BELLEII Cloud Toy:
a suite of tools for a simplified deployment of OpenNebula based Cloud Infrastructures matching BELLEII VMDIRAC

Antonio Amoroso, Flavio Astorino, Fabrizio Bianchi, Marco Destefanis,
Marco Maggiora, Jacopo Pellegrino

University of Turin and INFN Turin

Fifth Belle II Italian Collaboration Meeting
Padua, May. 31st 2016

VMDIRAC for BELLEII

- **Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan**

1. VMDIRAC and BelleDIRAC: development to provide an environment for the cloud
2. sophisticate approach to the use of commercial clouds (Amazon AWS)
3. makes use of:
    1. CVMFS (Stratum 0 @ CERN and different Stratum 1 elsewhere)
    2. squid server on commercial cloud for site caching
    3. VMDIRAC for VMs instantiation by JobAgents
    4. OpenStack as Clound Infrastructure (CI) manager

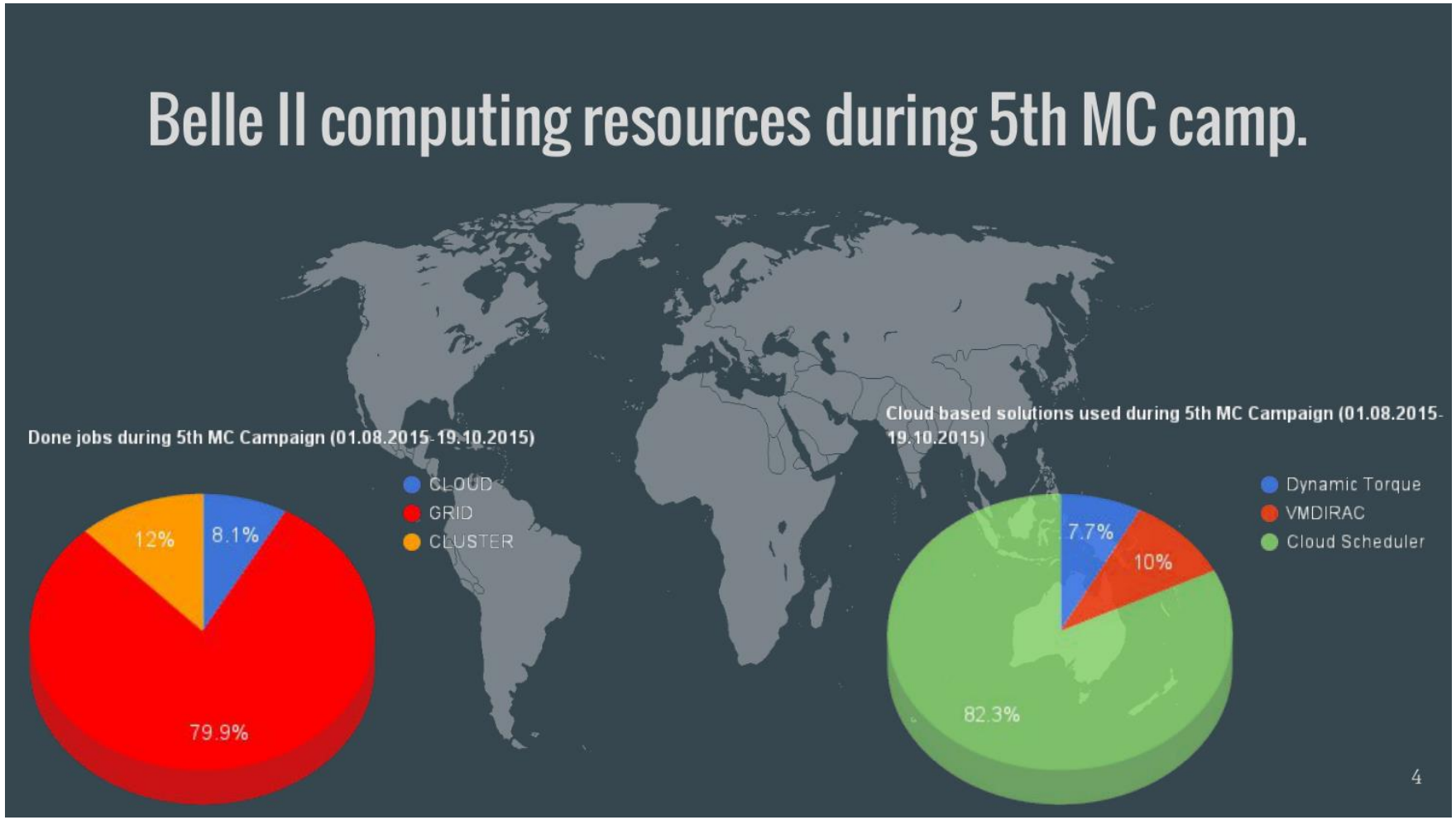
VMDIRAC for BELLEII

- **Rafal Grzymkowski:**
  VMDIRAC and BelleDIRAC development to provide an environment for the commercial clouds

Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan

VMDIRAC for BELLEII

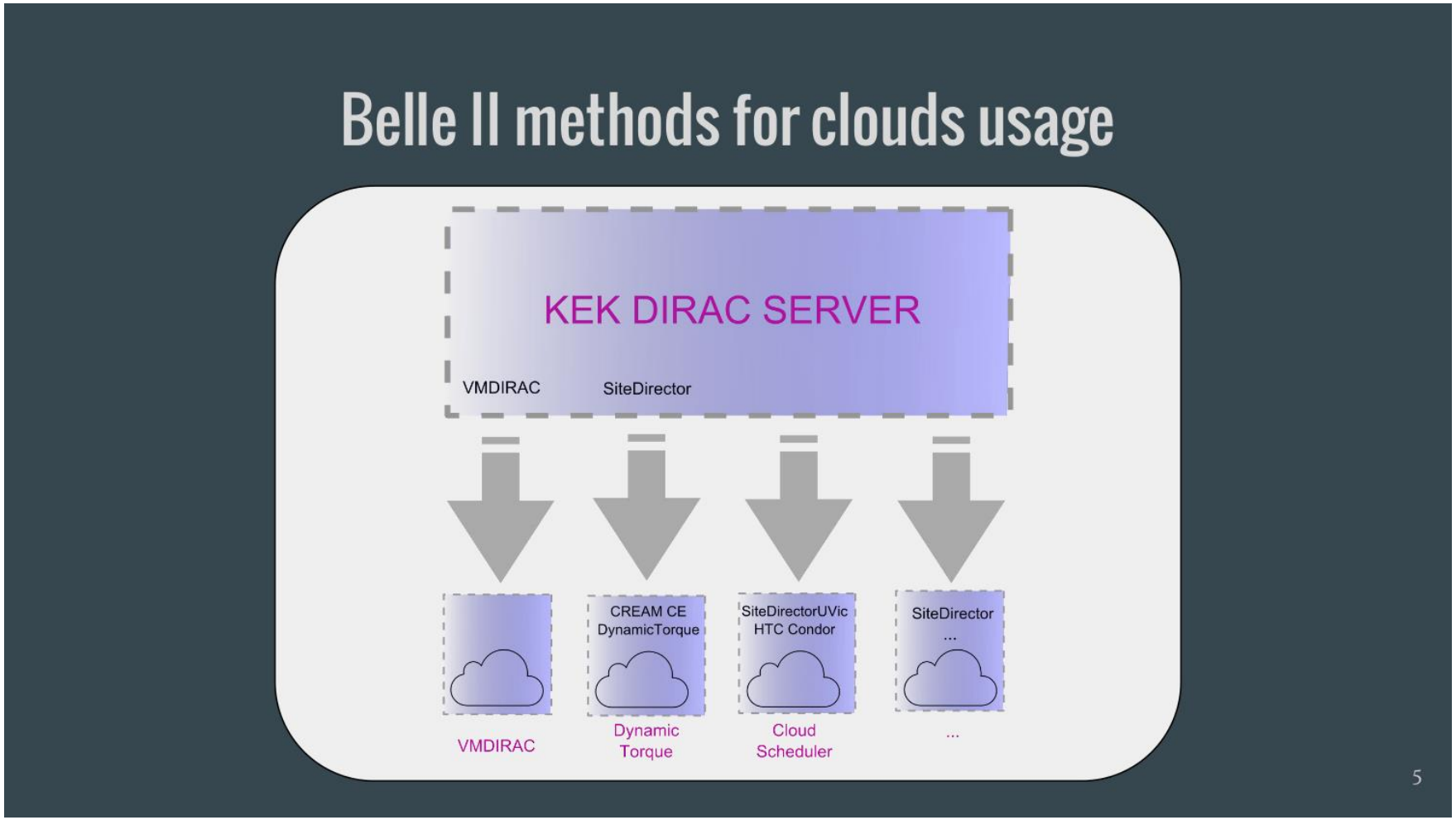
- **already tested:**
  during 5th MC camp



Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan

VMDIRAC for BELLEII

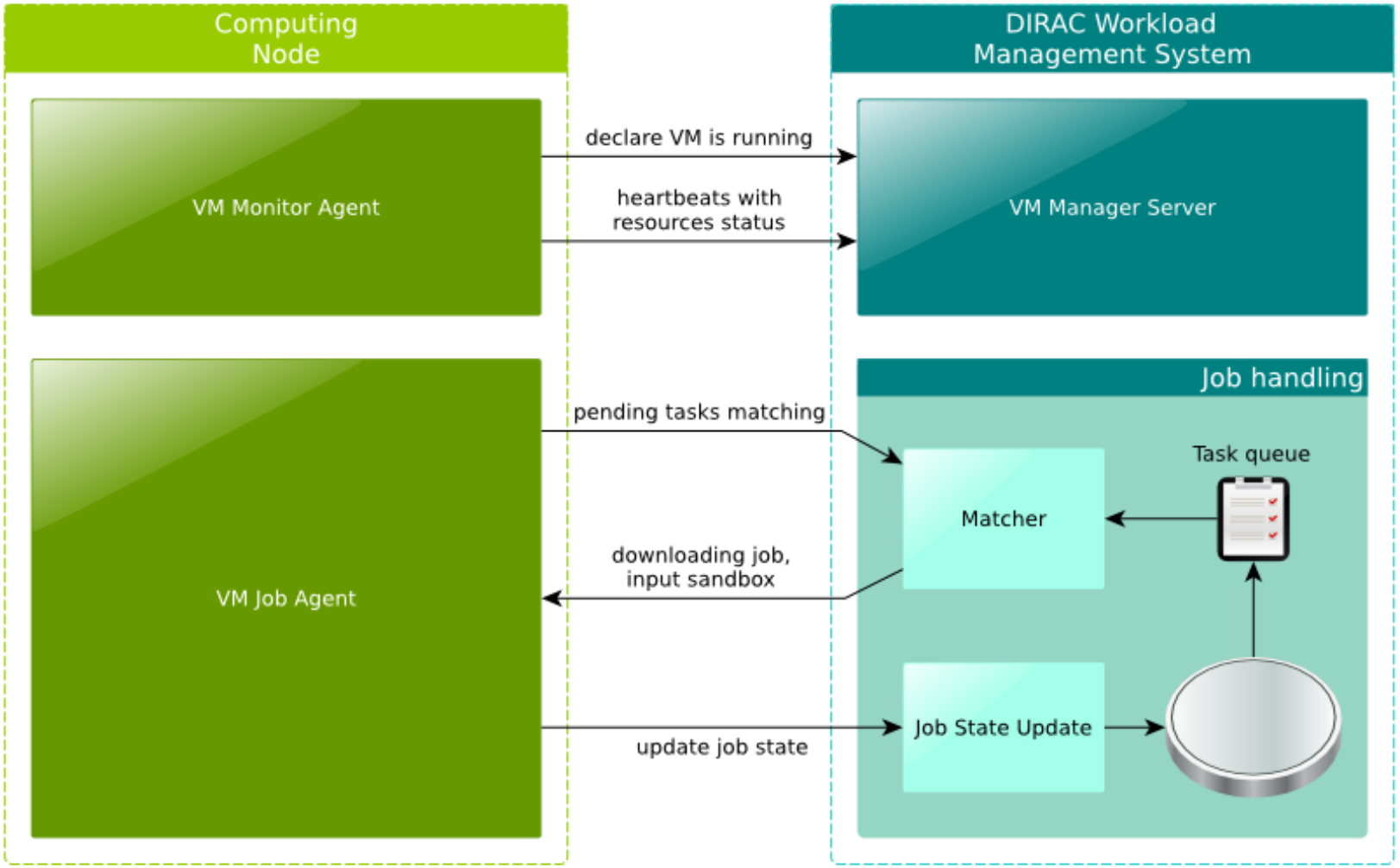
- **one option for KEK:**
  one of the possible approaches to cloud resources for KEK DIRAC server



Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan

VMDIRAC for BELLEII

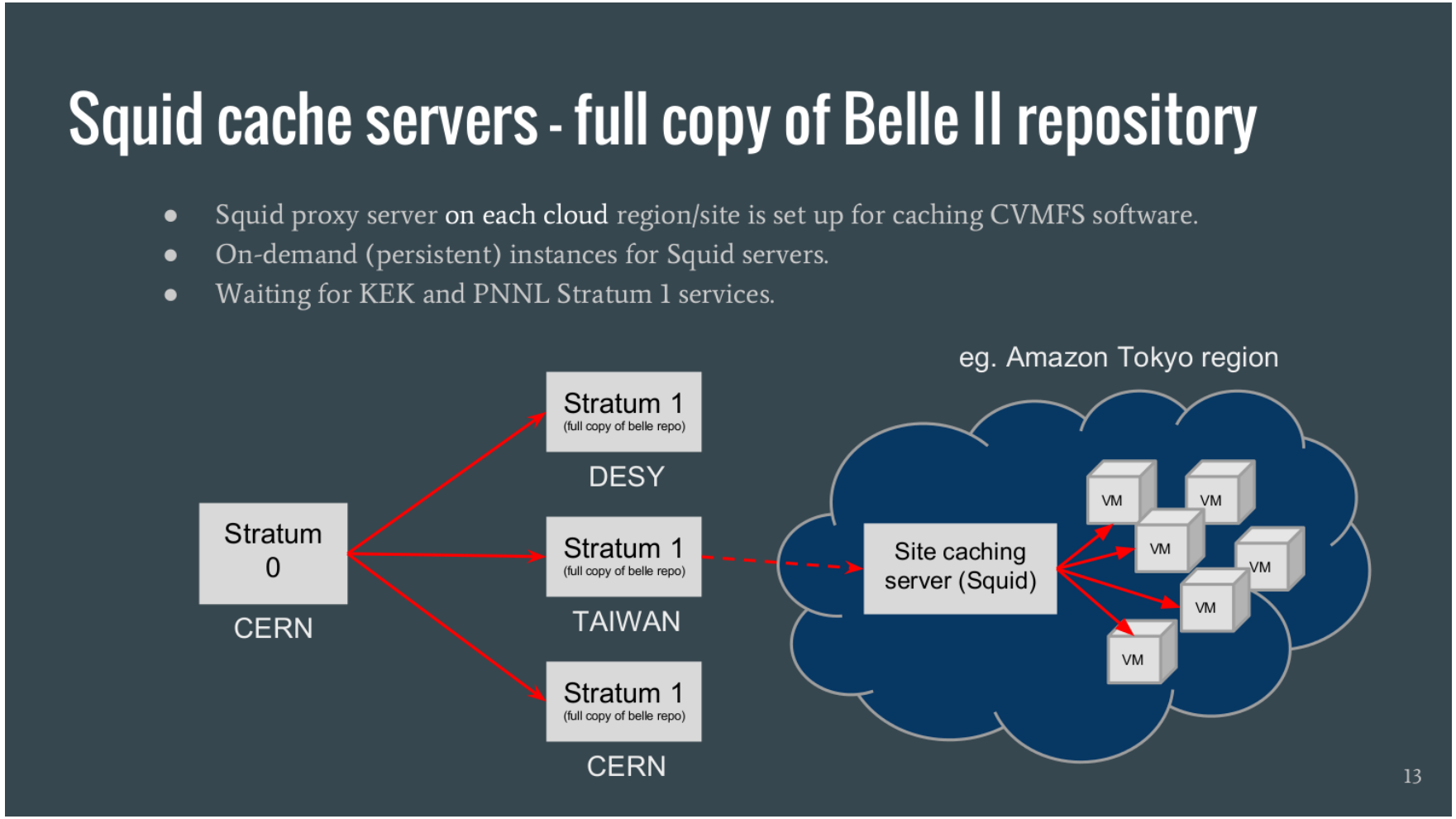
- **VMDIRAC:**
  Agents rule the remote Computing Node



Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan

VMDIRAC for BELLEII

- **Squid:**
  provides local caching
  on cloud infrastructure



Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan

An alternate approach…

And what if…

instead of exploiting commercial (i.e. no control or LTS!) at high prices

we make life easier for those academic sites

who wants to deploy CI able to cope with

VMDIRAC & BELLEII-DIRAC?

VMDIRAC for BELLEII

- **Rafal Grzymkowski, 22nd B2GM 20/10/2015, KEK, Japan**

  1. VMDIRAC and BelleDIRAC: development to provide an environment for the cloud
  2. sophisticate approach to the use of commercial clouds (Amazon AWS)
  3. makes use of:
     1. CVMFS (Stratum 0 @ CERN and different Stratum 1 elsewhere)
     2. squid server on commercial cloud for site caching
     3. VMDIRAC for VMs instantiation by JobAgents
     4. OpenStack as Clound Infrastructure (CI) manager

- **why choosing to use commercial clouds?**
  1. lack of computational resources → effective elastic multi-tenant cloud funded by more stakeholders
  2. lack of dedicated manpower → centralized CI management by VMDIRAC
  3. lack of cloud computing skills → automatic deployment of a CI optimized for BELLEII SW
  4. heavy maintenance and migration load → “market” of VM images and templates optimized for BELLEII SW

An alternate approach…

And what if…

instead of exploiting commercial (i.e. no control or LTS!) at high prices

we make life easier for those academic sites

who wants to deploy CI able to cope with

VMDIRAC & BELLEII-DIRAC?

Ingredients:

- OpenNebula
- rOCCI
- kickstart
- squid

B2CT:
BELLEII Cloud Toy v0.1

Easy Cloud Infrastructure Setup

Goal: deploy an OpenNebula hypervisor
minimizing the user interaction
during the installation process

Mean: server installation via usb key

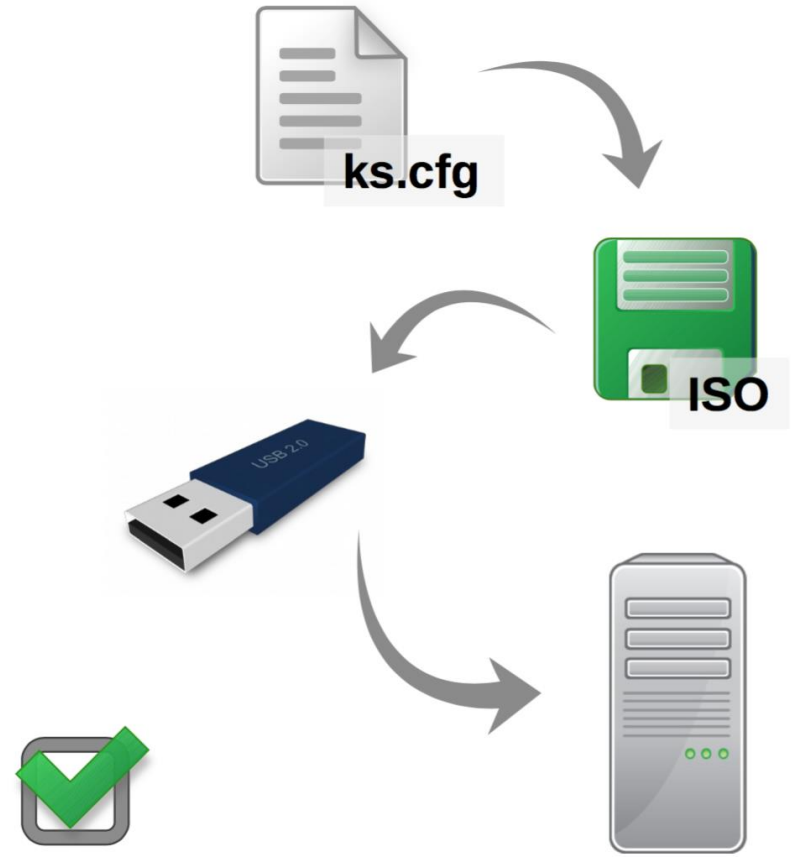
B2CT tested on:

Machine	Dell Server
CPU	2 x Intel(R) Xeon(R) E5-2650 v3 @ 2.30GHz
Cores	20 physical 40 hyper treading
RAM (GB)	160 GB

B2CT: a step by step receipt

Can/must be perfomed at every major upgrade of OpenNebula/rOCCI/squid or BELLEII VMDIRAC

- create kickstart file
- prepare customized ISO
- make bootable usb key
- install on server
- check everything works



B2CT: Kickstart.cfg

A kickstart file contains the information needed to perform the installation in order to **avoid the user providing them.**

Some parameters may be left to user input
(network parameters, keyboard layout, …).

It is possible to specify **packages or additional software** to be installed.

Software installed via kickstart:
OpenNebula, squid proxy, and rOCCI

B2CT: customized ISO

Starting from a **kickstart** and a **standard iso** it is possible to modify the iso so that it will look for the given kickstart at installation time.

Standard iso adopted for B2CT:

CentOS-6.7-x86_64-netinstall.iso

Care the **location of the kickstart**:

better to address the usb drive via its “label”.

B2CT: bootable Usb Key

Format (FAT32, mbr) a usb drive (2GB is enough).

Label must correspond to the **name** indicated in the customized iso creation process. Otherwise the kickstart will not be found when the installation begins.

Make the usb bootable:

- either via command line
- or using an application such as Unetbootin

B2CT: installation on Server

Start or reboot the machine and plug in the bootable usb drive.

From the BIOS menu choose the ***boot from usb drive*** option.

The installation will ask the user to provide the parameters not specified within the kickstart file, then the installation will proceed autonomously.

For instance, the user can provide:

- if the host will be a server providing services to the CI
- or will be one of the machines "only" hosting VMs for the worker nodes
- the network parameters

The machine will reboot when the installation is over.
Remove the usb drive and let the machine boot normally.

B2CT: test the Installation

Once the machine is up and running, users may control that the parameters have been properly set (network interface is up, storage is mounted, datastores interfaced …)

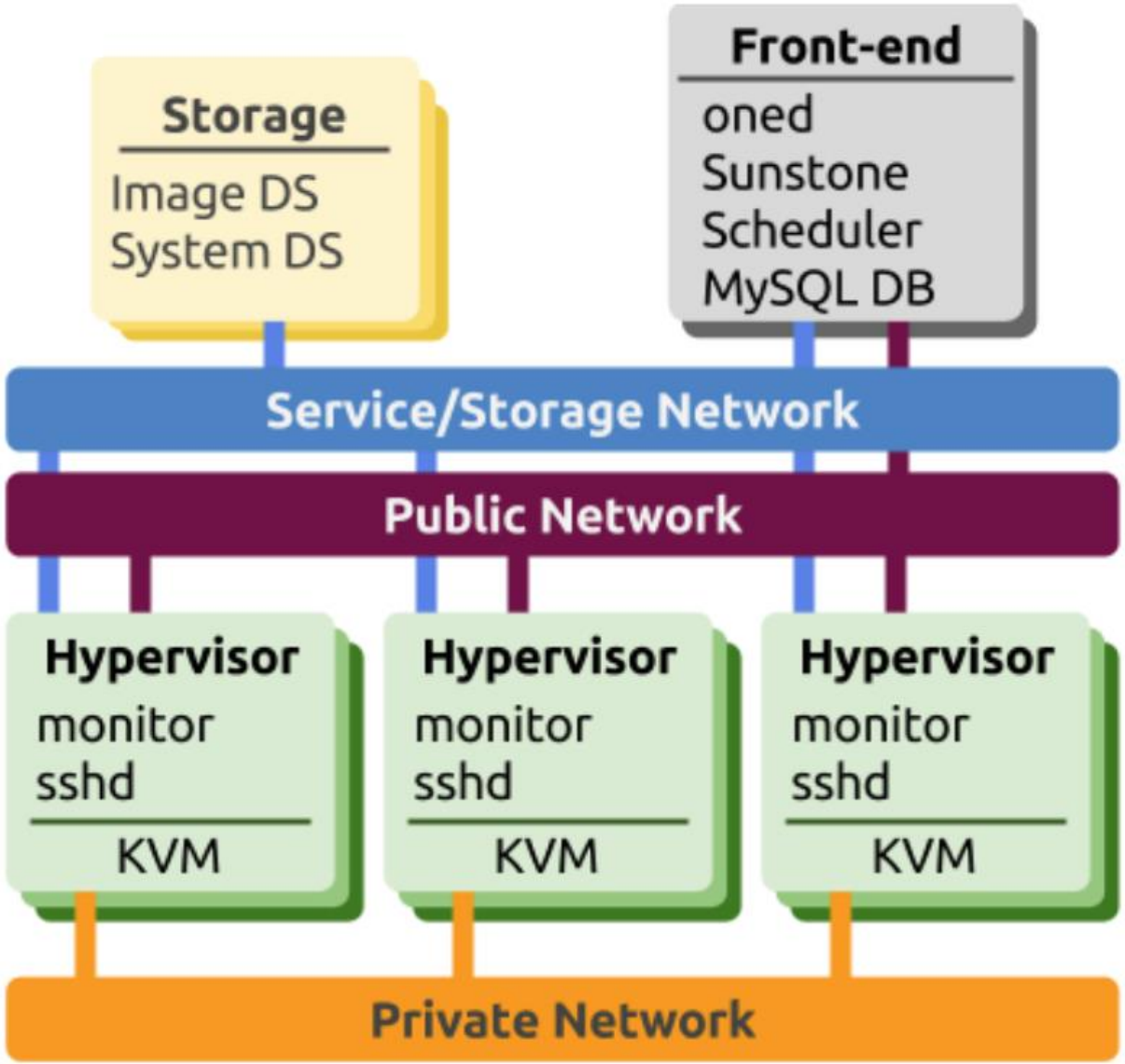
The entire installation process requires about 1 hour
(on the hardware used for the test).

The Sunstone hypervisor interface allows to verify that OpenNebula is working fine.

Validation proceeds creating virtual machines and:

- running jobs locally
- submitting jobs via VMDIRAC

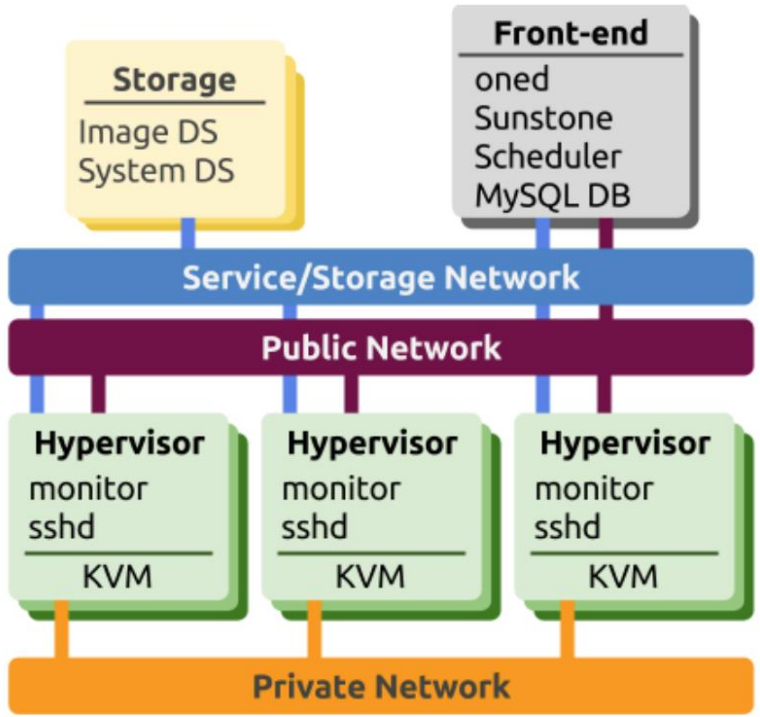
OpenNebula: Open Cloud Reference Architecture



OpenNebula: physical hosts

Servers that will host the Virtual Machines:

- often called “Hypervisors” (like the software)
- KVM
  (OpenNebula supports also vCenter and Xen)
- monitoring daemons
- sshd for system connection



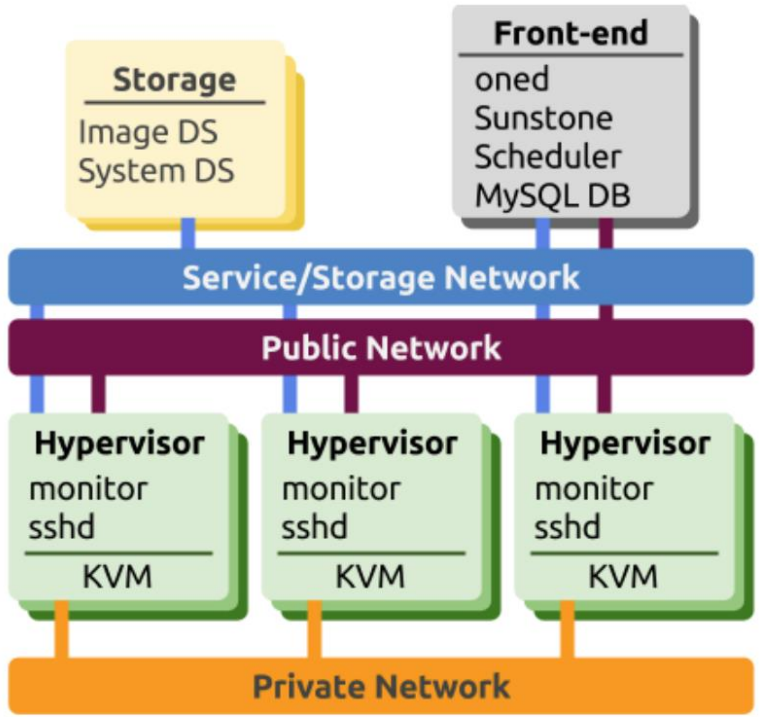
OpenNebula: networks

Used by OpenNebula and the infrastructure:

- Service and Storage network:
  - monitoring and control information
  - image transfers

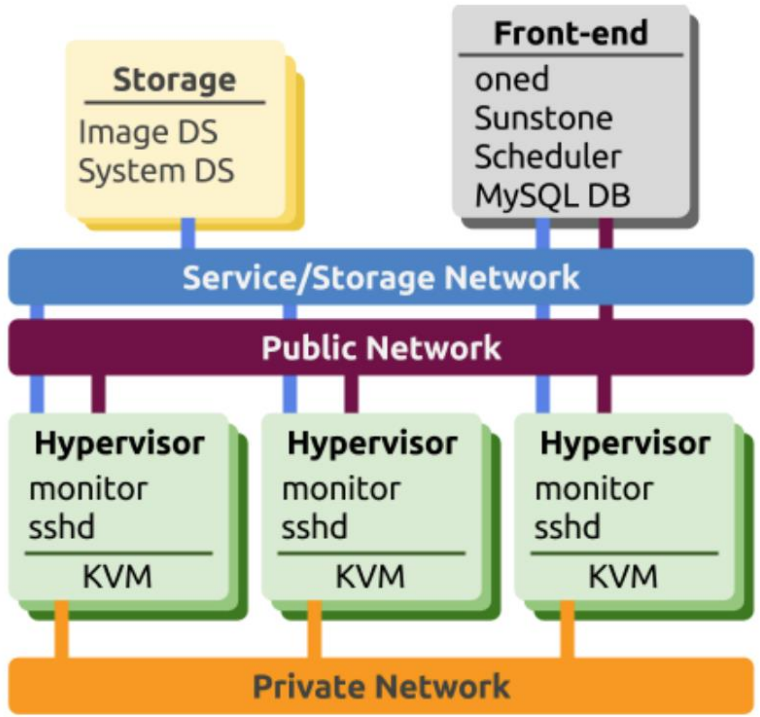
Used by the Virtual Machines:

- Private Network:
  - private IPs
  - intra-cloud communications
- Public Network:
  - public IPs
  - incoming connectivity to VMs



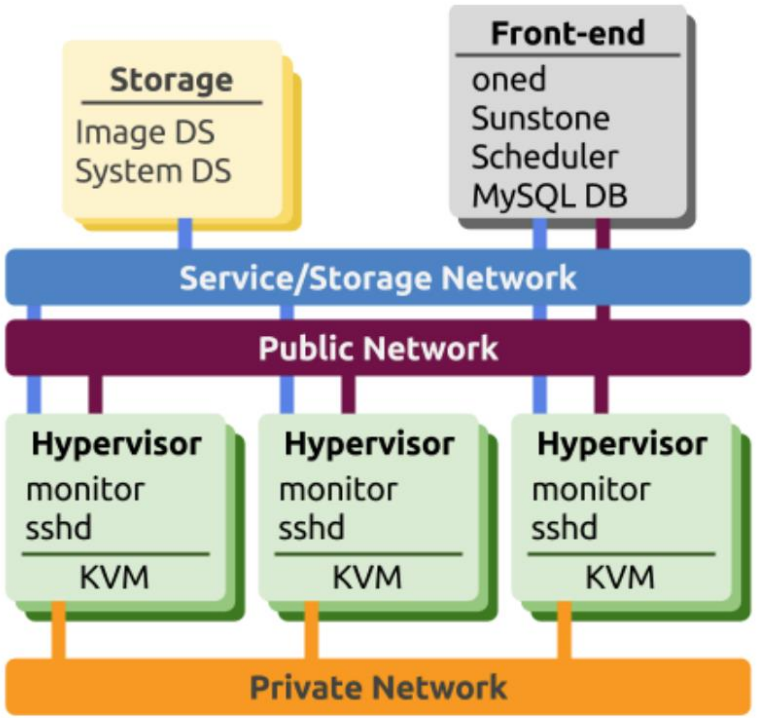
OpenNebula: storage

- Service datastores don't necessarily need to be shared across VMs:
  - images can be transferred to the hypervisors' disk through ssh and started locally
- Image Repository Datastore:
  - holds the OS images
- System Datastore
  - holds the running instances
  - if it's a shared FS, VMs can be "live-migrated"



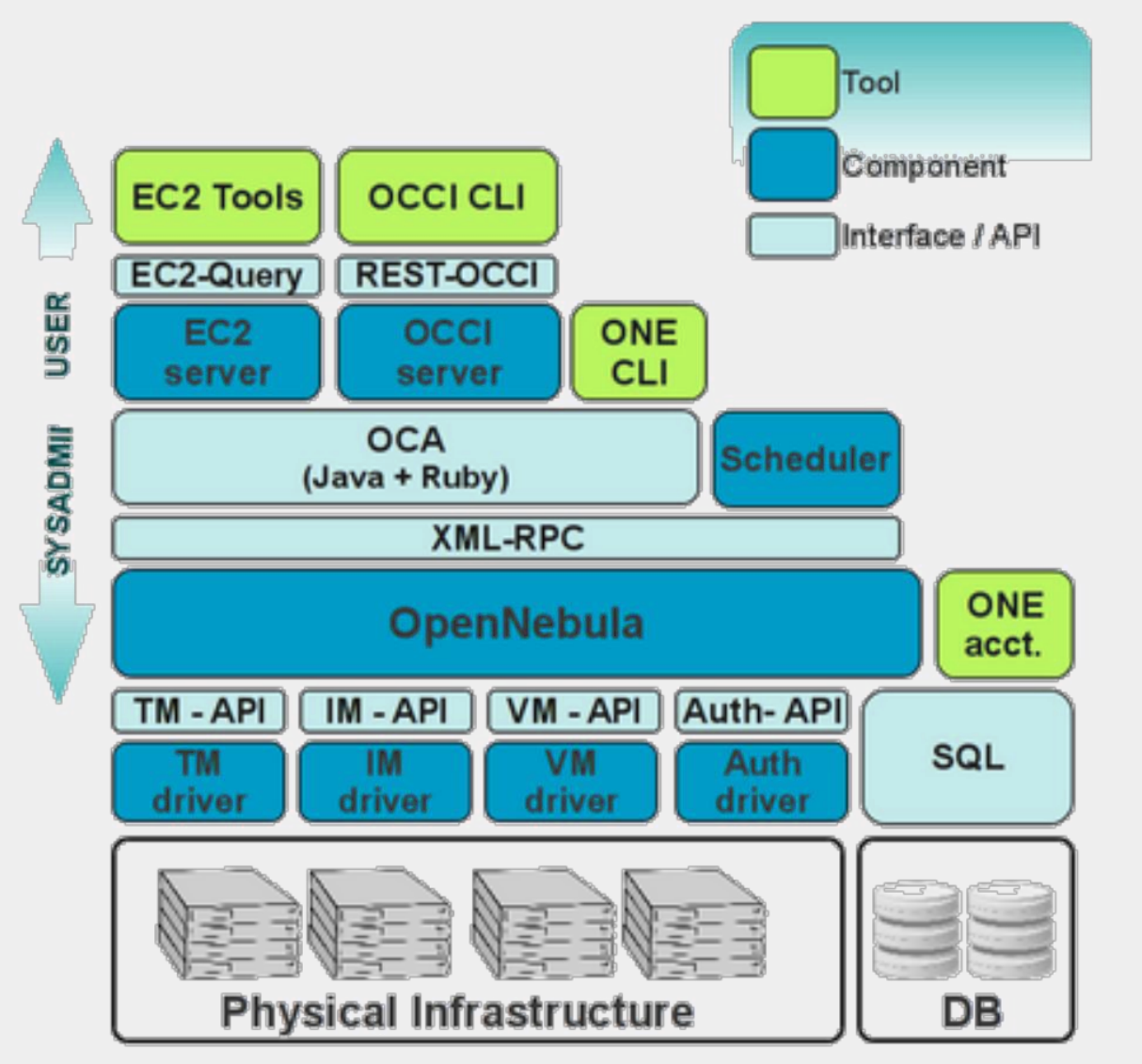
OpenNebula: the control node

- Runs the OpenNebula stack:
  - oned (the main daemon)
  - schedd (the VM scheduler)
  - Sunstone (the web-based GUI)
  - MySQL DB backend (can be separate)
  - API services (OCCI or EC2)
  - advanced services (OneFlow, OneGate,…)



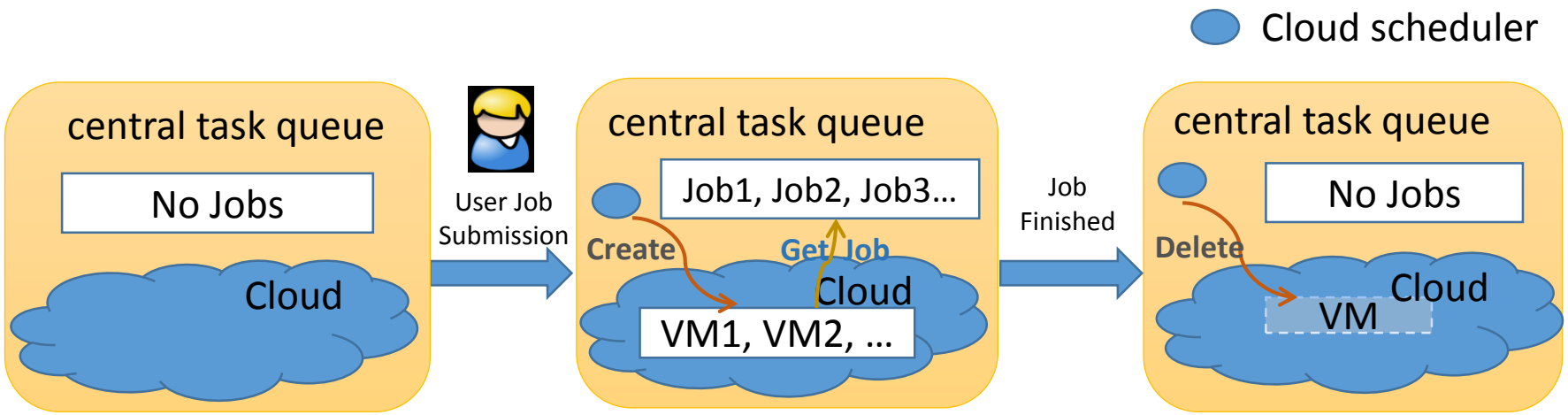
  - control node unavailability does not affect running VMs
  - only control on them (start & stop, monitoring,…)

OpenNebula: internal architecture



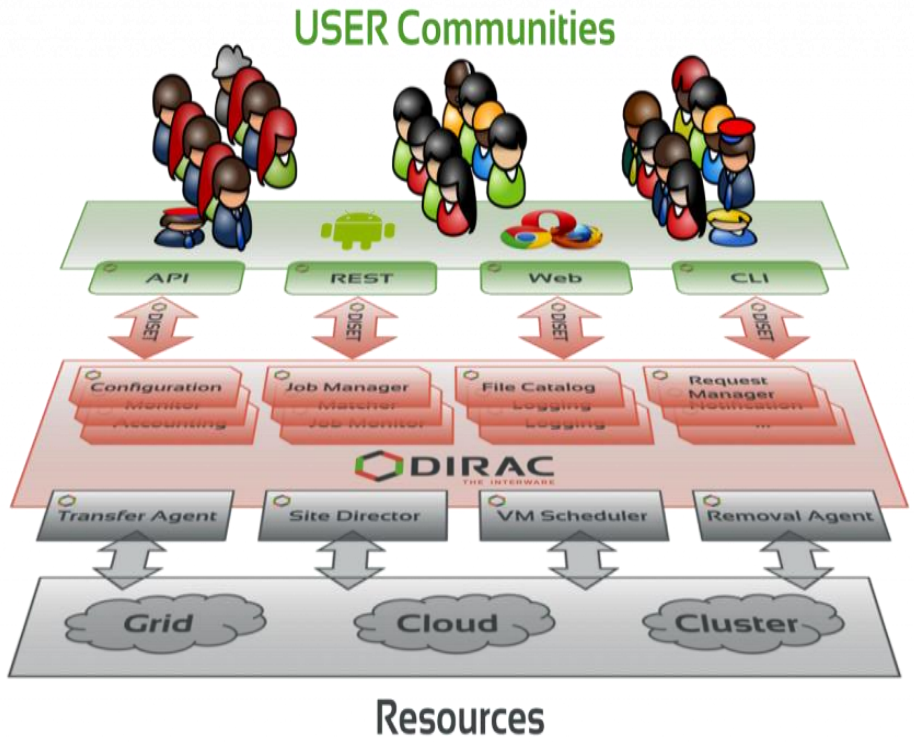
Elastic cloud

- On-demand usage
  - Elastic way to use cloud
  - Don’t occupy resources before jobs are coming
    - Save money when you use commercial cloud
  - VMDIRAC is one of the way allowing to use clouds elastically
    - HTCondor + Cloud scheduler, elastiq
  - Need central task queue and cloud scheduler



DIRAC

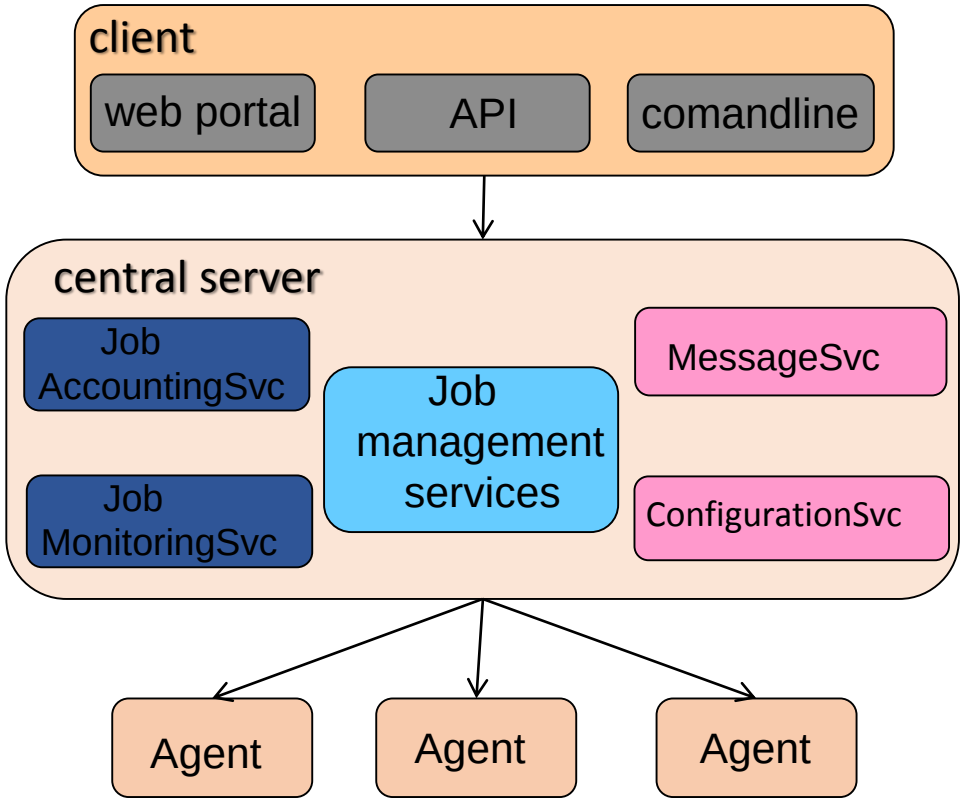
- DIRAC allows to interconnect computing resources of different types as a interware
  - Grid
  - Standalone Cluster
  - Desktop grid
  - Cloud



DIRAC systems

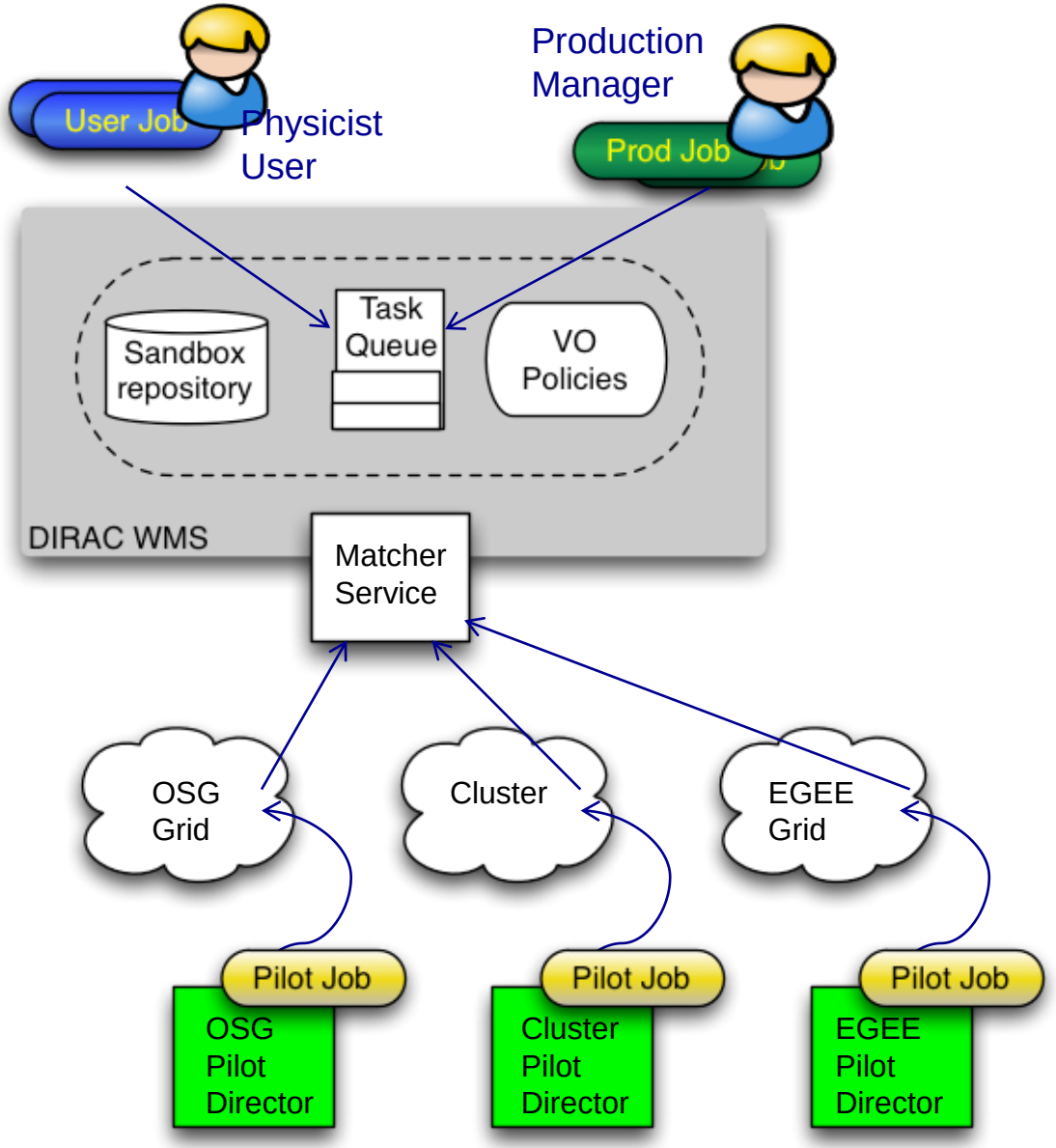
- VMDIRAC is one of DIRAC systems
  - Workload management, Data management….

- Each system consist of similar components:
  - services: passive components, permanently running, waiting for queries or requests
  - agents: light and active components which run as independent processes to fulfill one or several system functions
  - clients
  - databases



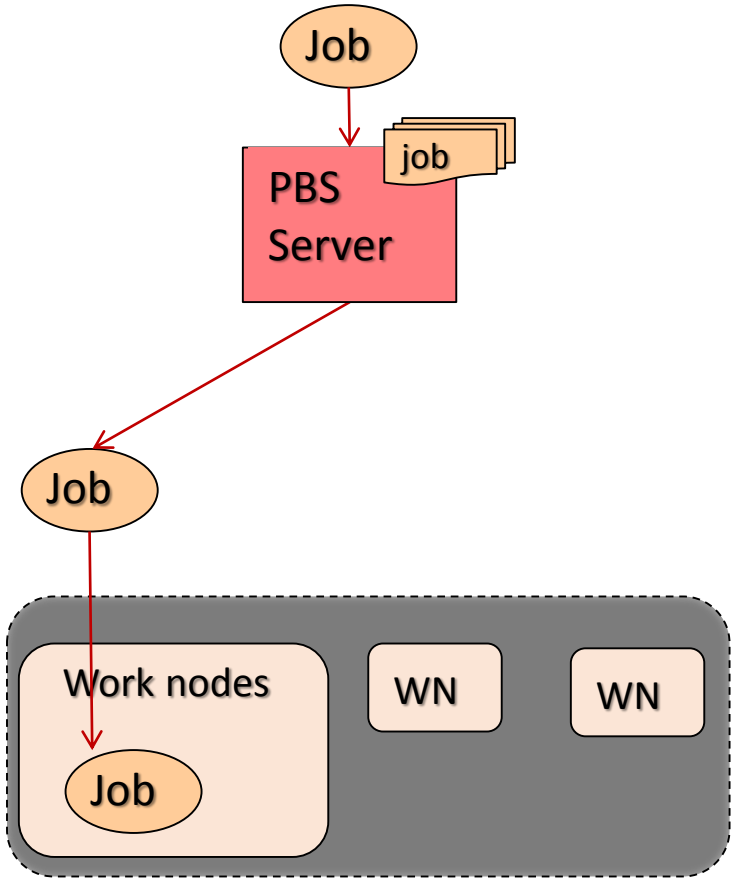
DIRAC workload management

- DIRAC is like a big cluster system over WAN
- Central task queue
  - User jobs are put into the task Queue
  - Job priorities are controlled with VO policies
- Pilot director
  - Connect with resource broker and submit proper pilots
  - Deal with heterogeneous resources
    - Every resource type need a pilot director
- Match service
  - Cooperate with pilot, Match proper user jobs to resources



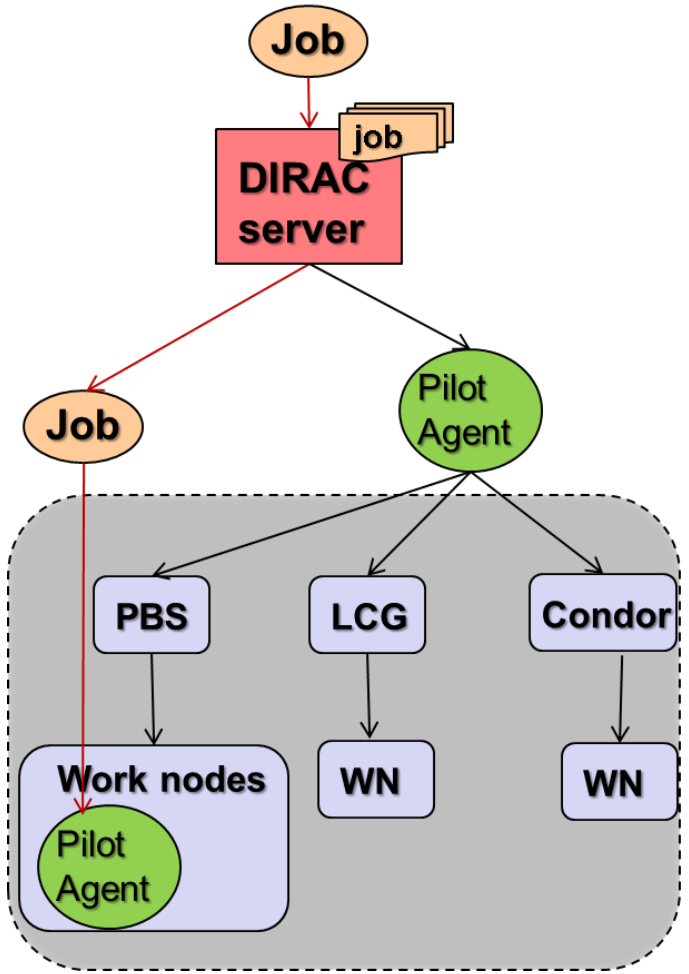
DIRAC push scheduling

- Two common ways to schedule jobs to resources
  - Push scheduling
  - Pull scheduling
- Push scheduling on clusters
  - User jobs is submitted to the local scheduler
  - Jobs are put into queues
  - Be arranged to WNs directly



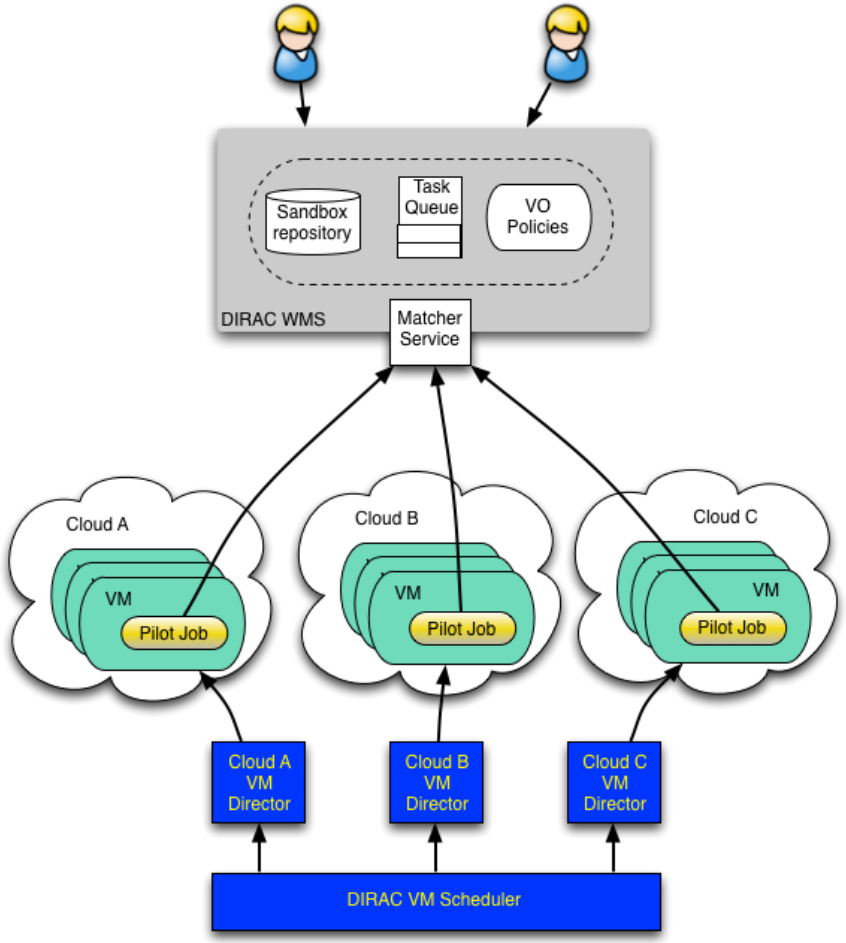
DIRAC pull scheduling

- Pull scheduling with pilot paradigm on DIRAC
  - Instead of send use jobs to resources directly
  - Pilot jobs are sent to resource brokers (CE, PBS…) as normal jobs
  - Pilot jobs start job agents
  - Job agents do
    - occupy a resource
    - set up environment
    - pull jobs from central queue
- Advantages
  - Avoid failure of user jobs because of hardware problem
  - Easy to fit in different resource environment



Cloud integration with DIRAC

- "VM director" instead of "Pilot director"
  - start VMs, instead of submitting pilot jobs
- VMs at boot time start "pilot job"
  - This makes the instantiated VMs behave just as other WNs with respect to the DIRAC WMS
- VM scheduler need to manage dynamic virtual machines according to job situation



Cloud integration with DIRAC

- Integrate Federated cloud into DIRAC
  - OCCI compliant clouds:
    - OpenStack, OpenNebula
  - CloudStack
  - Amazon EC2

- Main functions
  - Check Task queue and start VMs
  - Contextualize VMs to be WNs to the DIRAC WMS
  - Pull jobs from central task queue
  - Centrally monitor VM status
  - Automatically shutdown VMs when no jobs need

VMDIRAC: architecture and components

- Dirac server side
  - VM Scheduler – get job status from TQ and match it with the proper cloud site, submit requests of VMs to Director
  - VM Manager – take statistics of VM status and decide if need new VMs
  - VM Director – connect with cloud manager to start VMs
  - Image context manager – contextualize VMs to be WNs
- VM side
  - VM monitor Agent – periodically monitor the status of the VM and shutdown VMs when no need
  - Job Agent – just like “pilot jobs”, pulling jobs from task queue
- Configuration
  - Use to configure the cloud joined and the image
- Work together
  - Start VMs
  - Run jobs on VMs

B2CT: virtual Machines

OS: SL5, SL6, Ubuntu 14

Test on local installation

Strict requirements on storage space

- the VM image has to be copied over the net each time a new VM is instantiated
- it could be a bottleneck depending on the requests

QCOW2 image format

- dynamic increase of the storage

Minimal OS installation

- only the required software is installed

B2CT: BELLEII Software

Access to the BELLE2 software

- install while preparing the VM — huge space required (28GB)
- install during the contextualization — huge space (28GB) and time (2h)
- mount the sw via CVMFS — less space required (3.7GB)

The CVMFS downloads only the needed files

Software download time

- normally it has to be downloaded many times
- squid proxy allows to download the files once for better performances

B2CT: instantiating BELLEII VMs

Copy the VM image to the server

Create a new image in OpenNebula

- a file .one will be provided
- the proper path have to be inserted

Create a new template in **OpenNebula**

- a file .txt will be provided
- the number of the image and the network have to be inserted

Instantiate a new VM

The machine is ready in less than 1 minute

The new VM can be used

Test performed with build-2016-04-10 sw version

B2CT and VMDIRAC: a (near) future?

- VMDIRAC deploys the “VM director” to a B2CT CI
- a repository provides images and templates of the VMs for the B2CT CI
- a proper VMs in instantiated if necessary and a pilot job is executed, or…
- an available VM receives from OpenNebula a pilot job
- any VM at instantiation contextualizes to BELLEII via CVMFS
- a squid server provides a fast local cashing during CVMFS contextualization
- the pilot job pulls jobs from the stack
- any unused VM is killed freeing resources

B2CT and VMDIRAC: advantages

Little and medium size sites:

- easier access to cloud technologies and reduced manpower for CI deployment
- reduced manpower for Hypervisor and server upgrades
- easy retrieval of images and templates optimized for BELLEII SW
- no manpower for BELLEII SW updates

KEK and BELLEII SW management:

- standardization of the sites participating to the BELLEII Distributed Computing (DSC)
- VM-pilots can be executed by a centralized VMDIRAC console at KEK:

  less manpower, larger control
- localized knowledge can provide images and templates to the whole BELLEII DSC