Dynamic provisioning of a HEP computing infrastructure on a shared hybrid HPC system

ACAT 2016, Valparaíso, Chile

Albert-Ludwigs-Universität Freiburg

Konrad Meier

konrad.meier@rz.uni-freiburg.de

Georg Fleig, Thomas Hauth, Günter Quast,
Michael Janczyk, Bernd Wiebelt, Dirk von Suchodoletz

UNI FREIBURG

Karlsruher Institut für Technologie

Motivation

- Modern HEP heavily relies on large computing resources
  - Simulation: CPU intensive , moderate I/O
  - Analysis: I/O intensive, moderate CPU usage
- Continuously growing demand for computing resources requires rethinking of traditional HEP-only clusters
- New approach: Hybrid High Performance Computing (HPC)
  - Using virtualization
  - User provides VM image

- Virtualization is a key component but additional ingredients are necessary to achieve dynamic provisioning of HEP infrastructure!

Motivation

Goals we want to achieve:

- Render shared HPC resources accessible → virtualization
- Dynamic allocation of resources (no static VMs)
- Integration of new resources transparent to HEP user

Our complete "virtualized HEP node" tool set:

1. Hybrid HPC Cluster : **OpenStack** (IaaS) — @ Uni Freiburg

+

2. Flexible batch system: **HTCondor**

+

3. On-demand cloud manager: **ROCED**

— @ KIT

1. Hybrid HPC Cluster

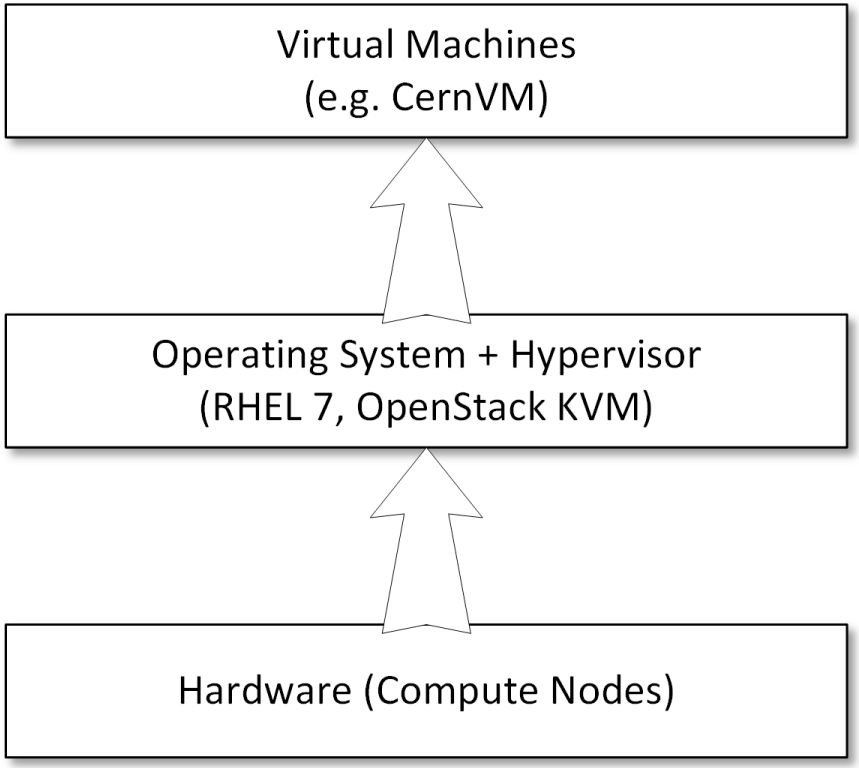
@Uni Freiburg

1. Hybrid HPC

- Provide classic HPC (“bare metal”) and virtualization on the same cluster
- No hardware partitioning between virtualization and classic HPC nodes
- OpenStack as virtualization management framework
- Allow users to provide own VM images with required software stack
- VM scheduling is integrated into HPC scheduler
- Implemented as part of the bwForCluster NEMO

1. Hybrid HPC

- Layer Model:



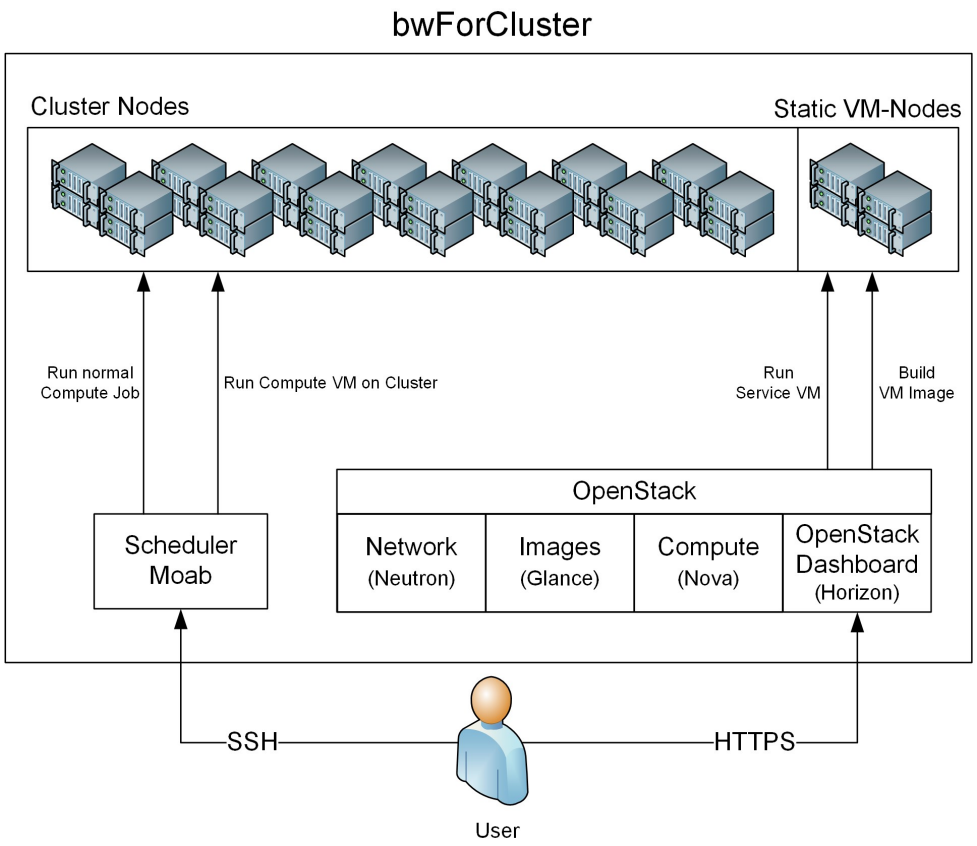
Virtualization provides:
- Virtual Research Environments

Bare metal provides:
- Resources for “classic HPC”
- Direct hardware access (Infiniband)

Provisioning:
- PXE Boot
- DNBD copy on write

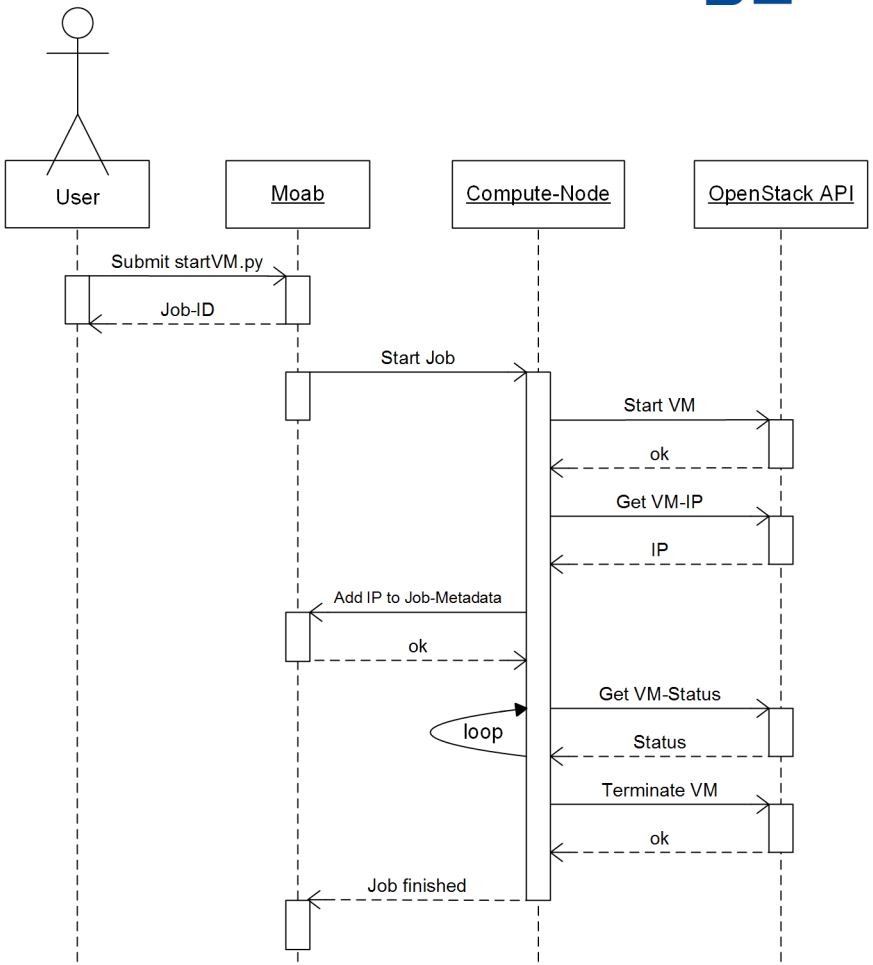
1. Virtualization Concept

- Interactive „Static VM-Nodes“ to run:
  - Build VM Images
  - Cluster services (e.g. Monitoring)
- Graphical OpenStack Webinterface provides easy access for testing/debugging and VM image creation
- For computation, VM images are started via the standard job submission procedure



1. Virtualization Concept

- Integration into HPC Scheduler
  - The integration is transparent to the scheduler
  - A VM is like any other cluster job
  - User can monitor and control the VM with standard scheduler tools (Job state, VM IP, cancel job)
  - Accounting and Fairshare are working



1. bwForCluster NEMO

- Prototype installation NEMO started late 2014 as a testbed with 1248 cores
- Located at Freiburg University, Computer Department
- Final Cluster will be available Q2 2016 with > 10000 cores
- OpenStack is deployed as a Infrastructure as a Service (IaaS) solution
- Shared by 3 diverse scientific user groups:
  Elementary Particle Physics, **N**euroscience, **M**icrosystem Engineering
- 19 Physics research groups in Baden Württemberg
  - 8 x KIT Karlsruhe
  - 5 x University of Freiburg
  - 4 x University of Heidelberg
  - 2 x University of Tübingen

2. Batch System

+

3. On-demand cloud manager: ROCED

@ KIT Karlsruhe

Karlsruher Institut für Technologie

2. Batch System

HTCondor as local and remote batch system

- Client-server-architecture
- Free & open source
- Specifically designed for HTC workloads
- Excels at integration of dynamic resources
- Integrate worker nodes beyond network zone boundaries
- Resilient, scales to >10k jobs (on a small machine, CMS setup handles >100k)
- Proven software, long term support expected
- ClassAds allow complex job routing
  - Submit jobs to local cluster, local cloud, remote cloud, ..

3. Cloud Manager: ROCED

- Developed by computing group of EKP@KIT since 2010
- **R**esponsive **o**n-demand **C**loud **E**nabled **D**eployment
  → Dynamic provisioning of cloud resources
- Modular structure, written in Python
- Independent of batch system and cloud site
  - Monitors queue of different batch systems
  - Requests VMs from several vendors/sites on demand
- Keeps track of all requested and running machines in its local machine registry
- Public release: https://github.com/roced-scheduler/ROCED

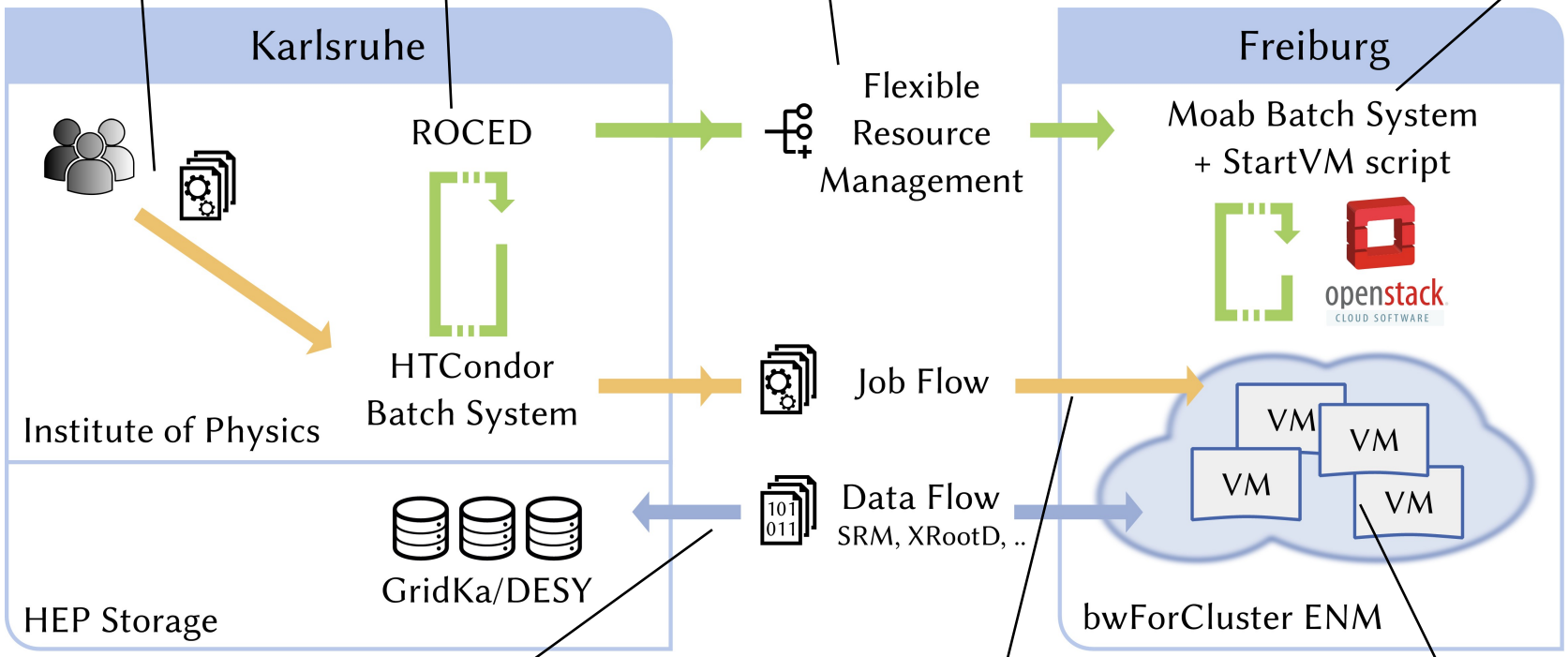
Dynamic Virtualization @ hybrid HPC Cluster

1. Submit job to local batch system

2. ROCED continuously monitors demand for computing resources

3. ROCED sends batch job containing VM request to Freiburg

4. StartVM script scheduled in Freiburg, requests VM on allocated HPC node



7. Input and output to HEP storage (e.g. GridKa)

6. Jobs get scheduled on virtualized HEP worker node

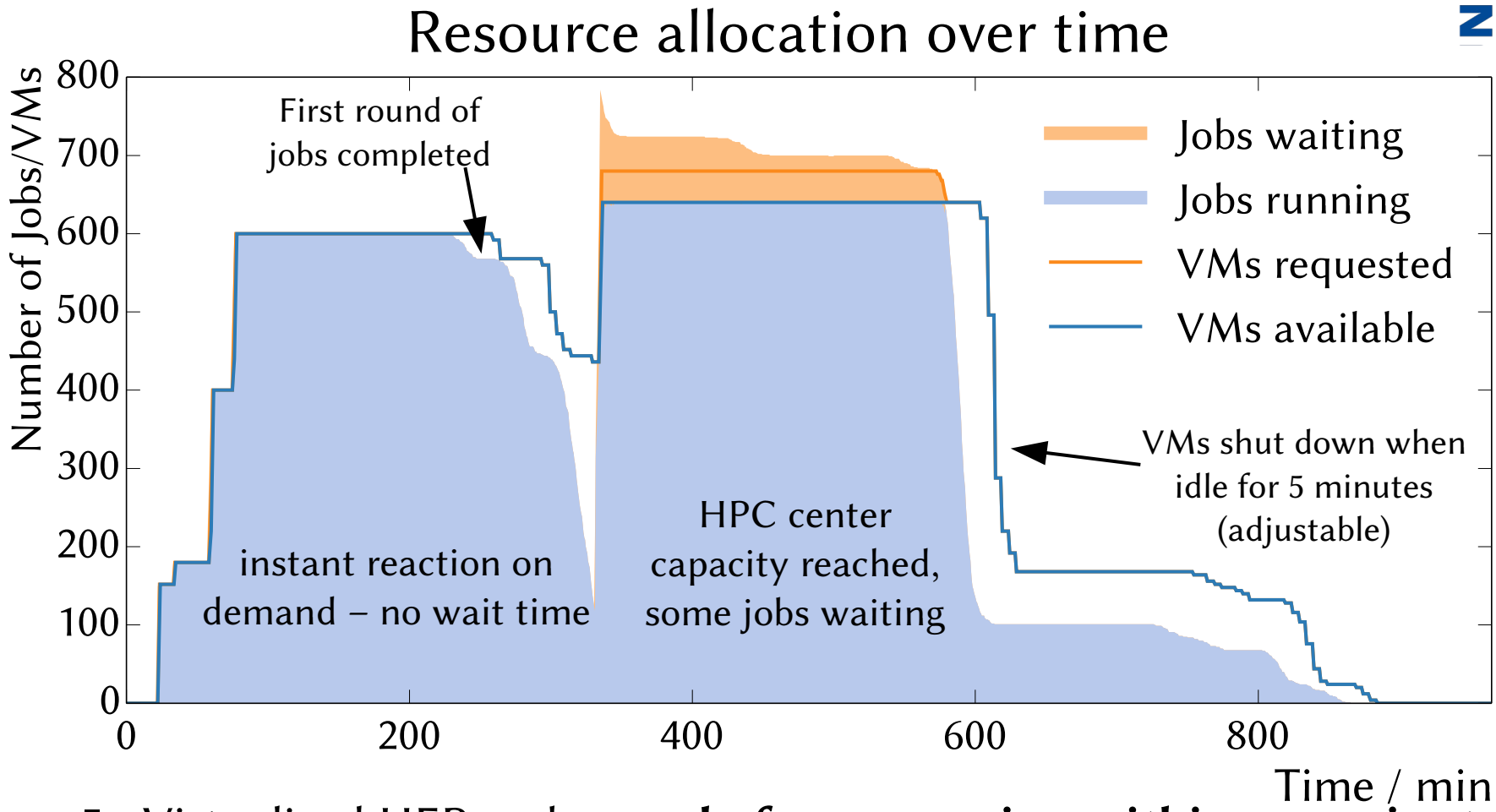
5. VM integrates in local batch system

Test and long term operation

Production Test with HEP Workflow

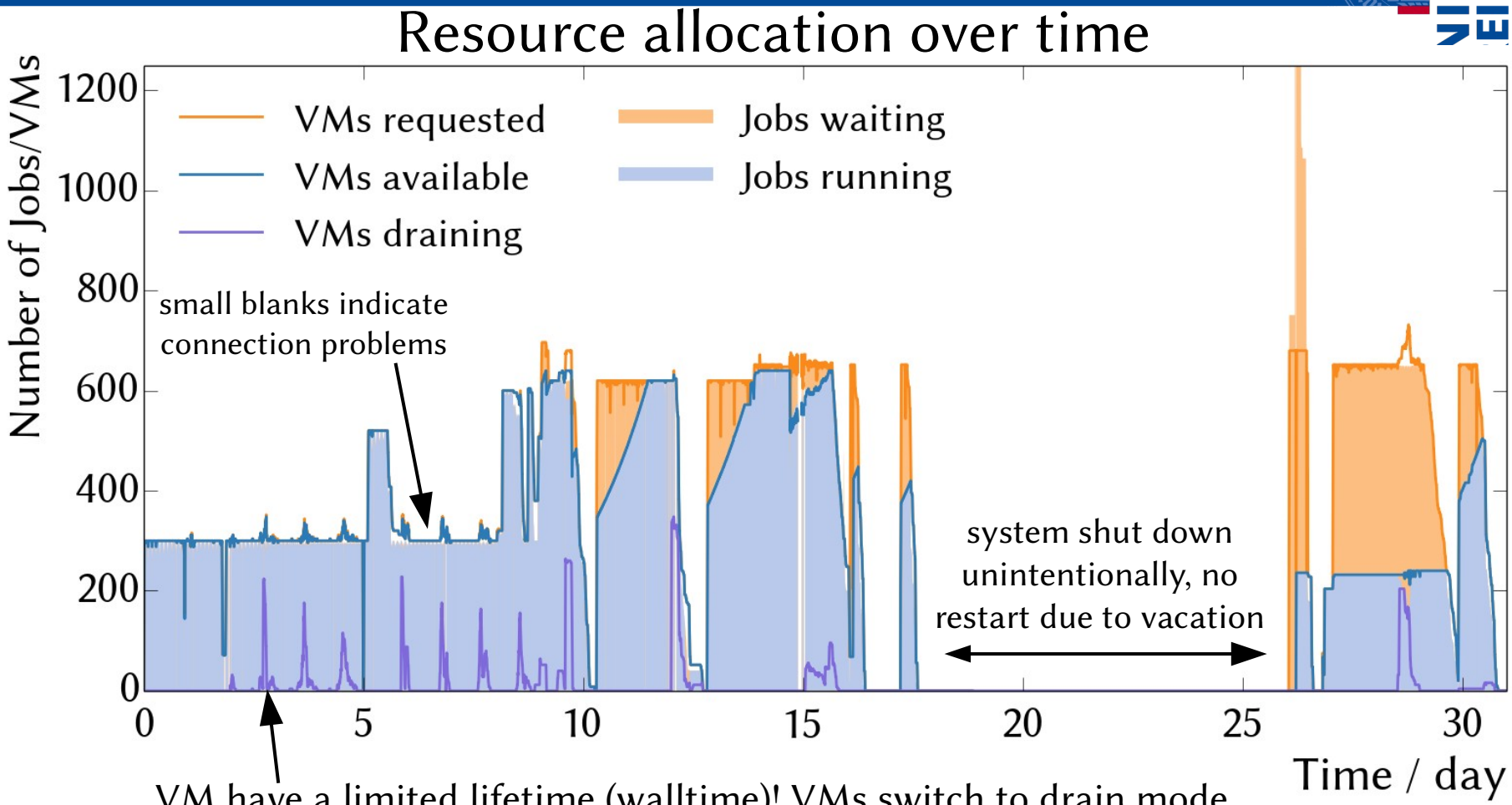
- **Testing the overall system** (ROCED, HTCondor, OpenStack) with a complex particle physics event simulation (CMS ttbar events @ 8TeV)
- VM Image with Scientific Linux 6.7 and CernVM for experiment software
- Steadily submitted new jobs of same workflow
- ~4h run time per job (shorter than usual, increases load on system)
- VM Lifetime is limited to 1 day (job walltime limit)
- Output was sent to GridKa storage (at KIT) via SRM

Production Test with HEP Workflow



- Virtualized HEP nodes **ready for processing within one minute**
- **Reuse of existing VMs** as long as new jobs are available

Long Term Usage



VM have a limited lifetime (walltime)! VMs switch to drain mode after ~1 day, some new VMs get requested to compensate for that

→ **Running stable for over a month**

Summary

- Hybrid Cluster Model provides
  - Efficient resource usage for classic HPC
  - Virtual Research Environments
  - Platform for certified software stacks (e.g. CernVM)
- Successfully rendered remote HPC resources accessible to HEP users
- Using the combined power of OpenStack, HTCondor and ROCED
- Access to new resources is completely transparent to user – no changes required in existing workflows or software
- System running stable for several weeks (over 10k jobs at a time)