Virtualizing a Batch Queuing System at a University Grid Center

Volker Büge [1,2], Yves Kemp [1], Günter Quast [1], Oliver Oberst [1], Marcel Kunze [2]

(1) University of Karlsruhe
(2) Forschungszentrum Karlsruhe

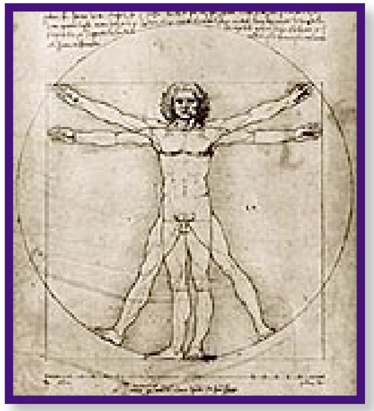
Outline:

- Particle Physics & Computing in Particle Physics
  - The Grid and the LHC Tier-Architecture
- OS problems encountered at Karlsruhe Tier 3
- Possible solutions:
  - Partitioning of clusters
  - Using XEN virtualization technique
  - Integration into Batch Queueing system
- Experience from running a prototype

Physics: Scales

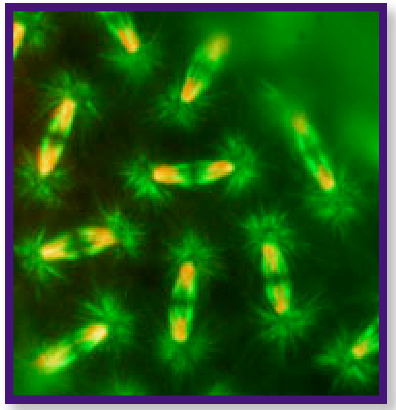
Galaxy: 10^{26}m

Earth: 10^{7}m

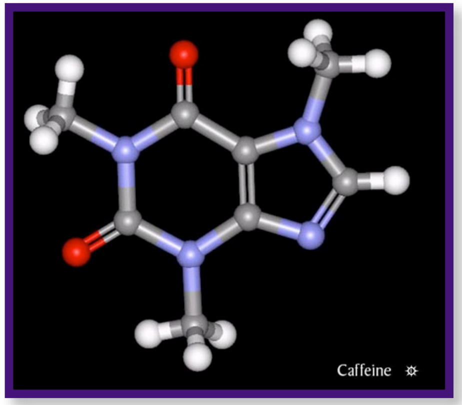


Man: 10^{1}m

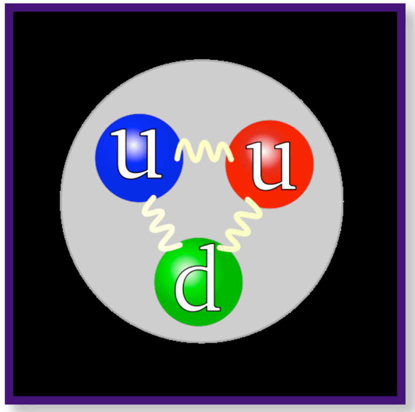
Cells: 10^{-4}m



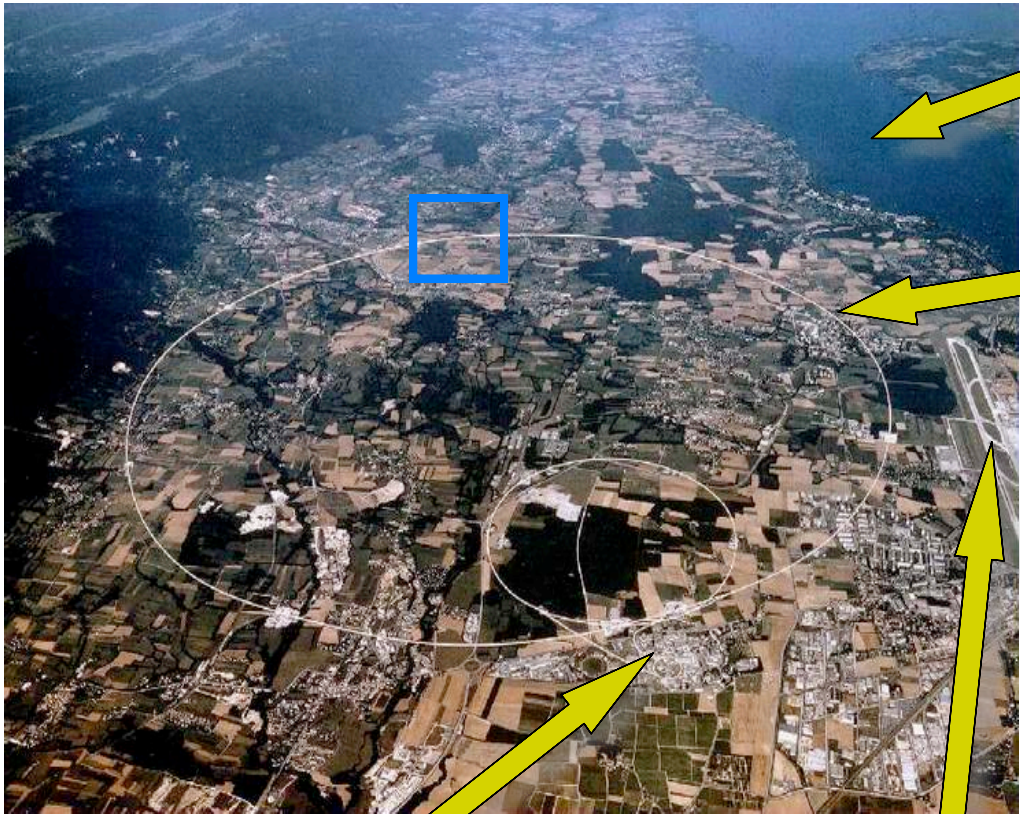
Molecules: 10^{-8}m



Elementary particles: 10^{-15}m



Particle Physics: Accelerator:



Lake Geneva

CERN

Airport

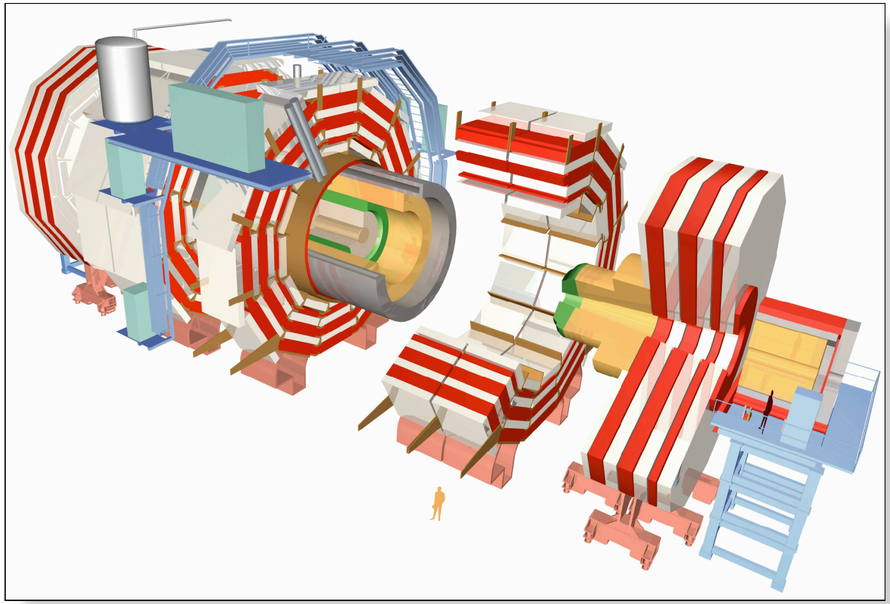
Large **H**adron **C**ollider

Circumference:	27 km
Beam energy:	7 TeV
(Proton →← Proton)	
Below surface:	100 m
Temperature:	-271 °C
Energy use:	1 TWh/a

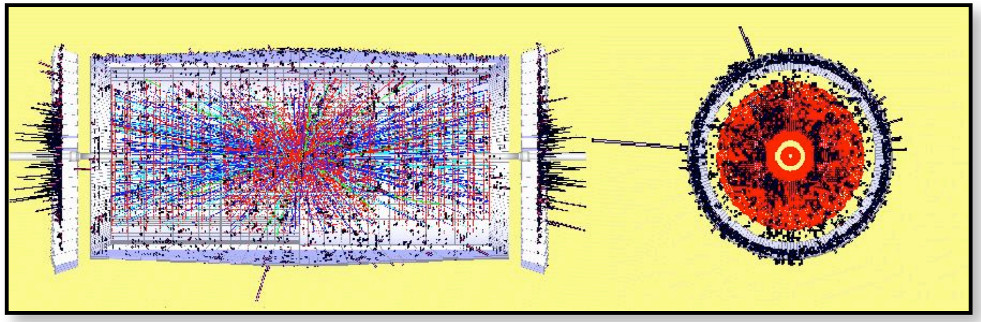
4 large experiments:

CMS	ATLAS
LHCb	ALICE

Particle Physics: Detectors:



Example of a collision:



Compact Muon Solenoid:

total weight: **12 500 T**
overall diameter: **15 m**
overall length: **21,5 m**
magnetic field: **4 Tesla**

Data rates:

Event size: 1.5 MB
Collision rate: 40 MHz
→60 TByte/s raw data
First reduction: 150 Gbyte/s
Second reduction: 225 Mbyte/s
→storage for subsequent analysis

Access to data in the LHC era

Constraints and Approaches:

- HEP experiments very expensive!
  - → redundant storage of 1.5 PetaByte per year only for CMS!
- not wise at one single computing centre
  - → distribution of data to participating computing centres all over the world
- huge datasets (~TeraByte) cannot be transferred to each user
  - → the analysis job goes "where the data set is"
- ensure access to these data for more then 2000 physicists from 182 institutes in 38 countries (in CMS)
  - → access to data and computing resources without local login for the user

The LHC experiments cope with these challenges using grid technologies – The Worldwide LHC Computing Grid

The Worldwide LHC Computing Grid (WLCG)

The WLCG Computing Model:

- computing centres are organised in a hierarchical structure
- different policies concerning computing power and data storage

The tiered architecture:

1 Tier0 (at CERN):

- *accepts raw data from detectors*
- *data transfer to Tier1 centres*

8 Tier 1 centres:

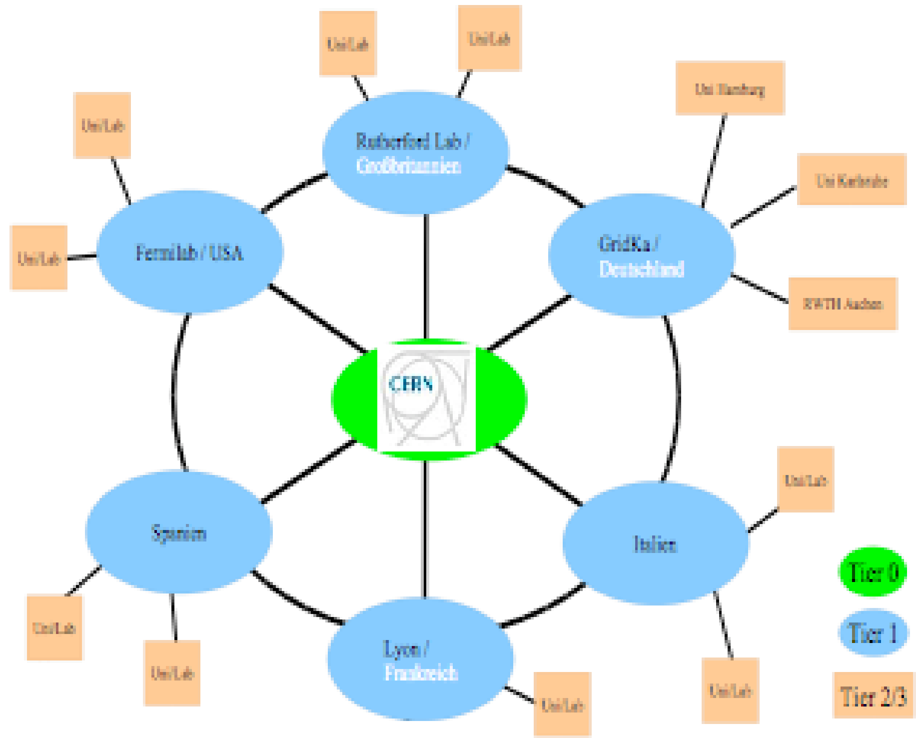
- *secure data archiving*
- *reconstruction & reprocessing*
- *coordination of associated Tier2s*

Several Tier 2 centres:

- *capacity for data-intensive analysis*
- *calibration & Monte Carlo simulation*

Multitude of Tier 3 centres at institutes:

- Offer ressources on *“best-effort” basis*



Tier 3 site at the University of Karlsruhe

- 30 Computing nodes
- 20 TB on file servers
- 100 Mbit/Gbit network

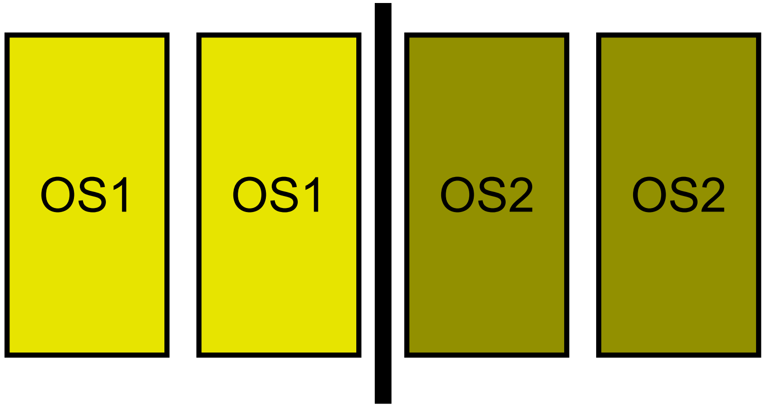
- 3 local user groups (working on different large-scale experiments)
  - CDF (20 users)
  - CMS (16 users)
  - AMS (6 users)
- Grid users through middleware:
  - Mainly CMS
  - Some CDF users (GlideCAF)

Problem: Different user groups

- CMS: Software requires SLC 3.0.X
- CDF: SL Fermi 3.0.X recommended
- AMS: Can easily recompile their software on different platforms
- gLite middleware: SLC 3.0.X recommended
- Now: Compromise possible: SLC 3.0.6 32bit
  - AMS could benefit from 64bit
- Future: Diverging needs:
  - e.g.: CMS SLC4, CDF SLC3
  - e.g.: CMS needs both SLC3 and SLC4
  - e.g.: Some need 32bit, other 64bit.
  - Sharing with other groups using modern distributions
- Additionally: Security issues:
  - Different user groups, same cluster

→ Partition your cluster! But how?

Static (vertical) partitioning



Example:

- 4 nodes, 2 groups
  - 2 nodes with OS1
  - 2 nodes with OS2
- Sharing common storage, network and control infrastructure

- Changes in the resource allocation difficult
- Old OS on new hardware problem persists
- No real resource sharing possible

Dynamic (horizontal) partitioning

- All nodes have two OS running all the time
- The OS needed gets all CPU and RAM resources
- Sharing all resources

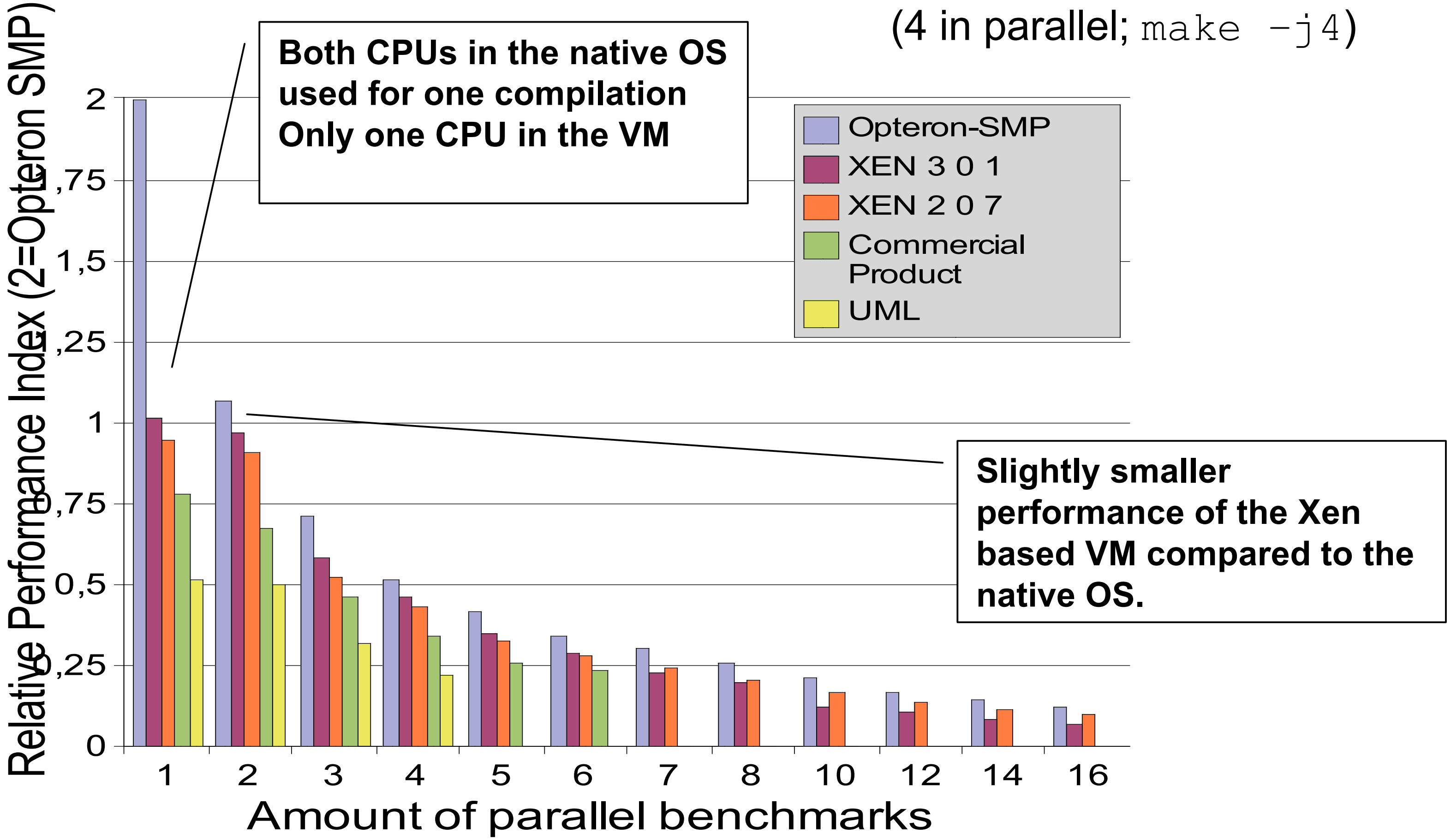
Using Virtualization

- Dynamic and fast changes in resource allocation
- Only host OS must fit the hardware
- Security and privacy through encapsulation

Virtualization Techniques: Performance comparison

Standard application benchmark: Linux kernel compilation

(4 in parallel; `make -j4`)



Performance considerations

- No noticeable performance loss due to virtualization:
  - Around 3-4% loss for CMS software (32 bit)

- Even performance gain is possible:
  - AMS group could benefit from 64 bit, but 32 bit common agreement
  - Galprop (AMS main application) runs 22% faster in a virtual 64-bit machine than on 32-bit native system! (Same Opteron hardware)

→ A overall performance gain can be possible

(at least no drastic performance losses)

Benefits:

- Optimal OS for each user group
- Security and privacy through encapsulation
- Enables possibiliy of desktop harvesting:
  - Desktop OS must support virtualization
- Easy deployment of OS
  - VM can be managed centrally
  - small adaptions at local site
  - local site admin only in charge of Dom0

Connection to the Batch Queue

Users do not login to the nodes: Using Batch Queuing Server!

Users are not to control the resources: Batch Queuing Server?

- The different VM running on one host are not independent:

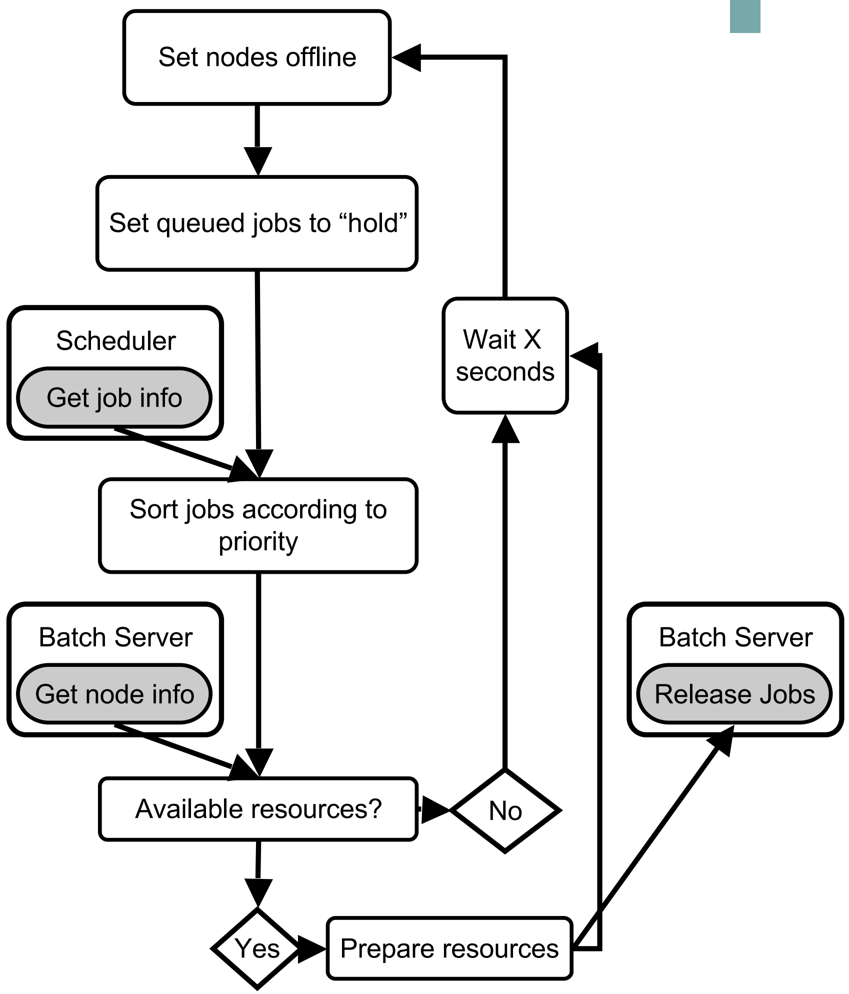
  They share the same resources
- The batch queue server must know about this sharing
  - Either natively
  - Or with the help of a separate process

Requirements of such a process:

- Independence of batch system server and scheduler:
  - No modifications
  - Flexibility
- Respect current policies:
  - Node occupancy
  - Prioritization

Implementation of process: Daemon

- Maui/Torque (used at EKP): Used to show working principle of daemon
- Daemon implemented in Perl language
  - First running on test system:
    - 2 Dual-Opteron simulating 19 nodes with 2 different OS
  - Now: Running on EKP production system with real users and real jobs
  - Working stable
- No changes to Maui/Torque necessary
- No change noticeable for users
- (Should be) easily adaptable for other products



Experiences at EKP

Setup

- 15 dual-core AMD64 boxes
  - 32bit ScientificLinux 3 VM
  - 64bit Debian VM
- 11 single-core 32bit boxes
  - 32bit ScientificLinux 3 VM
- 2 jobs per core
- Only one VM active (i.e. all memory and jobs)
- Host: Small debian 32/64bit with Xen 3.0.2 patched 2.6.16.27 kernel

Experiences during testing

- ~500 user jobs served
- Transparent to user
- No jobs lost or on wrong machine
- Lacks flexibility:
  - Advanced configuration in PBS only difficult to implement (max_jobs per user,…)
  - Reimplementation in daemon
- **Native integration in batch server preferable!!!**

Conclusions & Outlook

- Particle Physics heavy users of grid computing
- Grid computing profits from virtualization techniques
- XEN most utilized virtualization technique
  - Very good performance behavior
- Virtualization of the Computing Nodes necessary:
  - Different user groups: OS & security, desktop harvesting…
- Integration into existing Batch Queuing system possible:
  - Daemon working stable on EKP production system
  - **Native integration preferable**
    - Grouping of nodes to resources
    - “pbs_hardware_mom” (running on host, governing VMs and contacting PBS server)

Horizontally partitioned cluster demanded in Grid computing!

Thanks to Christophe Saout, Michal Kreps, Iris Gebauer and the EKP Admin team for their help!