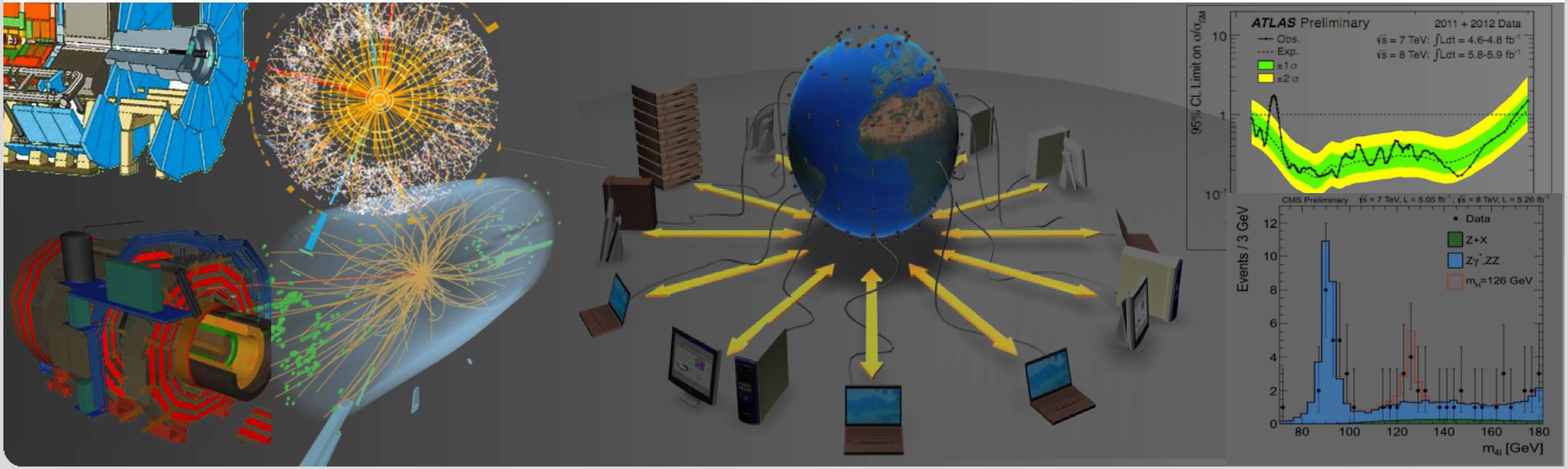
Karlsruher Institut für Technologie

Integration of Clouds in HEP Grid

Günter Quast

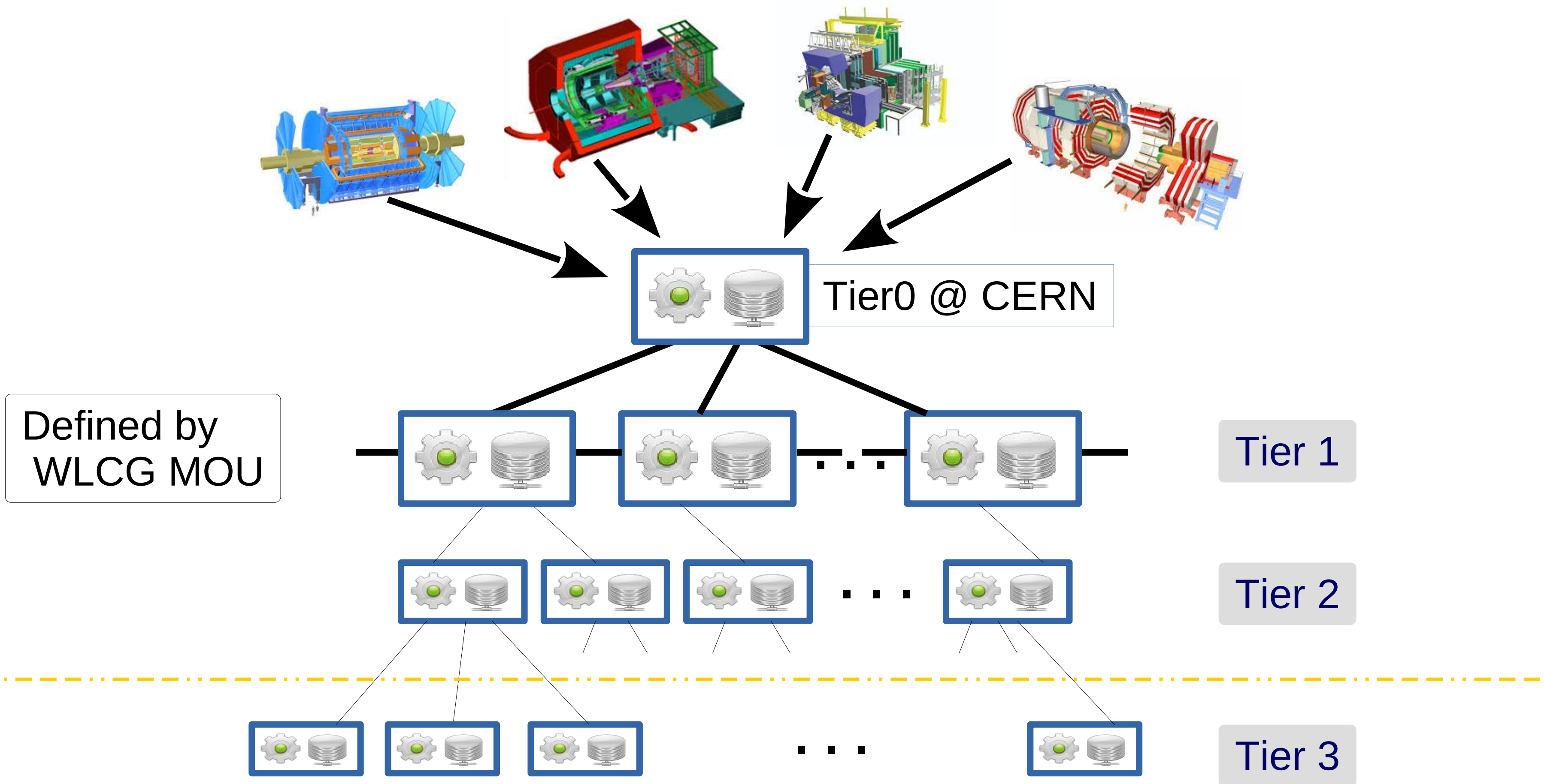
Fakultät für Physik
Institut für Experimentelle Teilchenphysik

KET Workshop Software & Computing



Grid to Cloud

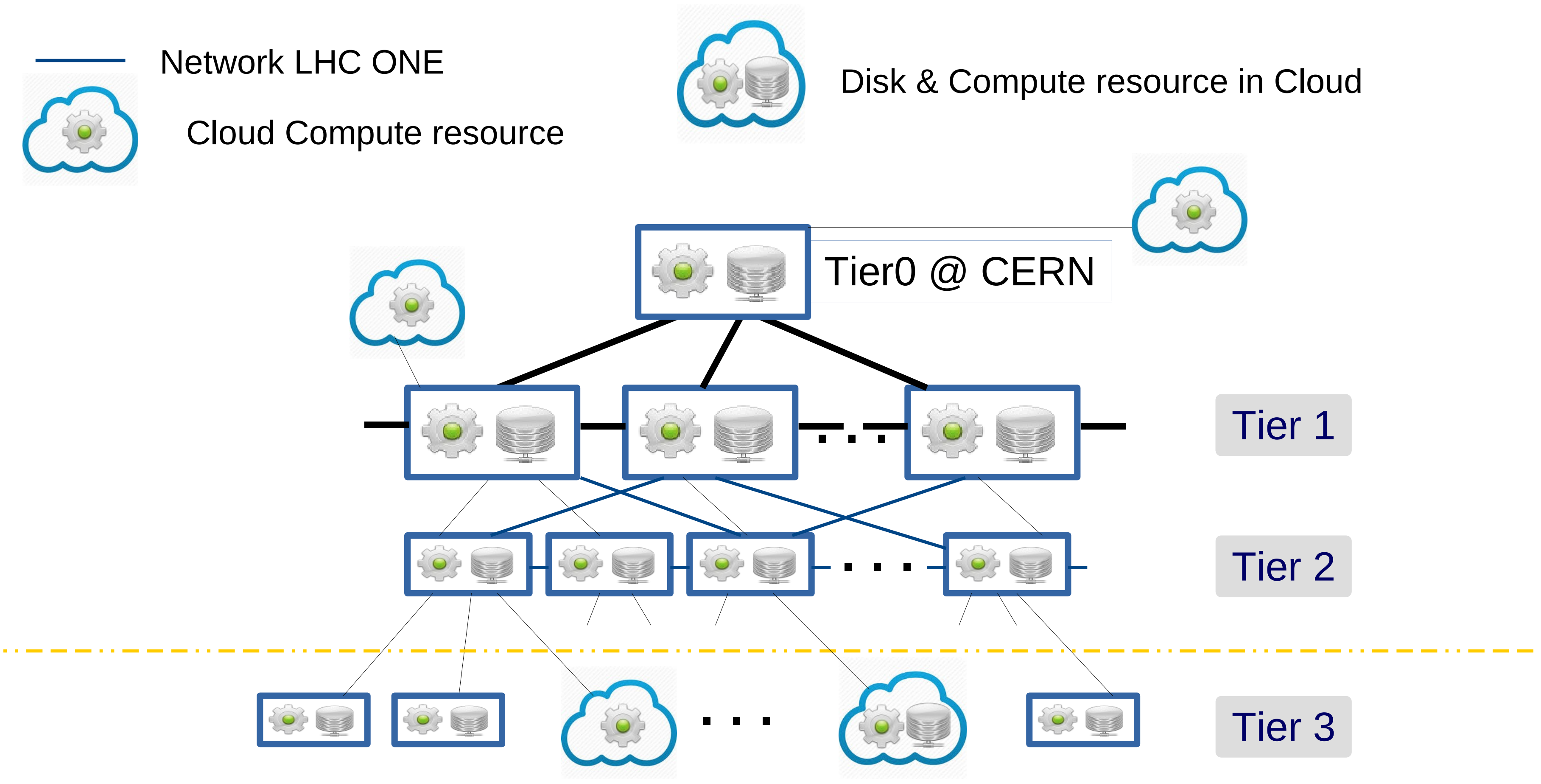
The original hierarchical World Wide LHC Computing Grid



All centers “owned” by particle physics groups
- common operating system (scientific linux)
- common services, in particular grid middleware
- common software stack

Grid to Cloud

Today with many more network connections **(Grid → Mesh)**
and first cloud additions



- LHC One Network connecting Tier2 / Tier2
- Data federations with remote access
- addition of “diskless Tier3”, and of Cloud Resources

Grid to Cloud

Future: (very ?) complex network of heterogeneous resources

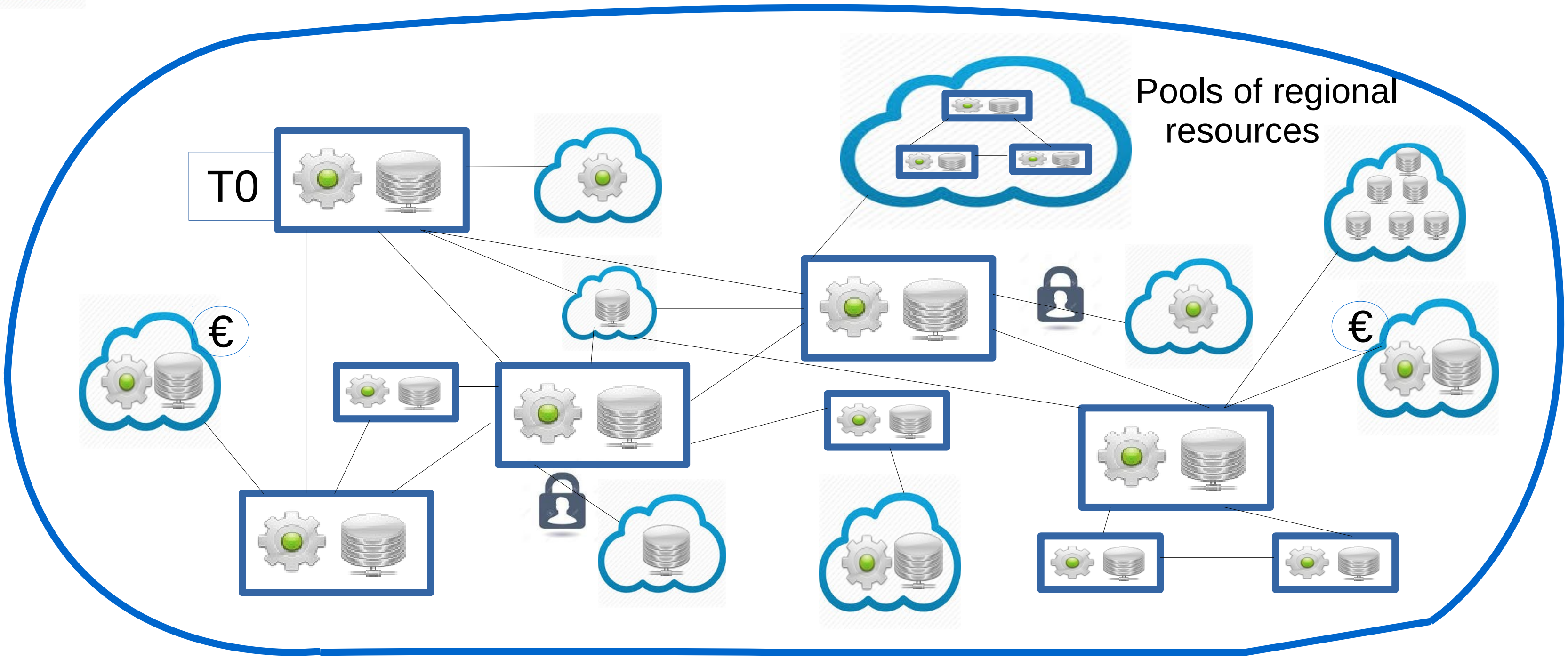

Classical WLCG center


Cloud Compute resource


Data Cache


Disk & Compute resource in Cloud


non-WLCG authentication



Challenges: (dynamic) workflow management and scheduling, optimisation of data placement, (dynamic) resource provisioning, authentication, billing, …

Cloud-enabling Technologies in HEP

CernVM:

- Virtual machine based on Scientific Linux (maintained by CERN)
- Very lightweight, can be directly deployed on various cloud sites

Container: Docker or Singularity

- encapsulates services, experiment software and user code

CernVM-FS:

- On-demand HTTP based file system (Caching via HTTP Proxy)
- Many big experiments use it to deploy software to WLCG compute centres
- works excellently also on cloud sites

HTCondor:

- Free and open-source batch system commonly used in HEP
- Excellent with integrating dynamic worker nodes (even behind NATed networks)

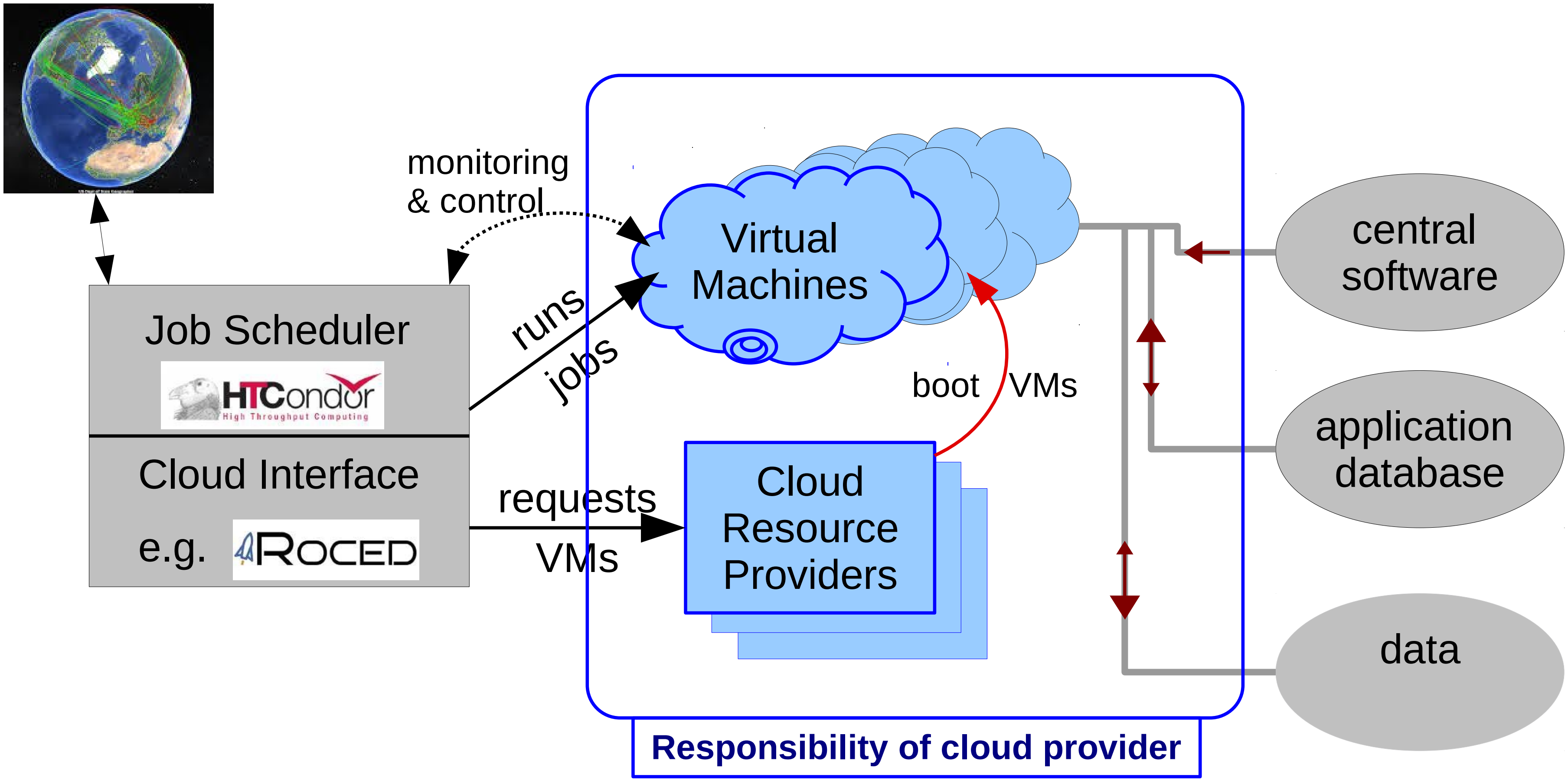
xRootD and **data federations** for remote access

Cloud Management Interface, e.g. ROCED [KIT]:

- Cloud scheduler that supports multiple cloud APIs (OpenStack, Amazon EC2 and other commercial providers, special HPC site adapters)
- Easily extendable thanks to modular design

  Parses HTCondor ClassAds and boots VMs on cloud sites depending on the number of queued jobs

Principle of including external resources

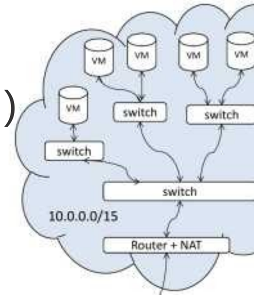


automated, dynamic provisioning of cloud resources
for suitable classes of jobs

Existing Examples

- Cloud-Resources
  - private (e.g. institute clusters)
  - commercial (Amazon, Google, Telekom …)

left R&D Phase since long



e.g Open Stack

ATLAS & CMS
HLT-Farms during LS 1

- HPC – Cluster
  - many HEP applications run
  - MPI interfaces and SAN not needed for HEP

→ **an expensive way!**

e.g. Super-MUC at LRZ in Munich, SDCC at CMS or the new bwHPC Cluster in Freiburg (ATLAS/CMS/LHCb)

– no Grid services / authentication
– fast, but small and expensive disk
– often small WAN bandwidth
– different „Site Policies“

special solutions for every single case → personnel !

- Commercial providers,
  e.g. HELIX NEBULA THESCIENCECLOUD

IaaS provided by two commercial consortia in final project phase

T-Systems · HUAWEI · ONEDATA · Open Telekom Cloud · divia

RHEA GROUP · sixsq · ONEDATA · EXOSCALE

- working with industry is different
- must define requirements precisely & monitor results

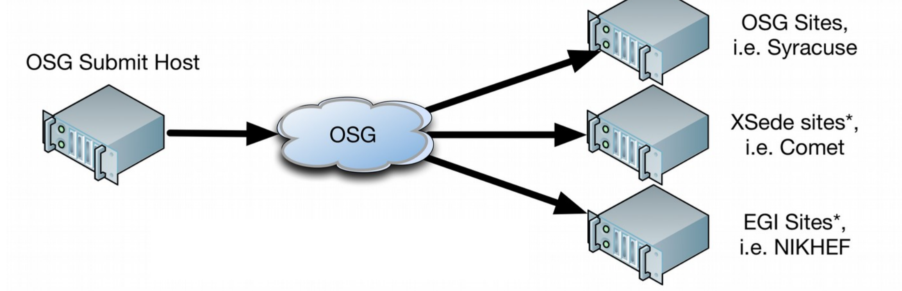
feeling of project participant:
"spent far more (unfunded) time than initially estimated"

Existing Examples (2)

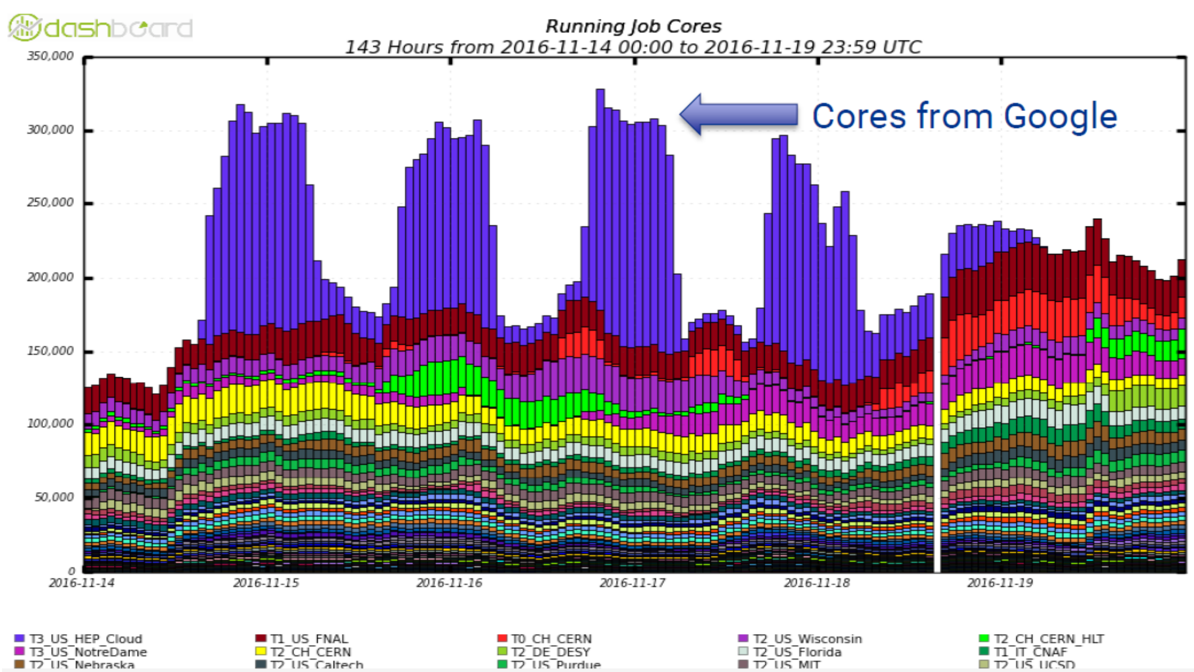
Open Science Grid (OSG) in the US

drive development of Science Clouds, centered around HTCondor, provide full software stack & documentation, organize schools and support

“Submit Locally, Run Globally”



Bursting into Google Cloud (2016)



Existing Examples (3)

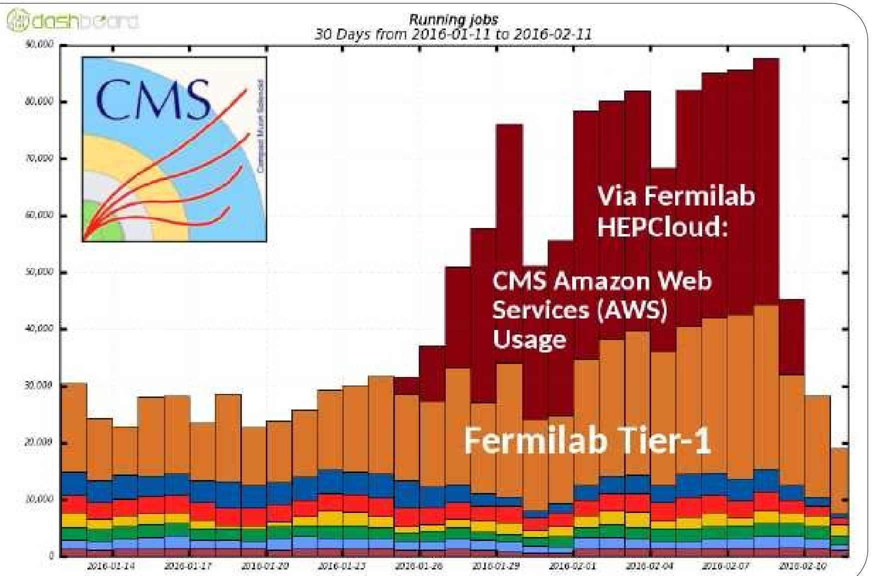
Amazon Cloud (Amazon Web Services)

provided to CMS via FNAL Tier1

– more than doubled available CPU at Tier Ones

– sponsored by provider, but cost on spot market approaching reasonable levels: (FNAL: 0.9 Cent/ CPU hour, Amazon: 1.4 Cent/CPU hour)

a similar project with Amazon also by ATLAS via BNL



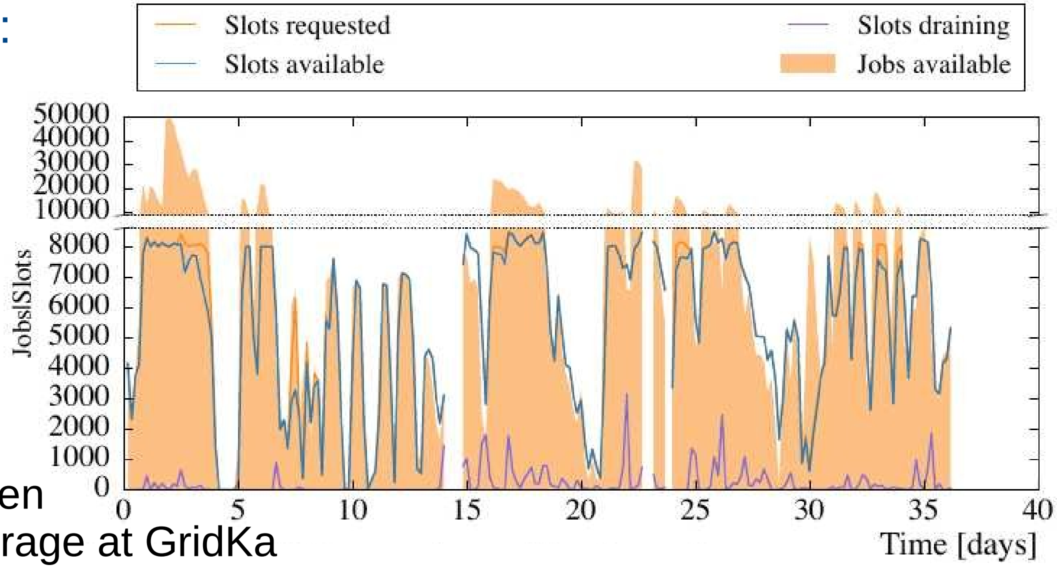
bwFOR cluster NEMO in Freiburg:

(for Neuro-Science, Elementary Particle Physics and Microsystems Engineering)

– fully virtualised set-up; controlled by ROCED (KIT) and HTCondor

– Production system scaled up to 11k virtualised cores, more than 7 million CPU hours of user jobs processed in four months

– saturating 20 GBit/s BelWü link between Karlsruhe - Freiburg and NRG Grid storage at GridKa



a word on Opportunistic Resources

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

- **very common** in university environments:
  - university clusters shared by many groups and communities
  - DFG-funded resources,

    typically allocated on approved request only for given time or # of CPU hours, only temporary storage as long as project runs
  - "empty" cycles on HPC systems
  - in the future:
    - cheap commercial CPU cycles

      to cover peak loads or to benefit from special offers on the "spot market"
    - dedicated "Science Clouds"

- Challenges:
  - often not at all corresponding to "typical" HEP setups

    need easy and light-weight access methods
  - must be dynamically managed and integrated into workflows

    need automatic cloud and batch interfaces
  - need careful monitoring of remote resource usage

    e.g. to avoid expensive network traffic from/to commercial cloud sites
  - "every such site is special"

    can only be integrated by "local" personnel
  - may require special access rights and authentication

    may need local "proxy" to submit jobs

Opportunistic Resources @ KIT

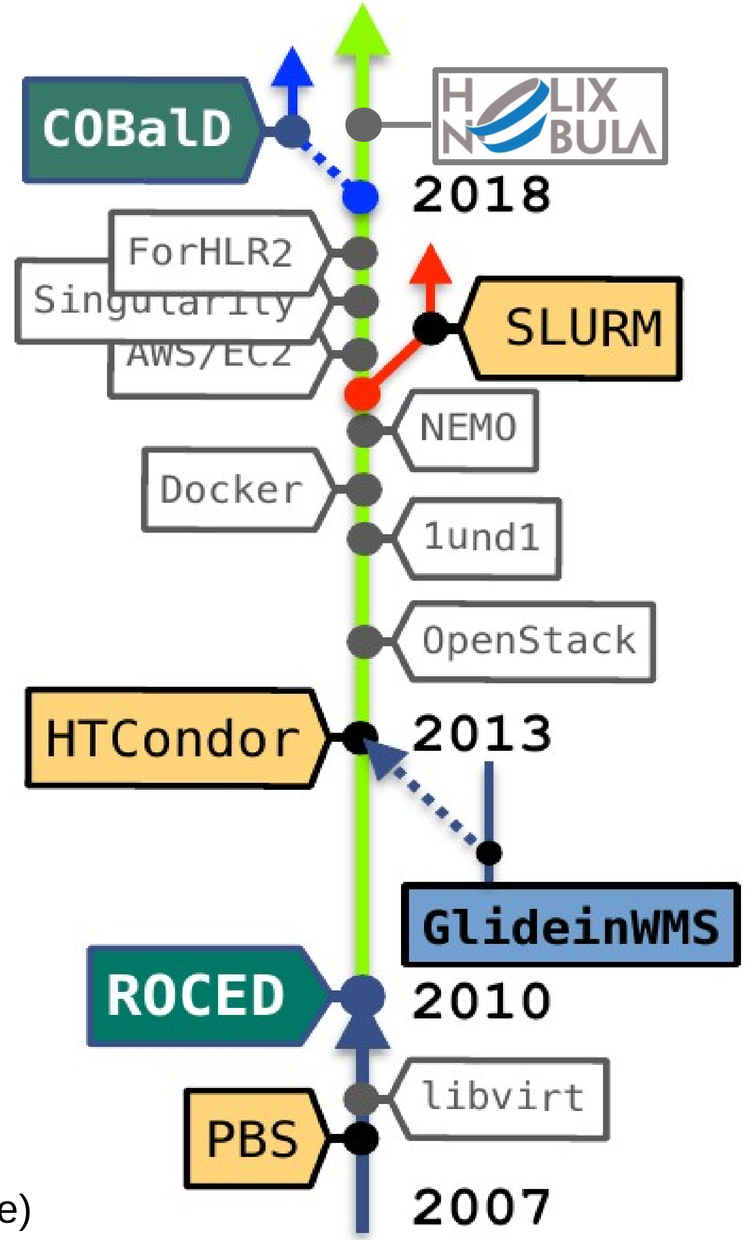
Longtime experience @ KIT with virtualisation, opportunistic resources and containers:

- *Software development*
  ViBatch, Roced (now → CBOalD), NaviX
- *Virtualisation* with Xen, kvm, OpenStack
- *Containers* Docker and Singularity
- *Opportunistic Resources*
  - Test systems (incl. Desktops) with OpenStack and Docker
  - HPC Clusters
    Ic1 @Uni Ka (ViBatch), ForHLR @KIT (Singularity), bwFORcluster NEMO @Uni Fr (OpenStack)
  - commercial Cloud providers
    AWS, 1&1 Cloud Services, OTC, ExoScale

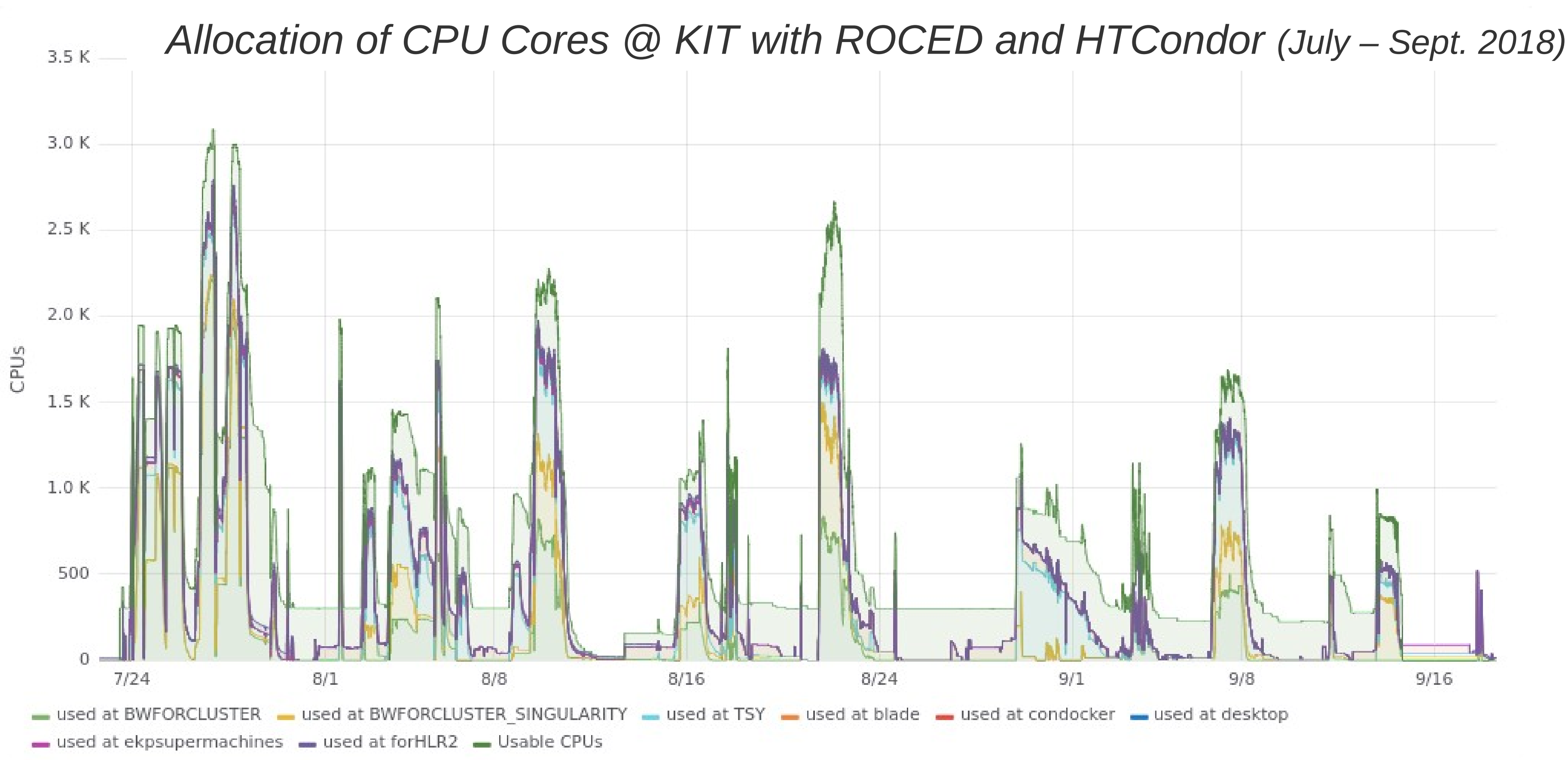
All controlled via a single HTCondor instance and cloud interface ROCED

many of them simultaneously and in production mode

(see next slide)



Usage of Opportunistic Resources @ KIT



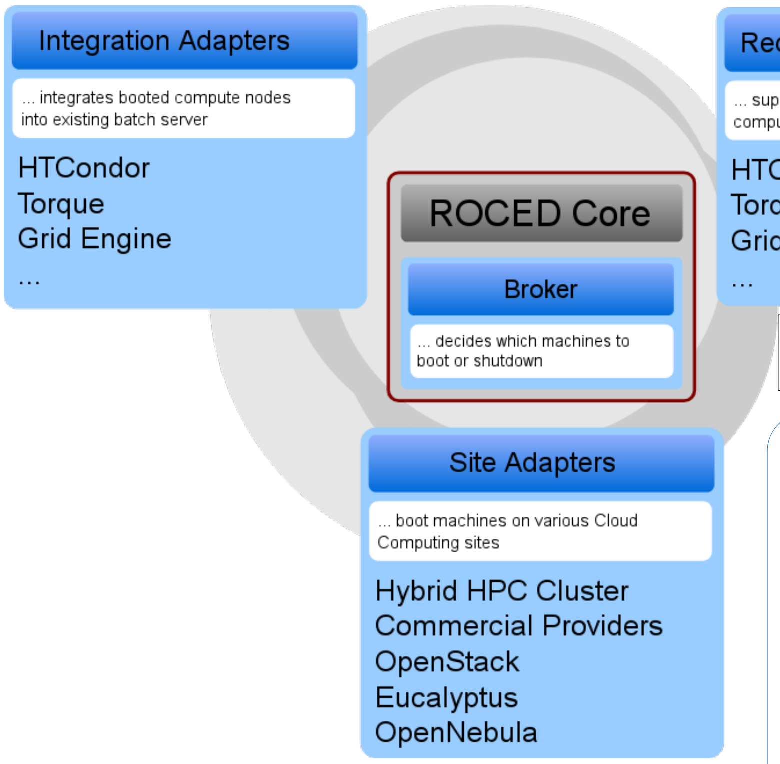
- ~3000 cores, dynamically allocated, at fives sites:
  bwFor NEMO (Fr), forHLR2 (Ka), TSystems (HelixNebula), local desktops & blade center
- low to medium I/O jobs, e.g. NNLO Calculations, Monte Carlo production, or generation of hybrid events by embedding MC in data
- Job scheduling is still partly “hand work”

Could Interface, here ROCED

ROCED (Responsive **O**n-Demand **C**loud-enabled **D**eployment):

Interface between **batch system** (Torque, HTCondor) and **cloud sites.**

- monitoring of computing needs (jobs in queue) and resources
- dynamic management of remote cloud resources:
  starting/stopping of virtualized remote worker nodes
- site adapters “know” how to provide vitualized or containerized resources



https://github.com/roced-scheduler/ROCED

ROCED has grown over time

– easy to integrate new providers

– difficult to manage many providers for many users

needs refactoring and further modularization

successor: **CObalD**

– the opportunistic balancing Demon

new development @ KIT for GridKa

Caching

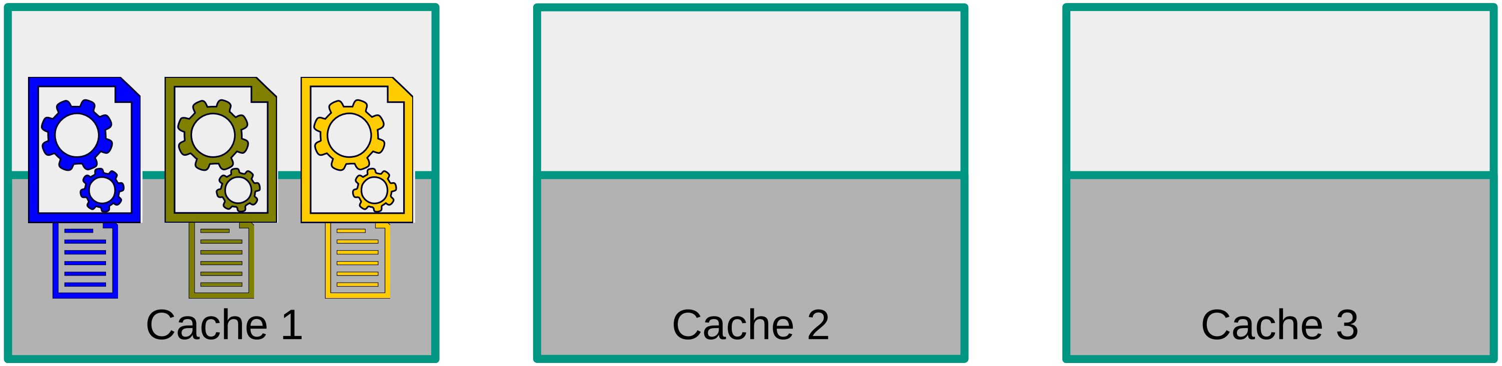
Caching is common solution for repeated access to the same data.

Cache data as close as possible to the CPU !

Suitable for

- **HEP workflows that process the same datasets frequently**
- **CPU resources without permanent storage**

Problematic on distributed resources with multiple caches:



Caching

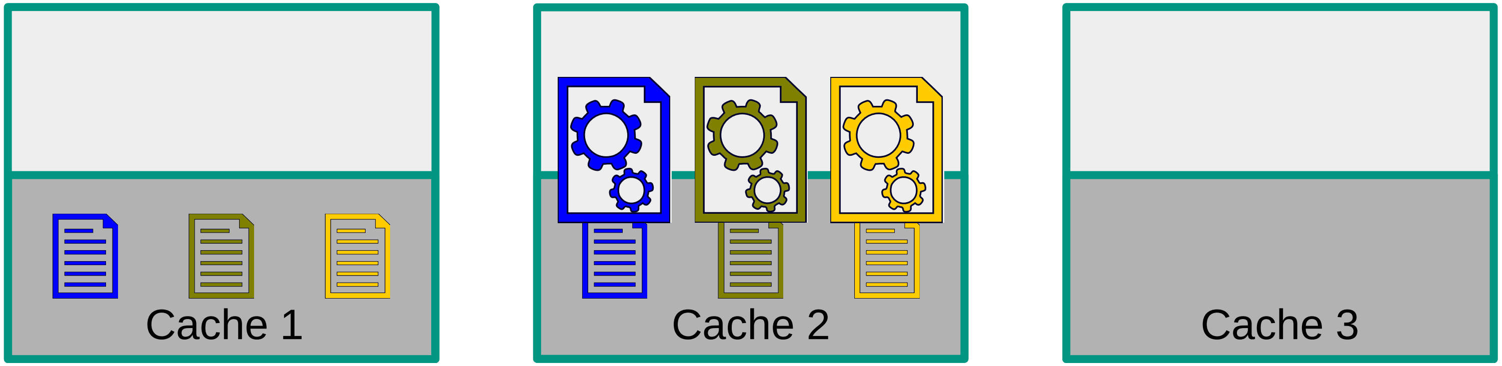
Caching is common solution for repeated access to the same data.

Cache data as close as possible to the CPU !

Suitable for

- **HEP workflows that process the same datasets frequently**
- **CPU resources without permanent storage**

Problematic on distributed resources with multiple caches:



Caching

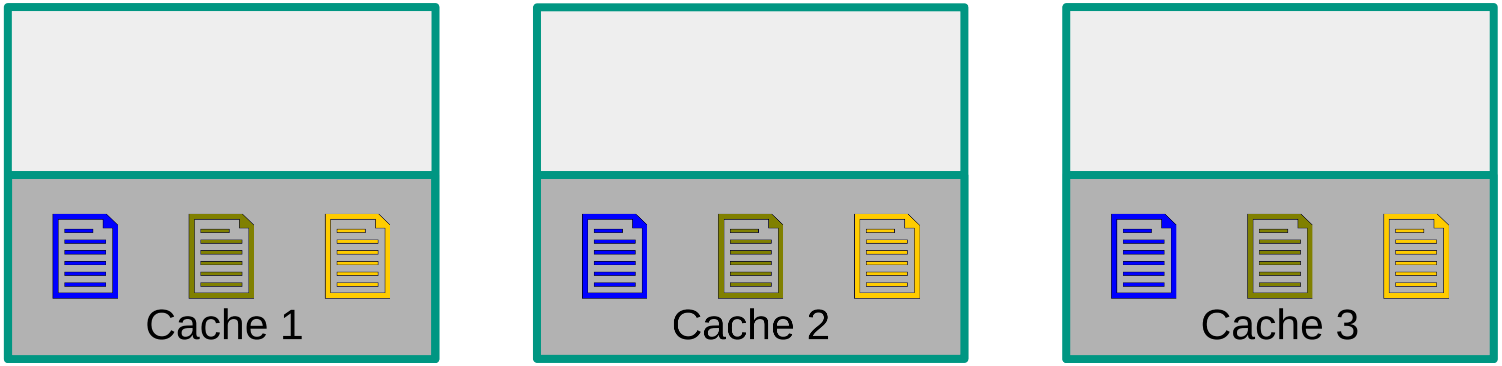
Caching is common solution for repeated access to the same data.

Cache data as close as possible to the CPU !

Suitable for

- **HEP workflows that process the same datasets frequently**
- **CPU resources without permanent storage**

Problematic on distributed resources with multiple caches:



Caching

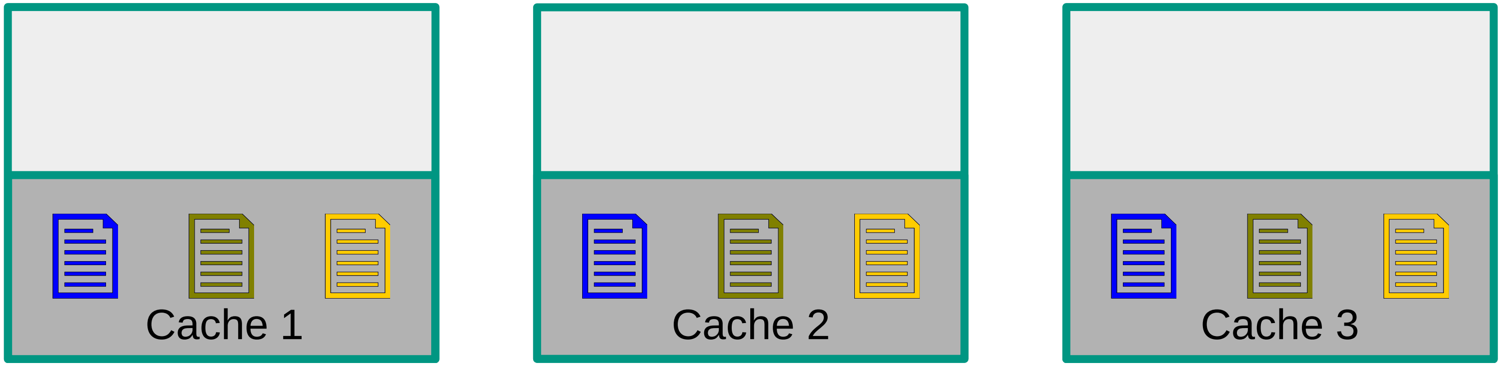
Caching is common solution for repeated access to the same data.

Cache data as close as possible to the CPU !

Suitable for

- **HEP workflows that process the same datasets frequently**
- **CPU resources without permanent storage**

Problematic on distributed resources with multiple caches:



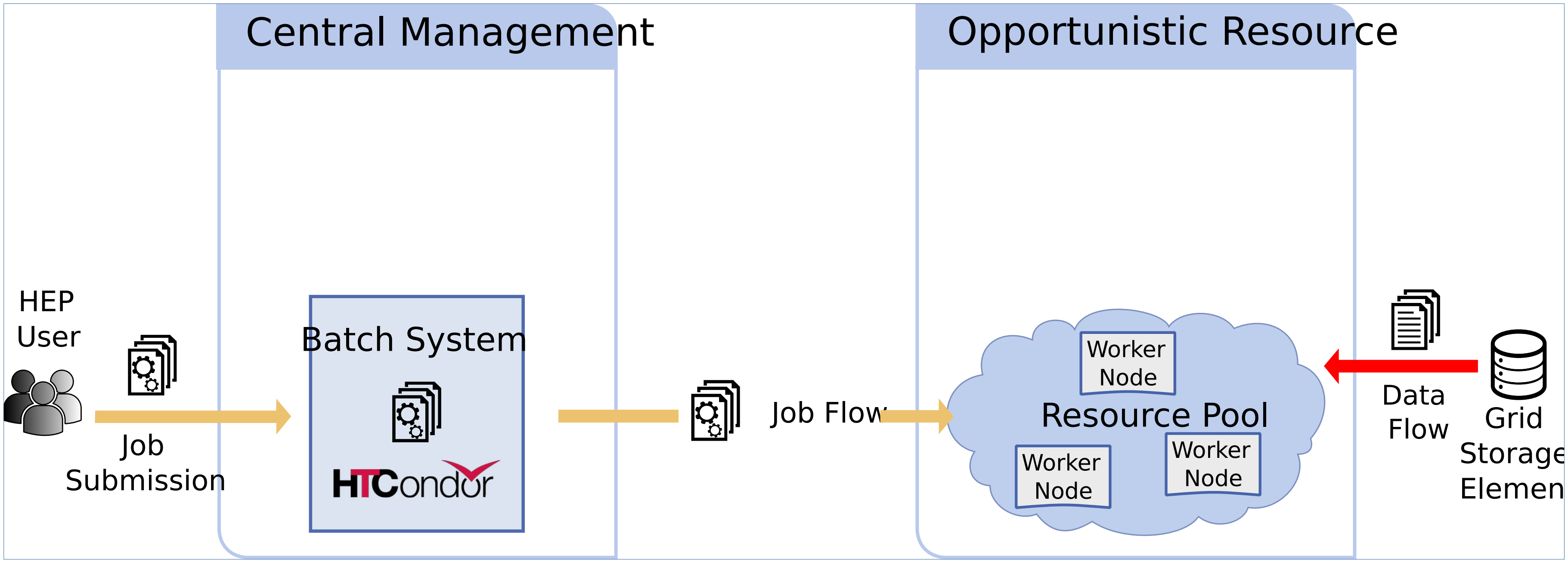
Waste of storage capacity due to replication of data!

⇒ **Caches must be “coordinated” !**

Coordinated Caching

Basic features for caching are provided by XRootD

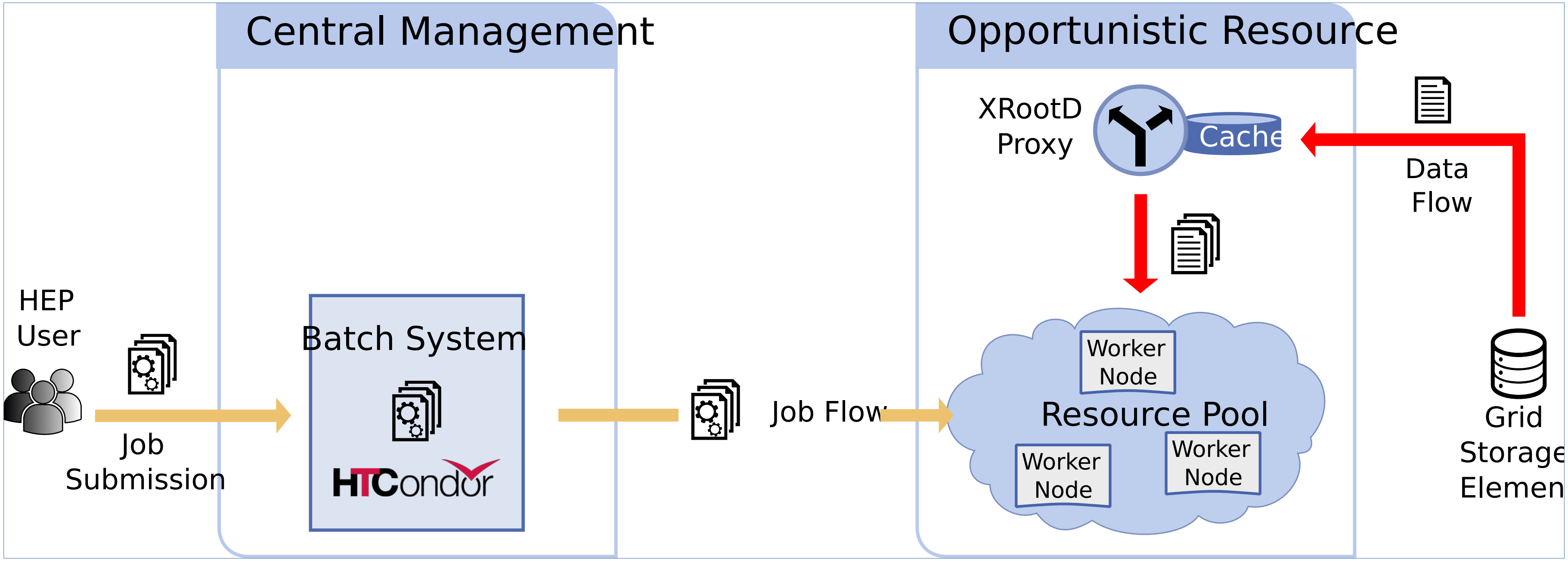
HTCondor can handle job-to-resource scheduling



Coordinated Caching

Basic features for caching are provided by XRootD

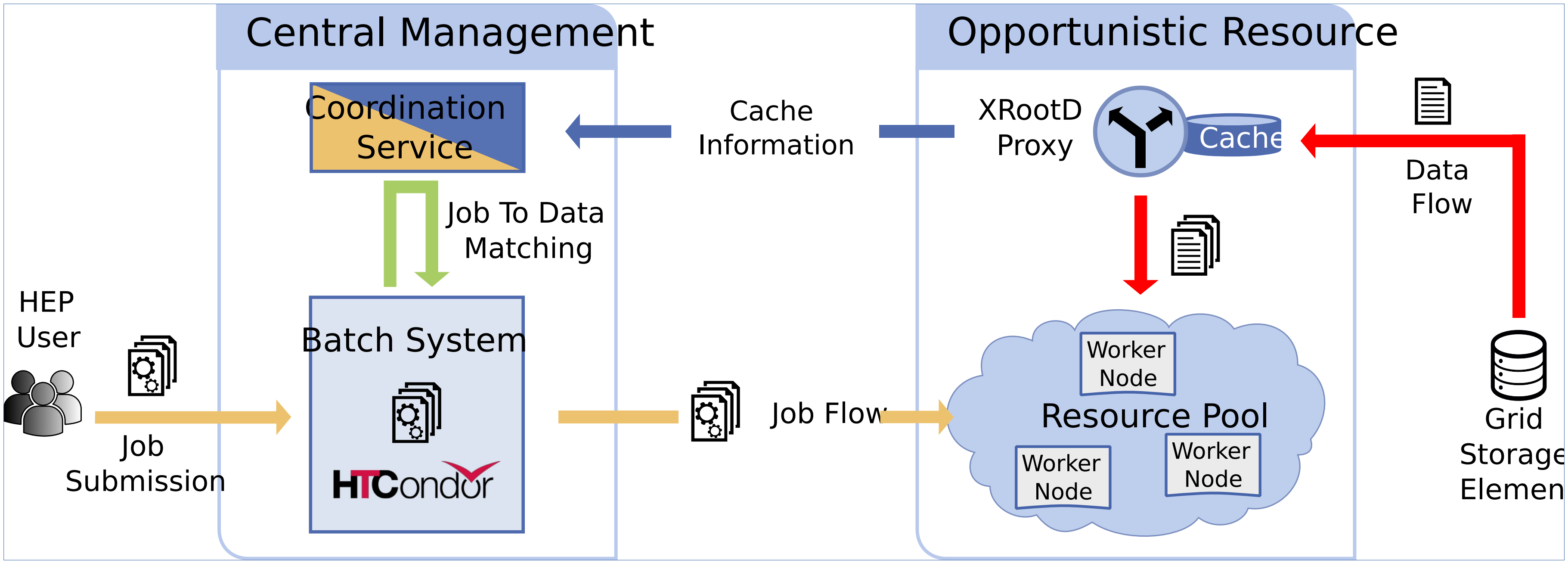
HTCondor can handle job-to-resource scheduling



Coordinated Caching

Basic features for caching are provided by XRootD

HTCondor can handle job-to-resource scheduling

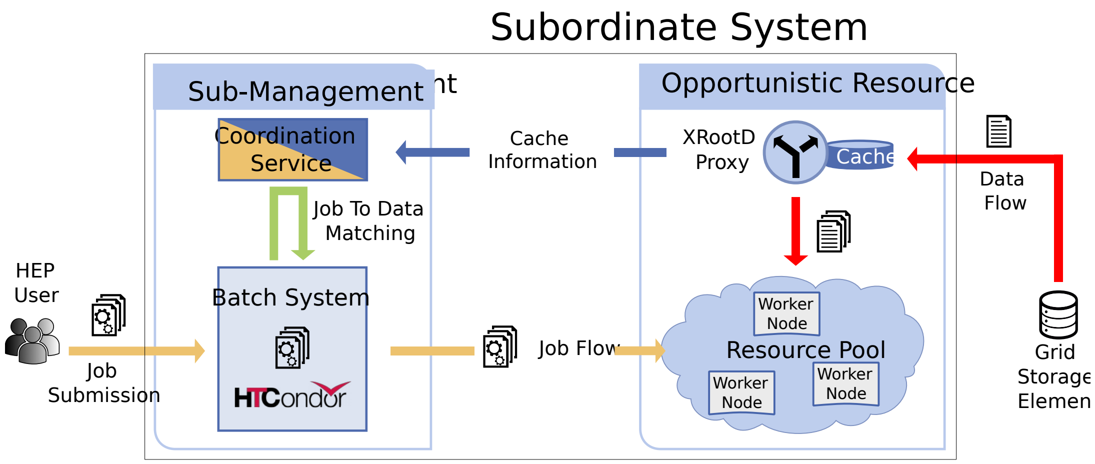


Coordination Service **NaviX** under development at KIT

based on long-term experience: "Data Locality via Coordinated Caching for Distributed Processing", M. Fischer et al., J. Phys.: Conf. Ser.762 012011 (2016)

Scalability as a design feature of **NaviX**

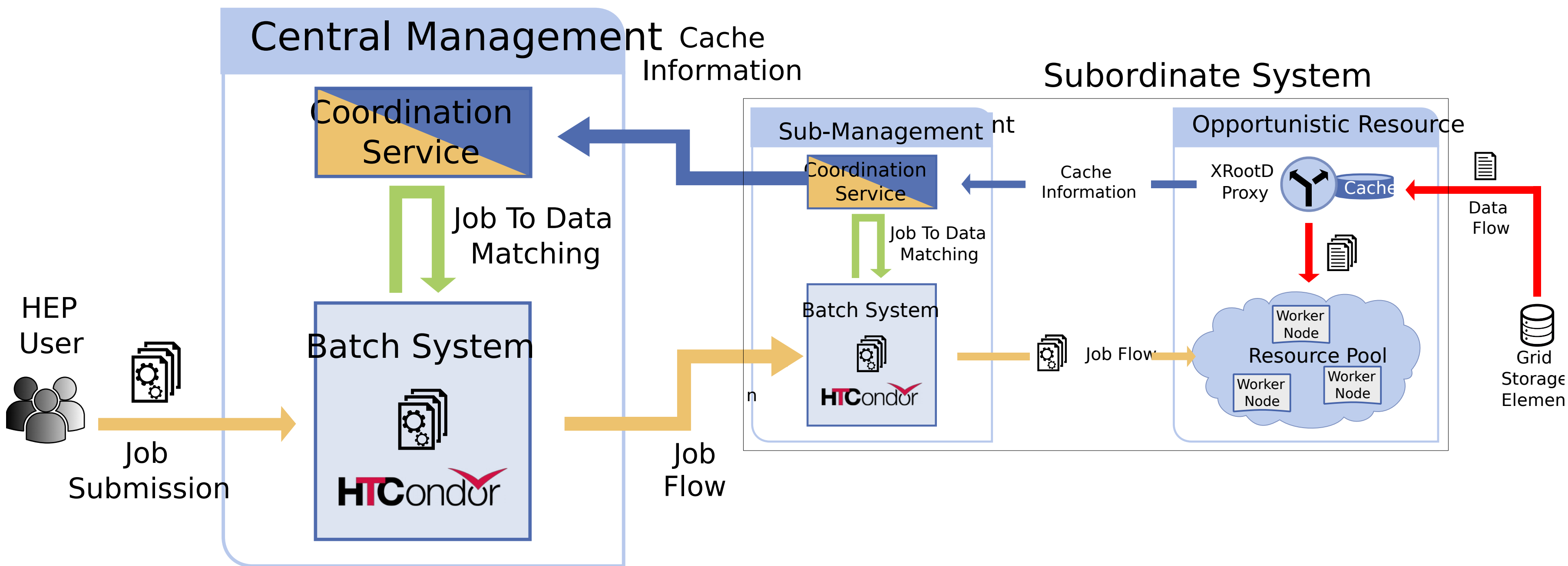
ideally, use the same components for local, site and regional caching



Scalability as a design feature of **NaviX**

ideally, use the same components for local, site and regional caching

- XRootD and HTCondor take care of hierarchical upscaling
- Job-to-Cache coordination can be performed at all levels with regard to the data location information of the subsystems.



Easy Setup of a working HEP-Site

"Tier3 in a Box"

"Tier3 must be supportable by University IT without special LHC skills"

Frank Würthwein

- Local IT supports hardware and manages user accounts
- LHC experts operate OS and services from remote

basic Software:

- HTCondor Batch system
- XroodD server, redirector & Cache
  server and cache connected to data federation of supported experiment
- CERN-VM FS and Squid server
  access to experiment software

Numerous instances already deployed at universities in the US

- an easy way to provide a working HEP environment to physicists anywhere
- concept can (and should be !) extended to **"Tier2 in a Box"**
- **can also set up remote production sites in cloud environments**
- provide "opportunistic resources" to WLCG when not (fully) used locally

however: Need **sufficient network bandwidth** (≥100 Gb/sec) to benefit

Setting up the "Box"

- VM-image(s) or container(s) set-up by (small) group of experts

  can be very lean, as most of the required software comes via CERN-VM FS

  however, some providers require using their own images,
  allowing only moderate modifications

  in such cases: can run HEP applications in a container

- permanent services (Squid server, XrootD, …) also provided in virtual machines or containers, but
  sometimes difficult to guarantee presence of these services at a given site
  (requires permanently running instances)

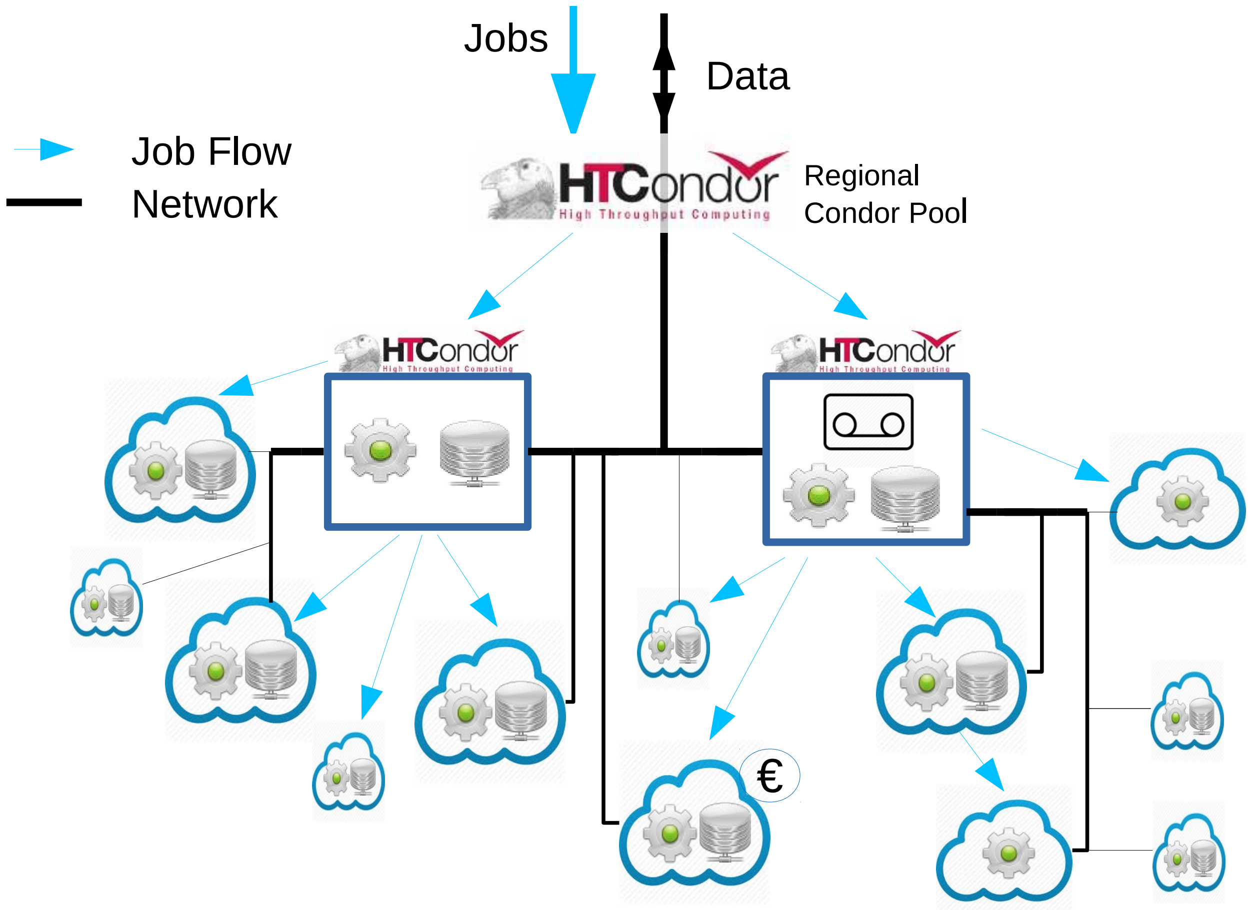
- images may contain site-specific configurations
  (require automated) image and configuration management

Containers are much easier than virtual machines !

Provisioning of VM-images (in particular) and containers
requires **additional R&D** and funding for **long-term support**

btw: since every one can contribute, this approach is
very beneficial for the training of young physicists !

Blueprint (?) for the German contribution to LHC computing >2025 ?



Conclusions

Cloud technologies for HEP sites

– have proven to work well and reliably

– allow for sharing of resources between groups, institutes and scientific communities

– give access to resources not available otherwise, incl. commercial providers

– set-up of the scientific compute environment may happen “@home”: **educational benefit for young people !**

– operate HEP sites with less effort: **“Tier 2 / 3 out of the Box”**

– Data Caching enables “diskless sites” (i.e. sites without centrally managed data store): temporary resources in clouds, institute clusters, ...

– Coordinated Caches in distributed environments reduce required disk space and increase bandwidth for data access

Cloud technologies are relevant future German contributions to LHC computing:

– Virtualised T2 and T3 ?

– bundle access to all national resources in one HTCondor pool ?

– Disk Space @ T2 and T3 as (coordinated) caches ?

ENDE