github.com/MatterMiners/tardis`

  DOI 10.5281/zenodo.3257718

- COBALD: `https://github.com/MatterMiners/cobald`

  DOI 10.5281/zenodo.1887873

# Appendix

# TARDIS and COBALD: Scalability

## Scalability

- decision based on managed resource information
- horizontal scaling of resources
  - up to 400 drones at ETP
  - 1152 drones for whole ForHLR II
  - ∼20000 drones for whole CMS WLCG computing
- one COBalD/TARDIS instance per provider (multi agent system)
- current operation mode is one COBalD/TARDIS instance per resource provider