Cloudscapes in Scientific Distributed Computing

D. Salomoni, INFN-CNAF
Davide.Salomoni@cnaf.infn.it

VII Seminario sul Software per la Fisica Nucleare, Subnucleare e Applicata – Alghero, 31/5 - 4/6/2010

Outline

1. Clouds and Grids
2. Resources
3. Integration of access interfaces to resources
4. Summary

Outline

1 Clouds and Grids

2 Resources

3 Integration of access interfaces to resources

4 Summary

Fashionable IT

Buzzwords of the day

- Clouds
- Taxonomy
- Virtualization
- Green Computing
- PaaS, SaaS, IaaS, ?aaS

The compulsory slide on definitions: Grids vs. Clouds

*The essence of the [definition] can be captured in a simple checklist, according to which a **Grid** is a system that:*

1. *coordinates resources that are not subject to centralized control. . .*
2. *. . . using standard, open, general-purpose protocols and interfaces. . .*
3. *. . . to deliver nontrivial qualities of service.*

(I. Foster, What is the Grid? A Three Point Checklist, 2002)

***Cloud computing** is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. (NIST Working Definition of Cloud Computing.)*

The compulsory slide on definitions: Grids vs. Clouds

*The essence of the [definition] can be captured in a simple checklist, according to which a **Grid** is a system that:*

1. *coordinates resources that are not subject to centralized control. . .*
2. *. . . using standard, open, general-purpose protocols and interfaces. . .*
3. *. . . to de*

(I. Foster, W *02)*

Exercise

Spot the difference between the two definitions.

***Cloud computing** is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. (NIST Working Definition of Cloud Computing.)*

In practice...

Distributed Computing Infrastructures (whatever they incarnation is) should:

- provide solutions for **resource discovery, usage, policing**
- honor contracted **Service Level Agreements**
- ensure proper **security enforcement** measures (authentication, authorization) are taken.

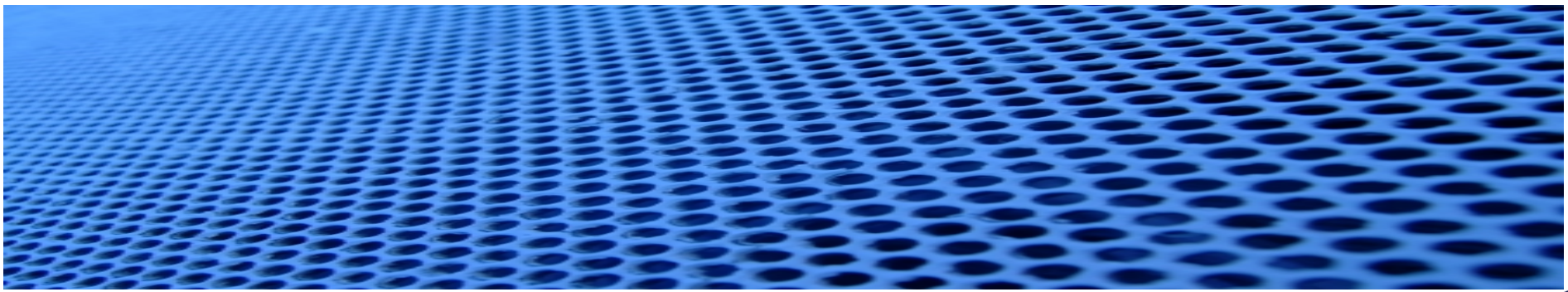
The Grid, from a User's Perspective

How

- Be part of a **Virtual Organization**. If you can't find one, you must set it up.
- Access the Grid via a **User Interface**, authenticating via an **X.509 digital certificate**.
- Specify your **job requirements** via a *Job Description Language*.
- Your job requirements will be **matched** against available resources. If suitable resources are found, your job will *sooner or later* run *somewhere*.
- You will be able to check job status, collect output, store, find and retrieve data.

Architecture

- Emphasis on **sharing** resources at a (virtual) organizational level.
- Mainly adopted by **scientific communities**, with limited industry uptake.
- Typically **batch-focused**, with limited provision for interactive, dynamic usage.



The Cloud, from a User's Perspective

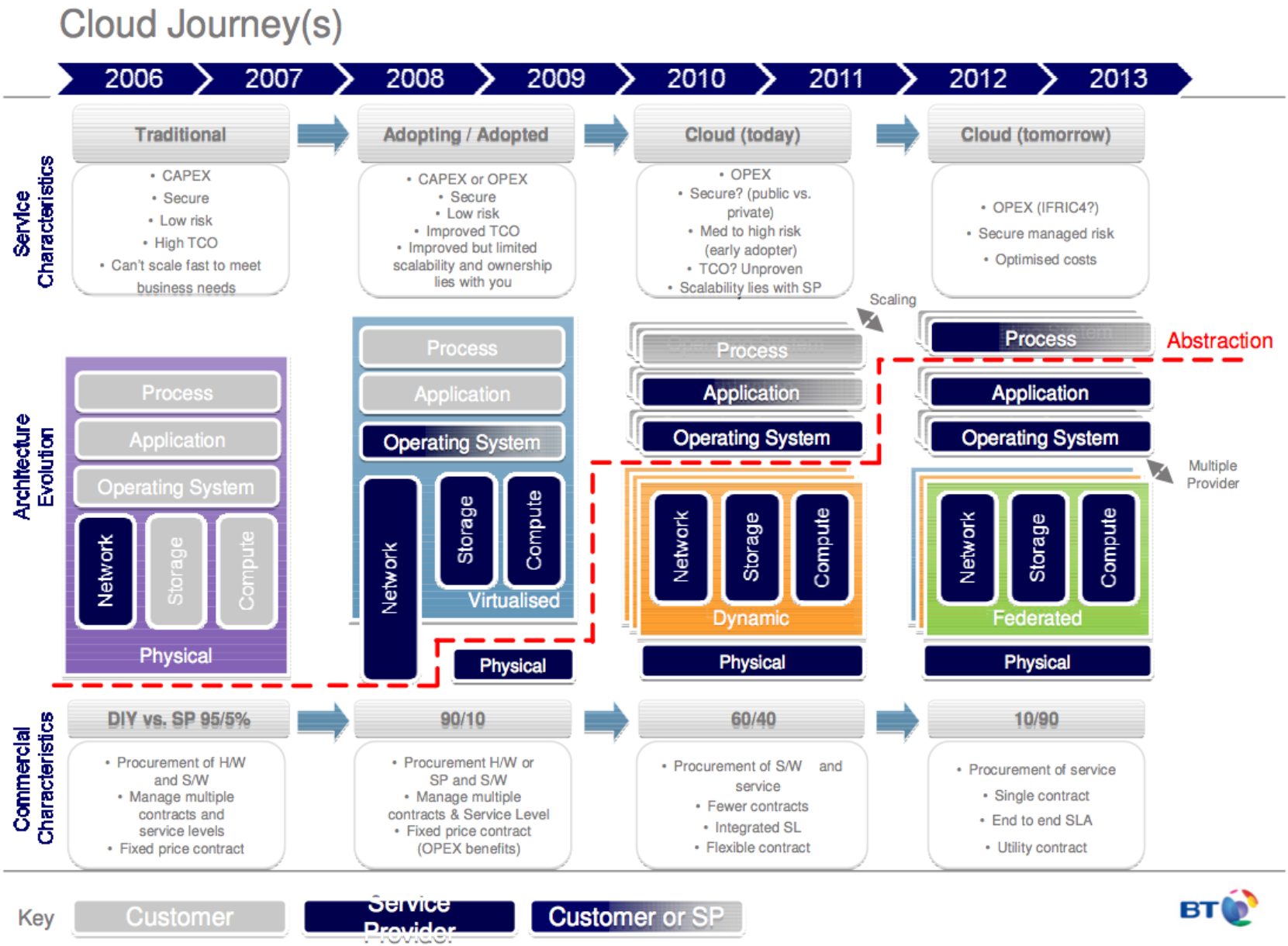
How

- Identify a **Service Provider**.
- Allocate your **seemingly infinite** desired resources, typically through Web Applications.
- Gain access to your resources (which can be services, software applications, hardware cores) through **pay-as-you-go models**.

Architecture

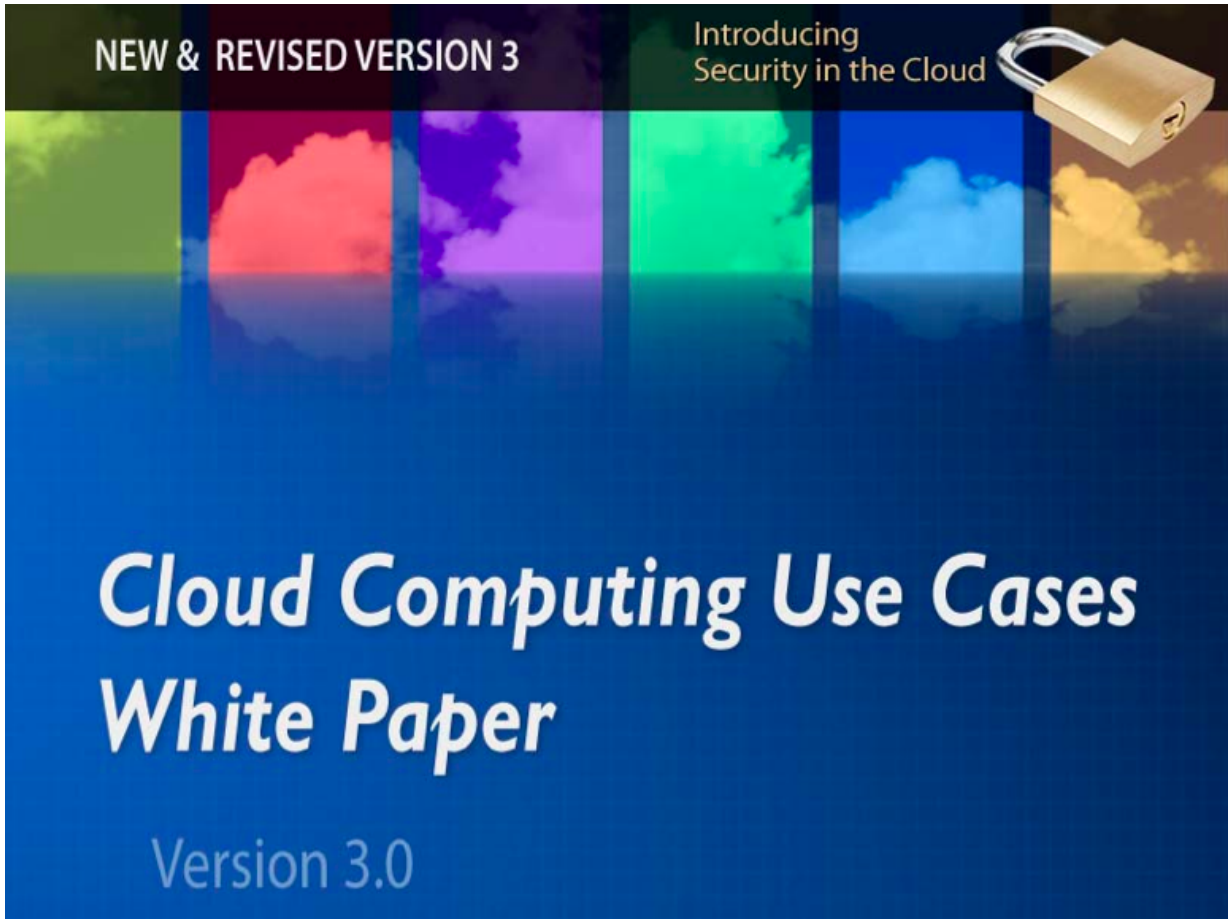
- Emphasis on **ease of access** to resources for *individual users*.
- Initiated within the **commercial sector**, with wide success.
- Several **level of abstractons** are possible: Infrastructure as a Service, Platform as a Service, Software as a Service, etc.

The evolution of cloudy promises



Which use cases are really there for the Cloud?

See the Whitepaper by the Cloud Computing Use Case Discussion Group*:



*http://goo.gl/aCN0

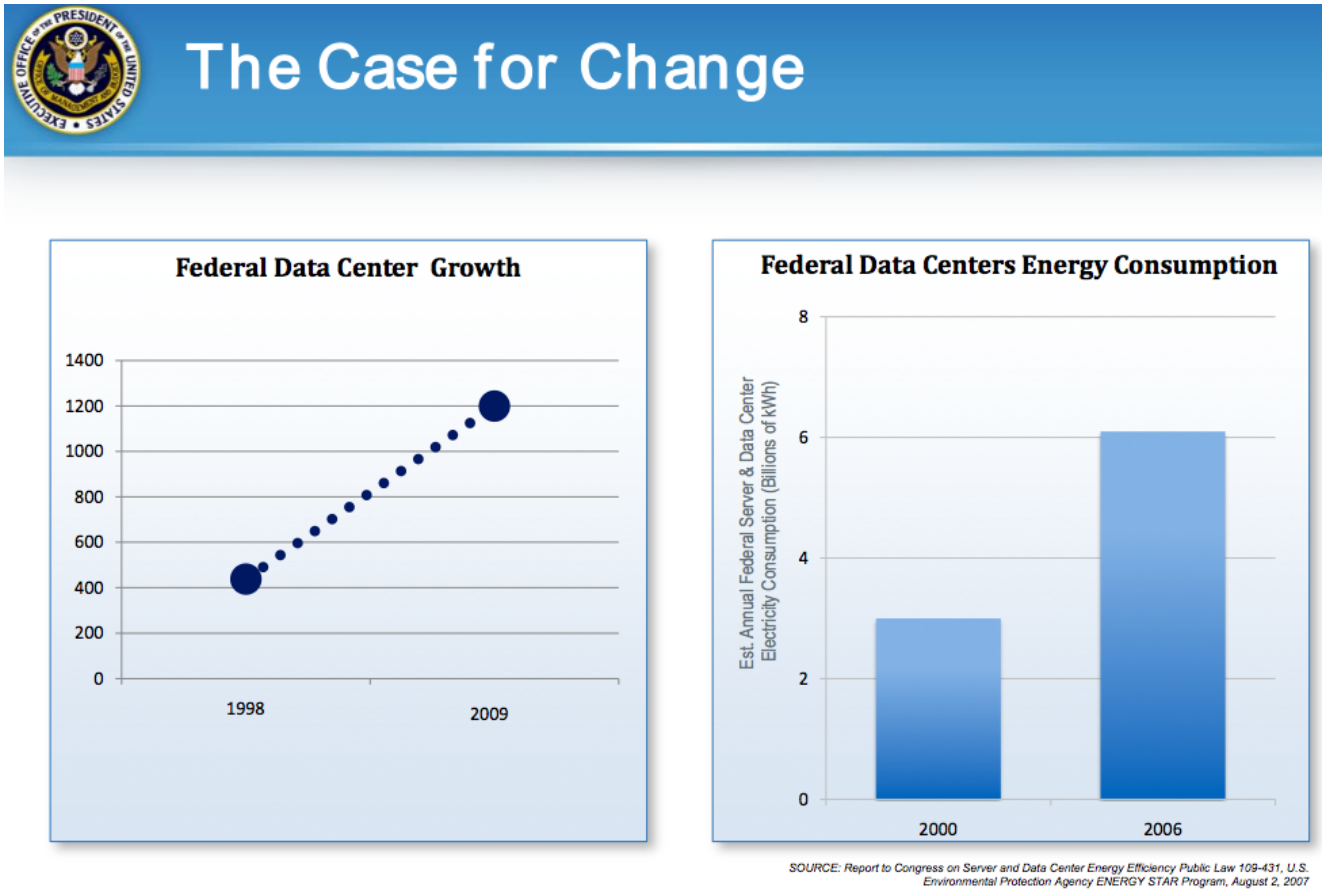
Mantras: Service, Cost Savings, Consolidation, Infinity, Utility. All On-Demand.

January 27, 2010: The UK government has unveiled a sweeping strategy to create its own internal "cloud computing" system – such as that used by Google, Microsoft and Amazon – as part of a radical plan that it claims could save up to £3.2bn a year from an annual bill of at least £16bn.

> *The key part of the new strategy [...] will be the concentration of government computing power into a series of about* ***a dozen*** *highly secure data centres, each costing up to £250m to build, which* ***will replace more than 500*** *presently used by central government, police forces and local authorities.*
>
> *By 2015, the strategy suggests, 80% of central government desktops could be supplied through a "shared utility service".*
>
> *The new "cloud" system will not include the security services such as MI5 or MI6, which have their own, separate systems.*

(This is just one of the several National Cloud Initiatives currently being pursued around the globe. Italy has IGI, the Italian *Grid* Infrastructure.)

More on cost savings



Taken from V. Kundra, Whitehouse Federal CIO, *The Economic Gains of Cloud Computing*, April 7, 2010.

Can this be really used in the scientific world?

It appears that at least in some cases the answer can be yes. Take for example the Gaia project* of the European Space Agency, whose goal is to survey about a billion stars to make an extremely precise three-dimensional map of our galaxy:

> *For the full 1 billion star project numbers [the Gaia Science Operations Development Team] calculated that they will analyze 100 million primary stars, plus 6 years of data, which will require a total of 16,200 hours of a 20-node EC2 cluster. That's an estimated total computing cost of 344,000 Euros. By comparison, an in-house solution would cost roughly 720,000 EUR (at today's prices) – which doesn't include electricity or storage or sys-admin costs. (Storage alone would be an additional 100,000 EUR.)*§

*http://goo.gl/R00j

§http://goo.gl/ZWt8

Sed contra. . .

Problems solved with Clouds? Not really.

Based on the successful multi-year experience built on Grids, some "heavy users" note that, for what regards Clouds:

- Inter-operation of multiple cloud providers is not a reality yet, and **vendor lock-in** is a big issue.
- Political, legal, or security-related considerations discourage the idea of **outsourcing "control"** to external entities.
- These concerns are particularly acute in the case of the interconnection of different components: computing, **storage** and network resources.
- For example: given the level of optimization that was needed for the interaction between storage and computing resources in High-Energy Physics experiments, it is debatable whether the same **performance** can be achieved by general purpose infrastructures, like commercial clouds.
  - Customers will pay either in terms of latencies, or in terms of extra (likely not-negligible) costs.

Sed contra. . .

Problems solved with Clouds? Not really.

Based on the successful multi-year experience built on Grids, some "heavy users" note that, for what regards Clouds:

- Inter-operation of multiple cloud providers is not a reality yet, and **vendor lock-in** is a big issue.
- Political, leg rage the idea of **outsourcing**
- These conce interconnection of different c rk resources.
- For example: given the level of optimization that was needed for the interaction between storage and computing resources in High-Energy Physics experiments, it is debatable whether the same **performance** can be achieved by general purpose infrastructures, like commercial clouds.
  - Customers will pay either in terms of latencies, or in terms of extra (likely not-negligible) costs.

Integrating Cloud Features

Can we *adapt and re-use* our existing Grid-related know-how and infrastructures?

New requirements to existing Grid infrastructures

While Grid interfaces are widely used esp. by large communities, Cloud computing offers significant advantages for many uses.

Ideally, though, one would like to adopt Cloud services so that:

- Resources are shared between *access interfaces* (Grid, Cloud, or else).
- Scalability is ensured.
- Existing services and agreements are not required to change.
- Resource center policies and know-how are honored.
- New services can attract both existing and new customers.

These are both key challenges and opportunities for existing Grid infrastructures.

Examples of services requested *today*

Some of the typical *new* service requests:

- Customer-definable software environments. This is a feature that finds several uses in "traditional" Grids as well.
- Setting up dynamic pools of virtual servers (e.g., user interfaces, or worker nodes for parallel interactive analysis). More generally, flexibly allocating hardware resources through complex advance-reservation requests.
- Instantiating pre-packaged, ready-to-go services.
- Truly distributed, on-demand, Cloud storage.
- Not everybody "speaks Grid": providing access to distributed, traditional Grid infrastructures as if they were not Grids, also to non-traditional users, like Public Administrations, or to the private sector.

The key problem is one of integration between several access interfaces (Grid, Cloud, or else).

Grids and Clouds: common grounds

Grids and Clouds (abstracting from the concept of a "Grid job", which one should regard as an implementation detail) basically target the use of resources.

The two terms come from different grounds, but really they are just different interfaces to access resources.

- Users may actually benefit joining an existing infrastructure, rather than building (or "buying") a new one.
  - This may actually not be a user's choice.
- Sharing of data and resources across Grid/Cloud interfaces should be encouraged.
- Leveraging on multi-year investments and know-how on Grids to incrementally evolve and build new services is a strategic decision.
- Grids like EGEE/EGI are production infrastructures, serving the scientific needs of many (big and small) research communities.

The question is then how in practice can you integrate Grids and Clouds.

Outline

1 Clouds and Grids

2 Resources

3 Integration of access interfaces to resources

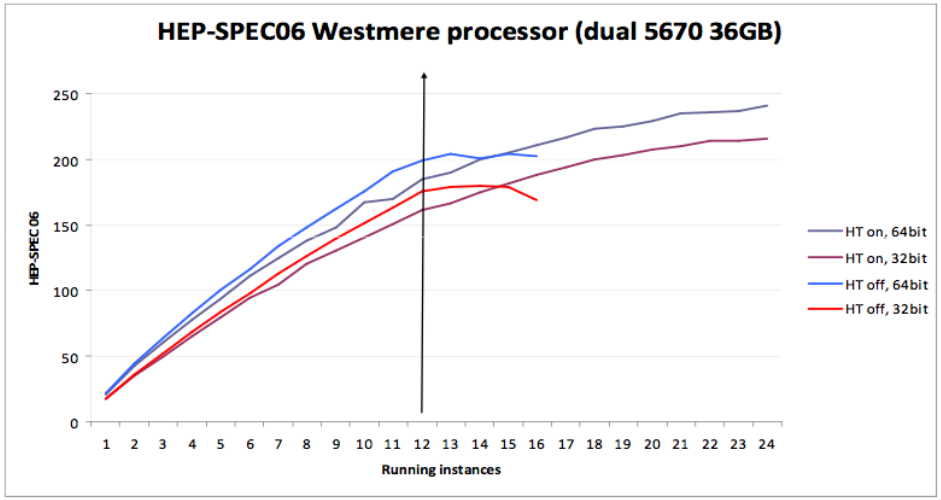
4 Summary

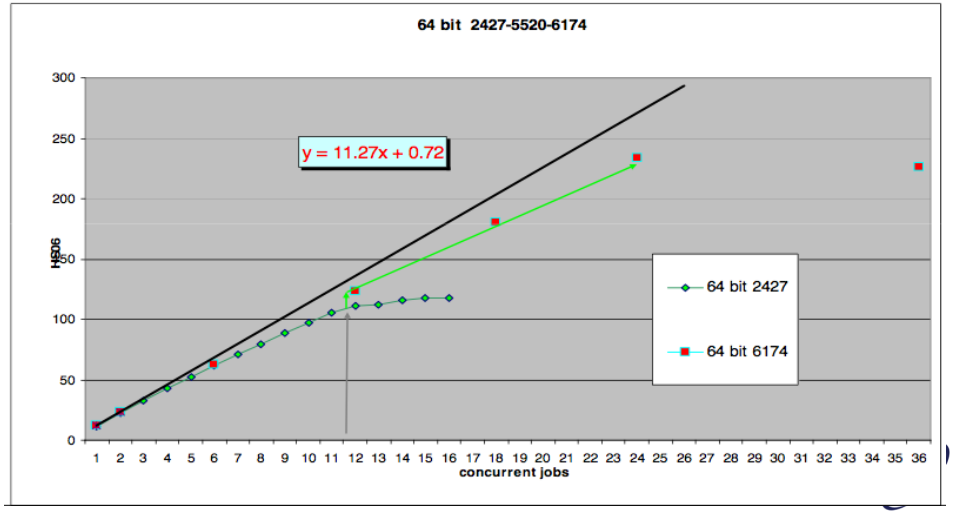
Computing resources: CPU

New processors:

- Intel: Gulftown → Westmere-EP CPU (2010, Q1)
- Intel: Beckton → Nehalem-EX CPU (2010, Q1)
- AMD: Maranello → Magny-Cours CPU (2010, Q1)
- AMD: San Marino → Lisbon CPU (2010, Q2)

CPU	Westmere Gulftown	Nehalem Beckton	Magny Cours	Lisbon
process	32 nm	45 nm	45 nm	45 nm
est GHz at launch	3.33++	2.26++	2.2	2.8
Cores & caches	6 cores 12 MB L3	8 cores 24 MB L3	12 cores 12 MB L3	6 cores 6 MB L3
Net DDR3 ch / socket	3	8	4	2
market	Mid-to-high end	Ultra high end	High end	Mid-to-high end
Perf per core est avg	1 x	0.7 x	0.55 x	0.75x





CPU: some more comparisons

Remarks:

- An Intel 5680 (Westmere) scores 13.8% higher in TPC-C and 25.1% higher in TPC-E than an AMD6176 (Magny-Cours)
- A current Intel Westmere scores ∼200% higher in TPC-C and ∼250% higher in TPC-E than an Intel 5400, released less than 2.5 years ago
- In HEP benchmarks, a motherboard with 2x Intel 5520 (2x 4-core, released March 2009) at 2.27GHz scores ∼41% higher in HEP-SPEC06 than a motherboard with 2x Intel 5420 at 2.50GHz (2x 4-core, released September 2007), with both systems running SL5[a].

Processor	HS06 SLC4	HS06 SL5
2x Intel 5420 @ 2.50GHz	63.10	68.25 (+8.2%)
2x Intel 5520 @ 2.25GHz	81.02	96.53 (+19.1%)
Difference	**+28.4%**	**+41.4%**

[a] http://goo.gl/CdWX

Processor Architecture Process	TPC	2-way	4-way	8-way	16-way
Core2 65nm Xeon 5300 QC 7300 QC	TPC-C TPC-E TPC-H	251,300 5160 only 17,686@100	407,079 479.51 34,990@100	841,809 804.0 46,034@300	- 1,250.0 -
Barcelona 65nm QC	TPC-C TPC-E TPC-H	- - -	471,883 - -	- - 52,860@300	- - -
Core2 45nm Xeon 5400 QC 7400 SC	TPC-C TPC-E TPC-H	275,149 317.45 -	634,825 729.65 -	Linux DB2 1,165.56 -	- 2,012.8 (R2) 102,778@3T
Shanghai 45nm QC	TPC-C TPC-E TPC-H	- - -	579,814 635.4 -	- - 57,685@300	- - -
Istanbul 45nm 6C	TPC-C TPC-E TPC-H	- - -	- - -	- - 91,558@300*	- - -
Nehalem 45nm Xeon 5500 QC 7500 8C	TPC-C TPC-E TPC-H	661,475† 850.0 51,086@100	- 2,022.64 -	- 3,141.76 -	- - -
Westmere 32nm Xeon 5600 6C 7600 12C	TPC-C TPC-E TPC-H	**803,068** **1,110.1**	future future future	future future future	future future future
Magny-Cours 45nm 12C	TPC-C TPC-E TPC-H	705,652 887.4 -	future future future	future future future	future future future

Source: http://goo.gl/tkD5

CPU: some more comparisons

Increasing GHz → Increasing # of cores

(Let's not be fooled too much by absolute numbers, especially when they come out of synthetic benchmarks.)

Probable pros:

- Decreased power consumption
- Increased density
- In the end, the market is always right (pro?)

Probable cons:

- Considerable complexity in programming efficient many-core applications
- Several subsystems are to carefully considered for what regards scalability: I/O, RAM, network

Remarks:

- An Intel
  and 25.
  Cours)
- A currer
  C and
  leased l
- In HEP
  (2x 4-cc
  ∼41% h
  2x Intel
  ber 200

Proce
2x Int @ 2.5
2x Int @ 2.2
Differ

[a] http:

	16-way
9	-
	1,250.0
300	-
	-
	-
300	-
32	-
6	2,012.8 (R2)
	102,778@3T
	-
	-
300	-
	-
	-
00*	-
	-
6	-
	-
	future
	future
	future
	future
	future
	future

But what about GPUs?

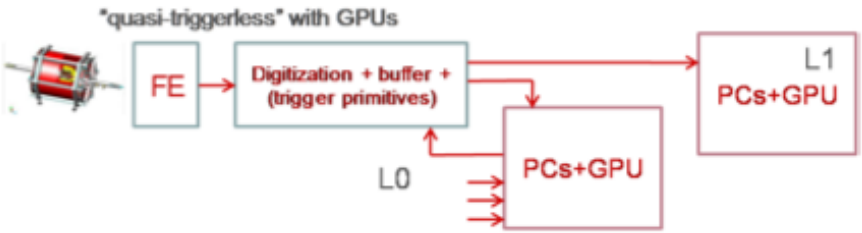
GPU	G80	GT200	Fermi
Transistors	681 million	1.4 billion	3.0 billion
CUDA Cores	128	240	512
Double Precision Floating Point Capability	None	30 FMA ops / clock	256 FMA ops /clock
Single Precision Floating Point Capability	128 MAD ops/clock	240 MAD ops / clock	512 FMA ops /clock
Warp schedulers (per SM)	1	1	2
Special Function Units (SFUs) / SM	2	2	4
Shared Memory (per SM)	16 KB	16 KB	Configurable 48 KB or 16 KB
L1 Cache (per SM)	None	None	Configurable 16 KB or 48 KB
L2 Cache (per SM)	None	None	768 KB
ECC Memory Support	No	No	Yes
Concurrent Kernels	No	No	Up to 16
Load/Store Address Width	32-bit	32-bit	64-bit

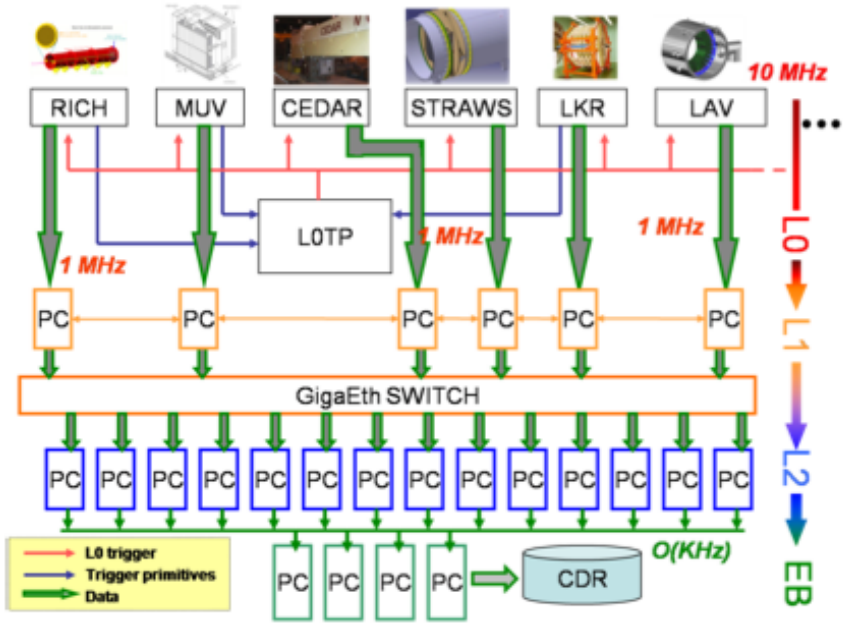
Thanks to R. Ammendola (also for the next 2 slides). For details, see his recent talk "Review on the GPU-related activities in INFN" at `http://goo.gl/WZOA`.

	Xeon X5670	Opteron 8439	ATI HD 5870	Tesla C1060	Tesla C2070
# of cores	6	6	1600	240	448
SP GFlops	140	134	2720	933	1030
DP GFlops	70	67	544	78	515
GiB of Mem	-	-	1	4	6
TDP (Watt)	95	105	188	188	247
Price	1600	2000	400	1500	< 2000
€ / GFlops	23	30	1.4	19	< 4

GPUs for fast triggering and pattern matching at the CERN experiment NA62

- NA62 aims at measuring the BR of the ultra-rare $K^+ \to \pi^+ \nu\nu$ decay with O(100) events, using a very intense kaon beam produced at the CERN SPS
- Effective, selective and lossless readout and trigger (TDAQ) systems are crucial to collect a huge statistics in a reasonable time (2 years of data taking)
- Three level trigger:
  - L0 hardware trigger level with fixed latency: 1ms
  - L1 and L2 software levels with variable latency
- 10 MHz in input at L0, 1 MHz in input at L1, O(100) KHz in input at L2



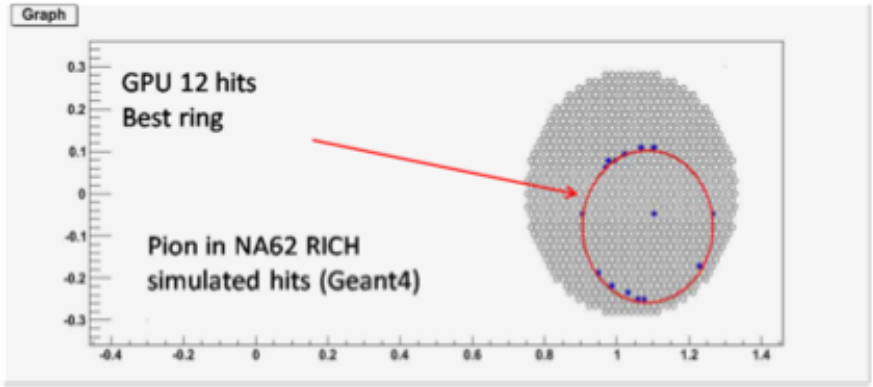


- High computing power, both in hardware and in software levels, could help to design a flexible and powerfull trigger system.
- Investigation to use massive parallel computing power in the Video Card processors (GPU) to implement pattern recognition and trigger algorithm

GPUs for fast triggering and pattern matching at the CERN experiment NA62

- The realtime use of the GPUs is slightly different from the parallel computing: very large input bandwidth, short latency, quasi-deterministic computing time, events processing parallelization, ...
- First attempt to understand the GPUs capabilities: fast pattern recognition in NA62 RICH
- The problem: find rings (with less than 20 hits each) in a sparse matrix (1000 points) at 10 MHz with a latency of 1 ms using a standard PC with a dedicated "video "card
- Several algorithms tested to fit the problem to the processor structure.
- Best result: 3.1 us per ring on NVIDIA TESLA C1060 (1 Teraflops) on sets of 1000 events

- Further development:
  - Use multiple video cards in parallel
  - Upgrade to the next generation GPUs (NVIDIA "FERMI "→ about 2 Teraflops)
  - Study use of GPU in a real time Operating System environment
  - Use a "smart "data link for data pre-processing on FPGA and DMA (Direct Memory Access)



NA62 Pisa Group:

G.Collazuol, G.Lamanna, M.Sozzi

Interactions between computing, storage, network resources: a simple example

A **minimalistic workflow for analysis** performed in the context of a physics experiment is such that in general a user will:

- Decide which analysis to perform for his new research.
- Develop the code which performs it; this is typically a high-level macro or a plugin of some experiment-based software framework.
- Ask a system about the data requirements:
  - Which files contain the needed information. This info is often in an experiment-based metadata repository or file catalogue.
- Ask another system to process his analysis. This could happen via a Grid, a Cloud, local (virtual) batch farms, or possibly even one's own computer.
- Collect the results.

Thanks to F.Furano and V.Vagnoni

A possible approach to the workflow

One could:

- Carefully choose where to send a processing job, e.g. to the place which best matches the needed data set.
- Use tools to create local replicas of the needed data files. High-Energy Physics data files are typically big and fairly static, so it is better to exploit locality if possible.
  - In the right places.
  - Eventually use tools also to push new (produced) data files to the "official" repositories. Think here of complementarity between "Grid" and "Cloud" tasks.
  - If overdone, this can be quite time and resource consuming.
- Any variation is possible. E.g., pre-populate everything before sending jobs.

Thanks to F.Furano and V.Vagnoni

A possible approach to the workflow

One could:

- Carefully choose where to send a processing job, e.g. to the place which best matches the needed data set.
- Use . High-Ene so it is bett
  - s to the
  - n "Grid"
  - If overdone, this can be quite time and resource consuming.
- Any variation is possible. E.g., pre-populate everything before sending jobs.

Remember: any variation is possible

I.e., what is best is left to *you* (application, framework developer) to decide.

Observe the pattern: we are still talking about distributed computing; but we can range from maximum to minimum trust in *middleware*.

- Let the system fully decide on resource discovery/selection and on job/data placement
  → a traditional, orthodox Grid with its Workload Management Systems (WMS)
- Let me talk to selected, trusted resource centers telling them to allocate resources for me
  → a *pilot-based* factory with e.g. Grid Computing Elements (CE)
- Let me manage myself entirely and allocate the services I need, when I need them
  → an orthodox Cloud-like model.

Thanks to F.Furano and V.Vagnoni

Virtualization, or Resource Abstraction

Nothing particularly new *per se*: virtual machines have existed for years, like other virtualization technologies, e.g. virtual memory, virtual storage, virtual networks.

- The IBM M44/44X explored paging systems and the virtual machine concept in 1965.
- The SoftPC software emulator of x86 hardware was introduced in 1988.
- The first version of the open-source XEN was released in 2003.
- FEDERICA (Federated E-infrastructure Dedicated to European Researchers Innovating in Computing network Architectures)* is a 30-months (1/2008-6/2010) EU co-funded project to support research in virtualization of e-Infrastructures integrating network resources and nodes capable of virtualization.

What changed is how powerful, ubiquitous, and (relatively) easy to use virtualization technologies are or are starting to be.

*http://www.fp7-federica.eu/

Virtualization: good, but...

Handle with caution: there are several considerations to be made when applying virtualization technologies, and they tend to rapidly vary with market changes.

- Virtualization still usually involves a few percentage points of performance loss.
- This loss might not be negligible when I/O is involved, especially if proper steps are not taken (typically, use of para-virtualized drivers.)
- How do you virtualize storage?
  A `Dropbox` model is relatively easy to design and possibly implement. But what about guaranteeing latency? I/O throughput? Virtual databases? Efficient virtual content distribution?
- Virtualized resources with a dynamic (perhaps short) lifetime pose new challenges to distributed file systems.
- Some resources are still difficult to virtualize: for example, GPUs.

Outline

1 Clouds and Grids

2 Resources

3 Integration of access interfaces to resources

4 Summary

A practical Grid/Cloud integration example: WNoDeS

The **W**orker **N**odes **o**n **De**mands **S**ervice (WNoDeS) is a software INFN is developing. It is built around a tight integration with a LRMS (a "batch system") and is running in production at the INFN Tier-1 Computing Center. Its main characteristics are:

- Full integration with existing computing resource scheduling, policing, monitoring and accounting workflows.
- On-demand virtual resource provisioning and VLAN support to dynamically isolate Virtual Machines depending on service type / customer requests.
- Support for users to select and access WNoDeS-based resources through Grid, Cloud interfaces, or also through direct job submissions.

The WNoDeS focus is on providing flexibility to both users and resource providers in a production environment.

A practical Grid/Cloud integration example: WNoDeS

The **W**orker **N**odes **o**n **De**mands **S**ervice (WNoDeS) is a software INFN is developing. It is built around a tight integration with a LRMS (a "batch system") and is running in production at the INFN Tier-1 Computing Center. Its main characteri

- Full … policing, mon…
- On-… dynami-cally … omer re-ques…
- Sup… through Grid…

The INFN Tier-1 – CNAF, Bologna

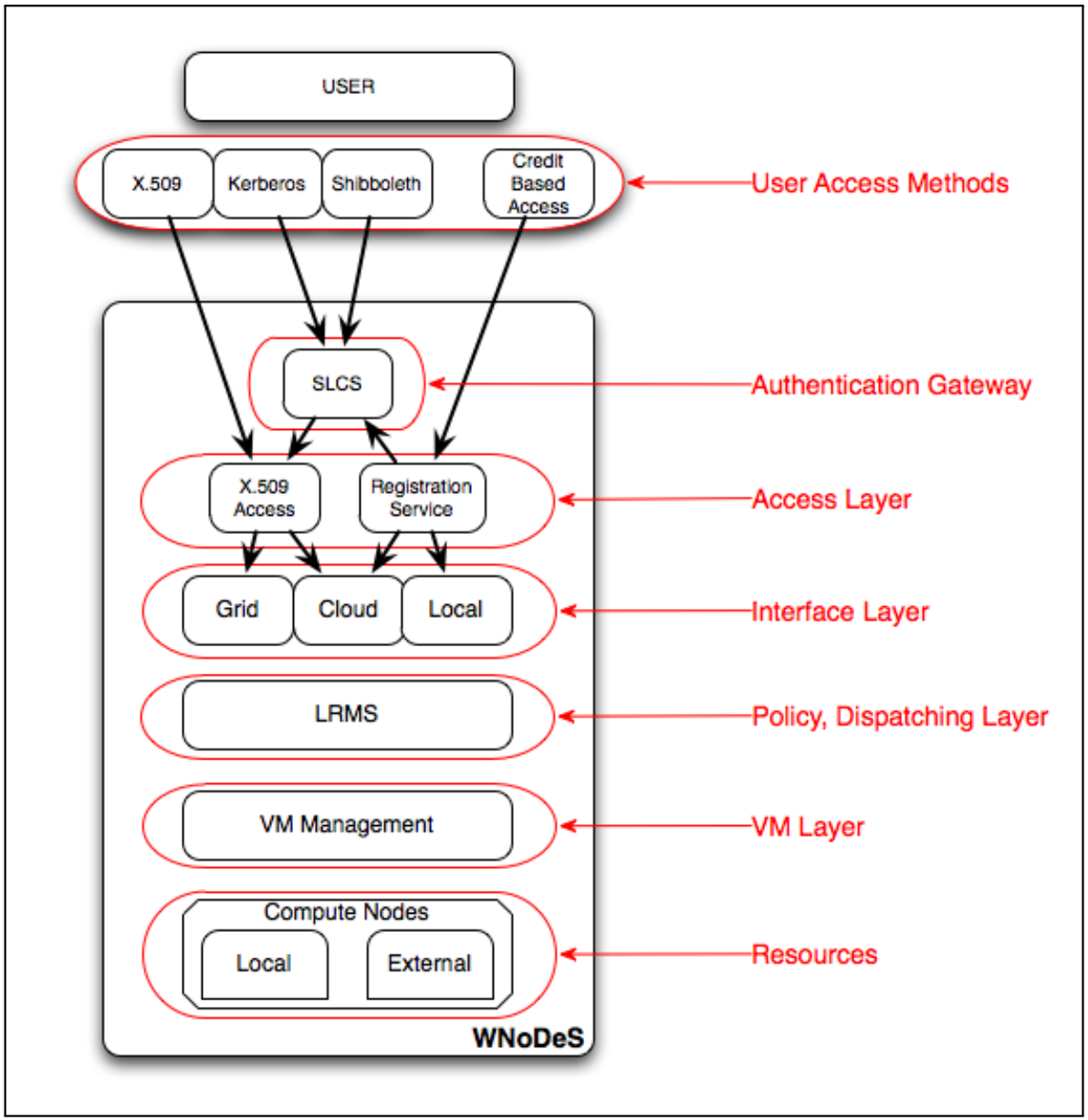
- ∼5500 CPU cores (∼8500 cores by 3Q10)
- CPU power (in HEP-SPEC06):

2009	∼23.5K
1Q10	∼48K (+104%)
3Q10	∼81K (+69%)

- 10 PB of tape / 5.2PB of disk space (∼7PB by 4Q10)
- 2 x 10 Gbit/s WAN links
- 3 power transformers, 7 chillers (∼2.5MW), 2 D-UPS + Diesel engine
- 20 supported experiments / Virtual Organizations
- 2009 average: more than 20K jobs executed per day (with peaks of ∼60K jobs per day)

The WNoDeS focus is on providing flexibility to both users and resource providers in a production environment.

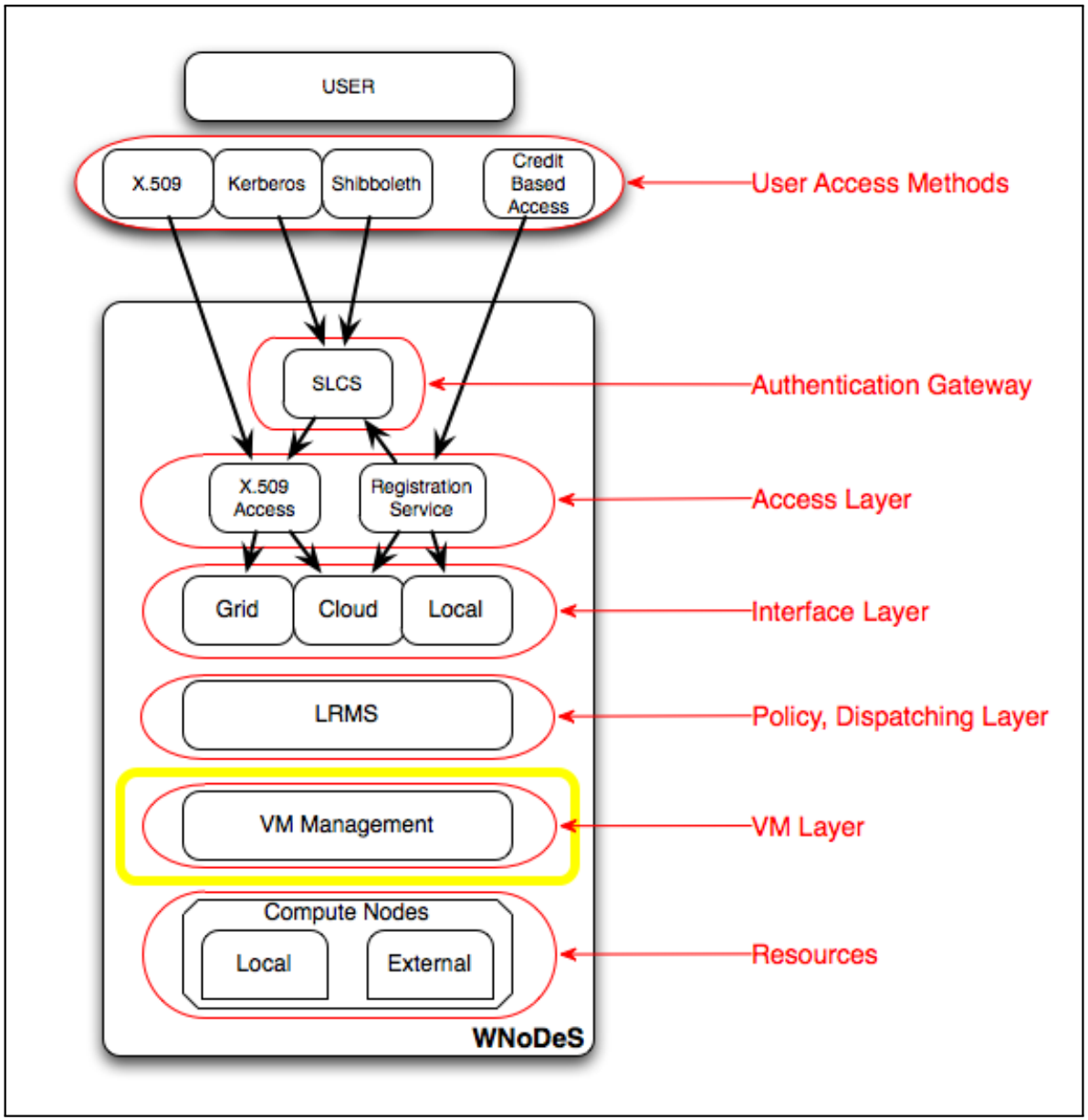
WNoDeS: overall architectural framework



Key points

1. On-demand virtual service provisioning
2. Flexible, integrated scheduling policies
3. Multiple access interfaces
4. Multiple authentication methods
5. Integrated access to existing (e.g., Grid) infrastructures
6. Access to external resources

WNoDeS: overall architectural framework



Key points

1. On-demand virtual service provisioning
2. Flexible, integrated scheduling policies
3. Multiple access interfaces
4. Multiple authentication methods
5. Integrated access to existing (e.g., Grid) infrastructures
6. Access to external resources

The WNoDeS VM layer, i.e. virtualization on-demand

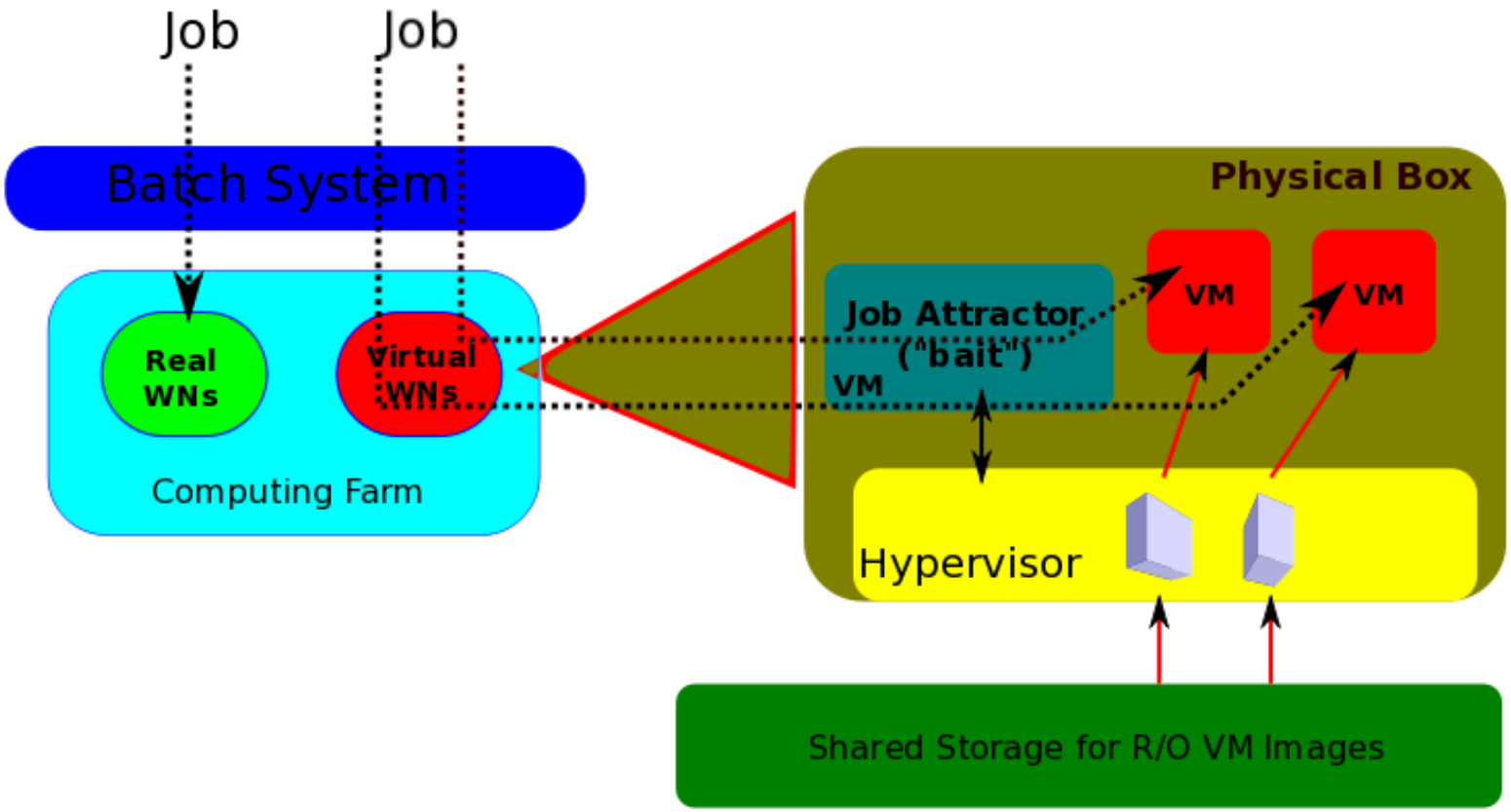
In the WNoDeS architecture, policing, resource allocation and scheduling is directly demanded to an underlying *batch system.* (LRMS)

All physical nodes have in the WNoDeS model the following configuration:

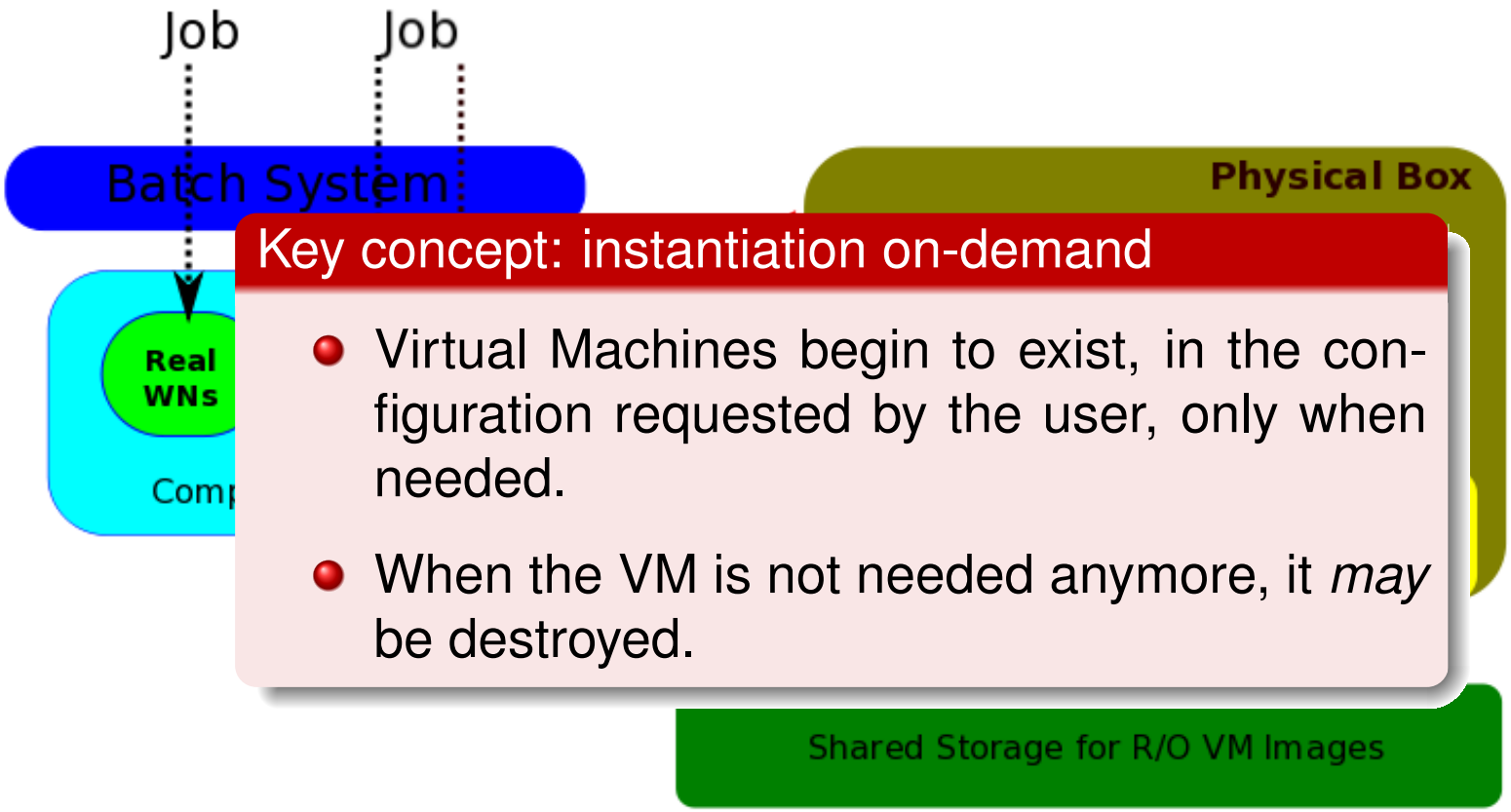
- An hypervisor, i.e. a software layer capable of instantiating Virtual Machines. The hypervisor need not be aware or part of the LRMS.
- A special Virtual Machine, called bait. The bait is part of the LRMS. Its task is to *publish available local resources* to the LRMS.
  → Important: the task of the bait is **not** to execute jobs, but just merely to attract them on the local physical system.

  A bait is always present on each of the physical nodes.
- Zero or more running Virtual Machines, meant to actually execute jobs. These VM are instantiated *on-demand* by the local hypervisor, upon appropriate requests made by the local bait.

WNoDeS: VM instantiation overview



WNoDeS: VM instantiation overview



Key concept: instantiation on-demand

- Virtual Machines begin to exist, in the configuration requested by the user, only when needed.
- When the VM is not needed anymore, it *may* be destroyed.

WNoDeS: VM management details (1)

(an important note first: the process flow previously shown is typically valid for *traditional* batch jobs. Handling of *Cloud requests* is slightly different. More on this later on.)

Here are some key characteristics of the WNoDeS VM instantiation layer:

- The supported batch system today is Platform Computing's LSF. Work is ongoing to port WNoDeS to other batch systems like Torque.
- The hypervisor is KVM.
- VM images are stored in read-only mode on some shared storage. Before a VM image is actually used on a given physical node, it will be locally copied there (*on-demand*, i.e. when the need actually arises.)
- Virtual Machines can be re-used, for example if a job coming to a given node requires a VM with exactly the same requirements of a previous job (this obviously saves some time in the VM preparation phase); or they can be re-generated each time a job arrives to a node.

WNoDeS: VM management details (2)

- To generate a VM, the so-called snapshot mode of KVM is used. In this mode, a VM is started, based on a *VM master copy* (this is the local copy of a given VM). When the VM is running, changes to the master copy are written to temporary files, which are automatically deleted upon termination of the VM.
  Note: this is not to be confused with the concept of *taking a snapshot of a running VM.*
- Thanks to the fully distributed nature of the VM instantiation mechanism (each physical node is fully independent from the others), there is no single point of failure (redundancy at the resource scheduling layer is taken care of directly by the LRMS). There is at the moment only one central service, which can be easily made redundant, called the WNoDeS Name Server (not show in the previous picture.)
- Virtual machines can be part, either upon system management or customer requests, of different Virtual LANs.

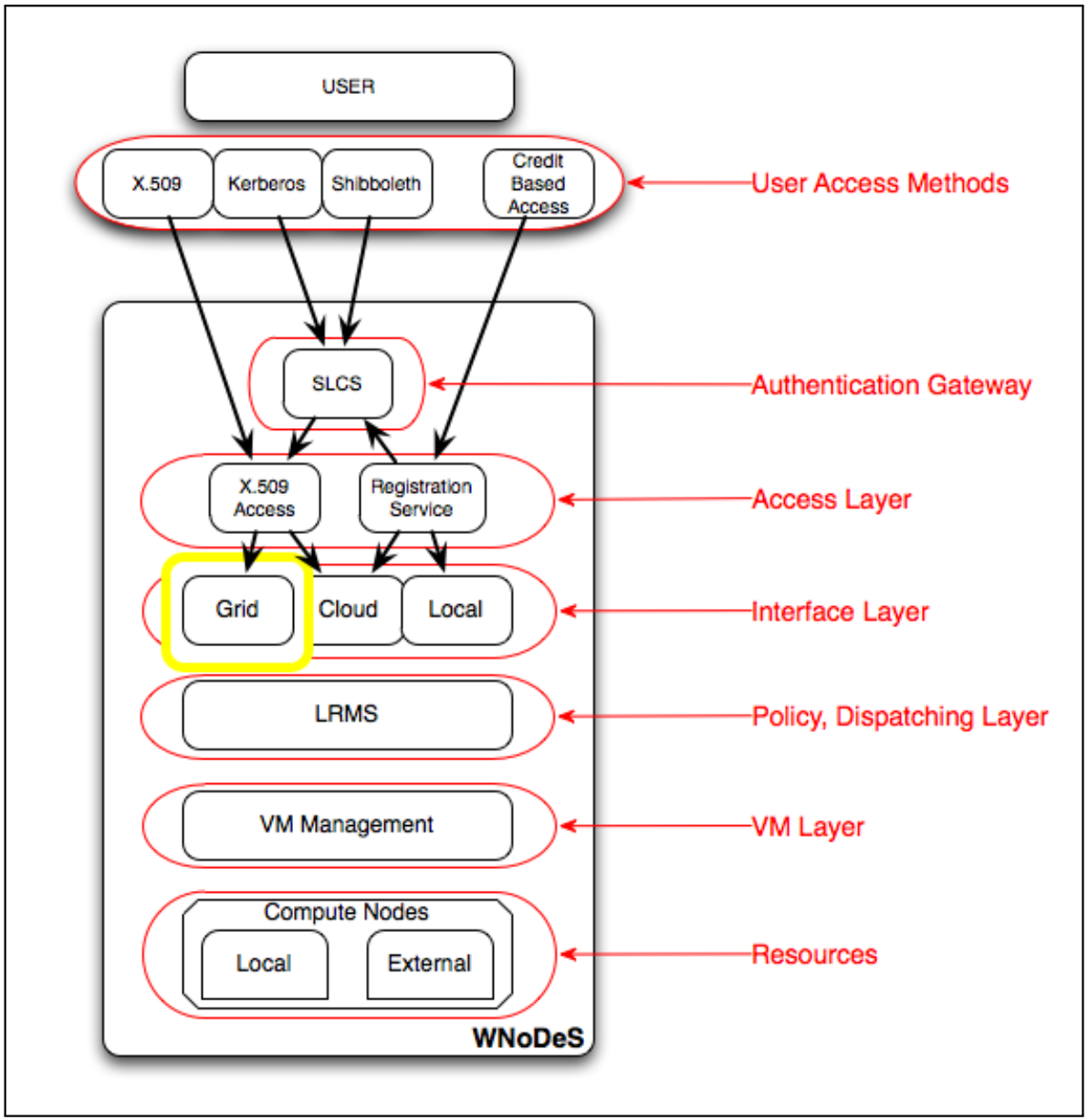
WNoDeS: VM management details (3)

Here are some features which are not currently available in the deployed WNoDeS release. They are being tested in a development branch:

- Advance Reservation or Quality of Service is handled through the corresponding mechanisms available in the underlying LRMS.
- Copying of Virtual Machine Images from shared storage currently happens via a standard Linux *copy* command. Other mechanisms can be used, like *http* (which allows scalability e.g. through http caching servers, or geographic distributions of site-independent VM images), or *Torrent*.
- Support for virtual storage, i.e. in the first place storage which is visible to Virtual Machines via local virtual disks.
- Support for network throttling. This is important e.g. in the case of Cloud requests.

This is all *local*. But integration with Grids is also supported.

WNoDeS: overall architectural framework

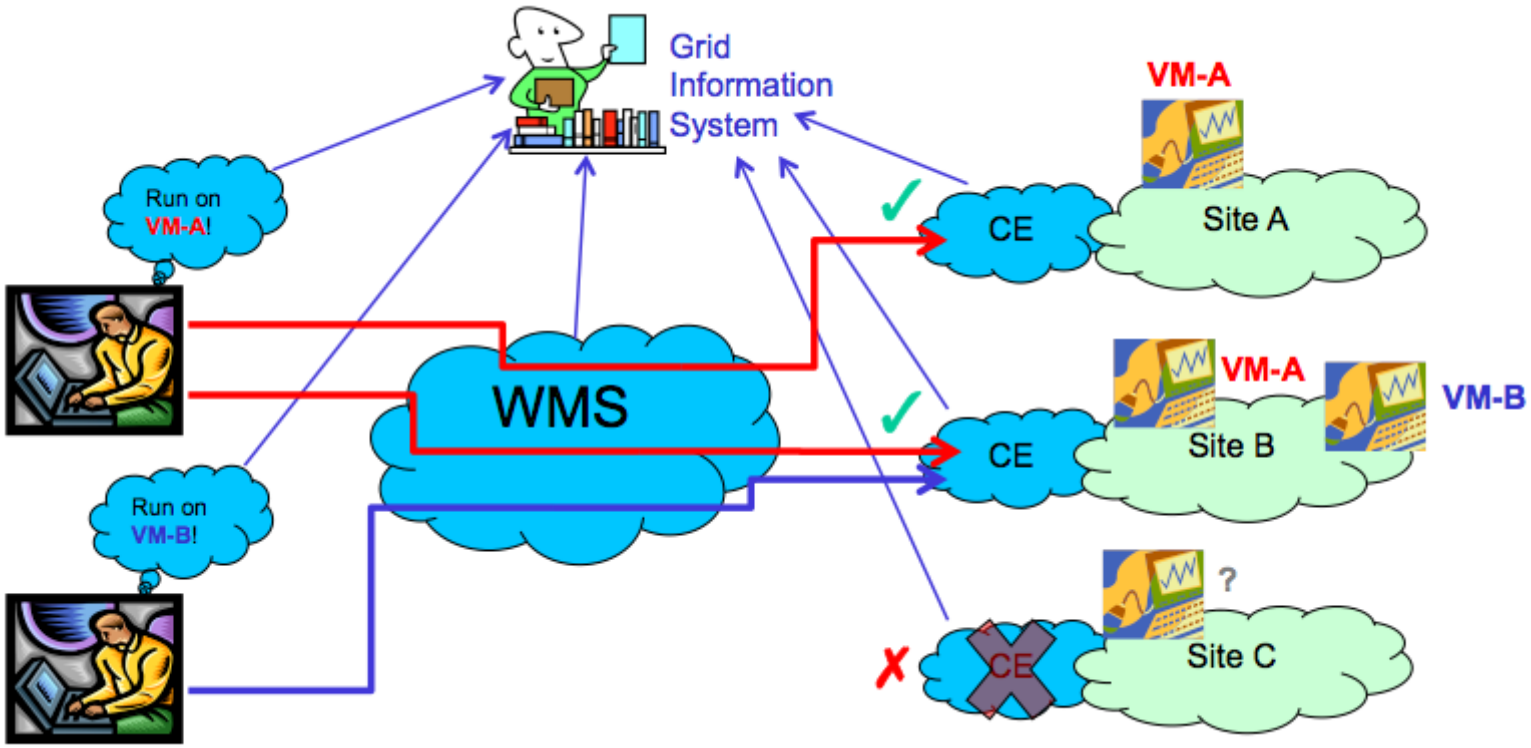


Key points

1. On-demand virtual service provisioning
2. Flexible, integrated scheduling policies
3. Multiple access interfaces
4. Multiple authentication methods
5. Integrated access to existing (e.g., Grid) infrastructures
6. Access to external resources

WNoDeS: integration with Grids (1)

Using WNoDeS, it is possible to select the desired execution environment through standard Grid tools.



WNoDeS: integration with Grids (2)

VM images may be selected through standard grid tools:

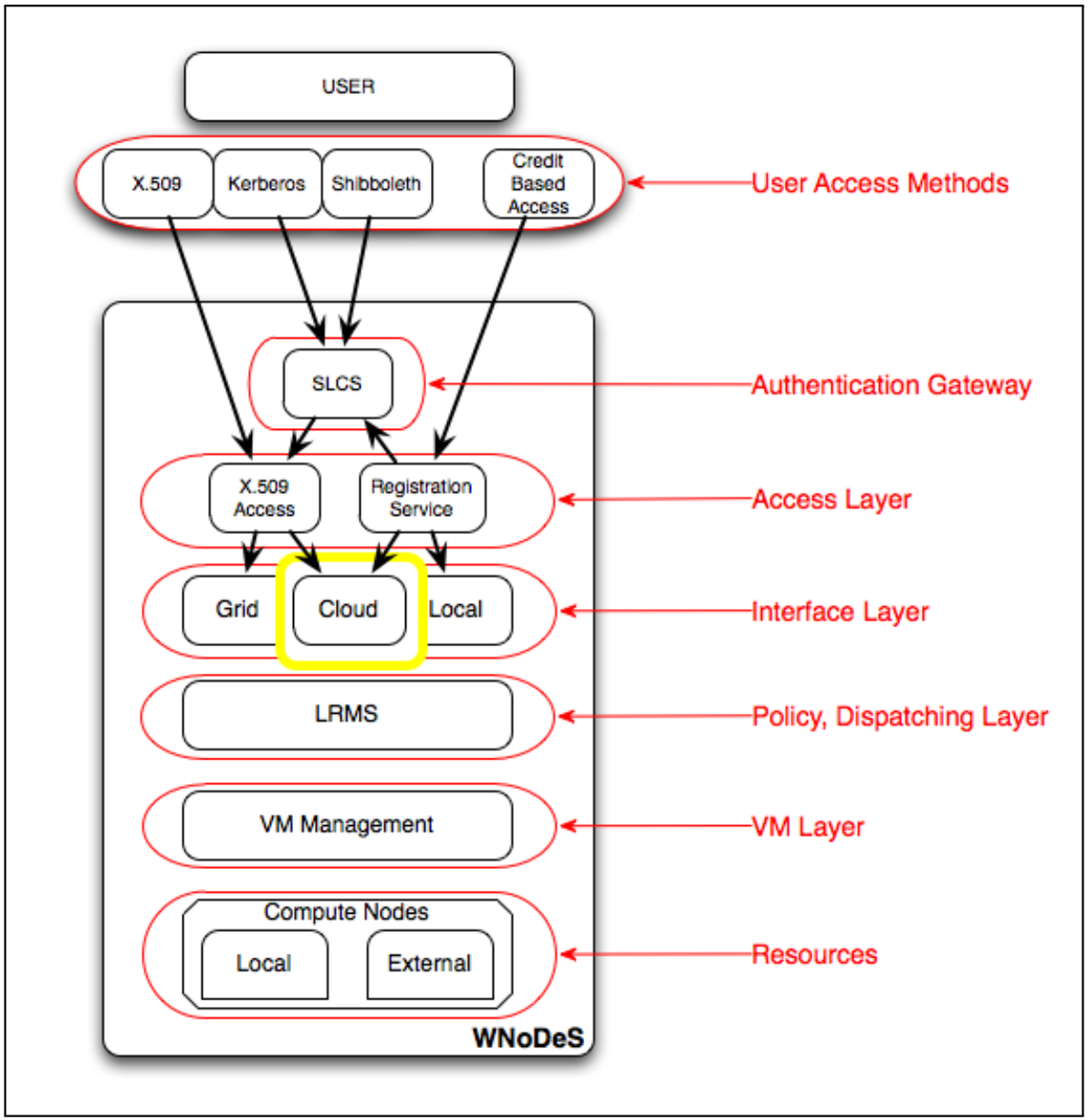
- Available VM images are published in the Grid Information System using the Glue attribute `SoftwareRunTimeEnvironment`.
- Selection of VM images is done by users via standard JDL statements, e.g.

```
CeRequirements = "(Member(\"vm_etics_slc44\",
  other.GlueHostSoftwareApplicationRuntimeEnvironment))"
```

- Users (or sets thereof) must be authorized by resource providers to locally use the selected VM images through proper WNoDeS configuration.
- This works today with the current CE-CREAM software.

Grid is covered. What about Cloud integration?

WNoDeS: overall architectural framework



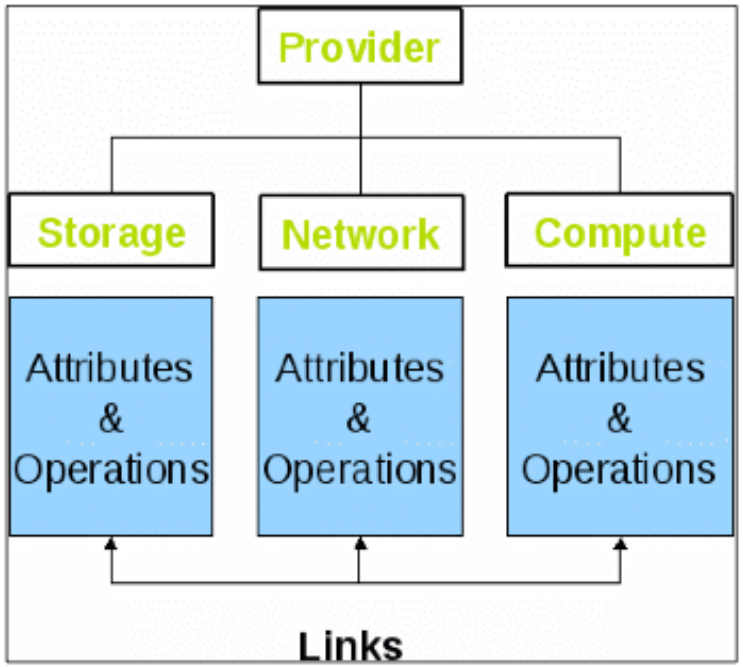
Key points

1. On-demand virtual service provisioning
2. Flexible, integrated scheduling policies
3. Multiple access interfaces
4. Multiple authentication methods
5. Integrated access to existing (e.g., Grid) infrastructures
6. Access to external resources

The OGF Open Cloud Computing Interface

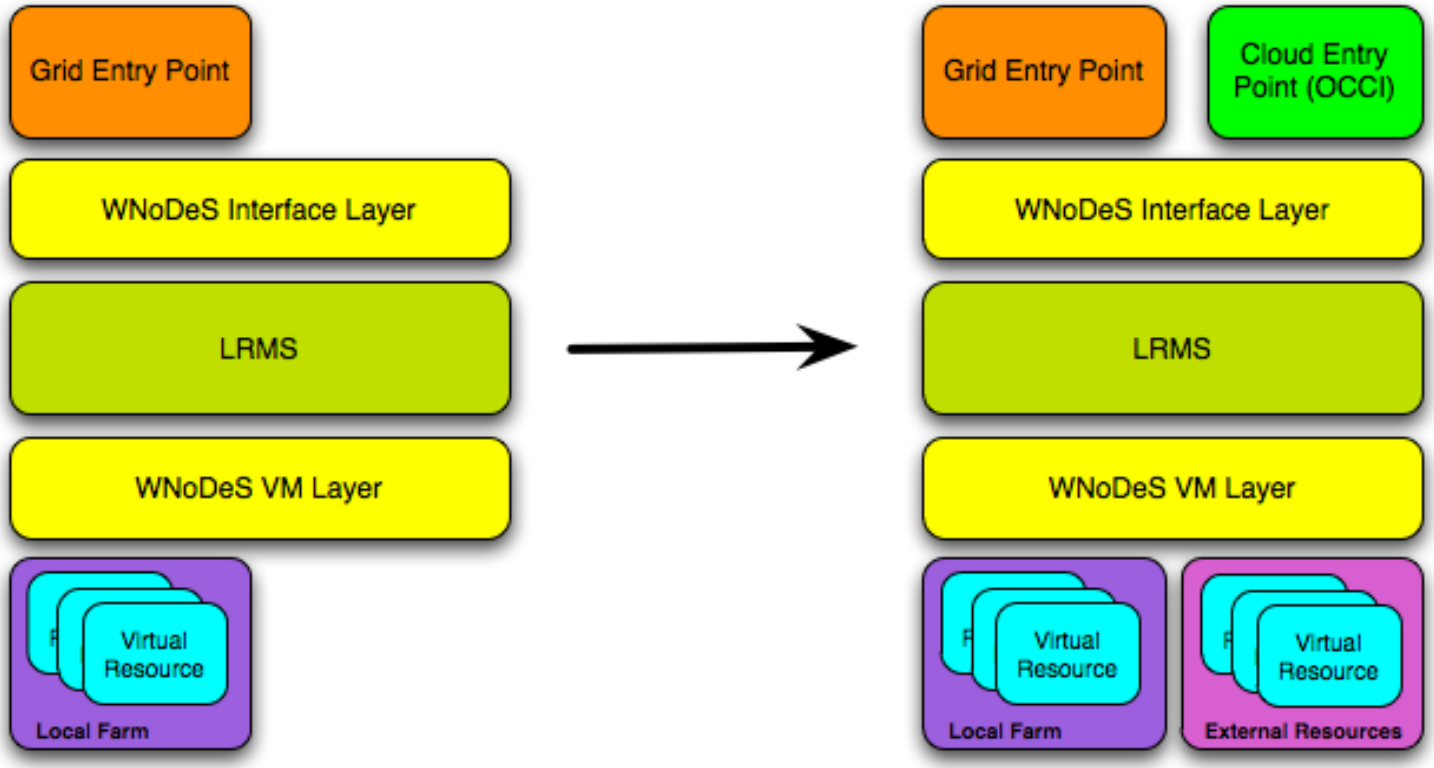
The Open Cloud Computing Interface (OCCI) API is being developed within the Open Grid Forum to access "Infrastructure as a Service" (IaaS) based Clouds.

It is a slim RESTful based API, allowing users to access and manage (computing, storage, network) resources using a Uniform Resource Identifier (URI).



Integrating Cloud services

WNoDeS delivers access to Cloud services through the Open Cloud Computing Interface, implementing a subset of the OCCI API, using X.509 authentication and exposing a REST interface.



WNoDeS Cloud Access today

The WNoDeS Cloud RESTful web service is accessible via a basic alpha-stage Web application (see demo).

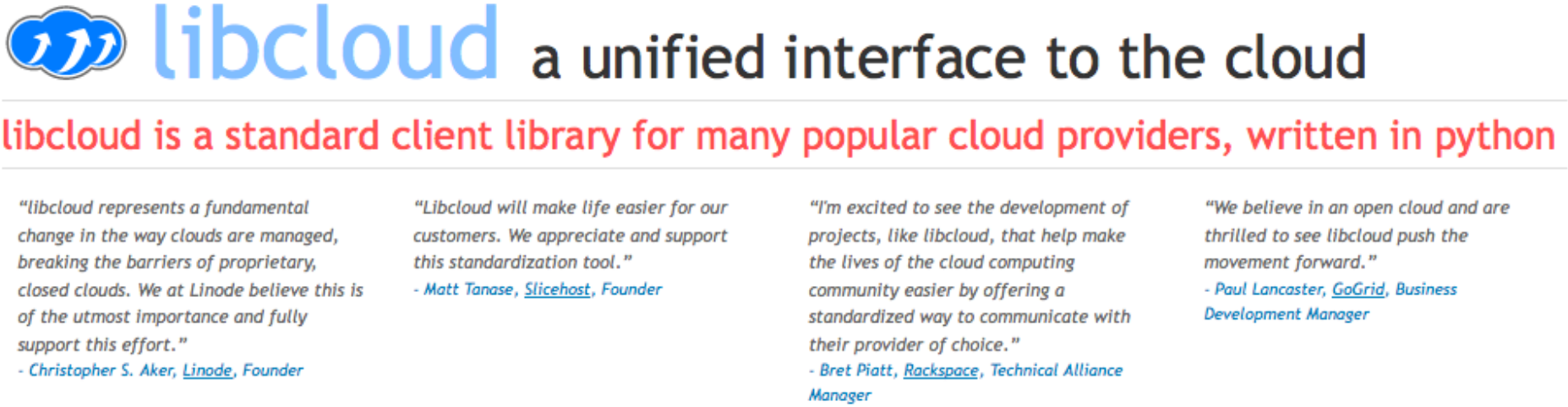
At the moment, upon a VM deployment request a "dummy job" is sent to the LRMS, and from there to a bait; eventually the dummy job runs on the allocated Cloud VM.

This has some disadvantages (like having the Cloud VM to be part of the LRMS cluster). The next WNoDeS version will support Cloud VM instantiations so that:

- A Cloud VM is totally oblivious of the LRMS.
- Control of the Cloud VMs is fully distributed, and consistency is ensured by the baits.

In case of Cloud allocations, only wallclock time is considered for accounting purposes.

Future Cloud enhancement: `libcloud` support



`libcloud` will allow the possibility to access WNoDeS Cloud capabilities not only via the Web App, but also via Python scripts. It will then be possible to programmatically instantiate and manage Cloud VMs.

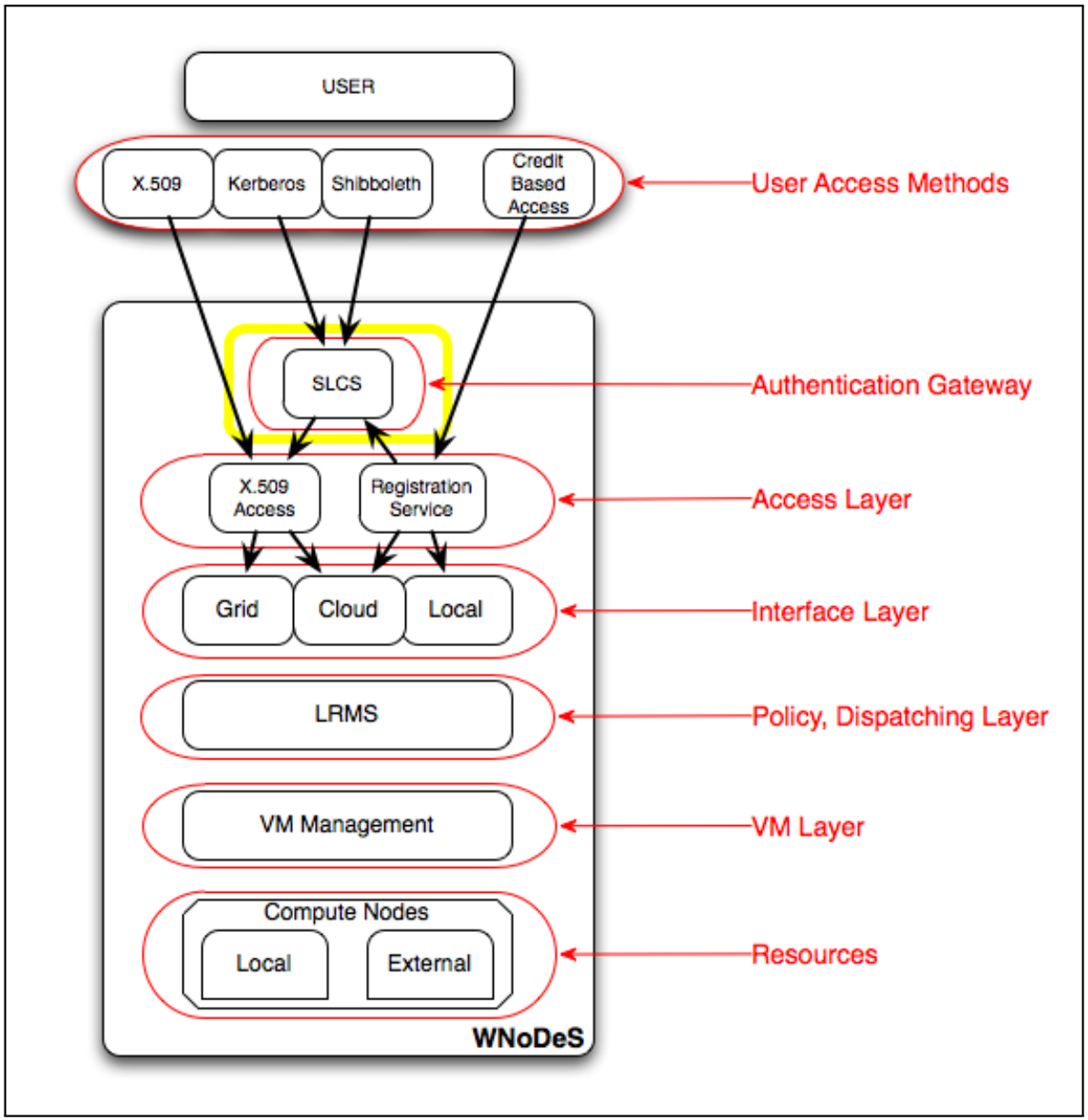
Link: `http://incubator.apache.org/libcloud`

libcloud simple usage example

```
from libcloud.types import Provider
from libcloud.providers import get_driver
from libcloud.deployment import MultiStepDeployment, ScriptDeployment, SSHKeyDeployment

RACKSPACE_USER = 'your username'
RACKSPACE_KEY = 'your key'

Driver = get_driver(Provider.RACKSPACE)
conn = Driver(RACKSPACE_USER, RACKSPACE_KEY)

# read your public key in
sd = SSHKeyDeployment(open("~/.ssh/id_dsa.pub").read())
# a simple script to install puppet post boot, can be much more complicated.
script = ScriptDeployment("apt-get install puppet")
# a task that first installs the ssh key, and then runs the script
msd = MultiStepDeployment([sd, script])

images = conn.list_images()
sizes = conn.list_sizes()

# deploy_node takes the same base keyword arguments as create_node.
node = conn.deploy_node(name='test', image=images[0], size=size[0], deploy=msd)
# <Node: uuid=..., name=test, state=3, public_ip=['1.1.1.1'], provider=Rackspace ...>
# the node is now booted, with your ssh key and puppet installed.
```

How could Cloud access be integrated into existing infrastructures like EGEE/EMI?

WNoDeS: overall architectural framework



Key points

1. On-demand virtual service provisioning
2. Flexible, integrated scheduling policies
3. Multiple access interfaces
4. Multiple authentication methods
5. Integrated access to existing (e.g., Grid) infrastructures
6. Access to external resources

The Grid AuthN/AuthZ mechanisms

Currently, grid tools provide:

- **Authentication** through X.509 certificates
  - Federation of Certification Authorities
  - Single sign-on (trust between users and resources without direct intervention of the organization in the process)
- **Authorization** through attribute certificates
  - Based on Virtual Organizations (VO)
  - VOs define "groups" and "roles" for users: **VOMS**

In general, certificate proxies are used on the infrastructure:

- Limited duration – may be automatically renewed
- May be dynamically delegated (in total or in part) to services that act on behalf of the user (e.g. Workload Management Systems)

Thanks to C.Grandi

Integrating authentication mechanisms

X.509-based access, widely adopted in Grids, is by no means the only authentication mechanism in use. Providing access to services for an expanded customer base means also the need to cater for several authentication methods.

The WNoDeS project is developing an authentication gateway to map several authentication mechanisms (Kerberos, Shibboleth) to a dynamically-assigned, short-lived X.509 personal certificate.

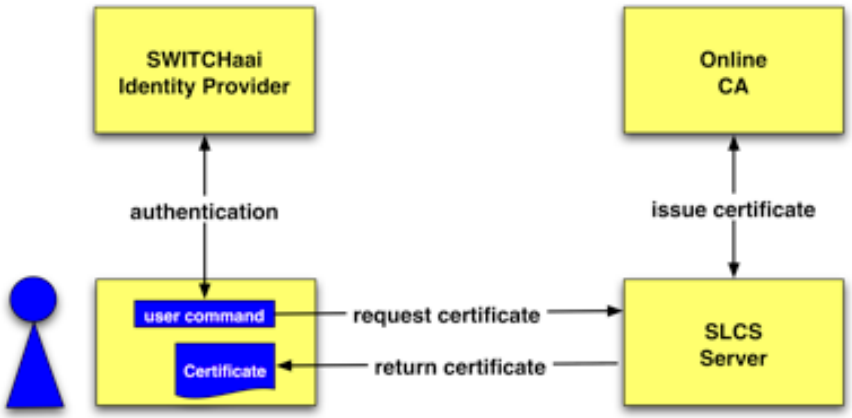
Authentication gateway advantages

- Authentication for the WNoDeS software framework converges around a single method (X.509).
- X.509 is the authentication mechanism used to access the Grid. Generating dynamic X.509 certificates opens up the possibility for e.g. users of Cloud services to access Grid (e.g. EGEE/EGI) resources.

Interoperability with Shibboleth

- Authentication and Authorization Infrastructures (AAI) are now an established reality, many times on a national basis, or at an organizational level.

  Shibboleth is based on the exchange of digitally signed assertions about users, expressed in a language called the "security assertion markup language" (SAML). Now, it is possible to issue short-lived X.509 credentials to users through a *Short Lived Credential Service* (SLCS), based on their successful authentication to a Shibboleth Identity Provider. For example, a SLCS service interoperating with gLite middleware has been in production at SWITCH since April 2007.



Thanks to SWITCH

Interoperability with Kerberos and other methods

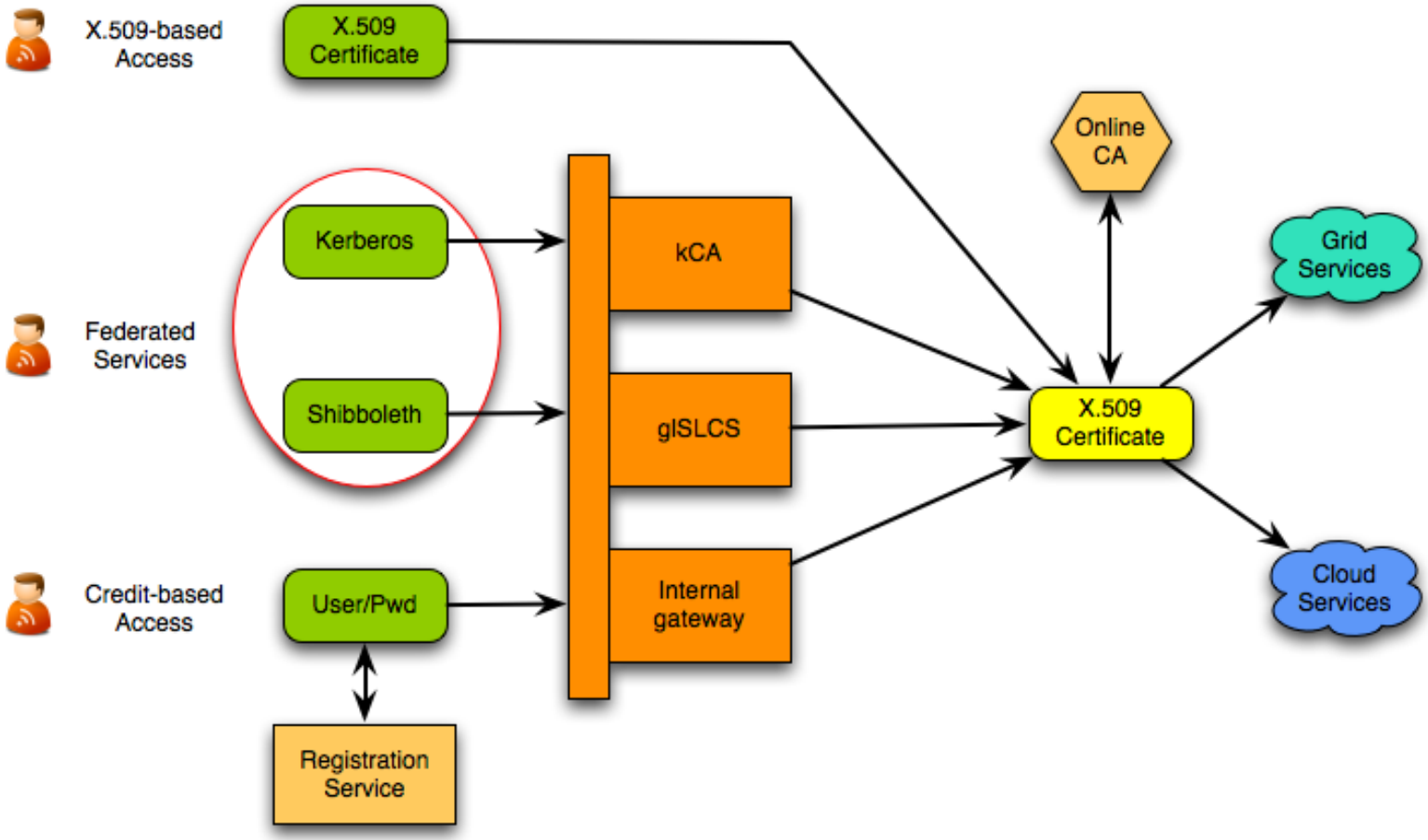
- Similarly to Shibboleth identify providers, several organizations may have site-wide or multi-site-wide **Kerberos** credential systems.

  Based on Kerberos credentials, it is possible to generate short-lived X.509 credentials through a "Kerberized Certificate Authority", or kCA. This is in production for example at the Fermi National Laboratories (FNAL).

- It is also possible to have the case of users wanting to access resources through other (possibly locally-defined) authentication methods. We will call this **credit-based access**. Typically, this type of user authentication is tied to a specific *registration service.*

  Through credit-based access, users may want to get access to Grid, Cloud or simply local resources, given proper agreements.

The WNoDeS authentication gateway



Outline

1 Clouds and Grids

2 Resources

3 Integration of access interfaces to resources

4 Summary

Where are we?

The WNoDes framework is running at the INFN Tier-1

- It is in production with currently ∼1500 on-demand Virtual Machines, O(10) supported Virtual Images, serving 20 different user communities; on average, more than 20,000 jobs are executed each day through WNoDeS.
- The plan is to have ∼4000 Virtual Machines by Summer 2010 and progressively integrate all Tier-1 resources.
- Distributed selection of VM images works either statically (on a customer-community basis), or dynamically (per-user), through standard Grid job submission commands.
- A pilot Cloud service is in place to allocate on-demand resources through OCCI and a web application.
- A first public release is planned for Q3 2010.

So, Cloud – What else is this for?

Beside the use cases mentioned at the beginning, some examples:

- A prototype of a virtual analysis facility: First experiences (*S Bagnasco et al 2010 J. Phys.: Conf. Ser. 219 062033*), see `http://goo.gl/Fs0S`

  > *[. . .] Leveraging on the virtualization of highly performant multi-core machines it is possible to build a fully virtual analysis facility on the same Worker Nodes that compose an existing LCG Grid Farm. [. . .]*

  (BTW: neither resource accounting, nor the static definition of the PROOF nodes, mentioned as problematic in the article above, are issues in WNoDeS)

- The WNoDeS software as a tool to support virtual pools of servers for interactive analysis and software development (*Submitted to CHEP10*)

Several work is still to be done especially in the I/O area, but premises are encouraging.

So, Cloud – What else is this for?

Beside the use cases mentioned at the beginning, some examples:

- A prototype of a virtual analysis facility: First experiences (*S Bagnasco et al 2010 J. Phys.: Conf. Ser. 219 062033*), see `http://goo.gl/Fs0S`

  formant multi-
  nalysis facil-
  existing LCG

  Integration

  WNoDeS encourages integration also with regard to data production and consumption using both Grid and Cloud interfaces.

  Data produced "via the Cloud" can be put in Grid-accessible storage repositories, and vice versa.

  (BTW: n … the PROOF nodes, r … sues in WN-oDeS)

- The WNoDeS software as a tool to support virtual pools of servers for interactive analysis and software development (*Submitted to CHEP10*)

Several work is still to be done especially in the I/O area, but premises are encouraging.

A couple of questions (1)

- Do we still need or want an e-Infrastructure for scientific computing? Do *you* care?

A recent poll by CommVault identified the following as the main obstacles or worries for cloud adoption:

- Security and privacy
- Reliability
- Cost
- Scalability (are there *really* infinite resources?)

As users (perhaps belonging to established communities), is it possible to formalize what our *difference* is? And as *providers*?

A couple of questions (2)

For example, elasticity is a great thing, but. . .

Cloud Elasticity Could Make You Go Broke

March 11th, 2009 · 13 Comments

Ever had a mobile phone and get a bill that was way, way more than you expected? You know what I mean. The day that bill for 700 dollars comes in and your eyes bug out of your head because you could swear (and in fact you do swear – at the customer service rep) that you could not possibly have exceeded your plan minutes? Or maybe you "pay as you

(Daryl Plummer, Gartner) – i.e., **On-Demand Overspending**

That's it – It's Shoppable (although perhaps not yet shippable)

[Cloud computing is] nothing more than a faddish term for the established concept of computers linked by networks. A cloud is water vapor. (Larry Ellison, co-founder and CEO, Oracle Corporation, September 2009)

The truth is rarely pure and never simple. (Oscar Wilde, The Importance of Being Earnest, 1895)

Thanks!

Davide.Salomoni@cnaf.infn.it

Further info

- WNoDeS Web: http://web.infn.it/wnodes
- Temi di ricerca: http://www.cnaf.infn.it/main/index.php/Link_Utili/Temi_di_ricerca