SaaS tutorial

CloudMIP platform

http://cloudmip.univ-tlse3.fr

Georges Da Costa

Thiebolt François

{dacosta,thiebolt}@irit.fr

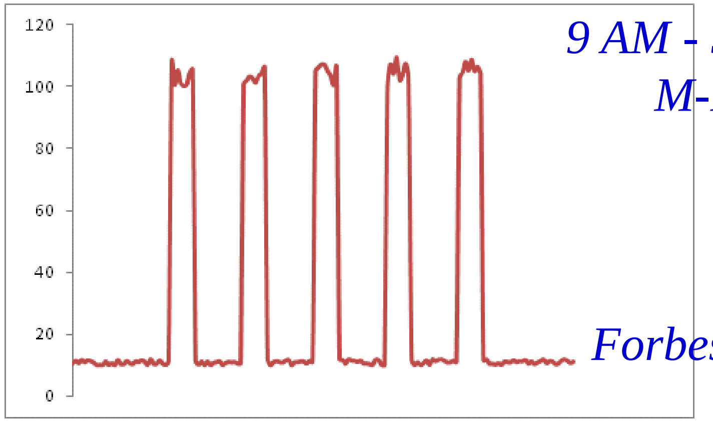
Plan

Part I - Principles

- Introduction to Clouds
- Infrastructure vision
- Cloud levels
- Standards
- Avantages / Disavantages
- Use cases

Why a new technology

- Large amount of data
  - Impossible to manage on a single computer
- Large amount of users
  - Workload varies a lot
- Grids are not always adapted
  - Must know final size
  - Start big
  - Be well-known
  - Expertise in software, hardware, administration
- How to start the next best service / web-site ?



Cloud vision of the solution

- Host the web site in a *Cloud*
- Provision new servers every day
    - unprovision them every night
- Pay just $0.10 per server per hour
- Let the cloud provider manage everything
    - Hardware
    - Administration!
- Just take care of the code and storage

Definition

- Cloud Computing is a general term used to describe a new class of network based computing that takes place over the Internet
- It is a collection/group of integrated and networked hardware, software and Internet infrastructure (called a platform)
- It uses the Internet for communication and provides hardware, software and networking services to clients.
- These platforms hide the complexity and details of the underlying infrastructure from users and applications by providing very simple graphical interface or API (Applications Programming Interface).

Same as Grid initial Goal!

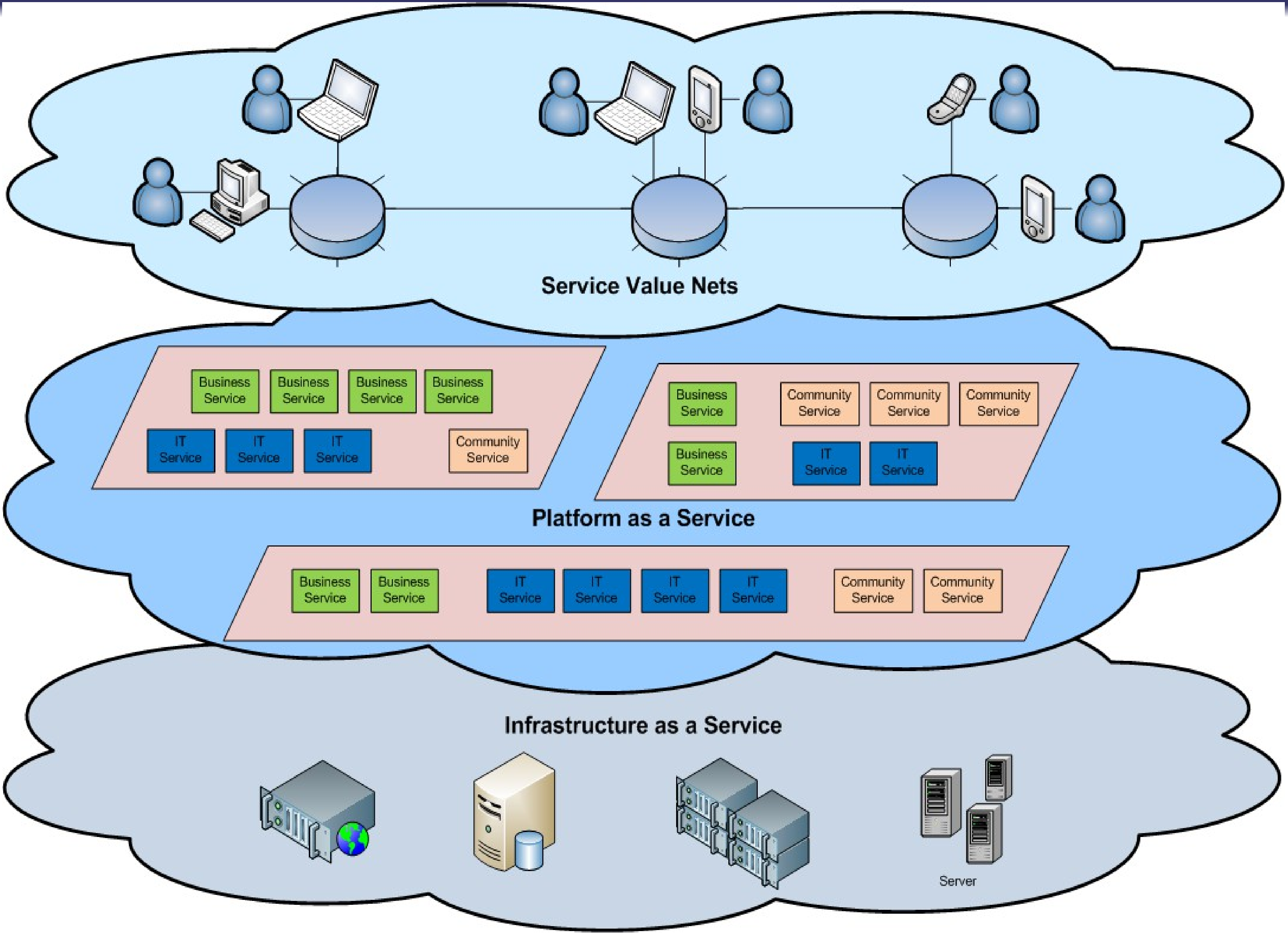
Some characteristics

- The platform provides
  - on demand services
  - always on, anywhere, anytime and any place.
- Pay for use and as needed, elastic
  - scale up and down in capacity and functionalities.
- Hardware and software services are available to the general public, enterprises, corporations and businesses markets.

Some characteristics

- Remotely hosted
  - Services or data are hosted on remote infrastructure.
- Ubiquitous
  - Services or data are available from anywhere.
- Commodified
  - The result is a utility computing model
  - Similar to gas and electricity

Architecture



Cloud (p)layers

- Application as a Service
  - MS Live/ExchangeLabs, IBM, Google Apps, Salesforce.com, Quicken Online, Zoho, Cisco
- Application Platform
  - Google App Engine, Mosso, Force.com, Engine Yard, Facebook, Heroku, AWS
- Server Platform
  - 3Tera, EC2, SliceHost,
  - GoGrid, RightScale, Linode
- Storage Platform
  - Amazon S3, Dell, Apple, ...

Service layers

Focus	Services	Description
Application Focused	Services	Services – Complete business services such as PayPal, OpenID, OAuth, Google Maps, Alexa
Application Focused	Application	Application – Cloud based software that eliminates the need for local installation such as Google Apps, Microsoft Online
Application Focused	Development	Development – Software development platforms used to build custom cloud based applications (PAAS & SAAS) such as SalesForce
Infrastructure Focused	Platform	Platform – Cloud based platforms, typically provided using virtualization, such as Amazon ECC, Sun Grid
Infrastructure Focused	Storage	Storage – Data storage or cloud based NAS such as CTERA, iDisk, CloudNAS
Infrastructure Focused	Hosting	Hosting – Physical data centers such as those run by IBM, HP, NaviSite, etc.

Basic Cloud Characteristics

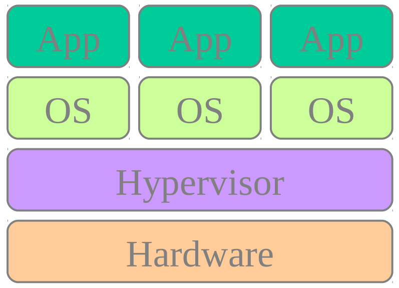
- The “**no-need-to-know**” in terms of the underlying details of infrastructure, applications interface with the infrastructure via the APIs.
- The “**flexibility and elasticity**” allows these systems to scale up and down at will – utilising the resources of all kinds (CPU, storage, server capacity, load balancing, and databases).
- The “**pay as much as used and needed**” type of utility computing and the “always on!, anywhere and any place” type of network-based computing.
- They are “**transparent**” as they can run whatever the underlying layers
- They are composed of classical servers and open-source software

SaaS: Software as a Service

- SaaS is a model of software deployment where an application is hosted as a service provided to customers across the Internet.
- SaaS is generally used to refer to business software rather than consumer software, falls under Web 2.0!
- By removing the need to install and run an application on a user’s own computer it is seen as a way for businesses to get the same benefits as commercial software with smaller cost outlay.
- Saas alleviates the burden of software maintenance/support, but users relinquish control over software versions and requirements.
- Terms that are used in this sphere include Platform as a Service (PaaS) and Infrastructure as a Service (IaaS).

Infrastructure :Virtualization

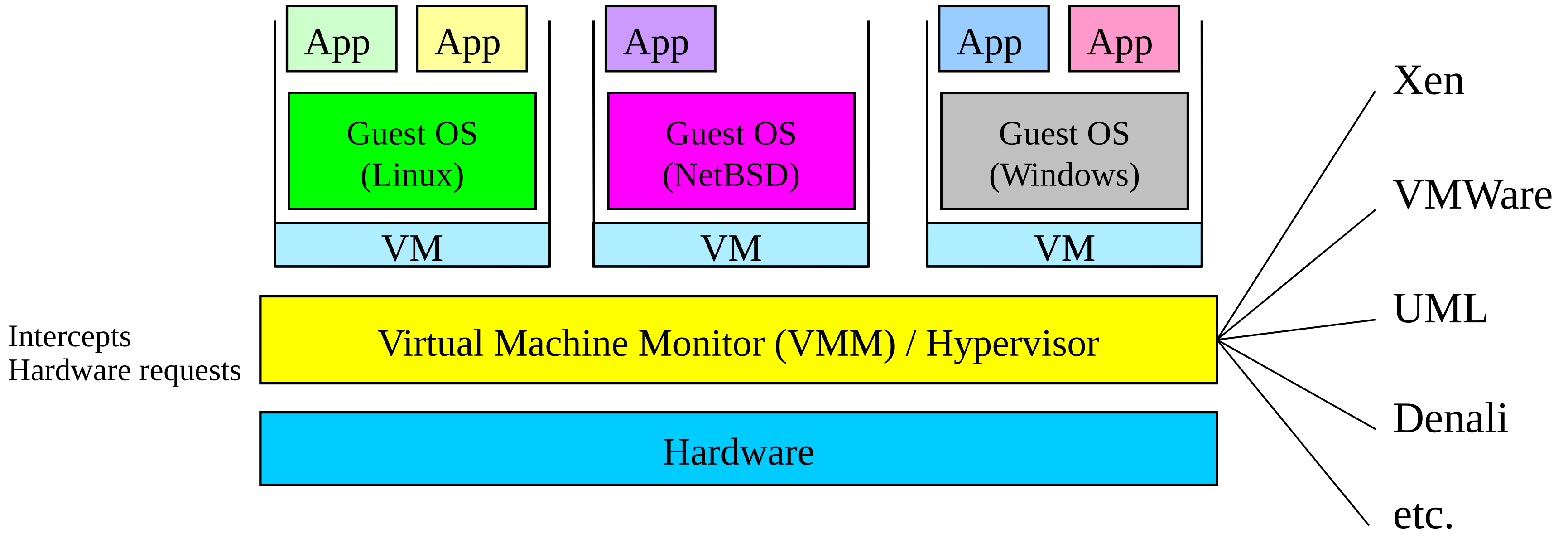
- Virtual workspaces:
  - An abstraction of an execution environment that can be made dynamically available to authorised clients by using well-defined protocols,
  - Resource quota (e.g. CPU, memory share),
  - Software configuration (e.g. O/S, provided services).
- Implemented on Virtual Machines (VMs):
  - Abstraction of a physical host machine,
  - Hypervisor intercepts and emulates instructions from VMs, and allows management of VMs,
  - VMWare, Xen, etc.
- Provide infrastructure API:
  - Plug-ins to hardware/support structures



Virtualized Stack

Virtual Machines

- VM technology allows multiple virtual machines to run on a single physical machine.



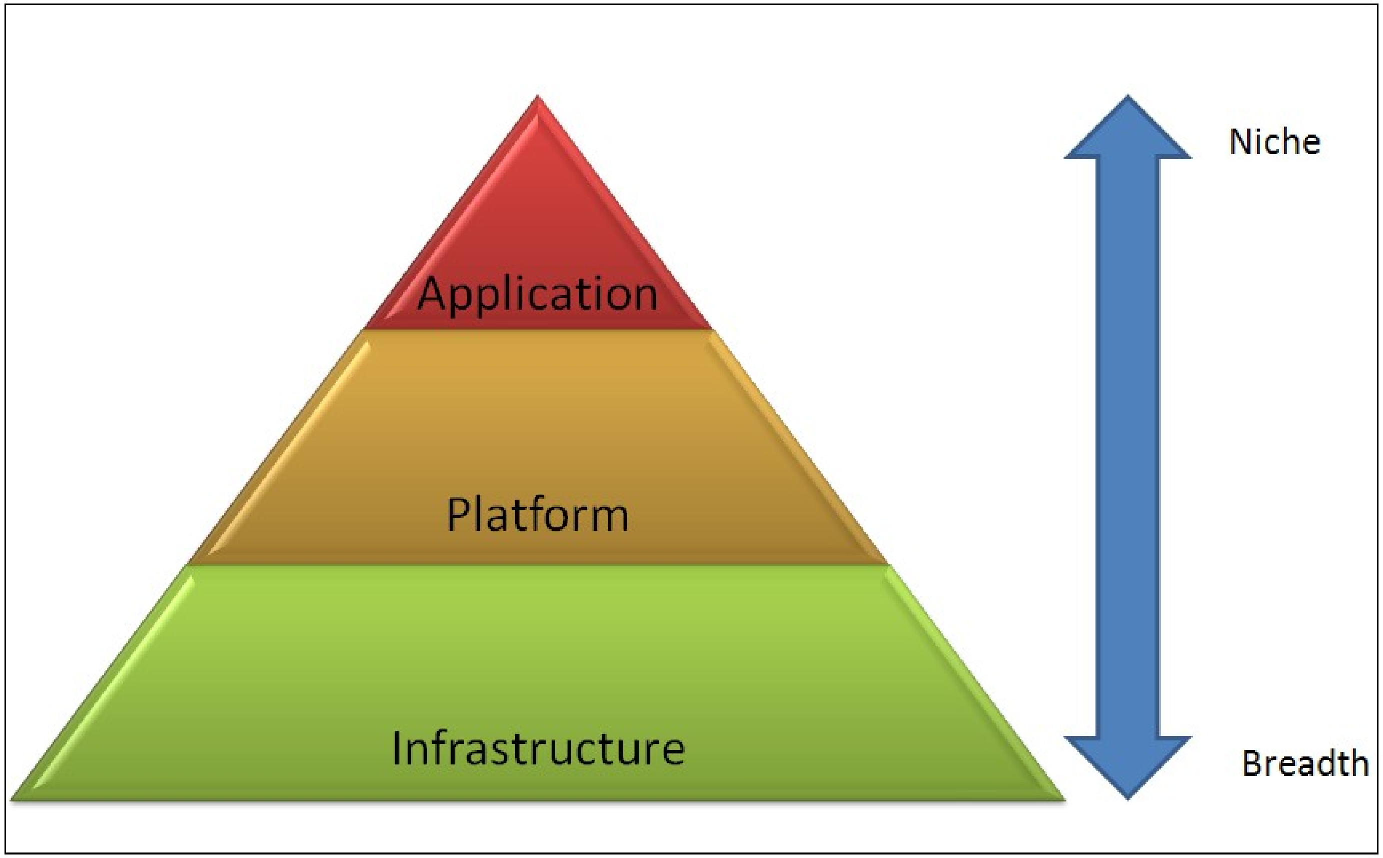
Performance: Para-virtualization (e.g. Xen, kvm) is very close to raw physical performance! Otherwise simulation-like which works but is slow

Virtualization in General

- Advantages of virtual machines:
  - Run operating systems where the physical hardware is unavailable,
  - Easier to create new machines, backup machines, etc.,
  - Software testing using “clean” installs of operating systems and software,
  - Emulate more machines than are physically available,
  - Timeshare lightly loaded systems on one host,
  - Debug problems (suspend and resume the problem machine),
  - Easy migration of virtual machines (shutdown needed or not).
  - Run legacy systems!
  - Not necessary to have one server for each service
  - Easier to manage increase of a service workload

Key Cloud concept : Multi-layers

- Application
  - build over
- Platform
  - build over
- Infrastructure



- Each level managed by different people

Three levels/Three points of view

SaaS
Software as a Service

PaaS
Platform as a Service

IaaS
Infrastructure as a Service

Three levels/Three points of view

SaaS
Software as a Service

- Software delivery model
  - No hardware or software to manage
  - Service delivered through a browser
- Advantage for users
  - Pay per use, updates free, trouble free, homogeneity
- Advantage for service provider
  - User lock-in, less fragmentation, recurrent fees

Three levels/Three points of view

- Consumer creates the software using tools and/or libraries from the provider
- The provider provides the networks, servers, storage and other services

PaaS

Platform as a Service

- Same benefits as SaaS
- Some vendors
  - Google App Engine, Mosso, AWS: S3, Windows Azure
- Some classical services : **Hadoop**, Data Base, Web Server,...

Three levels/Three points of view

- Offers ressources : Virtual machines, routers, servers, disks,...
- Same benefits as SaaS and PaaS
- Amazon EC2, Azure Services Platform, DynDNS, Google Compute Engine, HP Cloud, Rackspace Cloud, …
- Main open-source middlewares
  - OpenNebula, OpenStack

IaaS
Infrastructure as a Service

Cloud computing takes virtualization to the next step

Paas & Iaas as Tools !

- You don’t have to own the hardware
- You “rent” it as needed from a cloud
- There are public clouds
    - e.g. Amazon EC2, and now many others (Microsoft, IBM, Sun, and others ...)
- A company can create a private one
    - With more control over security, etc.
- Classical method : Hybrid cloud
    - Internal cloud for core functions
    - External cloud for absobing peak demand

Goal 1 – Cost Control

- Cost
  - Many systems have variable demands
    - Batch processing (e.g. New York Times)
    - Web sites with peaks (e.g. Forbes)
    - Startups with unknown demand (e.g. Instagram)
  - Reduce risk
    - Don't need to buy hardware until you need it

Goal 2 - Business Agility

- More than scalability - elasticity!
  - In big health care business
    - Used to take 3 - 4 months to give a department a server cluster, then they would keep it for themselves!
  - Using EC2, about 5 minutes!
    - And they give it back when they are done!
- Scaling back is as important as scaling up

Goal 3 - Stick to Our Business

- Most companies don't WANT to do system administration
    - Forbes says:
        - *We are a publishing company, not a software company*
- But beware:
    - Do you really save much on sys admin?
    - You don't have the hardware, but you still need to manage the OS!

How Cloud Computing Works

- Various providers let you create virtual servers
    - Set up an account, perhaps just with a credit card
- You create virtual servers ("virtualization")
    - Choose the OS and software each "instance" will have
    - It will run on a large server farm located somewhere
    - You can instantiate more on a few minutes' notice
    - You can shut down instances in a minute or so
- They send you a bill for what you use

Any Nasty Details? Yes !

- How do I pick a provider?
- Am I locked in to a provider?
- Where do I put my data?
- What happens to my data when I shut down?
- How do I log in to my server?
- How do I keep others from logging in (security)?
- How do I get an IP address?
- Etc.

What purpose and benefits?

- Classical companies and applications, are infrastructure dependent
  - Cloud computing is infrastructure-less.
- Cloud infrastructure is “pay as used and on demand”
  - Savings in capital and operational investment!
- Clients can:
  - Put their data on the platform instead of on their own desktop PCs and/or on their own servers.
  - They can put their applications on the cloud and use the servers within the cloud to do processing and data manipulations etc.
- It is :
  - transparent (users do not know what is behind the scenes!)
  - highly scalable (scale up and down as needed)
  - on-demand, pay as needed and as used.

Cloud-Sourcing

- Why is it becoming a Big Deal:
  - Using high-scale/low-cost providers,
  - Any time/place access via web browser,
  - Rapid scalability; incremental cost and load sharing,
  - Can forget need to focus on local IT.
- Concerns:
  - Performance, reliability, and SLAs,
  - Control of data, and service parameters,
  - Application features and choices,
  - Interaction between Cloud providers,
  - No standard API – mix of SOAP and REST!
  - Privacy, security, compliance, trust…

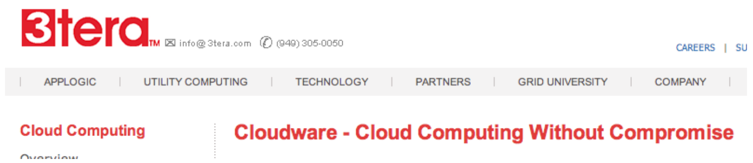
CloudMIP

Commercial Cloud Offerings

amazon webservices™ Amazon Elastic Compute Cloud (Amazon EC2) - *Beta*

GOGRID beta A ServePath Company

Sun microsystems TAP INTO THE POWER OF NETWORK.COM


3tera™ info@3tera.com (949) 305-0050 CAREERS | SU

APPLOGIC | UTILITY COMPUTING | TECHNOLOGY | PARTNERS | GRID UNIVERSITY | COMPANY

Cloud Computing

Cloudware - Cloud Computing Without Compromise

b-hive

MOSSO the hosting cloud

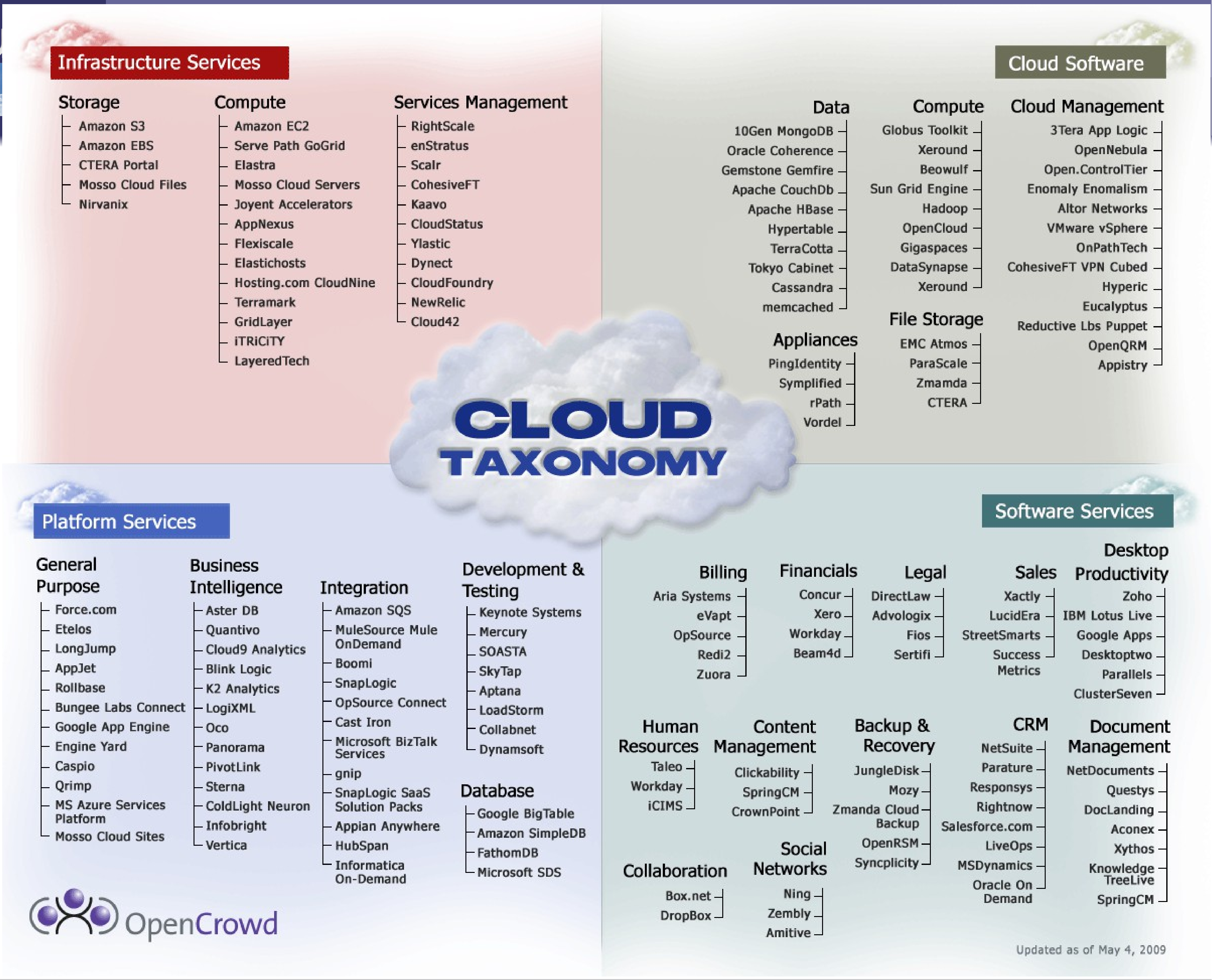
Engine Yard™

Joyent

VERIO An NTT Communications Company

- Problem of vendor lock-in

Cloud Taxonomy



CLOUD TAXONOMY

Infrastructure Services

Storage
- Amazon S3
- Amazon EBS
- CTERA Portal
- Mosso Cloud Files
- Nirvanix

Compute
- Amazon EC2
- Serve Path GoGrid
- Elastra
- Mosso Cloud Servers
- Joyent Accelerators
- AppNexus
- Flexiscale
- Elastichosts
- Hosting.com CloudNine
- Terramark
- GridLayer
- iTRiCiTY
- LayeredTech

Services Management
- RightScale
- enStratus
- Scalr
- CohesiveFT
- Kaavo
- CloudStatus
- Ylastic
- Dynect
- CloudFoundry
- NewRelic
- Cloud42

Cloud Software

Data
- 10Gen MongoDB
- Oracle Coherence
- Gemstone Gemfire
- Apache CouchDb
- Apache HBase
- Hypertable
- TerraCotta
- Tokyo Cabinet
- Cassandra
- memcached

Compute
- Globus Toolkit
- Xeround
- Beowulf
- Sun Grid Engine
- Hadoop
- OpenCloud
- Gigaspaces
- DataSynapse
- Xeround

Cloud Management
- 3Tera App Logic
- OpenNebula
- Open.ControlTier
- Enomaly Enomalism
- Altor Networks
- VMware vSphere
- OnPathTech
- CohesiveFT VPN Cubed
- Hyperic
- Eucalyptus
- Reductive Lbs Puppet
- OpenQRM
- Appistry

Appliances
- PingIdentity
- Symplified
- rPath
- Vordel

File Storage
- EMC Atmos
- ParaScale
- Zmamda
- CTERA

Platform Services

General Purpose
- Force.com
- Etelos
- LongJump
- AppJet
- Rollbase
- Bungee Labs Connect
- Google App Engine
- Engine Yard
- Caspio
- Qrimp
- MS Azure Services Platform
- Mosso Cloud Sites

Business Intelligence
- Aster DB
- Quantivo
- Cloud9 Analytics
- Blink Logic
- K2 Analytics
- LogiXML
- Oco
- Panorama
- PivotLink
- Sterna
- ColdLight Neuron
- Infobright
- Vertica

Integration
- Amazon SQS
- MuleSource Mule OnDemand
- Boomi
- SnapLogic
- OpSource Connect
- Cast Iron
- Microsoft BizTalk Services
- gnip
- SnapLogic SaaS Solution Packs
- Appian Anywhere
- HubSpan
- Informatica On-Demand

Development & Testing
- Keynote Systems
- Mercury
- SOASTA
- SkyTap
- Aptana
- LoadStorm
- Collabnet
- Dynamsoft

Database
- Google BigTable
- Amazon SimpleDB
- FathomDB
- Microsoft SDS

OpenCrowd

Software Services

Billing
- Aria Systems
- eVapt
- OpSource
- Redi2
- Zuora

Financials
- Concur
- Xero
- Workday
- Beam4d

Legal
- DirectLaw
- Advologix
- Fios
- Sertifi

Sales
- Xactly
- LucidEra
- StreetSmarts
- Success Metrics

Desktop Productivity
- Zoho
- IBM Lotus Live
- Google Apps
- Desktoptwo
- Parallels
- ClusterSeven

Human Resources
- Taleo
- Workday
- iCIMS

Content Management
- Clickability
- SpringCM
- CrownPoint

Backup & Recovery
- JungleDisk
- Mozy
- Zmanda Cloud Backup
- OpenRSM
- Syncplicity

CRM
- NetSuite
- Parature
- Responsys
- Rightnow
- Salesforce.com
- LiveOps
- MSDynamics
- Oracle On Demand

Document Management
- NetDocuments
- Questys
- DocLanding
- Aconex
- Xythos
- Knowledge TreeLive
- SpringCM

Collaboration
- Box.net
- DropBox

Social Networks
- Ning
- Zembly
- Amitive

Updated as of May 4, 2009

Cloud Storage

- Several large Web companies (such as Amazon and Google) have plenty of unused storage
  - Rent it to others
- Cloud storage
  - data are stored remotely
  - data are temporarily cached on desktop computers, mobile phones or other Internet-linked devices.
- Classical exemple :
  - Amazon’s Elastic Compute Cloud (EC2)
  - Simple Storage Solution (S3)

Amazon Simple Storage Service (S3)

- Unlimited Storage.
- Pay for what you use:
  - $0.20 per GByte of data transferred,
  - $0.15 per GByte-Month for storage used,
  - Second Life Update:
    - 1TBytes, 40,000 downloads in 24 hours - $200,

Utility Computing – EC2

- Amazon Elastic Compute Cloud (EC2):
  - Elastic, marshal 1 to 100+ PCs,
  - Machine Specs…,
  - Fairly cheap!
  - Powered by Xen – a Virtual Machine:
  - Different from Vmware and VPC as uses “para-virtualization” where the guest OS is modified to use special hyper-calls:
  - Hardware contributions by Intel (VT-x/Vanderpool) and AMD (AMD-V).
  - Supports “Live Migration” of a virtual machine between hosts.
- Linux, Windows, OpenSolaris
- Management Console/AP

EC2 – The Basics

- Load your image onto S3 and register it.
- Boot your image from the Web Service.
- Open up required ports for your image.
- Connect to your image through SSH.
- Execute you application…

Amazon Elastic MapReduce

- Hadoop: java framework
- Distributed manipulation of large amounts of data
- Data Intensive distributed applications.
- Amazon EC2 + Amazon S3 (provided as SaaS).
- Use cases:
  - Web Indexing.
  - Data Mining.
  - Machine Learning.
  - Financial Analysis.
  - Scientific Simulation.

Opportunities and Challenges

- The use of the cloud provides a number of opportunities:
  - It enables services to be used without any understanding of their infrastructure.
    - Cloud computing works using economies of scale:
    - It potentially lowers the outlay expense for start up companies, as they would no longer need to buy their own software or servers.
    - Cost would be by on-demand pricing.
    - Vendors and Service providers claim costs by establishing an ongoing revenue stream.
  - Data and services are stored remotely but accessible from “anywhere”.

Opportunities and Challenges

- Dependence on others that could possibly limit flexibility and innovation:
  - The others are likely become the bigger Internet companies like Google and IBM, who may monopolise the market.
  - Some argue that this use of supercomputers is a return to the time of mainframe computing that the PC was a reaction against.
- Security could prove to be a big issue:
  - It is still unclear how safe out-sourced data is and when using these services ownership of data is not always clear.
- There are also issues relating to policy and access:
  - If your data is stored abroad whose policy do you adhere to?
  - What happens if the remote server goes down?
  - How will you then access files?
  - There have been cases of users being locked out of accounts and losing access to data.

Advantages of Cloud Computing

- Lower computer costs:
  - You do not need a high-powered and high-priced computer to run cloud computing's web-based applications.
  - Since applications run in the cloud, not on the desktop PC, your desktop PC does not need the processing power or hard disk space demanded by traditional desktop software.
  - When you are using web-based applications, your PC can be less expensive, with a smaller hard disk, less memory, more energy efficient processor...
  - In fact, your PC in this scenario does not even need a CD or DVD drive, as no software programs have to be loaded and no document files need to be saved.

Advantages of Cloud Computing

- Improved performance:
  - Fewer large programs in local means better performance
    - Less memory used, same for CPU,...
  - Computers in a cloud computing system boot and run faster because they have fewer programs and processes loaded into memory…
- Reduced software costs:
  - Instead of purchasing expensive software applications, you can get most of what you need for free-ish!
  - most cloud computing applications are free today, such as the Google Docs suite.
  - better than paying for similar commercial software
  - which alone may be justification for switching to cloud applications.

Advantages of Cloud Computing

- Instant software updates:
  - No more choice between obsolete software and high upgrade costs.
  - For web-based applications, updates are automatic
    - available the next time you log into the cloud.
  - For web-based application, you log-in to the latest version
    - without needing to pay for or *download* an upgrade.
- Improved document format compatibility.
  - No more worry about the compatibilty of your documents with other users' applications or OSes
  - No more format incompatibilities when is sharing documents and applications in the cloud.

Advantages of Cloud Computing

- Unlimited storage capacity:
  - Cloud computing offers virtually limitless storage.
  - 1 Tbyte hard drive compared to Pbytes in the cloud.
- Increased data reliability:
  - Unlike desktop computing, in which if a hard disk crashes and destroy all your valuable data, a computer crashing in the cloud should not affect the storage of your data.
    - if your personal computer crashes, all your data is still out there in the cloud, still accessible
  - In a world where few individual desktop PC users back up their data on a regular basis, cloud computing is a data-safe computing platform!

Advantages of Cloud Computing

- Universal document access:
  - Documents are not linked to a physical object
  - All you need is a computer and an Internet connection
  - Documents are instantly available from wherever you are
- Latest version availability:
  - When you edit a document at home, that edited version is what you see when you access the document at work.
  - The cloud always hosts the latest version of your documents
  - as long as you are connected, you are not in danger of having an outdated version

Advantages of Cloud Computing

- Easier group collaboration:
  - Sharing documents leads directly to better collaboration.
  - Many users do this as it is an important advantages of cloud computing
    - multiple users can collaborate easily on documents and projects
- Device independence.
  - You are no longer tethered to a single computer or network.
  - Changes to computers, applications and documents follow you through the cloud.
  - Move to a portable device, and your applications and documents are still available.

Disadvantages

- Requires a constant Internet connection:
  - Cloud computing is impossible if you cannot connect to the Internet.
  - Since you use the Internet to connect to both your applications and documents, if you do not have an Internet connection you cannot access anything, even your own documents.
  - A dead Internet connection means no work and in areas where Internet connections are few or inherently unreliable, this could be a deal-breaker.

Disadvantages

- Does not work well with low-speed connections:
  - Similarly, a low-speed Internet connection, such as that found with dial-up services, makes cloud computing painful at best and often impossible. ADSL/3G is the minimum.
  - Web-based applications require a lot of bandwidth to download, as do large documents.
- Features might be limited:
  - This situation is bound to change, but today many web-based applications simply are not as full-featured as their desktop-based applications.
  - For example, you can do a lot more with Microsoft PowerPoint than with Office web Apps offering

Disadvantages

- Can be slow:
  - Even with a fast connection, web-based applications can sometimes be slower than accessing a similar software program on your desktop PC.
  - Everything about the program, from the interface to the current document, has to be sent back and forth from your computer to the computers in the cloud.
  - If the cloud servers happen to be backed up at that moment, or if the Internet is having a slow day, you would not get the instantaneous access you might expect from desktop applications.
  - Changing fast due to improved javascript runtime and new paradigms (Hadoop)

Disadvantages

- Stored data might not be secure:
  - With cloud computing, all your data is stored on the cloud.
    - The questions is How secure is the cloud?
  - Can unauthorised users gain access to your confidential data?
    - Patriot Act ?
- Stored data can be lost:
  - Theoretically, data stored in the cloud is safe, replicated across multiple machines.
  - But on the off chance that your data goes missing, you have no physical or local backup.
    - Put simply, relying on the cloud puts you at risk if the cloud lets you down. Also you must trust the Cloud.

Disadvantages

- HPC Systems:
  - Difficult to run HPC applications on MPI/OpenMP!
  - Scheduling is important with this type of application
    - as you want all the VM to be co-located to minimize communication latency!
- General Concerns:
  - Each cloud systems uses different protocols and different APIs
    - may not be possible to run applications between cloud based systems
  - Amazon has created its own DB system (not SQL), and workflow system (many popular workflow systems out there)
    - Normal applications need to be adapted

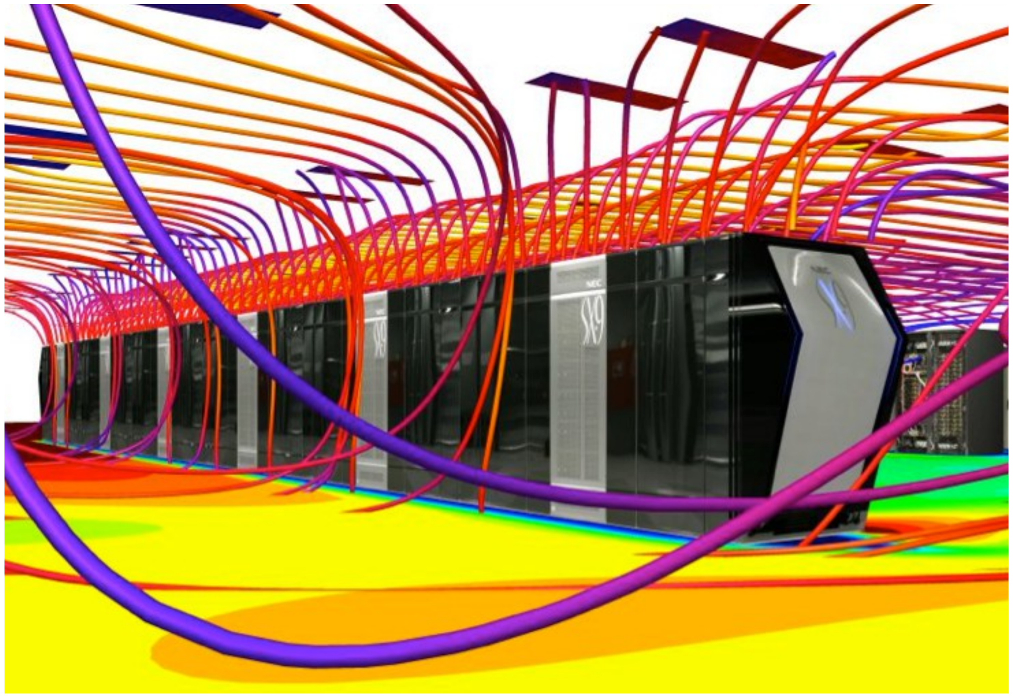
Clouds, a glimpse of Future ?

- No real novelty in Cloud computing ?
  - More merging of centralised and distributed ideas
  - The Grid idea made Reality !
- However some big concerns for users !
- Many new open source systems
  - Can run on local cluster
  - Can merge local cluster and outside clouds
- Next step : make it easy to manage
  - Currently most adaptation is done by hand !
  - What is needed is smart SaaS !

SaaS use case : CFD

- European FP7 project that tackles Energy efficiency in Data Centers.
- Poland, UK, France, Germany, Spain
  - High needs in CFD Simulation
- Thermal model of a data-center

- ComputeBox blueprints
- 3D Models and thermal Models
  - CFD on the Cloud

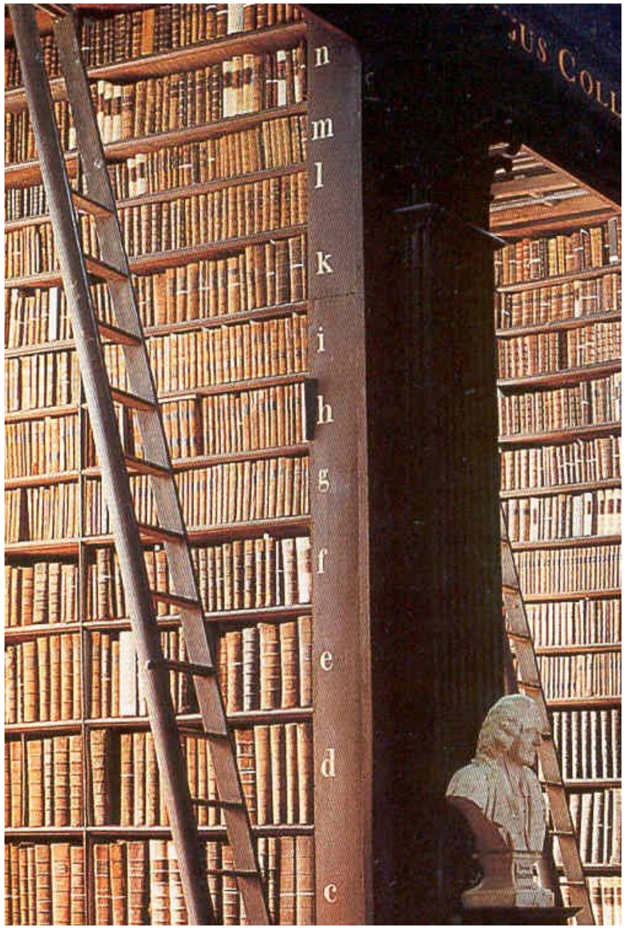


CHRISTMANN

Fine grained description using complex 3D models of a whole Data Center for CFD simulation. This encompasses CRACs, racks and Compute Box made of Computer On Module boards including fans, heaters ...

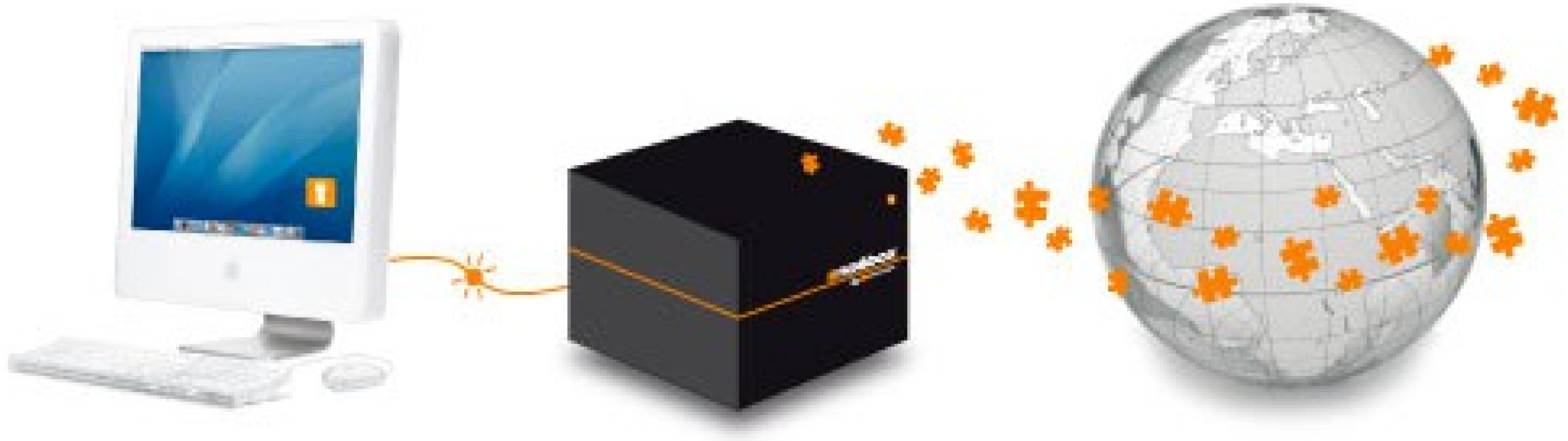
Saas use case: Library

- Complex yet simple
  - Library managers are not CS
  - Large corpus of references
  - Data mining is required
- Shared information between libraries
  - Open data, interoperability



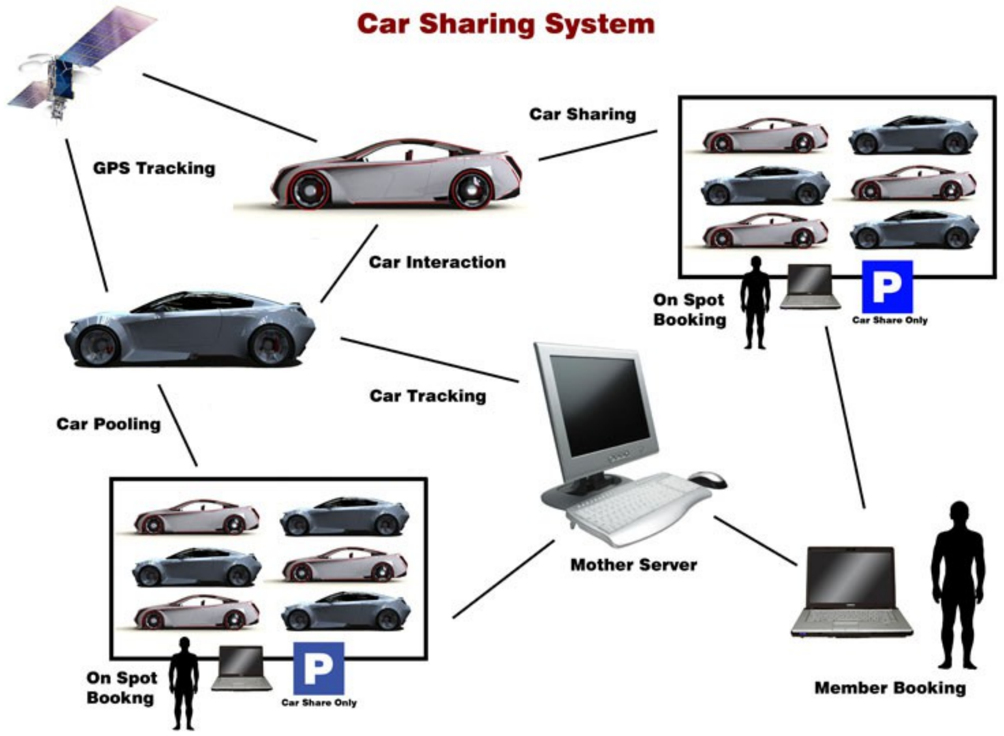
Saas use case : Saves

- Distributed storage : Dropbox, Box.net, Sparkleshare
- Rely on several services out of their scope
  - Identification
  - Raw storage
  - Workload balancing, localization



Saas use case : car sharing

- Low performance, high security
  - Cost sharing
  - Identity management
- High performance
  - Localisation
  - Path finding



Plan

Part II – OpenNebula / CloudMIP

- CloudMIP: a short overview,
- OpenNebula: APIs,
- Instantiation,
- Contextualisation,
- Rights management,
- Work in progress: GreenIT & others,
- France-Grilles operations.

Trends in France

- Movement toward SaaS in France
  - CNRS proposes services instead of funding servers (Core-Cloud)
    - Computing services
    - Storage services
    - Database and BigData services
    - Core services (mail,...)
    - Still managed by hand
  - Toulouse University moved/moves several services to the Cloud and plan to continue
    - Mail, web but also chemical computing
  - IRIT laboratory investigate high density plateforms for energy-efficient virtualization

The CloudMIP platform

Facts and resources

- Who : Pr Jean-Marc Pierson (manager), Dr François Thiebolt (system), DTSI Network team,
- Where : hosted at the Paul Sabatier University's Data Center (along with Grid5000-Toulouse and GridMIP platforms),
- Taskforce : 1 Dr-enginner (40%), 1 software engineer (30%),
- Fluids consumption annual cost (est.) : between 4k€ and 10k€.

2 x Dell M1000e 32 blades

Hardware, system, middleware ...

- 2 x Dell M1000e chassis each filled with 16 blades ⇨ 256 **physical** cores, 1TB RAM, 15To Disk
- System : Scientific Linux 6.4 x86_64,
- VM provisioning system : **OpenNebula** 4.2 (spice protocol support) with **KVM** hypervisor,
- Monitoring : Zabbix (http://www.zabbix.com) ---*combine Nagios and Ganglia capabilities.*

+ power monitoring **+** seconds to launch a hundred of multi-Gb VMs **+** open-source software

The CloudMIP platform

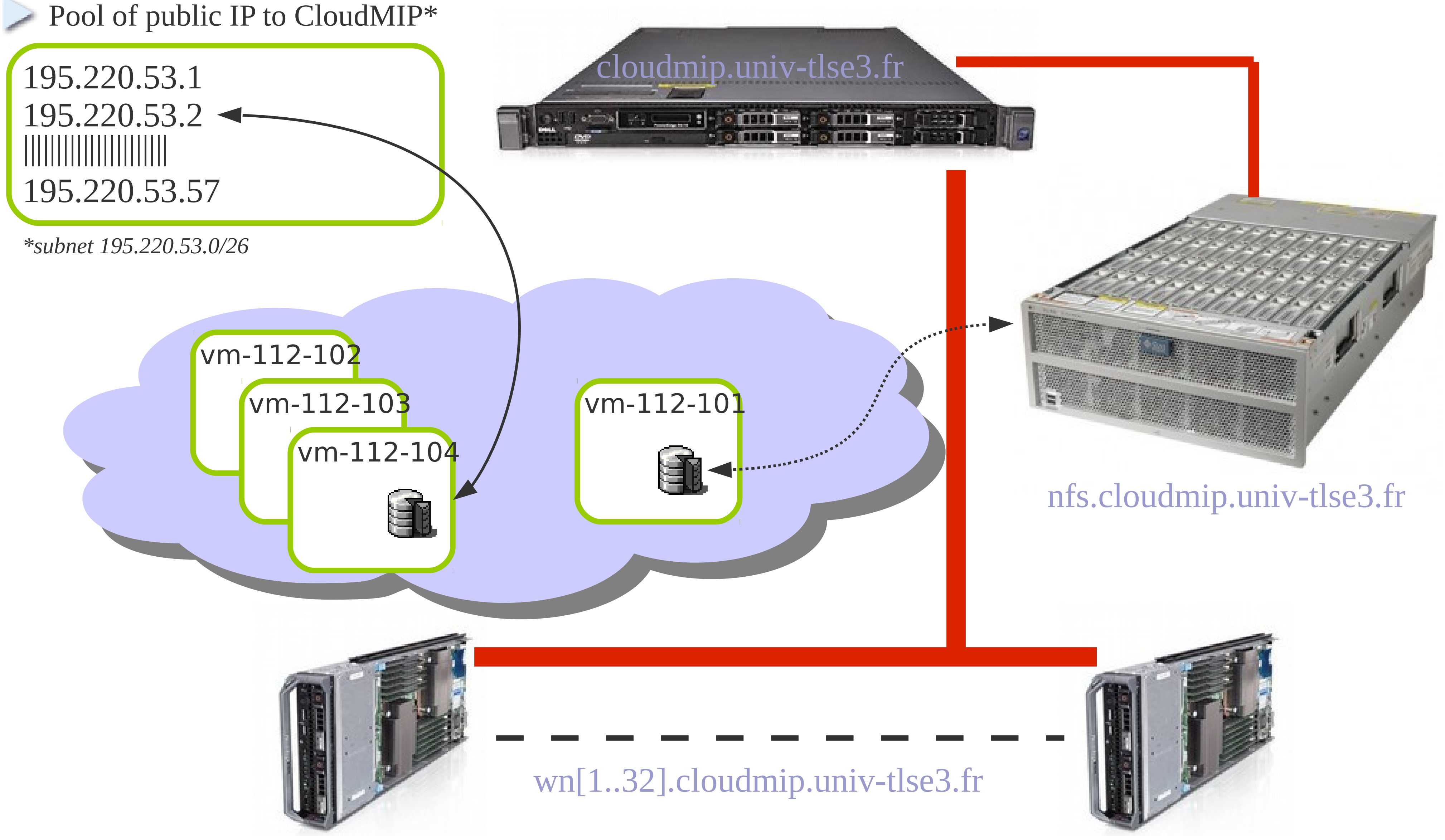
… additional details

- The front-end node : http://cloudmip.univ-tlse3.fr/,
- The monitoring system : cloudmip.univ-tlse3.fr/zabbix*,
- 32 blades (8 cores @ 2.4Ghz, 32GB ram, 2 x 146GB SAS 15ktpm RAID0),
  - ⇨ means upto 256 Amazon EC2 M1 instance (1 **physical** dedicated CPU and 4GB ram)
- OpenNebula 4.2 (KVM) with Qcow2 delta images to speedup deployment,
  - ⇨ a hundred of VMs in just a few seconds :)
- 1s resolution **power** and **thermal monitoring** of each node (Zabbix),
- A 24TB, 700MB/s NFS server shared with Grid5000 and GridMIP,
- Ways for the users to gain access to their VMs from the Internet :
  - ssh, vpn, spice display forwarding** (**done**),
  - #1000 ports on the front-end node dedicated to routing (**done**),
  - #60 dedicated public IPs with dynamic routing (**done**).

** login : green, passwd : cloudmip*

The CloudMIP platform

- Pool of public IP to CloudMIP*



*subnet 195.220.53.0/26

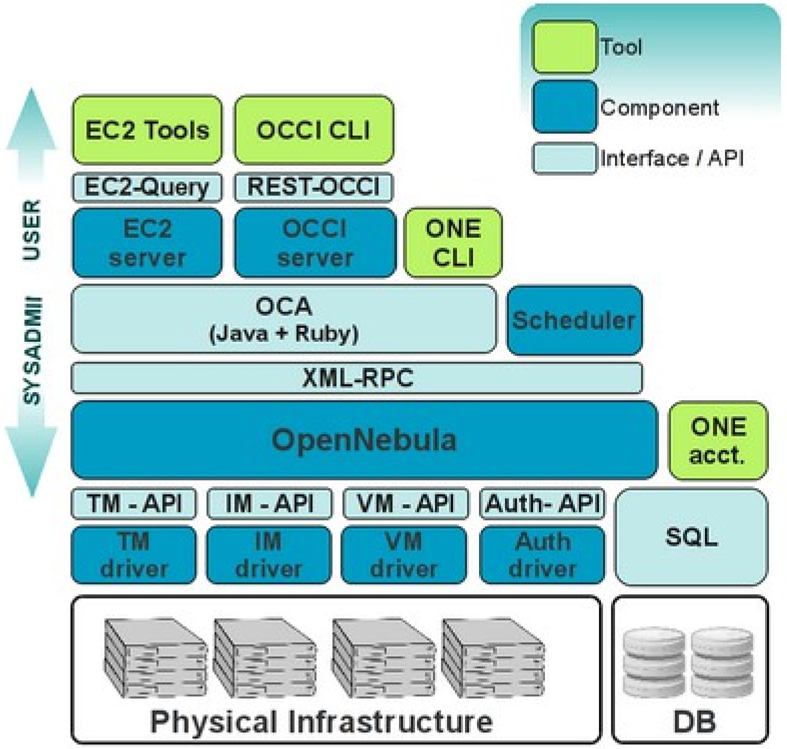
OpenNebula APIs

Various ways to interact with OpenNebula:

- XML-RPC / OpenNebula API,
- OCCI http://occi-wg.org or rOCCI (Ruby OCCI),
- Amazon EC2 (AWS).

OCCI and *EC2* interfaces are used for compatibility
• Allow usage of classical Cloud tools
• Independant of the exact OpenNebula version
• Does not provide access to all OpenNebula API
• Built over the XML-RPC interface

XML-RPC interface let tools access all API
• Lower cost and more precise
• Need some tweaks when OpenNebula version changes



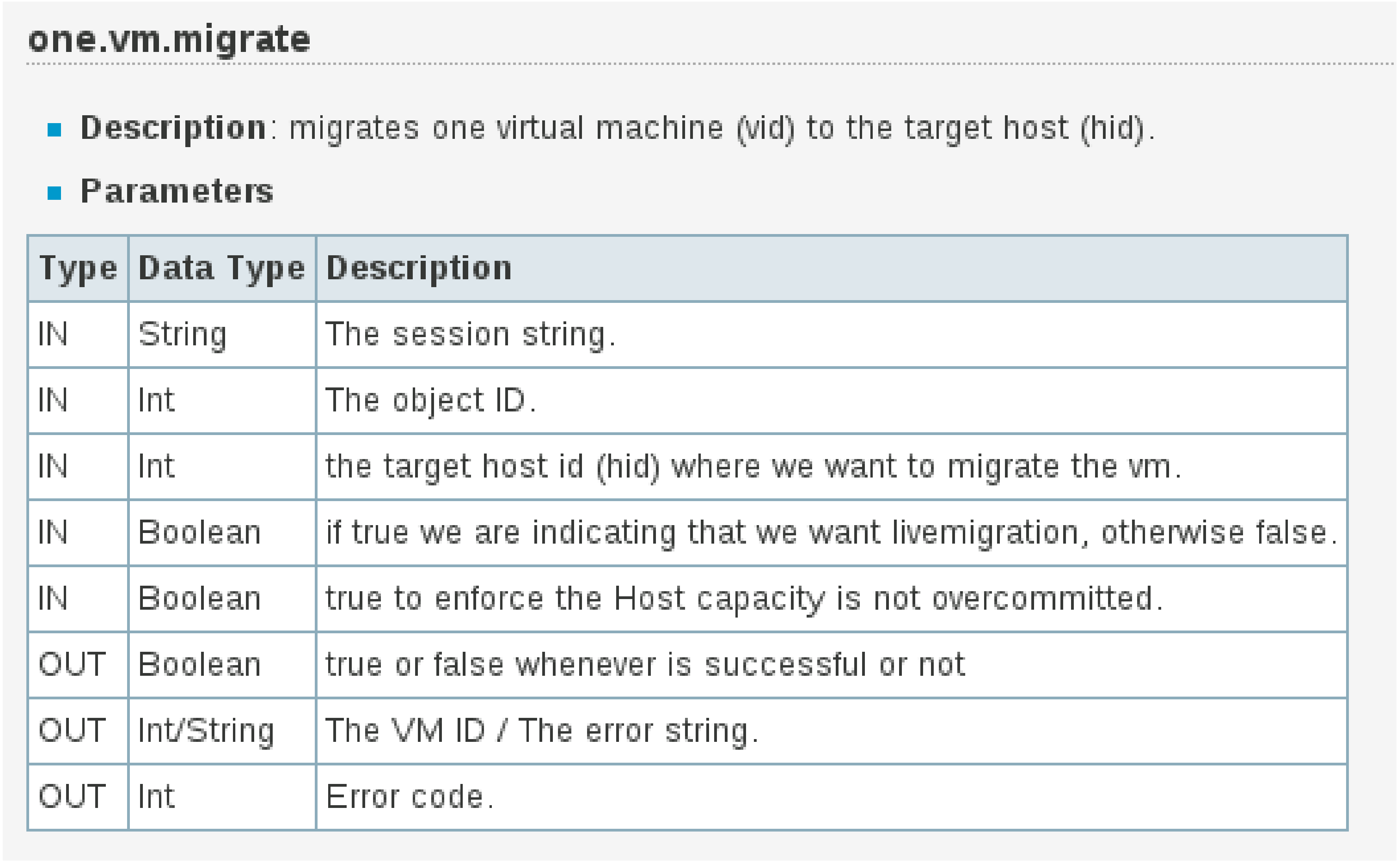
Note: The new CIMI standard from DMTF addresses several of the issues OCCI suffers (no embedded authentication, no multiple VM instantiation) … but lacks implementation right now.

Python OpenNebula Exemple

```
#!/usr/bin/python
import xmlrpclib, os, sys
# ---Start xmlrpc client to opennebula server
server = xmlrpclib.ServerProxy('http://localhost:2633/RPC2')
new_vm = server.one.template.instantiate('login:passwd', 0, '','')
resource_id  =  new_vm[1]
print server.one.vm.migrate('login:passwd', resource_id, 160, True, False)
```

- Usually with the canevas :
  - (Success, Result, ErrorCode) = server.one.....
    - Success : boolean
    - Result : string
    - ErrorCode : integer

OpenNebual API



one.vm.migrate

- **Description**: migrates one virtual machine (vid) to the target host (hid).
- **Parameters**

Type	Data Type	Description
IN	String	The session string.
IN	Int	The object ID.
IN	Int	the target host id (hid) where we want to migrate the vm.
IN	Boolean	if true we are indicating that we want livemigration, otherwise false.
IN	Boolean	true to enforce the Host capacity is not overcommitted.
OUT	Boolean	true or false whenever is successful or not
OUT	Int/String	The VM ID / The error string.
OUT	Int	Error code.

http://opennebula.org/documentation:rel4.2:api

Instantiation

VM instantiation using commands of OpenNebula or its API:

- **CLI** #> onetemplate instantiate <template_ID>
- **XML-Rpc** server.one.template.instantiate('login:passwd', template_ID, ",")

... but instantiation come along with contextualisation:

- Instantiation @ IaaS level → VM's template contextualisation ought to contain at least SSH_KEY for root user (~/.ssh/authorized_keys),
- Instantiation @ SaaS level → additional contextualisation is needed to start an apache service for example and to make the workload dispatcher @ front-node aware of this new VM supporting the desired application.

Apache libcloud: abstraction of different Cloud API providers
This leverage the needs for a heterogeneous cloud instantiation mechanism like in the France-Grilles intercloud platform.

Contextualisation

The OpenNebula way:

- First: VMs awareness of the contextualisation process

To make a VM aware of the contextualisation, we ought to install a RPM (RHEL6x compatible Vms):

```
#> oneimage persistent <image ID>
#> onevm create <options> <image ID>
#> ssh root@<IP new VM>
[VM]#> rpm -ivh http://dev.opennebula.org/<path>/one-context_4.2.1.rpm
[VM]#> exit
#> onevm <VMID> shutdown
#> onevm nonpersistent <image ID>
```

**http://dev.opennebula.org/attachments/download/722/one-context_4.2.1.rpm*

- Contextualisation section within the template file

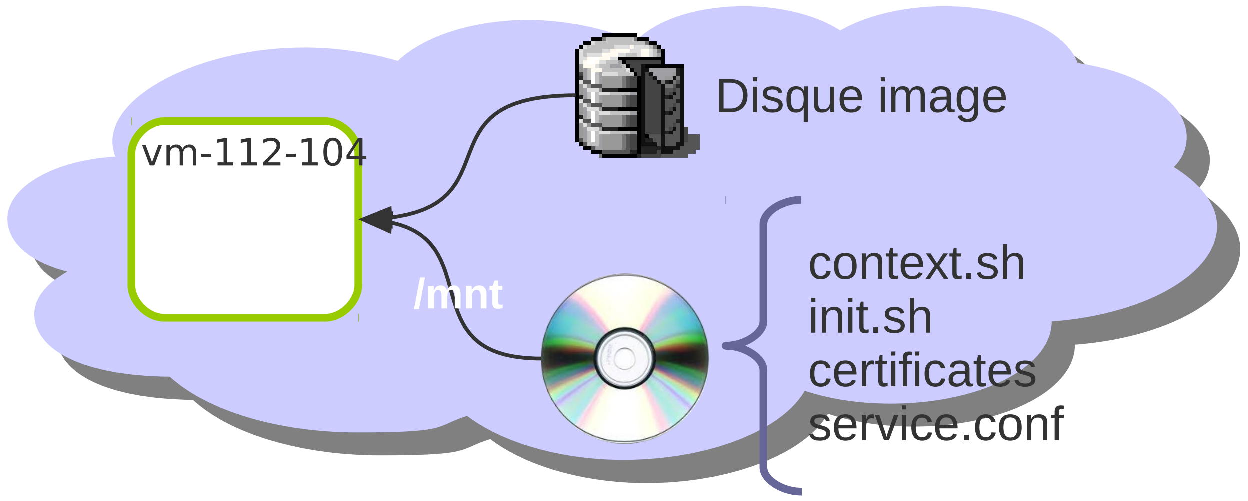
```
NAME    = SL64v2
CPU="1"
MEMORY="2048"
VCPU="1"
… … … …
CONTEXT=[
     SSH_PUBLIC_KEY="$USER[SSH_PUBLIC_KEY]" ]
… … … …
```

Note: We never use Network contextualisation because IP allocation and DNS registration is already handled by our CloudMIP DHCP server in co-ordination with OpenNebula.

This will add at least users' public key to the root's authorized_keys.

Contextualisation (next)

- @Instantiation, OpenNebula create a CDROM image with several init files.



- context: file that contains configuration variables, filled by OpenNebula with the parameters specified in the template file,
- init.sh: script called by VM at start that will configure specific services for this VM instance,
- certificates: directory that contains certificates for some service,
- service.conf: service configuration.

- @VM boot, ISO image is mounted within rootFS and content is exploited by previously installed contextualisation-aware RPM package (`vmcontext service`).

Contextualisation (next)

```
NAME    = SL64v2

CPU="1"
MEMORY="2048"
VCPU="1"

DISK=[
  BUS="virtio",
  DRIVER="qcow2",
  IMAGE="SL64v2",
  IMAGE_UNAME = "admin" ]
NIC=[
  MODEL="virtio",
  NETWORK="production",
  NETWORK_UNAME="admin" ]
OS=[
  ARCH="x86_64",
  BOOT="hd",
  MACHINE="rhel6.4.0" ]
GRAPHICS=[
  TYPE="spice",
#  PORT    = "5900",
  LISTEN  = "0.0.0.0" ]
CONTEXT=[
        SSH_PUBLIC_KEY="$USER[SSH_PUBLIC_KEY]" ]
# Console (don't forget to pass the option 'console=ttyS0' to the kernel args)
RAW = [
        type = "kvm",
        data = "<devices>
                <serial type='pty'>
                        <source path='/dev/pts/1'/>
                        <target port='0'/>
                        <alias name='serial0'/>
                </serial>
                <console type='pty' tty='/dev/pts/1'>
                        <source path='/dev/pts/1'/>
                        <target type='serial' port='0'/>
                        <alias name='serial0'/>
                </console>
        </devices>"  ]
```

Sample template file from the CloudMIP platform with simple contextualisation for VM instantiation @ IaaS level.

The USER contextualisation variable "SSH_PUBLIC_KEY" comes from current user templates variables.

```
#> oneuser list
ID NAME       GROUP    AUTH     ......
 3 francois   users    core     ......
#> oneuser show 3
USER 3 INFORMATION
ID                : 3
NAME              : francois
GROUP             : users
PASSWORD          : <encrypted passwd>
AUTH_DRIVER       : core
ENABLED           : Yes

USER TEMPLATE
SSH_PUBLIC_KEY="ssh-rsa <bla bla bla>"
TOKEN_PASSWORD="fc65cef9269c08719ba0d8bc04a..."
```

Contextualisation (next)

- Start VM

```
#> onevm create <template file>
ID: 595
#> onevmgetip*
  ID USER    GROUP  NAME   STAT UCPU  UMEM HOST TIME        IPv4
595 francois users      SL64v2  runn   0         0K      wn28   0d 00h00  172.28.112.101 (vm-112-101)
#> ssh root@vm-112-101
```

** custom command of the CloudMIP platform to ease IP retrieval of the VM*

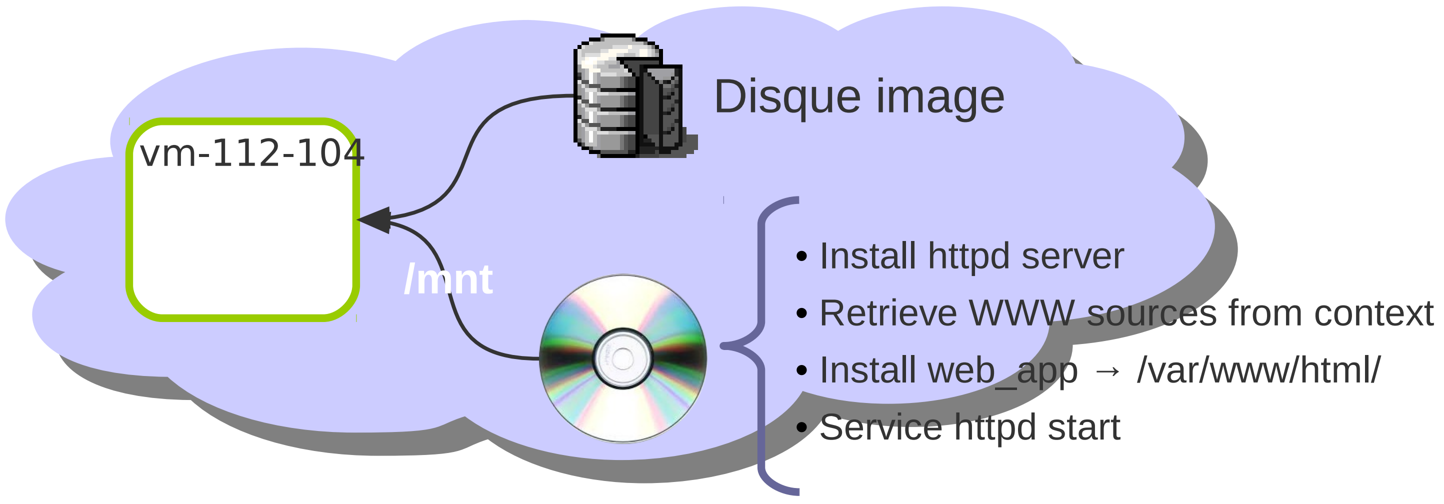
- [Addon]*To setup additional template variables @ user level*

```
#> cat oneuser_addon.one
SSH_PUBLIC_KEY="ssh-rsa <public key> ==francois@cloudmip.univ-tlse3.fr"
<additional variables>
#> oneuser update <user ID> update oneuser_addon.one -a
```

Contextualisation (next)

SaaS level contextualisation

When it's time for the workload repartition mechanism associated with a web application to start a new instance ⇨ VM contextualisation @ application level


vm-112-104
Disque image
/mnt
• Install httpd server
• Retrieve WWW sources from context
• Install web_app → /var/www/html/
• Service httpd start

Contextualisation (next)

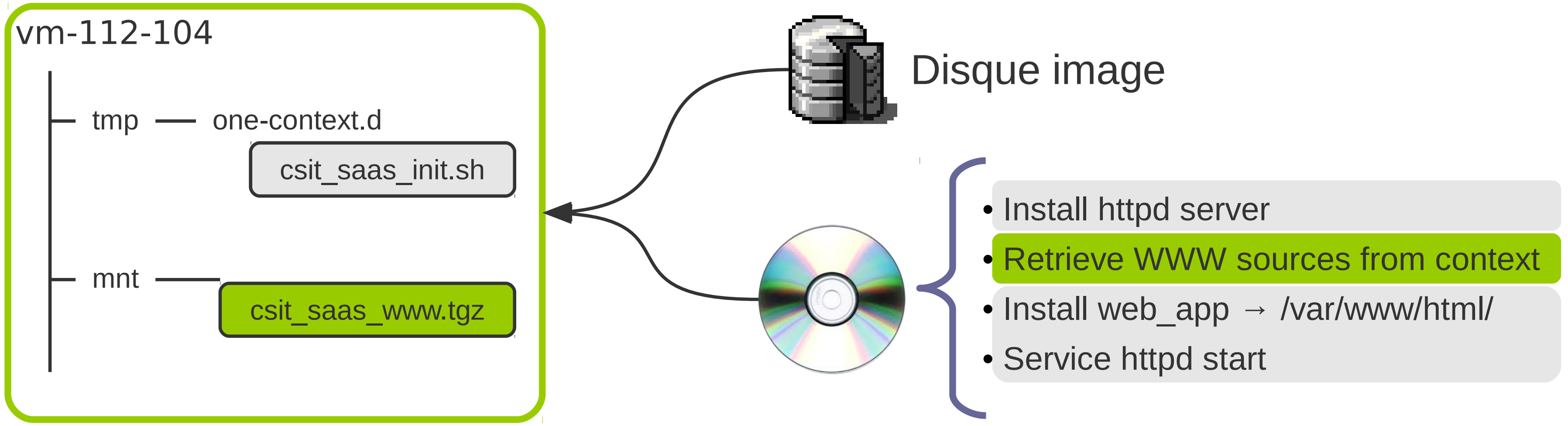
SaaS level contextualisation: template file

```
NAME    = SL64v2

CPU="1"
MEMORY="2048"
VCPU="1"

DISK=[
  BUS="virtio",
  DRIVER="qcow2",
  IMAGE="SL64v2",
  IMAGE_UNAME = "admin" ]
NIC=[
  MODEL="virtio",
  NETWORK="production",
  NETWORK_UNAME="admin" ]
OS=[
  ARCH="x86_64",
  BOOT="hd",
  MACHINE="rhel6.4.0" ]
GRAPHICS=[
  TYPE="spice",
#  PORT    = "5900",
  LISTEN  = "0.0.0.0" ]
CONTEXT=[
        files="/cloudmip/home/francois/csit_saas_init.sh /cloudmip/home/francois/csit_saas_www.tgz",
        init_scripts="csit_saas_init.sh",
        WWW_SRC="csit_saas_www.tgz",
        SSH_PUBLIC_KEY="$USER[SSH_PUBLIC_KEY]" ]
```

Note: To enable users to use the 'file' context attribute, VM_RESTRICTED_ATTR = “CONTEXT/FILES” parameter ought to be commented-out in ***oned.conf*** *file.*

Contextualisation (next)

SaaS level contextualisation: application deployment script



csit_saas_init.sh

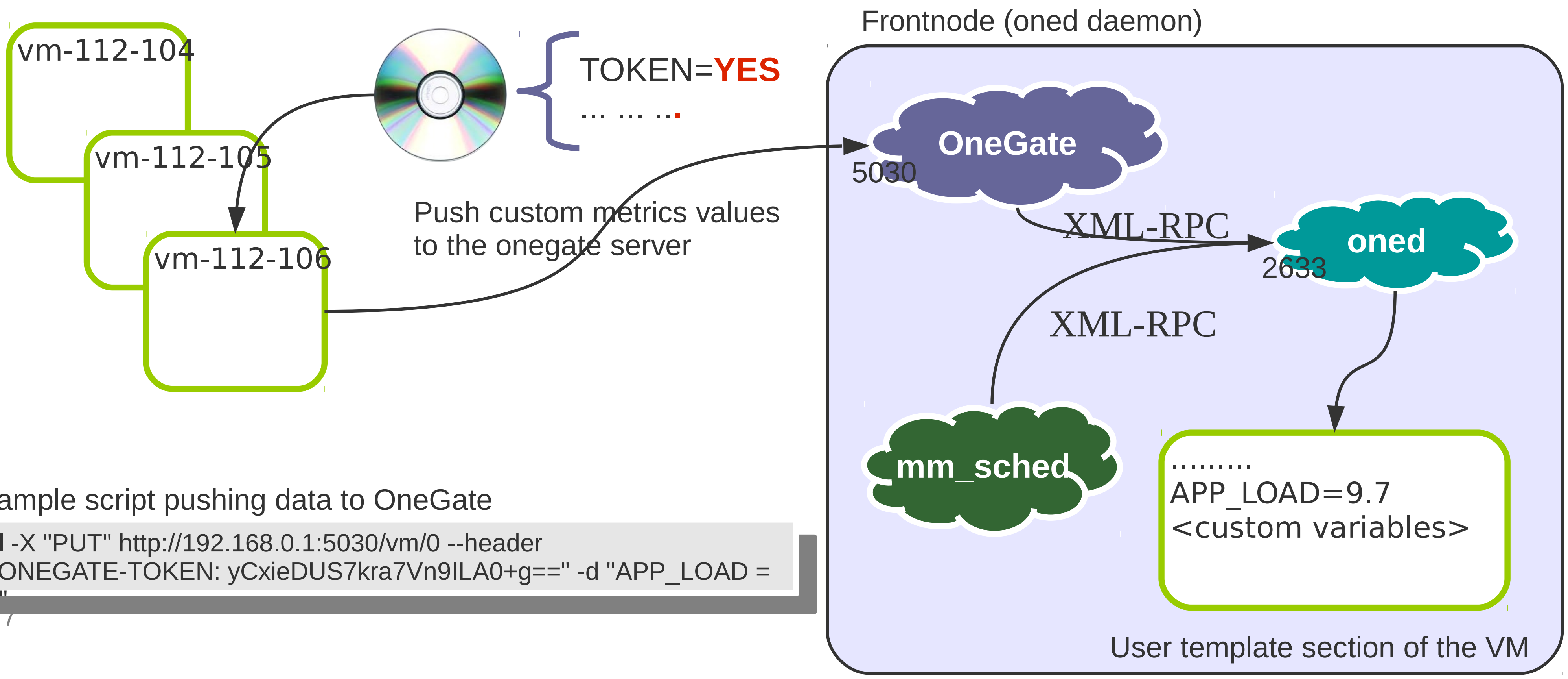
```
#!/bin/bash
#
# Start network
service network restart
# Install software
yum -y install httpd
# Retrieve web application from context
if [ -f /mnt/${WWW_SRC} ]; then
    cd /var/www/html/
    tar -z -xvf /mnt/${WWW_SRC}
    chown -R apache:apache /var/www/html
fi
# Start application
service httpd start
```

Note: In order to install the required software within this installation script, we have to manually start the network. This is due to the fact that CloudMIP do not use OpenNebula's Network contextualisation.

SaaS monitoring

OneGate: VMs applications monitoring (OpenNebula 4.2)

OneGate allows Virtual Machine guests to push monitoring information to OpenNebula. These informations are written within VMs variables templates thus accessible either from the Sunstone GUI or CLI. Users and administrators can use it to gather metrics, detect problems in their applications, and trigger OneFlow elasticity rules.



Example script pushing data to OneGate

```
curl -X "PUT" http://192.168.0.1:5030/vm/0 --header
"X-ONEGATE-TOKEN: yCxieDUS7kra7Vn9ILA0+g==" -d "APP_LOAD =
9.7"
```

SaaS life-cycle management

OneFlow: VMs life-cycle management (OpenNebula 4.2)

OneFlow is a new feature of the latest OpenNebula release that allows management of VMs life-cycle:

- Automatic deployment of VMs with dependencies, ex. A front node with two DBs and 10 workers,
- Elasticity rules to scale up/down the system according to the load provided by the OneGate monitoring system or the CPU, MEMORY, NET_TX and NET_RX sent back by the virtualisation drivers.

```
"roles": [
    {
      "name": "frontend",
      "cardinality": 1,
      "vm_template": 0,

      "min_vms" : 1,
      "max_vms" : 5,
...
```

```
"elasticity_policies" : [
    {
      "expression" : "LOAD > 3",
      "type" : "CHANGE",
      "adjust" : 2,

      "period_number" : 3,
      "period" : 10
    },
    ...
  ]
```

ACLs

Some of the users authentication mechanisms within OpenNebula

It exists several ways to authenticate users against an OpenNebula installation:

• Core: users get declared along with group membership,

• LDAP: users make use of their platform-wide LDAP account,

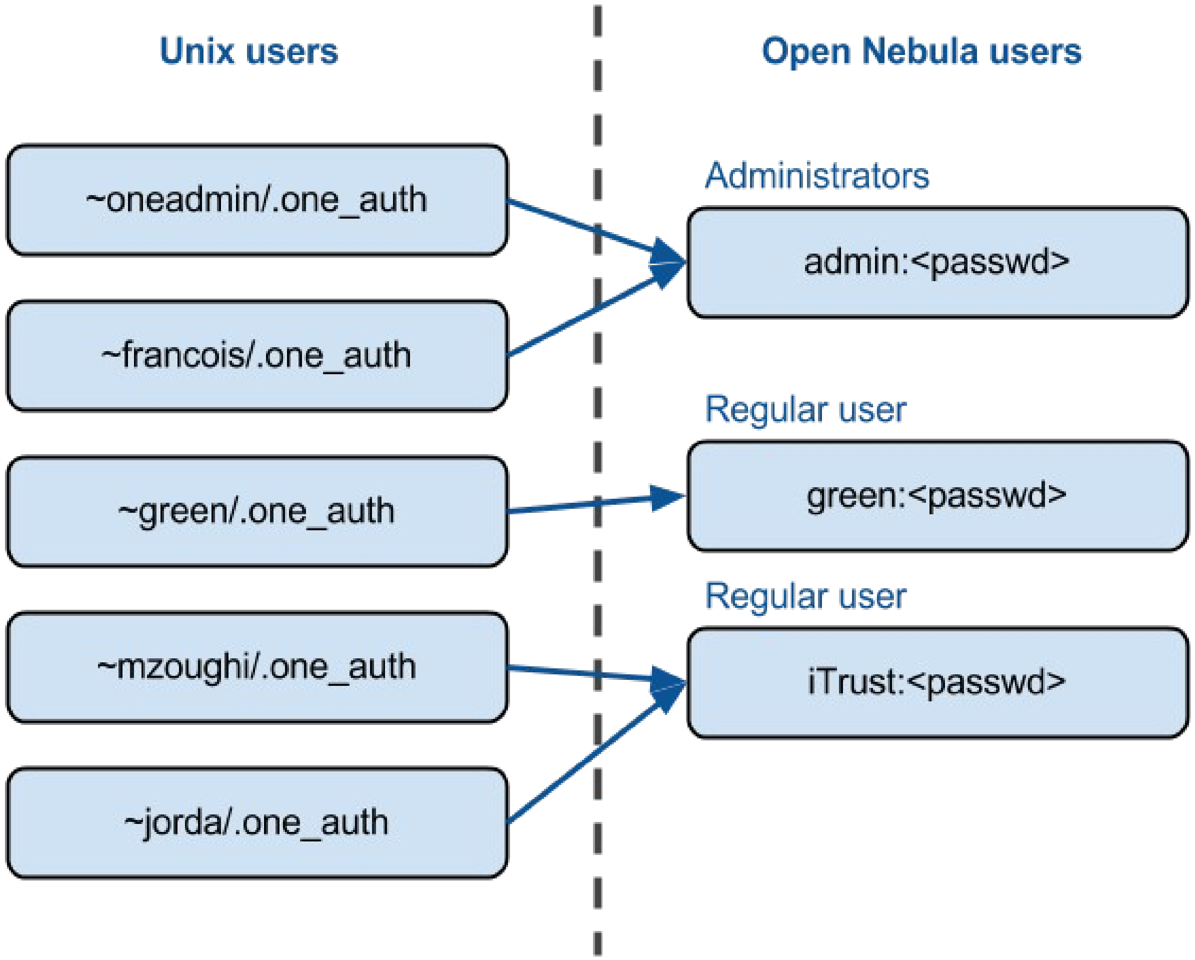
• X509: suitable for large audience or inter-cloud infrastructures involving broard range of people all over the world,

• SSH: users already created at the OpenNebula level ought to login using the 'oneuser login' procedure. This generates a 1hour validity token (default).

ACLs (next)

In the '**core**' authentication model, local users are mapped to OpenNebula users and groups according to a **.one_auth** file each user own in its account. The same apply to the '**ldap**' authentication driver.

*Note: the **.one_auth** file is used for both **core** and **ldap** authentication mechanisms. This file introduce a severe security risk as it holds clear text password of the user. In CloudMIP, we modified the authentication mechanism so users get automatically registered once they login onto the platform frontend or through the Sunstone interface.*



ACLs (next)

Users rights management

By default, all users belongs to the user group.

```
oneadmin@cloudmip[~] onegroup list
  ID NAME         USERS           VMS       MEMORY             CPU
   0 oneadmin         2             -            -               -
   1 users           35    230 /    0    896G /  0    242.0 / 0.0
```

Severall right for several capabilities. Exemple, allowing users to migrate their Vms:

```
oneadmin@cloudmip[~] oneacl create "@1 VM/* ADMIN"
```

Note: @1 means group ID 1 (i.e users)

Another exemple: allowing users to use any host for their VMs:

```
oneadmin@cloudmip[~] oneacl create "@1 HOST/* USE"
```

ACLs granularity levels: users, groups, resources, operations

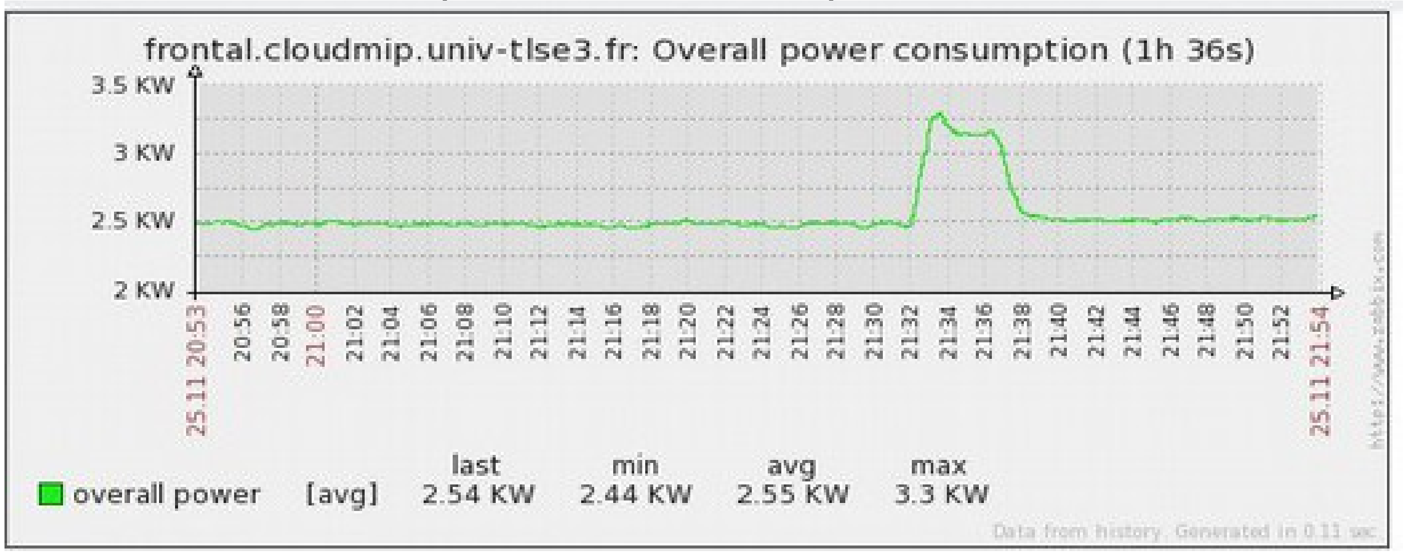
```
oneadmin@cloudmip[~] oneacl list
   ID      USER RES_VHNIUTGDCO   RID OPE_UMAC
    0        @1     V-NI-T----     *     ---c
    1        @1     -H--------     *     -m--
    2         *     ---------O     *     ---c
    3        @1     -H--------     *     u---
    4        @1     V---------     *     --a-
```

Operations
U : USE
M : MANAGE
A : ADMIN
C : CREATE

Resources
V : VM
H : HOST
N : NET
I : IMAGE
U : USER
T : TEMPLATE
G : GROUP
D : DATASTORE
C : CLUSTER
O : DOCUMENT

OpenNebula limitations

- Scalibilities issues due to the central role of the oned daemon which communicates with the others services through XML-RPC
  - Failed deployment when too many VMs are instanciated simultaneously
- Difficult feedback from Vms : requires pooling !
  - Some informations are mandatory for managing and Deploying SaaS
    - Load of the VMs
    - Status of the software stack : Failures, Latency
    - Clear view of non-classical information such as power consumption and heat
- Still in development : Intrusive changes between versions
- Diffusion
  - Current war field is hard between OpenNebula (academic), OpenStack (industry backed) and comercial offers (Amazon, Azure,...).



Work in progress

- Green IT
  - Reduce energy consumption for the planet ?
    - Reduce cost and show a good face
    - Ekotrope raises $1.7M to expand energy efficiency SaaS

- Autonomic systems
  - Reduce human intervention, errors
  - Self healing, scalability

GreenIT : what's next ?

Power consumption of processes* : **ECTOP**

Tool to estimate the power consumed on each running process of the machine

Light weight**, several sensors (PerfCounters, CPU%, Memory, CPU temperature, …) and wattmeters (ACPI, G5K PDUs, CloudMIP, RECS, ...).

Two estimators implemented

Inverse model (PE_IC): calibration with power meter

$$P^{PID} = \frac{P^{Node} \times CPU^{PID}_{time}}{CPU^{Node}_{time}}$$

Linear model (PE_MMC2):

$$P^{PID} = \frac{(P^{Node}_{max} - P^{Node}_{min}) \times CPU^{PID}_{time}}{CPU^{Node}_{time}} + \frac{P^{Node}_{min}}{procs}$$

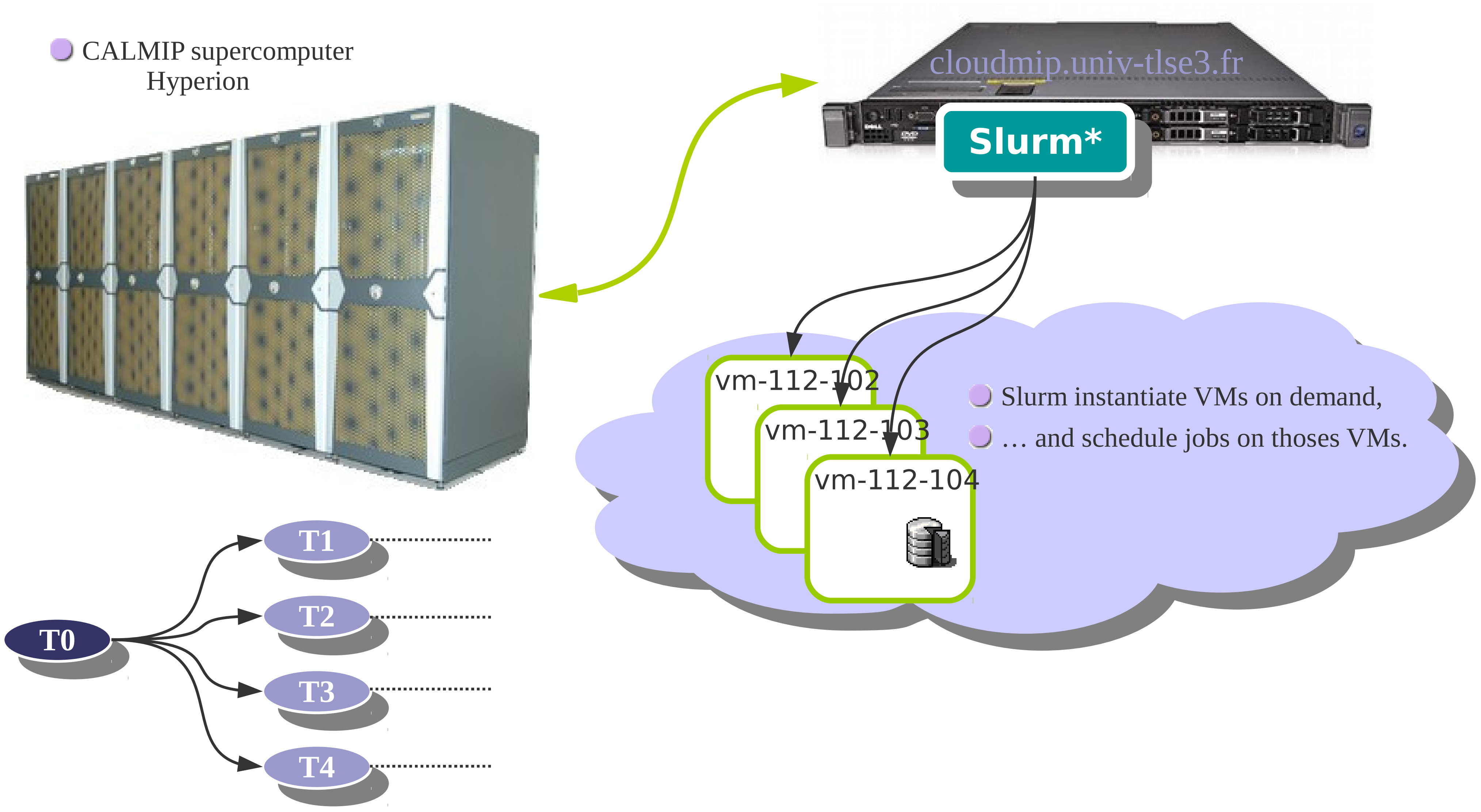
*** as low as Memory: 3Kb, CPU: 0.3%*

KVM hypervisor ==> each VM is a process thus leading to a possible evaluation of the power consumption of each VM!

**ongoing researches from Leandro Fontoura Cuppertino   email:fontoura@irit.fr*

Autonomic system: What's next

- Offloading **embarrassingly parallel worload** from the Hyperion supercomputer to CloudMIP.

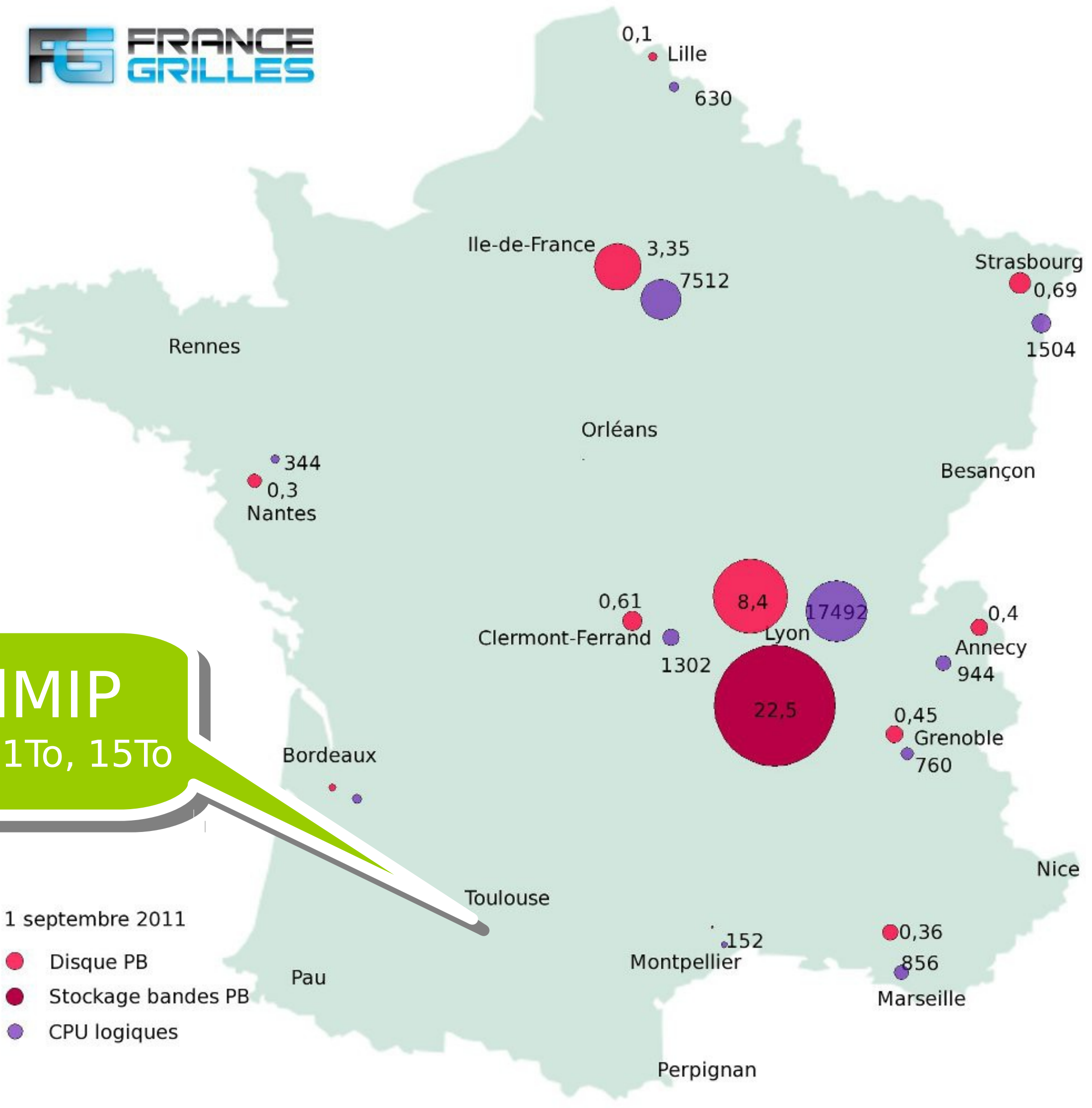


- Slurm instantiate VMs on demand,
- … and schedule jobs on thoses VMs.

France-Grilles

GIS France-Grilles

- Grid production sites (e.g. EGEE tier1),
- Inter-Cloud platform (IRIT, CC-IN2P3, LAL, IPHC...).



CloudMIP
256 cores*, 1To, 15To

**physical cores. Upto 512 cores with hyper-threading.*

France-Grilles Inter-Cloud

Inter-Cloud @ France-Grilles

One of the France-Grilles aims is to build an inter-cloud infrastructure. This will leverage the needs for both regular and SaaS applications that will benefits of such a large-scale VMs deployment capability.

Heterogeneous platform authentication / contextualisation / instantiation

To addess the contextualisation issue in a multi-providers cloud configuration, France-Grilles started to use the Cloud-Init solution [OpenNebula driver is on way],

Several ways to instantiate VMs: OCCI (or future CIMI from DMTF) or libcloud and deltacloud at the client side [tests in progress],

Authentication through VOMS with X509 certificates [tests in progress]

FG VMDirac

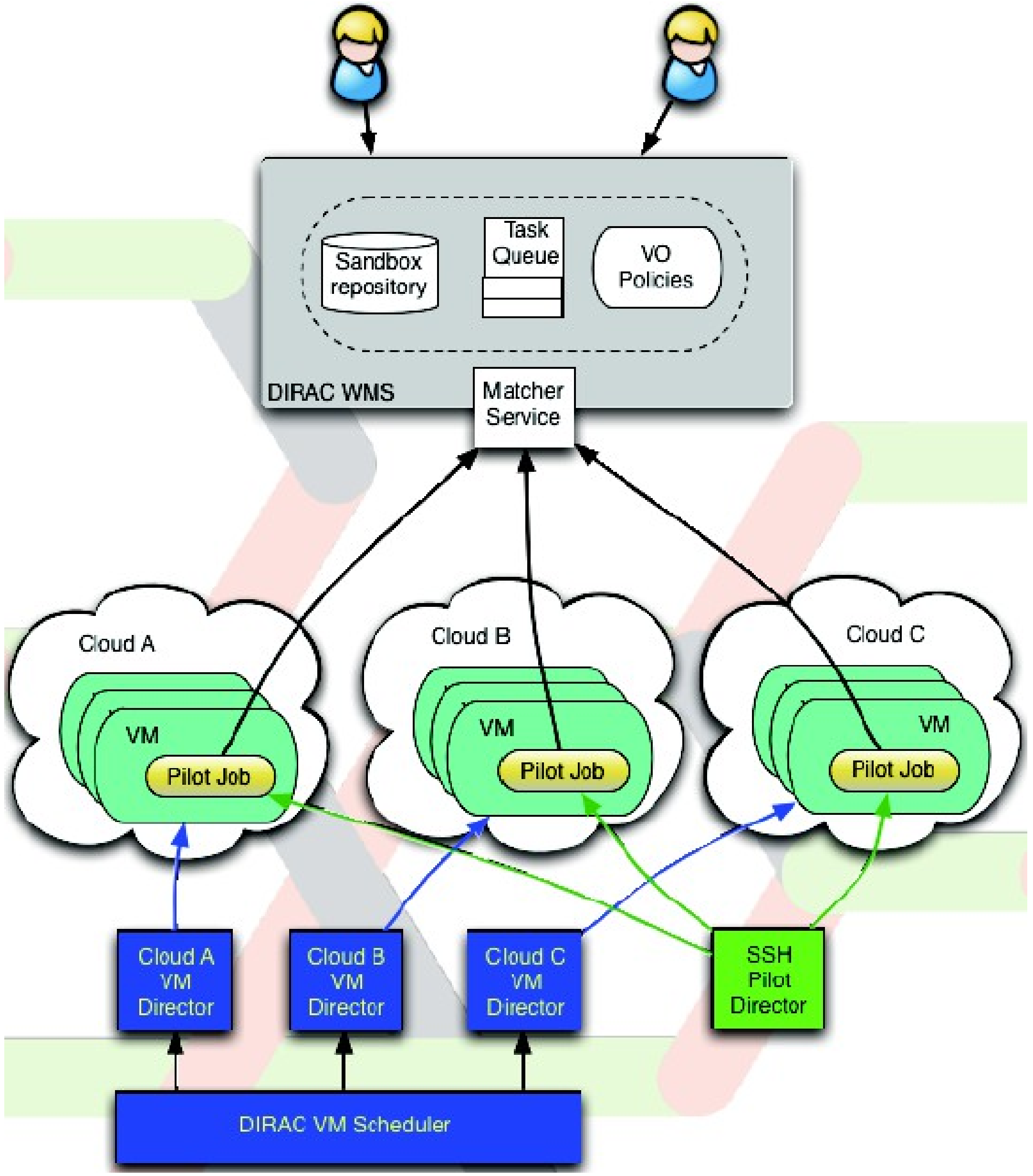
DIRAC is a framework for distributed computing written in Python and originally used for the BELLE experiment.

The VMDirac scheduler features:

- Dynamic VM spawning taking the Task Queue state into account,
- Discards VMs automatically when no more needed.

At the VMs side:

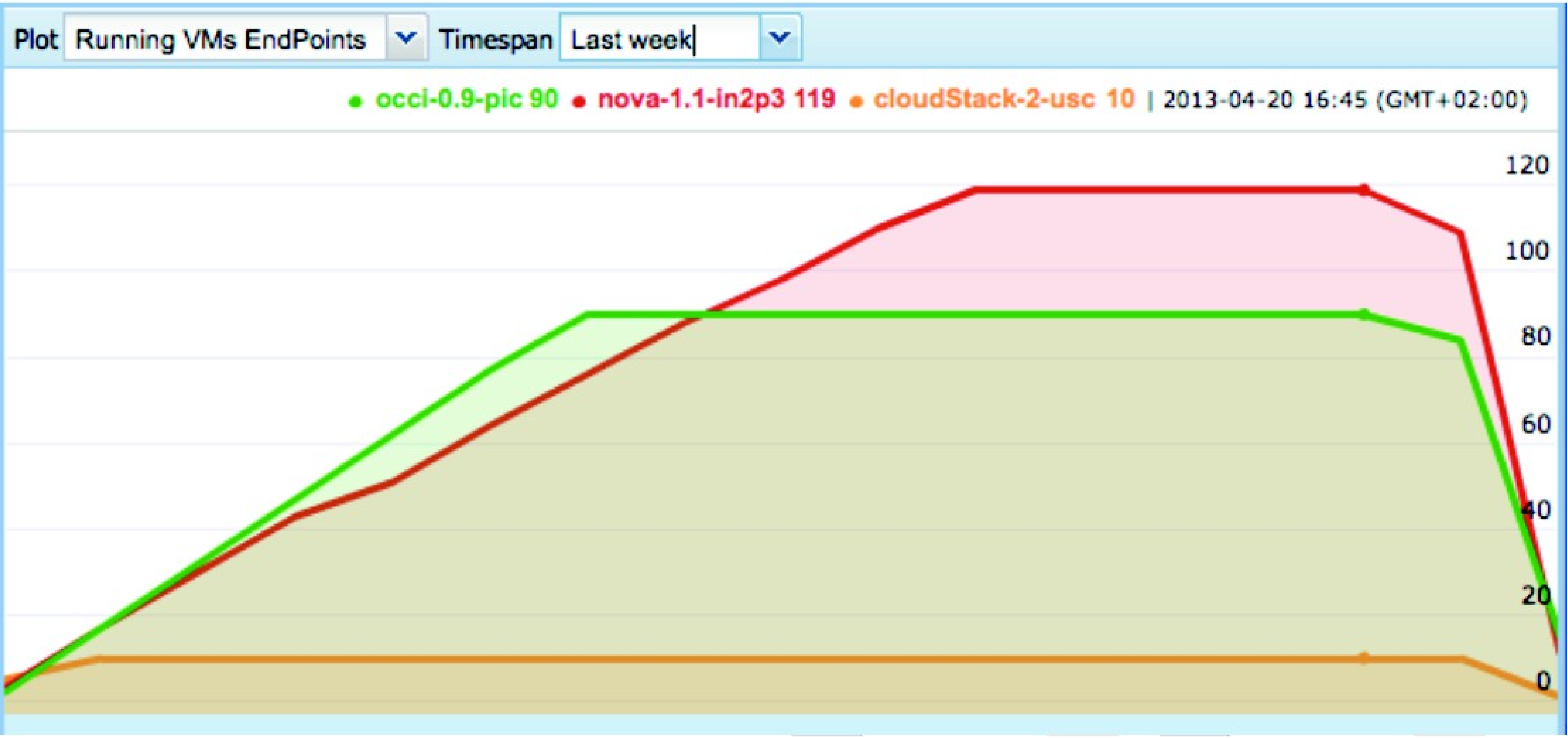
- User account and public key for SSH access,
- Internet-grade IPs (no private ones),
- At boot time, the VM start the "Pilot Job",
- Contextualisation with .ISO image (OpenNebula –PIC), amiconfig (OpenStack –CC/IN2P3) and ad hoc image (CloudStack –USC),
- Start Job Agent and VM monitoring Agent.



Work from Andrei Tsaregorodtsev (CPPM), Victor Mendez (PIC), Victor Fernandez (USC), Mathieu Puel (CC/IN2P3).

FG VMDirac (next)

Total 250 VM slots at 3 sites:
USC, PIC, CC/IN2P3 with 3 different cloud managers
1 virtual CPU, 2GB RAM ==> up to 219 simultaneous VMs achieved.



Work from Andrei Tsaregorodtsev (CPPM), Victor Mendez (PIC), Victor Fernandez (USC), Mathieu Puel (CC/IN2P3).

Plan

Part III – Example

- Usual workflow
- Reality kicks in
- Three tier example
- Practical solutions
- Monitoring !

SaaS objectives

- Managing services !
  - Quality of service
    - Performance, Energy
  - Failure management
- Autonomic management system
  - Humans can not manage thousands of VMs

Usual SaaS workflow

- Several steps
  - Detect a situation (overload, underload, failure, ...)
  - Take a decision
  - Implement the decision
    - Exemple : run a new service instance
      - Start a new instance
      - Advertise this new instance

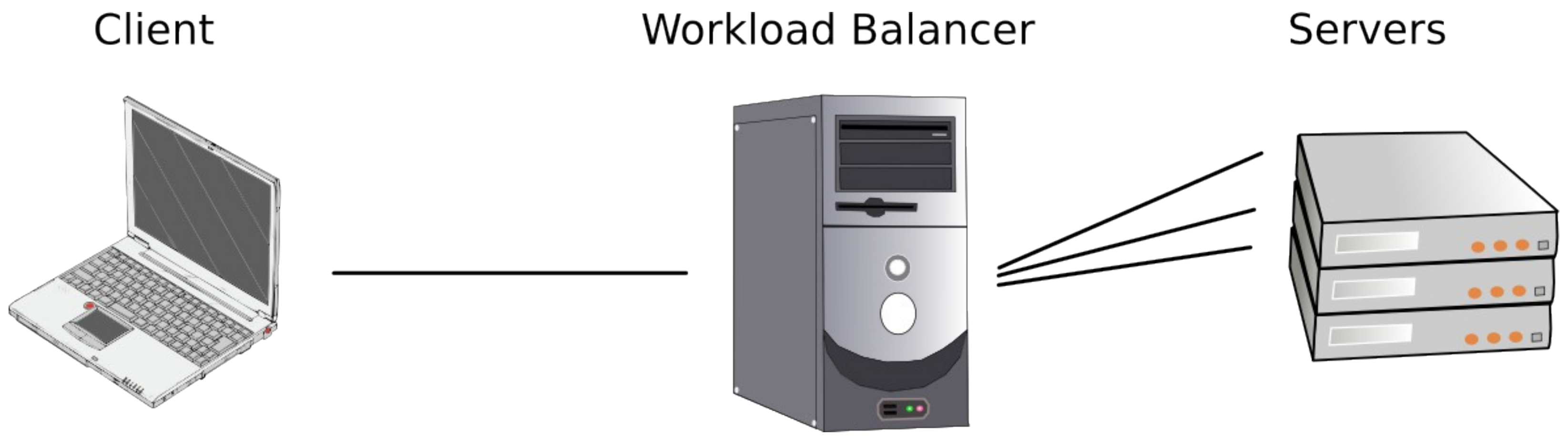
Not so simple

- OpenNebula is not aware of services
  - Running a VM is like starting a host
  - Need to be aware of when the services actually starts !
  - Several techniques
    - The service register itself once started
    - The deployment infrastructure *ping* the service
  - Need of communications between deployment infrastructure and the services

Communication

- From service to deployment infrastructure
  - Special configuration from the service
  - Data about IP/port to transmit
  - By hand
    - Wait for the VM to finish boot
    - Not convenient
  - Using contextualization
    - Send relevant information as context

Classical Three Tier services



- Workload balancer is aware of servers
- Push
  - All new servers warn the Workload Balancer
- Pull
  - The Workload Balancer checks the servers status

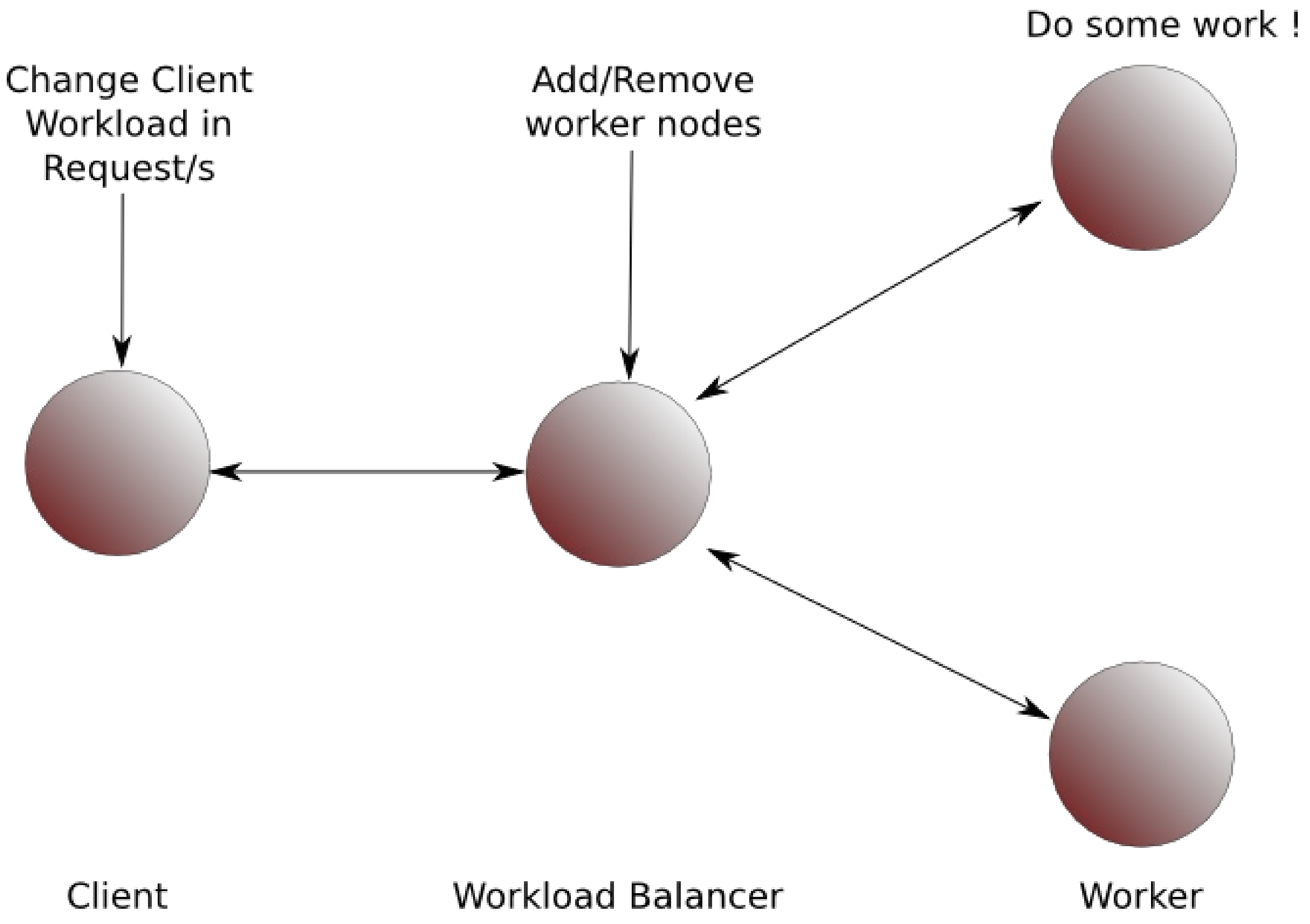
Three Tiers Toybox

XML-RPC coms

Worker
- Send requests
- Change load

Balancer
- Distribute
- Manage workers

Workers
- Just work



170 lines of python

5 CLI tools (including tools to update the client and the workload balancer)

http://www.irit.fr/~Georges.Da-Costa/toybox.tgz

Simple scenario

- Temporal idealistic view
  - Detect an overload
  - Decide to run a new worker
  - Start a new worker
  - Advertise the new worker to the Balancer

Should be quite easy !

Java OpenNebula API

```
public String getVmIpFromId(Integer vmId)
        {
                String ipAdress = null;
                Object[] params = new Object[] { new String("login:passwd"),  vmId };
                try {
                        Object[]result = (Object[]) client.execute("one.vm.info", params);
                        String info = (String)result[1];
                        ipAdress = xml_getdata(info, "IP")
                        if(!((Boolean)result[0]))
                                Error("Cannot retrieve ip adress of the VM :" + result[2] + " " + result[1]);
                } catch(Exception e) {
                        Error("Cannot retrieve ip adress of the VM ");
                        e.printStackTrace();
                }
                return ipAdress;
        }
```

- But only works once the VM finished to boot

Python OpenNebula API

```
#!/usr/bin/python
import xmlrpclib, os, sys, time
# ---Start xmlrpc client to opennebula server
server = xmlrpclib.ServerProxy('http://localhost:2633/RPC2')
new_vm = server.one.template.instantiate('login:passwd', 0, '','')
resource_id  =  new_vm[1]
time.sleep(5)
print server.one.vm.migrate('login:passwd', resource_id, 160, True, False)
```

- Connects to OpenNebula
- Runs a new VM
- Migrates it !
- Rq : does not always work (waiting time is hardcoded)

Real World

- Theoretical workflow
  - Instantiate a VM
  - Update the workload balancer
  - Done
- But !
  - Some VM simply fail
    - High number of Vms
    - High workload on the xml-rpc *bus*
  - Problem of latency
  - Push is not sufficient !

Practical solutions

- Heavy use of the monitoring infrastructure
  - **one.vm.info**
  - But not for all VMs
  - **one.vmpool.info**
  - Used for multiple VMs status queries
  - Status information
    - Running
    - Failed
    - Pending
    - ...

Initial script (aka Wait...)

Example for our Toybox application

- Instanciate balancer
- Wait for callback and check for failure
- Obtain it's IP
- Run a worker (with Balancer IP as parameter)
- Wait for callback (Let the worker contact the balancer)
- ...

And periodicaly

Classical service management

- Check if services are still alive
- Check their load

Less classical possibilities

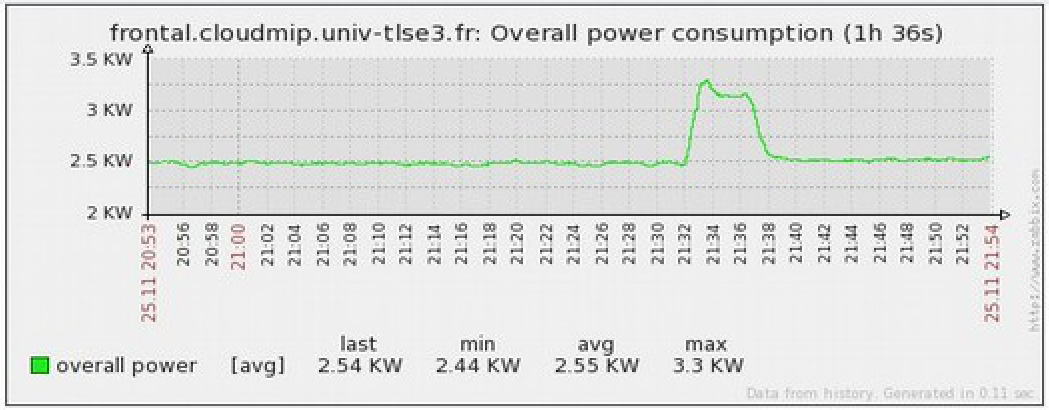
- Depending on the load : migrate some services
  - Consolidate !
    - Reduce energy and improve global efficiency

GreenIT : power metering

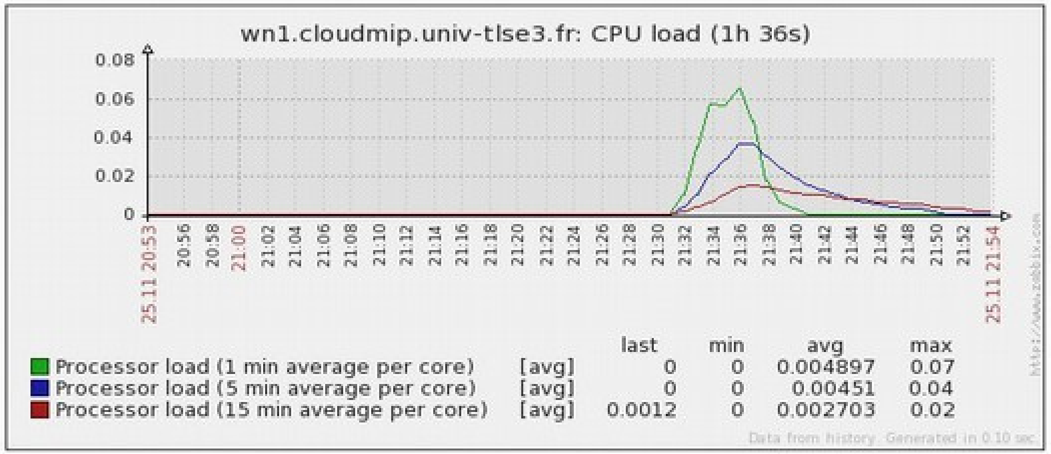
- Launching 248 VMs

```
#> onetemplate instantiate SL64 -m 248
```

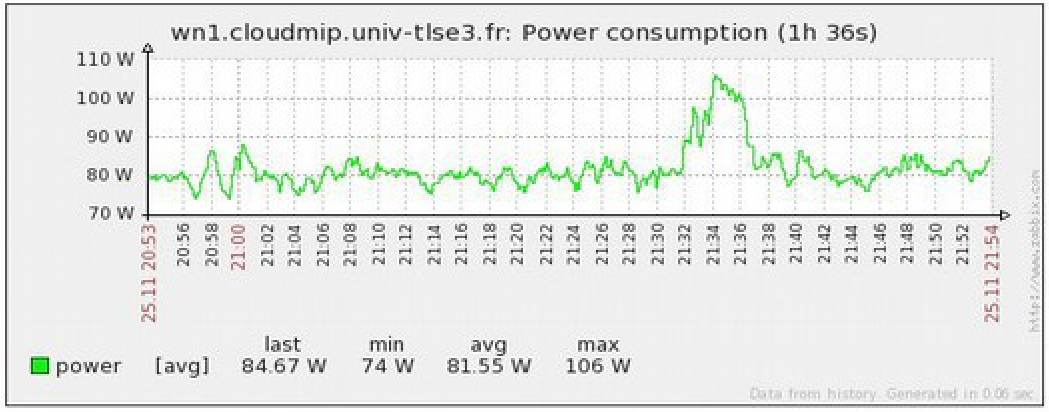
==> leads to eight VMs on each of the 31 nodes, thus each VM using one physical CPU.
==> Max. power consumption is 3.3kw



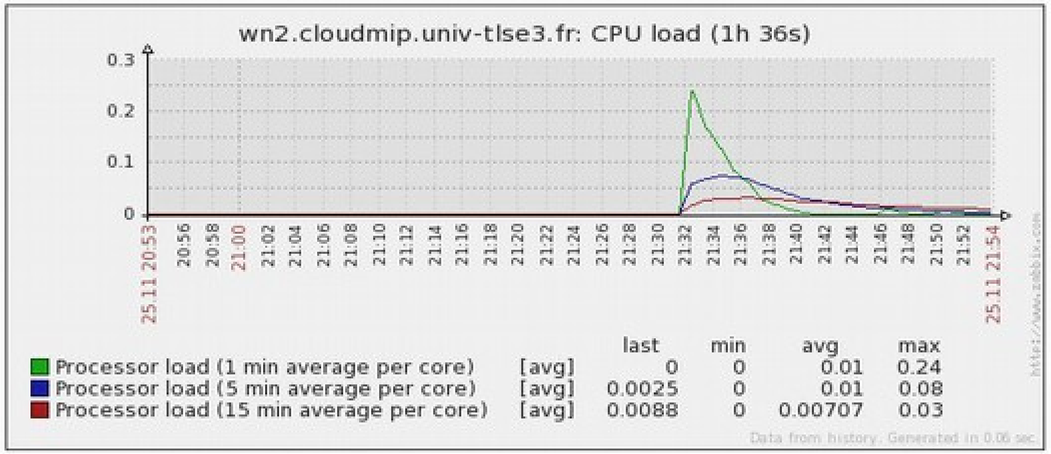
		last	min	avg	max
🟩 overall power	[avg]	2.54 KW	2.44 KW	2.55 KW	3.3 KW



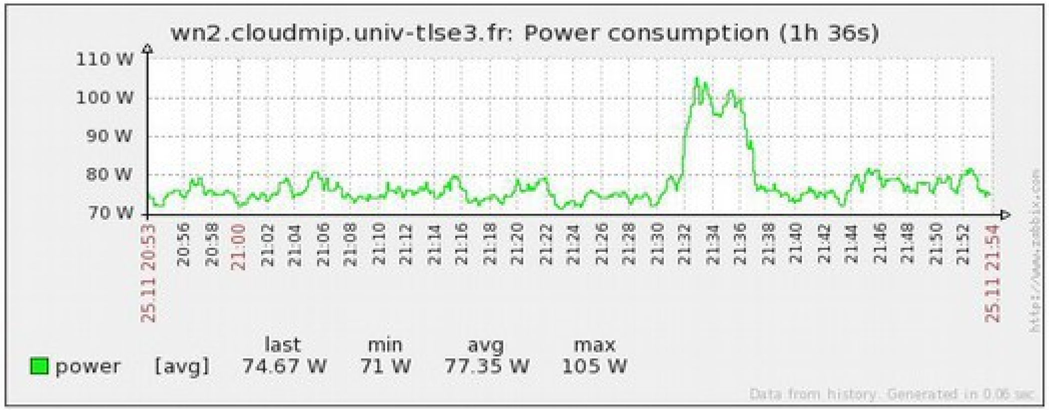
		last	min	avg	max
🟩 Processor load (1 min average per core)	[avg]	0	0	0.004897	0.07
🟦 Processor load (5 min average per core)	[avg]	0	0	0.00451	0.04
🟥 Processor load (15 min average per core)	[avg]	0.0012	0	0.002703	0.02



		last	min	avg	max
🟩 power	[avg]	84.67 W	74 W	81.55 W	106 W



		last	min	avg	max
🟩 Processor load (1 min average per core)	[avg]	0	0	0.01	0.24
🟦 Processor load (5 min average per core)	[avg]	0.0025	0	0.01	0.08
🟥 Processor load (15 min average per core)	[avg]	0.0088	0	0.00707	0.03



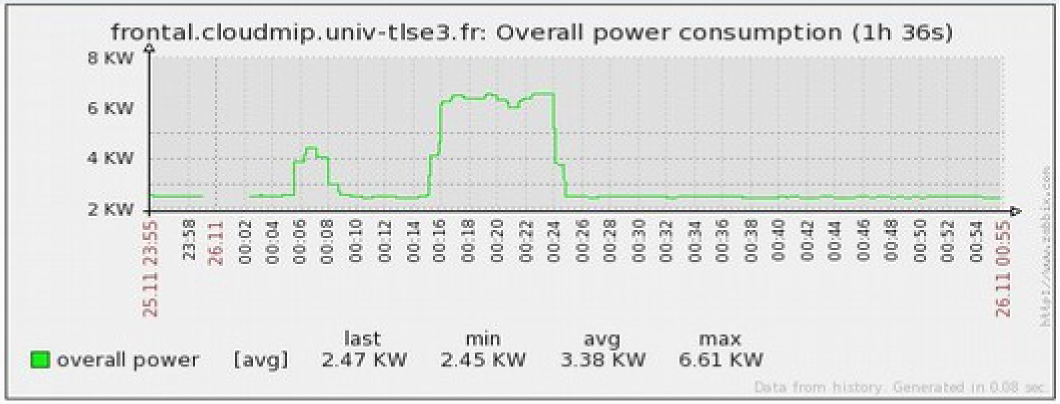
		last	min	avg	max
🟩 power	[avg]	74.67 W	71 W	77.35 W	105 W

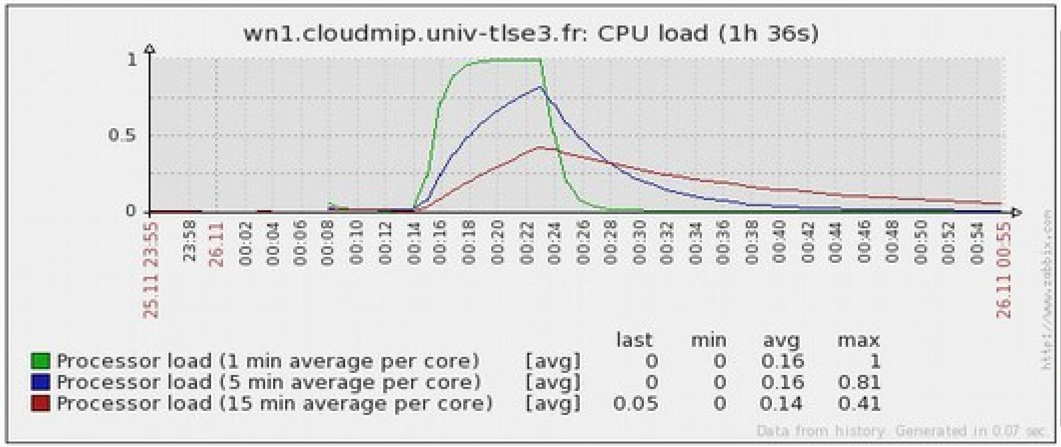
GreenIT : power metering

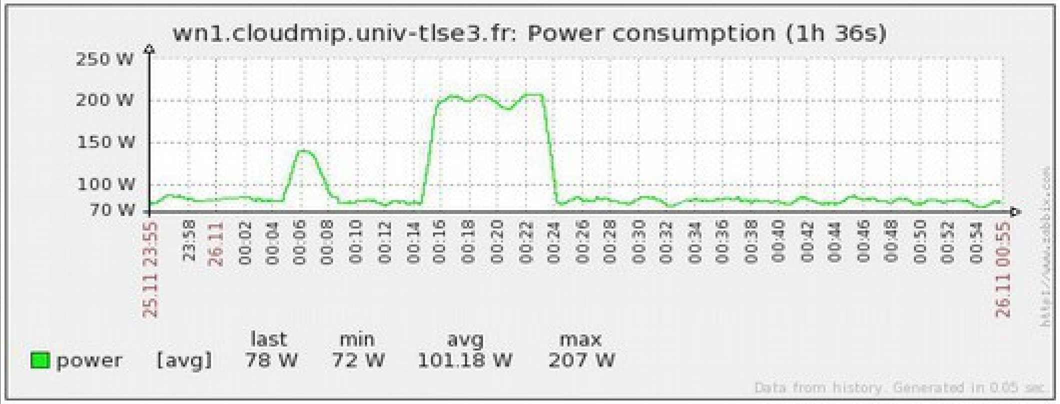
- Launching stress test on all nodes

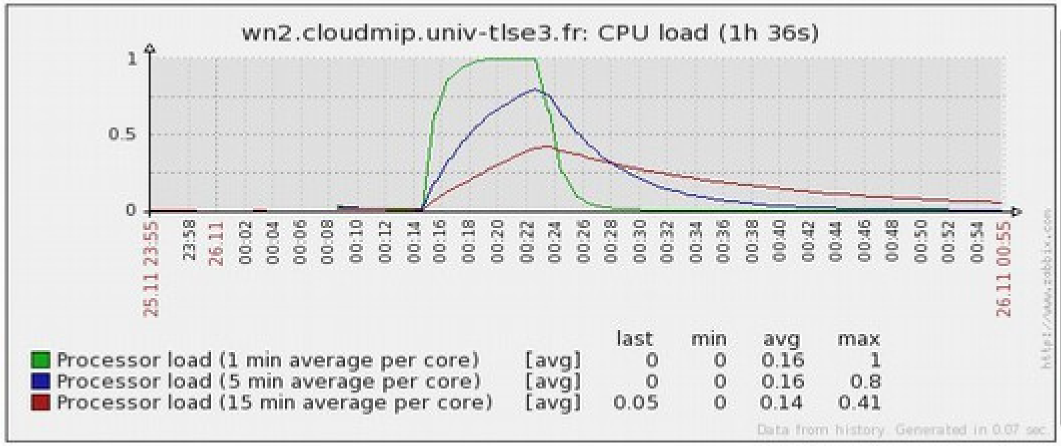
```
#> pdsh -w wn[1..32] -- openssl speed -multi 8
```

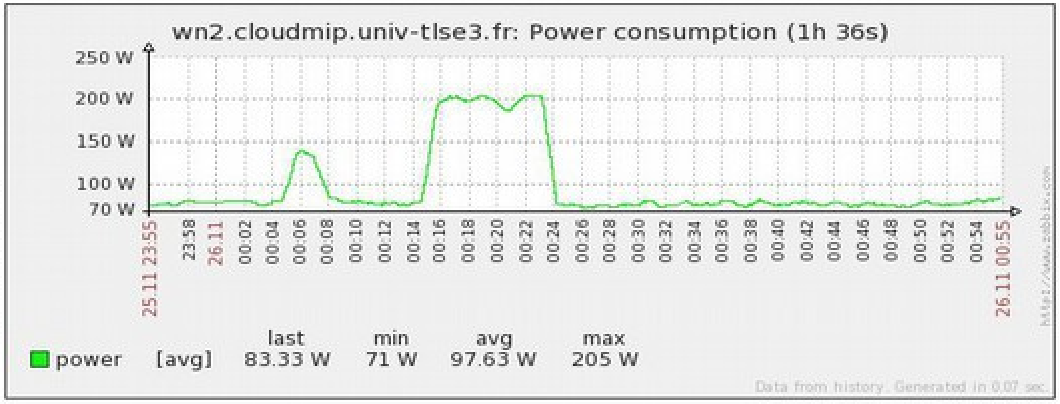
==> Peak power consumption is about 6.6kw



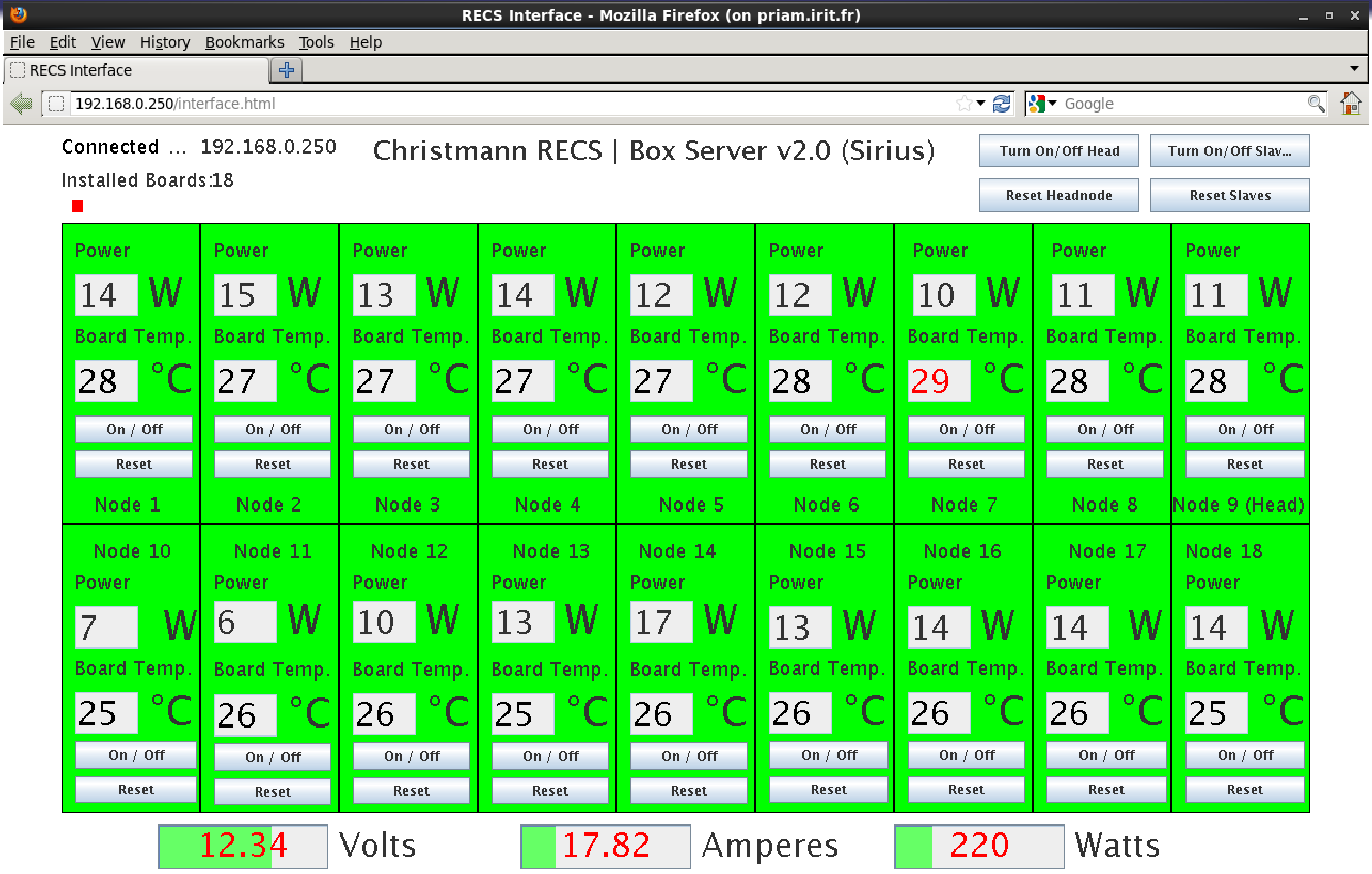








GreenIT : 18 node platform



RECS Interface - Mozilla Firefox (on priam.irit.fr)

192.168.0.250/interface.html

Connected ... 192.168.0.250

Installed Boards:18

Christmann RECS | Box Server v2.0 (Sirius)

Node	Power (W)	Board Temp. (°C)
Node 1	14	28
Node 2	15	27
Node 3	13	27
Node 4	14	27
Node 5	12	27
Node 6	12	28
Node 7	10	29
Node 8	11	28
Node 9 (Head)	11	28
Node 10	7	25
Node 11	6	26
Node 12	10	26
Node 13	13	25
Node 14	17	26
Node 15	13	26
Node 16	14	26
Node 17	14	26
Node 18	14	25

12.34 Volts 17.82 Amperes 220 Watts

The END

- Slides: http://www.irit.fr/~Georges.Da-Costa
  - Teaching → CSIT Tutorial
- Contact:
  - Georges.Da-Costa@irit.fr
- We are open for future collaborations
  - Energy efficiency in clouds systems

The SEPIA team

The SEPIA team (IRIT: Pr Jean-Marc Pierson / N7: Pr Daniel Hagimont) mainly focuses on GreenIT, autonomic and distributed systems (like cloud filesystems).

10 permanents (4 Pr, 5 MCF, 1 Dr-engineer)

1 engineer, 1 Post-doc (CoolEmAll),
21 PhD students.

CoolEmAll (FP7),
SVC (Grand Emprunt),
SOP and Control Green (ANR).

Toulouse platforms :
Director : Pr Jean-Marc Pierson

- **Grid5000-Toulouse (560),**
- **GridMIP (128),**
- **CloudMIP (256),**
- **RECS 18nodes 1U board.**

Pau platform :

- **PireCloud (128).**

GreenIT : RECS platform

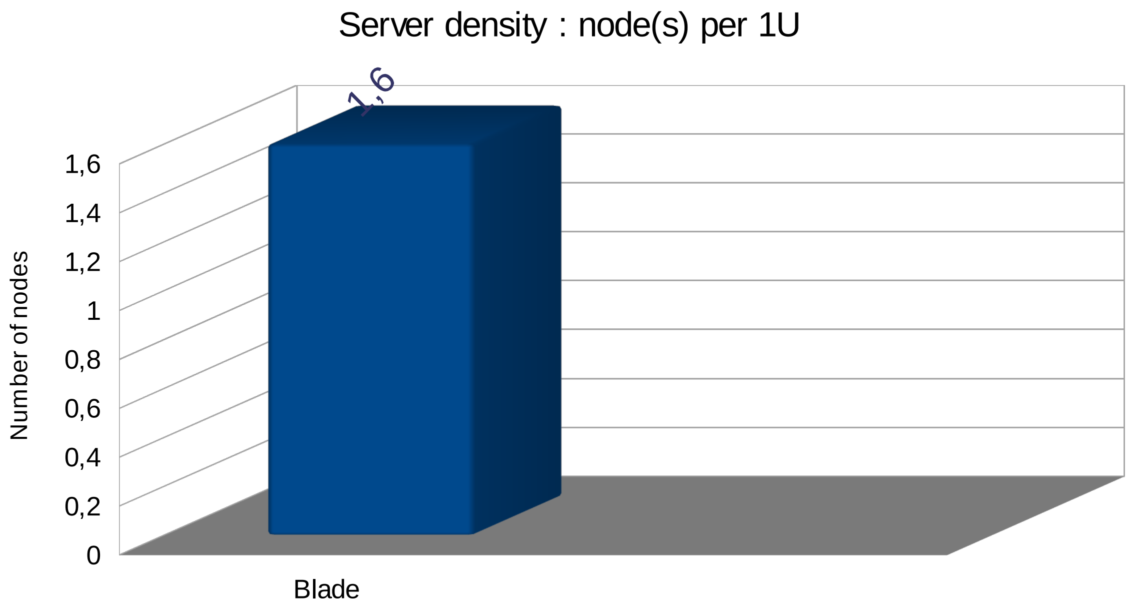
RECS : extreme density servers from Christmann Gmbh*

RECS 2.0 : upto 18 nodes* (Atom, i7, ARM ...) to fit in a standard 1U rack !

***COM-Express boards**

And RECS 3.0 : upto 72 ULP-COM boards!

CHRISTMANN INFORMATIONSTECHNIK + MEDIEN



**http://www.christmann.info/*   *Member of the CoolEmAll project (FP7)*

GreenIT : RECS platform