

Clouds in WLCG

Claudio Grandi (INFN Bologna)

Disclaimer

- This is not a technical presentation – Will not discuss technologies and compare solutions
- Try to give an overview of the activities and of the motivation behind them
- Try to identify possible evolution paths in the short/medium term
- Try to identify possible pitfalls

Why Clouds in WLCG?

- CERN perspective
 - Outsource hardware maintenance (distributed Tier-0)
 - Simplify internal resource provisioning (IaaS)
 - Industrial standard: end of EC-funded projects era
- Experiments perspective
 - Natural extension of pilot-jobs – Gain control on resources
 - Ease the reuse of specialized *online* farms to *offline* tasks
 - Industrial standard: buy resources on the market if needed
 - Not on the critical path, though
- Sites perspective
 - Virtualization simplifies support in multi-VO sites
 - In future may have access to industrial software
 - May join the resource provisioning business (?)

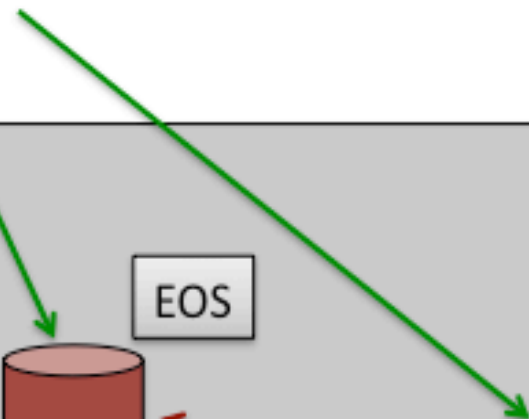
Distributed Tier-0

- Extension of the CERN Tier-0 at Budapest
 - Batch and disk storage split across the two sites



Tier-0 now

LHCOPN
LHCONE
Internet



Tapes



CASTOR

EOS



Batch

Detectors

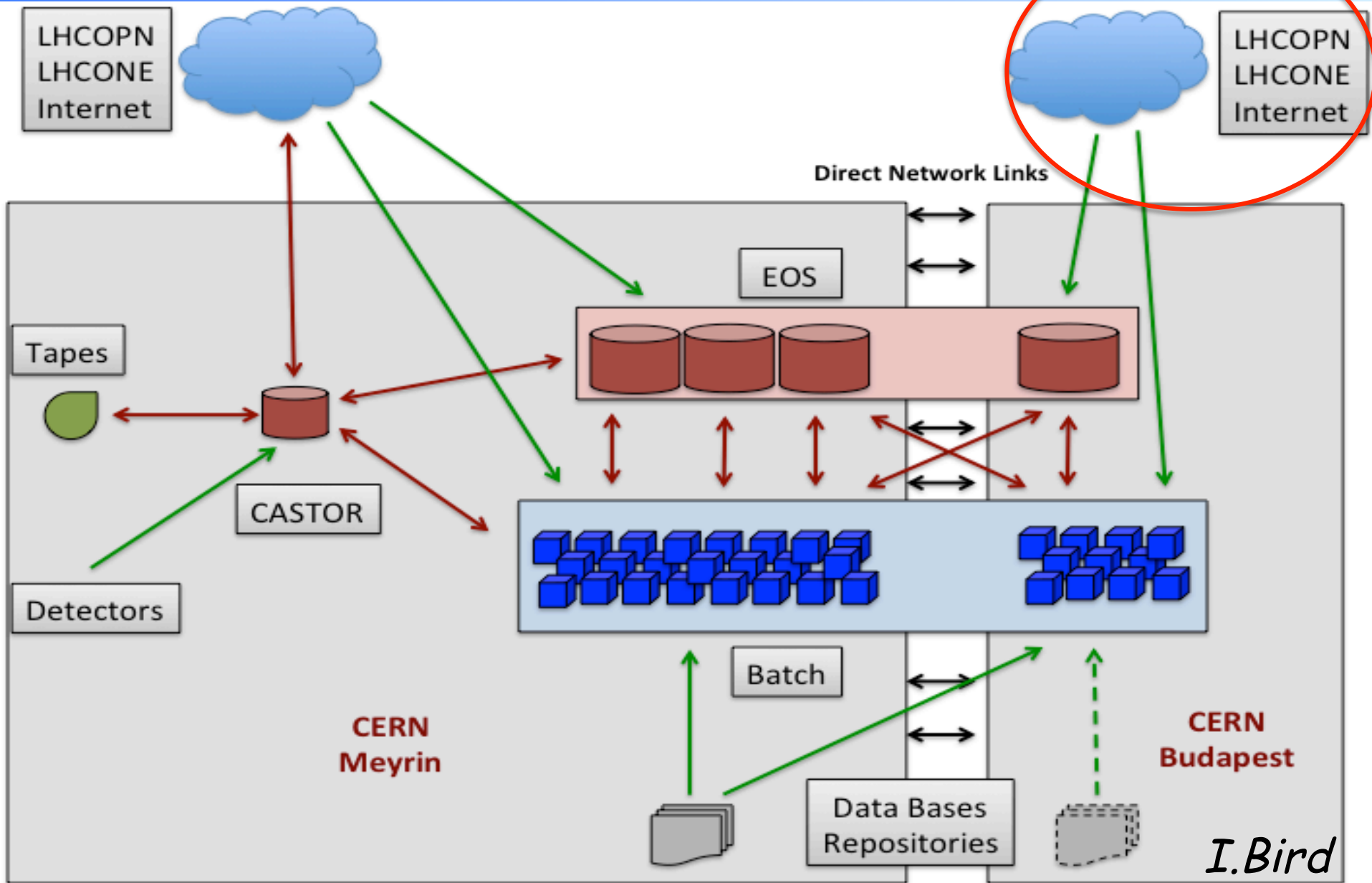
Data Bases
Repositories



CERN
Bld 513+613

I.Bird

Distributed Tier-0 ?



Distributed Tier-0

- 2x100 GbE already active
- Work ‘as-if’ in another building
 - 30 ms latency

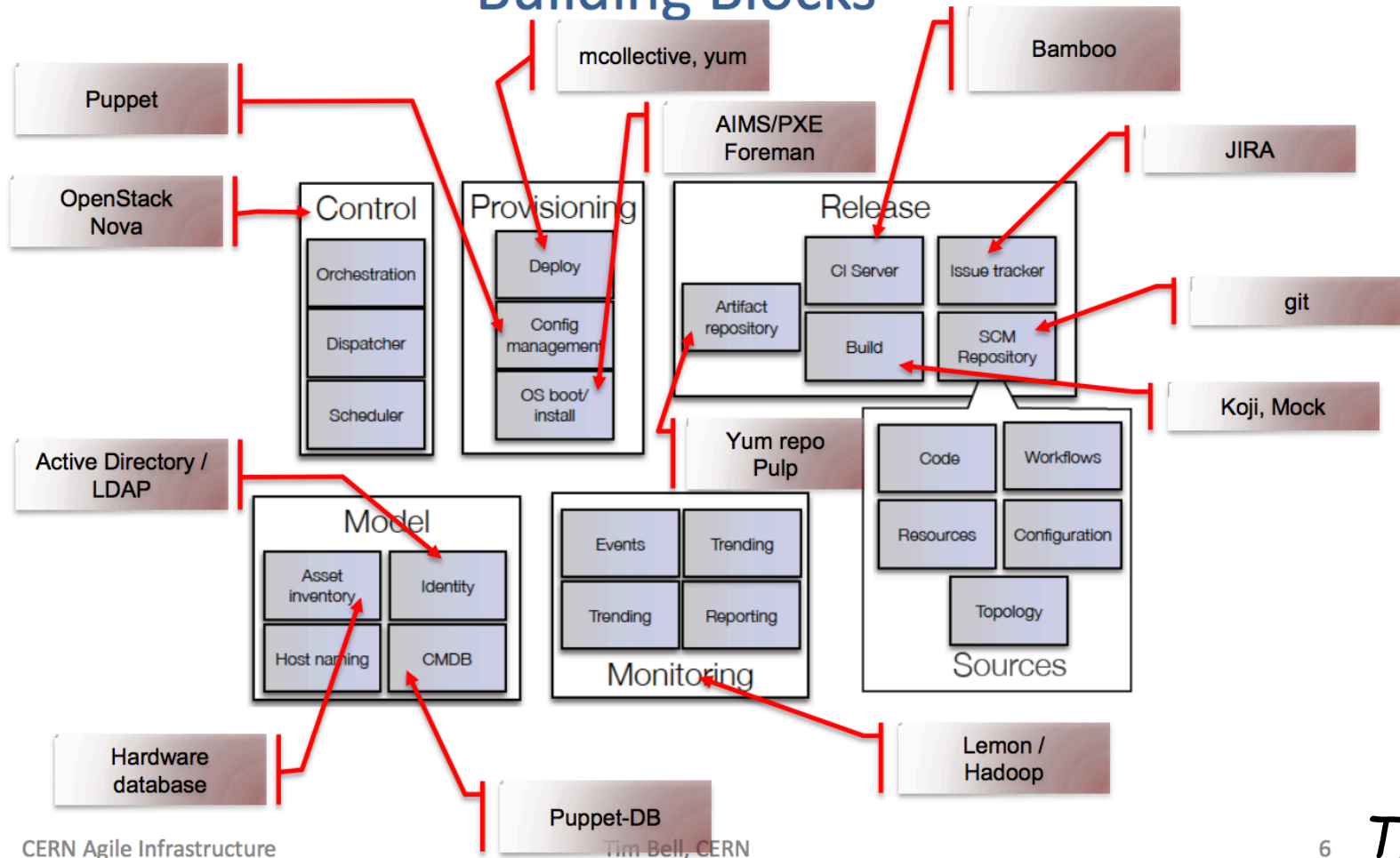


- Access to remote resources via a Cloud interface
- Developing the CERN Agile Infrastructure to manage local and remote resources

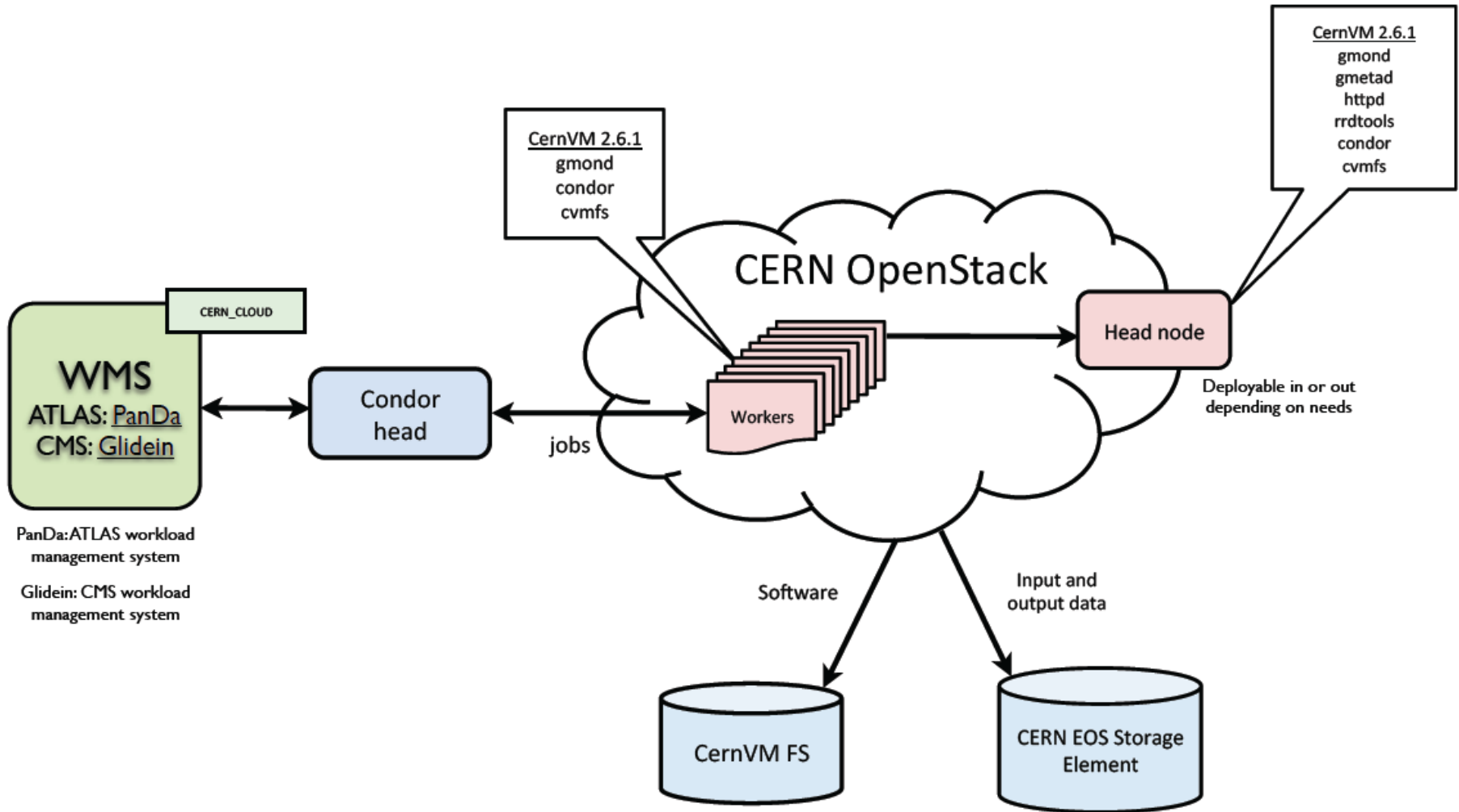
CERN Agile Infrastructure

- Provide a new toolset to manage CERN resources
- Uses OpenStack to control resources



Building Blocks



Experiment testing on Agile

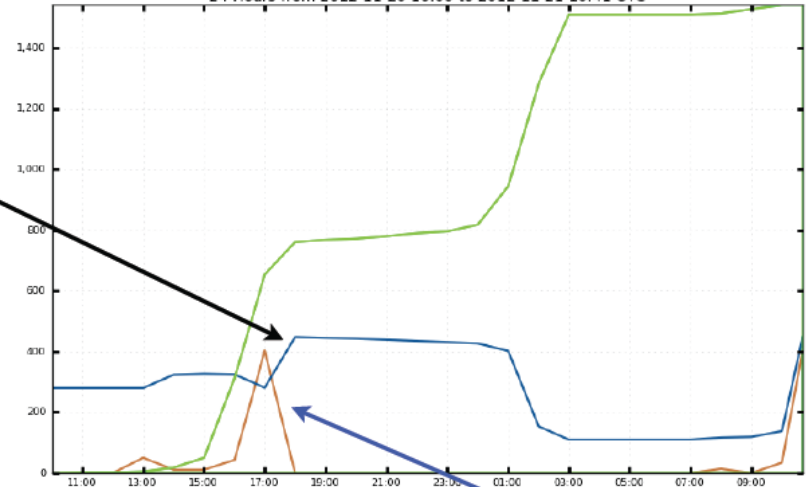


Experiment testing on Agile

- Tests on Agile by ATLAS and CMS (LHCb starting)
 - ~ 1600 cores (400 4-cores VMs, 2 GB/core)
 - CernVM for image provisioning and contextualization
 - Using CernVMFS and Xrootd (EOS) to access software and data
- Statically allocated VMs (euca-tools, nova client)
 - Switching to automatic provisioning (see later) 
- ATLAS: HammerCloud tests and real production
- CMS: MC production and analysis
- First results:
 - Very good job efficiency (less than 1% failures)
 - CPU efficiency impacted by choices on local disk access
 - Initially up to 20% inefficiencies, now mostly understood 



OpenStack CMS testing
24 Hours from 2012-11-20 10:00 to 2012-11-21 10:41 UTC

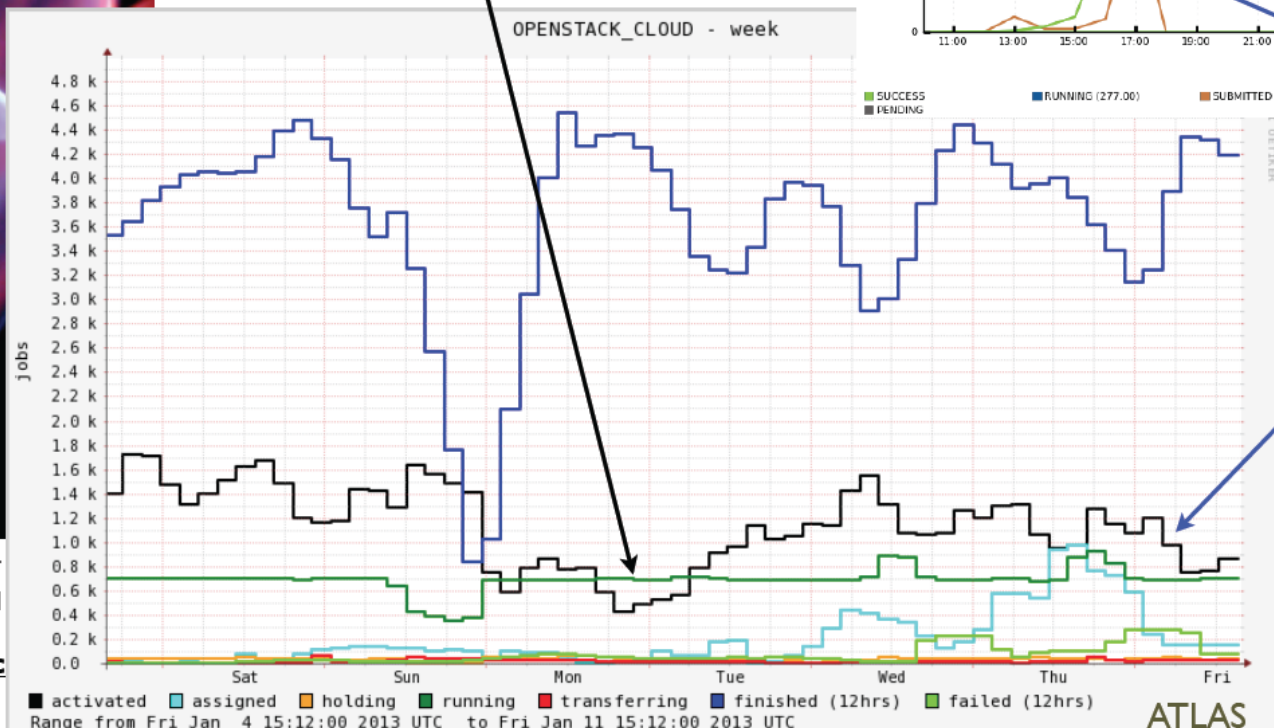


~450 jobs running in parallel

~700 jobs running in parallel

pending jobs correctly absorbed

Plots taken from PanDA monitoring and Dashboard



ATLAS

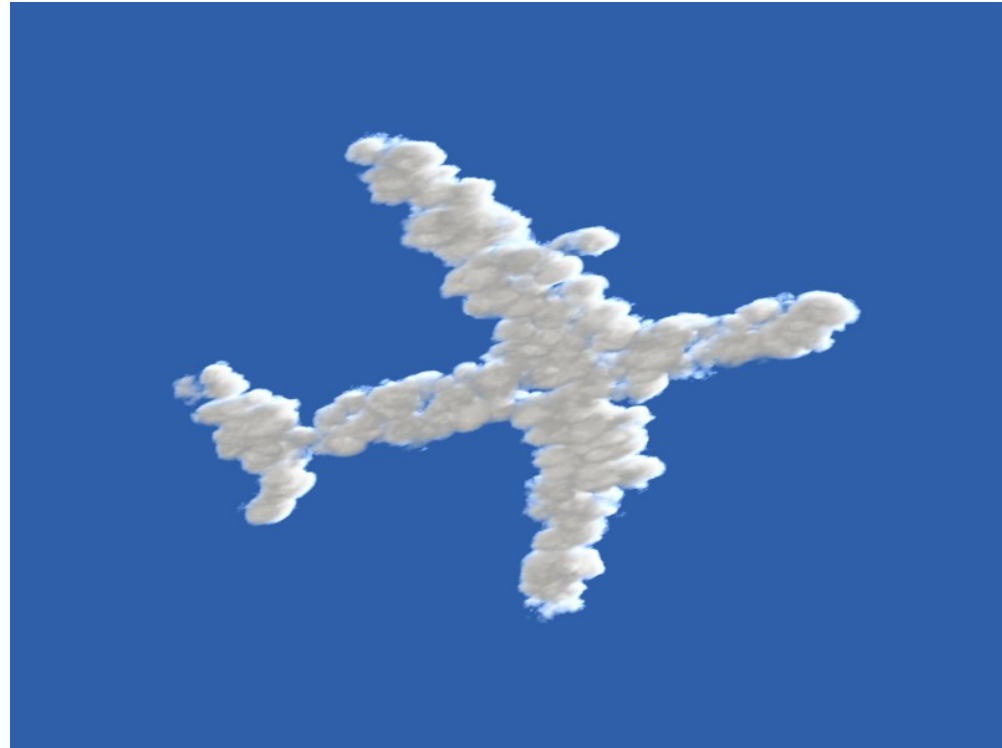
M. Cinquilli



CERN - 15 January 2013

From pilots to clouds

- Pilot-job frameworks implement separation of resource allocation and job management
 - Job management is entirely in experiments domain
 - Resource allocation continues to be shared
- Cloud interfaces can't do job management but are optimized for resource allocation
 - A pilot-job framework can be extended to use a Cloud interface instead of a Grid interface to automatically provision resources without modifying the job management part

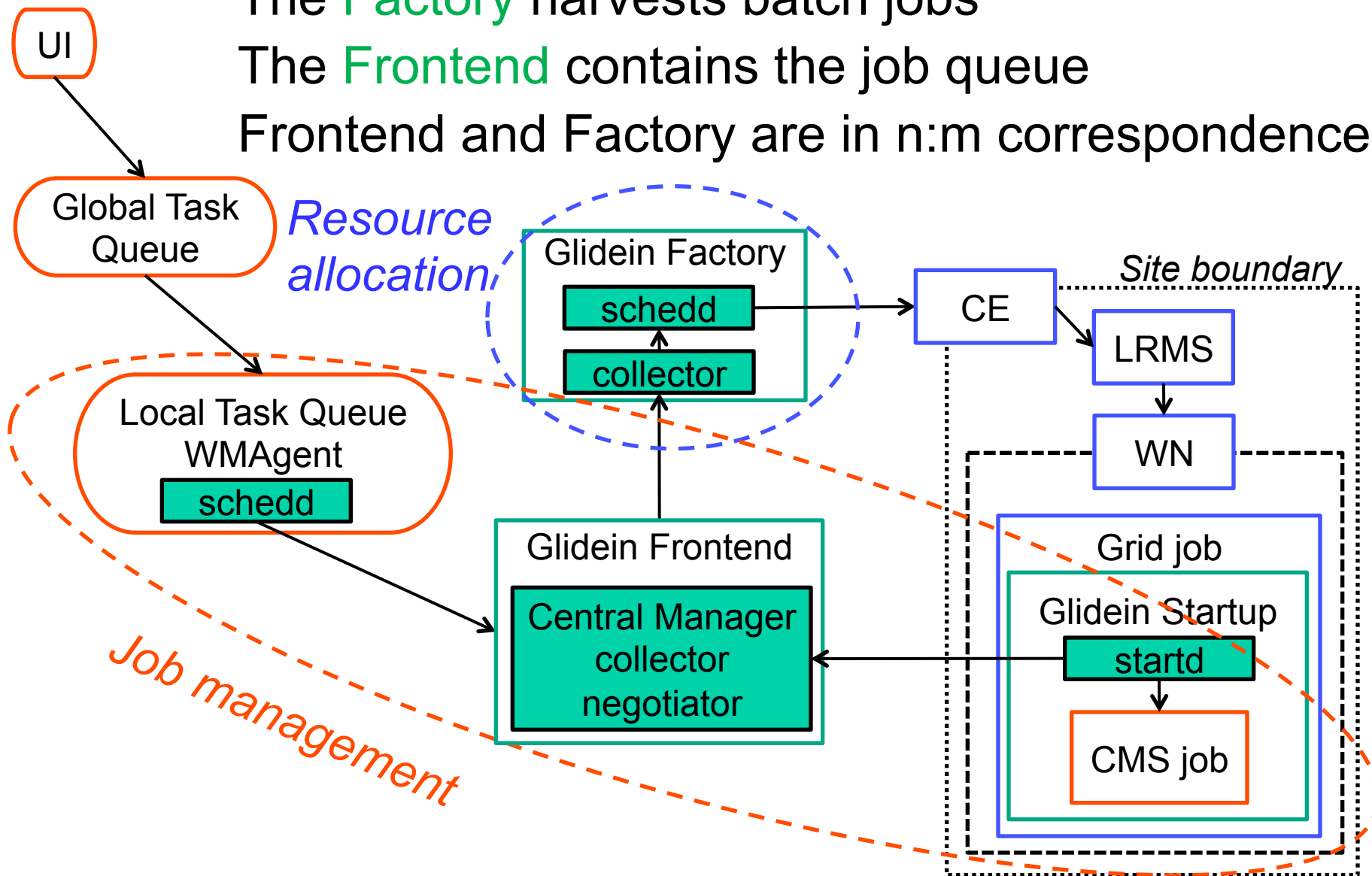


Example: CMS glidein system

The **Factory** harvests batch jobs

The **Frontend** contains the job queue

Frontend and Factory are in n:m correspondence

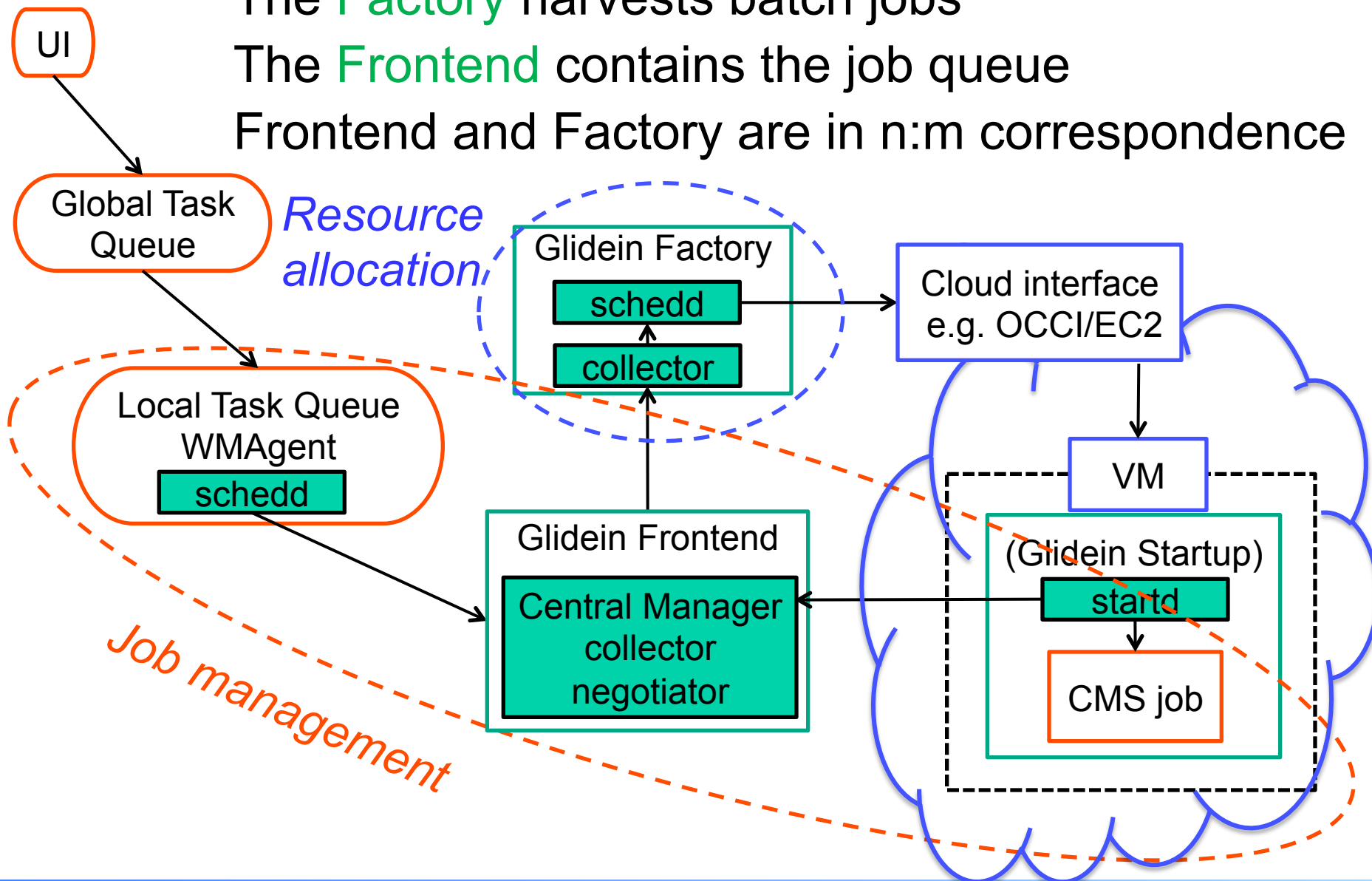


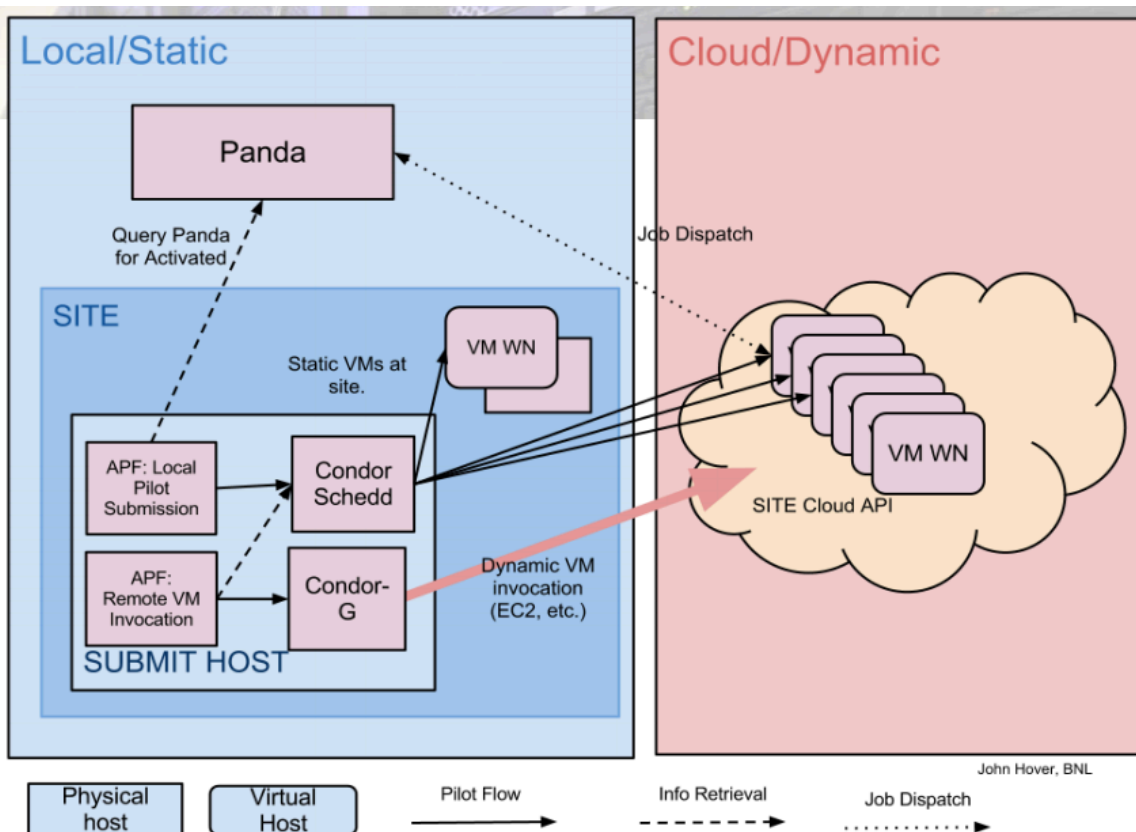
Example: CMS glidein system

The **Factory** harvests batch jobs

The **Frontend** contains the job queue

Frontend and Factory are in n:m correspondence

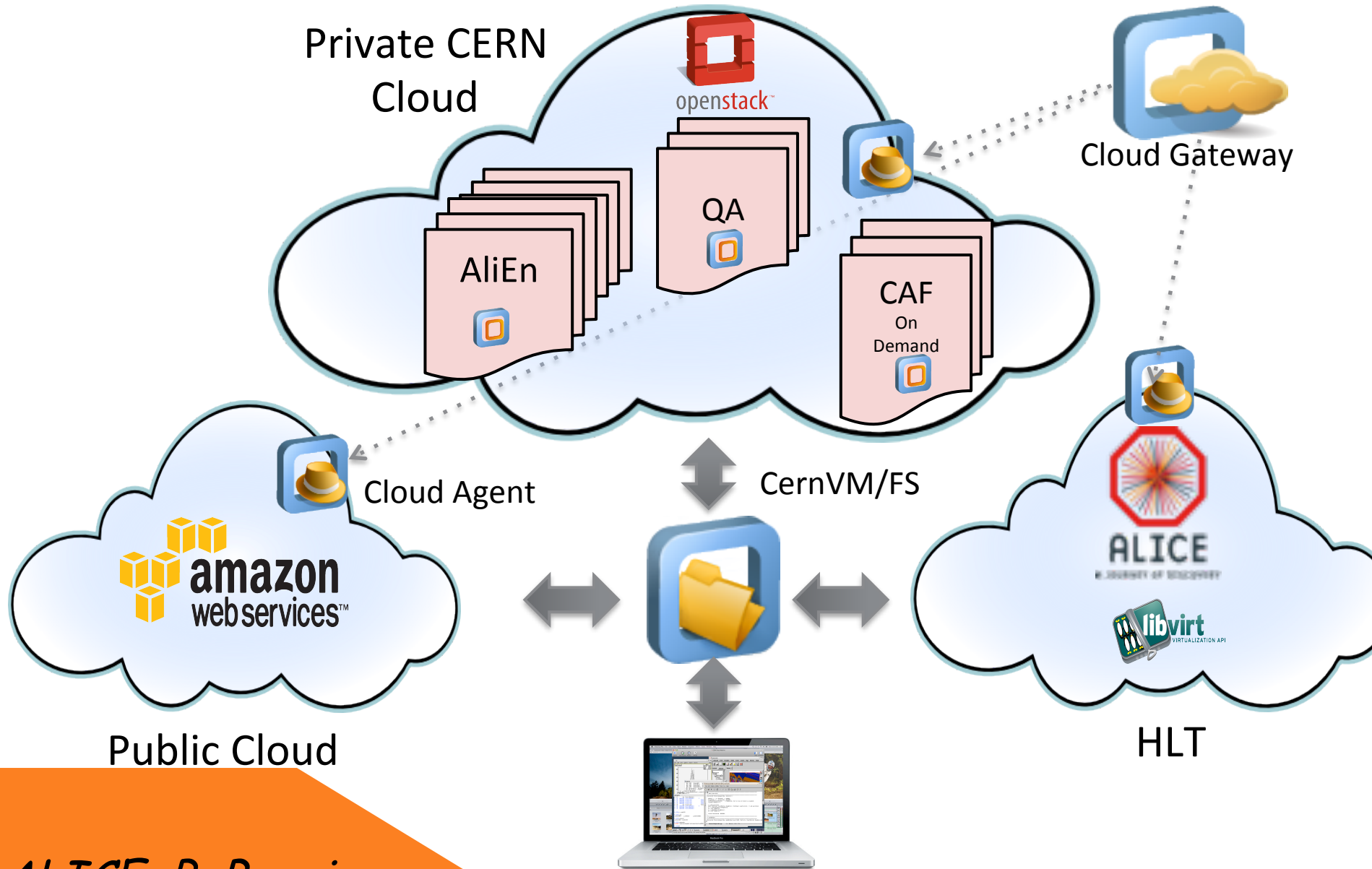






- Under development in BNL
- **Prototype by end February**
- Full support for VM lifecycle
 - Flexible algorithms to boot/terminate running VMs
 - Cascading hierarchies: programmatic scheduling on *local* → *private* → *commercial* clouds based on job priority and cloud cost

ATLAS, F. Barreiro Megino

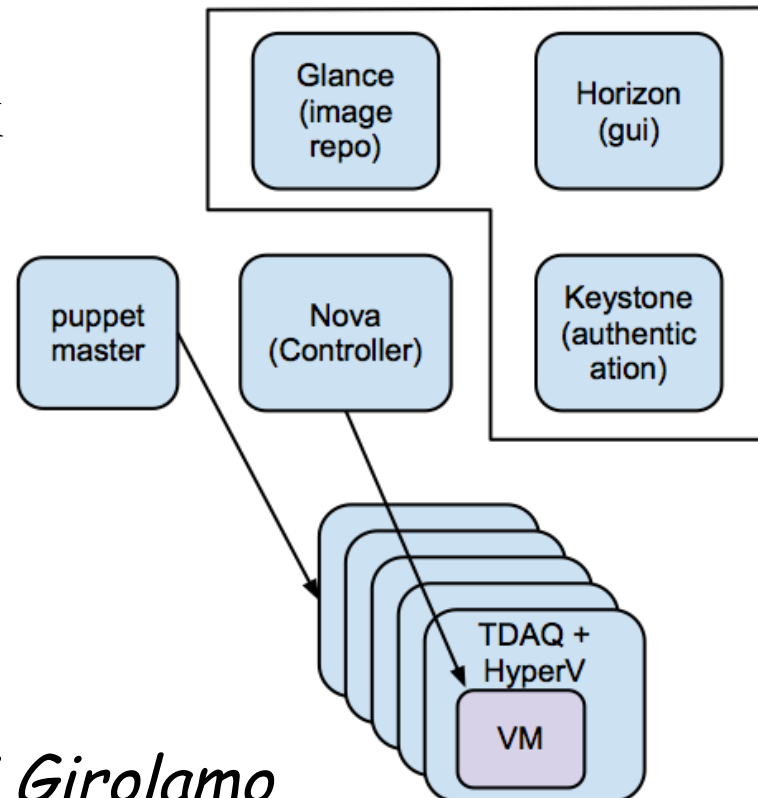
BIG PICTURE



Online farms for offline work

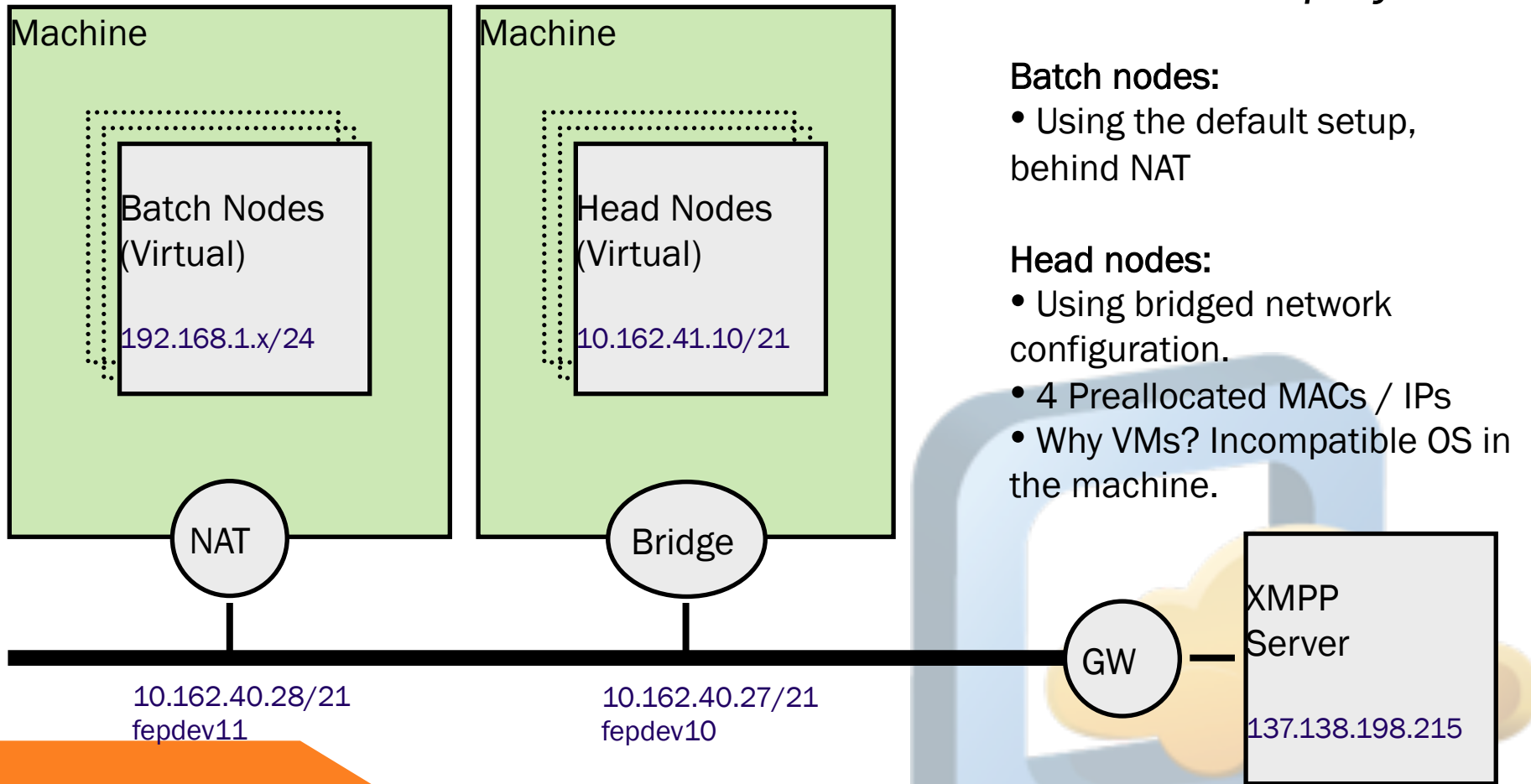
- During LS1 online farms will be mostly idle – e.g.
 - Cores: ATLAS: 14264; CMS: 13312; LHCb: >16000
- Develop a usage model working not only in LS1 but also during technical stops and even inter-fill pauses
 - Complete decoupling of online and offline environments
 - Fast reclaim of resources 
 - Easy internal scheduling (single resource requestor)
- Cloud interface for all experiments but LHCb
 - ALICE uses libvirt with CERNVM for image provisioning
 - ATLAS and CMS use OpenStack
 - Image provisioning: CernVM (ATLAS); BoxGrinder (CMS)
- Using CernVMFS and Xrootd (EOS) to access software and data
- Network configuration to the T0 needs attention 

- Setup a testbed
 - Few machines of same kind of the one in HLT farm
 - Similar network and VM/host configuration
 - Ideally SLC6 netbooted using the same image needed by the HLT sw
- Setup OpenStack



ATLAS, A. Di Girolamo

Testing deployment on ALICE HLT



Alice HLT Deployment

Batch nodes:

- Using the default setup, behind NAT

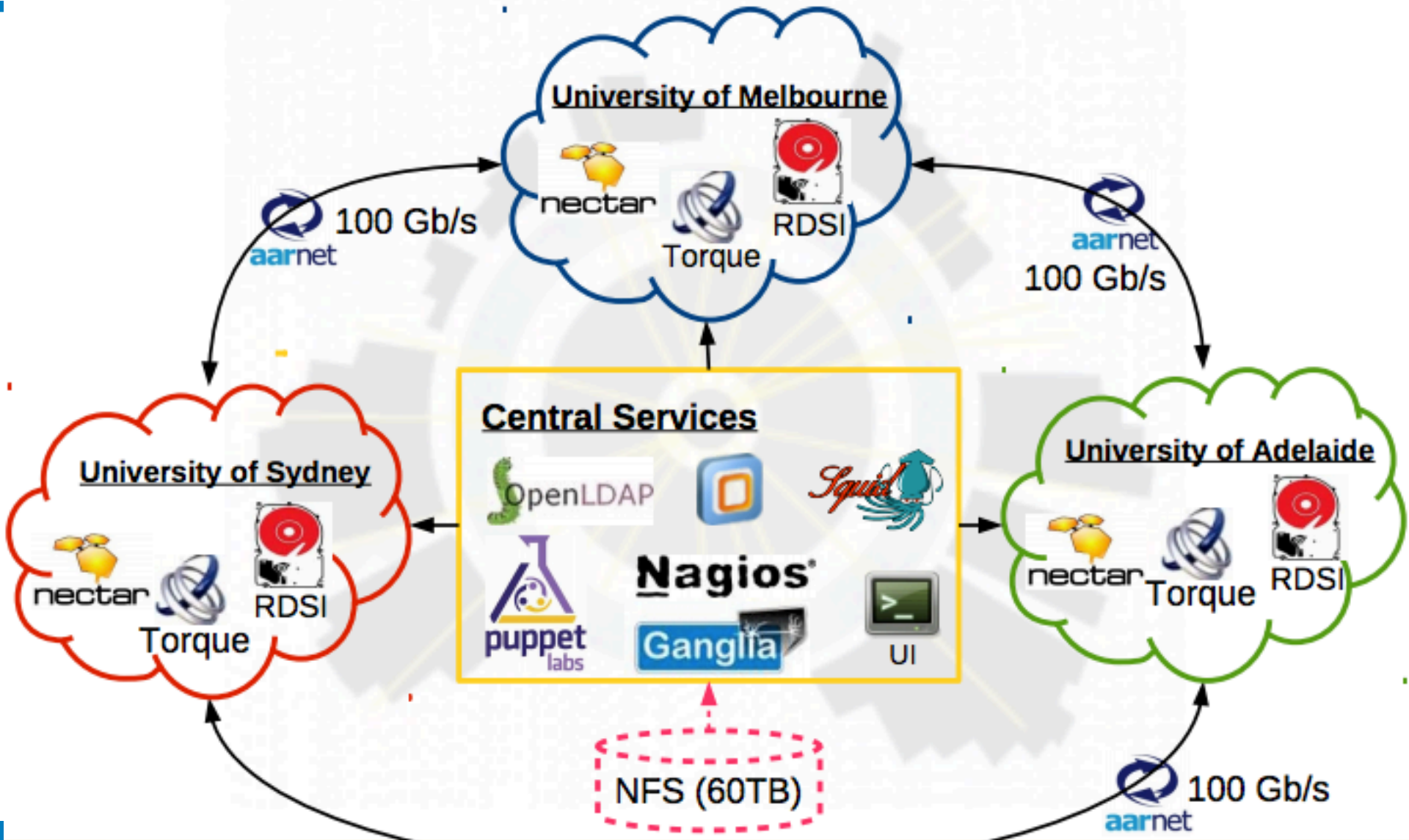
Head nodes:

- Using bridged network configuration.
- 4 Preallocated MACs / IPs
- Why VMs? Incompatible OS in the machine.

Overall experiments status





- All experiments getting ready to use clouds
 - Public (commercial) clouds (e.g. Amazon)
 - Possibility to absorb usage peaks
 - But still too expensive for our needs
 - Private clouds (e.g. CERN Agile Infrastructure)
 - Opportunistic use, e.g. Federated Tier-3
 - Be ready for an interface change at our sites
 - Dedicated clouds (e.g. online farms)
 - Efficient use of owned resources
- Building compatibility with heterogeneous systems via plugins in the resource allocation components
 - E.g. Condor-G (ATLAS, CMS), Alice's Cloud aggregator, LHCb's VMDIRAC
- Biggest open issue is data management to ensure scalability
 - Currently based on federated storage (Xrootd)





Going multicore

- Experiments software frameworks are being modified to be able to exploit multicore systems
 - Simple event-level parallelization almost in production
 - Benefits especially in terms of memory usage
 - Deeper parallelization still in the future (e.g. via Intel TBB)
- Fits naturally in a Cloud resource provisioning
 - It is anyhow possible to configure the VO job management framework to run more single-core jobs on the same Cloud-provided resource
 - It is also possible to adapt the current Grid-based infrastructure to run multicore jobs
- Single-multicore jobs do not have impact on the resource provisioning model

- Advantages:
 - HV OS under site's control; VM OS under VO control
 - Pause/resume/migrate? 
 - Data preservation projects based on virtualization
- Disadvantages:
 - Performance (especially in local disk access)? 
- Critical issues:
 - (i.e. may become a disadvantage if not done correctly)
 - Contextualization interface
 - While there is reasonable agreement on the interface (EC2 is a de-facto standard) there are several contextualization interfaces (amiconfig in CernVM, CloudInit most commonly used) 
 - May become a nightmare to maintain images and contextualization mechanisms that depend on the infrastructure 

Resource scheduling

- Not job scheduling!
- Inter-cloud scheduling
 - Experiments need to choose the cloud to which they want to ask for resources
 - Need to know the features offered by each infrastructure
- Intra-cloud scheduling
 - Not seen by experiment. A site business?
 - Often considered a “legacy” concept
 - True as far as you use the cloud approach to the end: “you pay for what you get”. In our environment we are still bound to sites that have to provide resources according to predefined shares and that have to respond to funding agencies for resources not 100% used
 - In my opinion we still need some intelligent scheduling (more than what is available in Nova or similar tools) but we need to be more coarse (e.g. longer allocation). Introduce the concept of lease? Economical model?





Ideal (?) scenario

- Start VM at a site if resources are available
- VM starts on a WN giving access to the whole WN
 - At least a fraction of the WN
- VM contextualised for the VO and starting the pilot agent
- Start pulling jobs from the CTQ
 - Depending on the WN configuration and jobs in the queue
 - * # cores, hyper threading, memory...
 - ☆ Run multiple jobs in parallel
 - ☆ Run parallel jobs
 - ☆ Up to the VO to optimise the WN usage
- Communication with the site
 - How long can I still run?
 - ☆ Allows the site to shut down a VM
 - ☆ Allows fairness in case running over pledge
 - VO should commit to not match new jobs if requested to stop
 - ☆ Should not be killed within a grace period (one day?)
 - If no matching jobs (no jobs or not allowed)
 - ☆ VM could either shutdown or sleep for a while, check again...



Starting/stopping a VM

- **Starting**
 - When there are tasks to execute
 - ☆ VO responsibility
 - When there are slots available
 - ☆ Site responsibility to get slots free if running under pledge
- **Stopping**
 - If there are no tasks to execute in the central task queue
 - ☆ VO responsibility
 - If the VO is running over pledge and the site is saturated
 - ☆ Site responsibility to ask the VO to stop some VMs
 - * Advanced warning?
 - * Not different from setting a queue max duration at some point
 - ☆ VO should commit to stop matching tasks and shutdown VMs

Accounting

- Very much related to scheduling and VM duration
 - Longer allocation times to optimize resource usage!
- Should charge unused time of unreleased resources
 - WCT is the obvious choice
 - In commercial clouds infrastructure provide *machine types* and account for the minimum guaranteed power provided
 - We are used to a finer grained accounting
- Accounting would be the basis of an economical model for resource scheduling
 - Cost changes with demand – auction model
 - In this model the site is NOT responsible for 100% usage of resources
 - Will funding agencies accept this?




- Site/cloud authentication and authorization only at the resource allocation level
 - Security at the job level is completely in VO space
 - No need to use glexec or anything like that
 - Site loses any fine grained (end user) control
- Access to proprietary clouds is simple...
- Access to public clouds is by ‘bank account’
- Access to private clouds requires agreement on the mechanisms
 - Identity federations
 - Mapping to *budget codes*
- Trust on VM images is an open issue
 - HEPiX developed a model. Is that accepted? No experience



- The Cloud Storage in the common meaning is not what is needed by HEP
- All Cloud tests used the Federated storage approach (remote data access) to process data
- Valid approach for a proof of concept but we need to address the need for efficient storage access
 - Proper use of contextualization may allow to get effective access to the local storage. Is that enough?
 - In the best case we may get what we already have...
- Cloud storage may be interesting for user data (the so called *home file system*)



Support of small VOs

- Once the Cloud model will be in production for LHC experiments, sites may provide them with a “private cloud” interface
- Current model may still be needed in order to support small VOs that may not be able to transition to a Cloud resource allocation model
- Coexistence of Grid and Cloud will be needed at the same sites 

Summary of “pitfalls”

- Lack of a uniform contextualization interface may limit the possibility to access resources
- Attitude to oversimplify intra-cloud scheduling may bring to non optimal usage of sites
- “Coarse” resource allocation (longer times) makes difficult to reach 100% usage of sites
- Loss of fine grained access control at sites
- Decreased data access efficiency as a result of more flexibility

Summary of work opportunities

- Automatic resource provisioning for experiments
- Adapt experiment frameworks to fast reclaim of resources
 - Pause/resume/migrate
- Coexistence model of Cloud and Grid interfaces
- Reduce inefficiencies due to virtualization
- Standardize contextualization interfaces
- Optimize performances in local storage access
- Optimize network usage in network critical environments
- Infrastructure for intra-cloud scheduling
 - E.g. standard definition of machine types and costs
- Improve intra-cloud scheduling (economical model?)
- Accounting on clouds
- Identity federations
- Cloud storage (e.g. user home directories on cloud)

Conclusions

- Clouds are an opportunity because address our (HEP) use case with industry-standard middleware
 - Independent of specific funding e.g. by EC
- Experiments getting ready to use Clouds
- Use of Cloud interfaces still not as efficient as the current (Grid) access model
- Major work needed in storage access optimization
- Work needed in accounting, federated identity management, intra- and inter-cloud scheduling