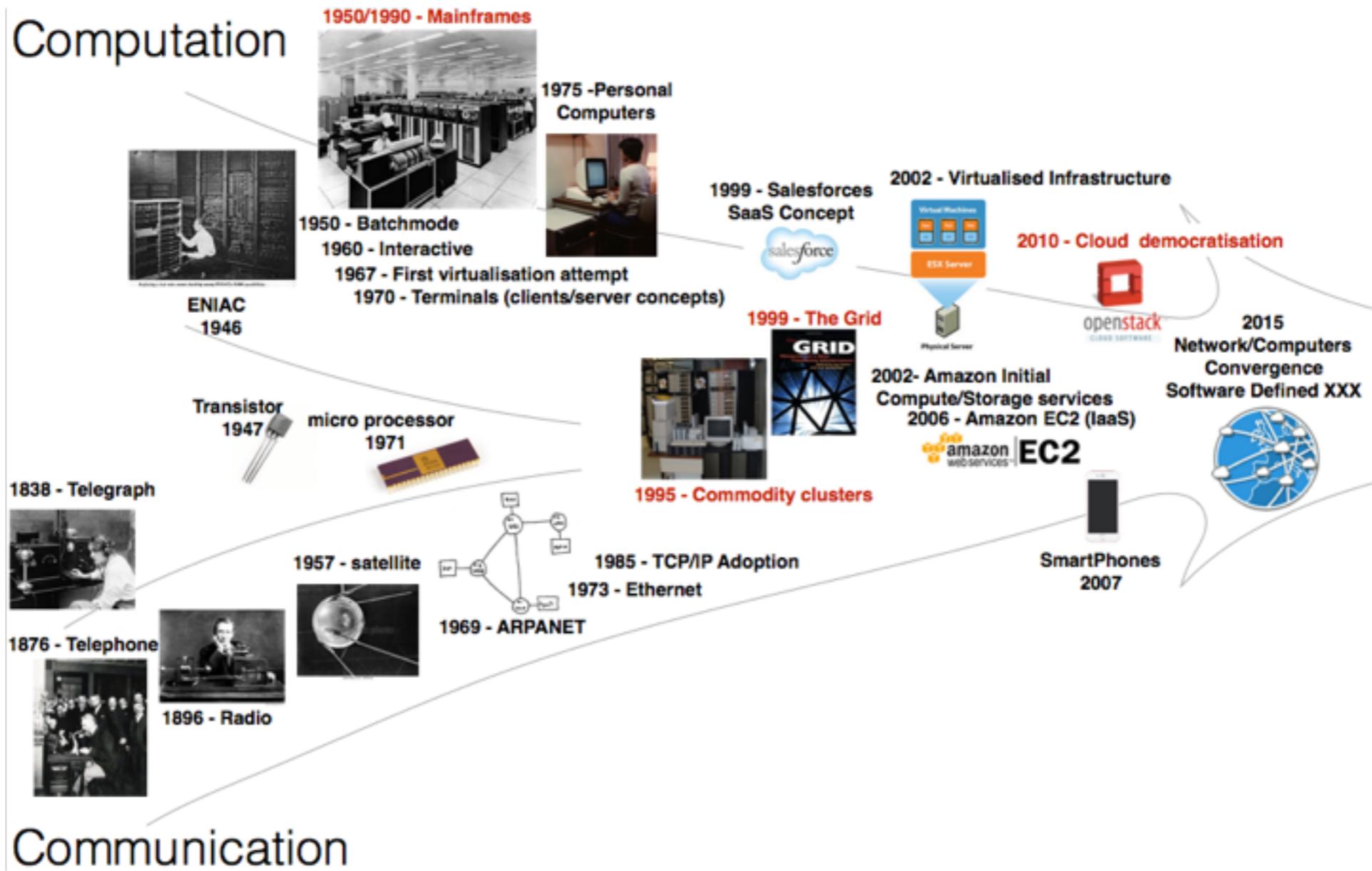


## Computation



# Utility Computing: From Mainframes to Clouds and Beyond!

Adrien Lebre  
RESCOM - June 2015, the 20th

# Who am I...

- Inria Researcher (on leave from an Ass. Prof. position in IMT Atlantique, aka Ecole des Mines de Nantes)
- Co-chair of the Virtualisation action of the RSD CRNS Research Group
- Chair of the Discovery Inria Project Lab
- Chair of the Fog/Edge/Massively Distributed Clouds WG OpenStack



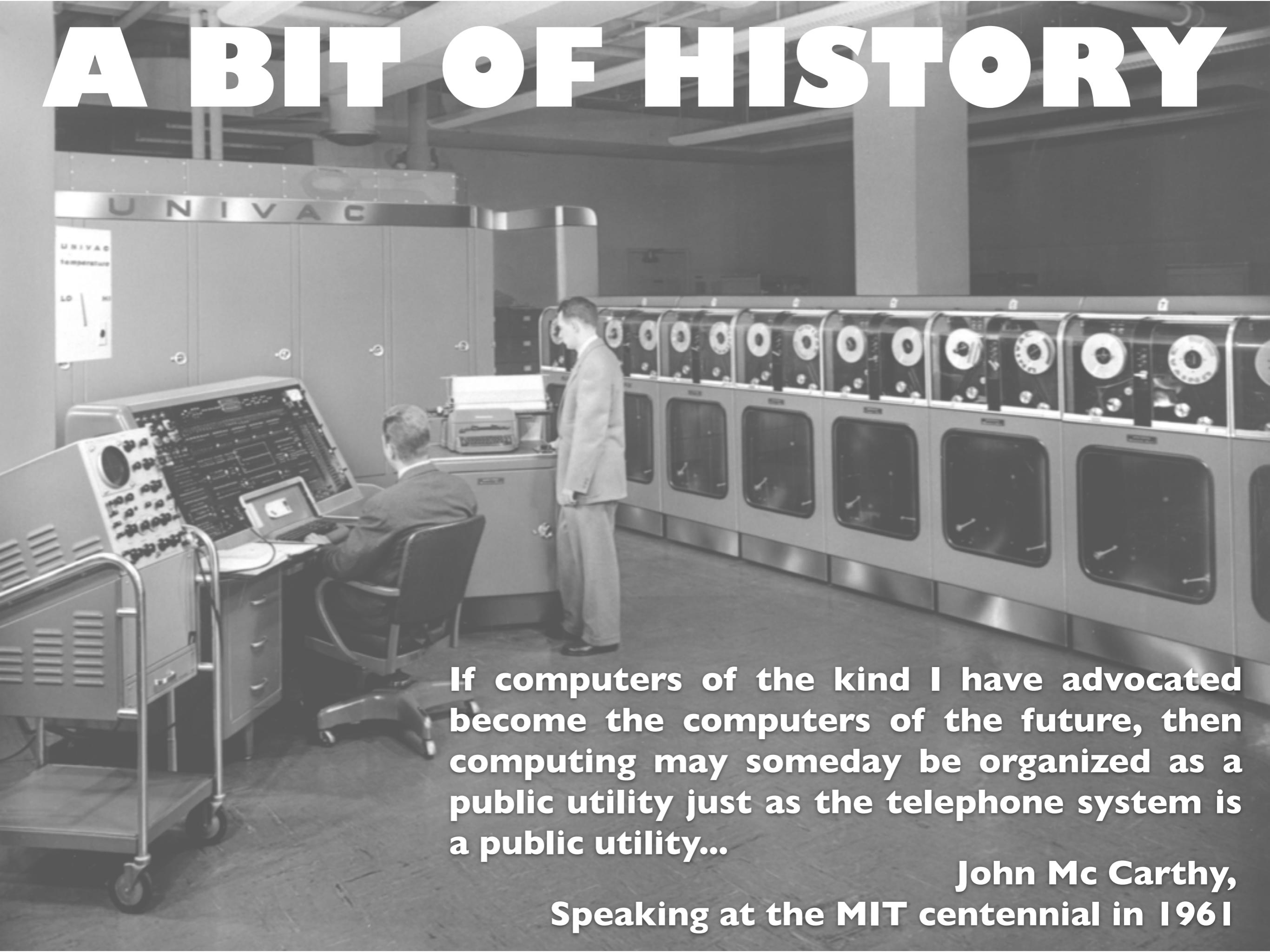
# Who am I...

- Inria Researcher (on leave from an Ass. Prof. position in IMT Atlantique, aka Ecole des Mines de Nantes)



A word cloud centered around the concept of Utility Computing. The most prominent words are 'UTILITY' (large blue), 'COMPUTING' (large green), and 'COMPUTER' (large black). Other significant words include 'SOFTWARE' (teal), 'DATA' (yellow), 'APPLICATION' (green), 'GRID' (orange), 'DECENTRALIZED' (green), 'SUPERCOMPUTER' (yellow), 'MULTIPLE' (yellow), 'LARGE' (black), 'WEB' (green), 'OPERATING' (green), 'HARDWARE' (green), 'GRID SERVER' (blue), 'DEMAND' (green), 'PUBLIC' (yellow), 'GRID' (orange), 'SEISMIC' (orange), 'PROCESS' (yellow), 'NEEDED' (green), 'METERED' (blue), 'MAINFRAME' (black), 'FRAME' (black), 'ARCHITECTURE' (yellow), 'INITIAL' (white), 'GEOGRAPHICALLY' (yellow), 'OPERATING' (green), 'NODES' (green), 'BUSINESS' (brown), 'PURPOSE' (orange), 'ENVISIONED' (yellow), 'INDUSTRY' (brown), 'WORKLOAD' (yellow), 'SHIFT' (orange), 'BEHEST' (blue), 'PROPAGATED' (blue), 'LAUNCHED' (black), 'ADVOCATED' (blue), 'DISTRIBUTED' (black), 'COMPUTATIONAL' (blue), 'RESOURCES' (green), 'SERVICE' (green), 'SPECIFICALLY' (brown), 'SINGLE' (yellow), 'KIND' (brown), 'ANNOUNCED' (yellow), 'CLUSTER' (green), 'CENTERS' (green), 'VIRTUALIZATION' (brown), 'COMPUTATION' (brown), 'ORGANIZATIONS' (brown), 'RENTED' (brown), 'SKEPTICISM' (blue), and 'INITIAL' (white).

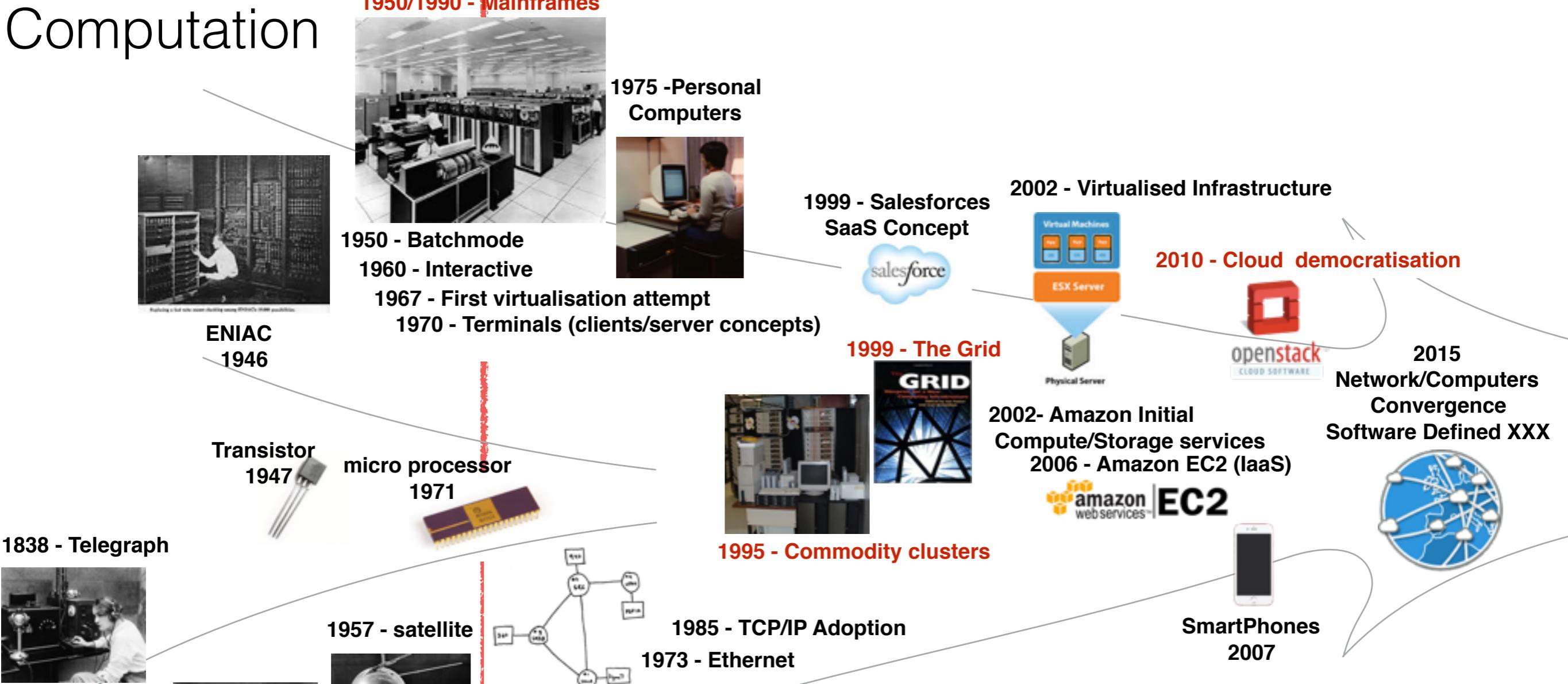
# A BIT OF HISTORY



If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility...

John Mc Carthy,  
Speaking at the MIT centennial in 1961

# Computation



# Communication

# Computation



ENIAC  
1946

1950/1990 - Mainframes



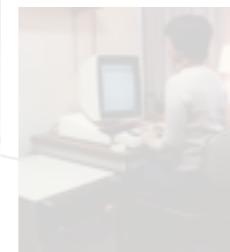
1950 - Batchmode

1960 - Interactive

1967 - First virtualisation attempt

1970 - Terminals (clients/server concepts)

1975 - Personal Computers



1999 - Salesforces SaaS Concept



2002 - Virtualised Infrastructure



2010 - Cloud democratisation



2015

Network/Computers Convergence Software Defined XXX



1838 - Telegraph



1876 - Telephone



1896 - Radio



1957 - satellite



1969 - ARPANET



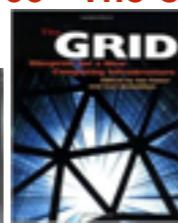
1985 - TCP/IP Adoption

1973 - Ethernet

1995 - Commodity clusters



1999 - The Grid



2002 - Amazon Initial Compute/Storage services



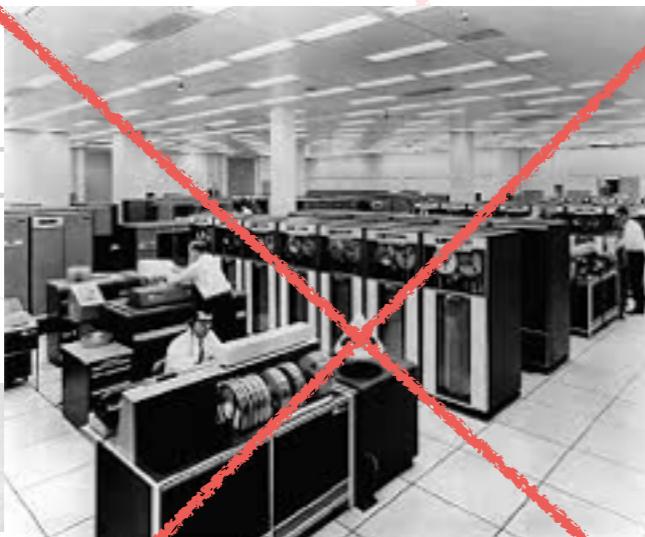
2006 - Amazon EC2 (IaaS)



SmartPhones  
2007

# Not discussed in this talk

Computat



**1950 - Batchmode**

**1960 - Interactive**

**1967 - First virtualisation attempt**

**1970 - Terminals (clients/server) concept**

1947

micro processor  
1971

1838 - Telegraph



1876 - Telephone



1896 - Radio

1957 - satellite



1969 - ARPANET



1975 - Personal Computers



1999 - Salesforces SaaS Concept



1999 - The Grid



2002 - Virtualised Infrastructure



2010 - Cloud democratisation



2015  
Network/Computers  
Convergence  
Software Defined XXX



SmartPhones  
2007

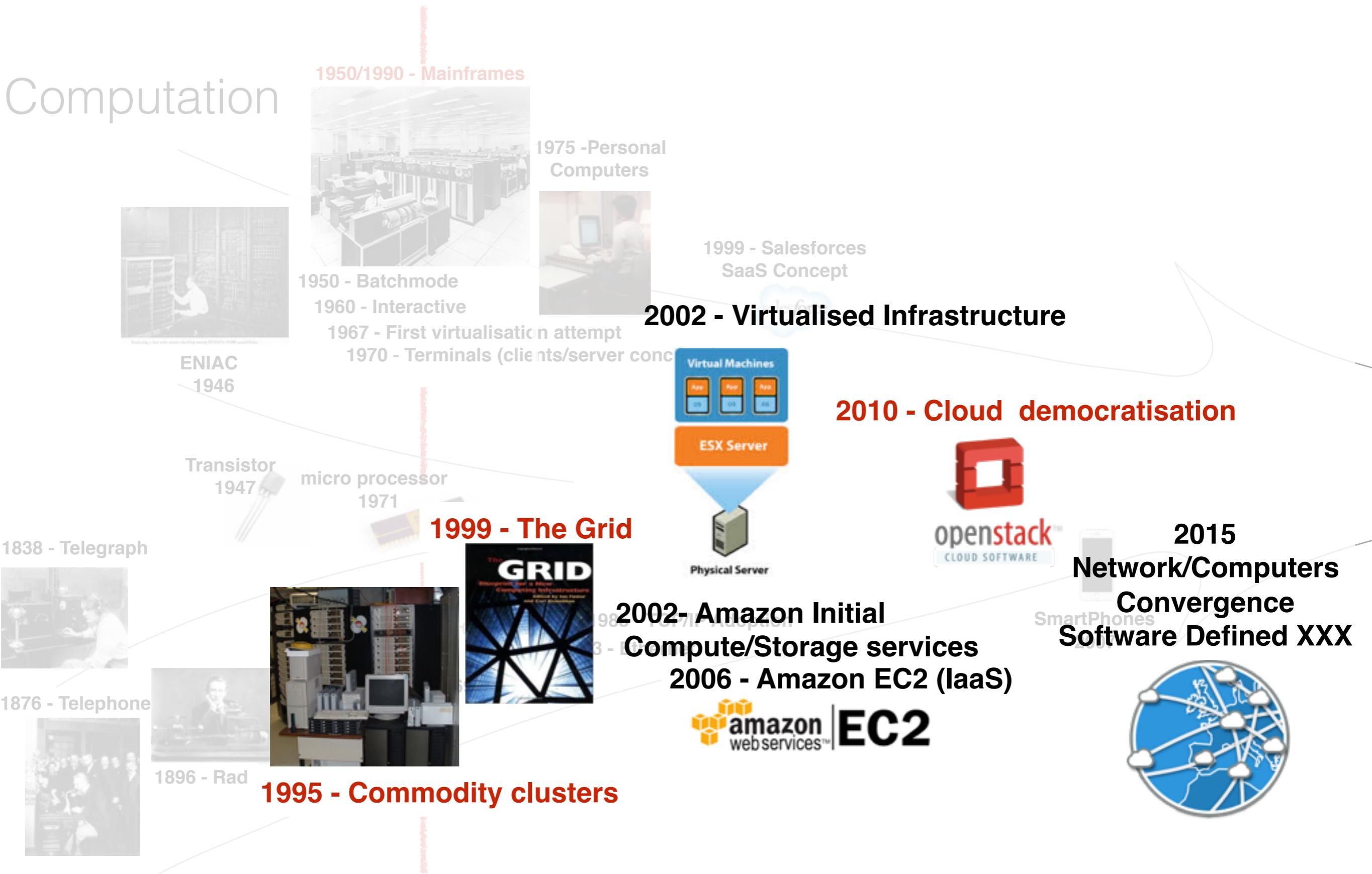
1995 - Commodity clusters

1985 - TCP/IP Adoption

1973 - Ethernet

1969

# Computation



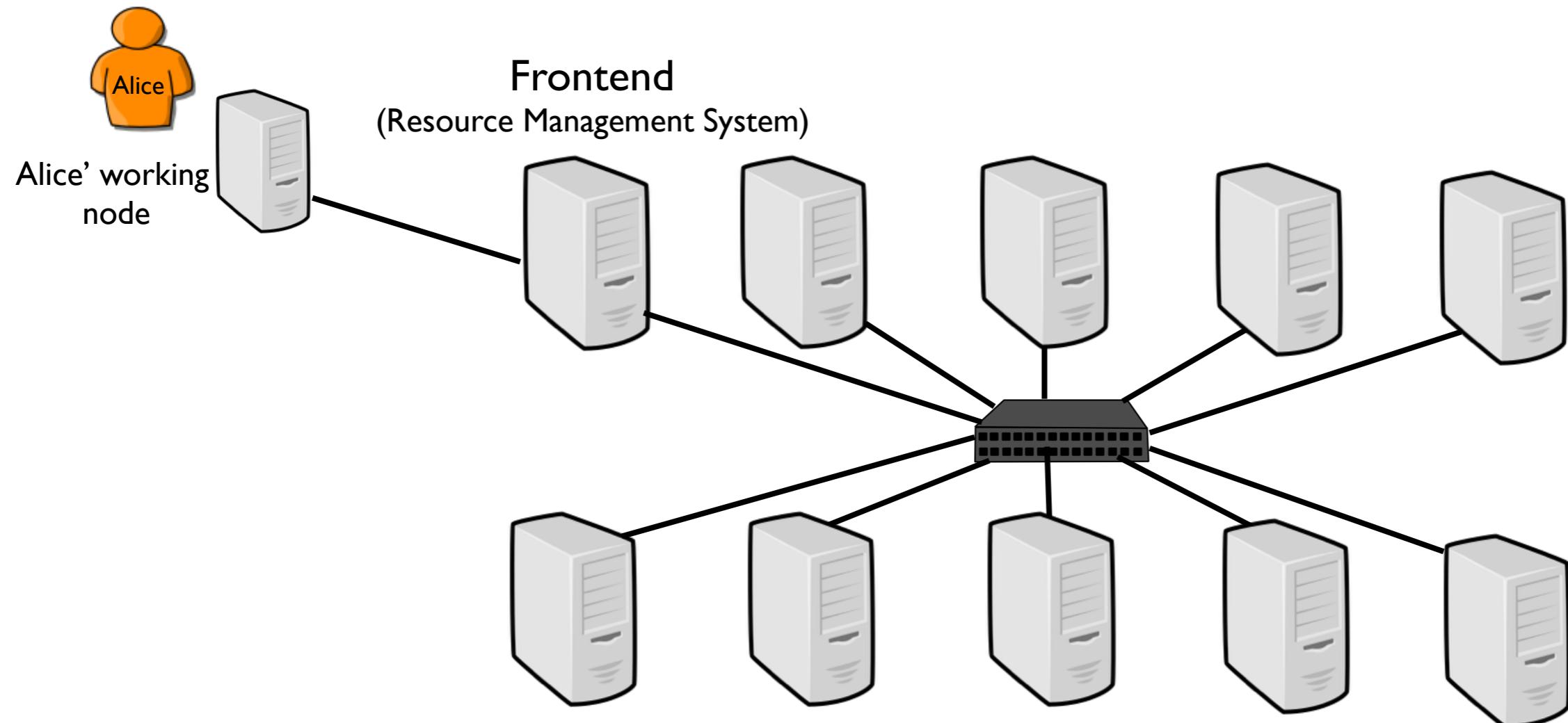
# Communication

# Looking back...

- **xxx Computing**  
Meta / Cluster / Grid / Desktop / “Hive” / Cloud / Sky ...  
⇒ **xxx as Utility Computing**
- A common objective: provide computing resources  
(both hardware and software) in a flexible, transparent,  
secure, reliable, ... way
- **Challenges**
  - Software/Hardware heterogeneity
  - Security (Isolation between applications, ...)
  - Reliability / Resiliency
  - Data Sharing
  - Performance guarantees...

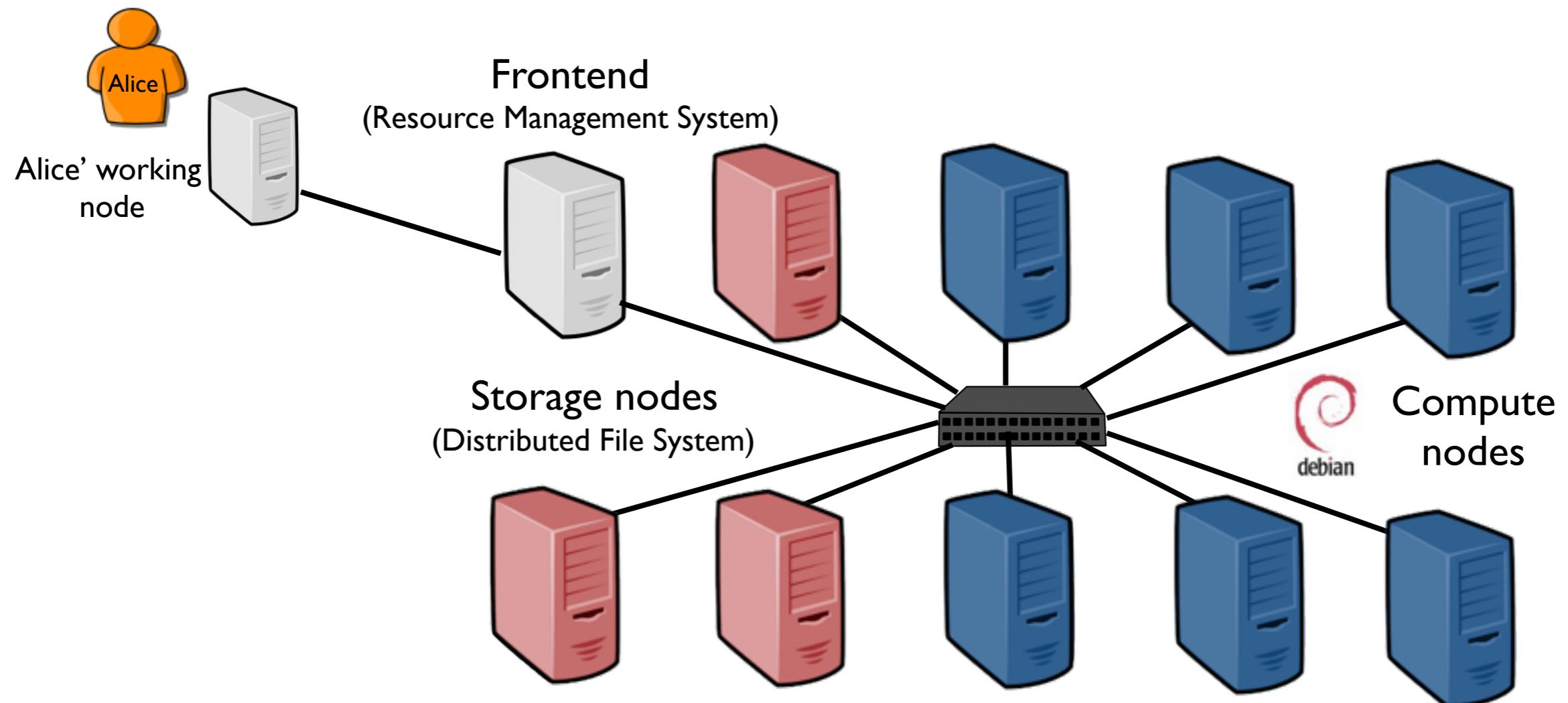
# Looking back...

- Network of Workstations 1990 / 20xx



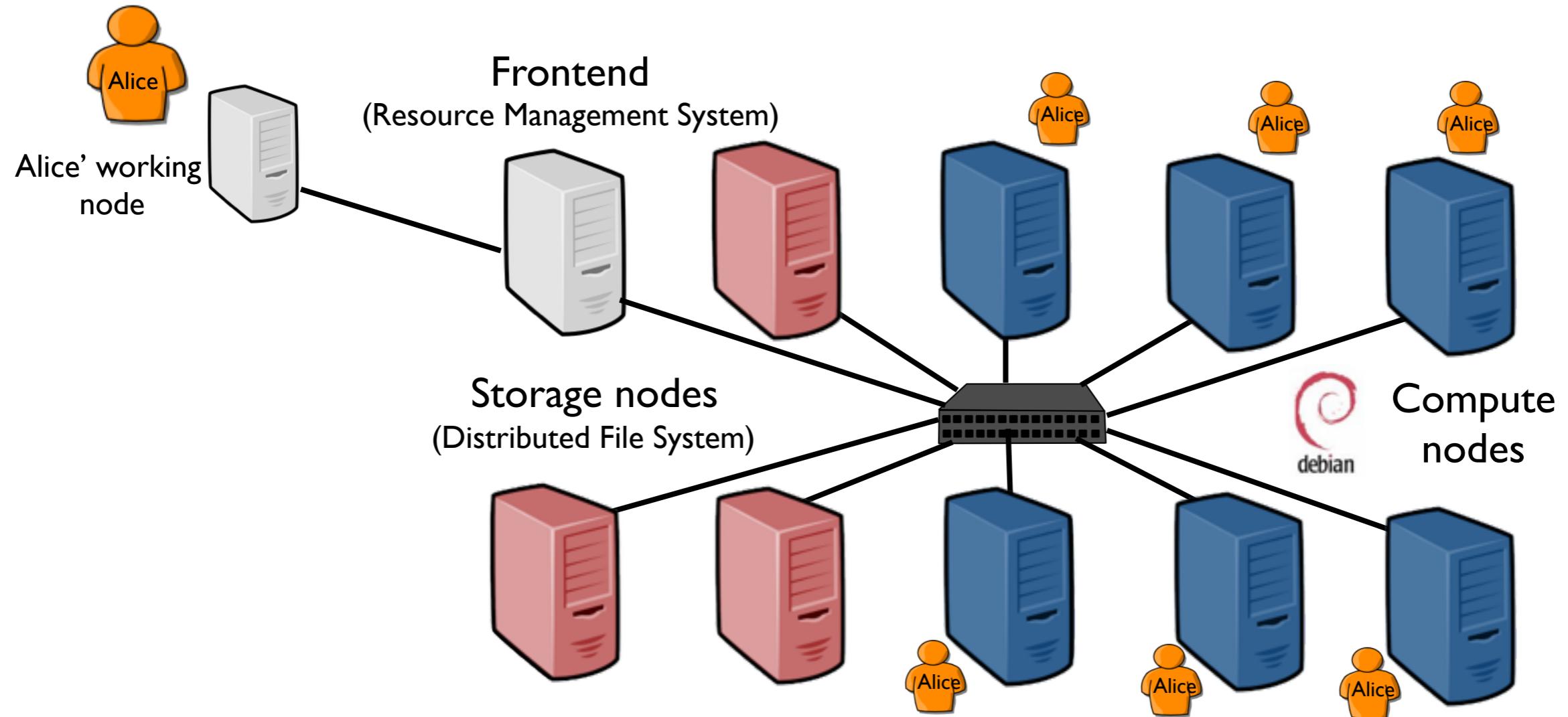
# Looking back...

- Network of Workstations 1990 / 20xx



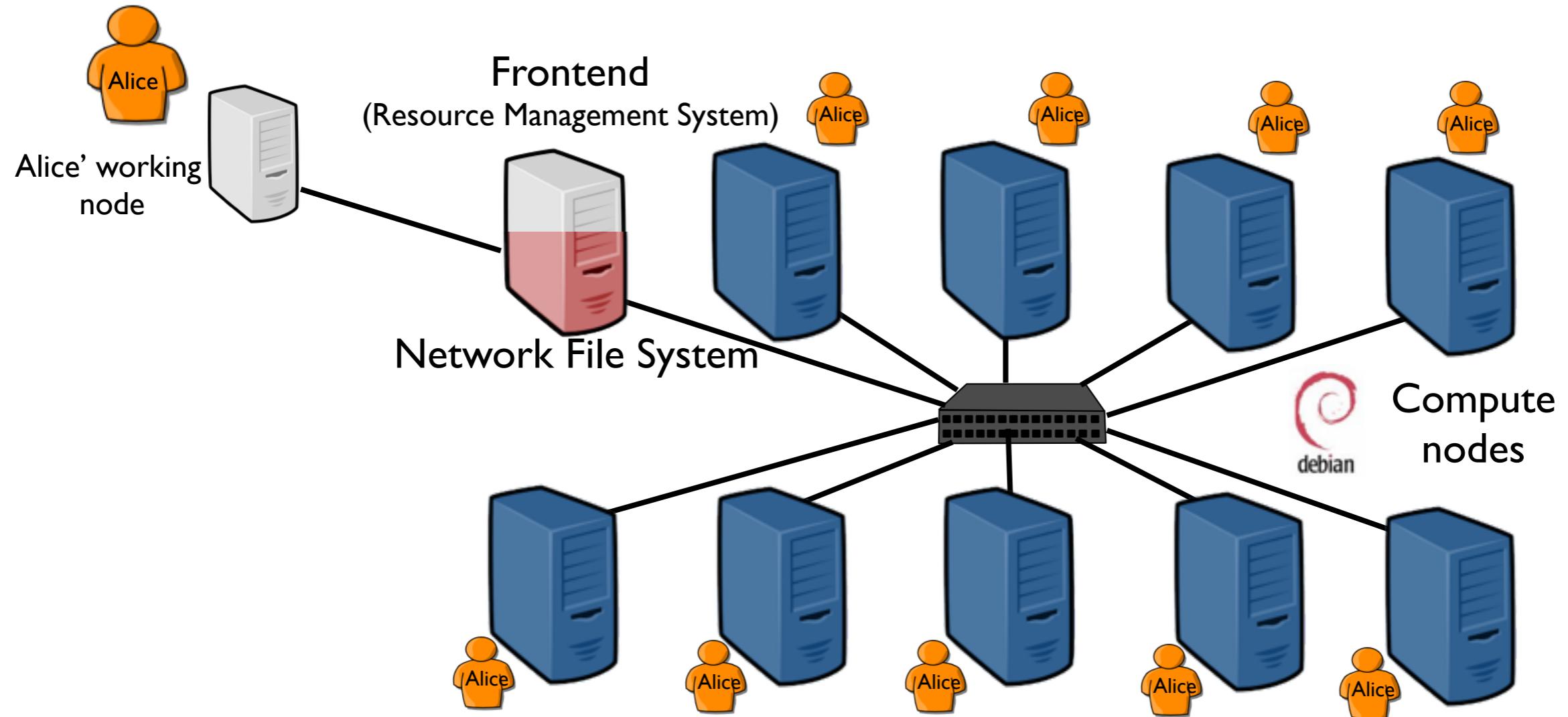
# Looking back...

- Network of Workstations 1990 / 20xx



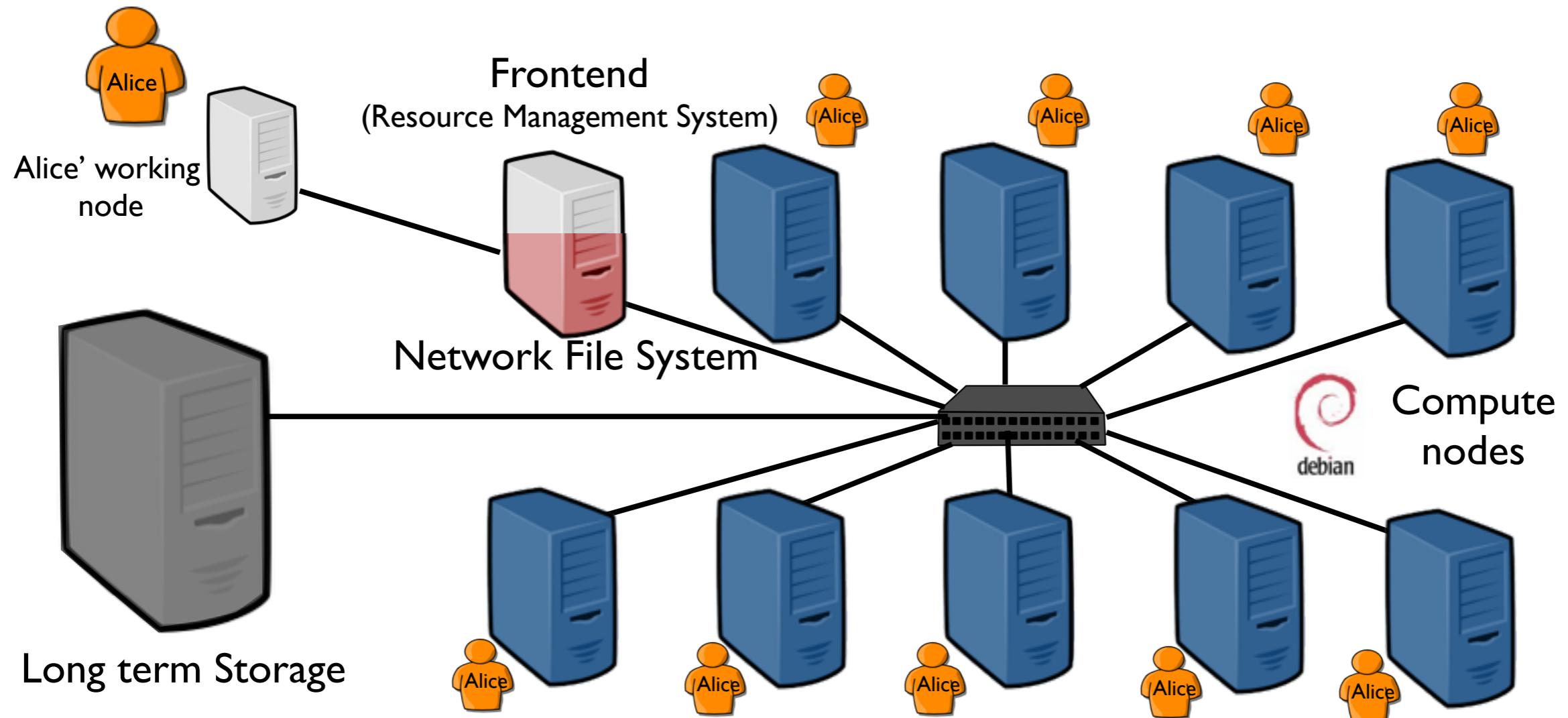
# Looking back...

- Network of Workstations 1990 / 20xx



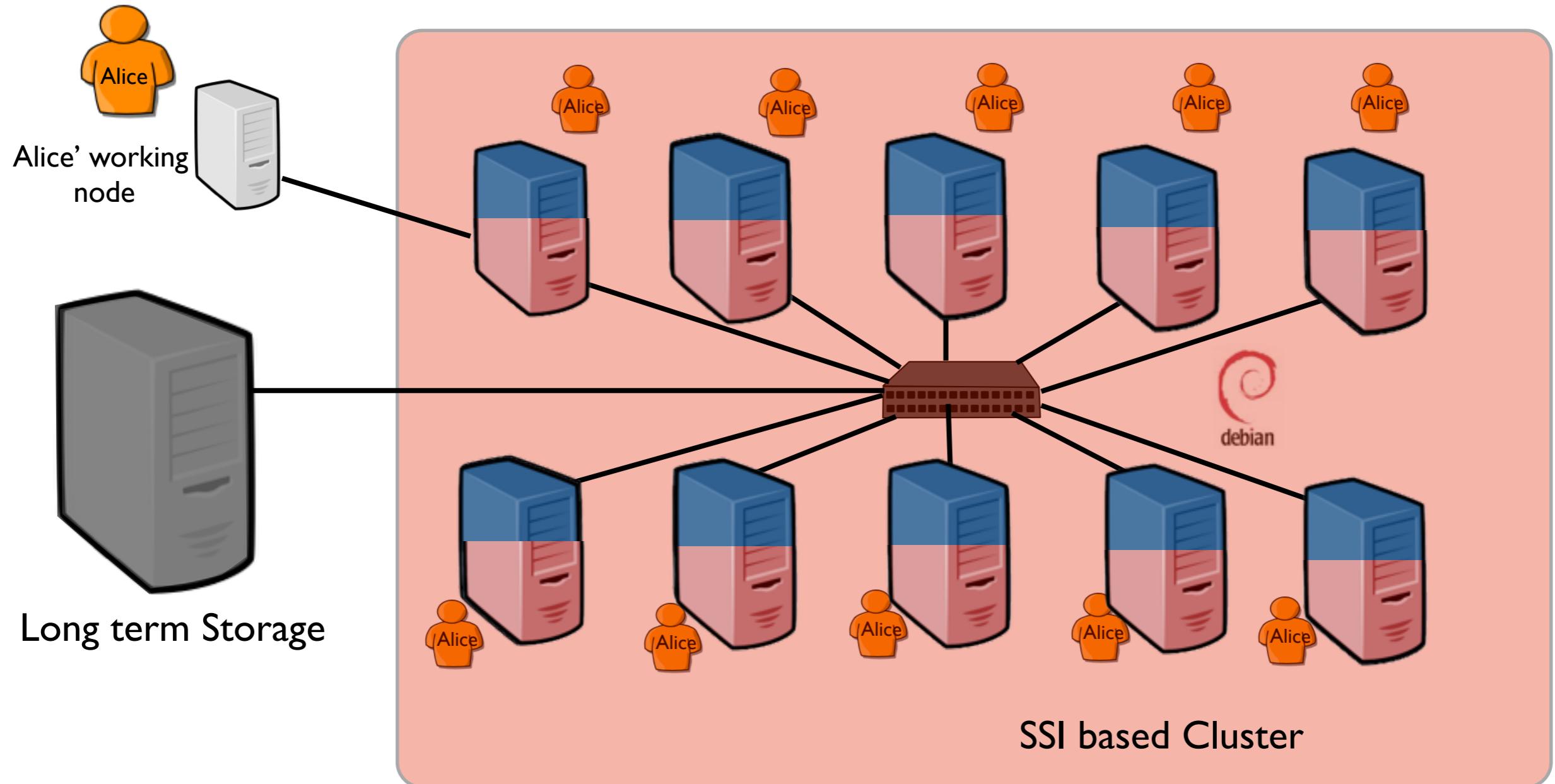
# Looking back...

- Network of Workstations 1990 / 20xx



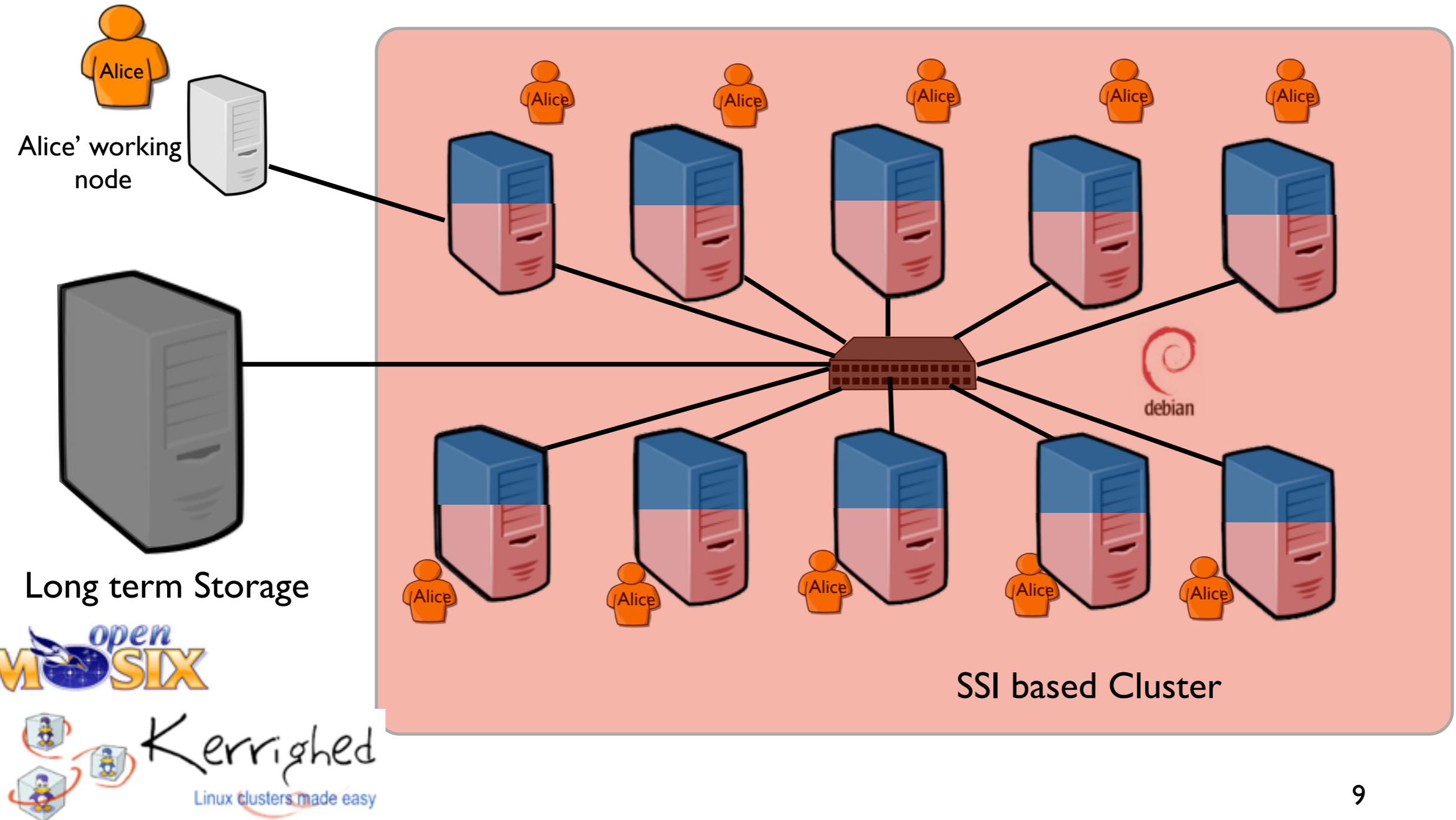
# Looking back...

- Network of Workstations 1990 / 20xx



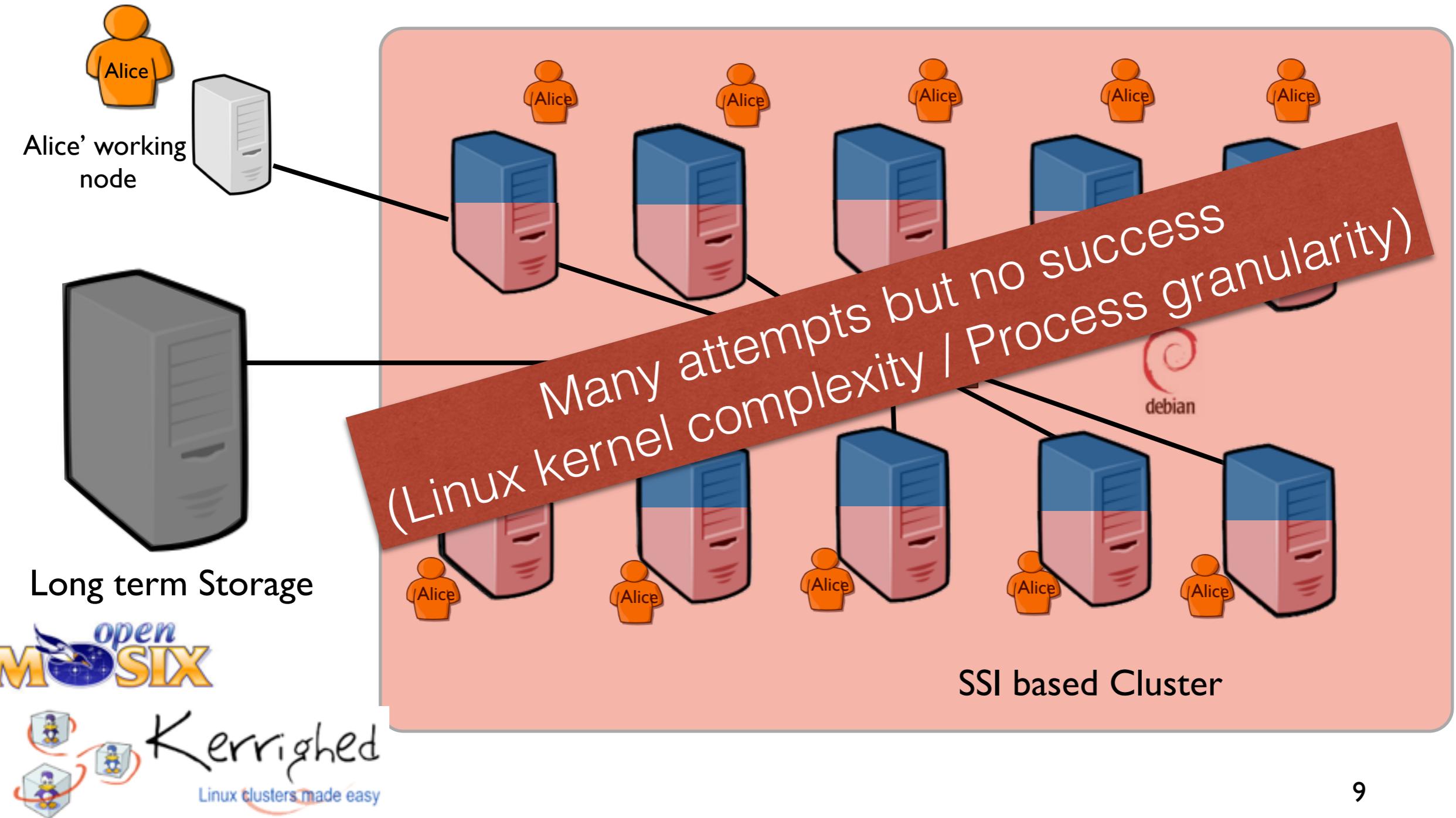
# Looking back...

- Network of Workstations 1990 / 20xx



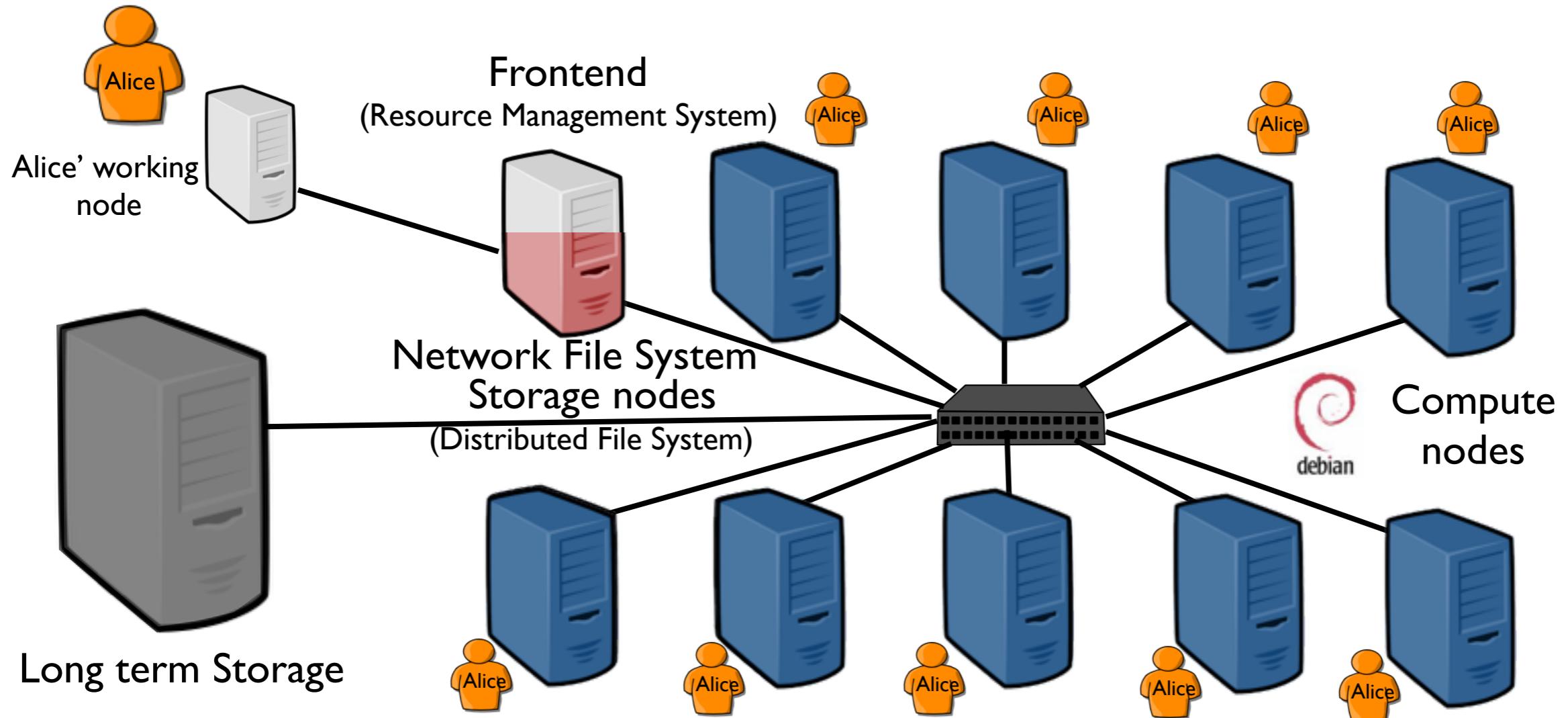
# Looking back...

- Network of Workstations 1990 / 20xx



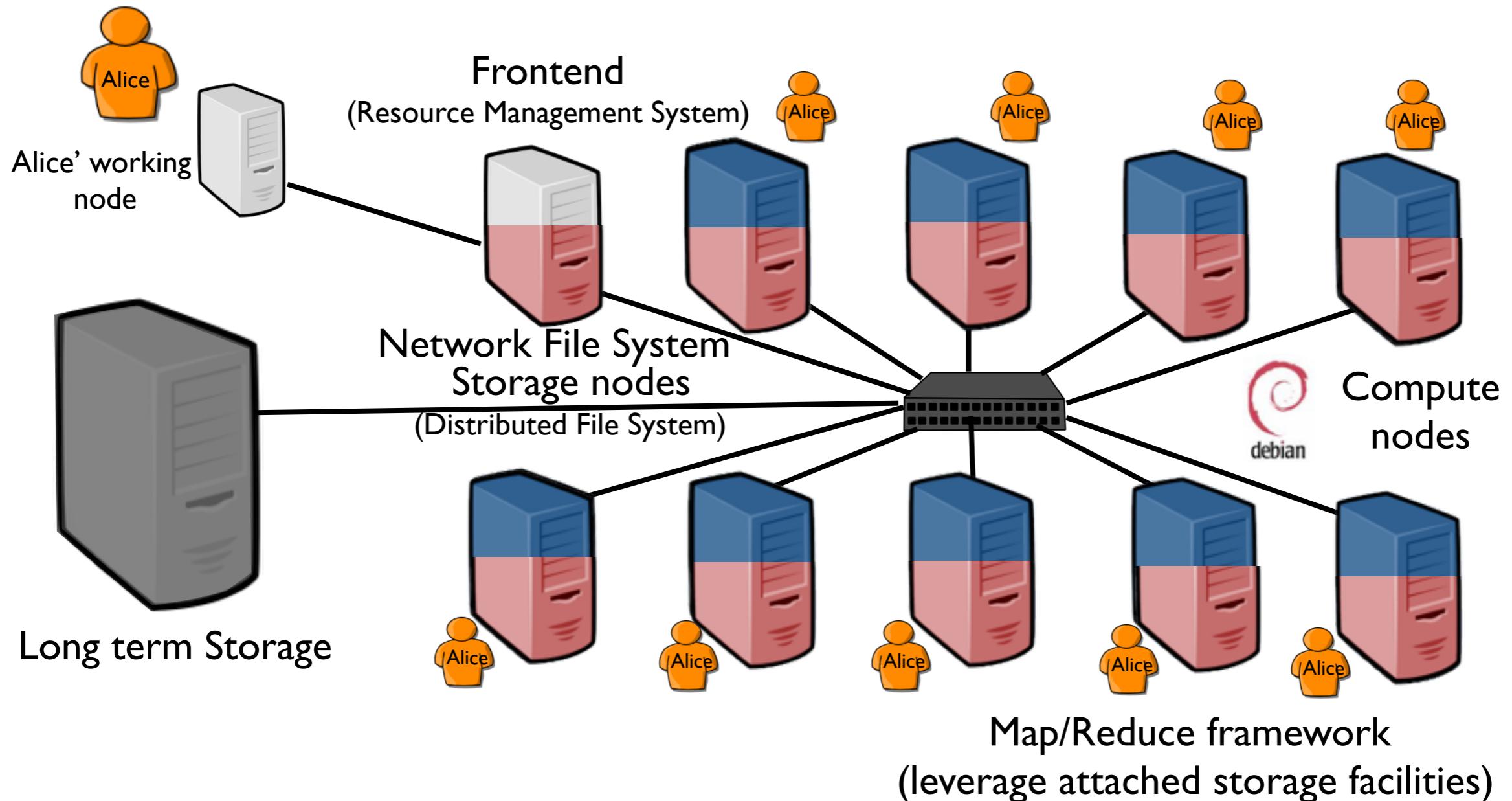
# Looking back...

- Network of Workstations 1990 / 20xx



# Looking back...

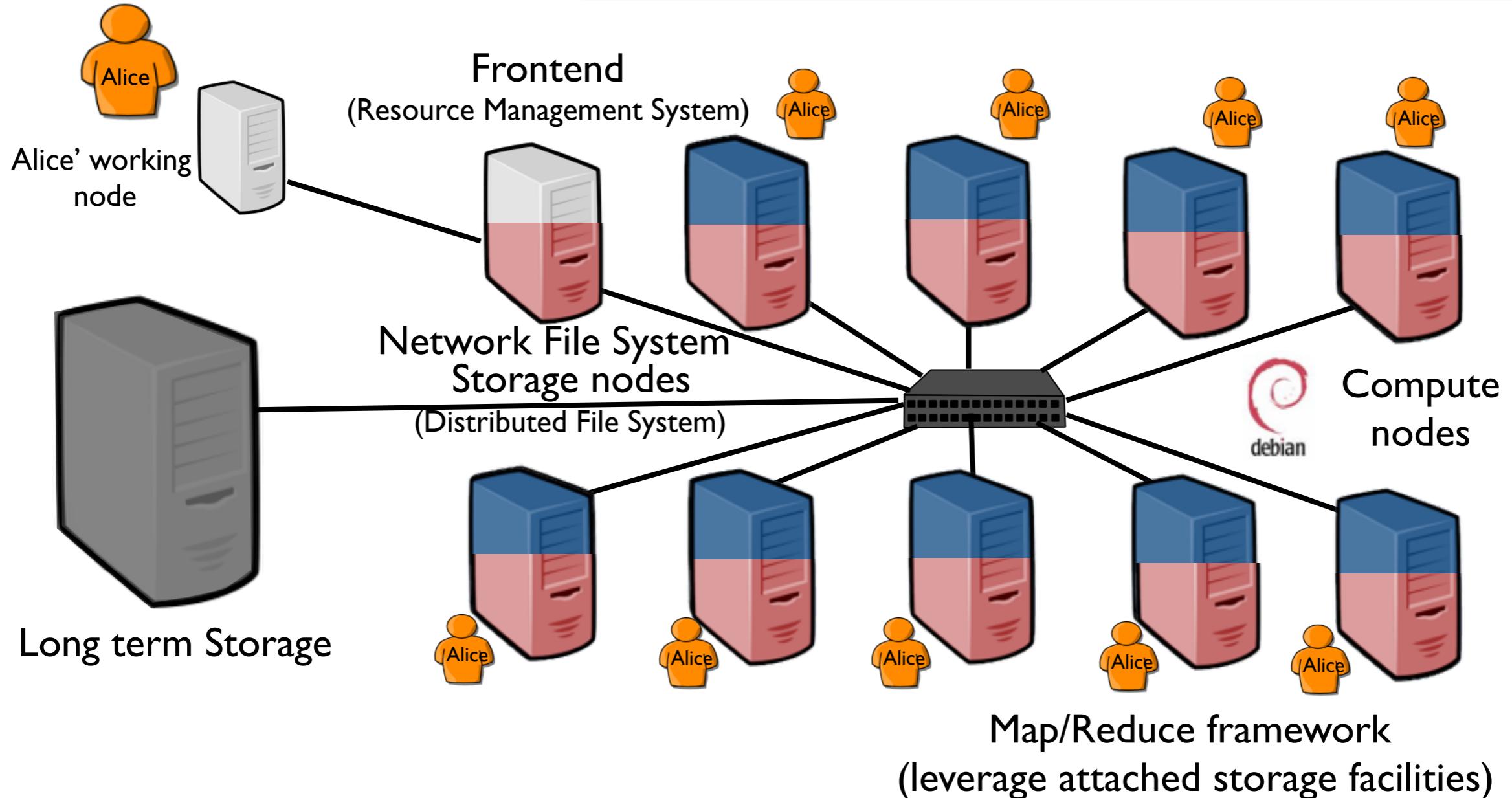
- Network of Workstations 1990 / 20xx



# Looking back...

- Network of Workstations 1990 / 20xx

## Lesson 4: S. Ibrahim



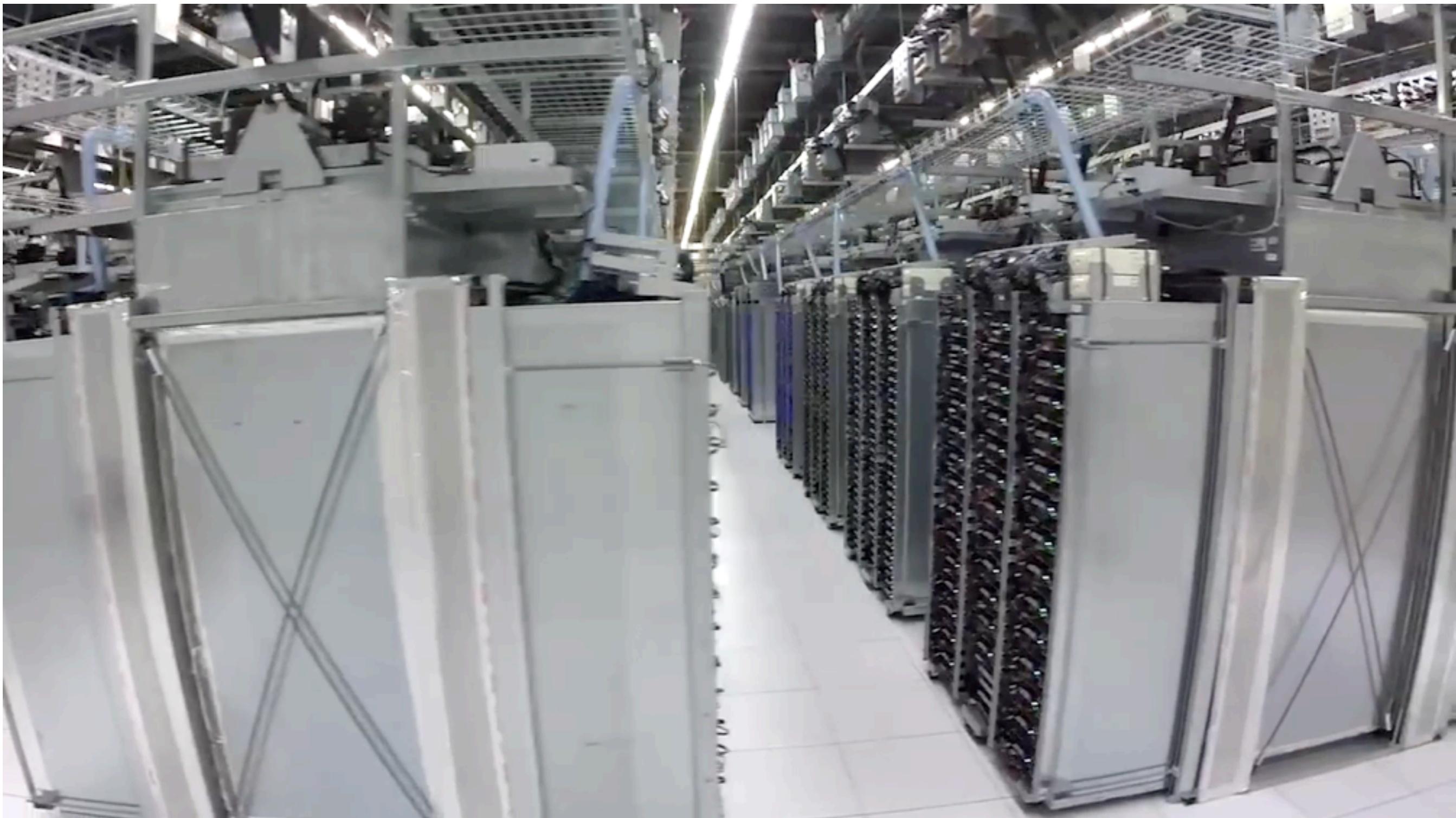
# Cluster in pictures

Some are famous...

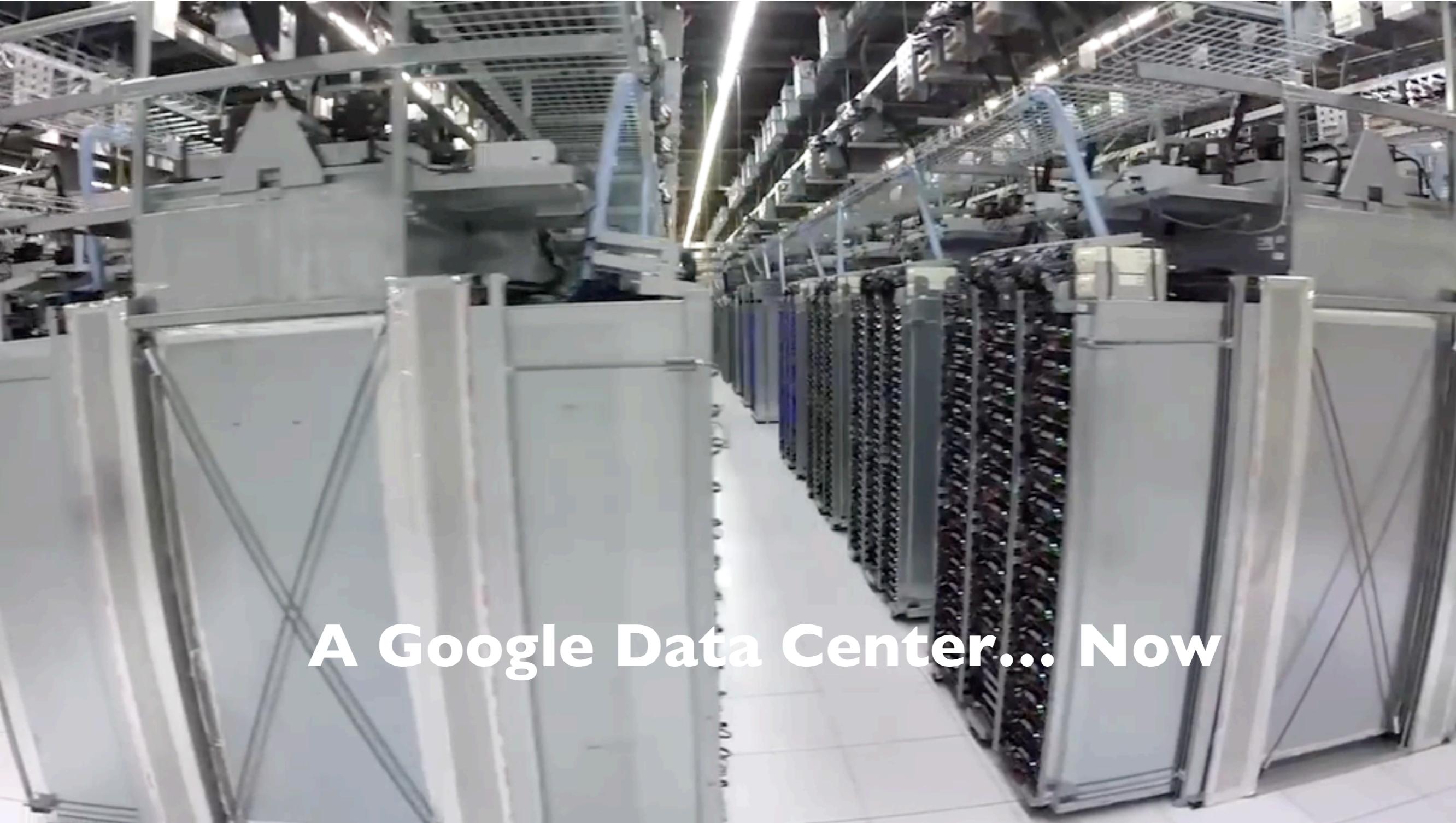


## Google cluster, 1998

# Cluster in pictures



# Cluster in pictures

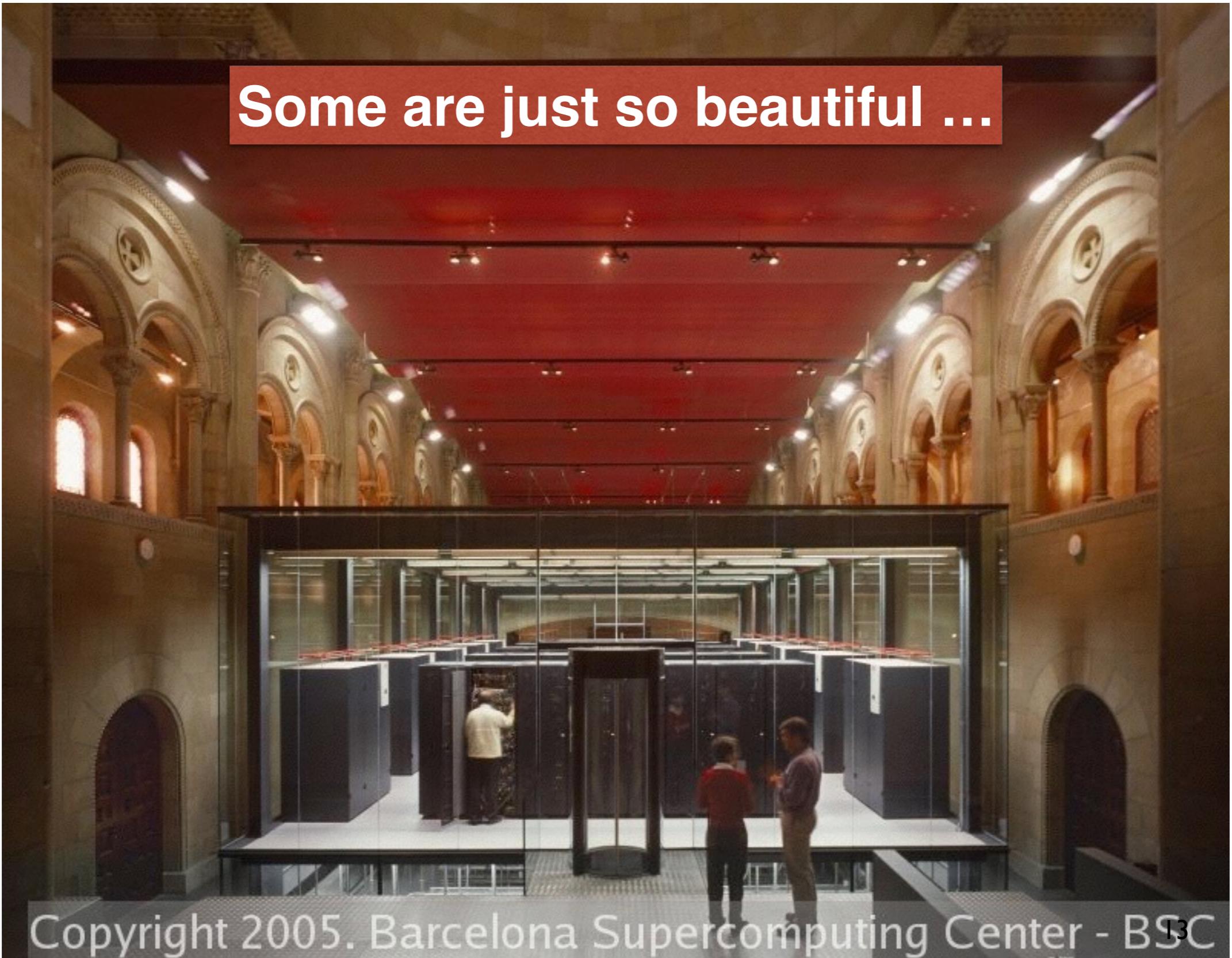


A Google Data Center... Now

# Cluster in pictures

# Cluster in pictures

Some are just so beautiful ...



# Cluster in pictures



Some are just so big ...

# Cluster in pictures

Some are just prospective ...



# Cluster in pictures

Some are just prospective ...



Lesson 3: G. Pierre

# Looking back ...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 201~~x~~

Exploit inactive time of machines interconnected to the Internet  
(Volunteers distributed computing)

Famous examples

SETI@home: *Search for Extra-Terrestrial Intelligence* (May 1999)  
BOINC: Berkeley Open Infrastructure for Network Computing

Clients/server model

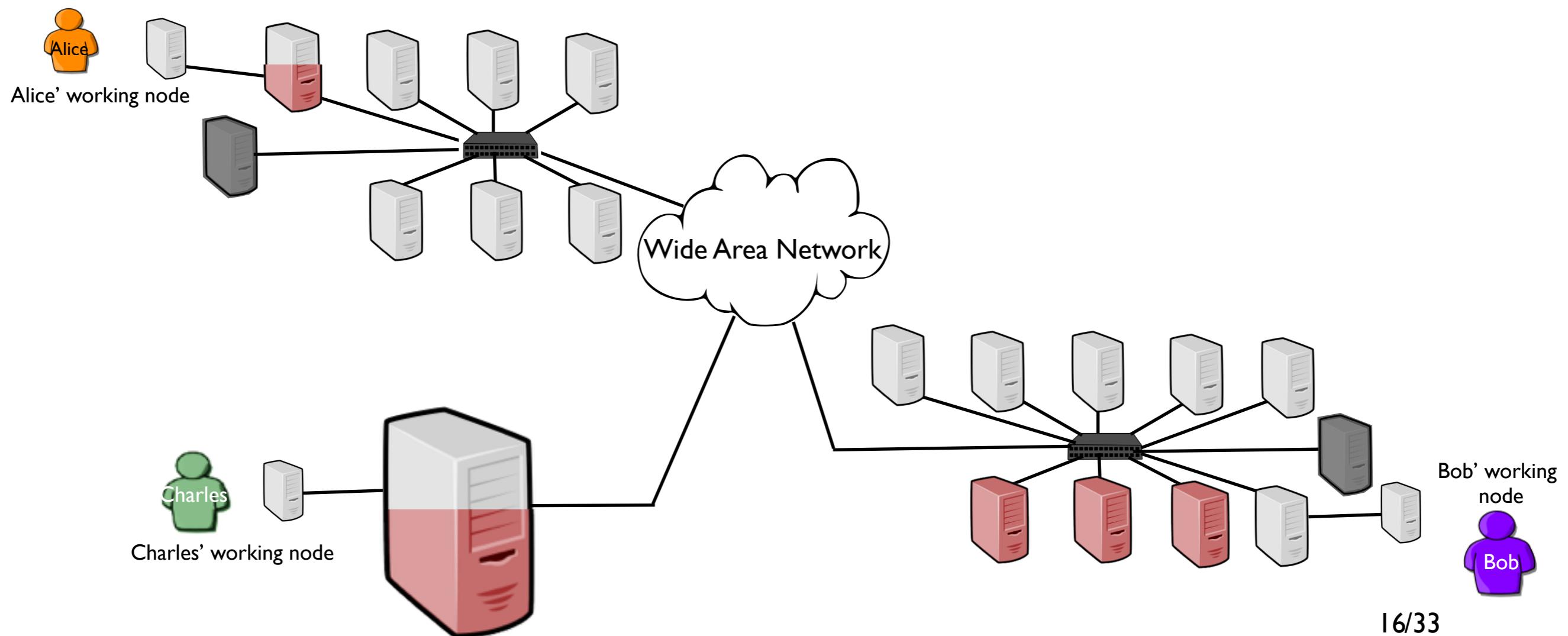
Security is the main issue

Strong limitations (SPMD model)



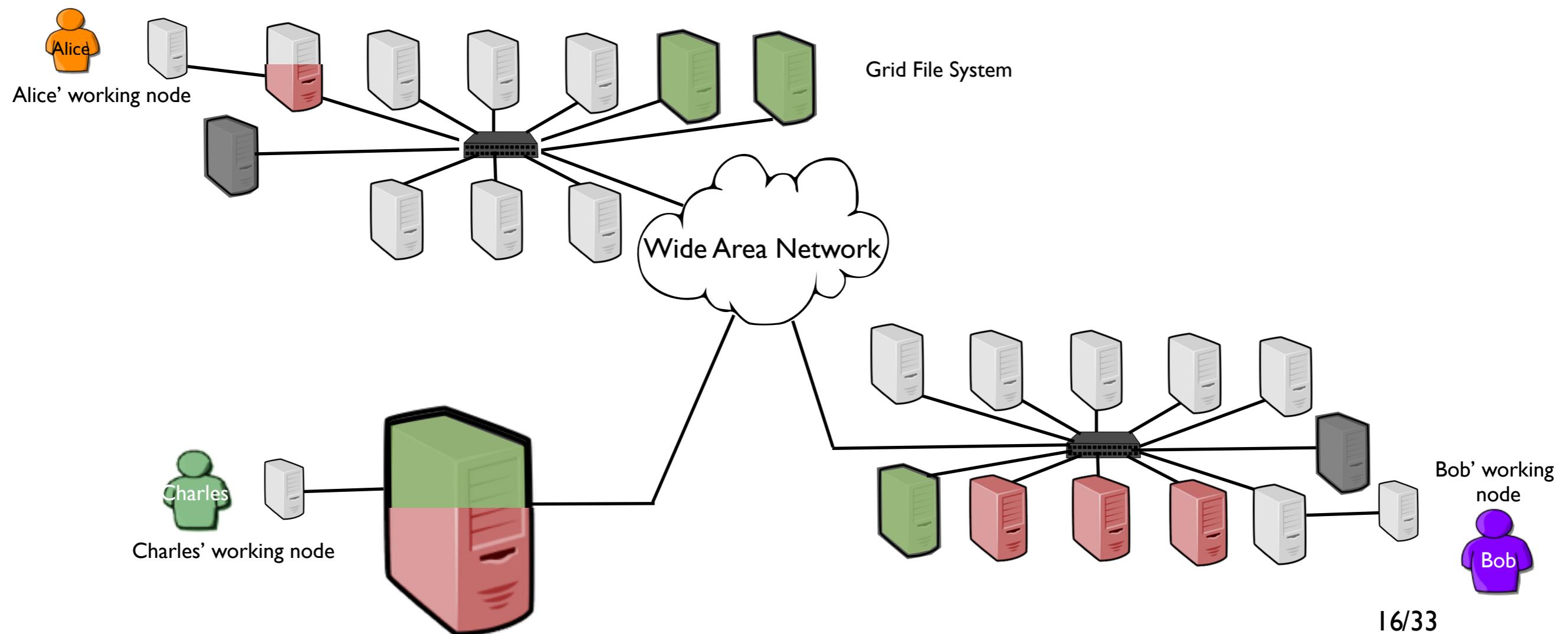
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



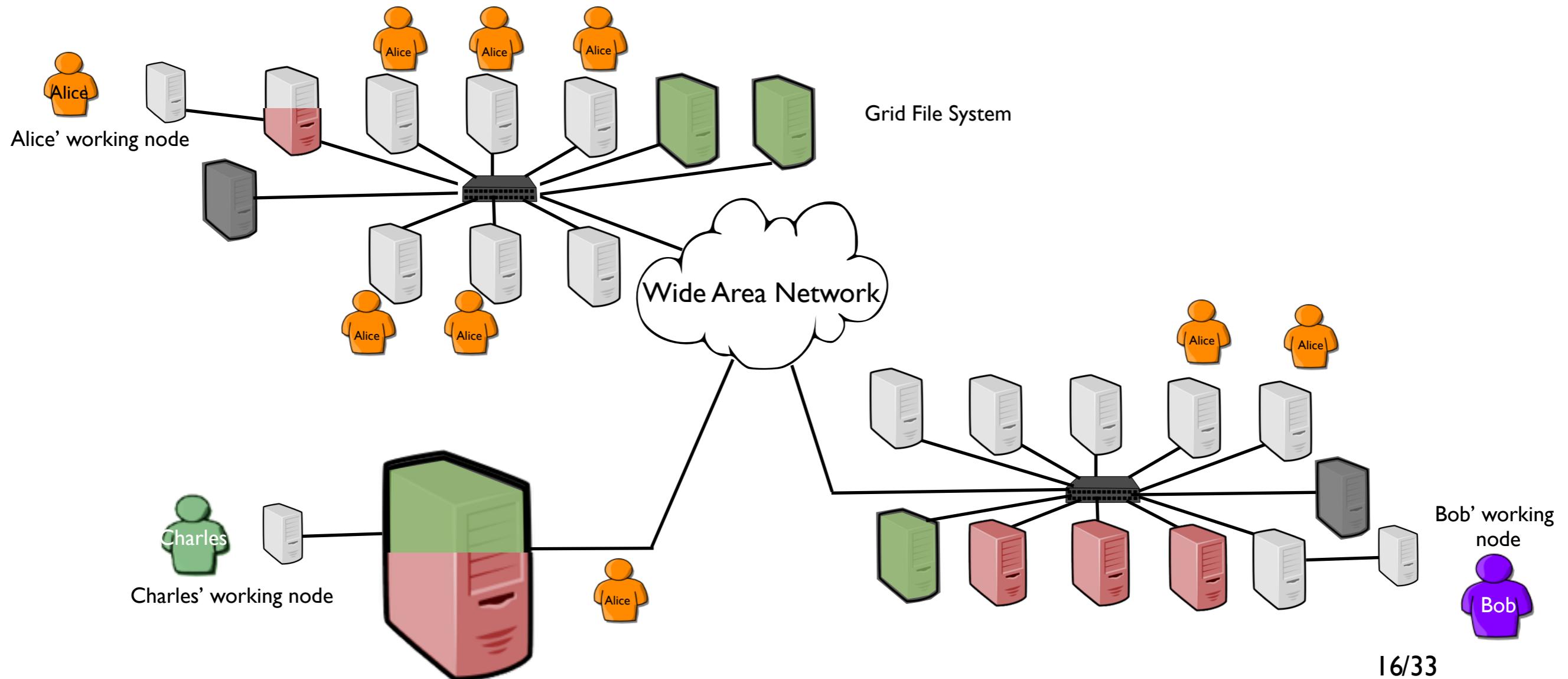
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



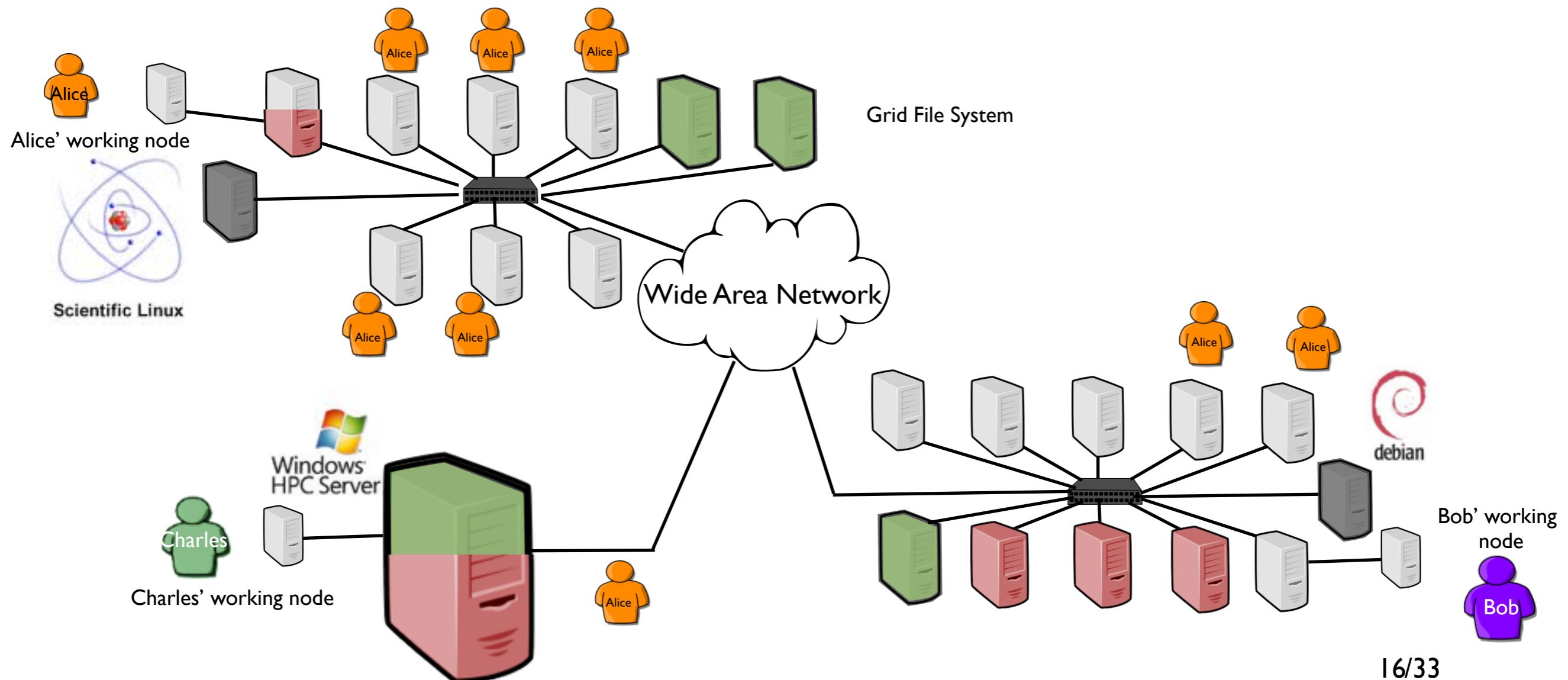
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



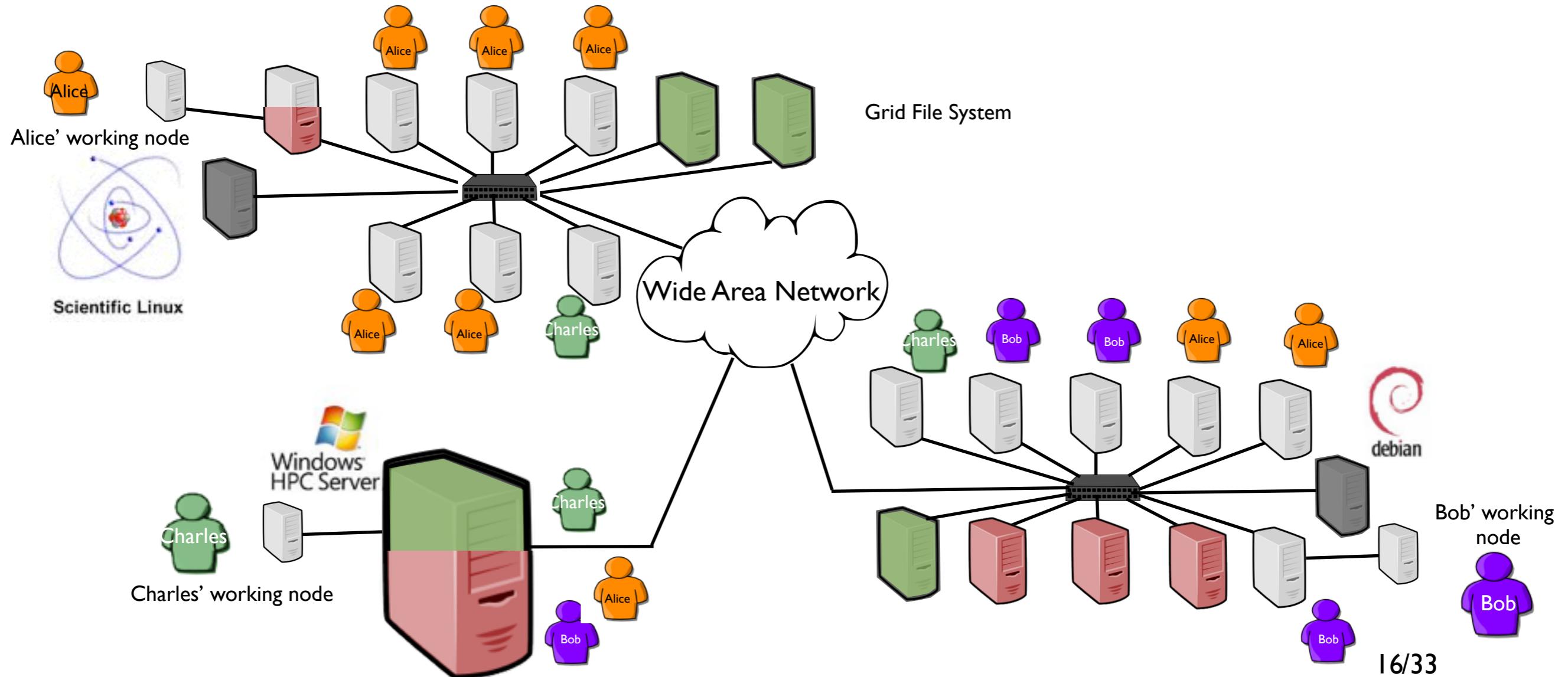
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



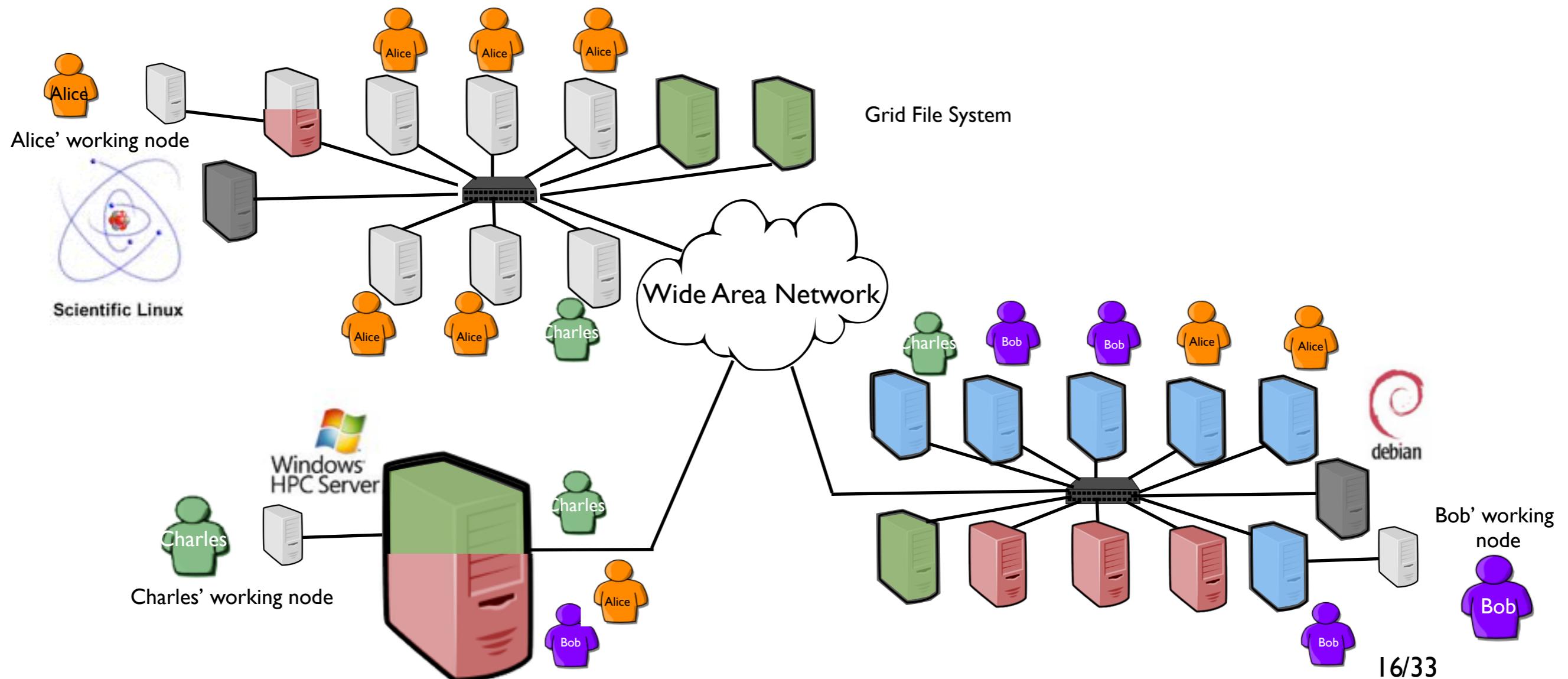
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



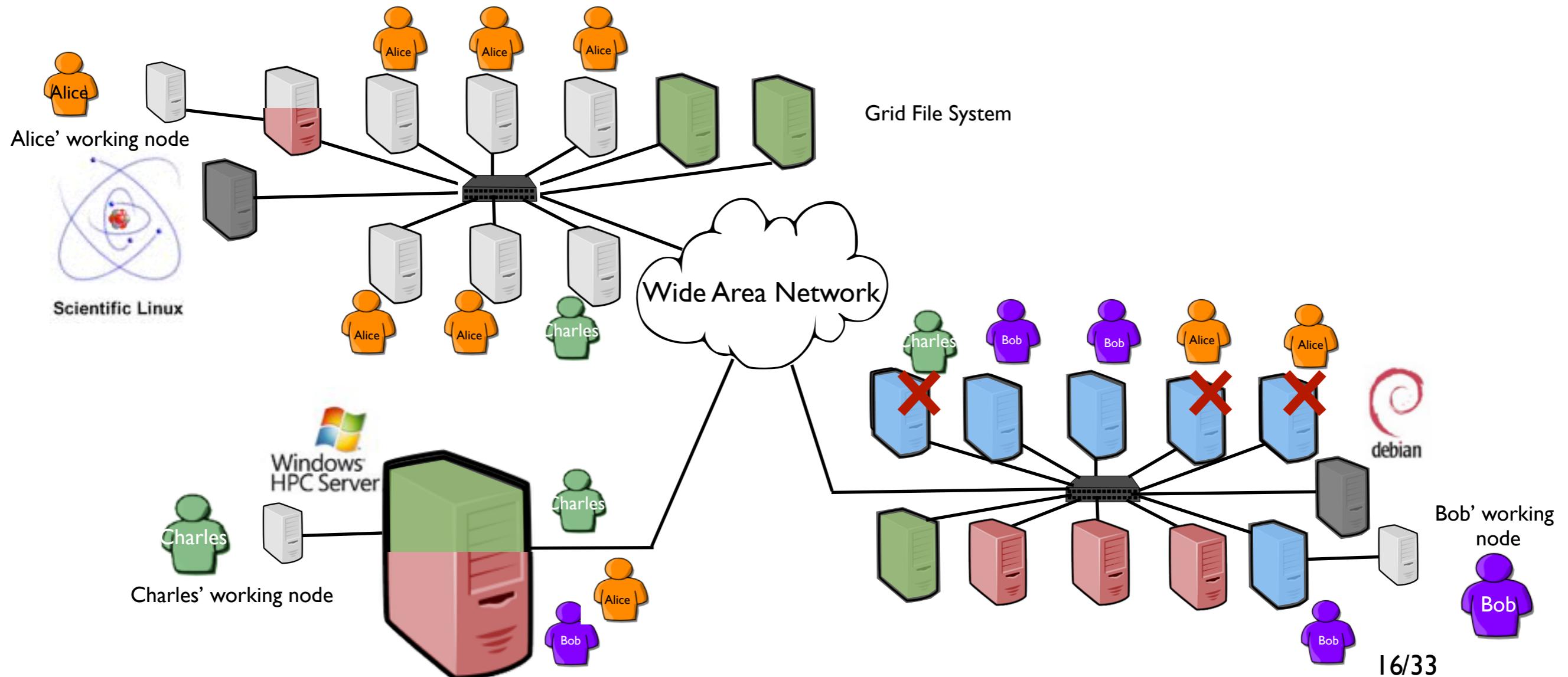
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



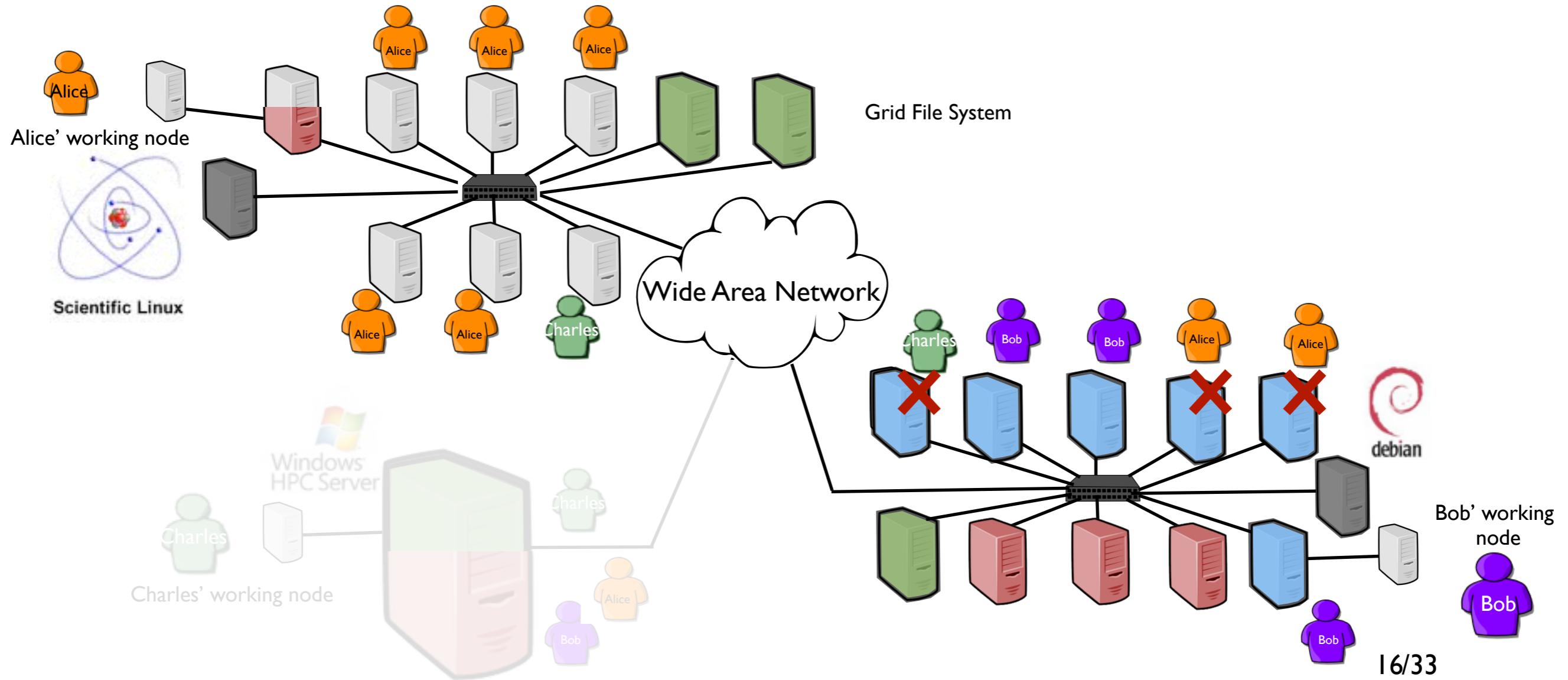
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



# Looking back...

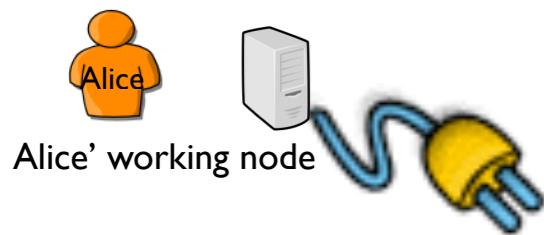
- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~



# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~

## What a Grid ! ? !



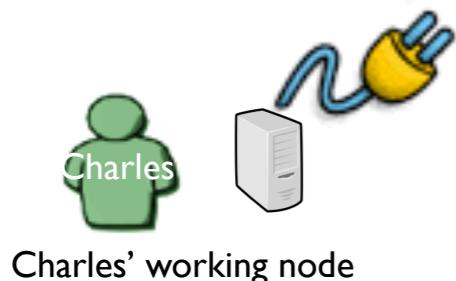
Resource booking (based on user's estimates)

Security concerns (job isolation)

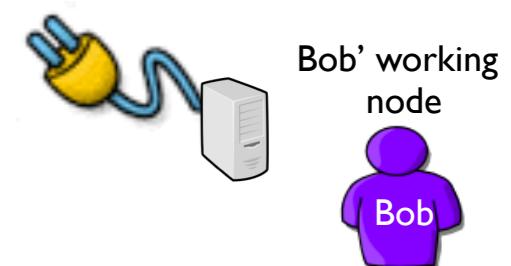
Heterogeneity concerns (hardware and software)

Scheduling limitations (a job cannot be easily relocated)

Fault tolerance issues



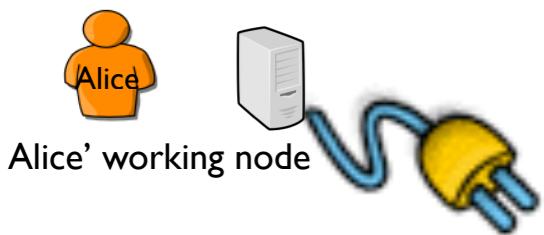
...



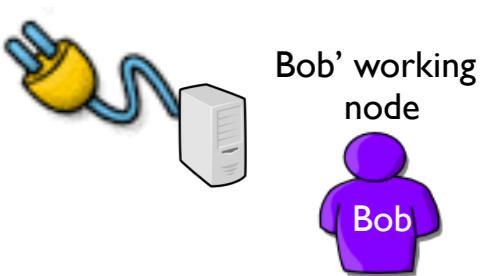
# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~

What a Grid ! ? !



A lot of progress has been done since the 90's and several proposals partially addressed these concerns.



# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~

## European Grid Infrastructure



EGI enables access to computing resources for European researchers from all fields of science, from high energy physics to humanities.

# Looking back...

- Network of Workstations 1990 / 20~~xx~~
- Desktop 1998 / 20~~xx~~
- Grid 1998 / 20~~xx~~

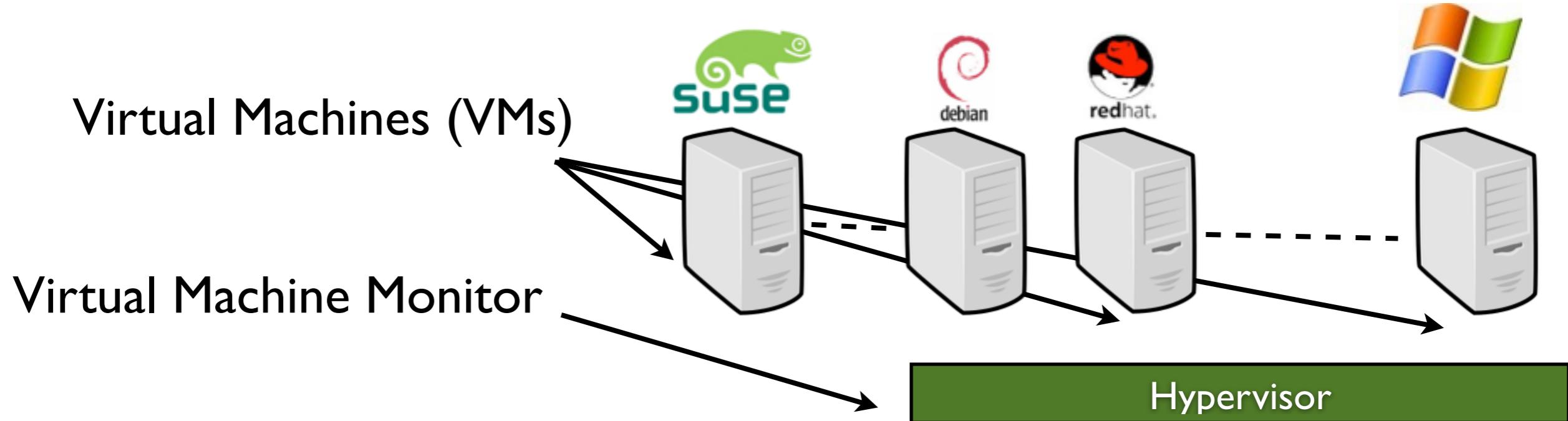
## European Grid Infrastructure



EGI enables access to computing resources for European researchers from all fields of science, from high energy physics to humanities.

# Looking back...

- System virtualization: One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)



“A **virtual machine** (VM) provides a faithful implementation of a physical processor’s hardware running in a protected and isolated environment.

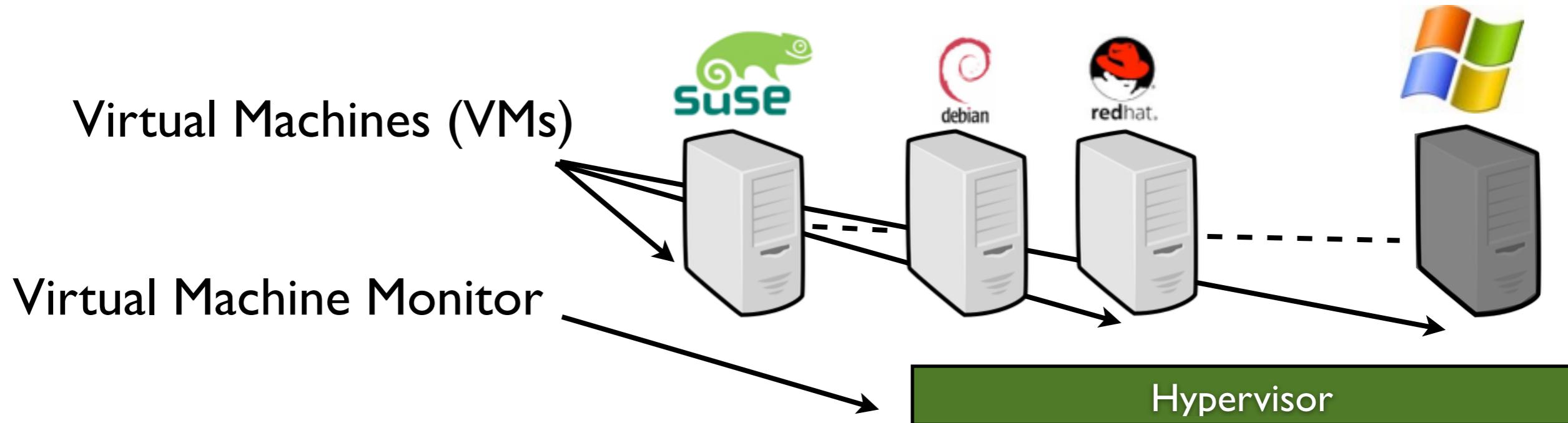
Virtual machines are created by a software layer called the **virtual machine monitor** (VMM) that runs as a privileged task on a physical processor.”



Physical Machine (PM)

# Looking back...

- System virtualization: One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)



“A **virtual machine** (VM) provides a faithful implementation of a physical processor’s hardware running in a protected and isolated environment.

Virtual machines are created by a software layer called the **virtual machine monitor** (VMM) that runs as a privileged task on a physical processor.”



Physical Machine (PM)

# A BIT OF HISTORY

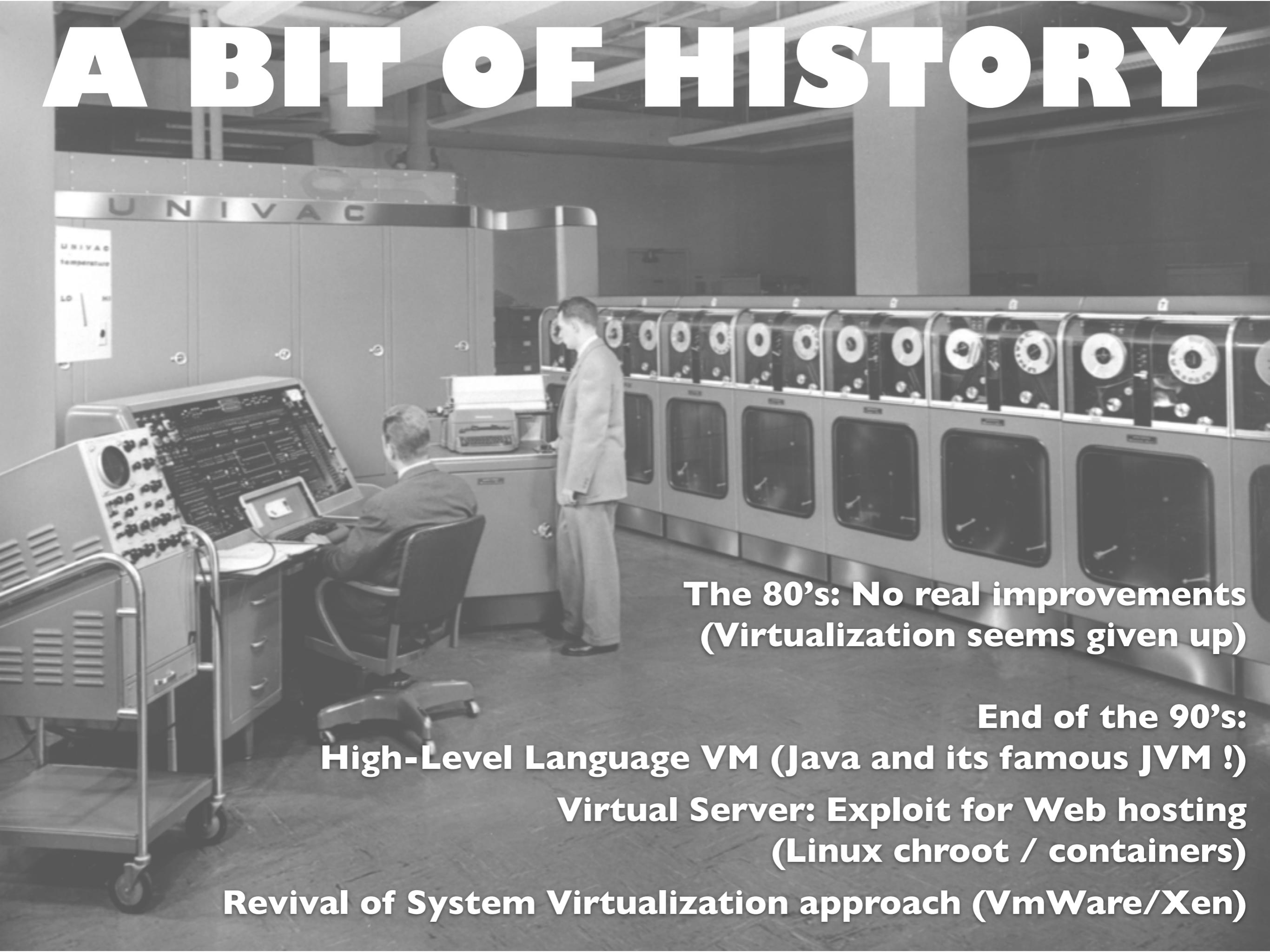


**Proposed in the 60's by IBM  
More than 70 publications between 66 and 73**

**“Virtual Machines have finally arrived. Dismissed for a number of years as merely academic curiosities, they are now seen as cost-effective techniques for organizing computer systems resources to provide extraordinary system flexibility and support for certain unique applications” .**

**Goldberg, Survey of Virtual Machine Research, 1974**

# A BIT OF HISTORY



**The 80's: No real improvements  
(Virtualization seems given up)**

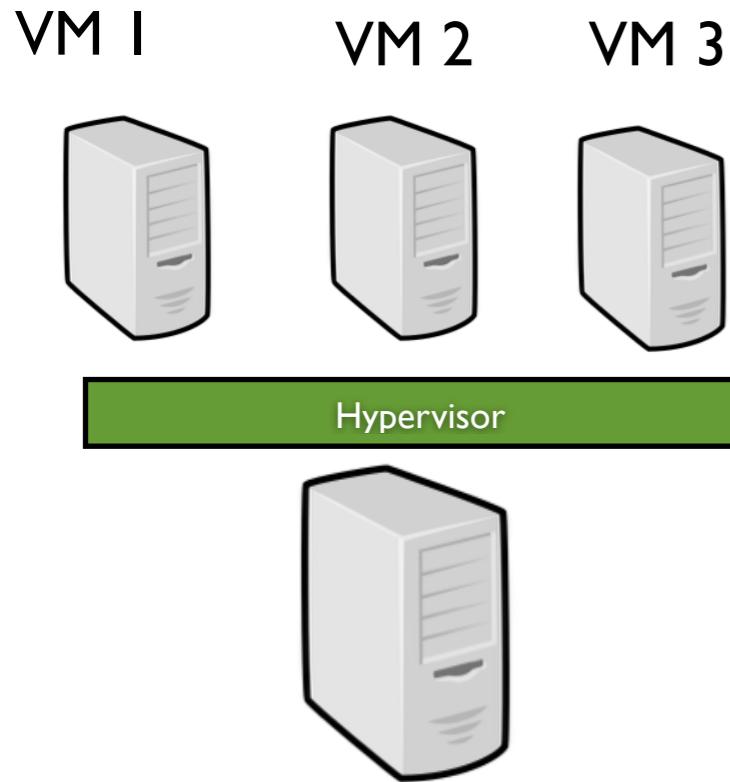
**High-Level Language VM (Java and its famous JVM !)**

**Virtual Server: Exploit for Web hosting  
(Linux chroot / containers)**

**Revival of System Virtualization approach (VmWare/Xen)**

# Looking back...

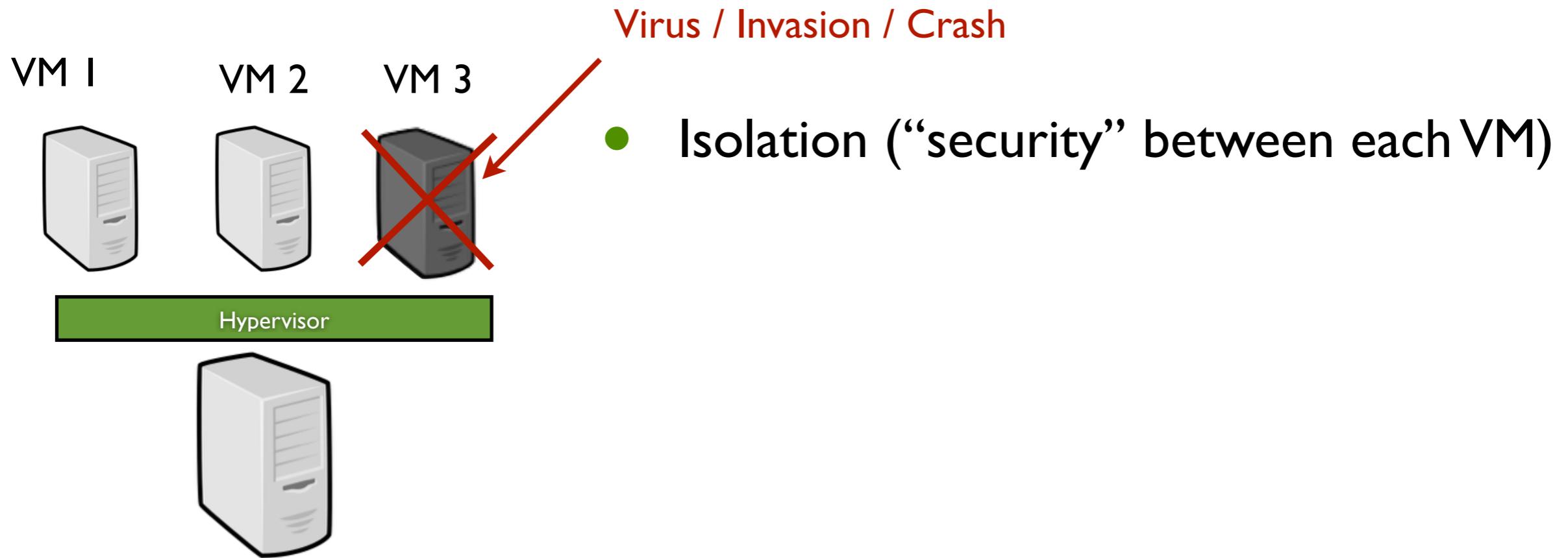
- System virtualization: a great sandbox



- Isolation (“security” between each VM)

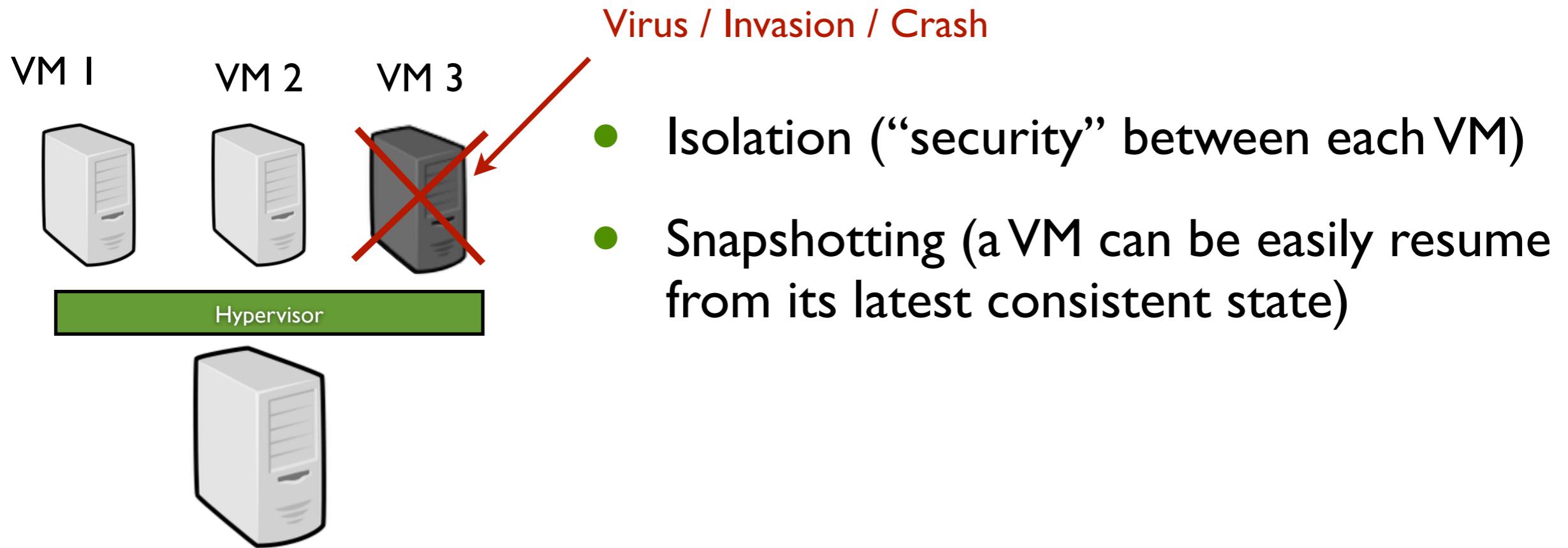
# Looking back...

- System virtualization: a great sandbox



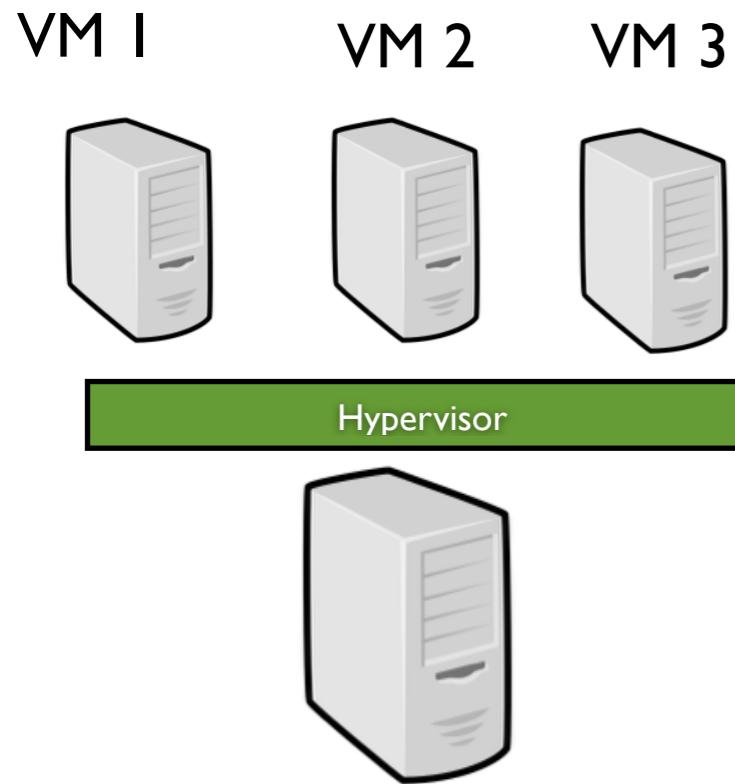
# Looking back...

- System virtualization: a great sandbox



# Looking back...

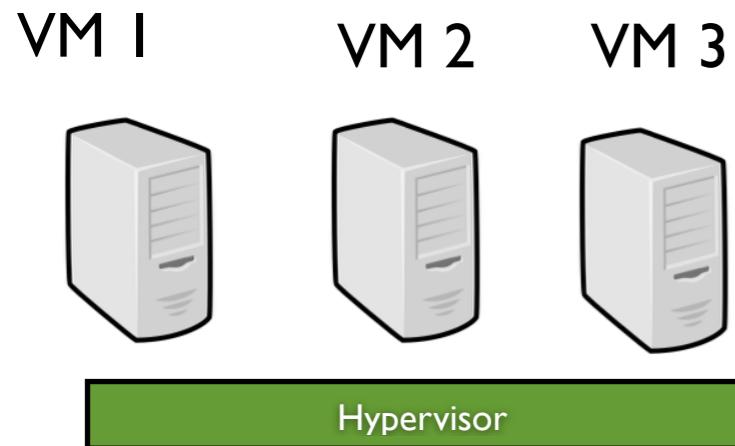
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

# Looking back...

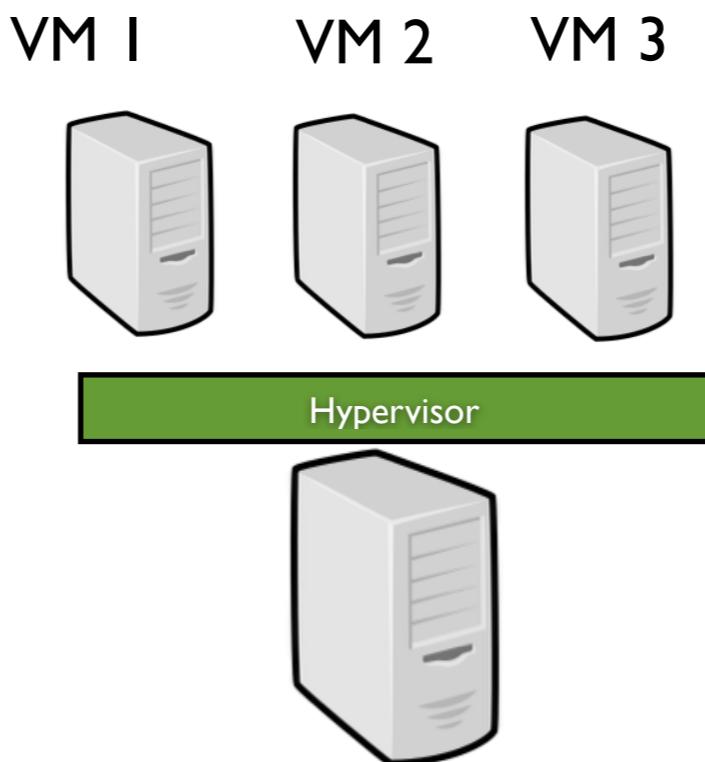
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

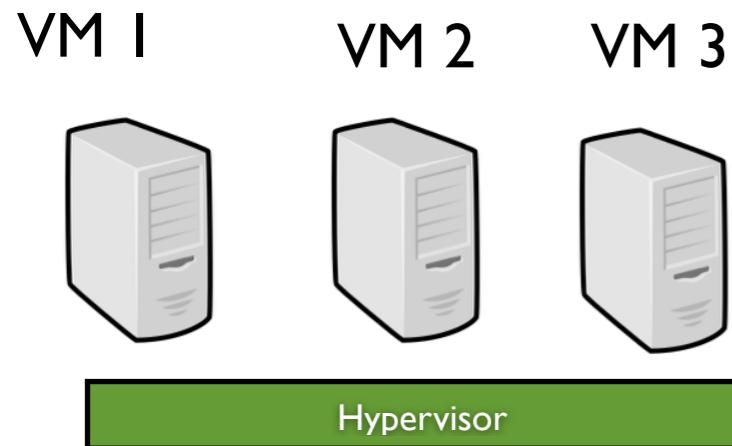


- Suspend/Resume



# Looking back...

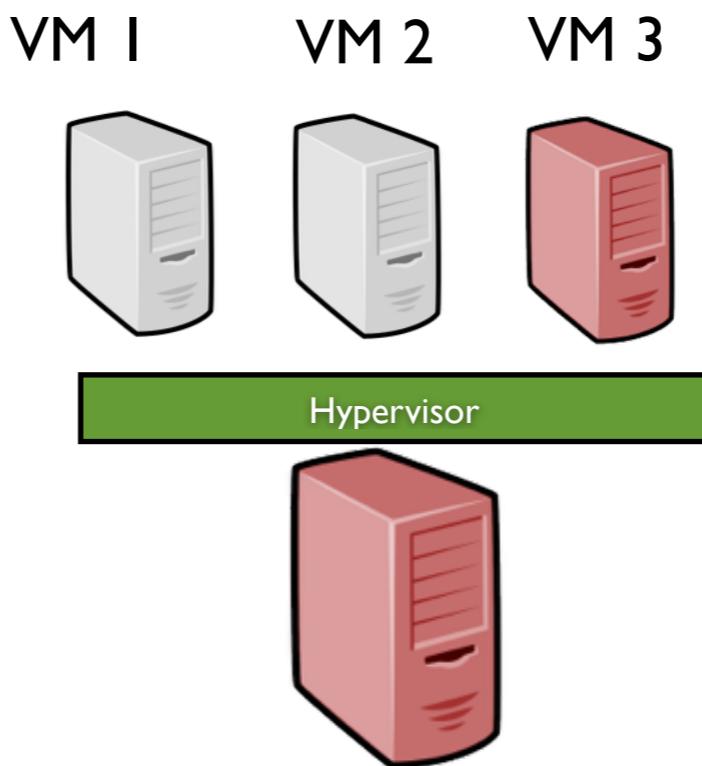
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

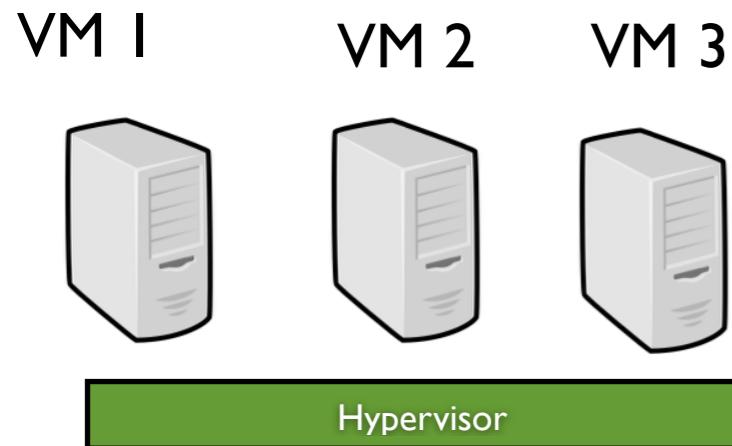


- Suspend/Resume

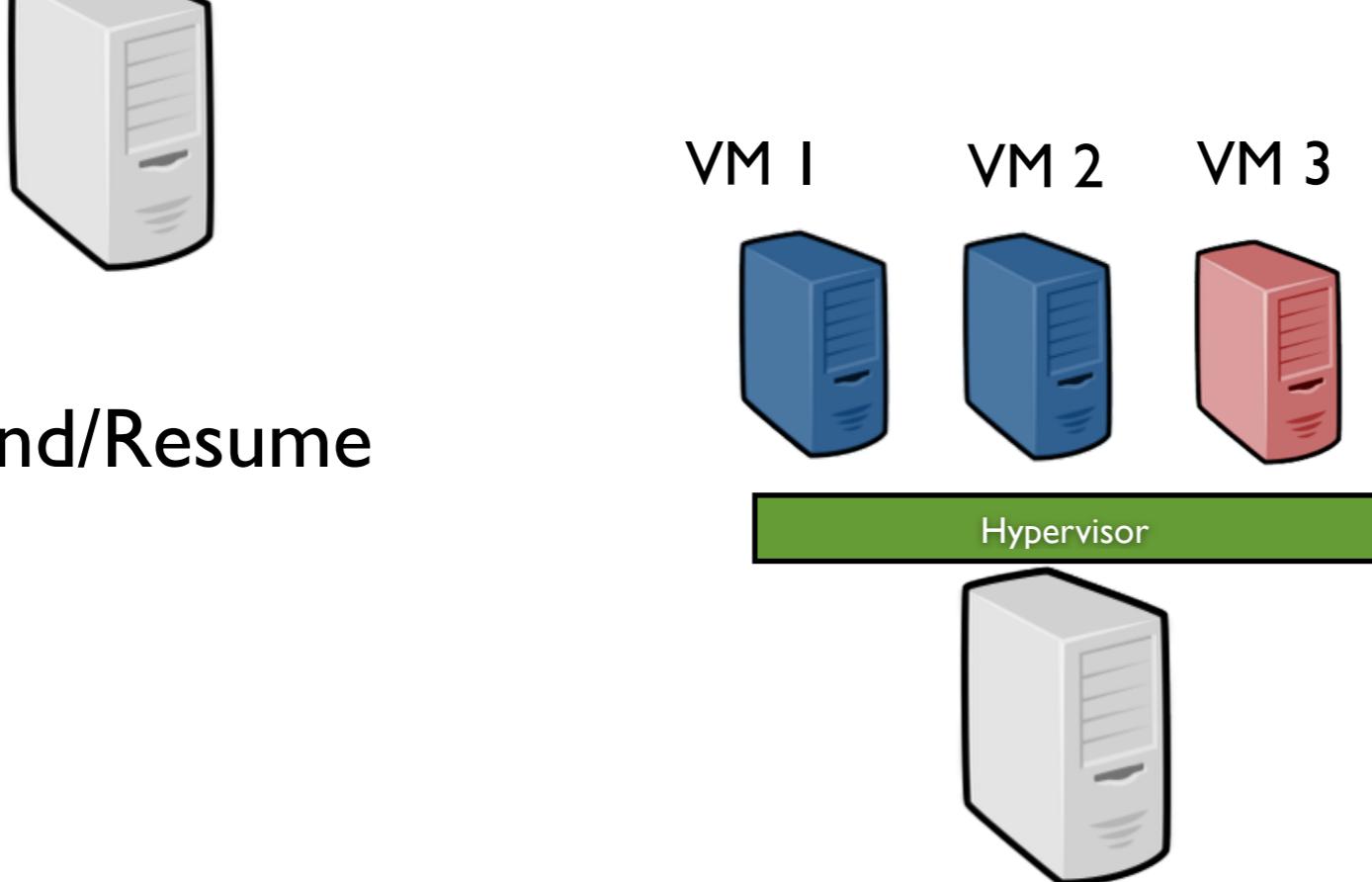


# Looking back...

- System virtualization: a great sandbox



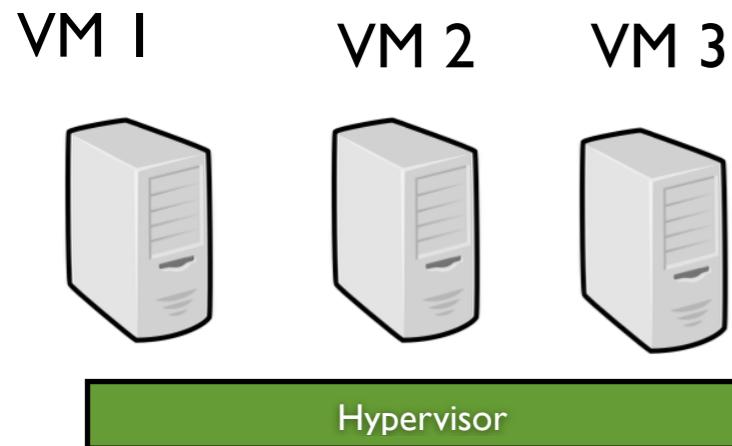
- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)



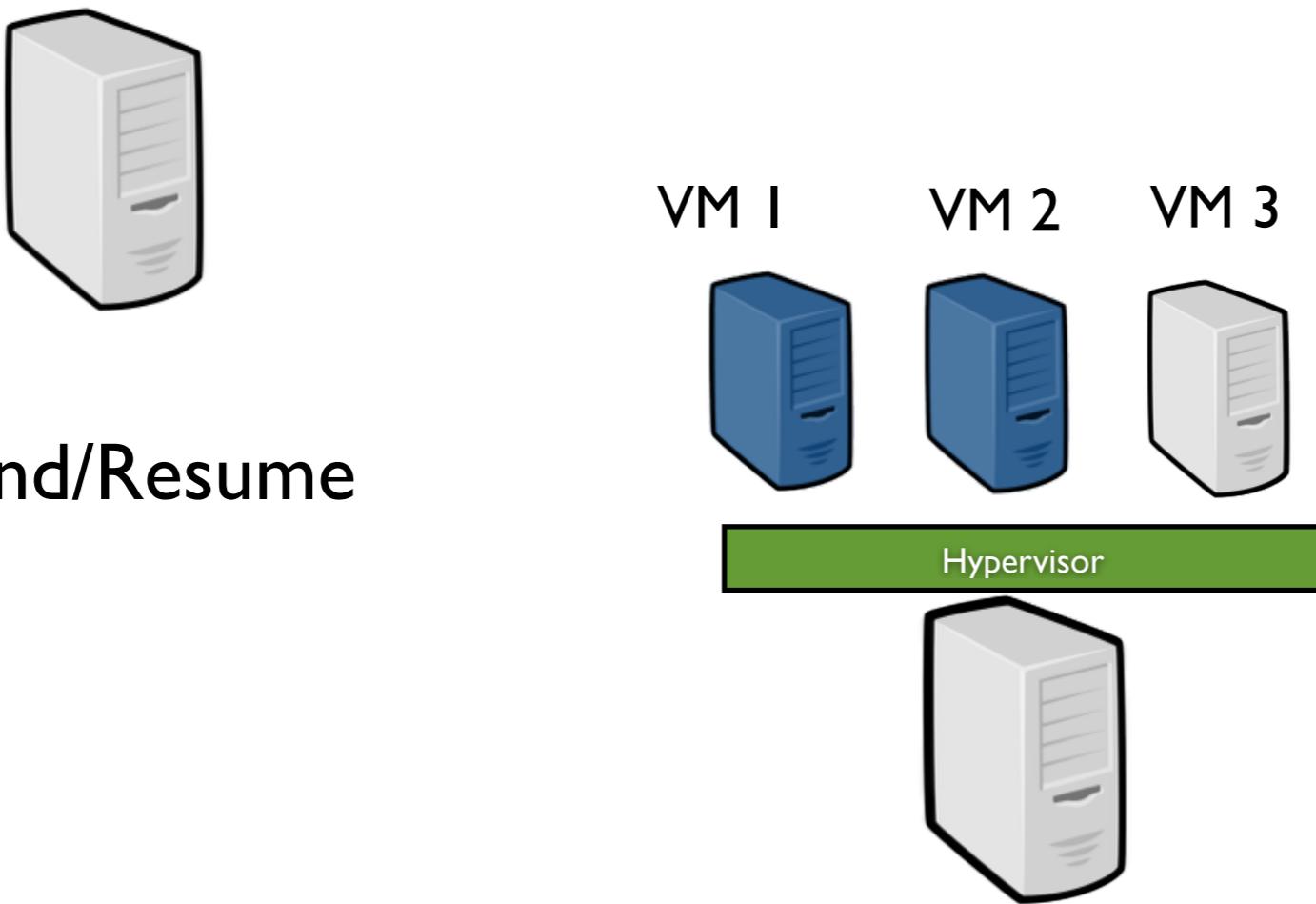
- Suspend/Resume

# Looking back...

- System virtualization: a great sandbox



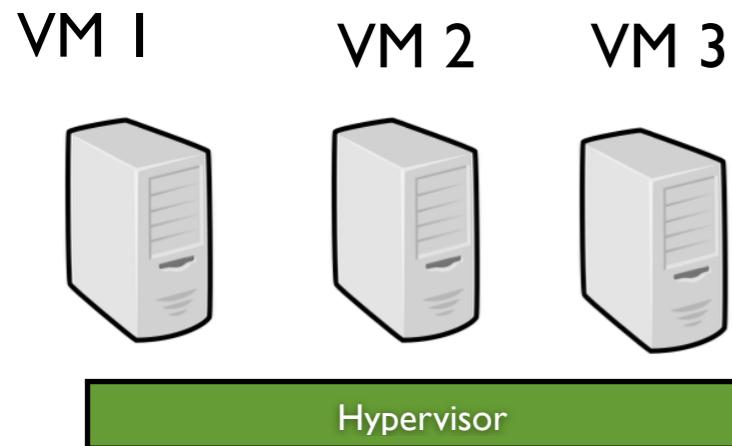
- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)



- Suspend/Resume

# Looking back...

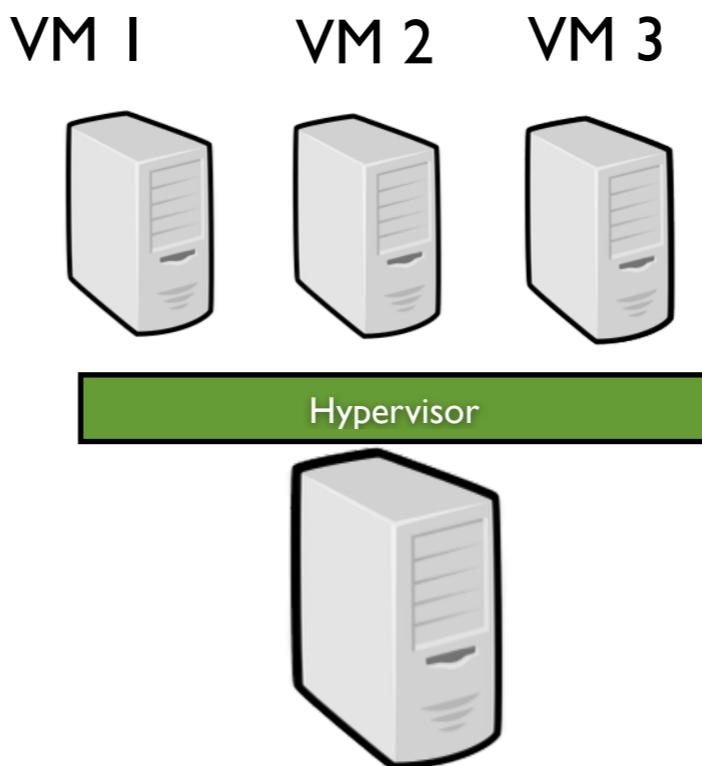
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

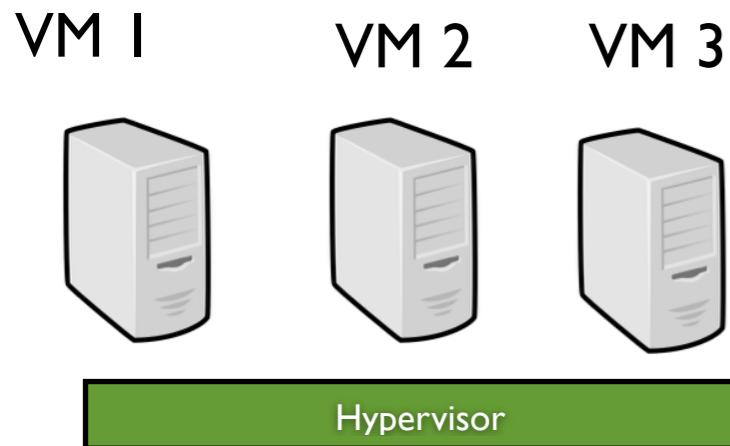


- Suspend/Resume



# Looking back...

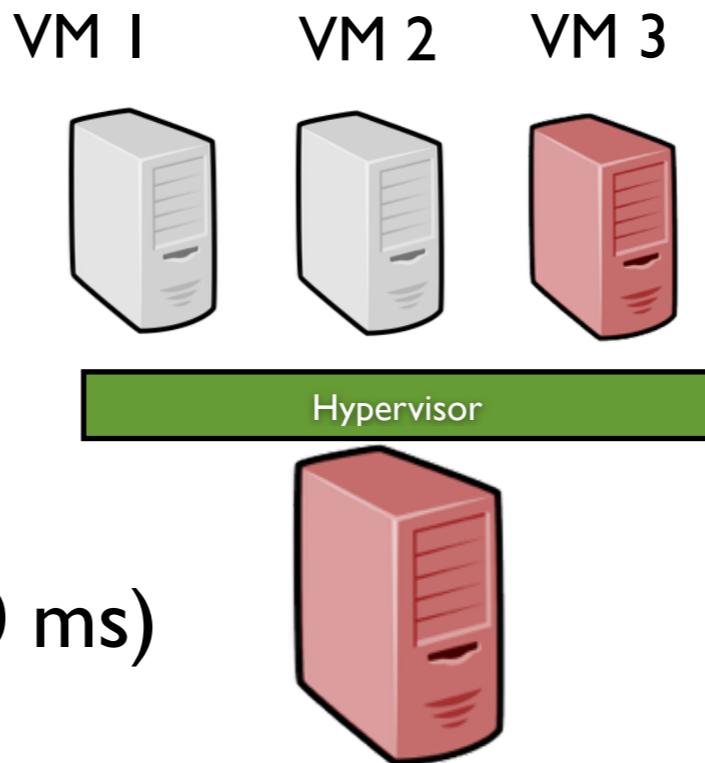
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

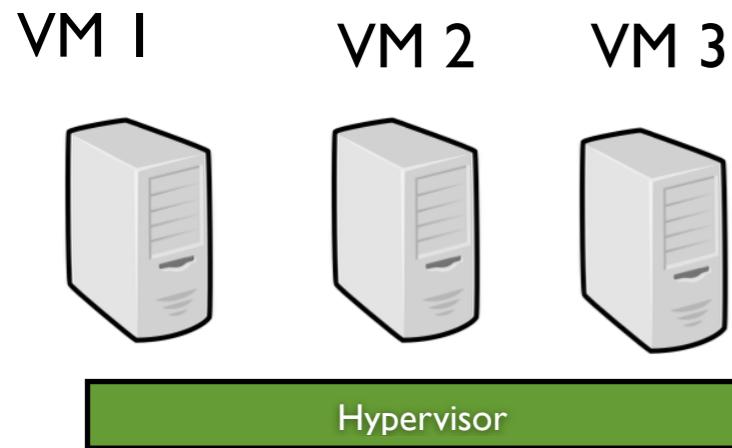


- Suspend/Resume
- Live migration  
(negligible downtime ~ 60 ms)  
Post/Pre Copy



# Looking back...

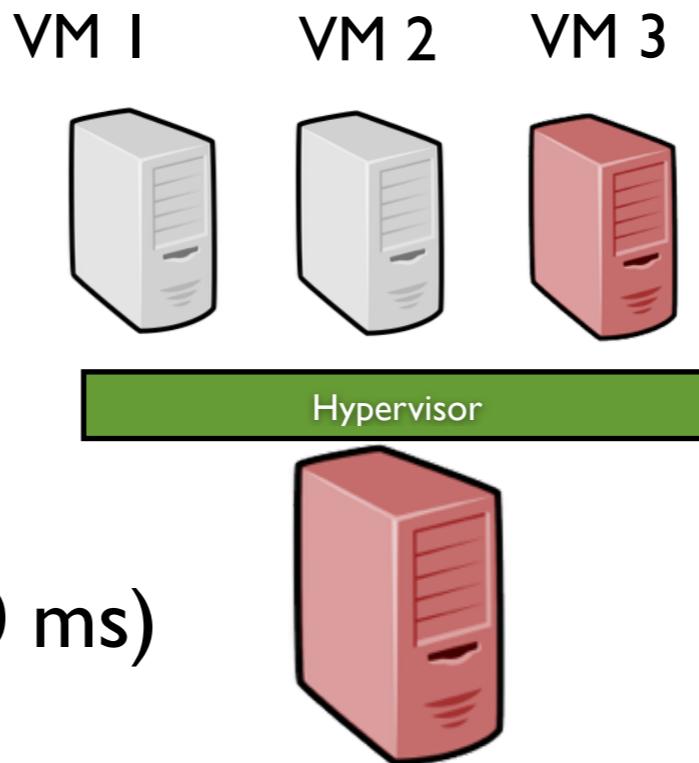
- System virtualization: a great sandbox



- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

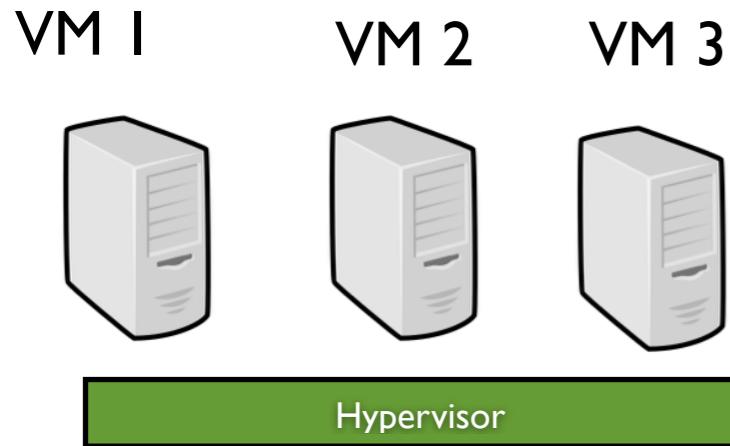


- Suspend/Resume
- Live migration  
(negligible downtime ~ 60 ms)  
Post/Pre Copy



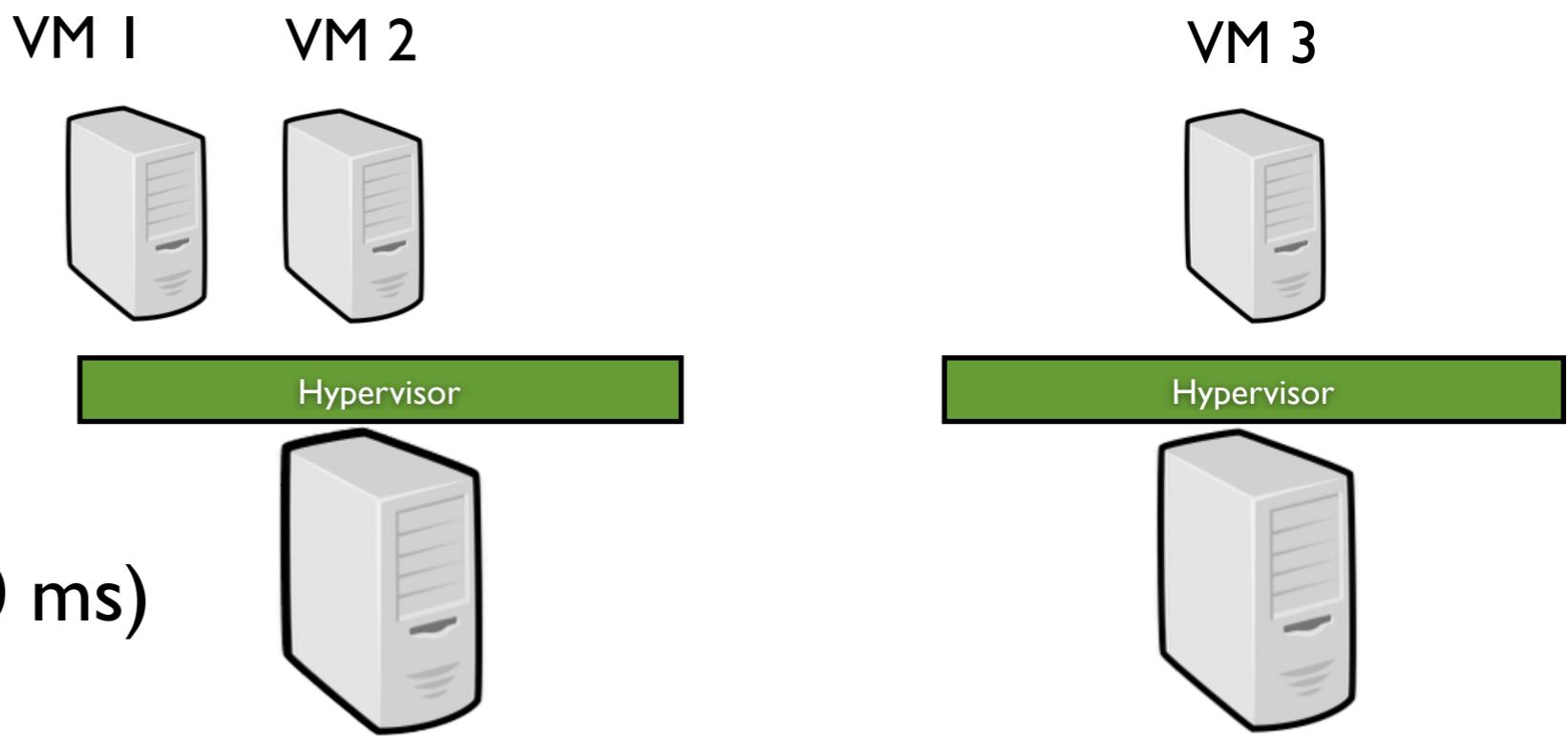
# Looking back...

- System virtualization: a great sandbox



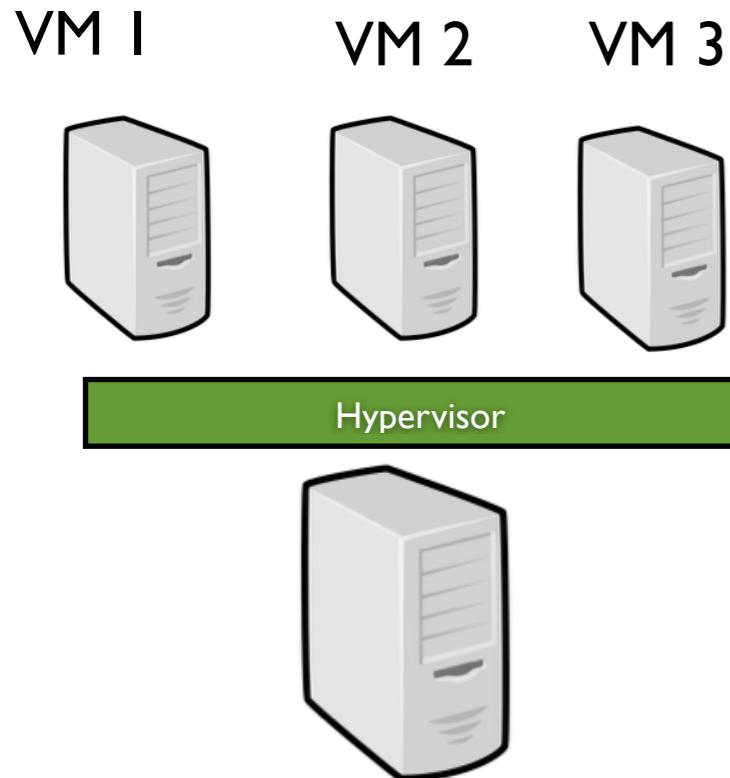
- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

- Suspend/Resume
- Live migration  
(negligible downtime ~ 60 ms)  
Post/Pre Copy



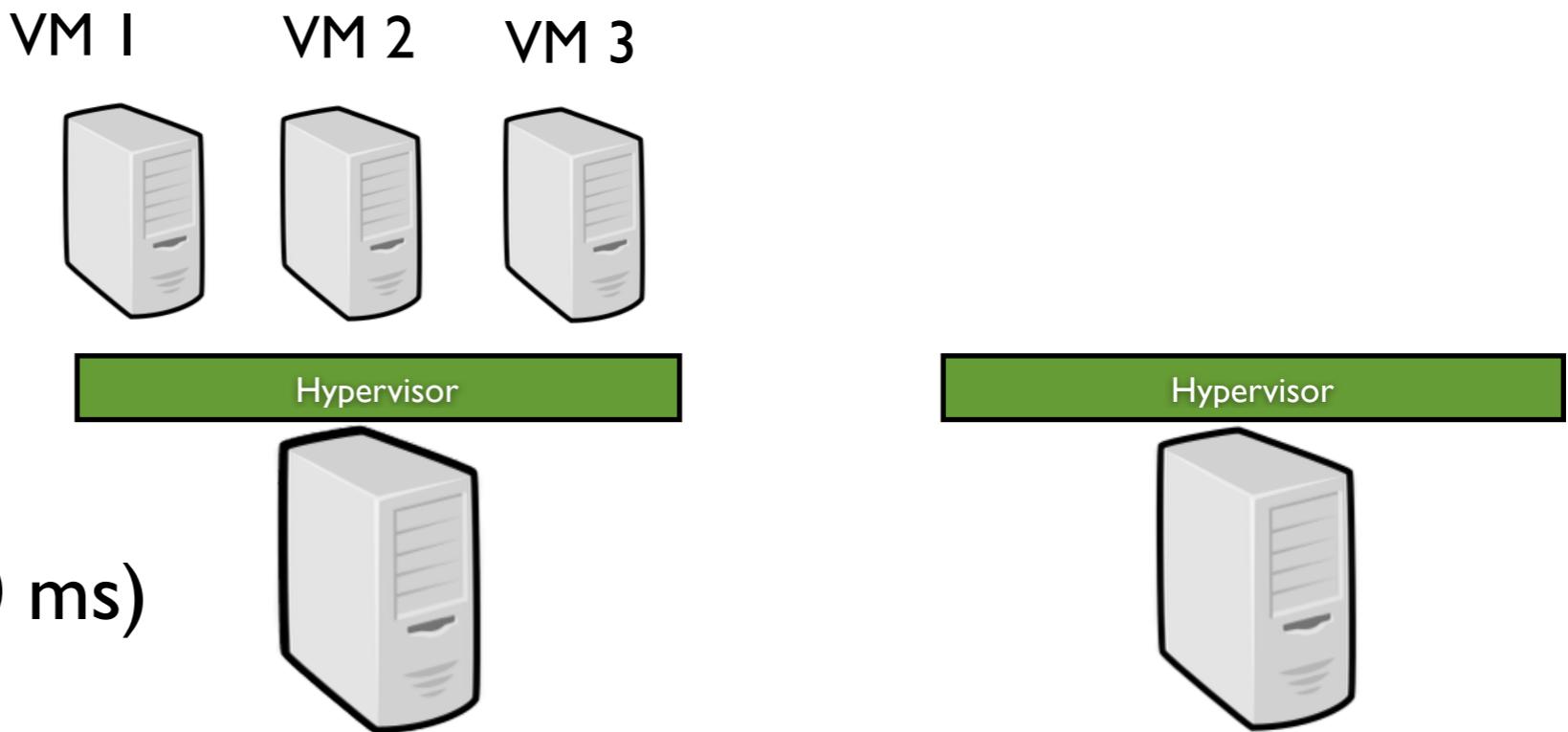
# Looking back...

- System virtualization: a great sandbox



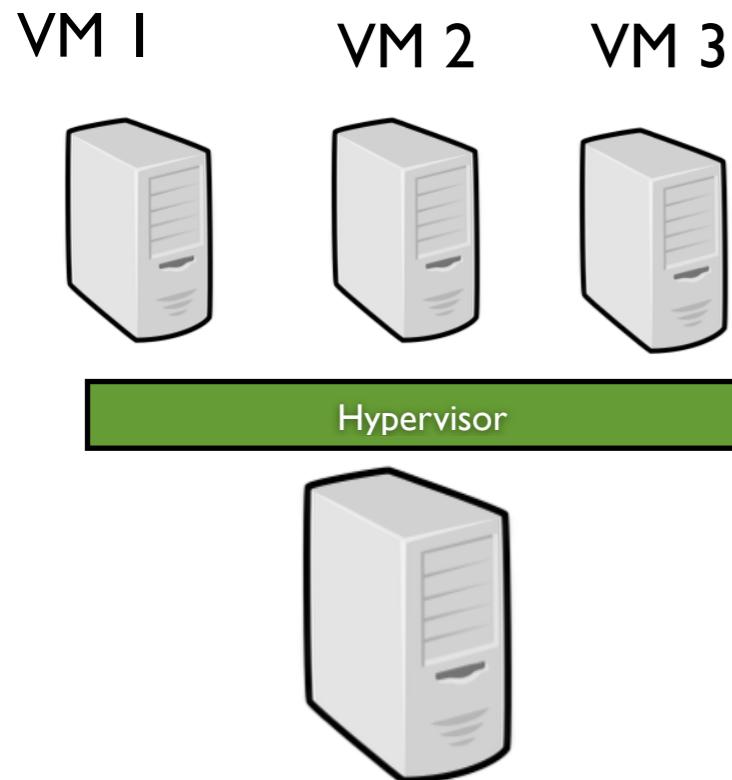
- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)

- Suspend/Resume
- Live migration  
(negligible downtime ~ 60 ms)  
Post/Pre Copy

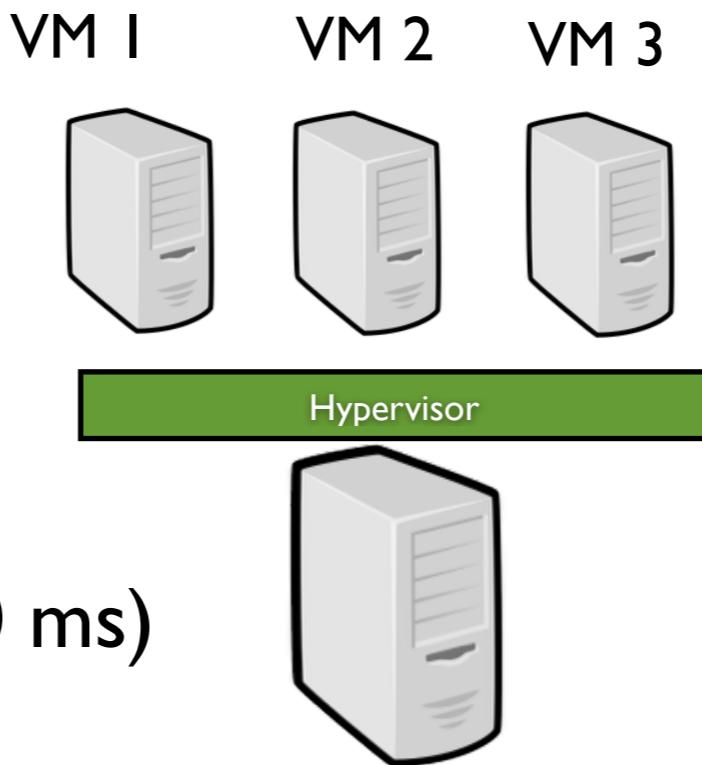


# Looking back...

- System virtualization: a great sandbox



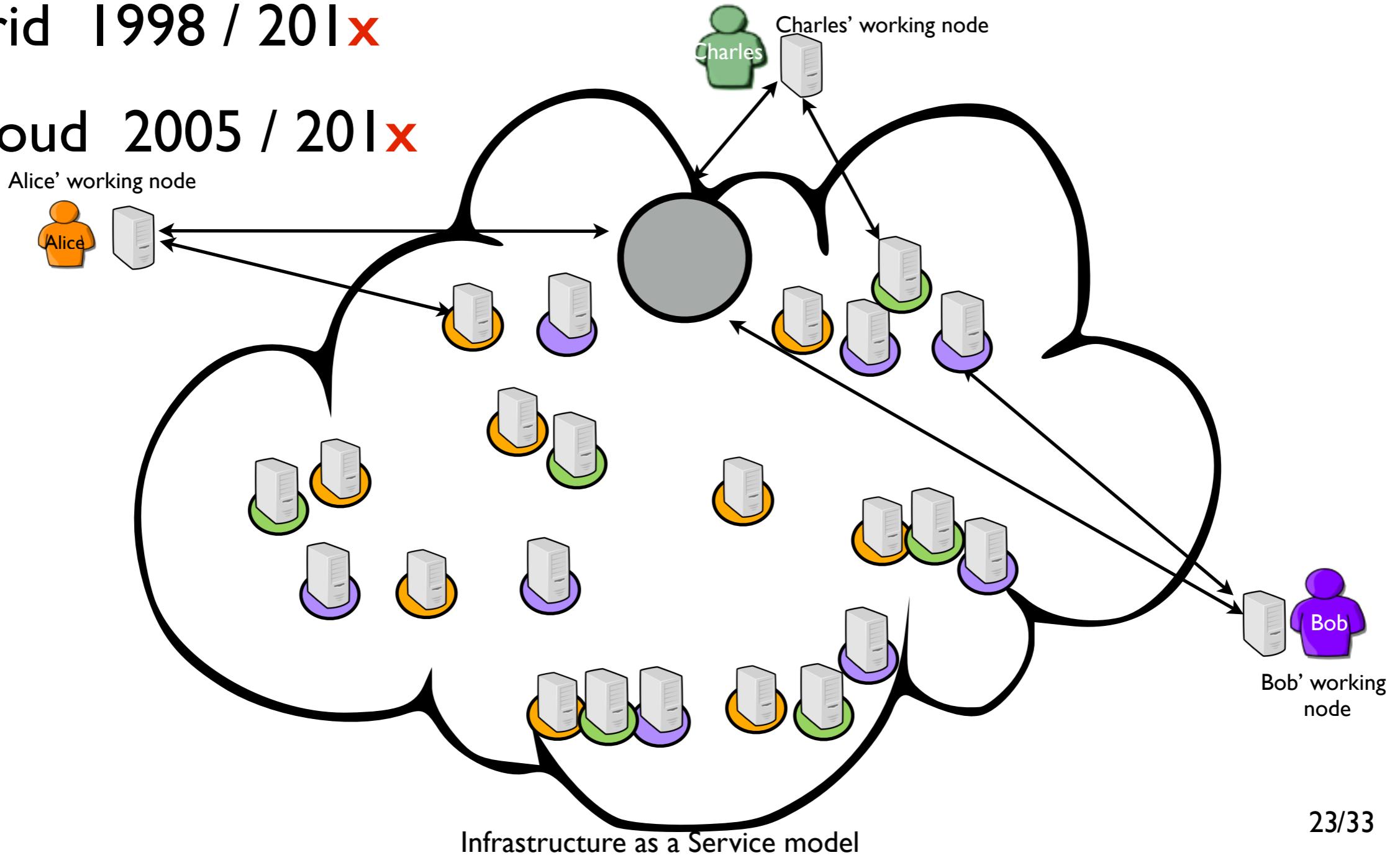
- Isolation (“security” between each VM)
- Snapshotting (a VM can be easily resume from its latest consistent state)



- Suspend/Resume
- Live migration  
(negligible downtime ~ 60 ms)  
Post/Pre Copy

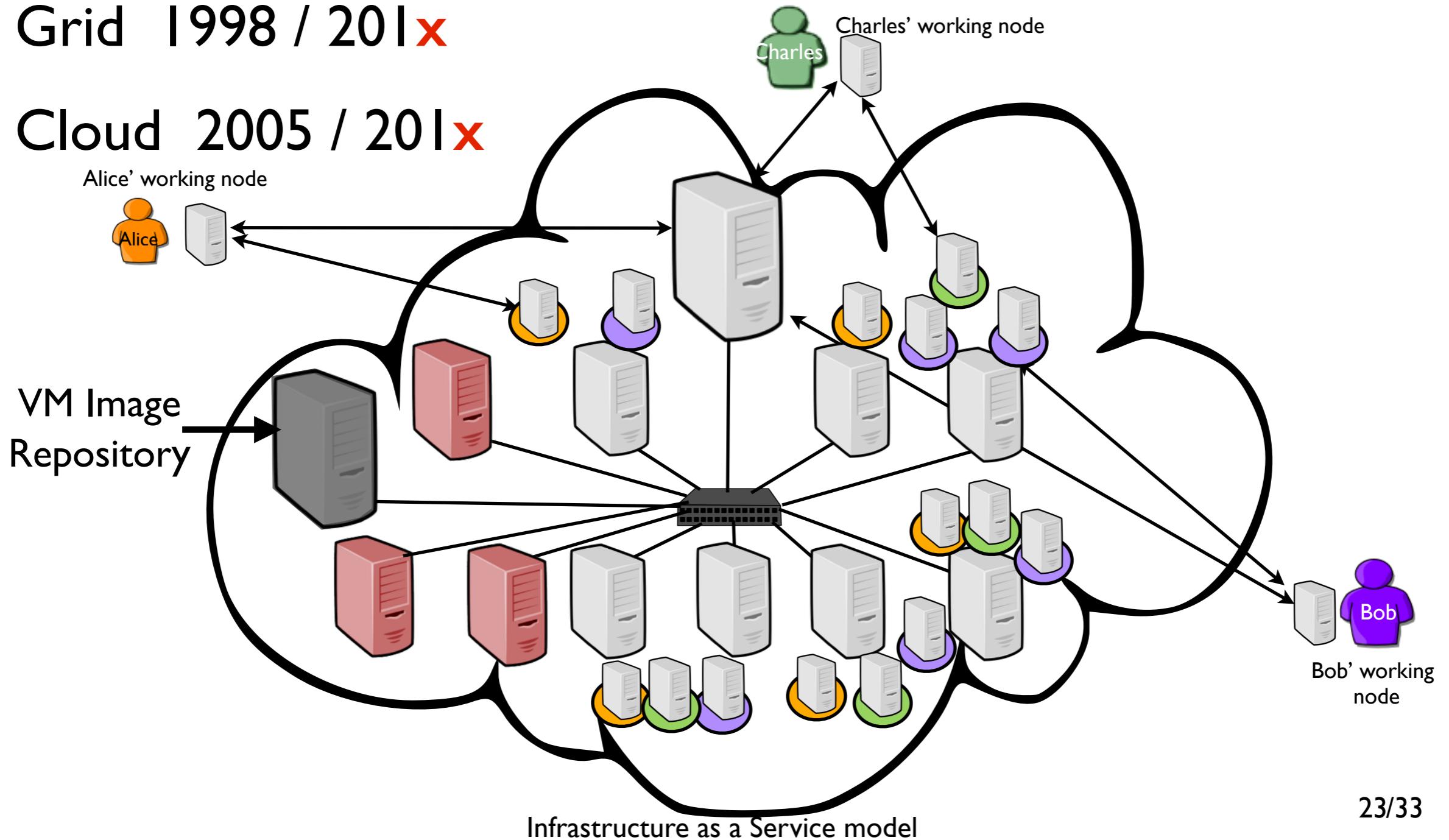
# Looking back ...

- Network of Workstations 1990 / 20~~xx~~
- Desktop Computing 1998/201~~x~~
- Grid 1998 / 201~~x~~
- Cloud 2005 / 201~~x~~



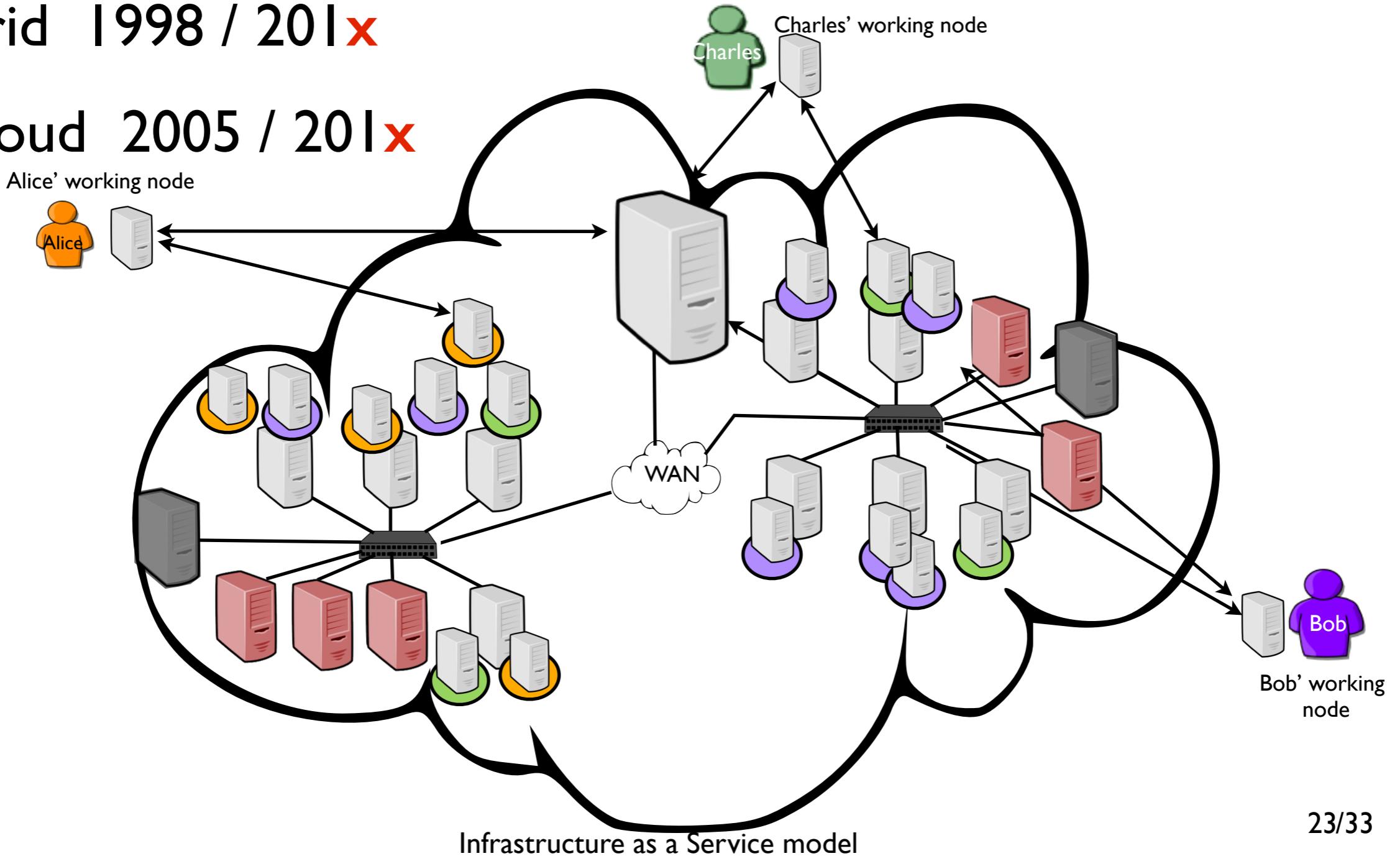
# Looking back ...

- Network of Workstations 1990 / 20~~xx~~
- Desktop Computing 1998/201~~x~~
- Grid 1998 / 201~~x~~
- Cloud 2005 / 201~~x~~



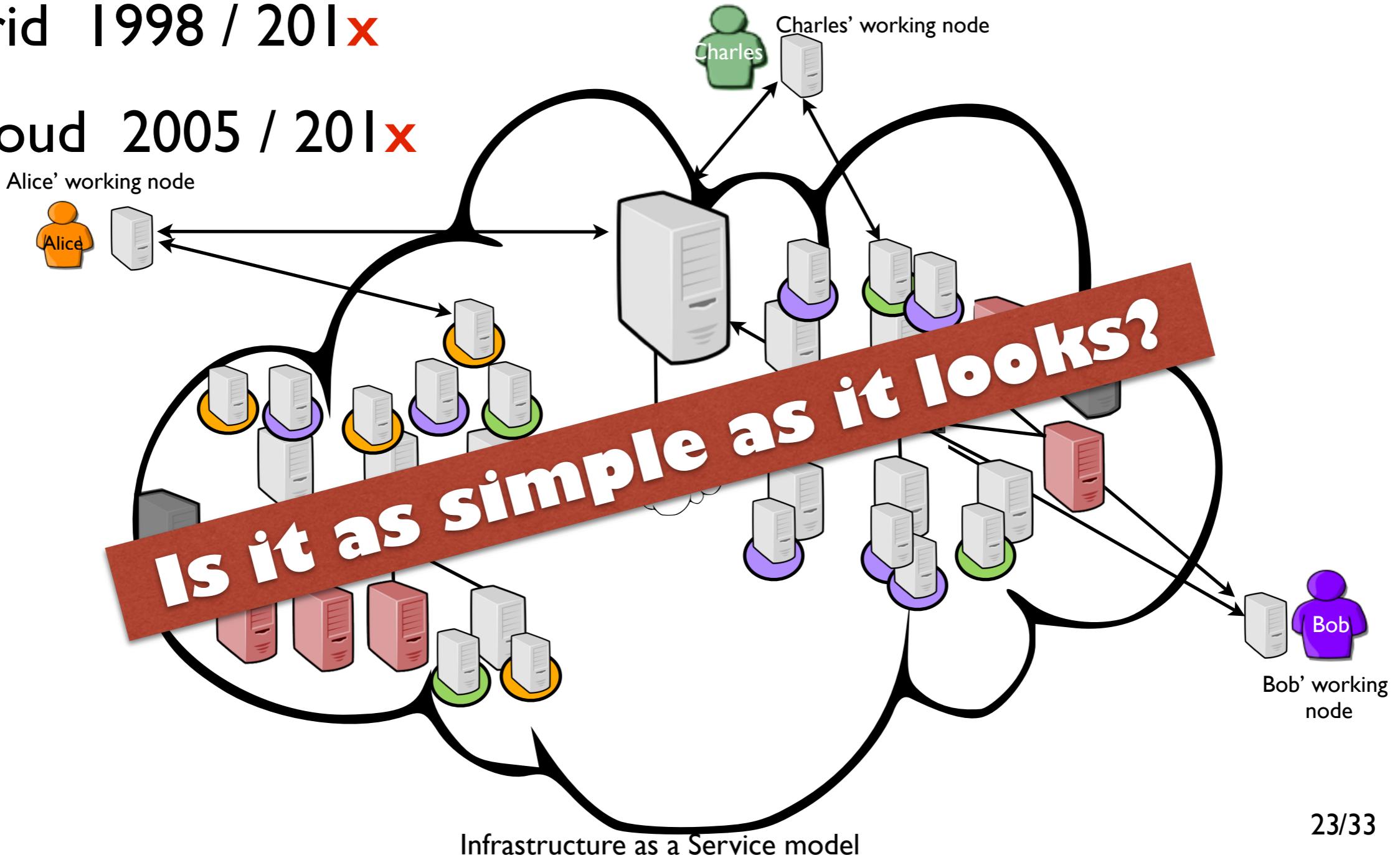
# Looking back ...

- Network of Workstations 1990 / 20~~xx~~
- Desktop Computing 1998/201~~x~~
- Grid 1998 / 201~~x~~
- Cloud 2005 / 201~~x~~



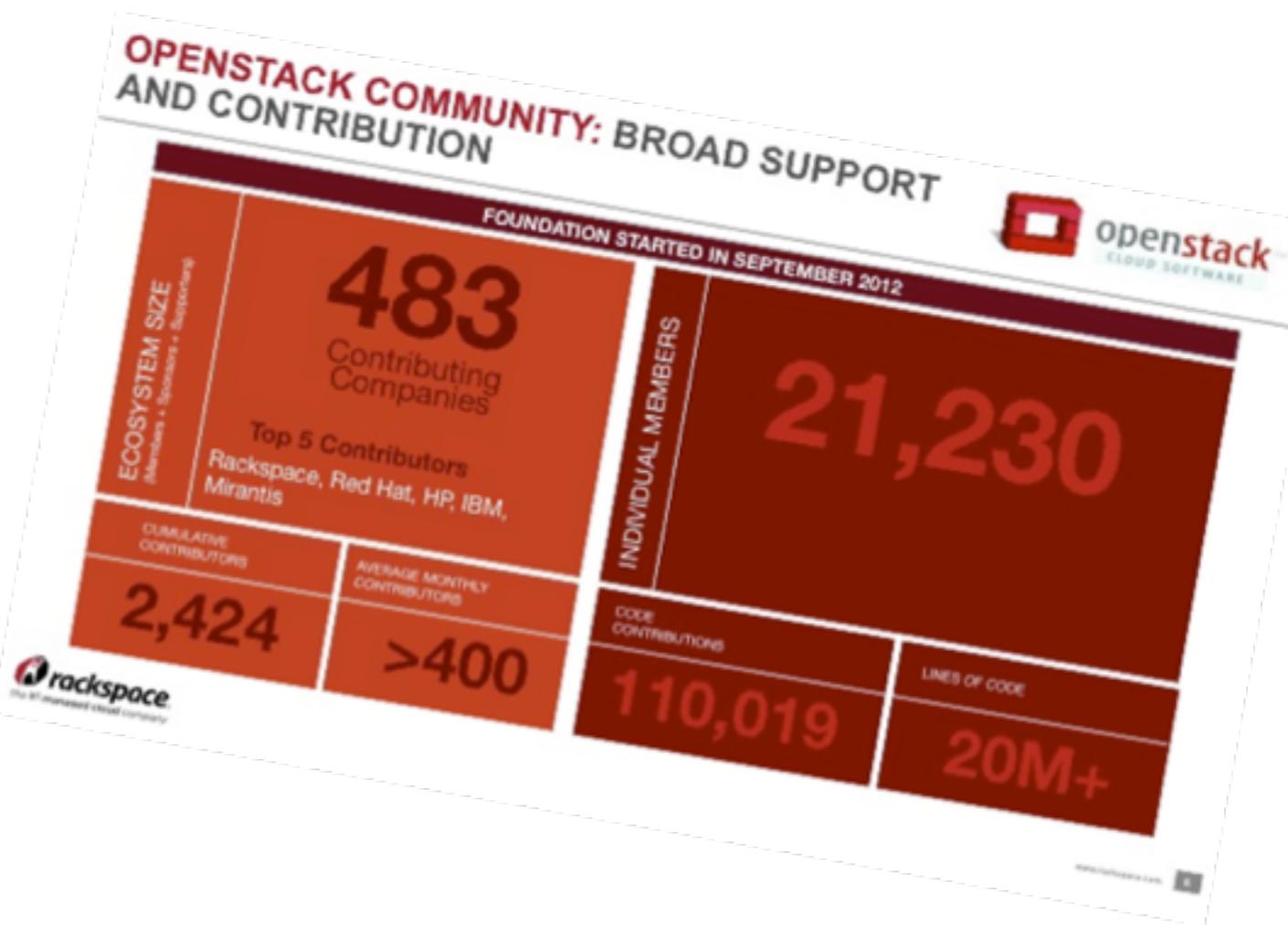
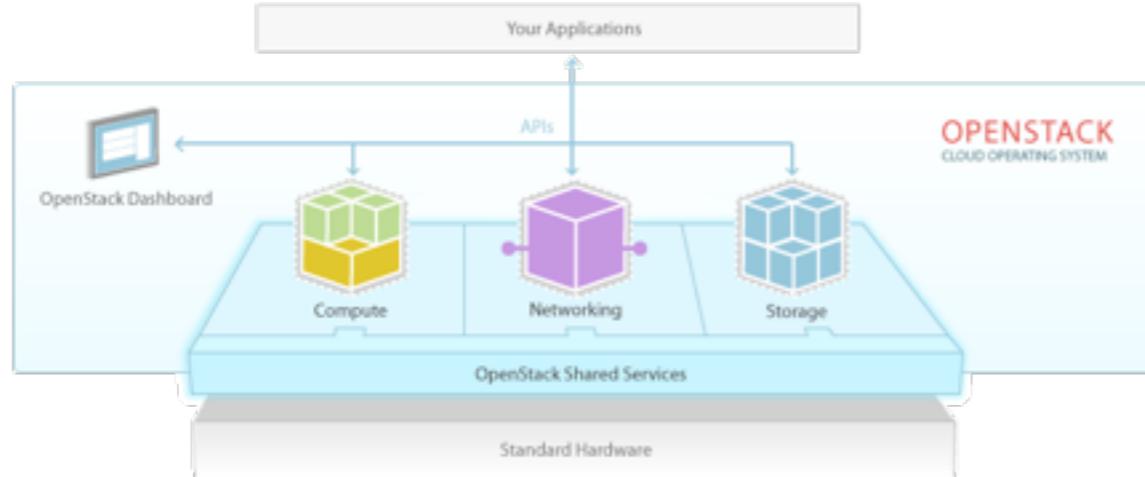
# Looking back ...

- Network of Workstations 1990 / 20~~xx~~
- Desktop Computing 1998/201~~x~~
- Grid 1998 / 201~~x~~
- Cloud 2005 / 201~~x~~



# OpenStack

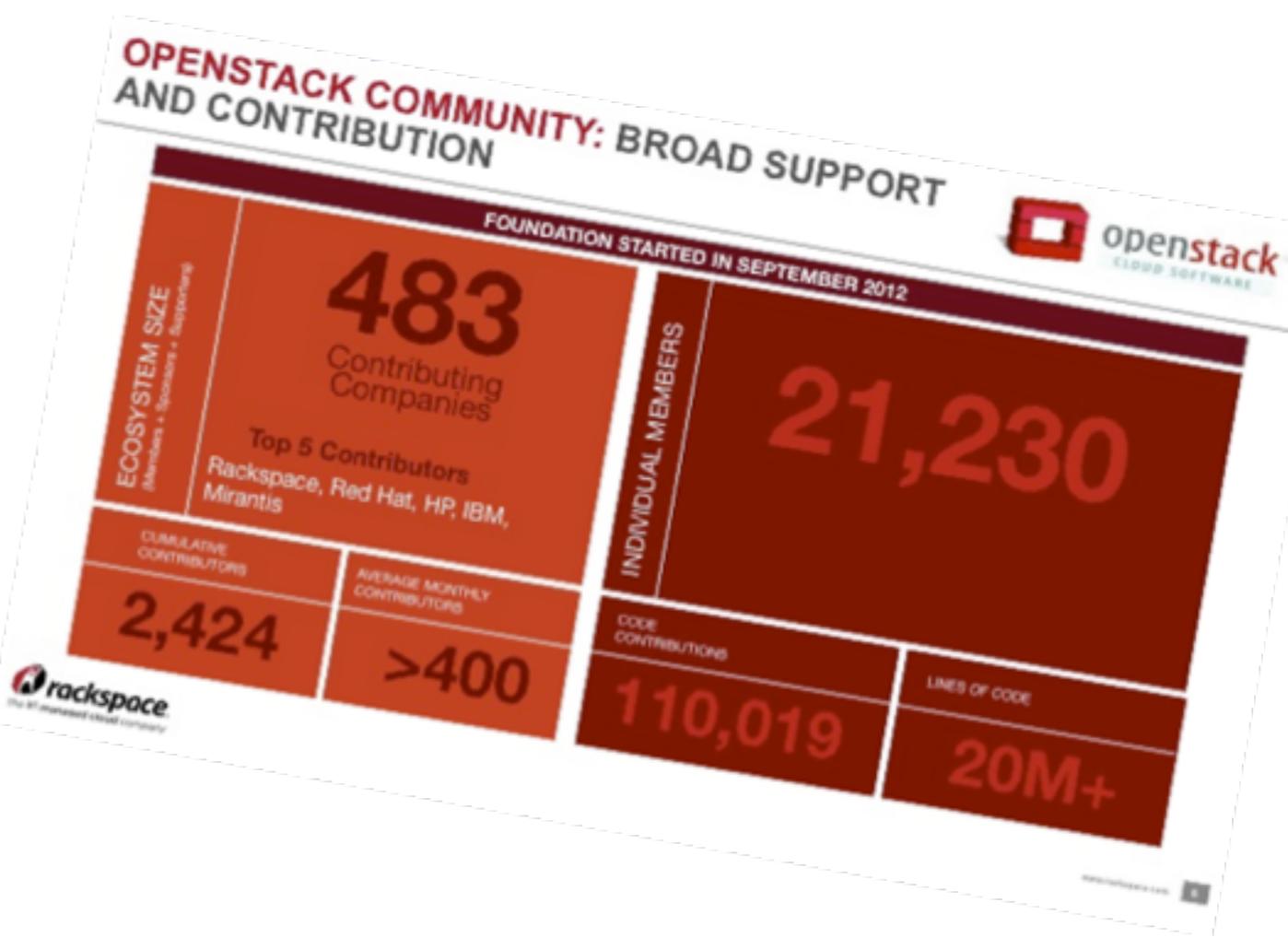
- Industry standard for creating public and private clouds



- A rich (and complex) ecosystem
- **20 Millions of LoC**, 164 services, some services are composed of sub-services (e.g. nova-scheduler, nova-conductor, ...)

# OpenStack

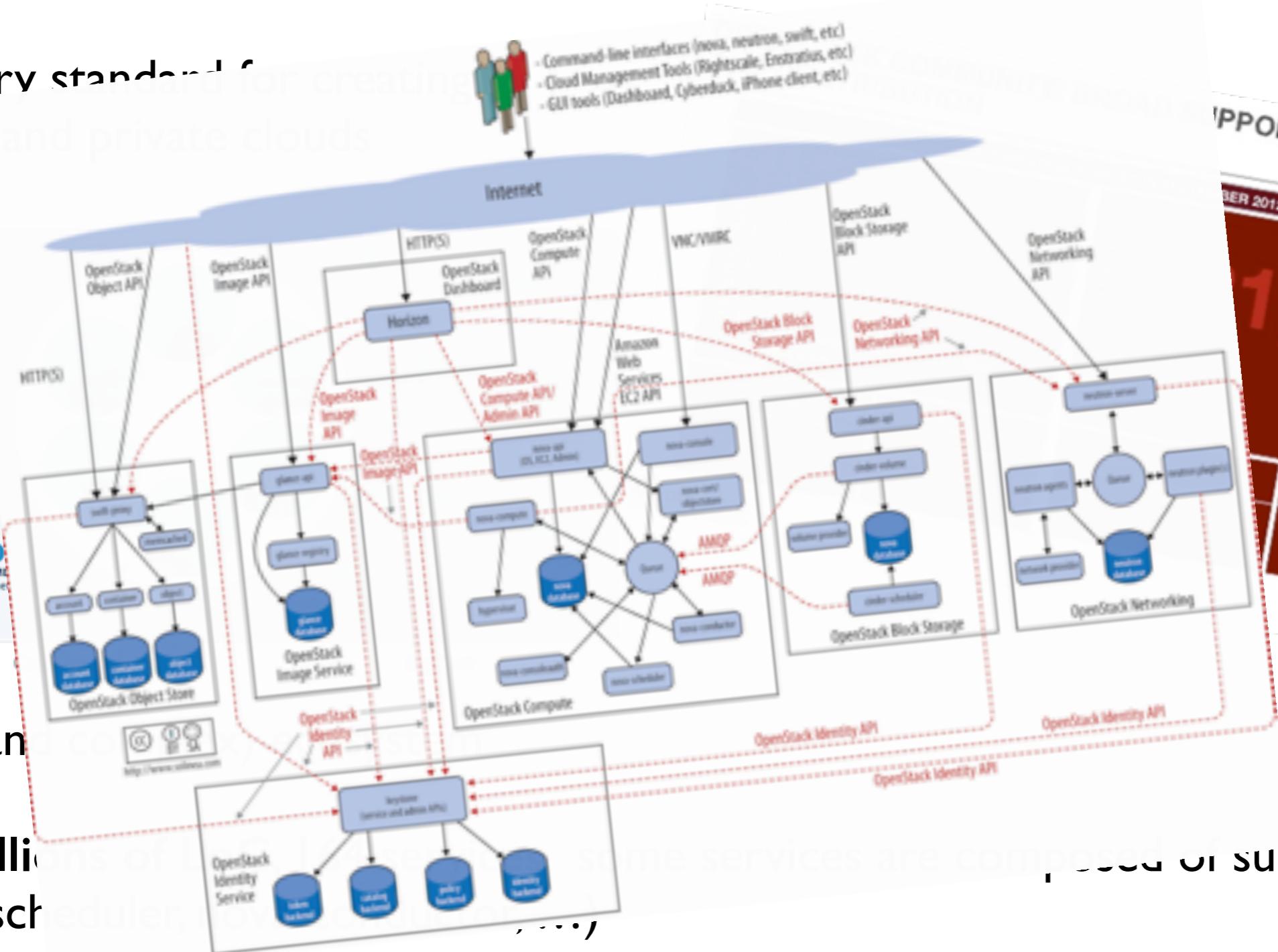
- Industry standard for creating public and private clouds



- A rich (and complex) ecosystem
- 20 Millions of LoC, 164 services, some services are composed of sub-services (e.g. nova-scheduler, nova-conductor, ...)

# OpenStack

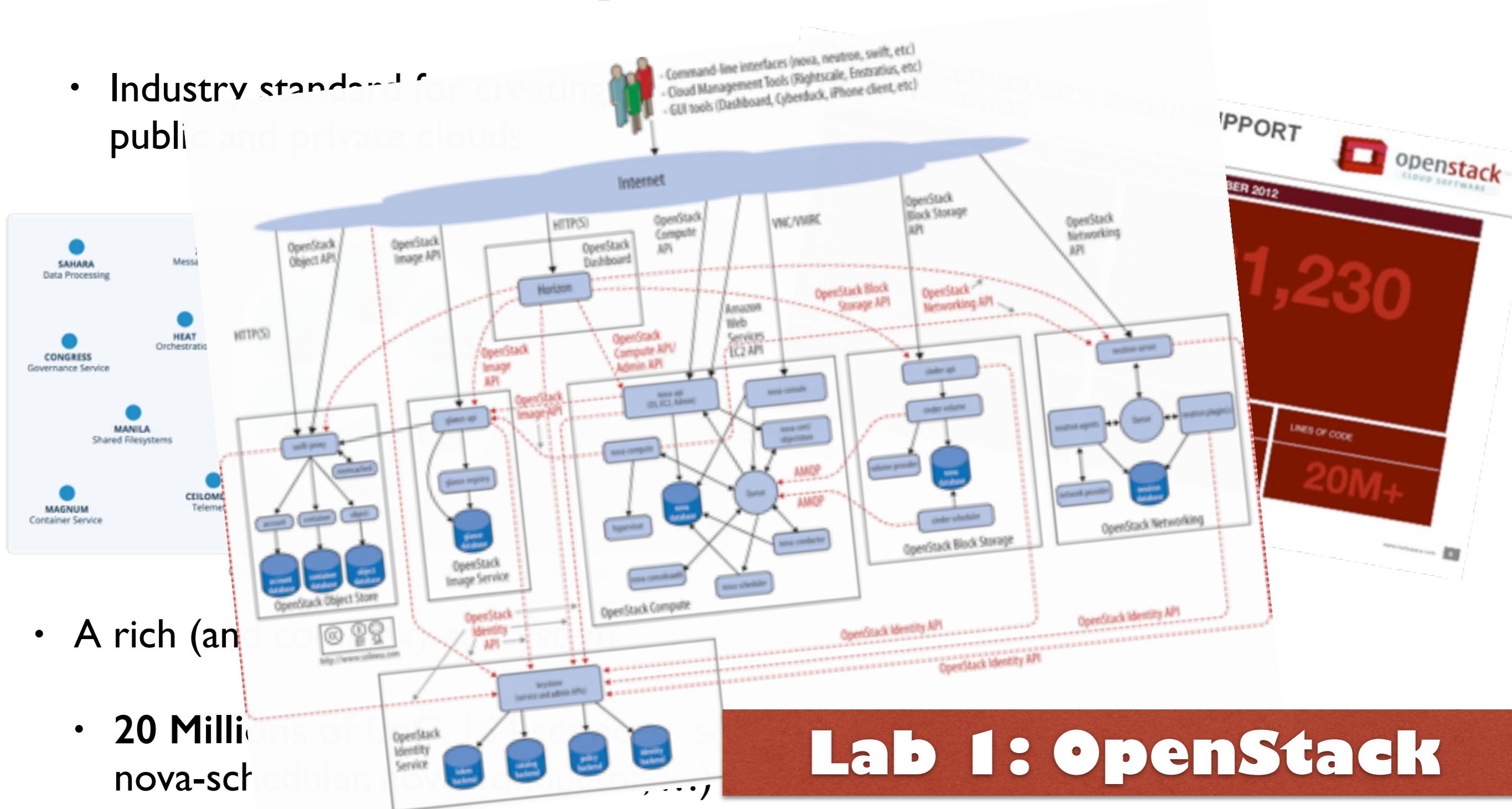
- Industry standards for public and private clouds



- A rich (and growing) set of services
- 20 Million lines of code
- The services are composed of sub-services (e.g. nova-scheduler)

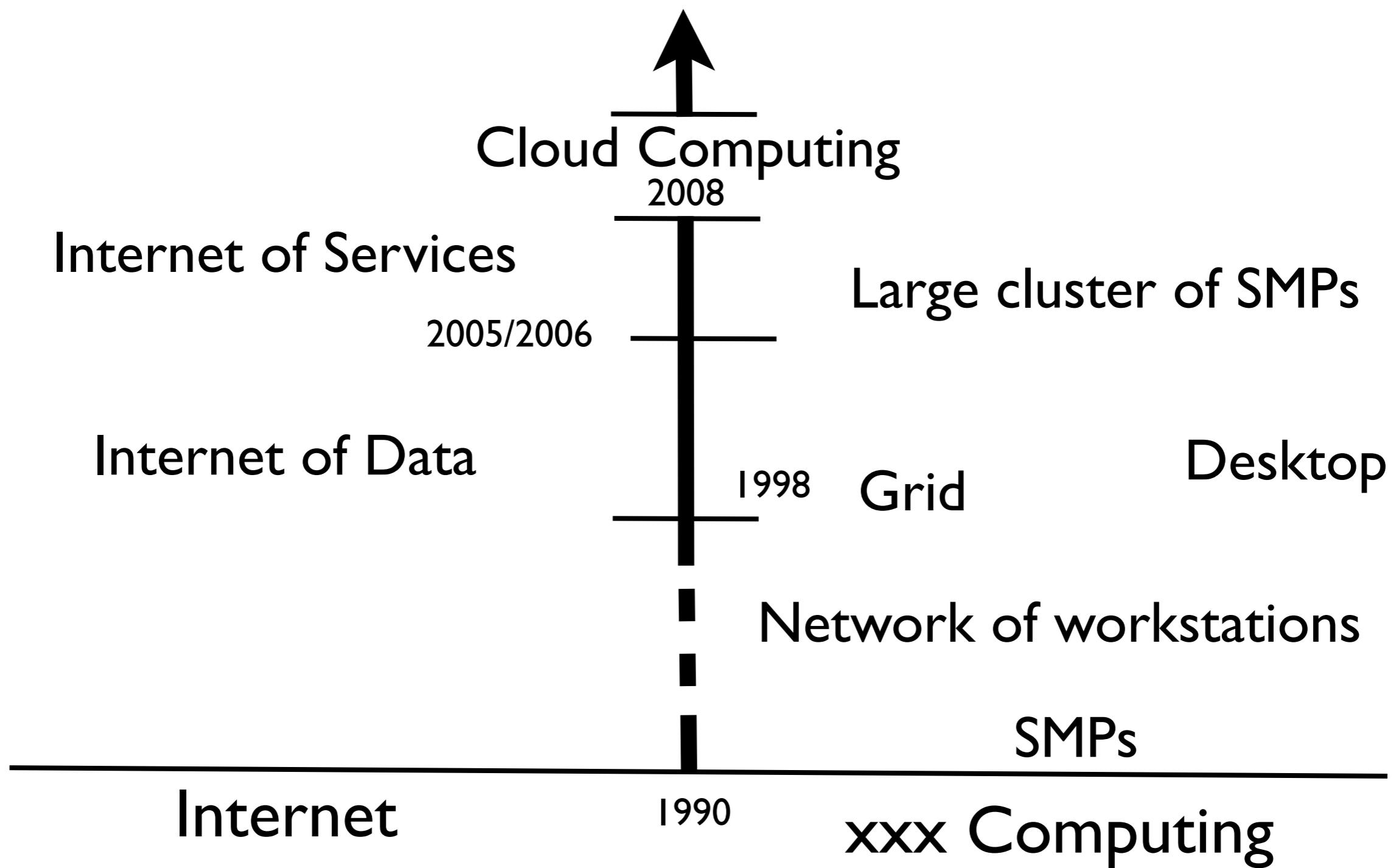
# OpenStack

- Industry standards for public



*The Cloud...just an infrastructure?*

# Internet + Distributed Computing ?



# Cloud Computing

- A “merge” between Internet and Distributed Computing

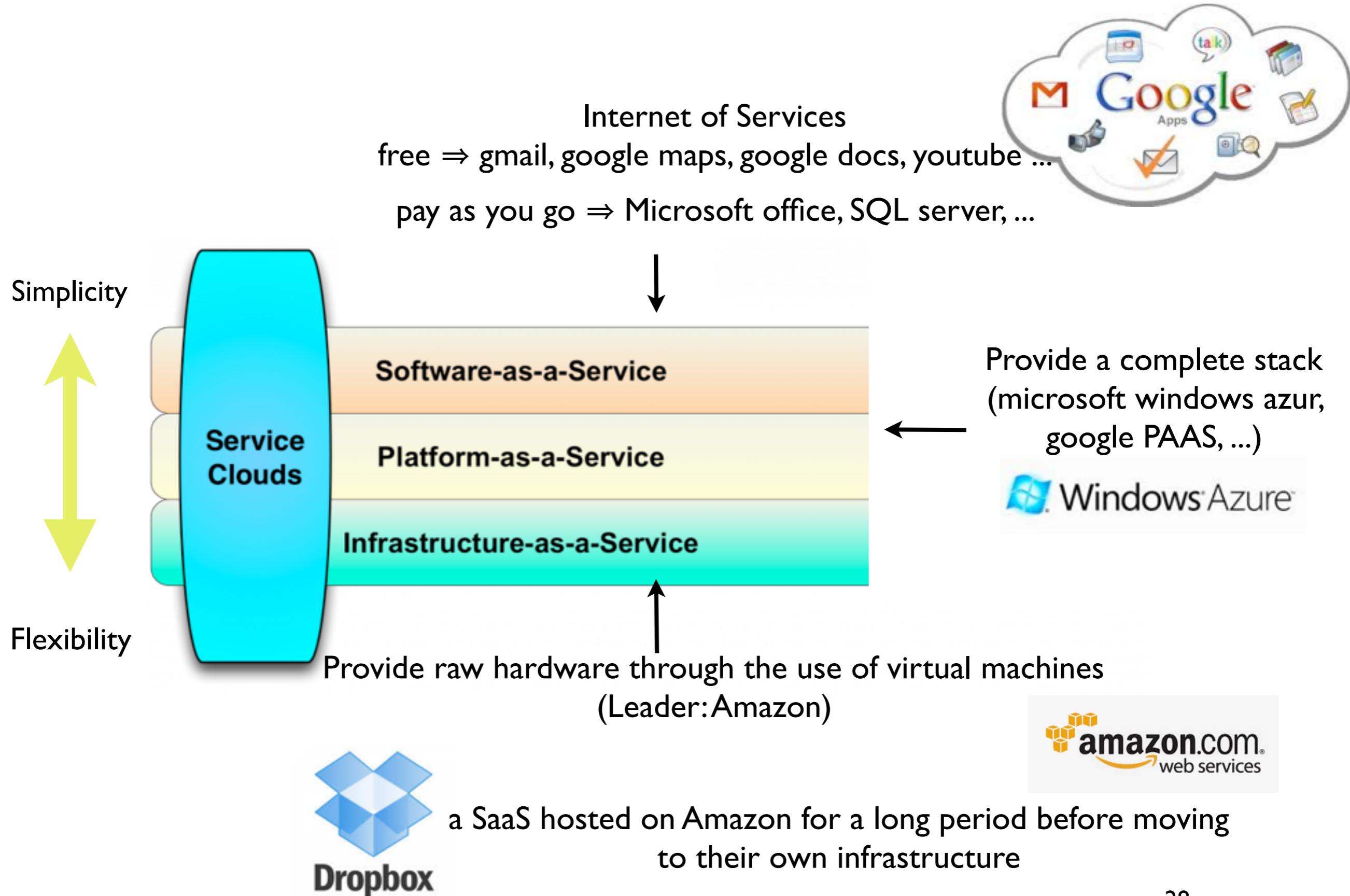
From Internet point of view:

Not only data/services but raw resources

From distributed computing point of view:

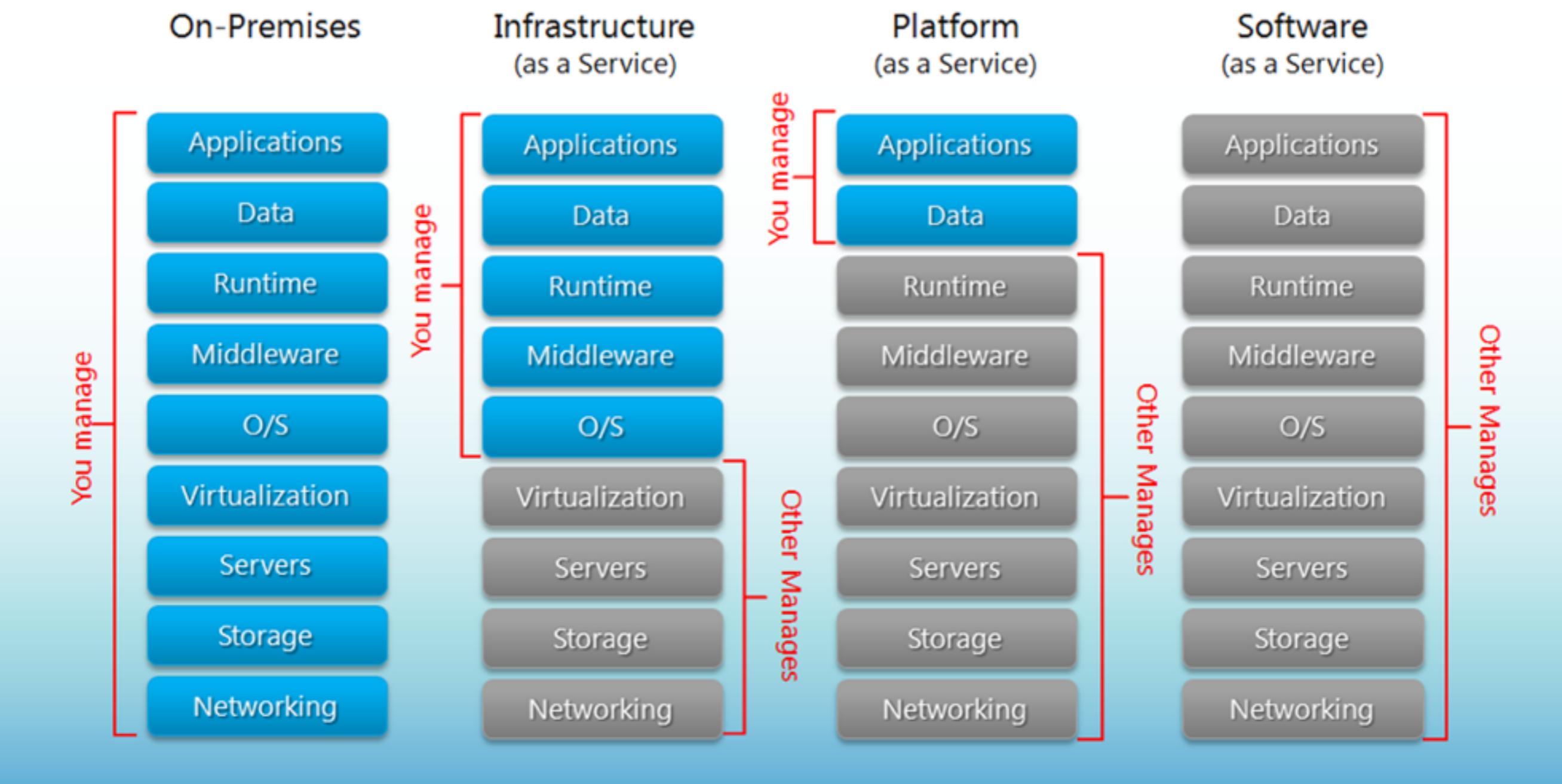
a common objective - provide computing resources (both hardware and software) in a flexible, transparent, secure, reliable, ... way

# SPI Classification



# Who is in charge of?

## Separation of Responsibilities



# The Cloud needs scalable infrastructure

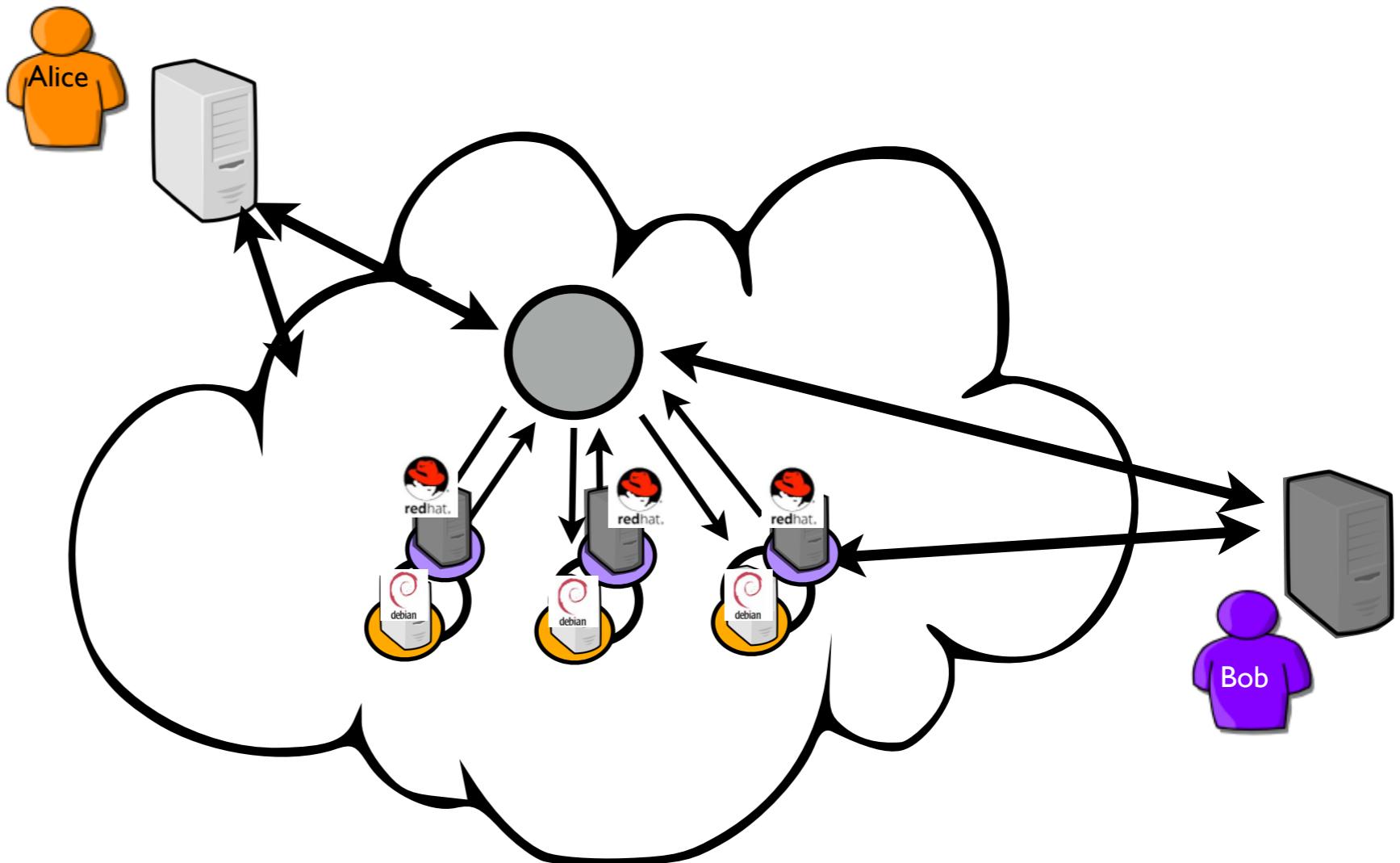
- Scalability: capacity to increase throughput as the size of the infrastructure increases.
- A scalable infrastructure requires scalable software and hardware architectures:
  - More resources must imply better performance
  - No single Point of Failure (PoF)
  - Efficient resource usage
  - Ability to manage heterogeneous resources

# The Cloud needs scalable infrastructure

- 2 strategies to scale up an infrastructure:
  - Vertical scaling: increase the capacity of individual resources (scale up).
  - Horizontal scaling: increase the number of resources (scale out)
- The Cloud: make scale in/out cheap and easier
  - Virtually infinite resources
  - Available and charged on demand
  - no contract

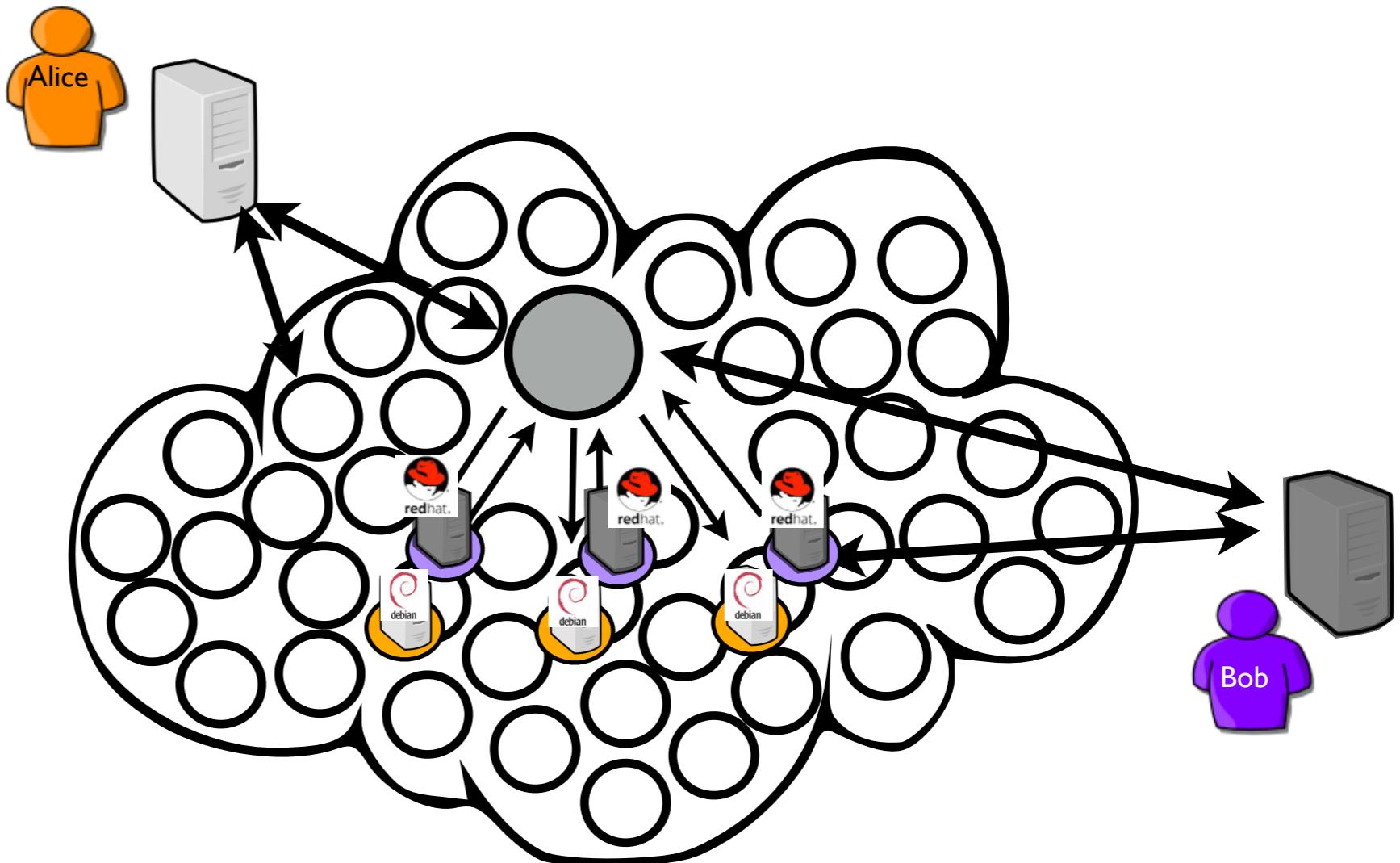
# Where we are?

- IaaS challenges



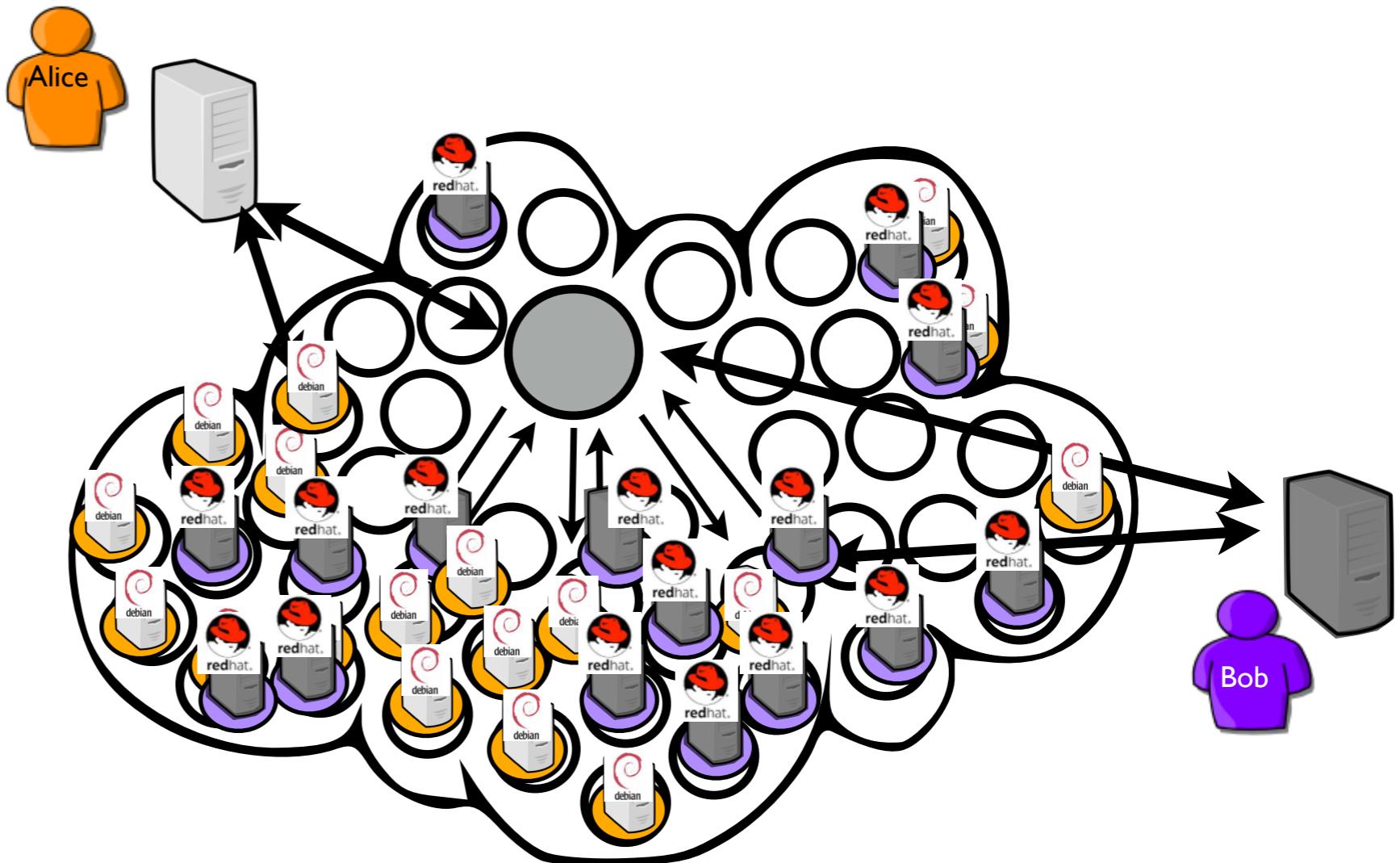
# Where we are?

- IaaS challenges



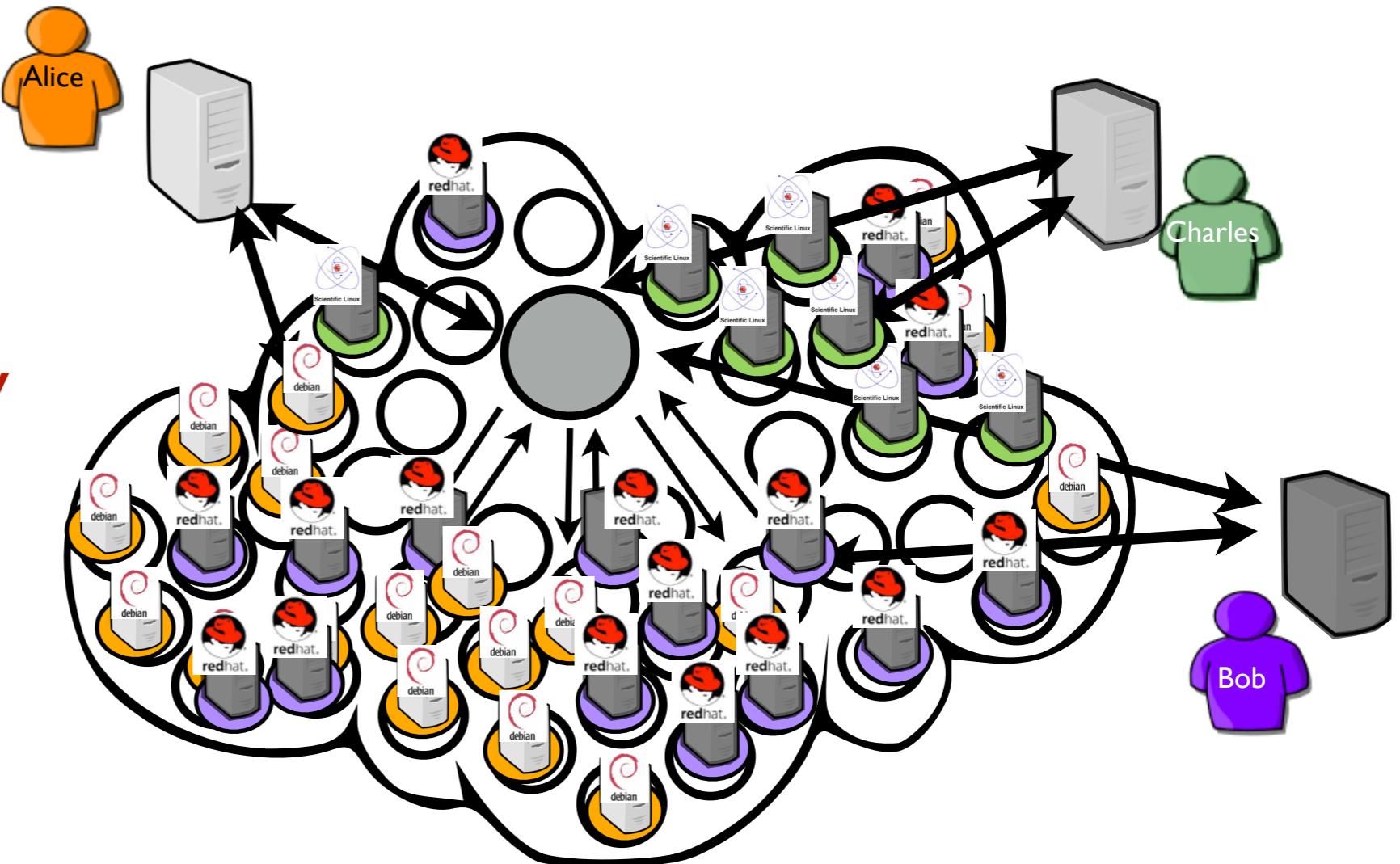
# Where we are?

- IaaS challenges



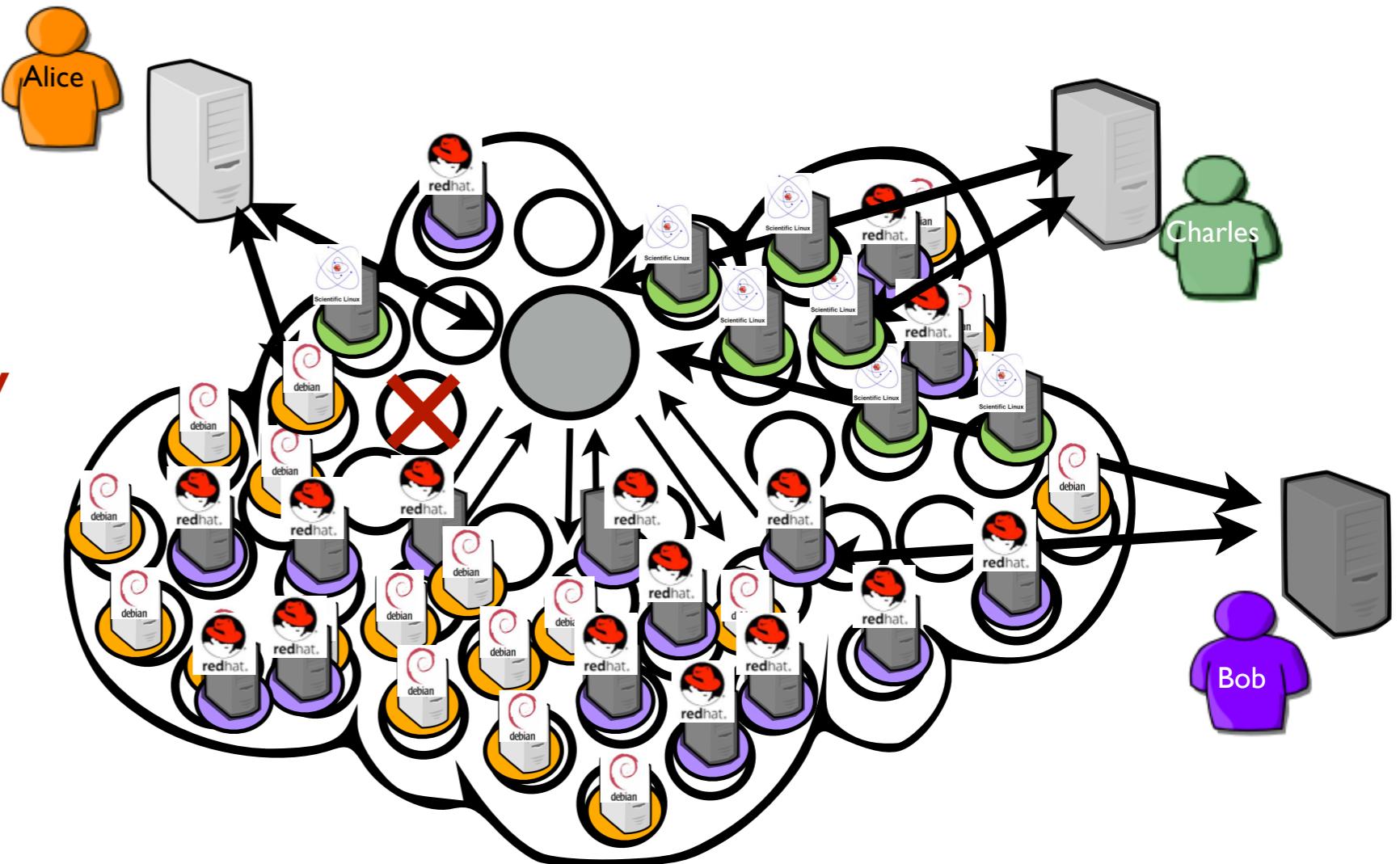
# Where we are?

- IaaS challenges  
**Scalability / Energy**



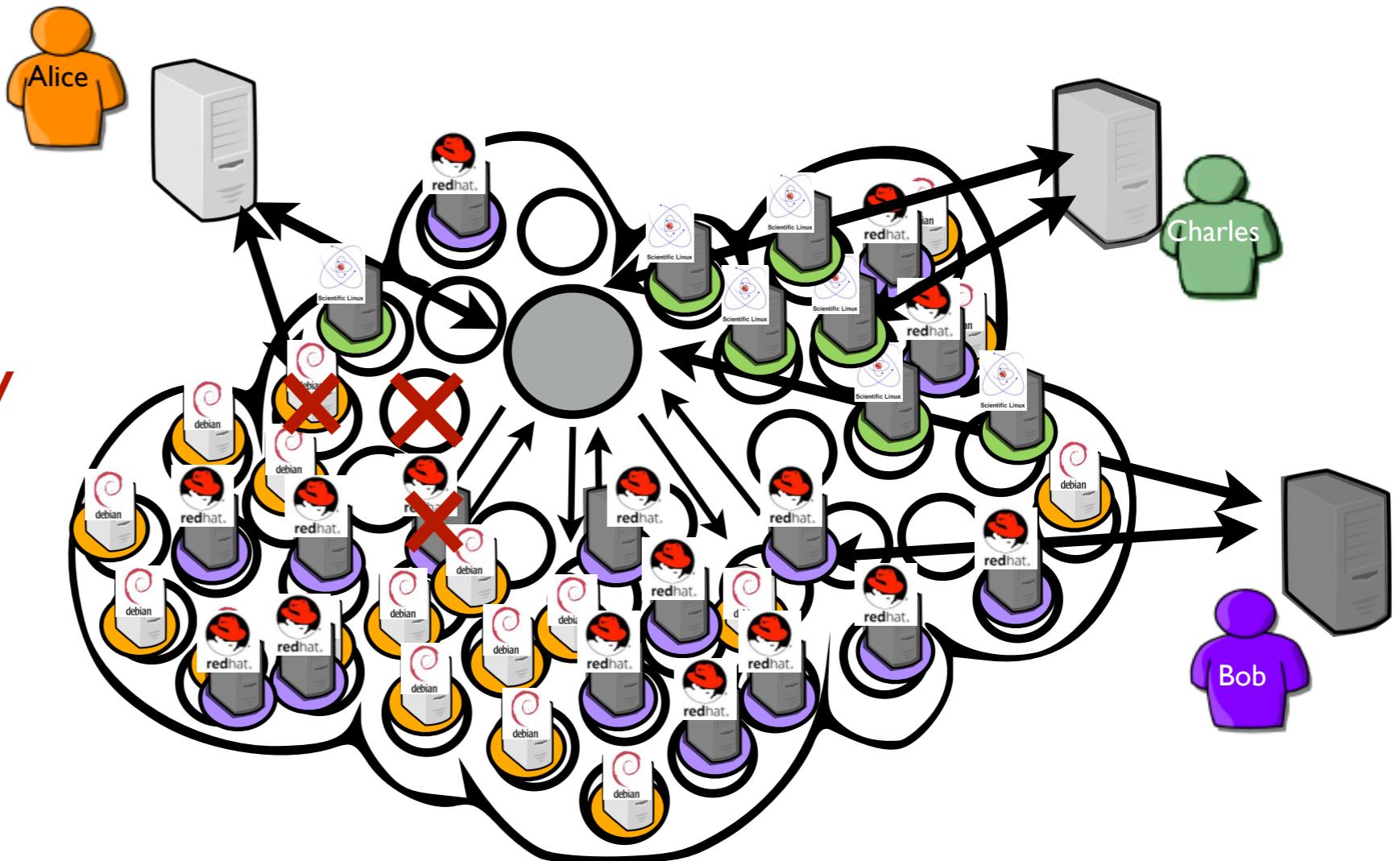
# Where we are?

- IaaS challenges  
**Scalability / Energy**



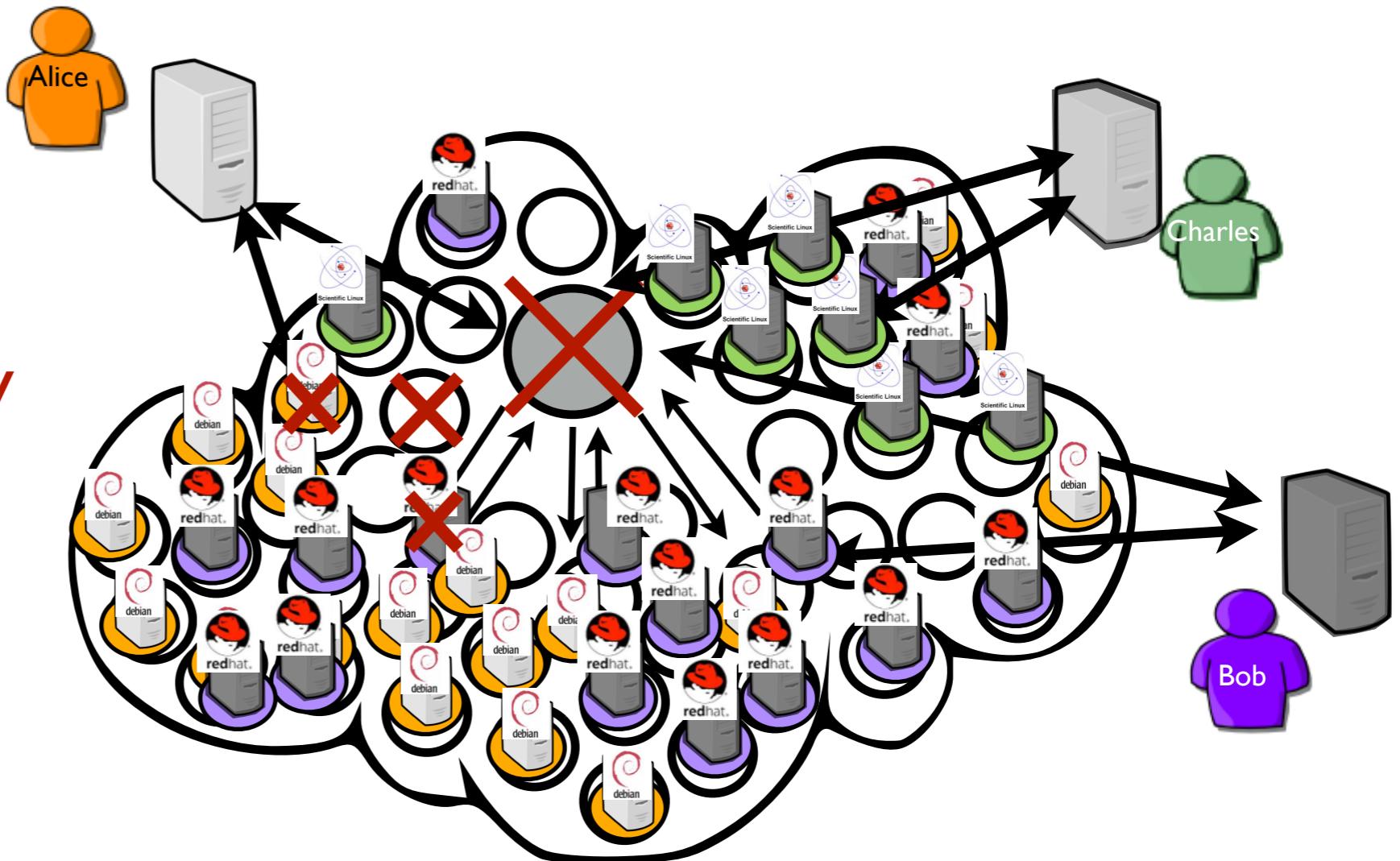
# Where we are?

- IaaS challenges  
**Scalability / Energy**



# Where we are?

- IaaS challenges  
**Scalability / Energy**  
**Reliability**

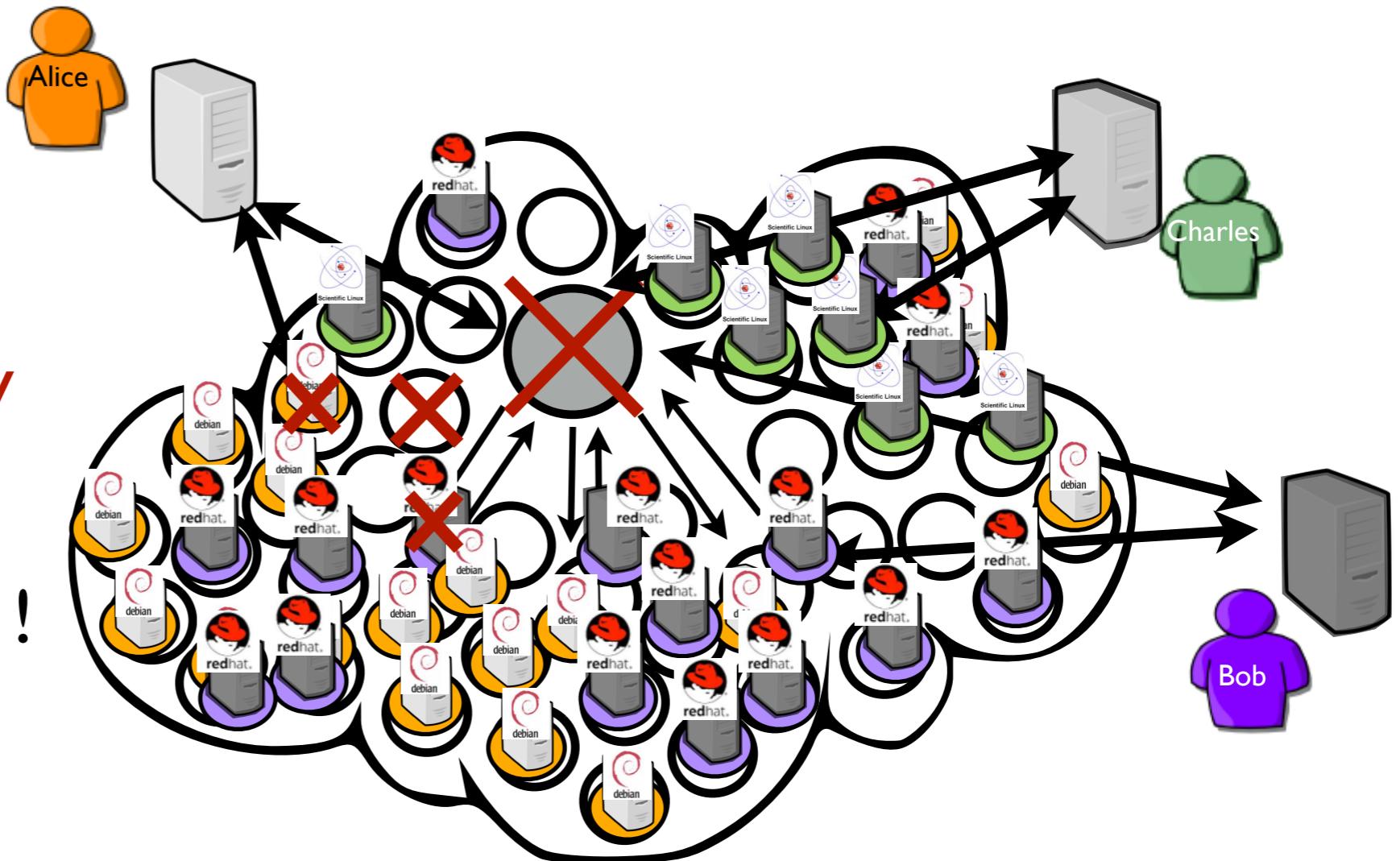


# Where we are?

- IaaS challenges

**Scalability / Energy  
Reliability**

nothing really new !

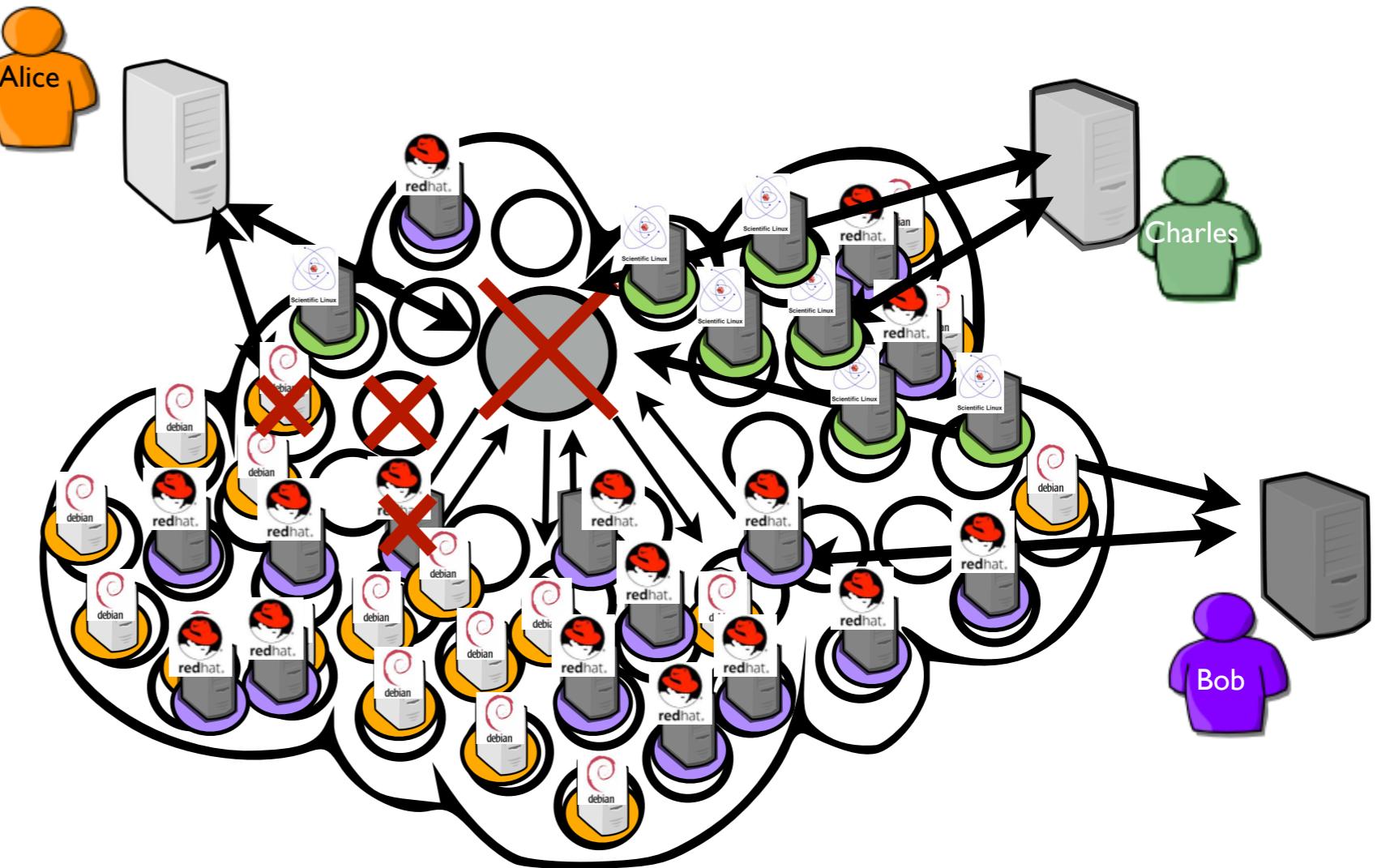


# Where we are?

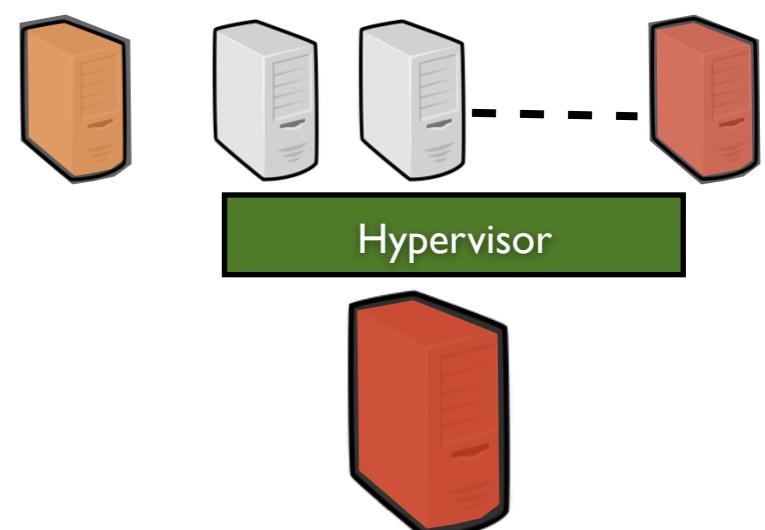
- IaaS challenges

**Scalability / Energy  
Reliability**

nothing really new !



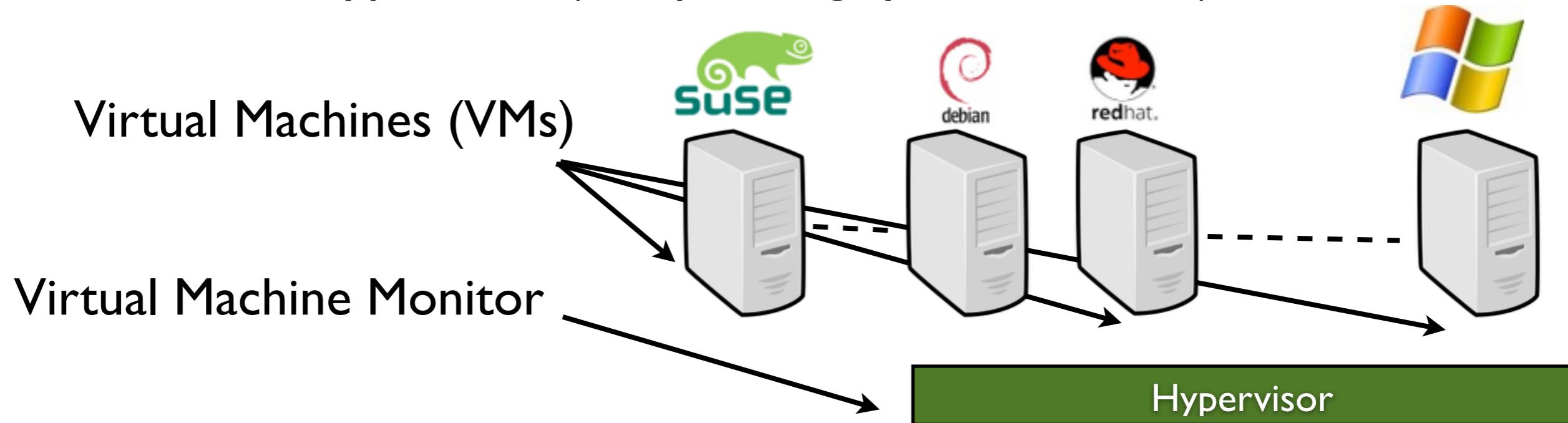
- Virtualize IT impacts performances !  
(difficulty to guarantee performances, SLAs)



# *Virtualisation and Performance*

# System Virtualisation

- System virtualization: One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)



“A **virtual machine** (VM) provides a faithful implementation of a physical processor’s hardware running in a protected and isolated environment.

Virtual machines are created by a software layer called the **virtual machine monitor** (VMM) that runs as a privileged task on a physical processor.”

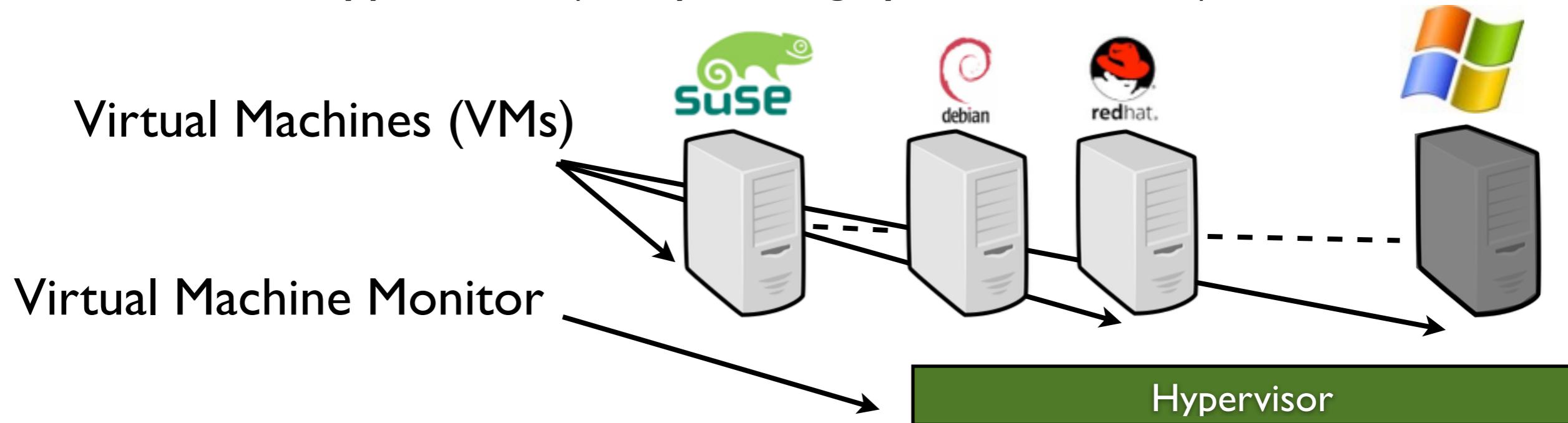


**Physical Machine (PM)**

Key player: XEN / KVM / VmWare ESX

# System Virtualisation

- System virtualization: One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)



“A **virtual machine** (VM) provides a faithful implementation of a physical processor’s hardware running in a protected and isolated environment.

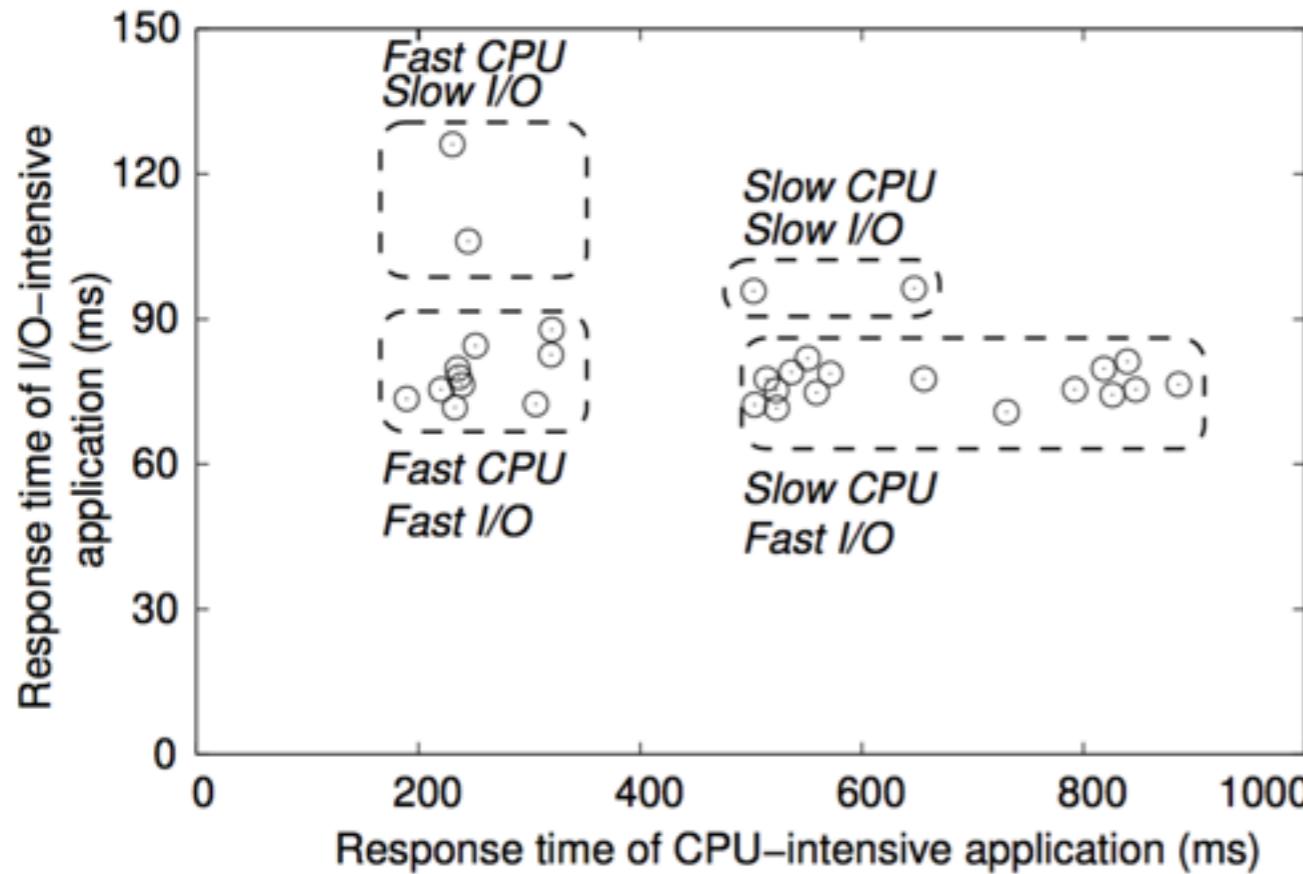
Virtual machines are created by a software layer called the **virtual machine monitor** (VMM) that runs as a privileged task on a physical processor.”



**Physical Machine (PM)**

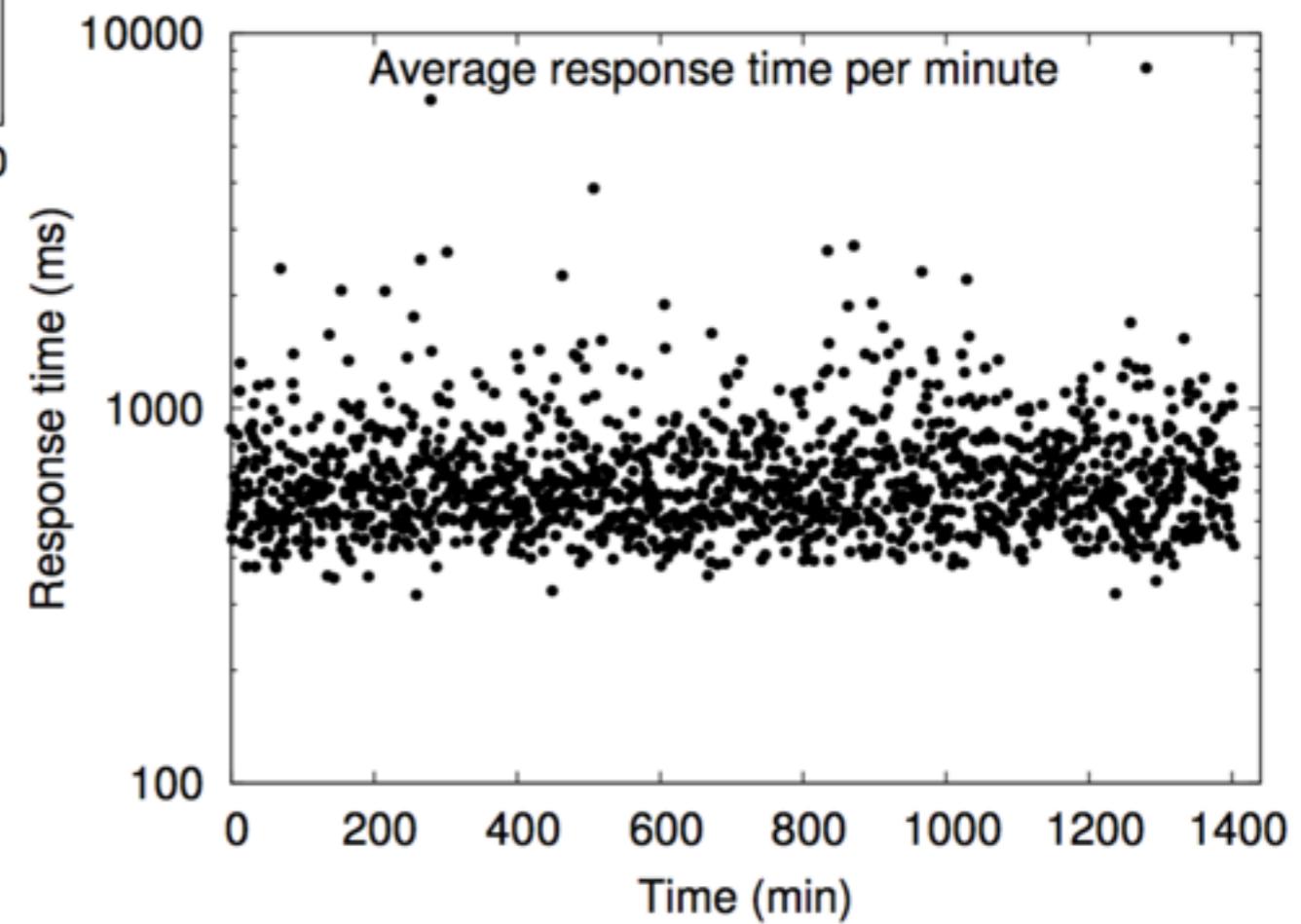
Key player: XEN / KVM / VmWare ESX

# Performance reproducibility [Dejli]



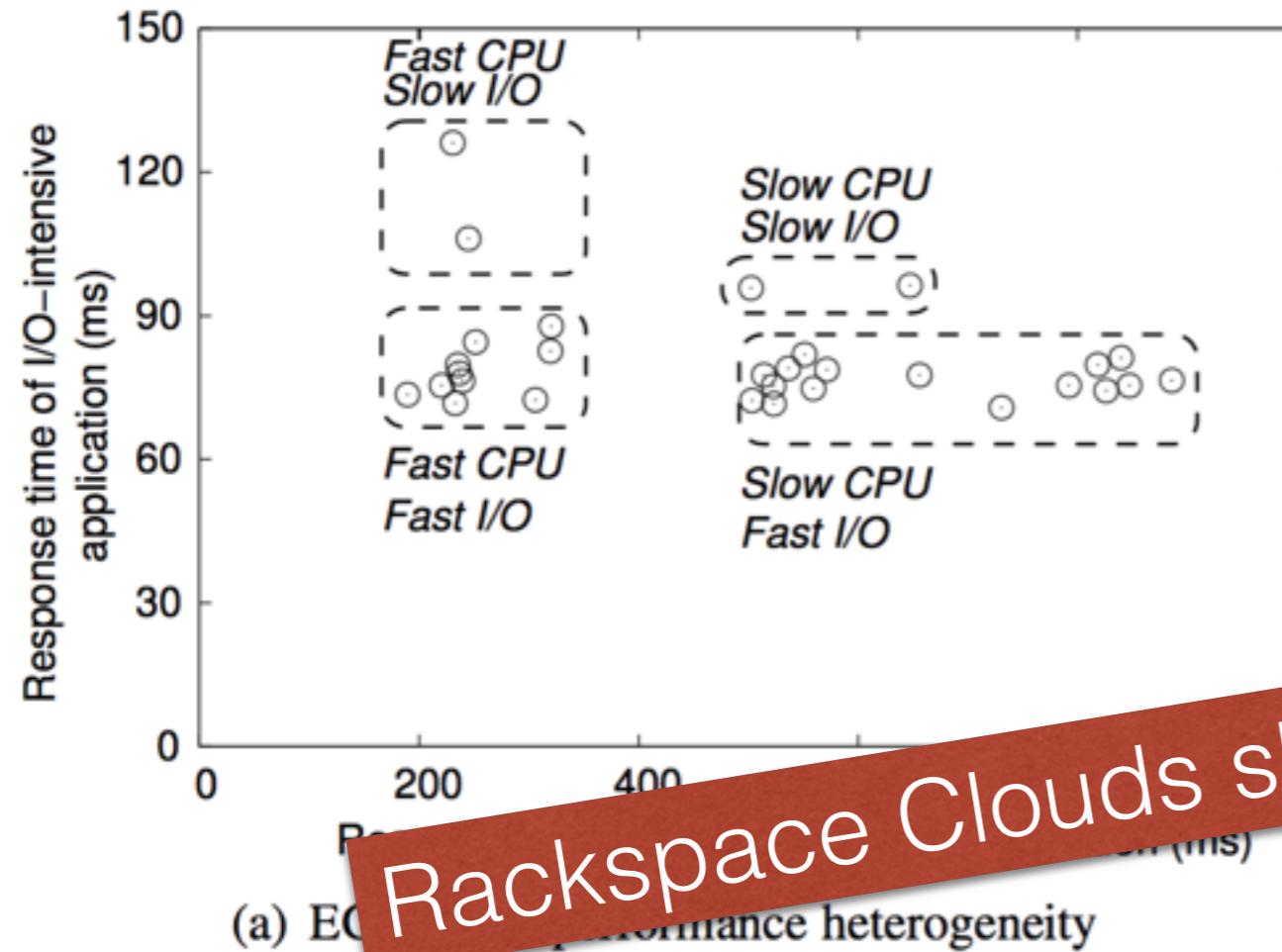
(a) EC2 Cloud performance heterogeneity

- Performance comparison of 30 'identical' EC2 instances

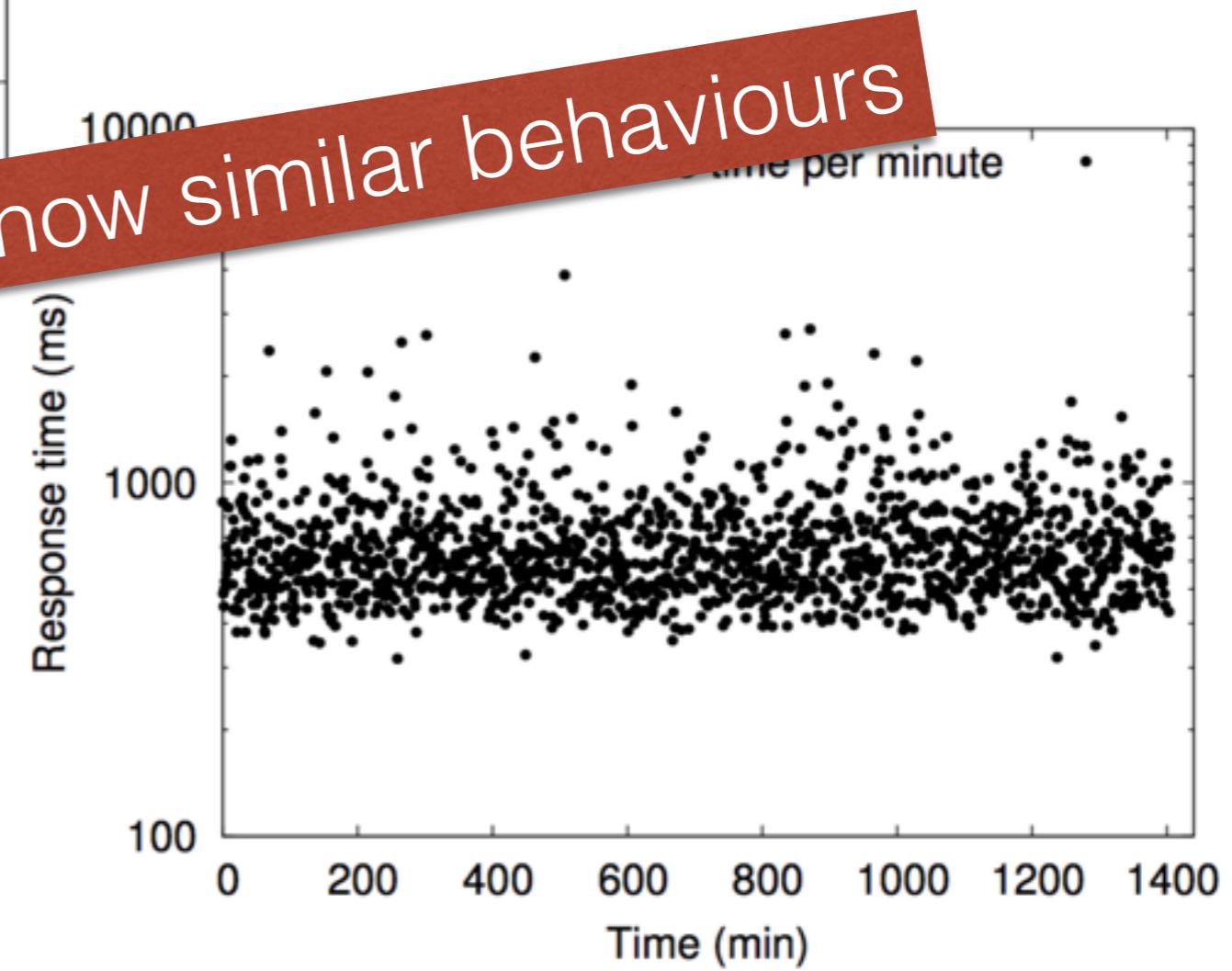


(b) Consistent performance of individual instance over time

# Performance reproducibility [Dejli]

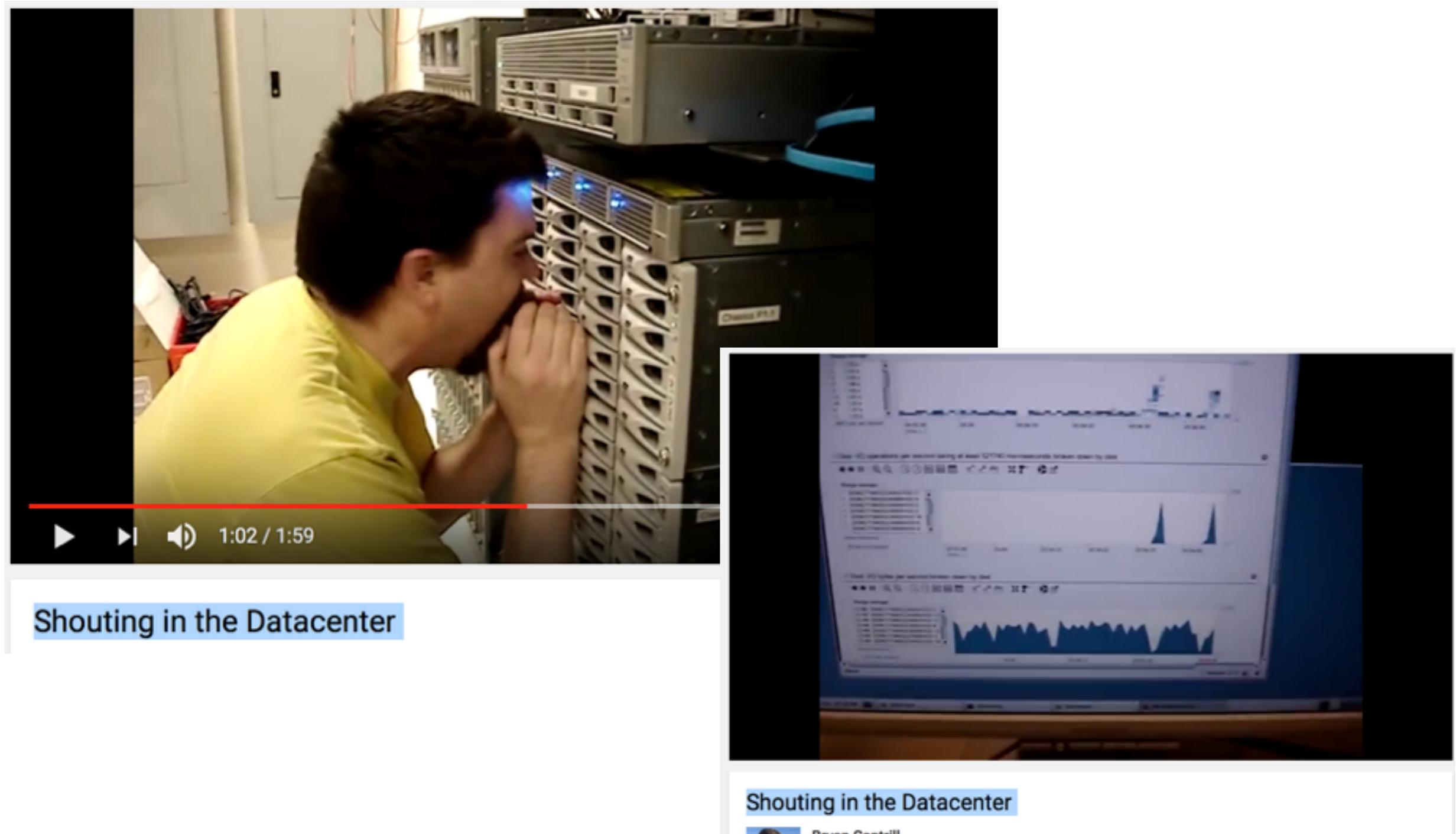


- Performance comparison of 30 'identical' EC2 instances



- Performance spikes  
duration: 1/3min  
Presumably caused by the  
launch/shutdown operations  
on other instances

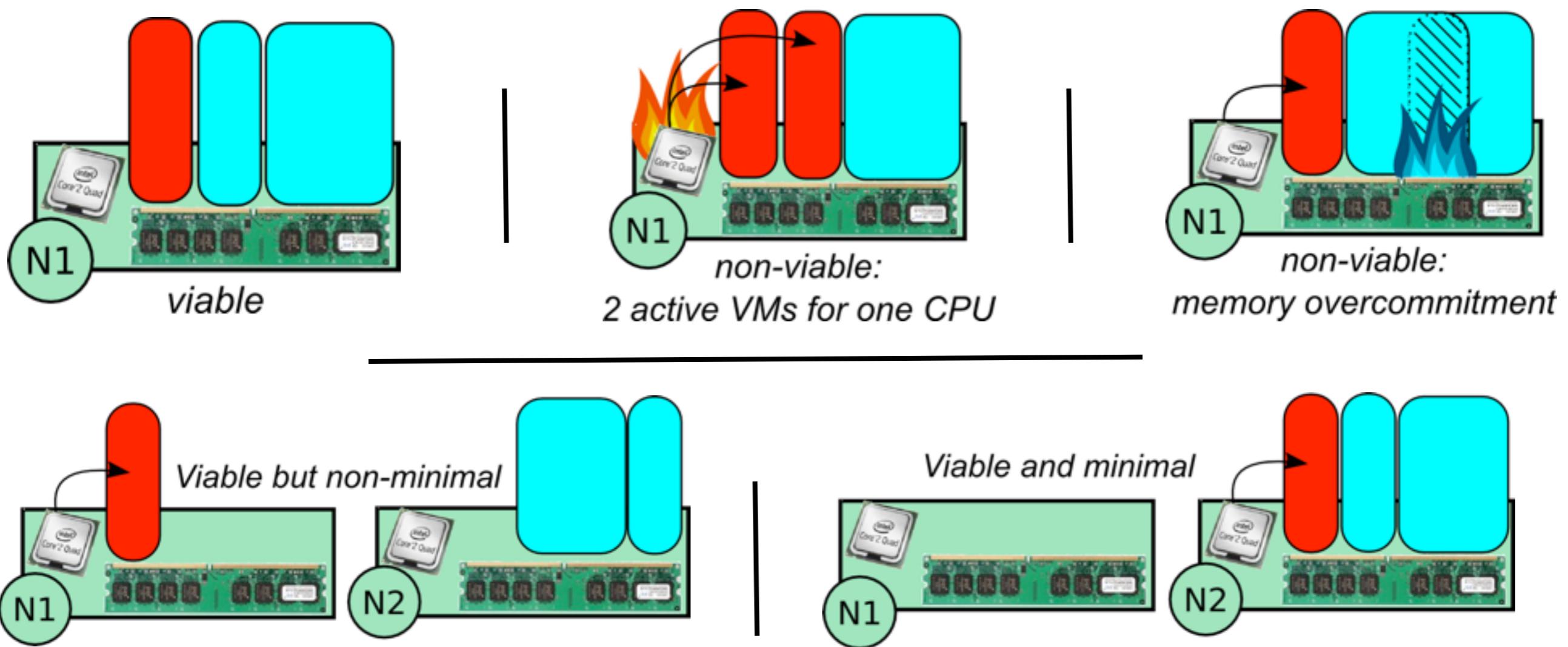
# Shouting in the Datacenter



<https://www.youtube.com/watch?v=tDacjrSCeq4>

# VM Placement and Performance

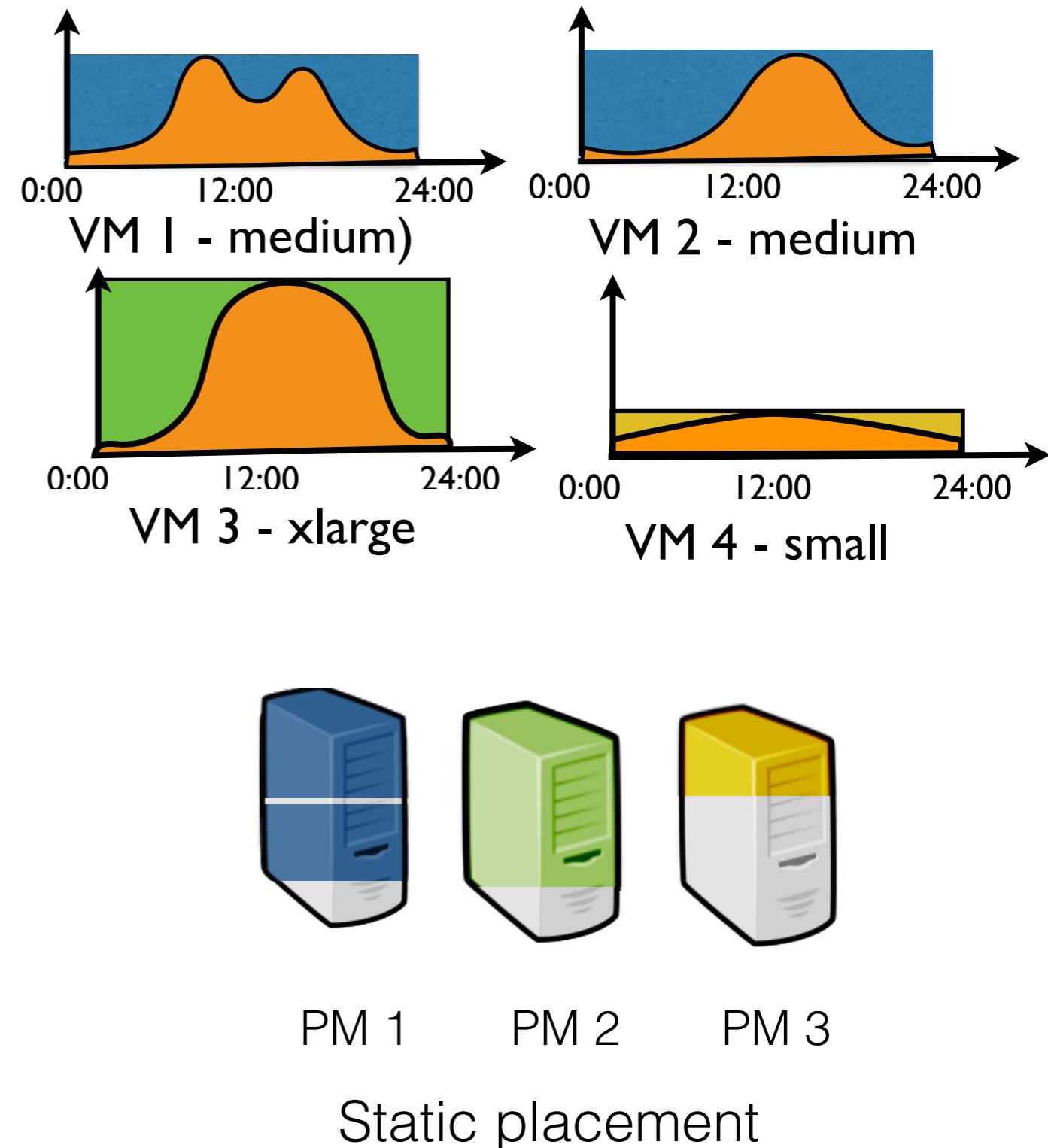
- Fine management of resources (efficiency and energy constraints)
- Find the “right” mapping between needs of VMs and resources provided by PMs



credits: F. Hermenier, OSDI poster session 2008

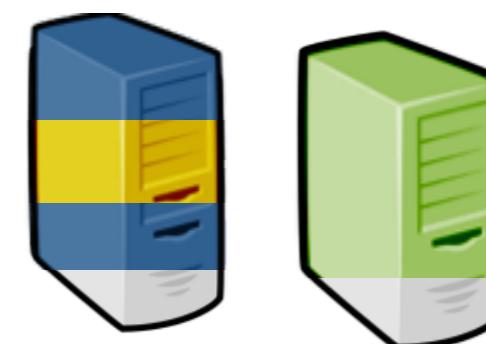
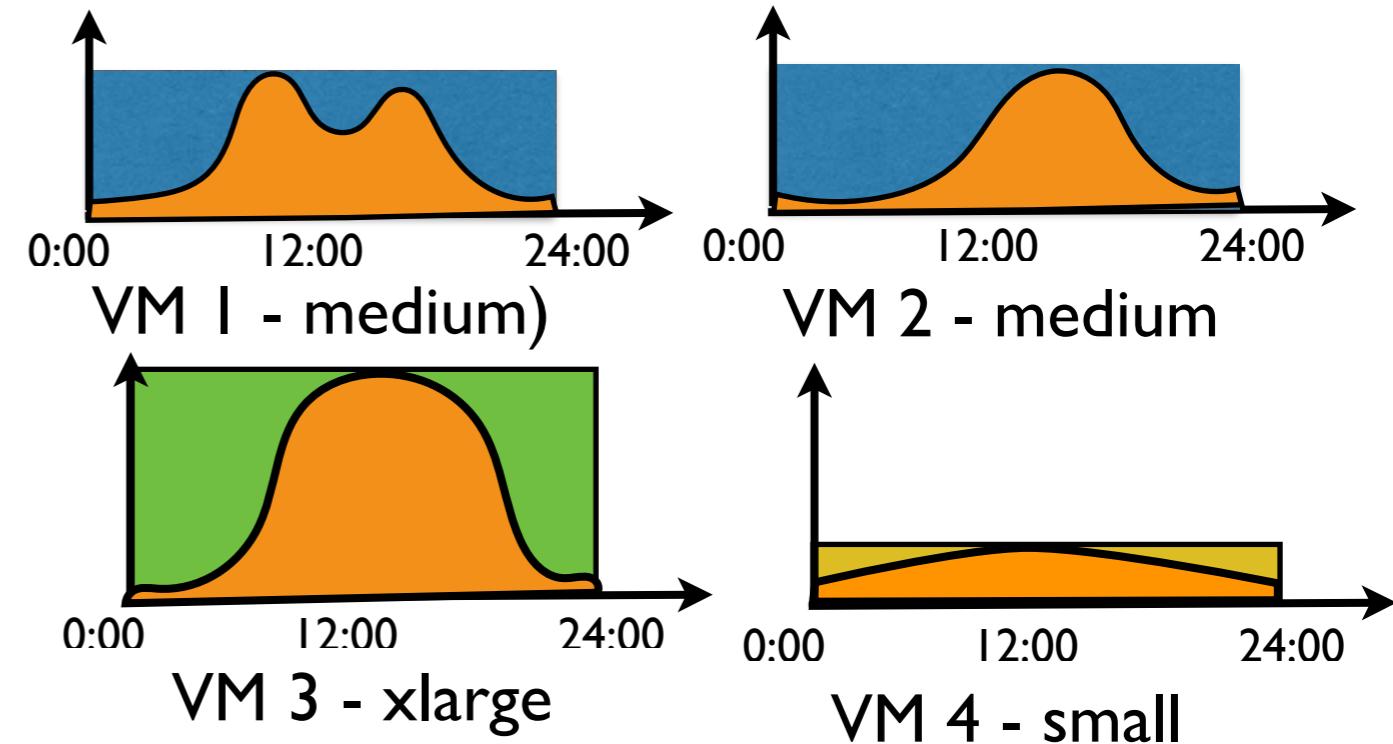
# Fluctuations of VM Requirements

- Static placement policies  
(as delivered by most of the popular Cloud Computing management systems)  
  
“Simple” but prevent CC providers to maximize the usage of CC resources (and thus their revenue)
- Advanced dynamic placement strategies to relocate VMs according to the scheduler objectives / available resources / waiting queue / ...



# Fluctuations of VM Requirements

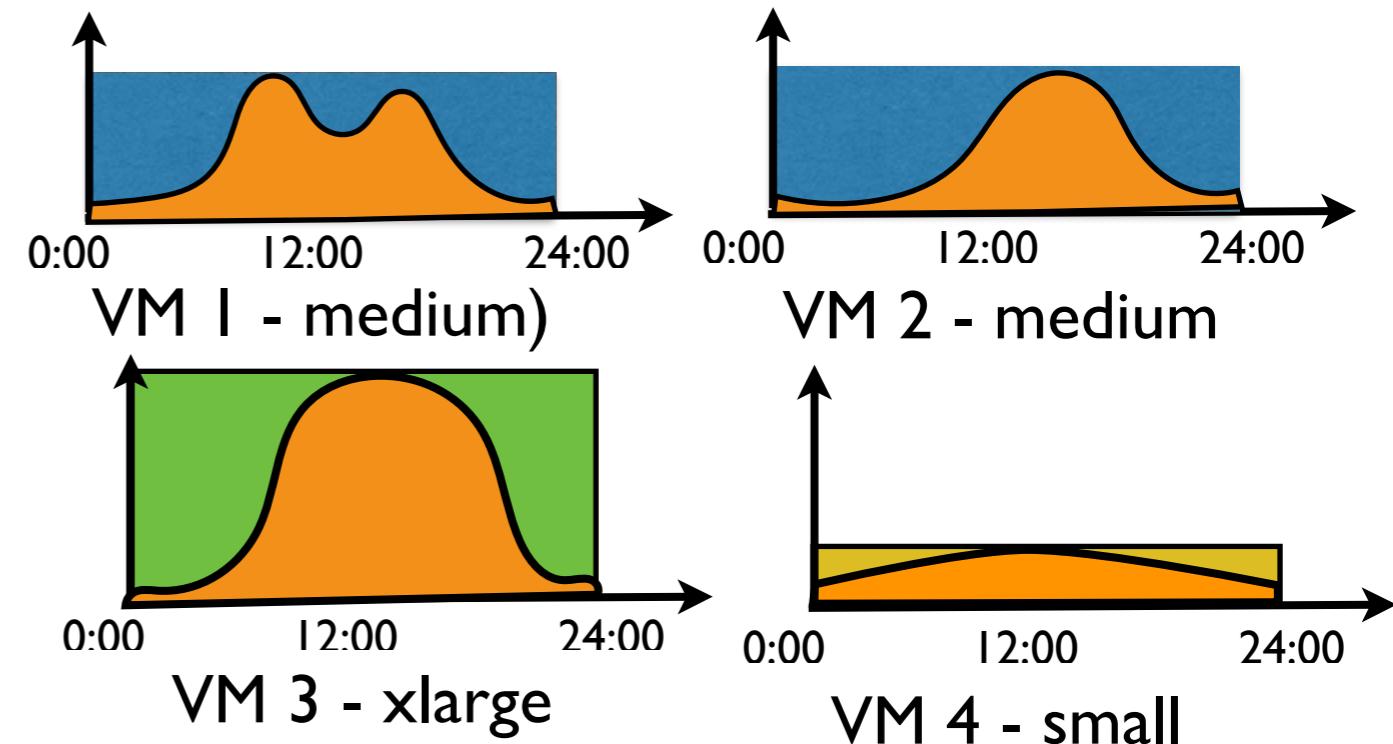
- Static placement policies  
(as delivered by most of the popular Cloud Computing management systems)  
  
“Simple” but prevent CC providers to maximize the usage of CC resources (and thus their revenue)
- Advanced dynamic placement strategies to relocate VMs according to the scheduler objectives / available resources / waiting queue / ...



PM 1      PM 2  
Dynamic placement

# Fluctuations of VM Requirements

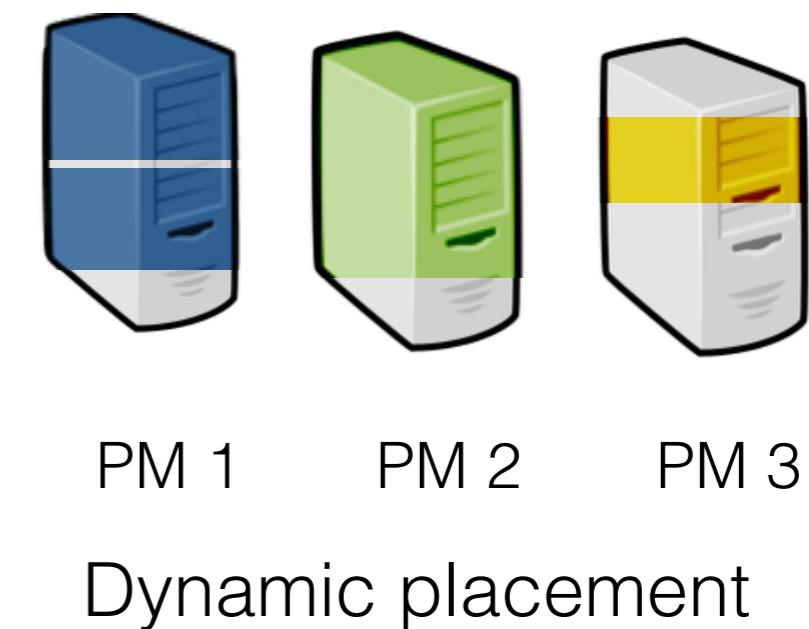
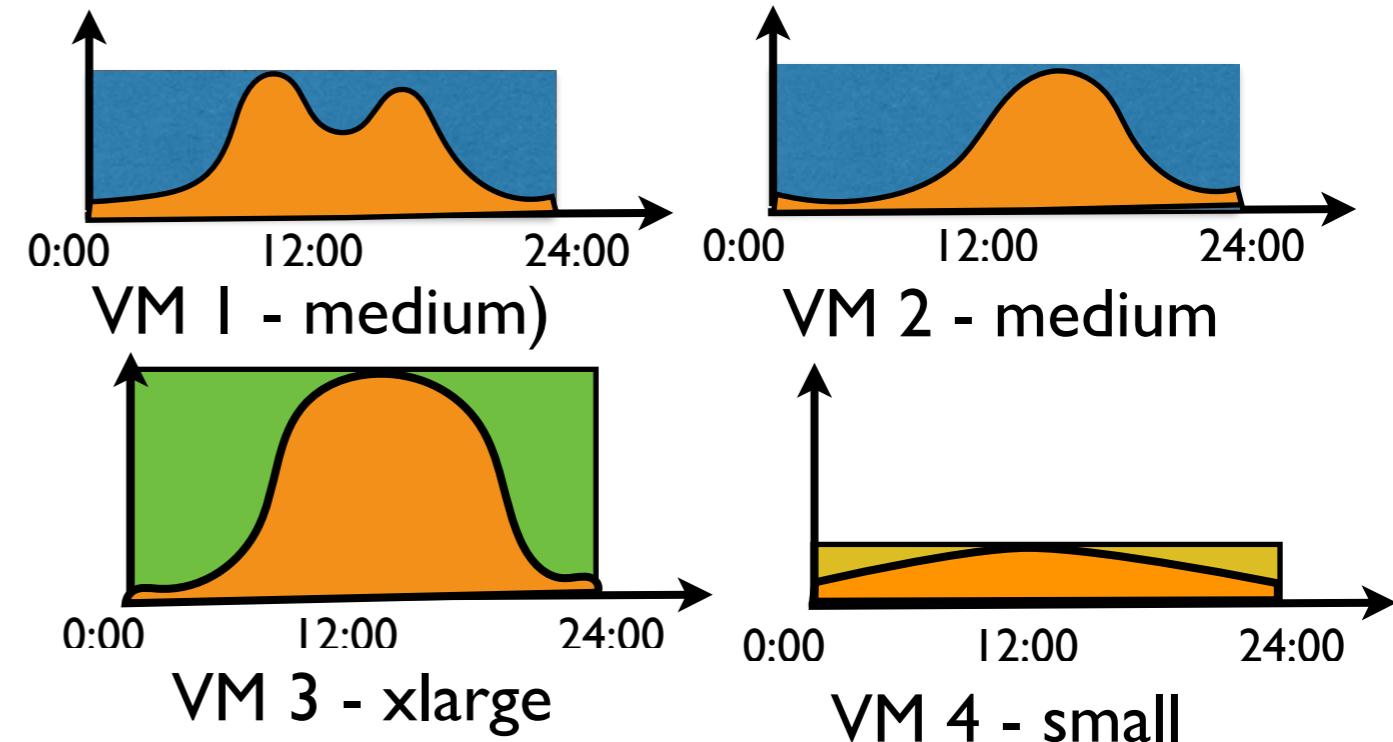
- Static placement policies  
(as delivered by most of the popular Cloud Computing management systems)  
  
“Simple” but prevent CC providers to maximize the usage of CC resources (and thus their revenue)
- Advanced dynamic placement strategies to relocate VMs according to the scheduler objectives / available resources / waiting queue / ...



PM 1      PM 2      PM 3  
Dynamic placement

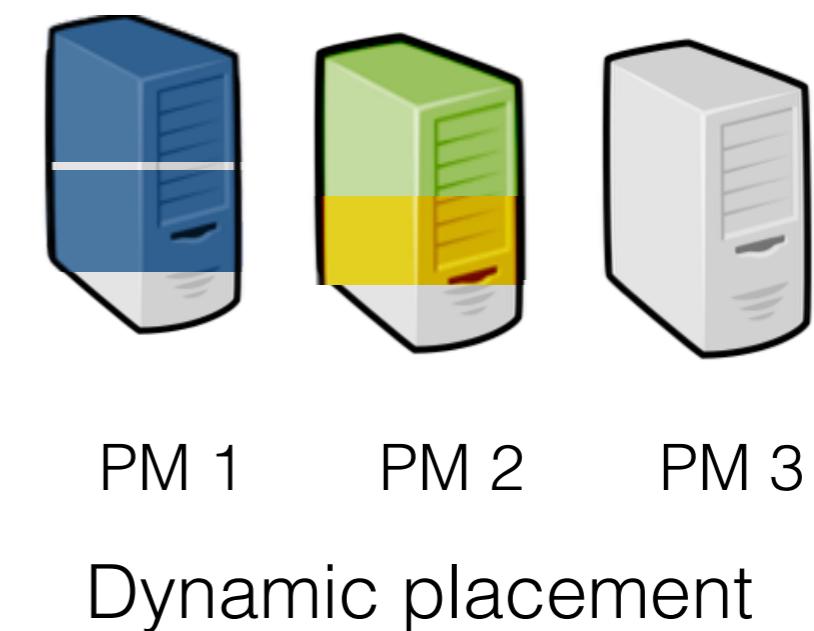
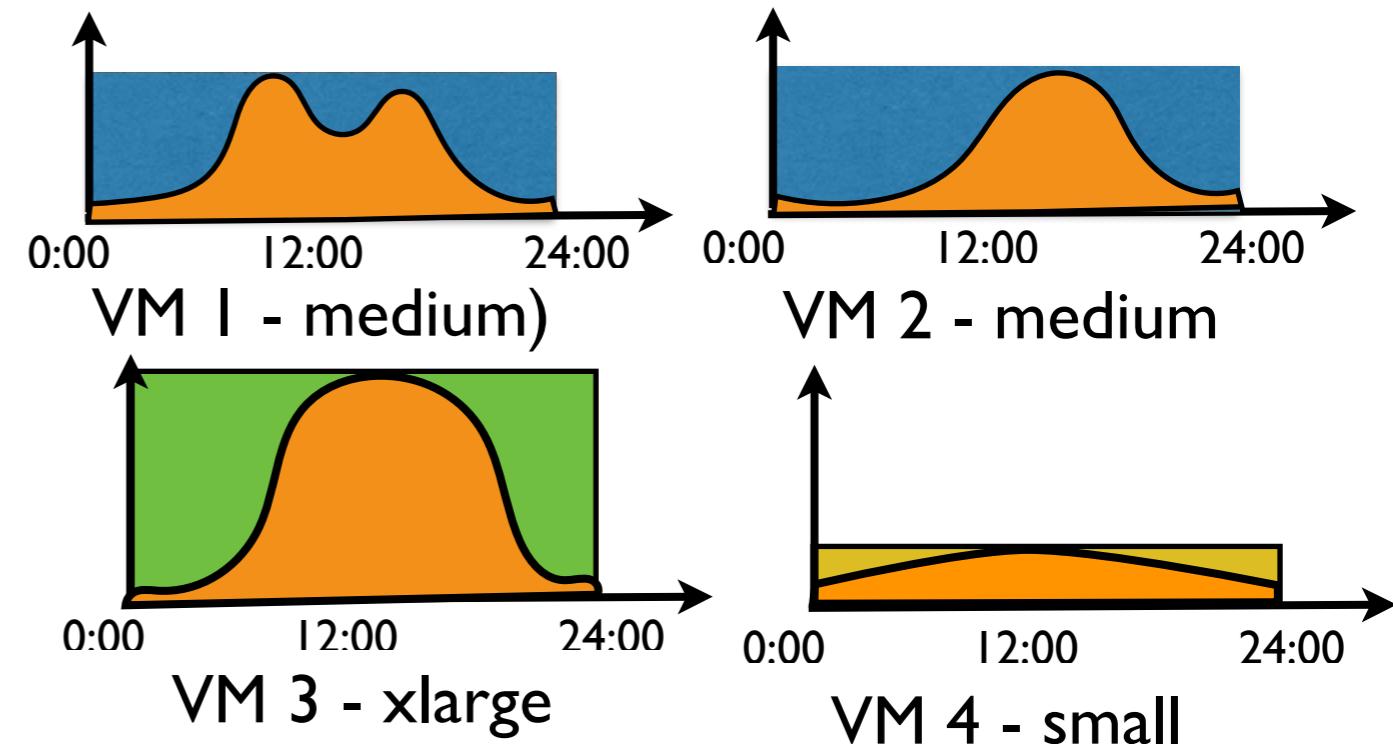
# Fluctuations of VM Requirements

- Static placement policies  
(as delivered by most of the popular Cloud Computing management systems)  
  
“Simple” but prevent CC providers to maximize the usage of CC resources (and thus their revenue)
- Advanced dynamic placement strategies to relocate VMs according to the scheduler objectives / available resources / waiting queue / ...



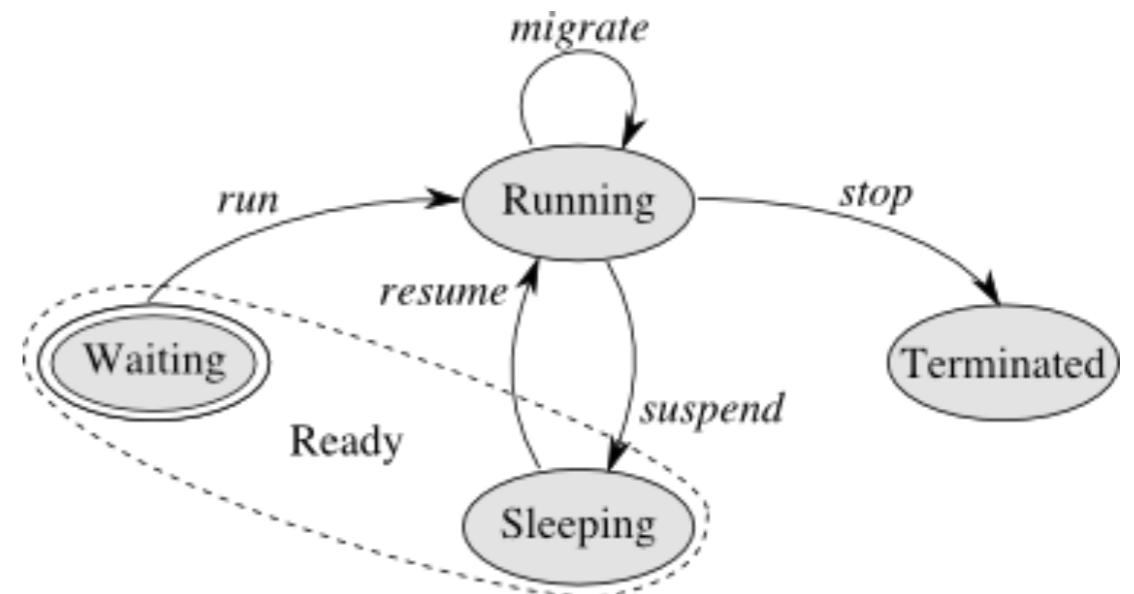
# Fluctuations of VM Requirements

- Static placement policies  
(as delivered by most of the popular Cloud Computing management systems)  
  
“Simple” but prevent CC providers to maximize the usage of CC resources (and thus their revenue)
- Advanced dynamic placement strategies to relocate VMs according to the scheduler objectives / available resources / waiting queue / ...



# Dynamic VM Placement Policies

- Generale idea: leverage VM capabilities to manipulate **VEs** in a similar way of usual processes on a laptop (a VE is a users' working environment, possibly composed of several interconnected VMs)
- Each VE is in a particular state
  - Perform VE context switches (a set of VM context switches) to reschedule/rebalance the LUC infrastructure [Her10]



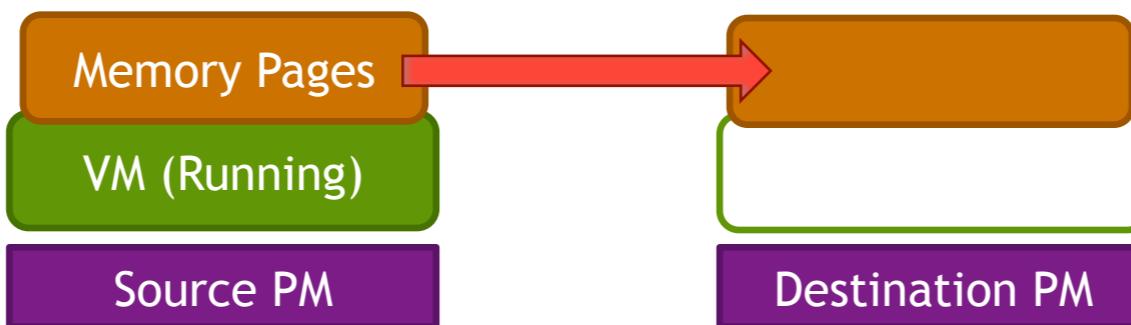
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



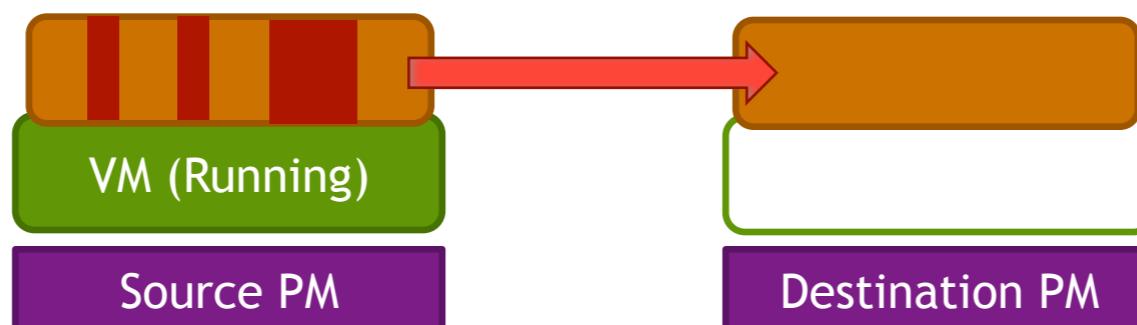
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



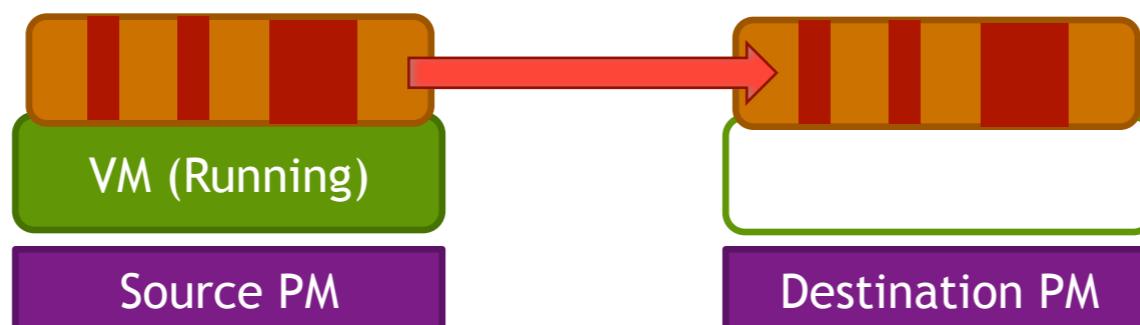
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



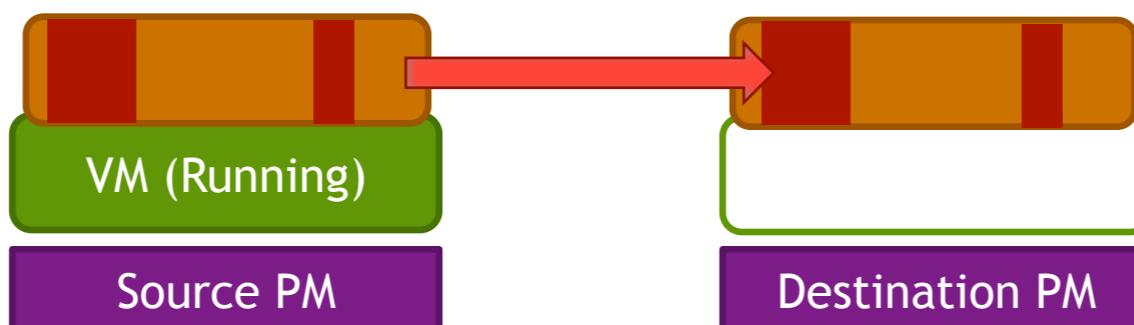
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



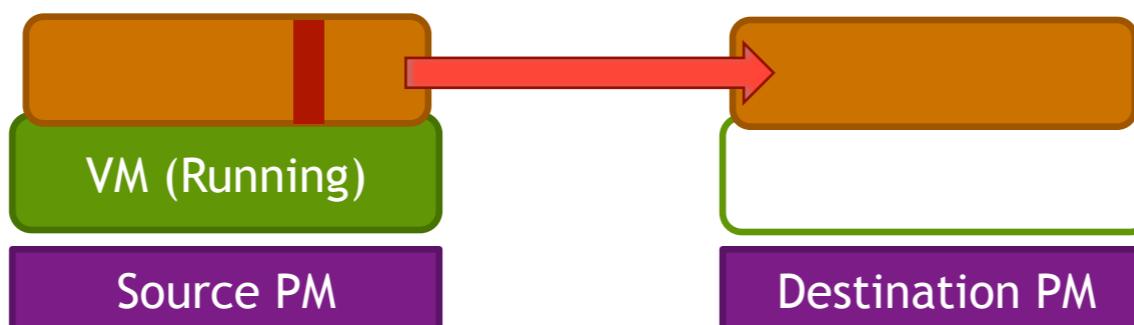
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



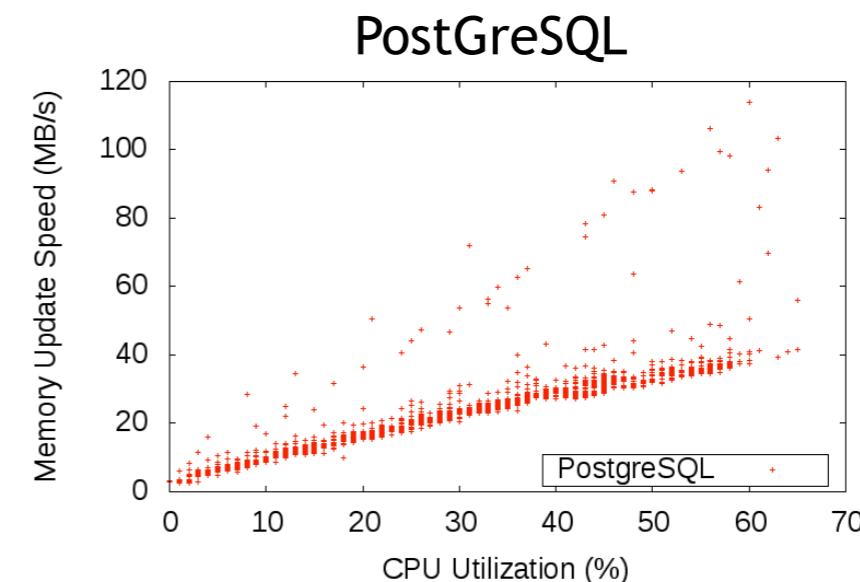
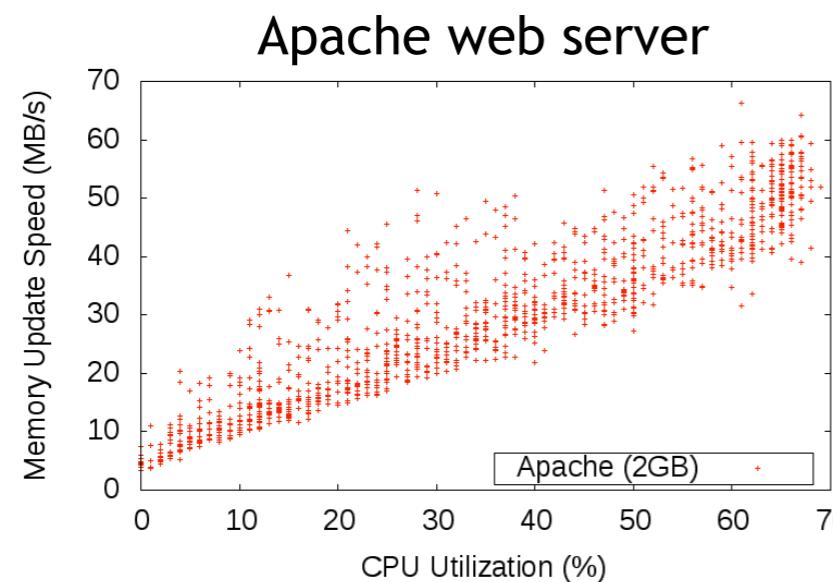
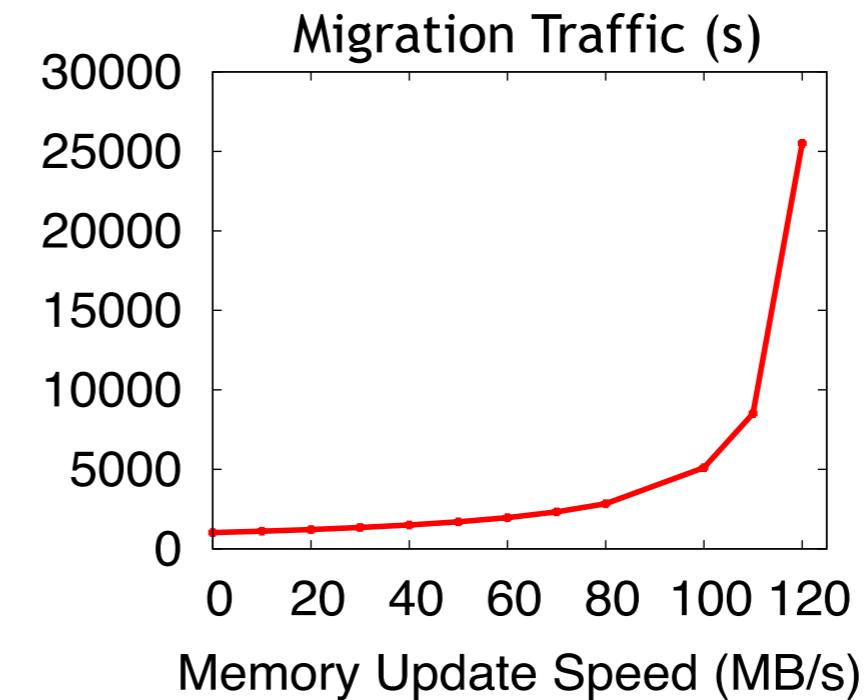
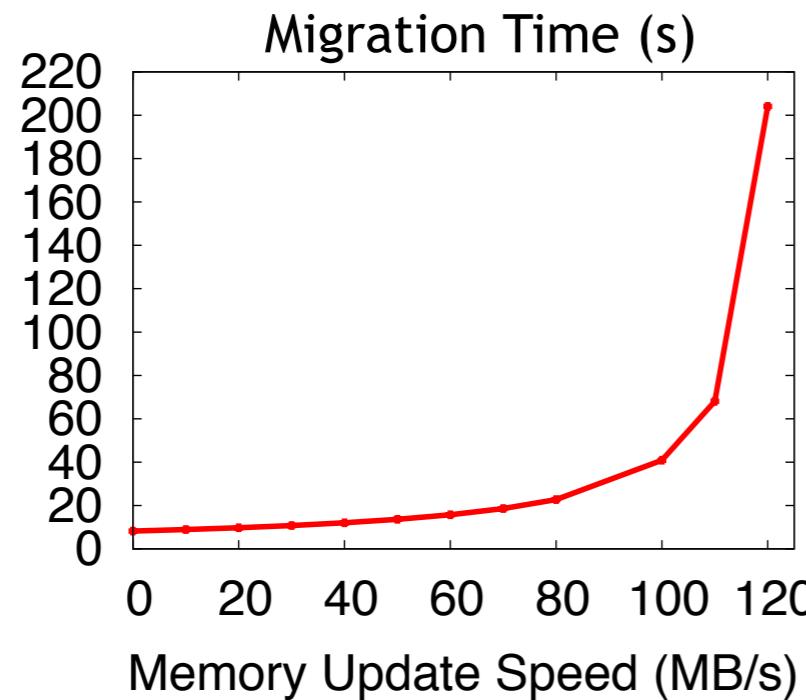
# Zoom on Live Migration

- Transfer VM's states to destination without stopping the guest OS (pre-copy algorithm)
  - I. Transfer all memory pages of the VM.  
(But, keep in mind the VM is still running at source.)
  2. Transfer updated memory pages during the previous step
  3. Iterate this step until the rest of memory pages becomes sufficiently small to meet an acceptable downtime (30ms in KVM).
  4. Stop the VM. Transfer the rest of memory pages and states



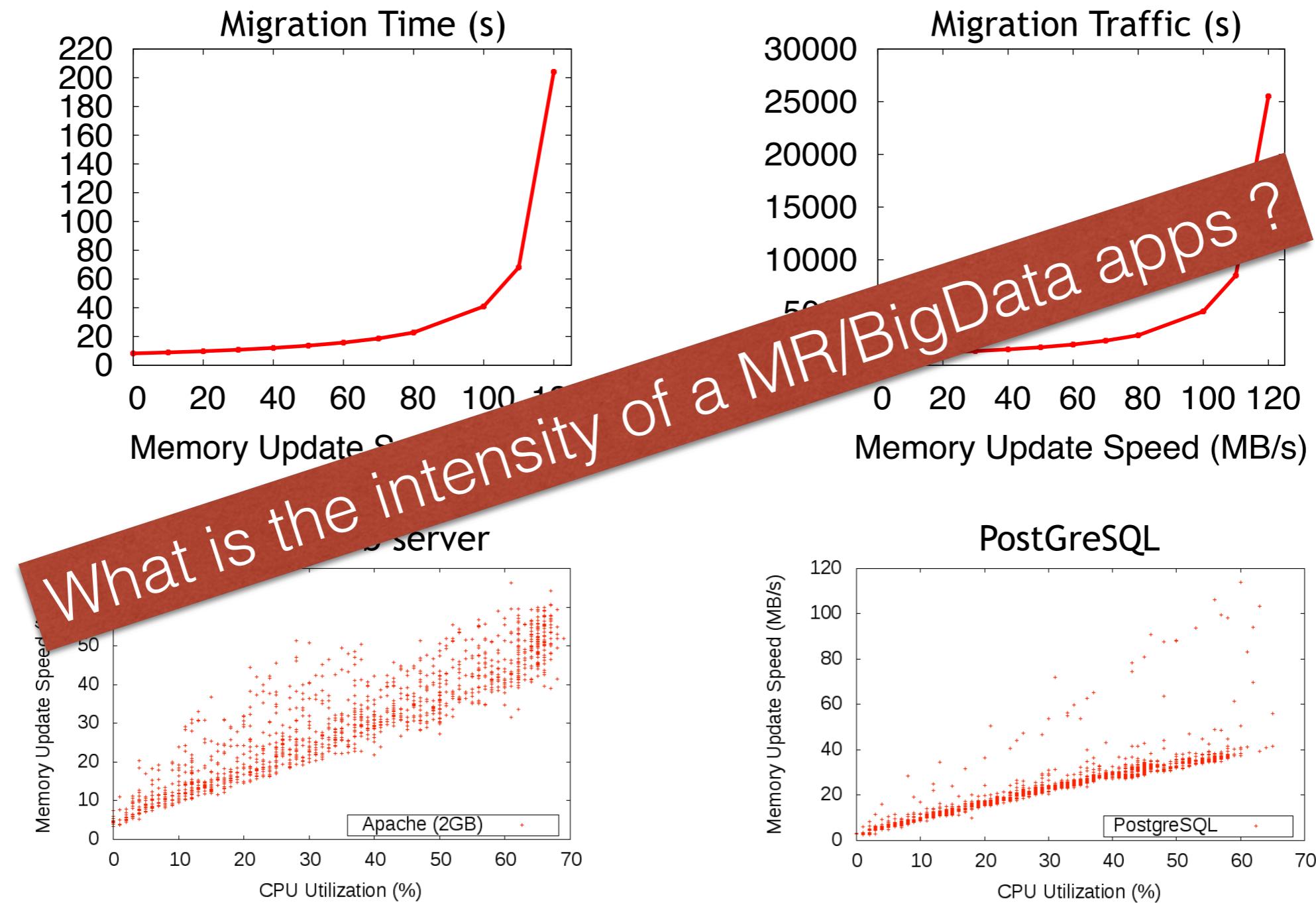
# Zoom on the live migration operation

- The more your VM is memory intensive, the longer the migration will be [Hiro13]

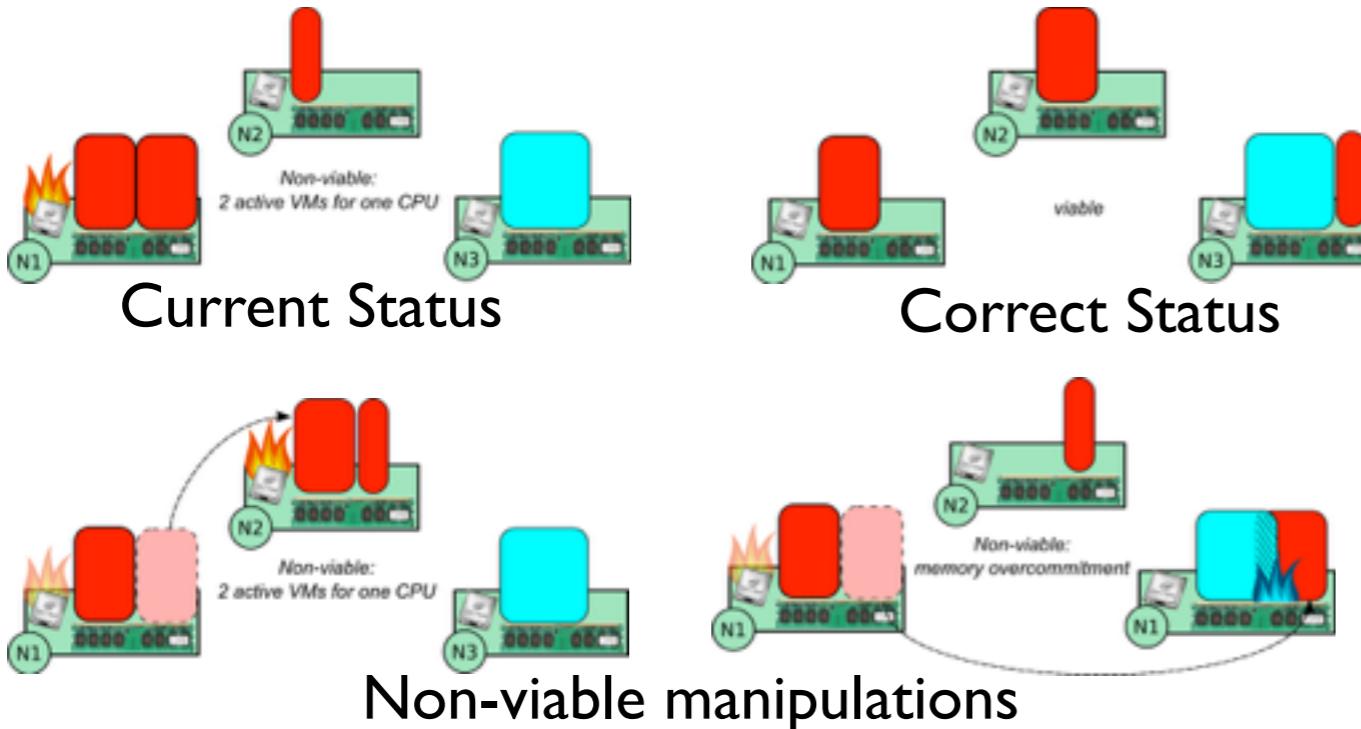


# Zoom on the live migration operation

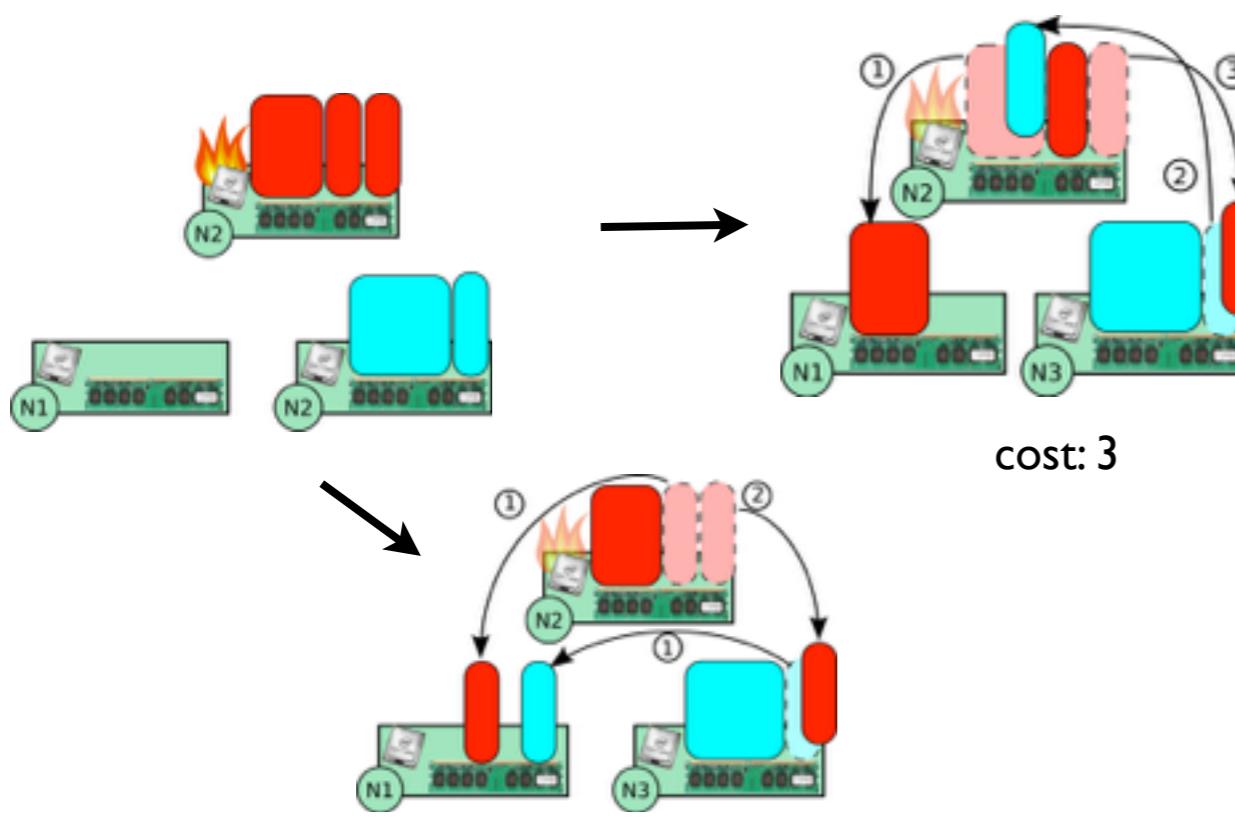
- The more your VM is memory intensive, the longer the migration will be [Hiro13]



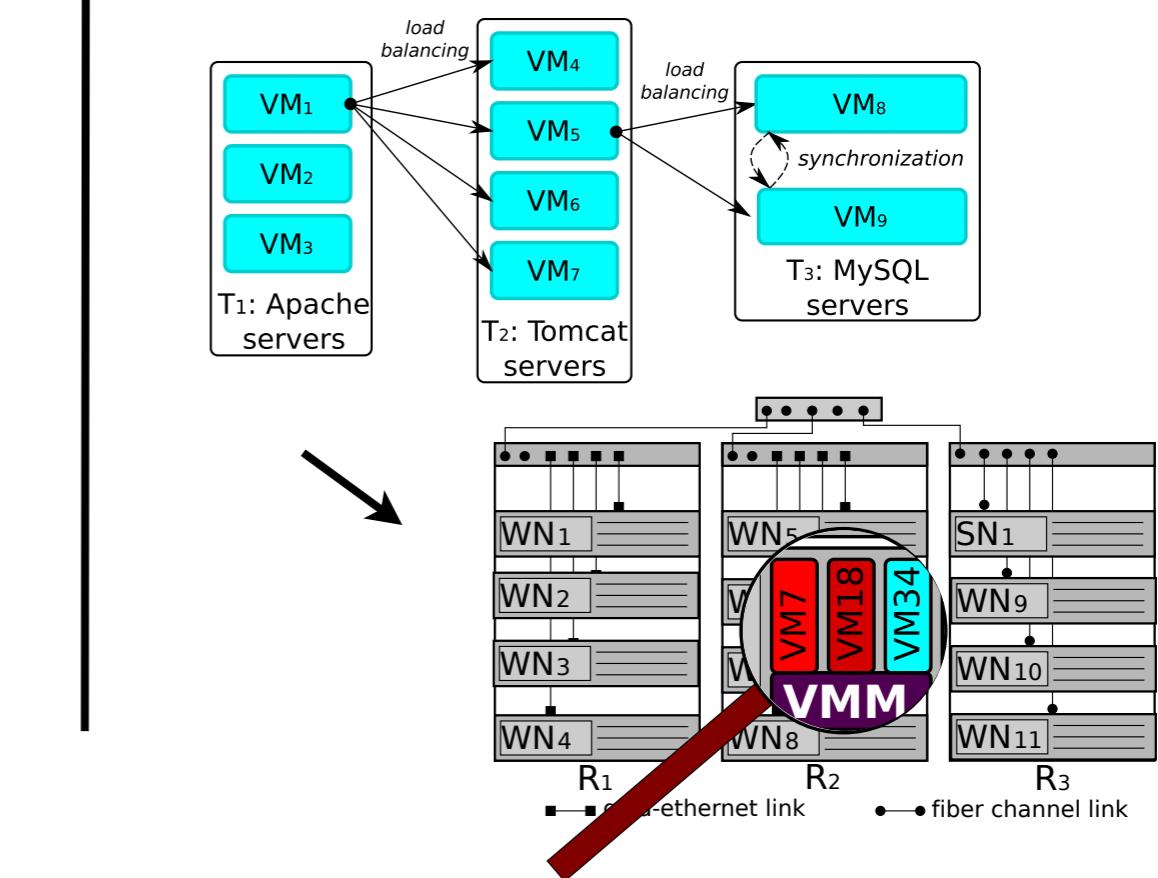
# Placement constraints (btrPlace)



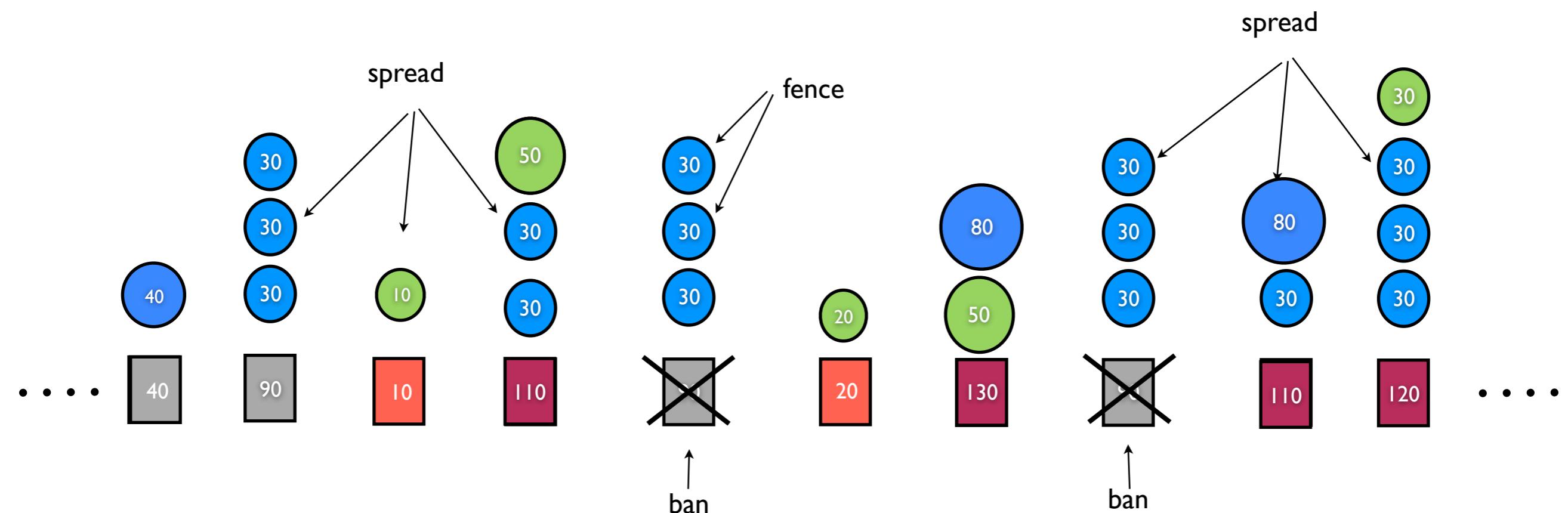
- Find the “right” mapping between needs of VMs, their constraints and resources provided by PMs [Her13]



credits: F. Hermenier, Sophia Antipolis University, [www.btrplace.org](http://www.btrplace.org)

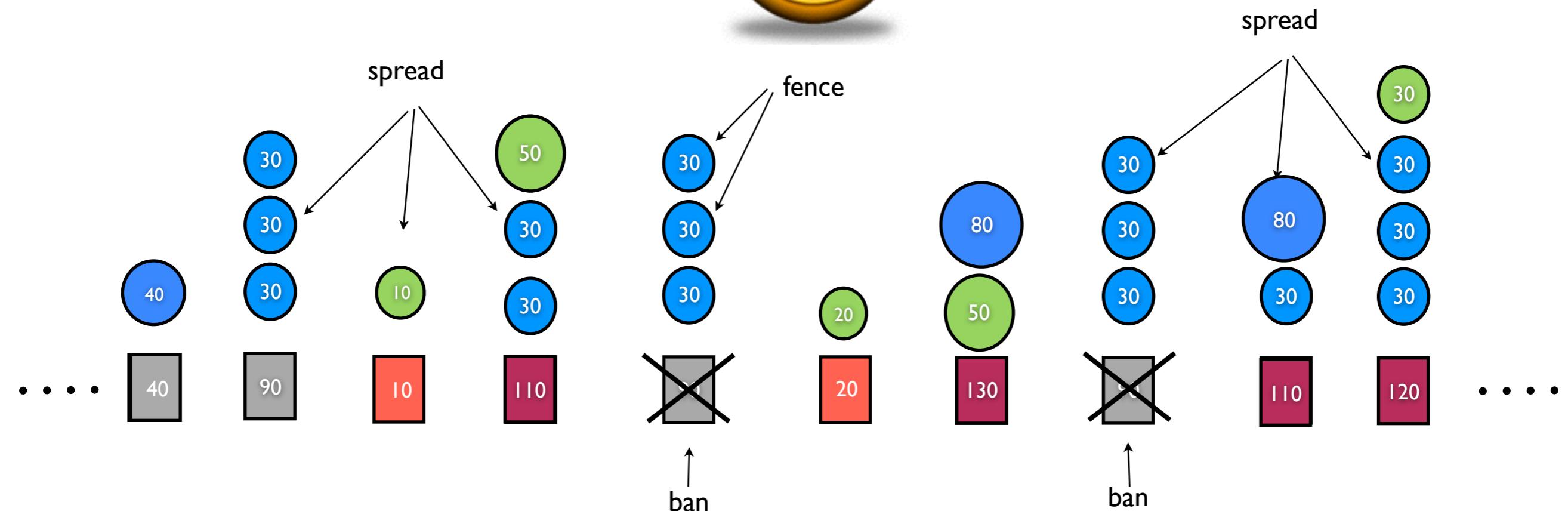


# a Small Example



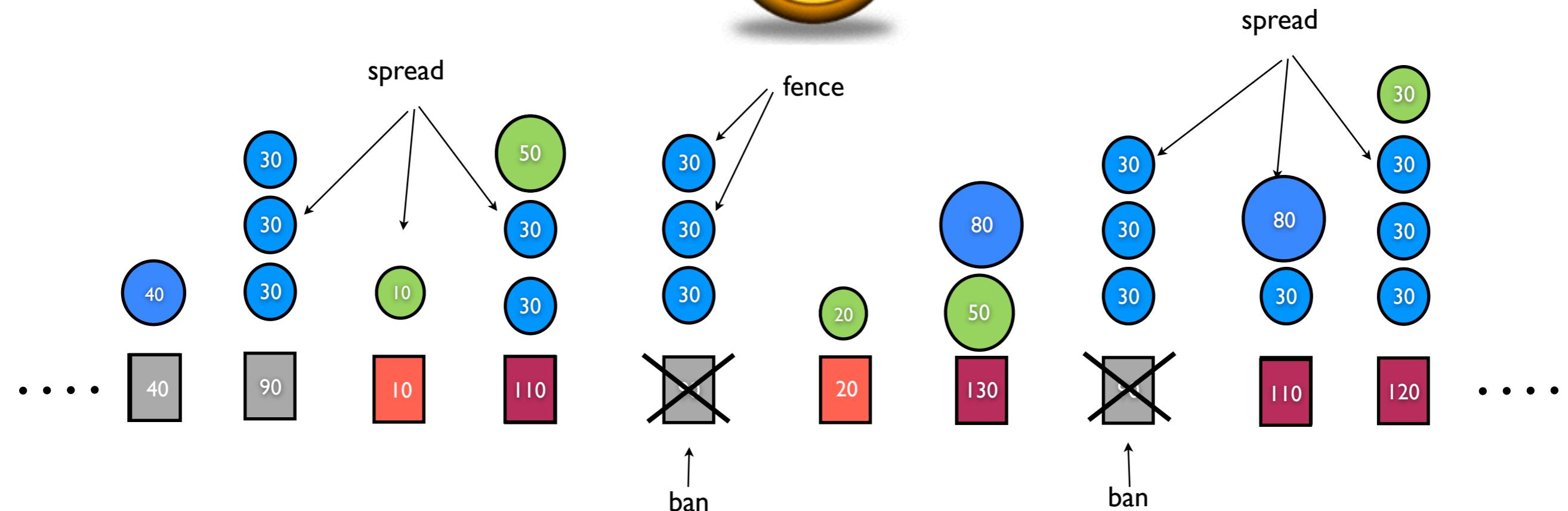
# a Small Example

# Only CPU is considered in this simple example



# a Small Example

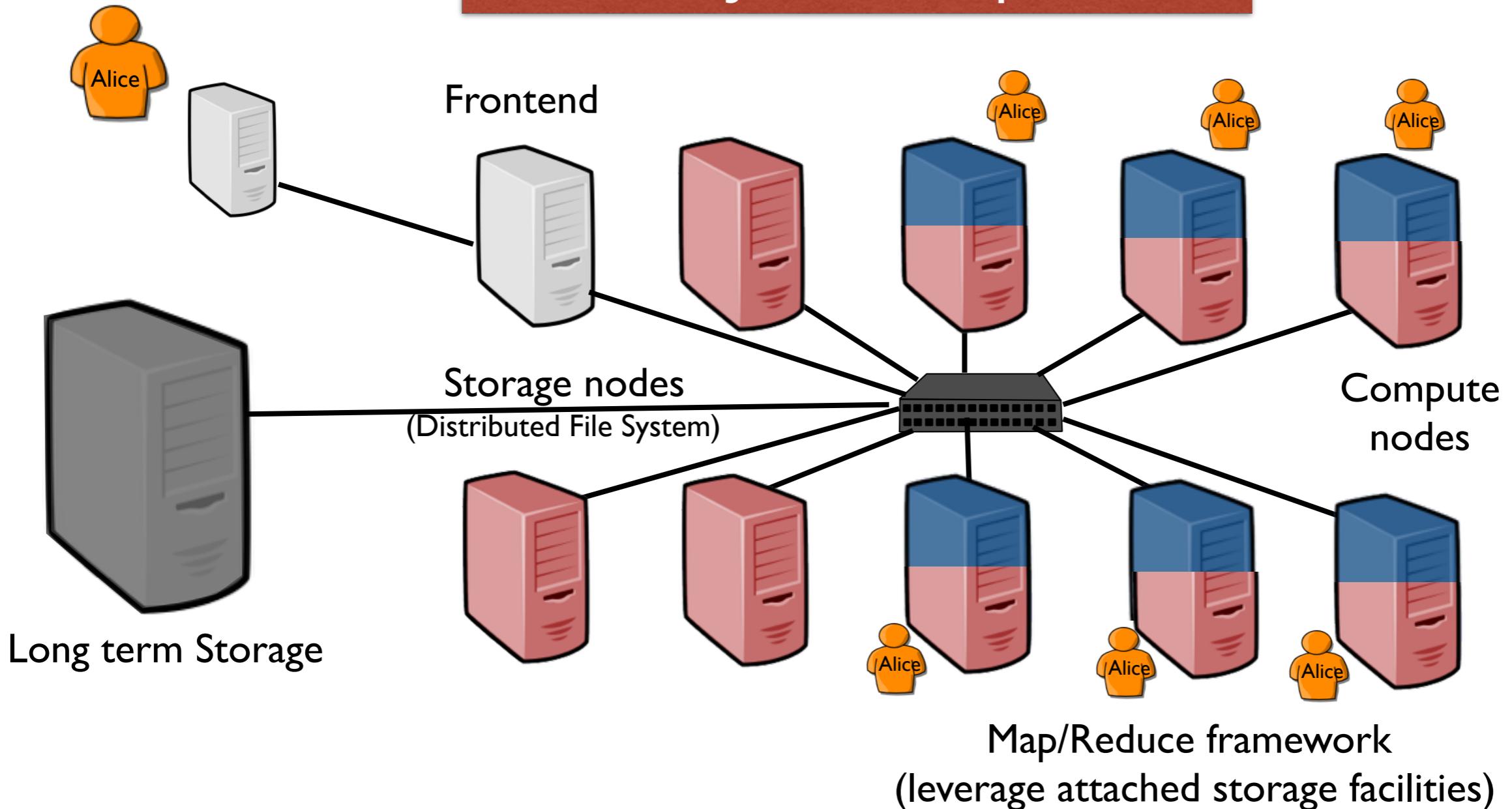
Only CPU is considered in this simple example



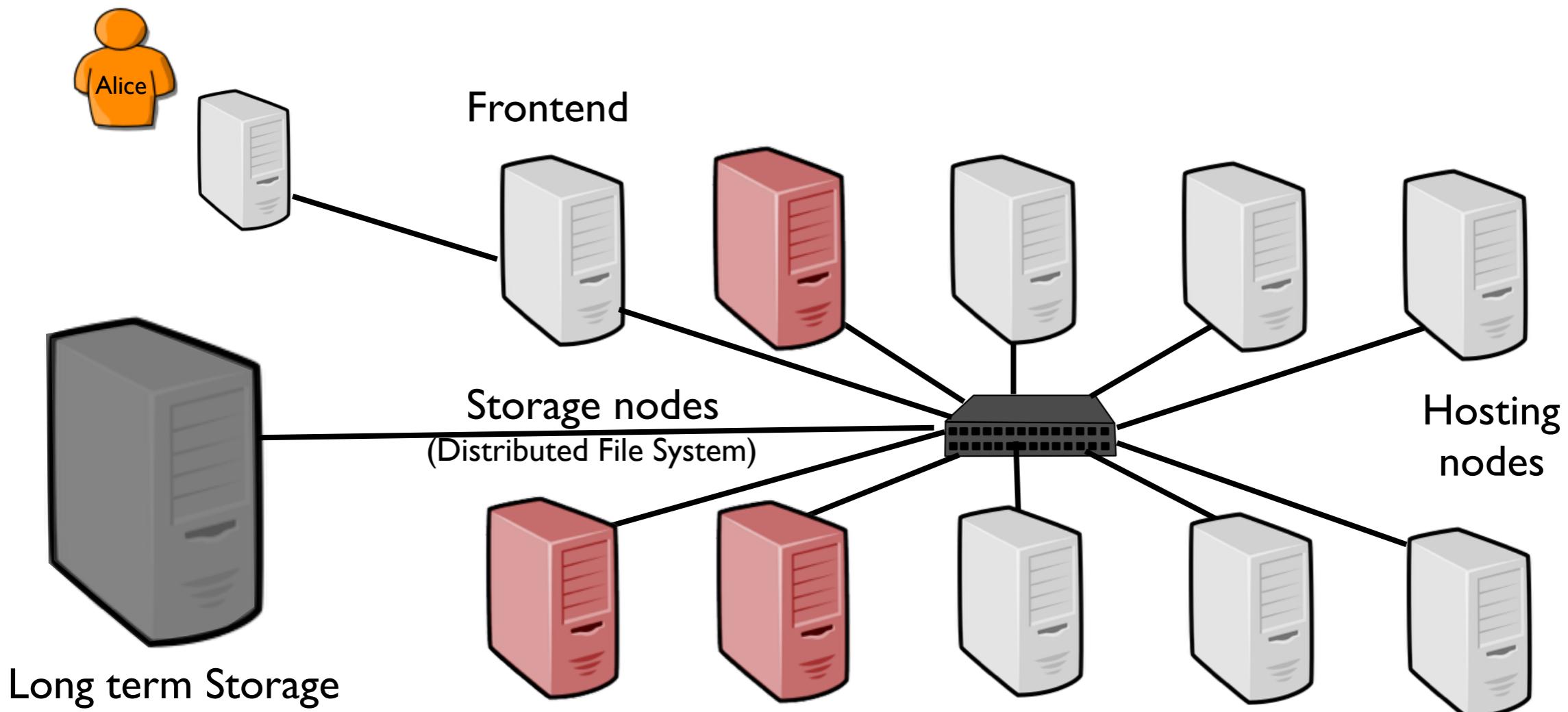
**Lesson 5: F. Hermenier**

# Another with Map/Reduce

What you expect !

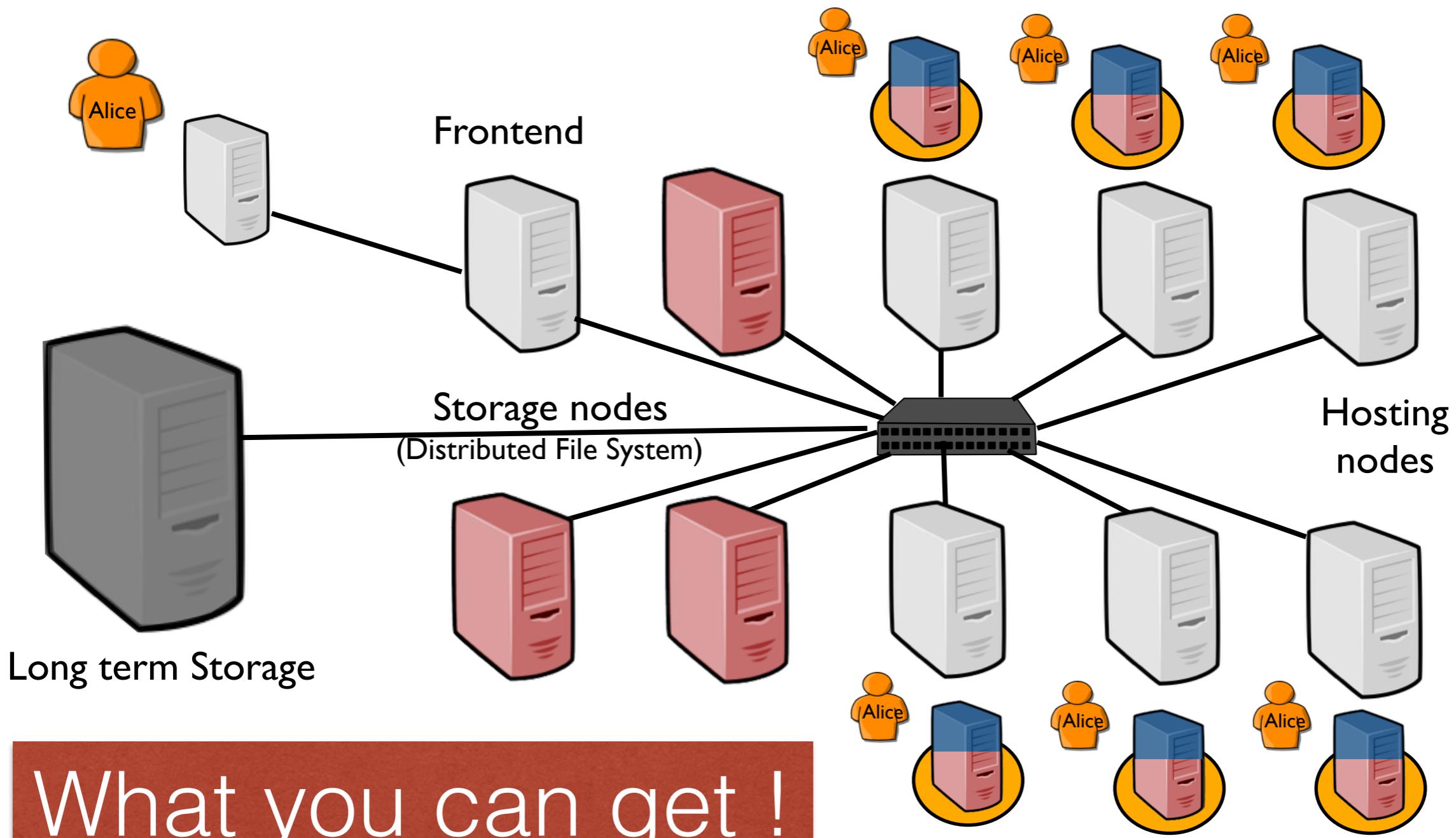


# Another with Map/Reduce

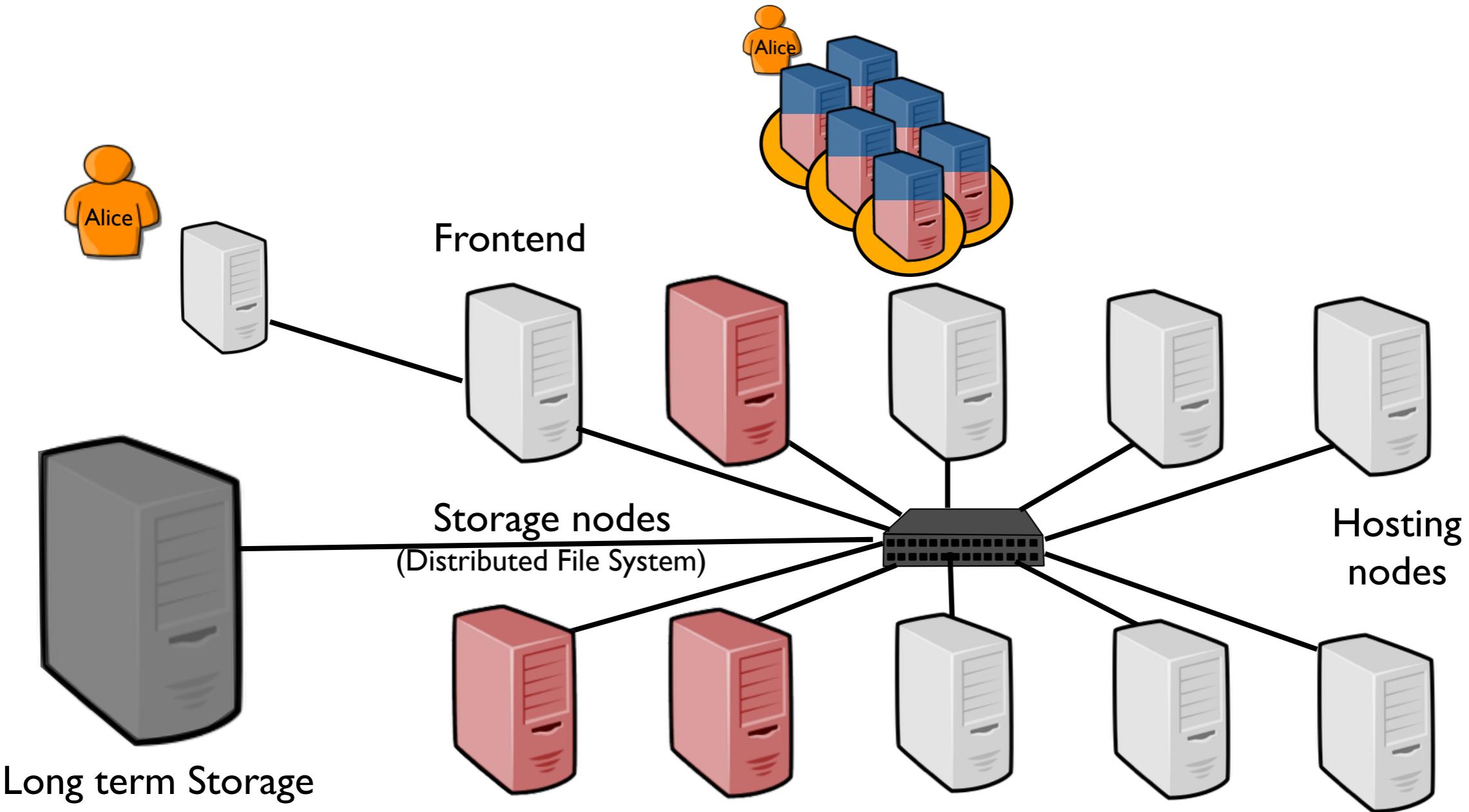


What you can get !

# Another with Map/Reduce

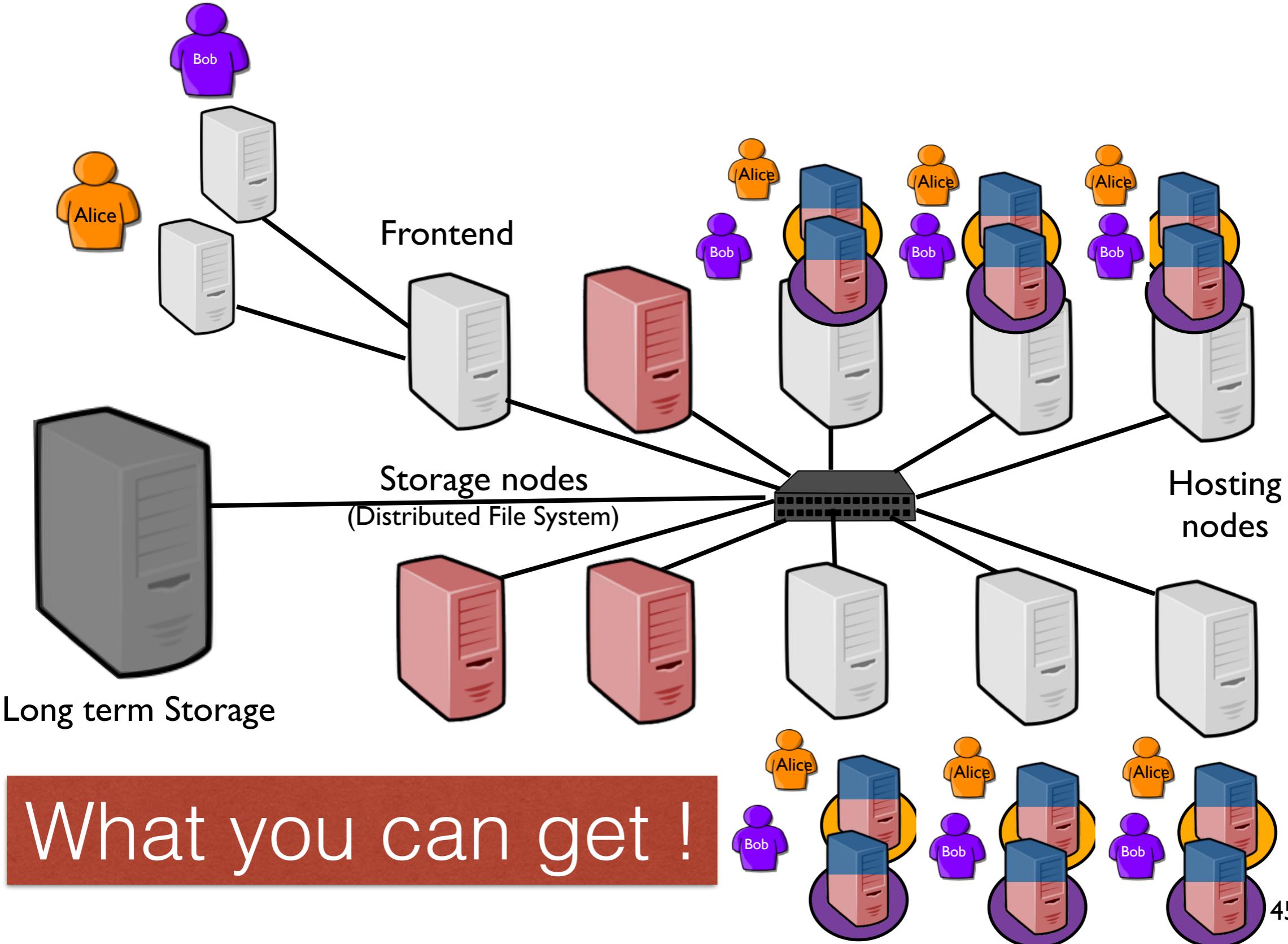


# Another with Map/Reduce



What you can get !

# Another with Map/Reduce



# Virtualization and Performance

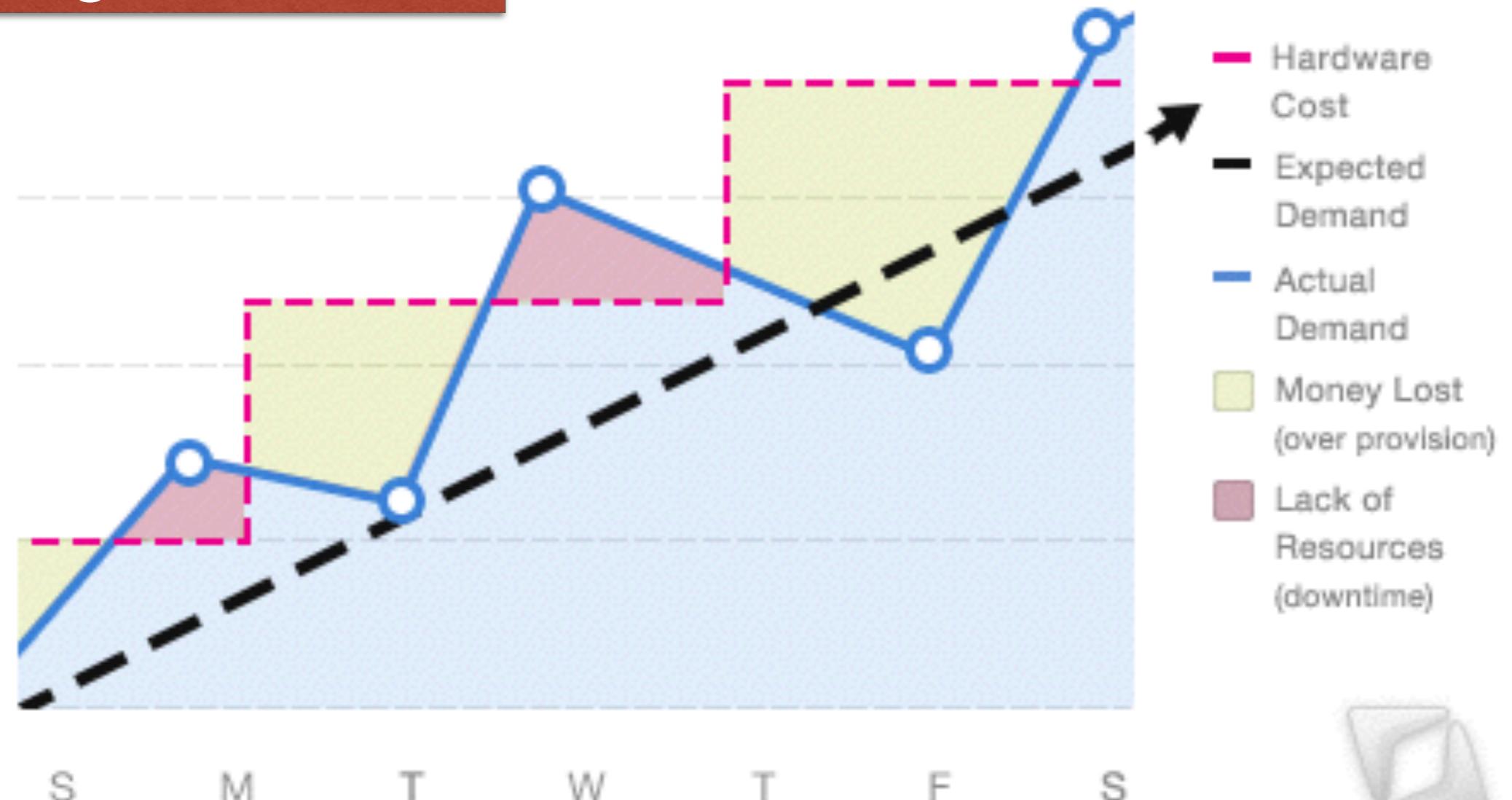
- Virtualization
  - Contextualization / portability / security “isolation” 😛
  - Hard to guarantee (reproducible) performance 😐
- Scheduling:
  - Mainly static ⇒ lead to energy/resource wastes
  - Dynamic scheduling strategies ⇒ Good achievements but still “food” for researchers (SLAs, migration overheads, ....)
- Mitigate/Control performance issues :
  - Nested virtualisation / **Containers** / Applications (autoscaling)
- I/O isolation/consolidation
  - An important challenge



# Autoscaling Mechanisms (few words)

Provisioning take time...

CALE DEMAND

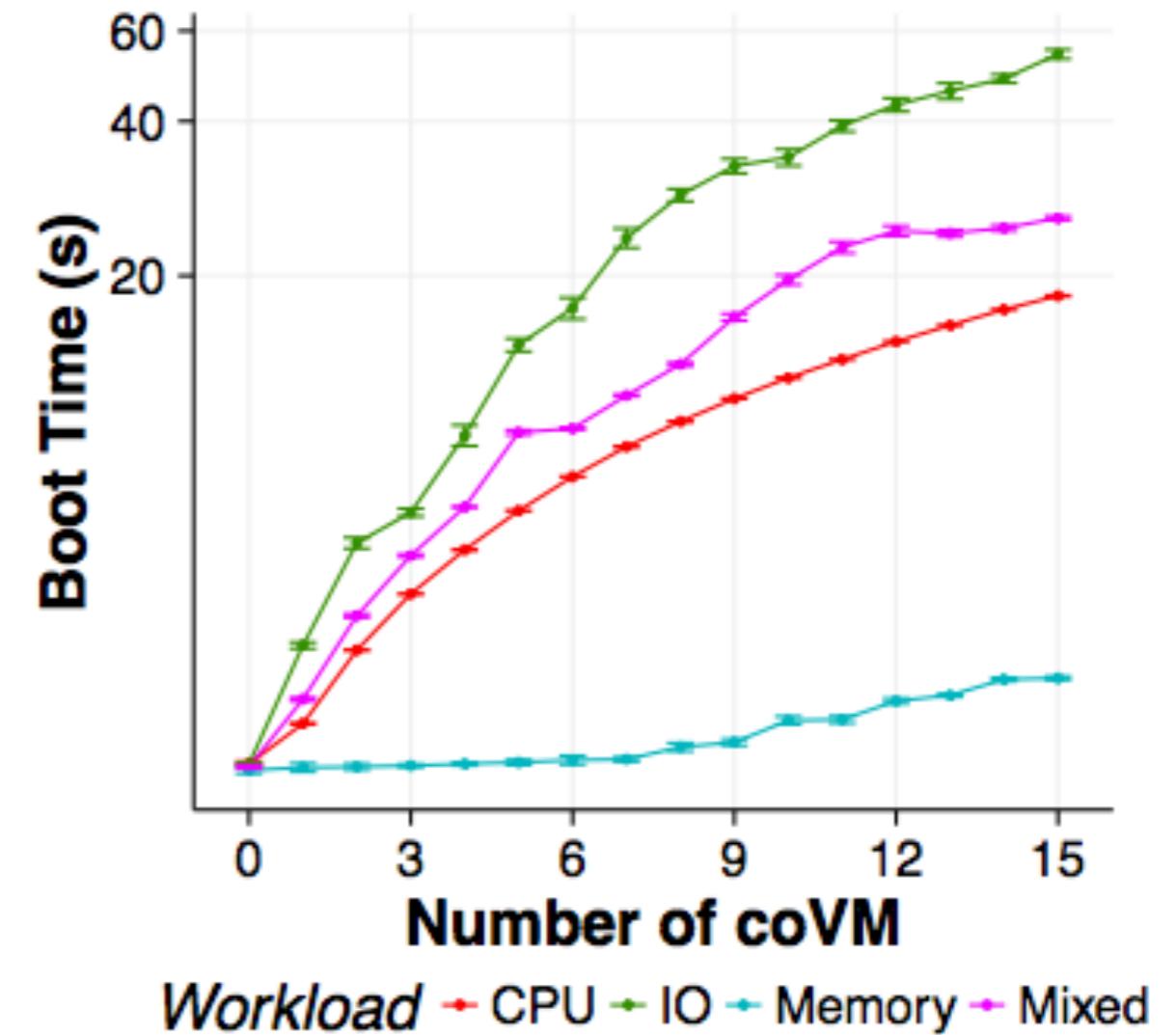
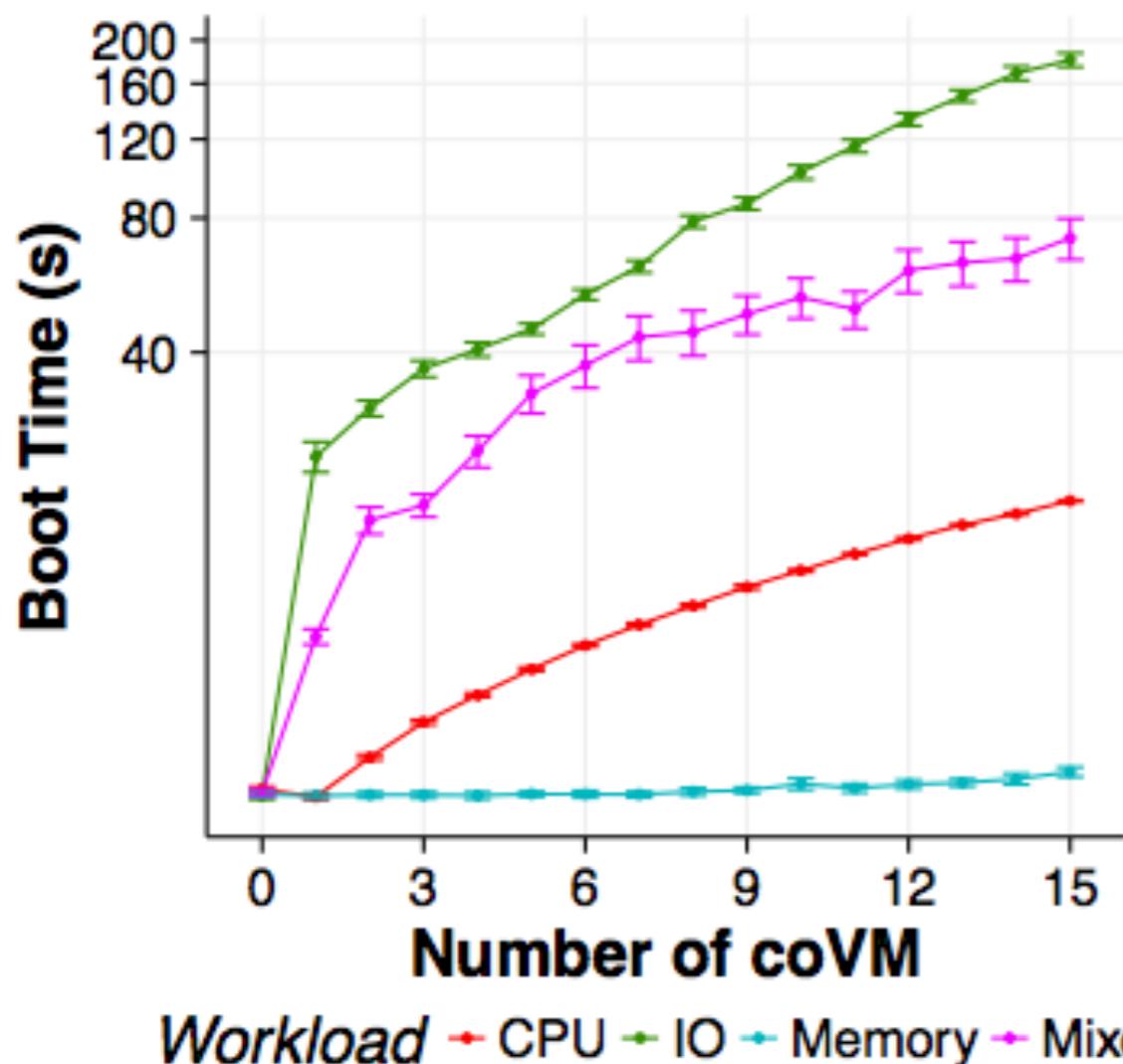


...especially if you are provision DB/Storage tiers.

# Autoscaling Mechanisms (few words)

Provisioning take time...

- [NGuyen17]



...especially if you are provisioning DB/Storage tiers.

*VMs make the control of performance  
harder but clouds are so good....*

*Anything else ?*

# Efficient data management

- IP over Avian Carriers



Request for commons 1149,  
Optimisation described in 2549 and 6214  
(packet loss ratio, latency, ...)



# Efficient data management

- IP over Avian Carriers



Request for commons 1149,  
Optimisation described in 2549 and 6214  
(packet loss ratio, latency, ...)



- But FedEx is still the most efficient way to share data

*"sneakernet: transfer of electronic information, especially computer files, by physically moving removable media... from one computer to another, usually in lieu of transmitting the information over a computer network"*

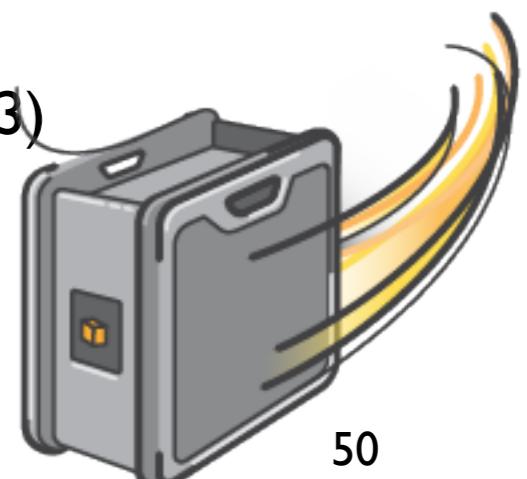
Google has used a sneakernet to transport large datasets, such as the 120 TB of data from of data from the Hubble Space Telescope.

Users of Google Cloud can import their data into Google Cloud Storage through sneakernet

Amazon introduced in 2015 the snowball

(Up to 50TBytes from your company to an AWS infrastructure and to S3)

<https://aws.amazon.com/importexport/>



*Ok but is there  
something more critical. . .*

# The Current Trend: Large off shore DCs

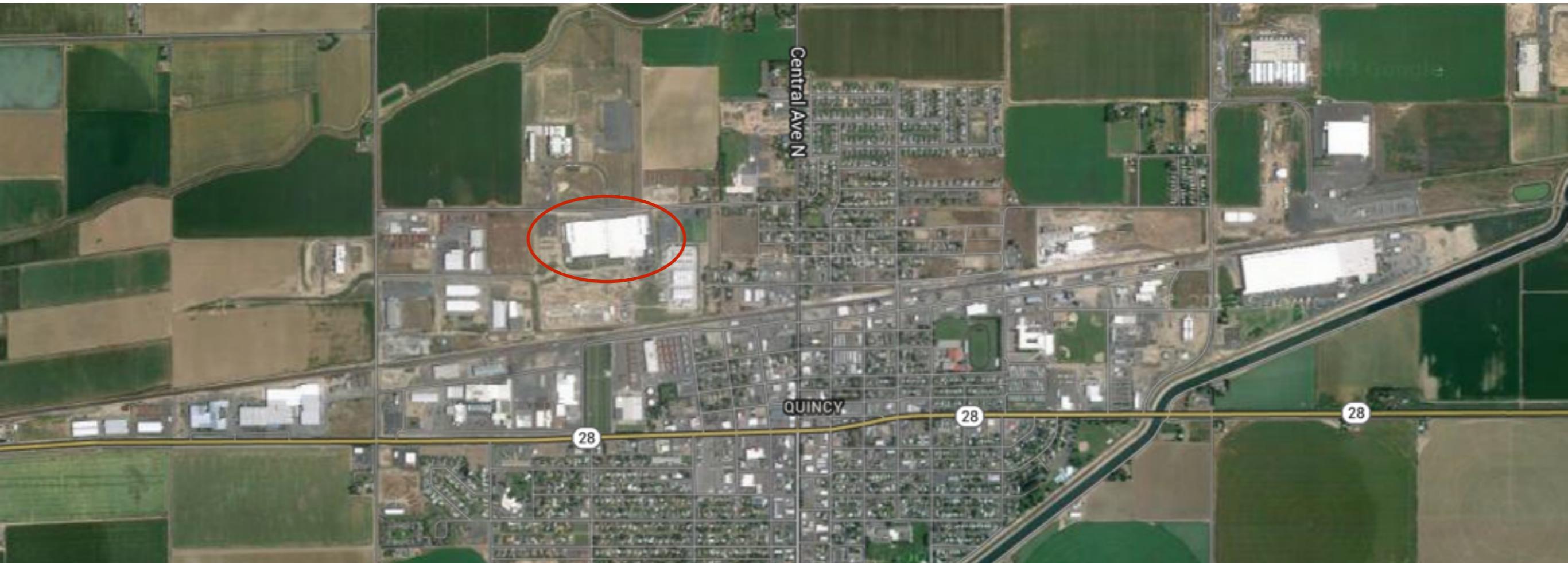
- To cope with the increasing UC demand while handling energy concerns but...



credits: [datacentertalk.com](http://datacentertalk.com) - Microsoft DC, Quincy, WA state

# The Current Trend: Large off shore DCs

- To cope with the increasing UC demand while handling energy concerns but...



credits: google map - Quincy

# The Current Trend: Large off shore DCs

- To cope with the increasing UC demand while handling energy concerns but...



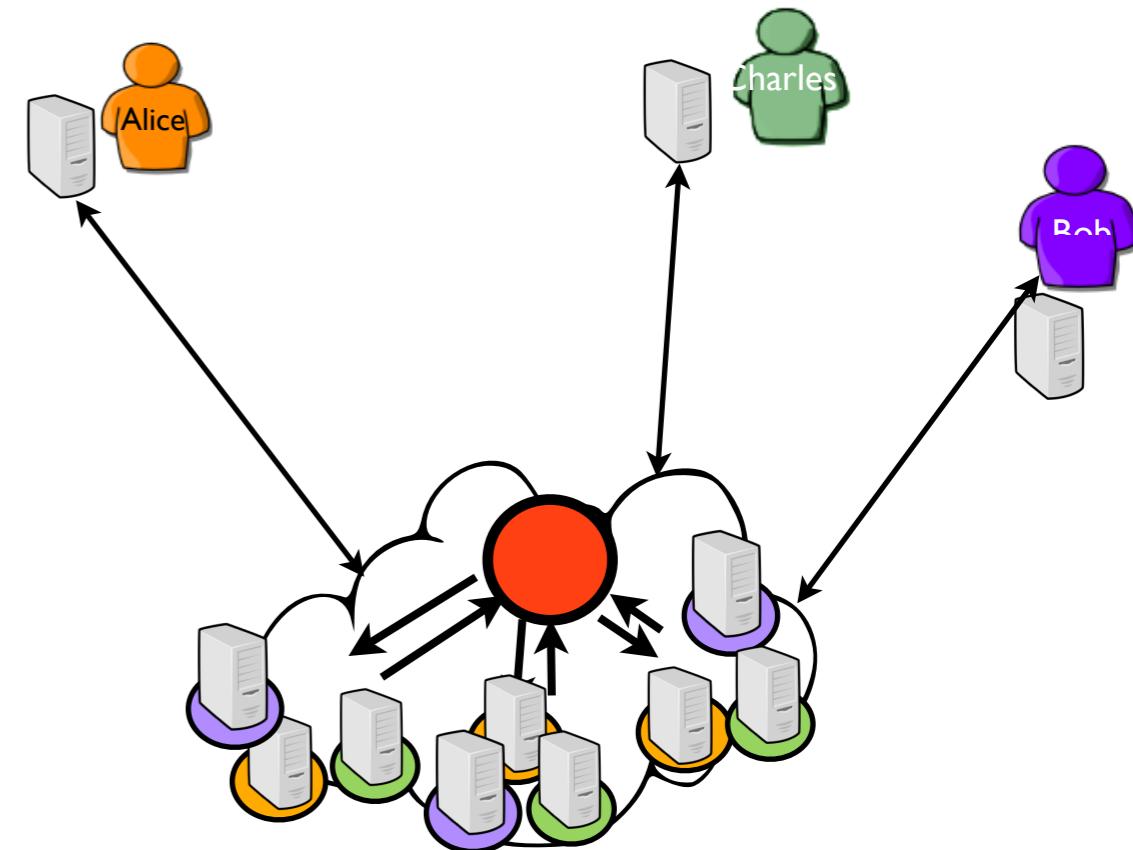
# The Current Trend: Large off shore DCs



credits: [coloandcloud.com](http://coloandcloud.com)

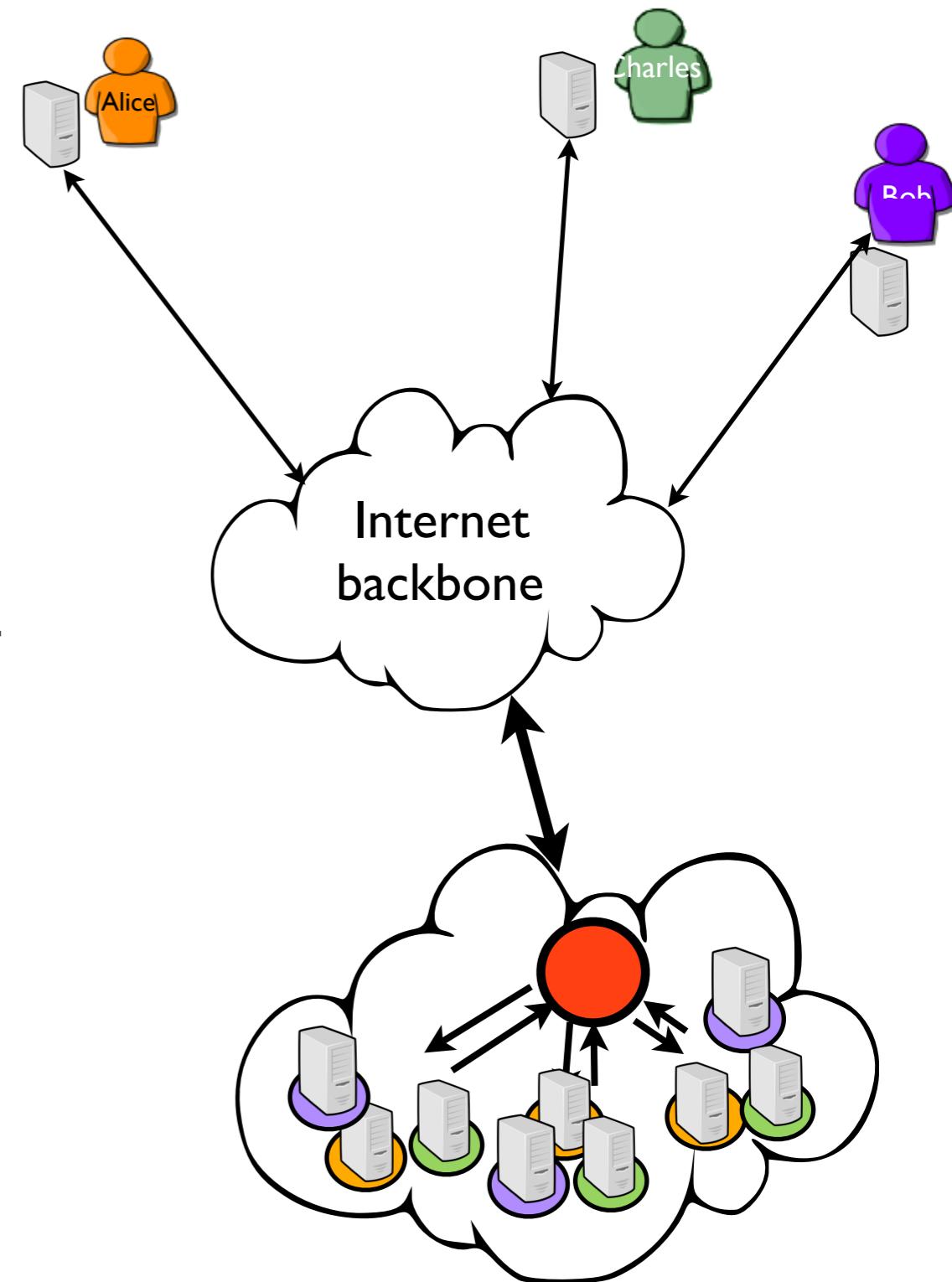
# Inherent limitations of current solutions

- Large off shore DCs to cope with the increasing UC demand while handling energy concerns but...
  - I. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)



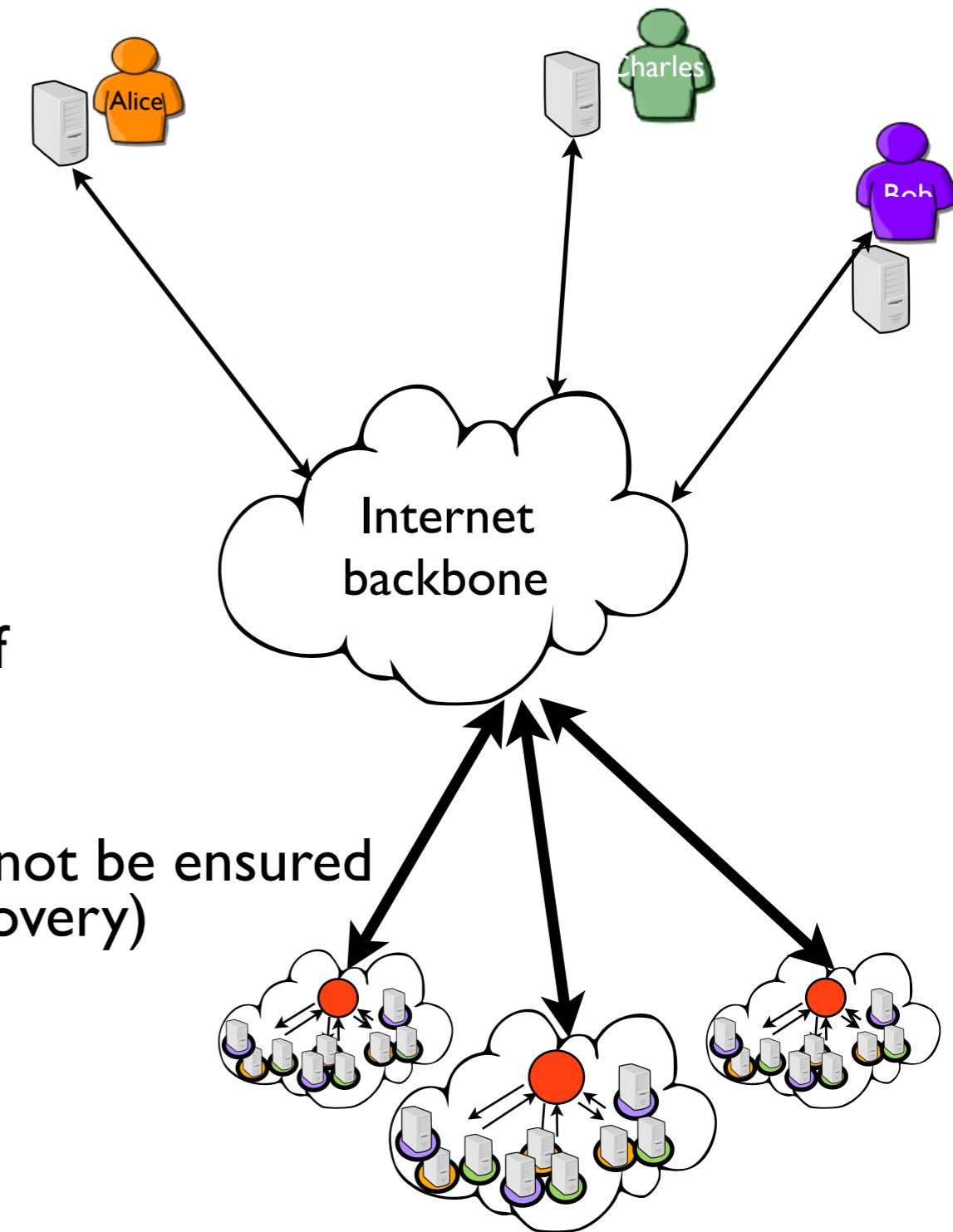
# Inherent limitations of current solutions

- Large off shore DCs to cope with the increasing UC demand while handling energy concerns but...
  - I. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)
  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms



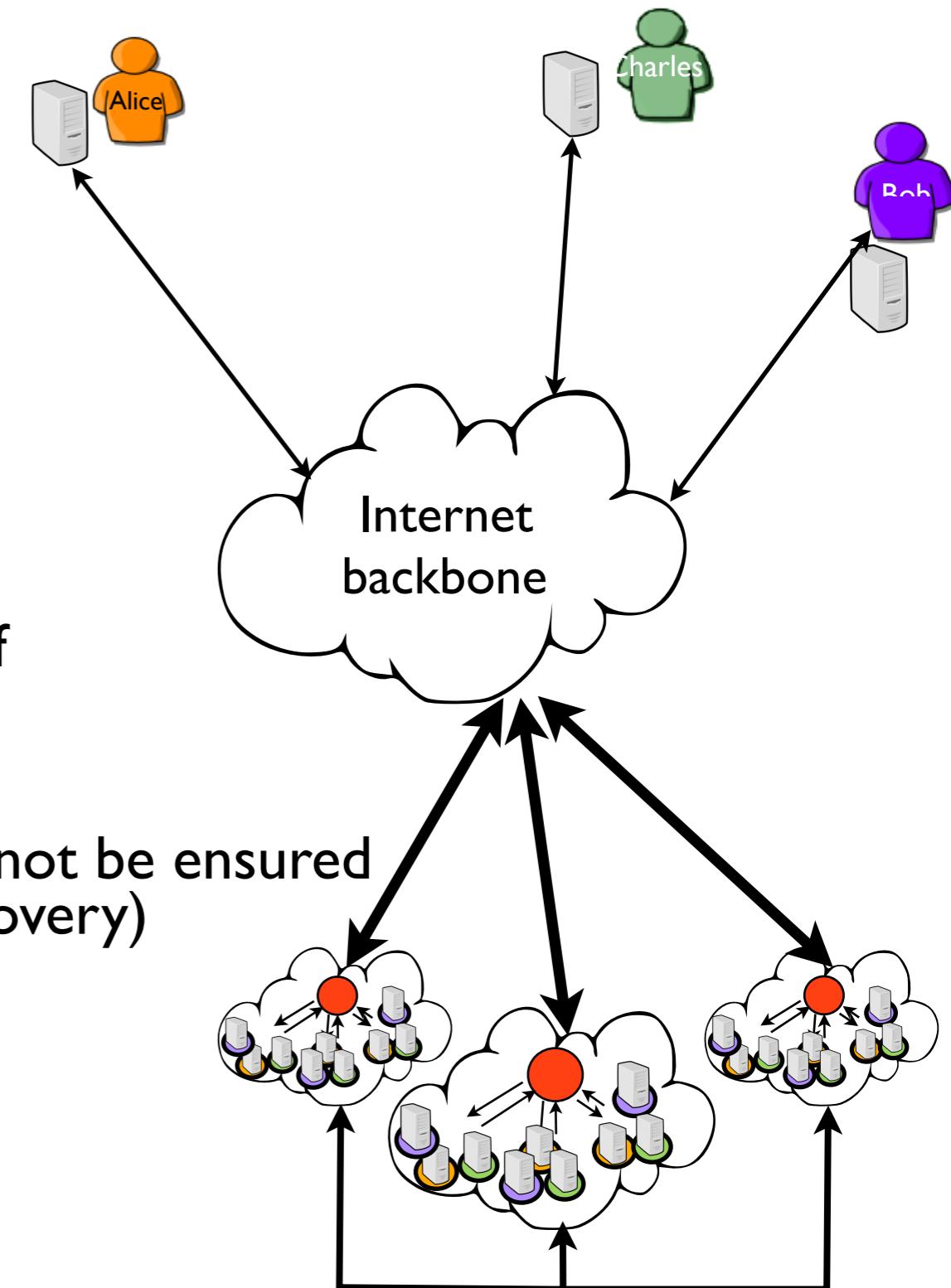
# Inherent limitations of current solutions

- Large off shore DCs to cope with the increasing UC demand while handling energy concerns but...
  1. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)
  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms
  3. The connectivity to the application/data cannot be ensured by centralized dedicated centers (disaster recovery)



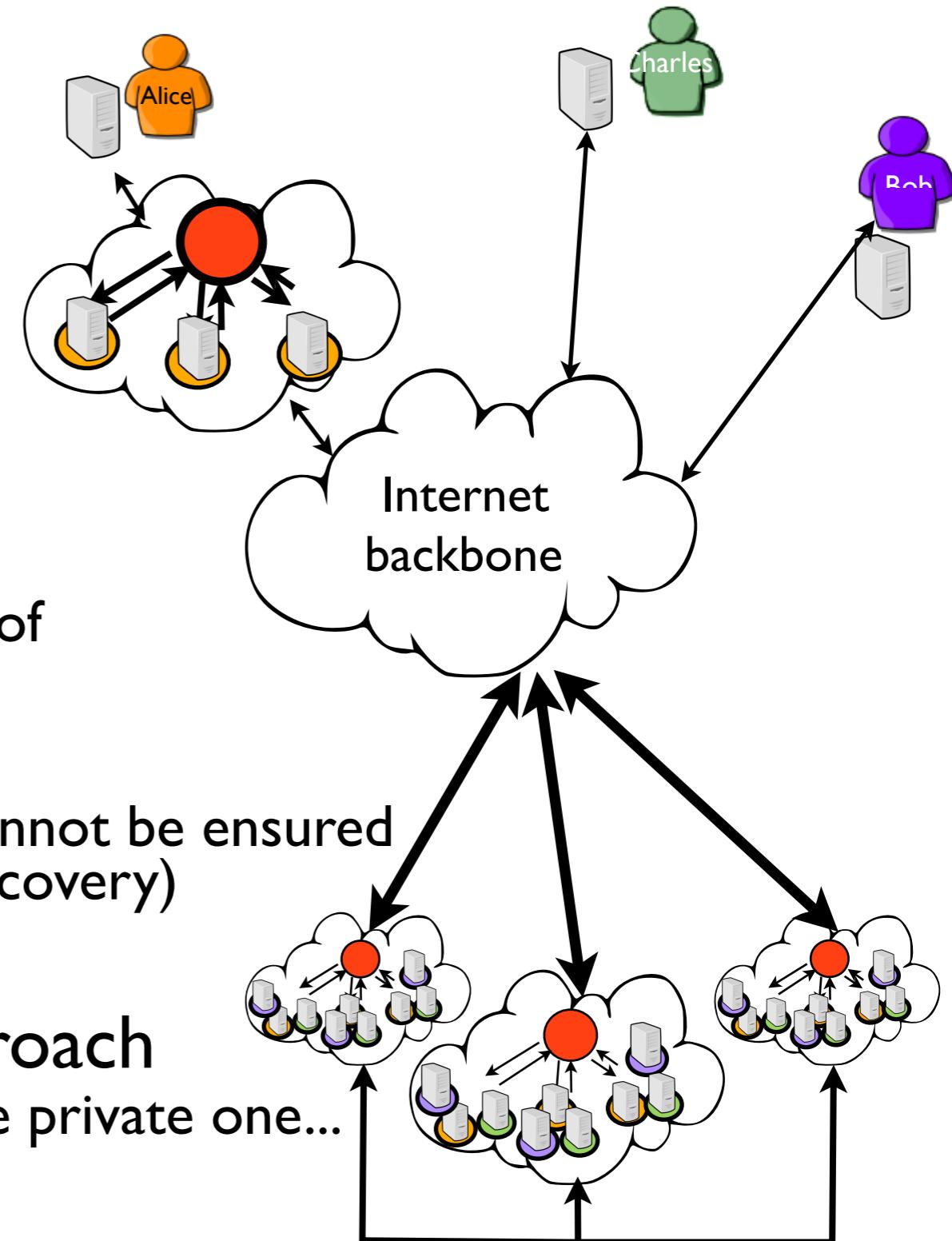
# Inherent limitations of current solutions

- Large off shore DCs to cope with the increasing UC demand while handling energy concerns but...
  1. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)
  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms
  3. The connectivity to the application/data cannot be ensured by centralized dedicated centers (disaster recovery)



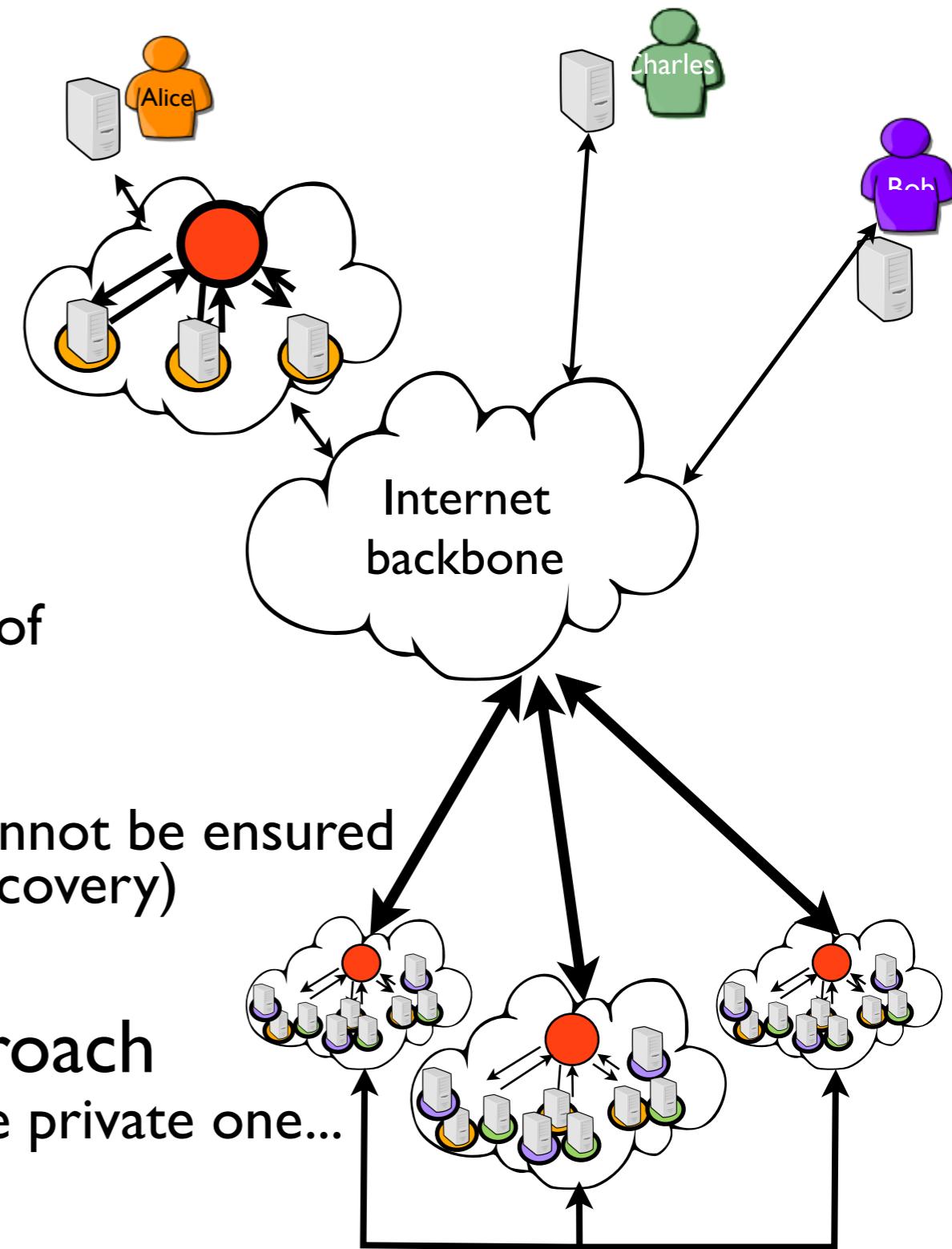
# Inherent limitations of current solutions

- **Large off shore DCs** to cope with the increasing UC demand while handling energy concerns but...
  1. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)
  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms
  3. The connectivity to the application/data cannot be ensured by centralized dedicated centers (disaster recovery)
- **Hybrid platforms: a promising approach**  
It depends how you are going to extend the private one...



# Inherent limitations of current solutions

- Large off shore DCs to cope with the increasing UC demand while handling energy concerns but...
  1. Externalization of private applications/data (jurisdiction concerns, PRISM NSA scandal, Patriot Act)
  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms
  3. The connectivity to the application/data cannot be ensured by centralized dedicated centers (disaster recovery)
- Hybrid platforms: a promising approach  
It depends how you are going to extend the private one...



Is there a way to address these concerns “all in one” ?

Micro/Nano DCs

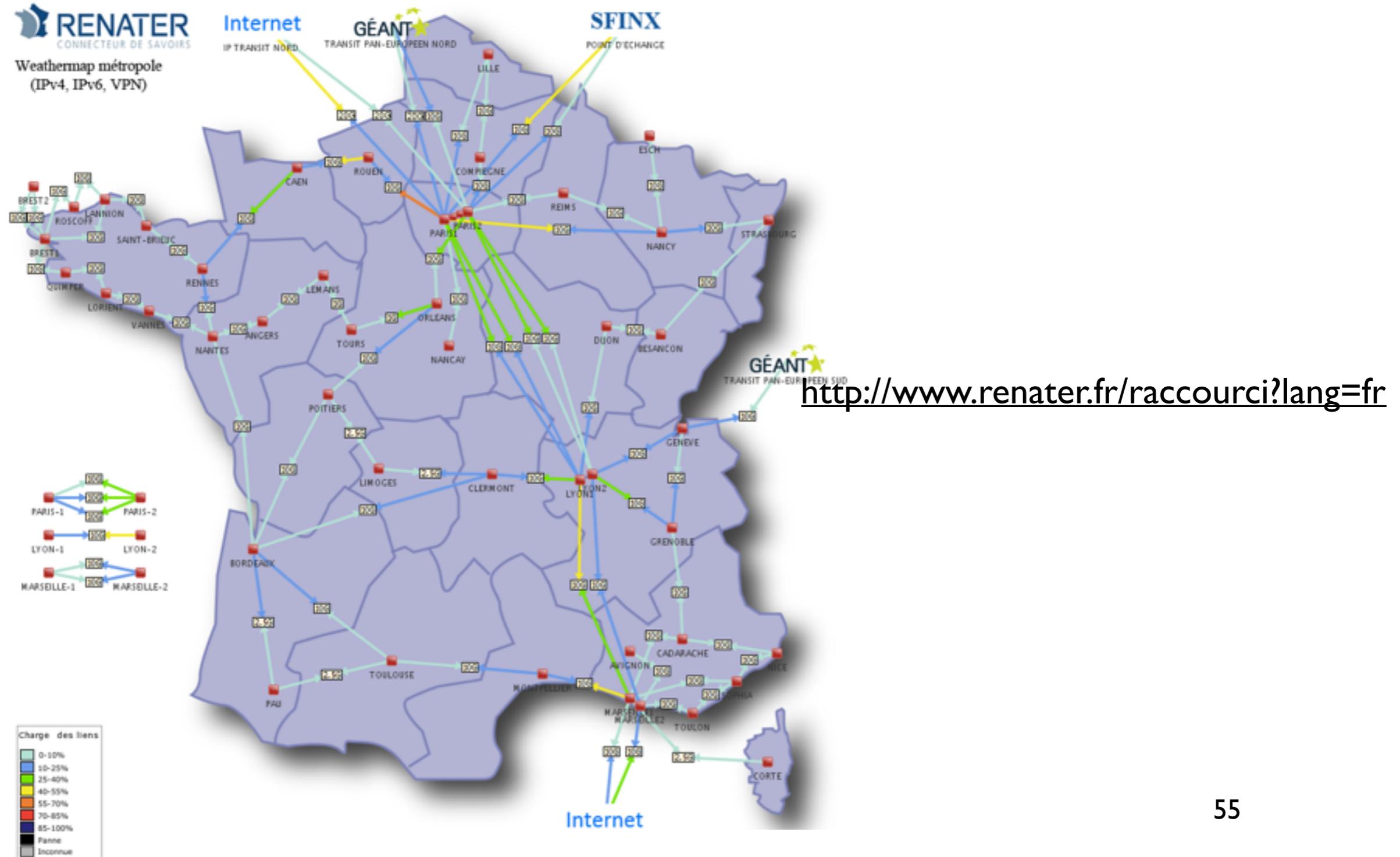
*$\mu$ DC at the edge !*

*How and where the  $\mu$ DC concept can be deployed ?*

# Beyond the Clouds...

- Locality-based UC infrastructures (aka. Fog/Edge)

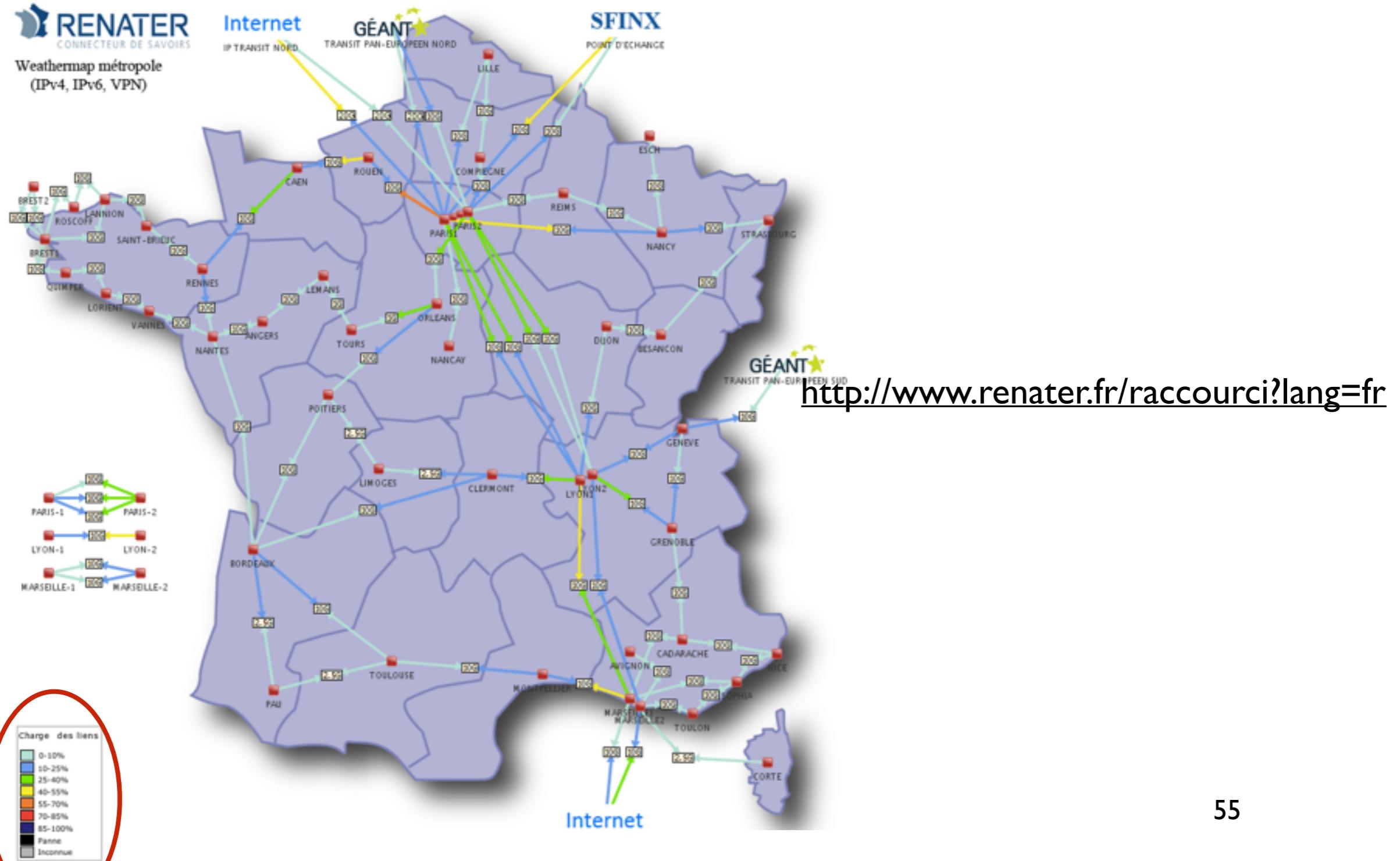
A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.



# Beyond the Clouds...

- Locality-based UC infrastructures (aka. Fog/Edge)

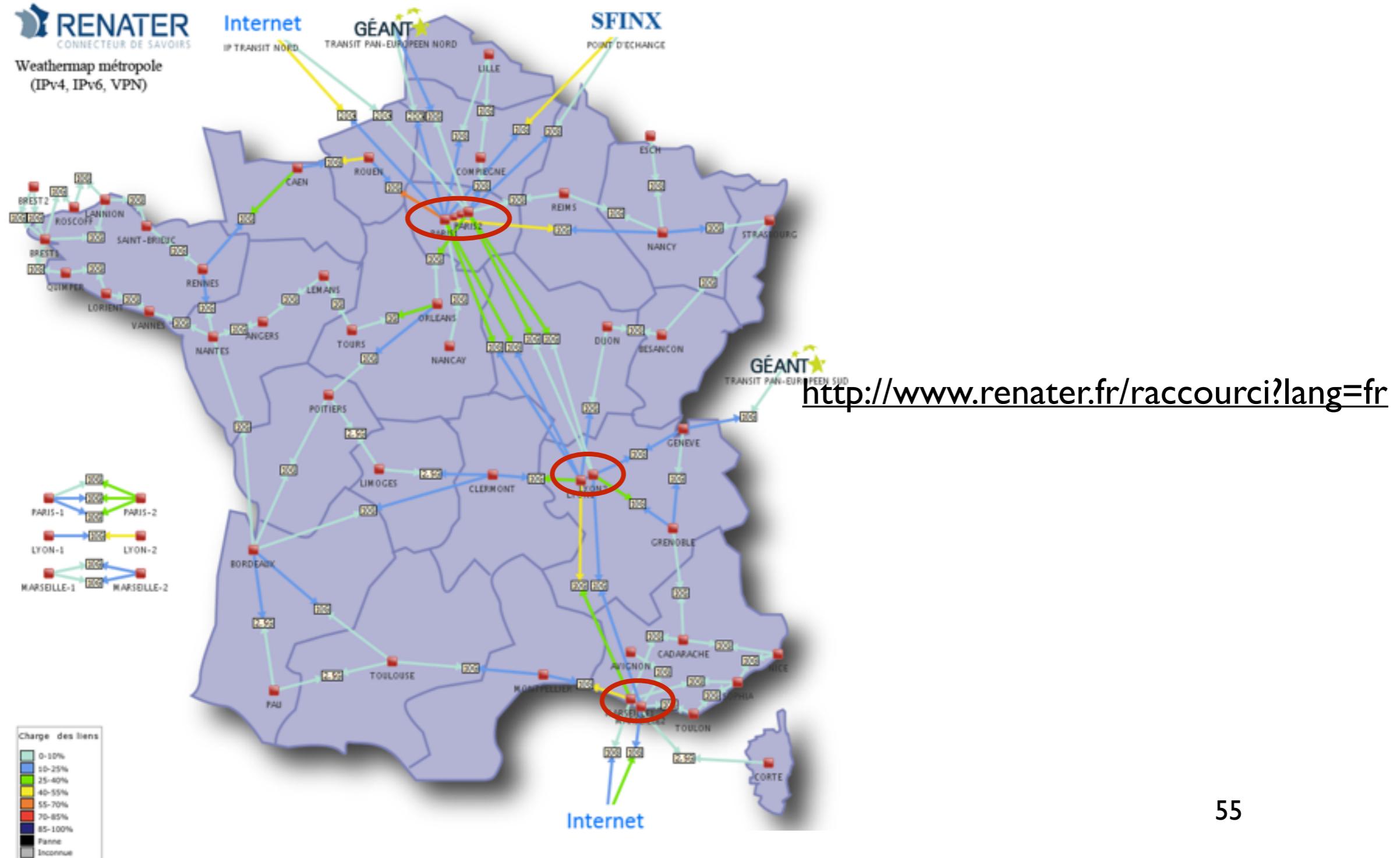
A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.



# Beyond the Clouds...

- Locality-based UC infrastructures (aka. Fog/Edge)

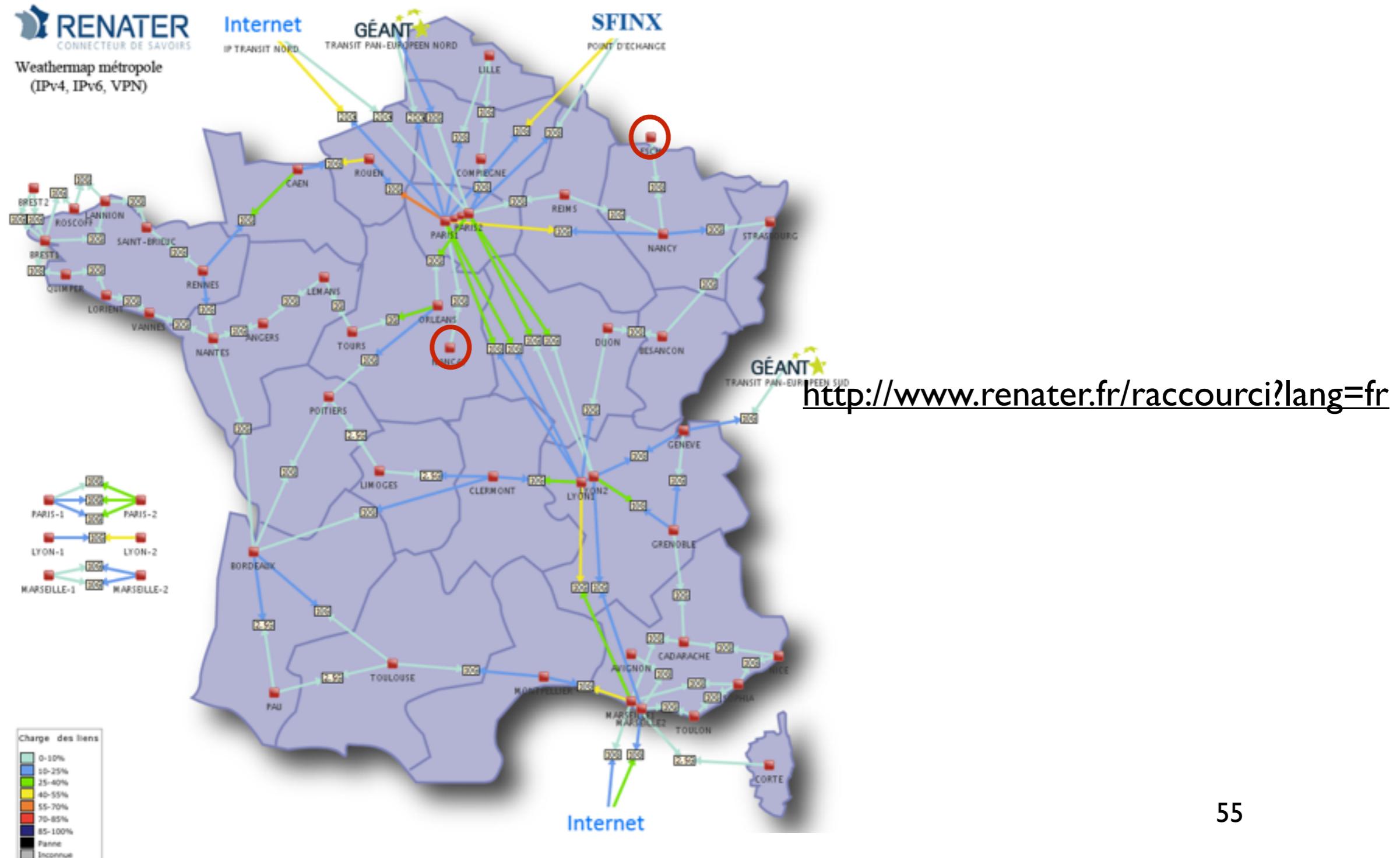
A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.



# Beyond the Clouds...

- Locality-based UC infrastructures (aka. Fog/Edge)

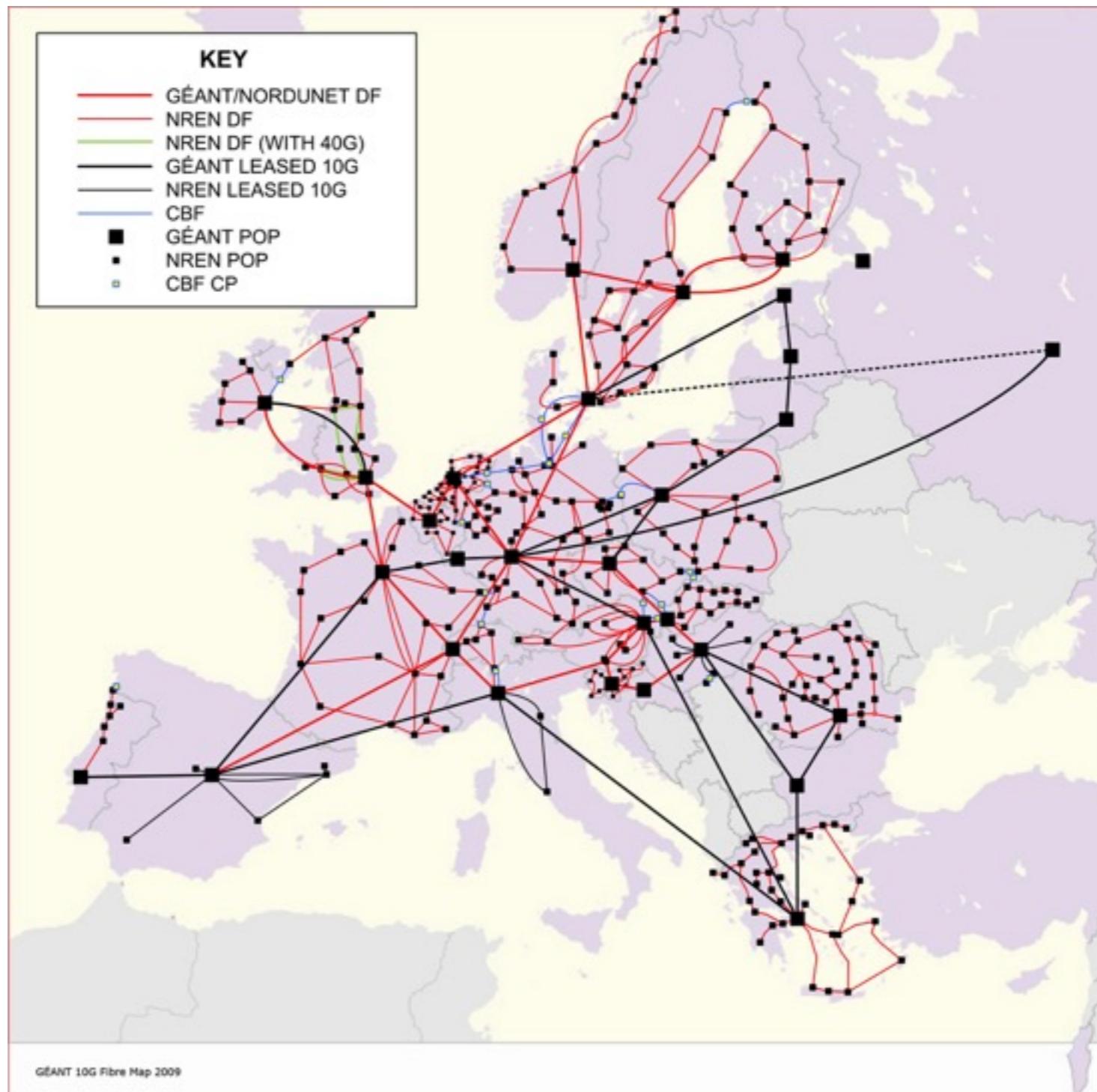
A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.



# Beyond the Clouds...

- Locality-based UC infrastructures (aka. Fog/Edge)

A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.



# ....The Fog/Edge Computing

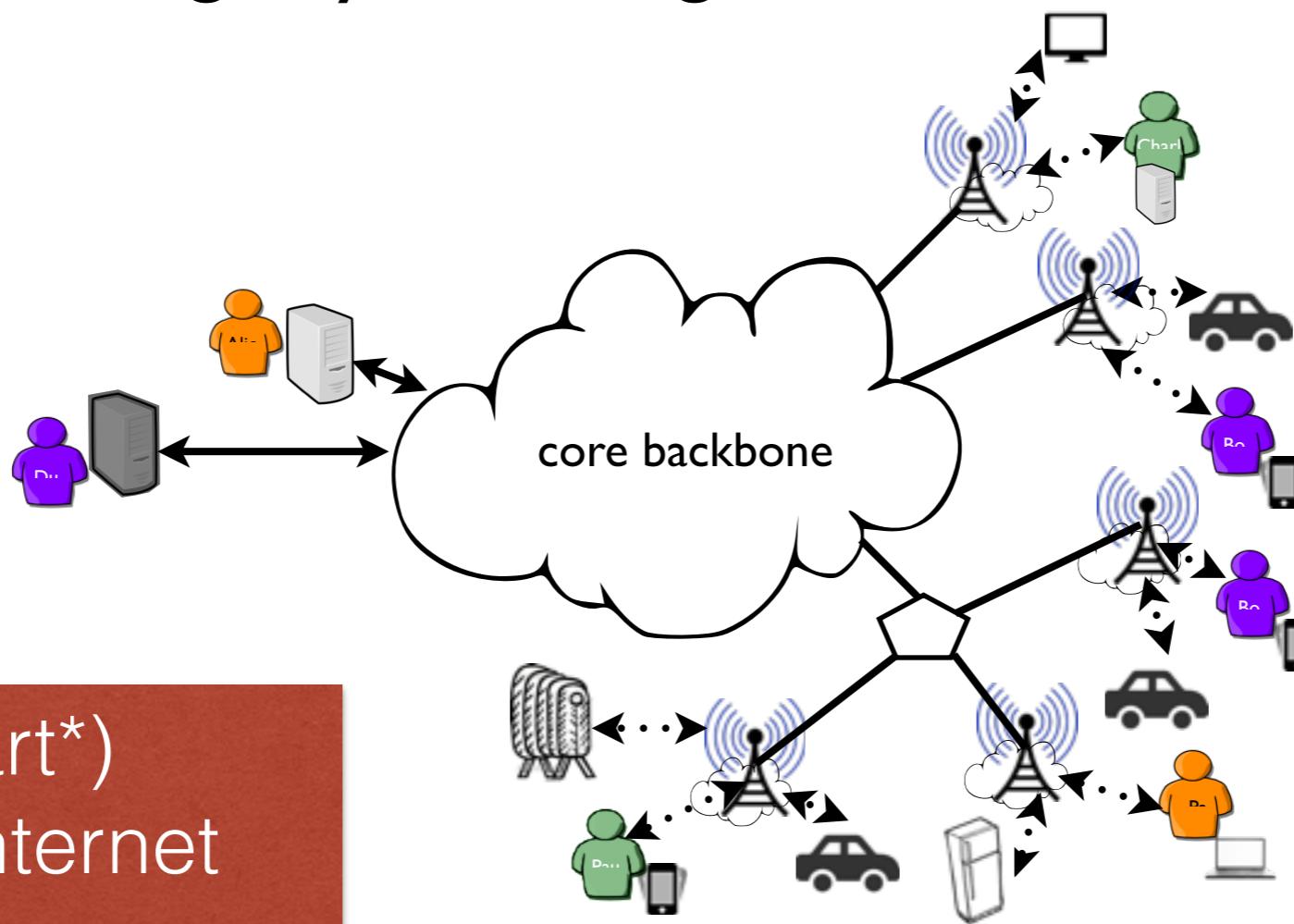
- Leverage network backbones

Extend any point of presence of network backbones (aka PoP) with servers (from network hubs up to major DSLAMs that are operated by telecom companies, network institutions...).

- Extend to the edge by including wireless backbones



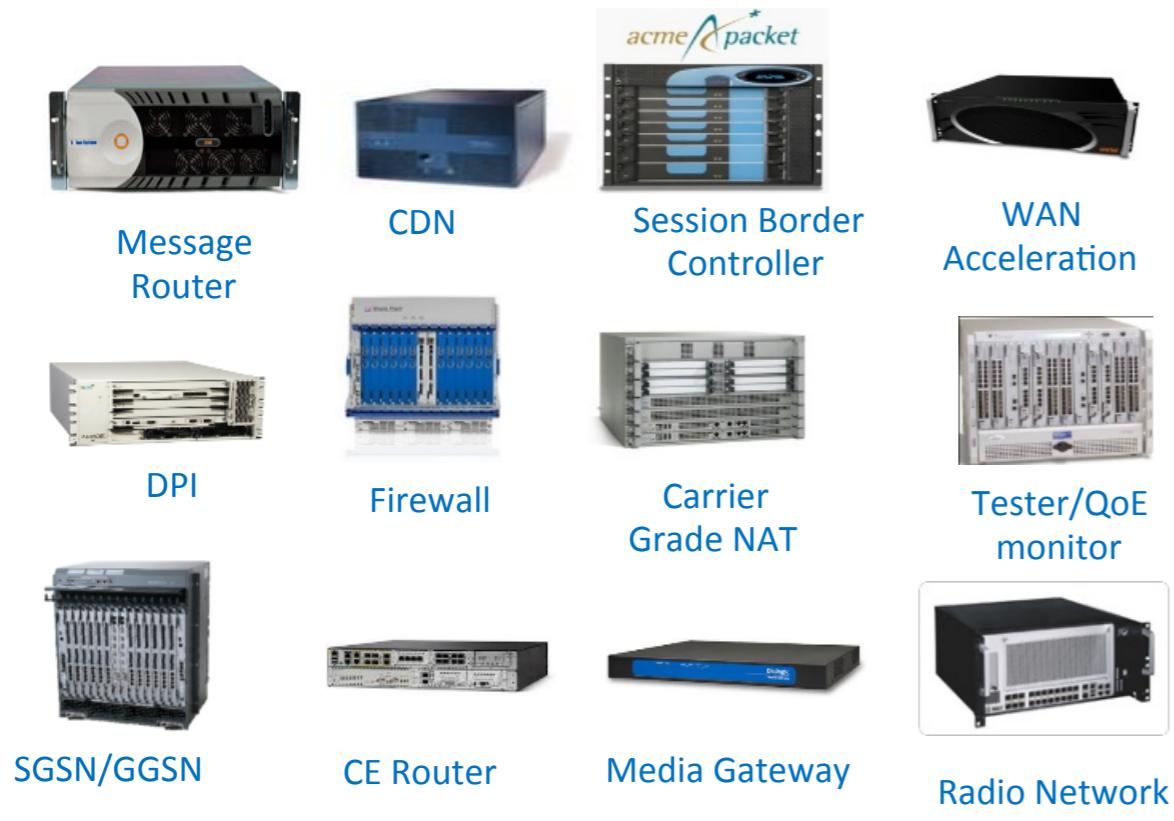
USA NREN



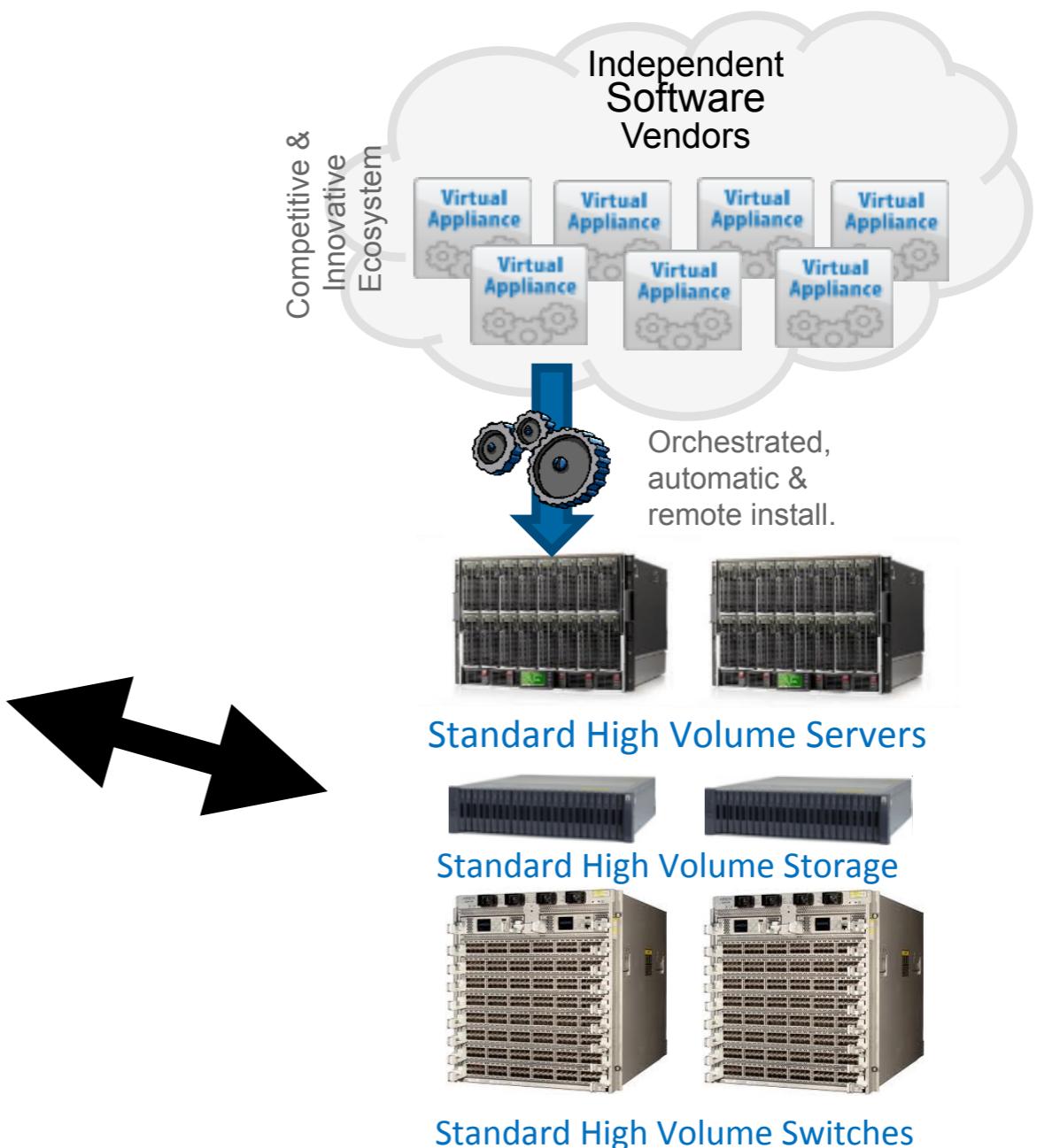
# IoT (smart\*) Industrial Internet NFV

# Virtual Customer Premises Equipment

## Classical Network Appliance Approach



Fragmented non-commodity hardware.  
Physical install per appliance per site.  
Hardware development large barrier to entry for  
new vendors constraining innovation &  
competition.



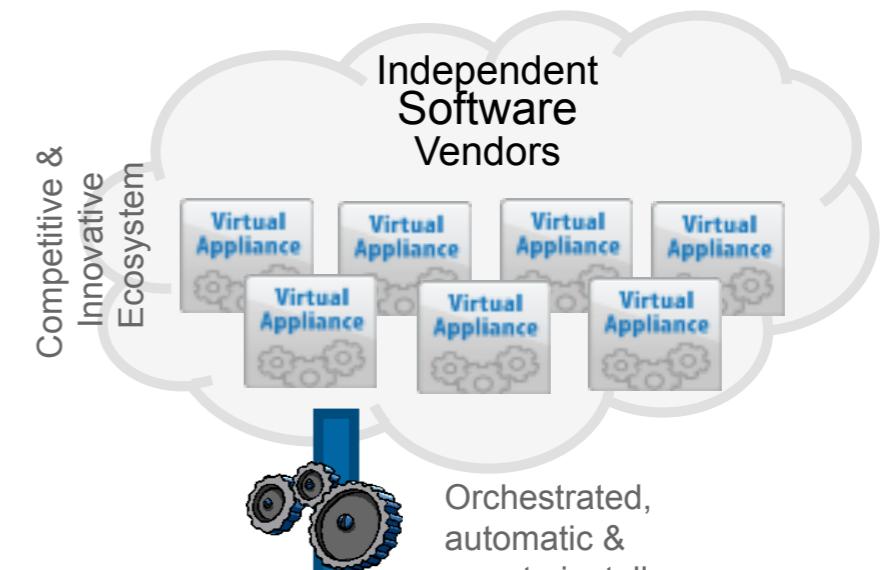
## Network *functions* Virtualisation Approach

# Virtual Customer Premises Equipment

## Classical Network Appliance Approach



Fragmented non-commodity hardware.  
Physical install per appliance per site.  
Hardware development large barrier to entry for  
new vendors constraining innovation &  
competition.



Standard High Volume Servers



## Network functions Virtualisation Approach

### Lesson 7: V. Conan

# Micro/Nano DCs



Sagrada Familia microDC  
(Barcelona, Spain)

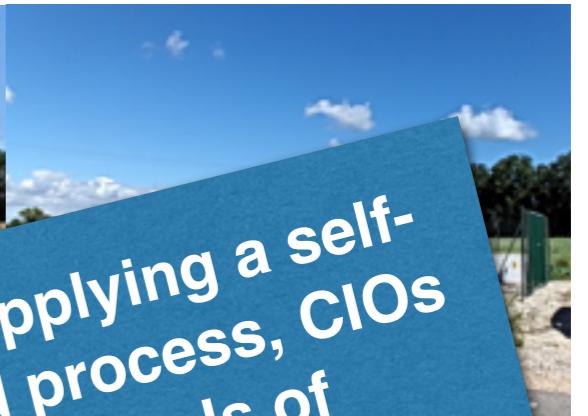


Deployment of a PoP of the Orange French backbone



MDC Industry - Brazil

# Micro/Nano DCs



Localized or **micro data centers** are a fact of life, but by **applying a self-contained, scalable and remotely managed solution and process, CIOs can reduce costs, improve agility, and introduce new levels of compliance and service continuity.**

Gartner 2015



Sagrada Familia microDC  
(Barcelona, Spain)

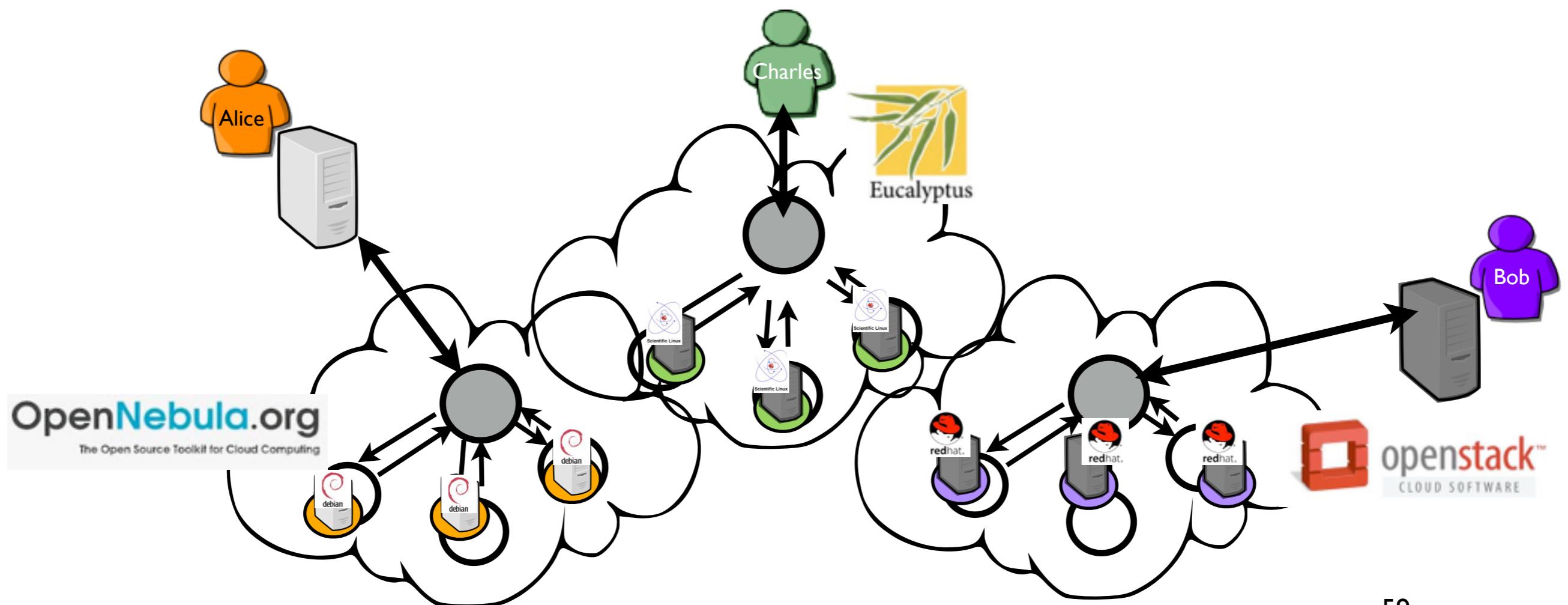


MDC Industry - Brazil

# A broker ?

- “federation of clouds” (sky computing,)

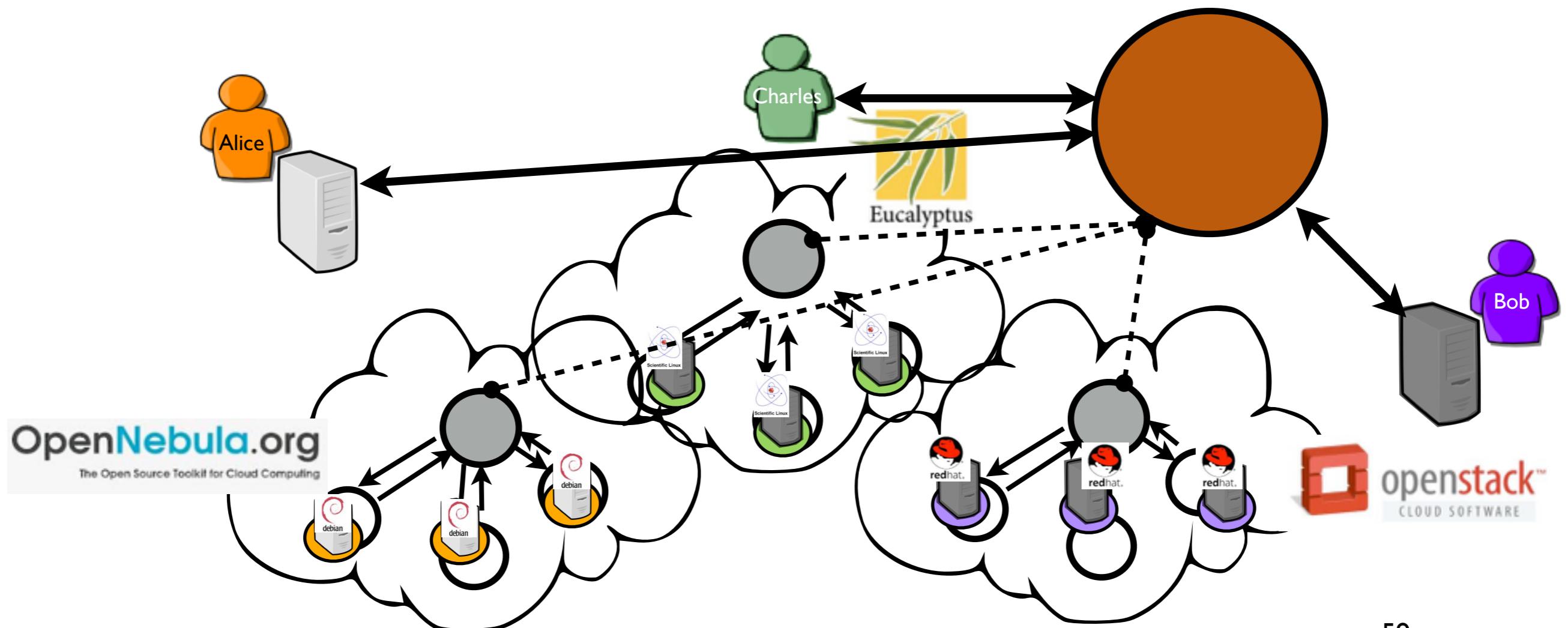
Sporadic (hybrid computing/cloud bursting) almost ready for production  
While standards are coming (OCCI, OVF, ....), current brokers are rather limited



# A broker ?

- “federation of clouds” (sky computing),

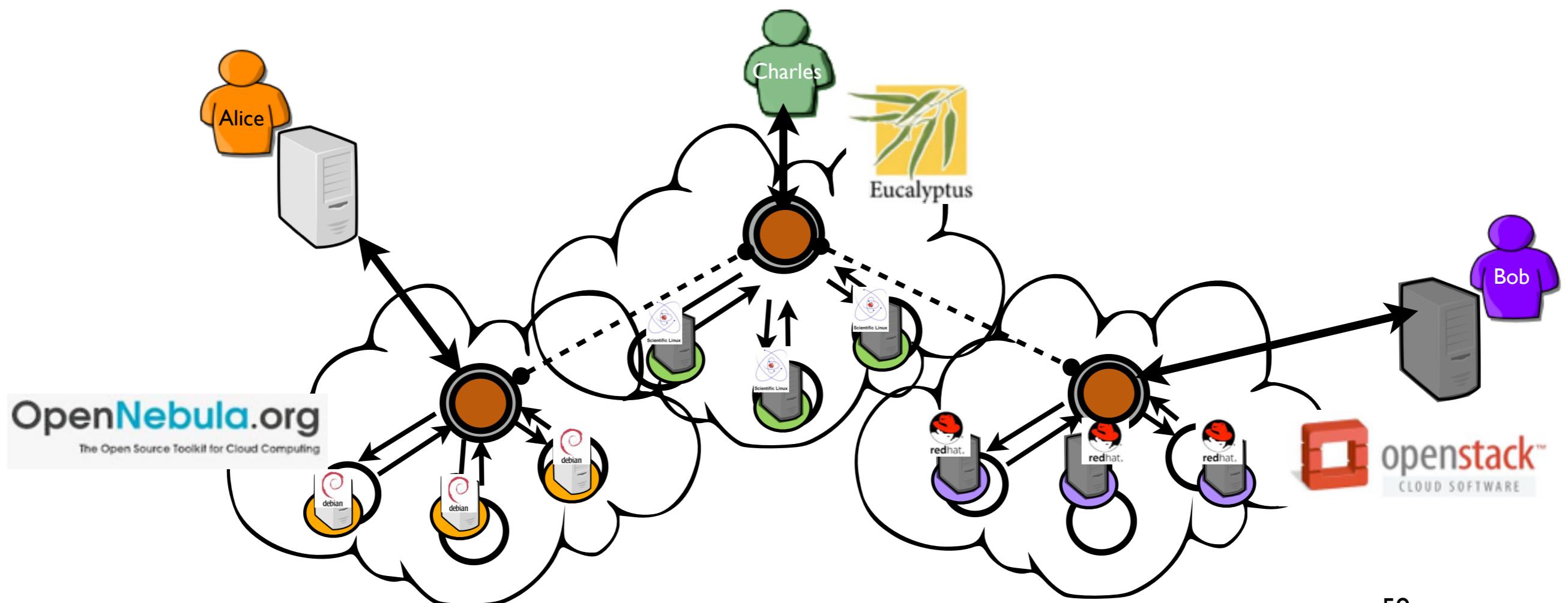
Sporadic (hybrid computing/cloud bursting) almost ready for production  
While standards are coming (OCCI, OVF, ....), current brokers are rather limited



# A broker ?

- “federation of clouds” (sky computing,)

Sporadic (hybrid computing/cloud bursting) almost ready for production  
While standards are coming (OCCI, OVF, ....), current brokers are rather limited

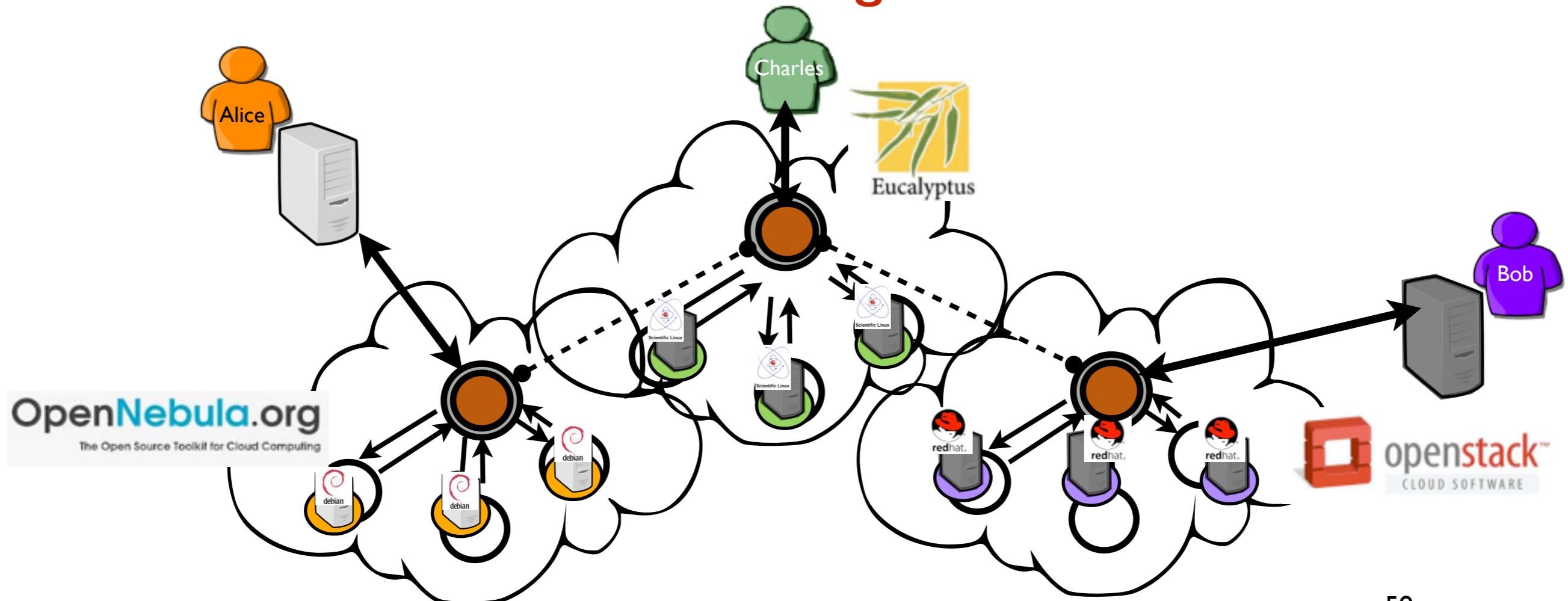


# A broker ?

- “federation of clouds” (sky computing,)

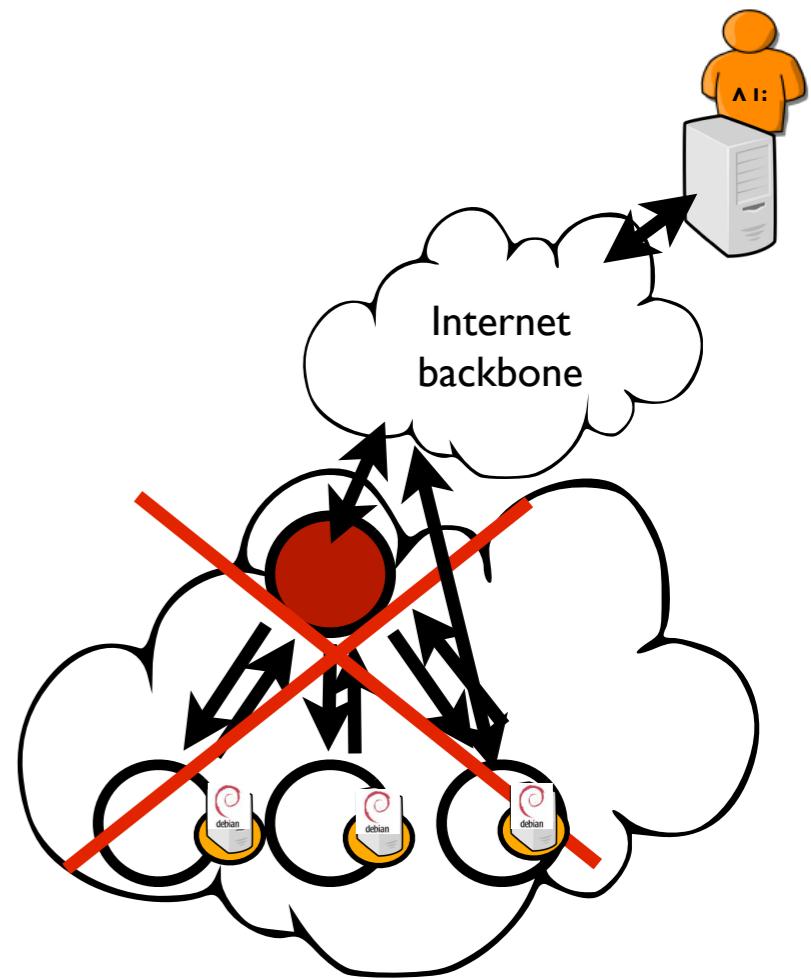
Sporadic (hybrid computing/cloud bursting) almost ready for production  
While standards are coming (OCCI, OVF, ....), current brokers are rather limited

**Advanced brokers must reimplement standard IaaS mechanisms while facing the API limitation**



# The DISCOVERY Proposal

- DIStributed and COoperative framework to manage Virtual EnviRonments autonomously



Do you Want more ! Visit  
<http://beyondtheclouds.github.io>

# The DISCOVERY Proposal

- DIStributed and COoperative framework to manage Virtual EnviRonments autonomously

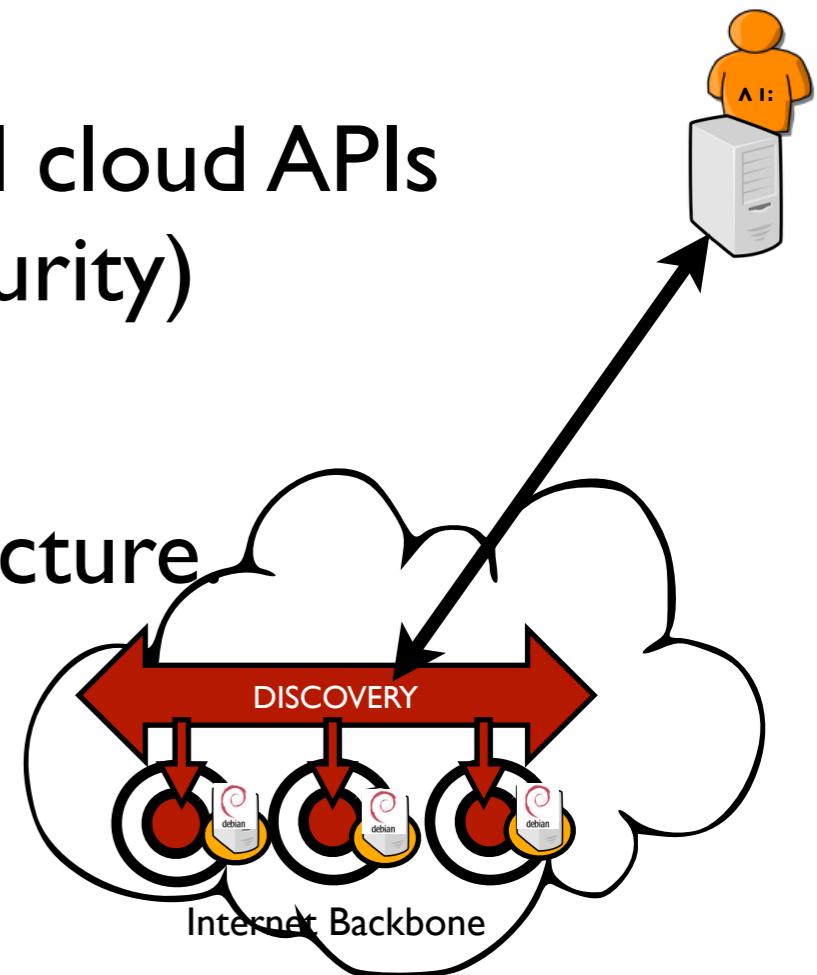


- A fully distributed IaaS system and not a distributed system of IaaS systems

We want to/must go further than high level cloud APIs  
(cross-cutting concerns such as energy/security)

- Leverage P2P algorithms and self-\* approaches to operate a LUC infrastructure

Do you Want more ! Visit  
<http://beyondtheclouds.github.io>



# Beyond IT !

- From sustainable data centers to a new source of energy

A promising way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users and to...



- Leverage “green” energy (solar, wind turbines...)

Transfer the green micro/nano DCs concept to the network PoP

Take the advantage of the geographical distribution



<http://parasol.cs.rutgers.edu>

- Leveraging the data furnaces concept

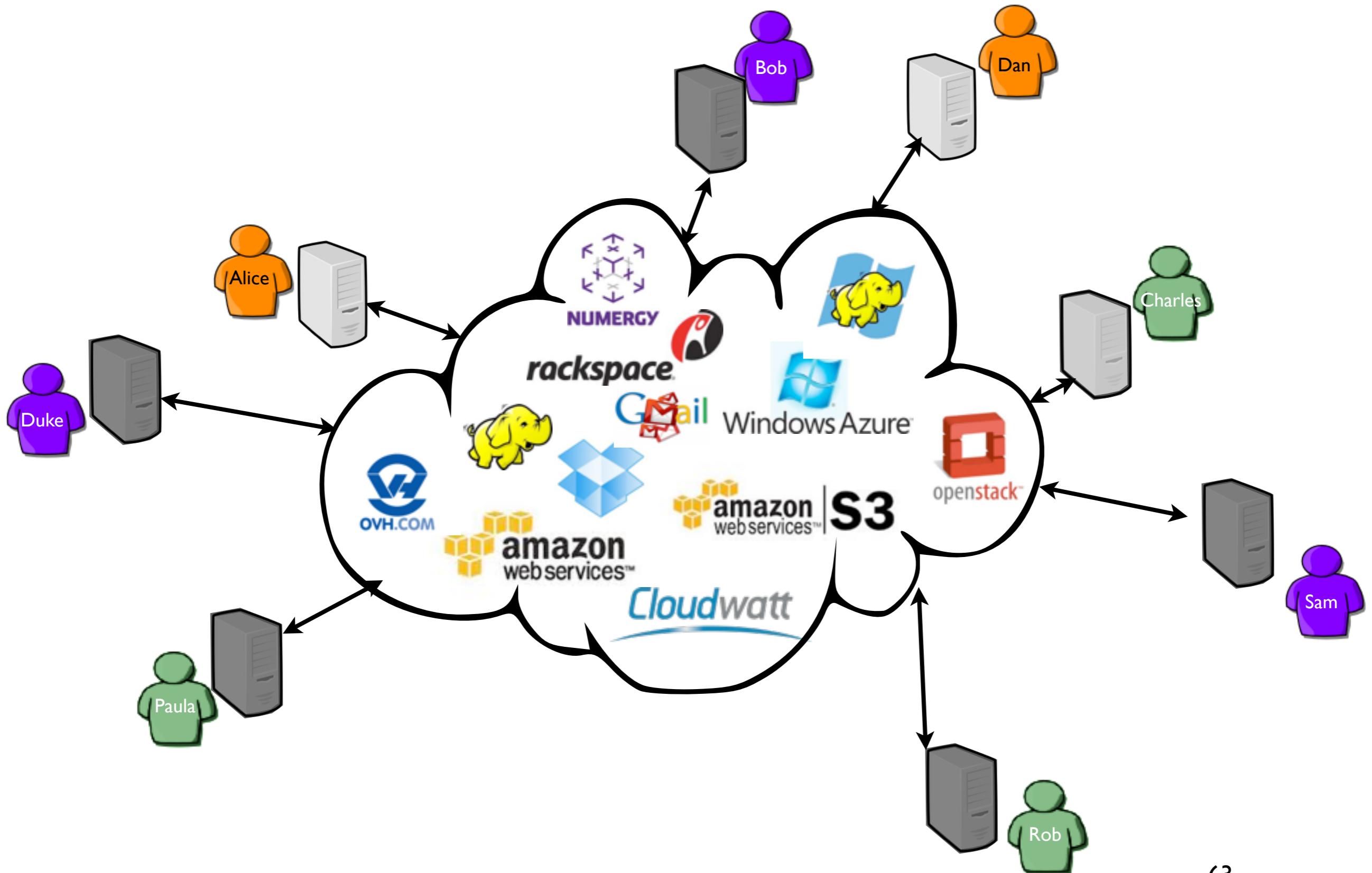
Deploy UC servers in medium and large institutions and use them as sources of heat inside public buildings such as hospitals or universities



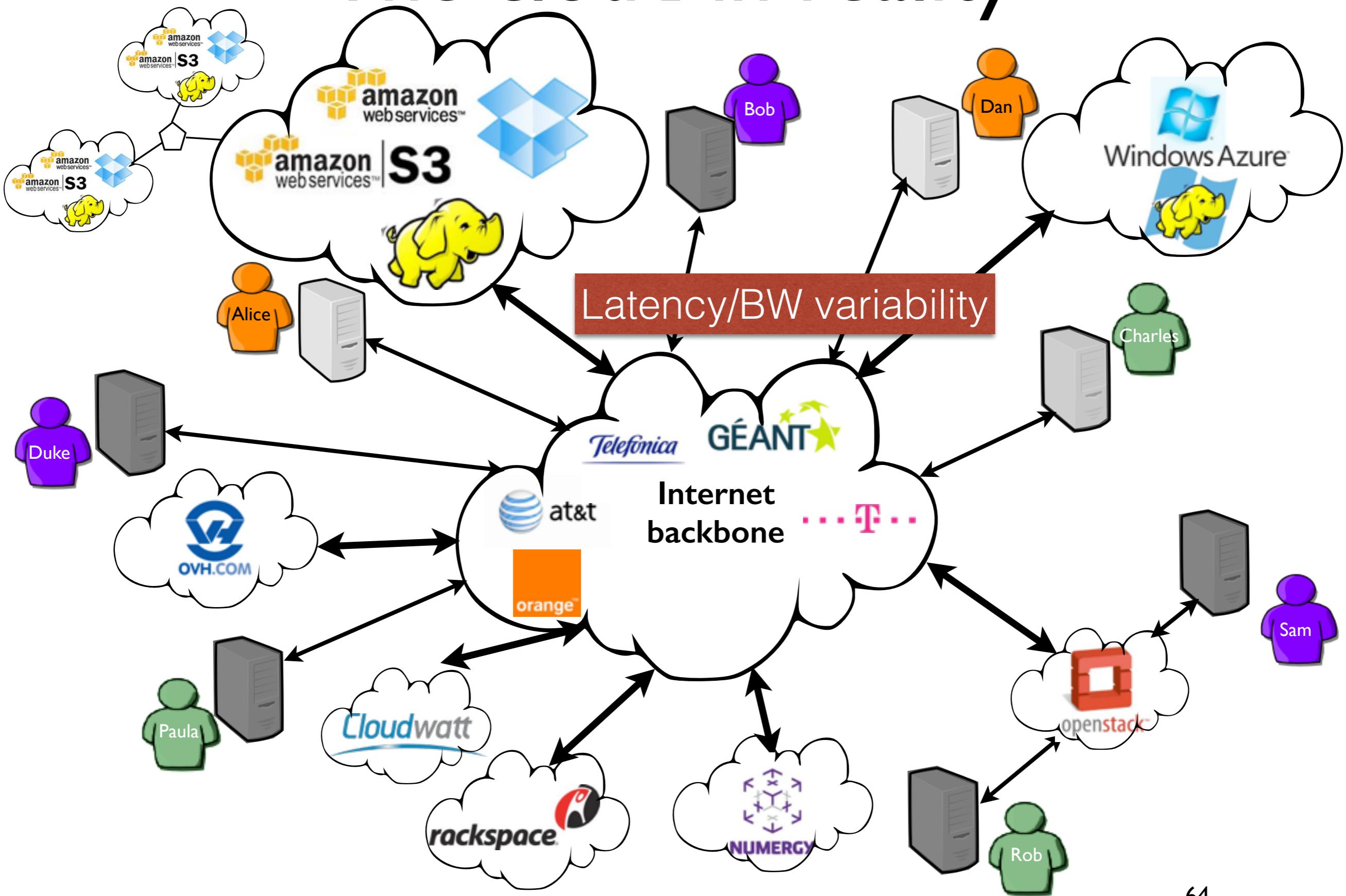
<https://www.aoterra.de>

## *Takeaway Message*

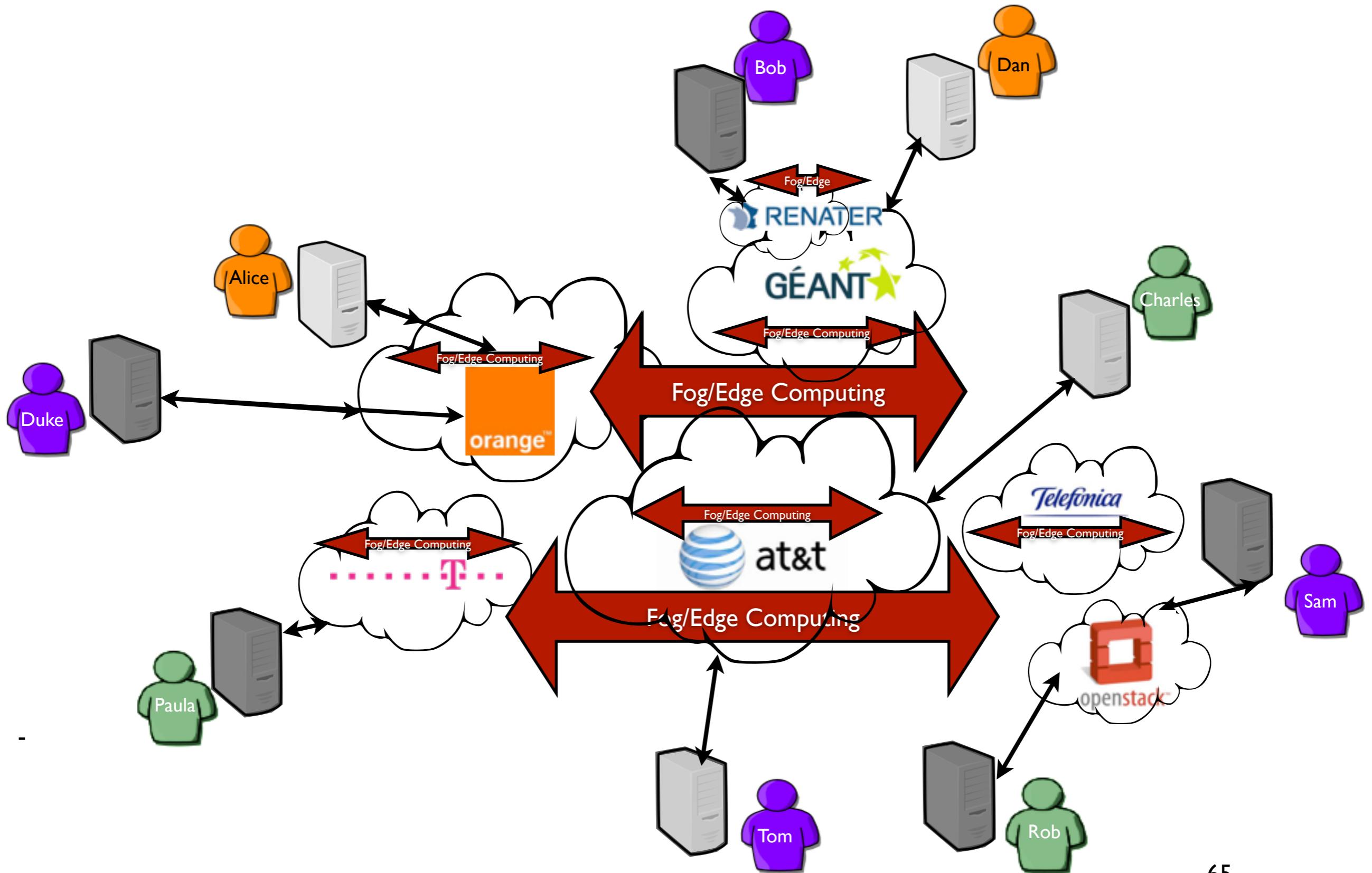
# The cloud from end-users



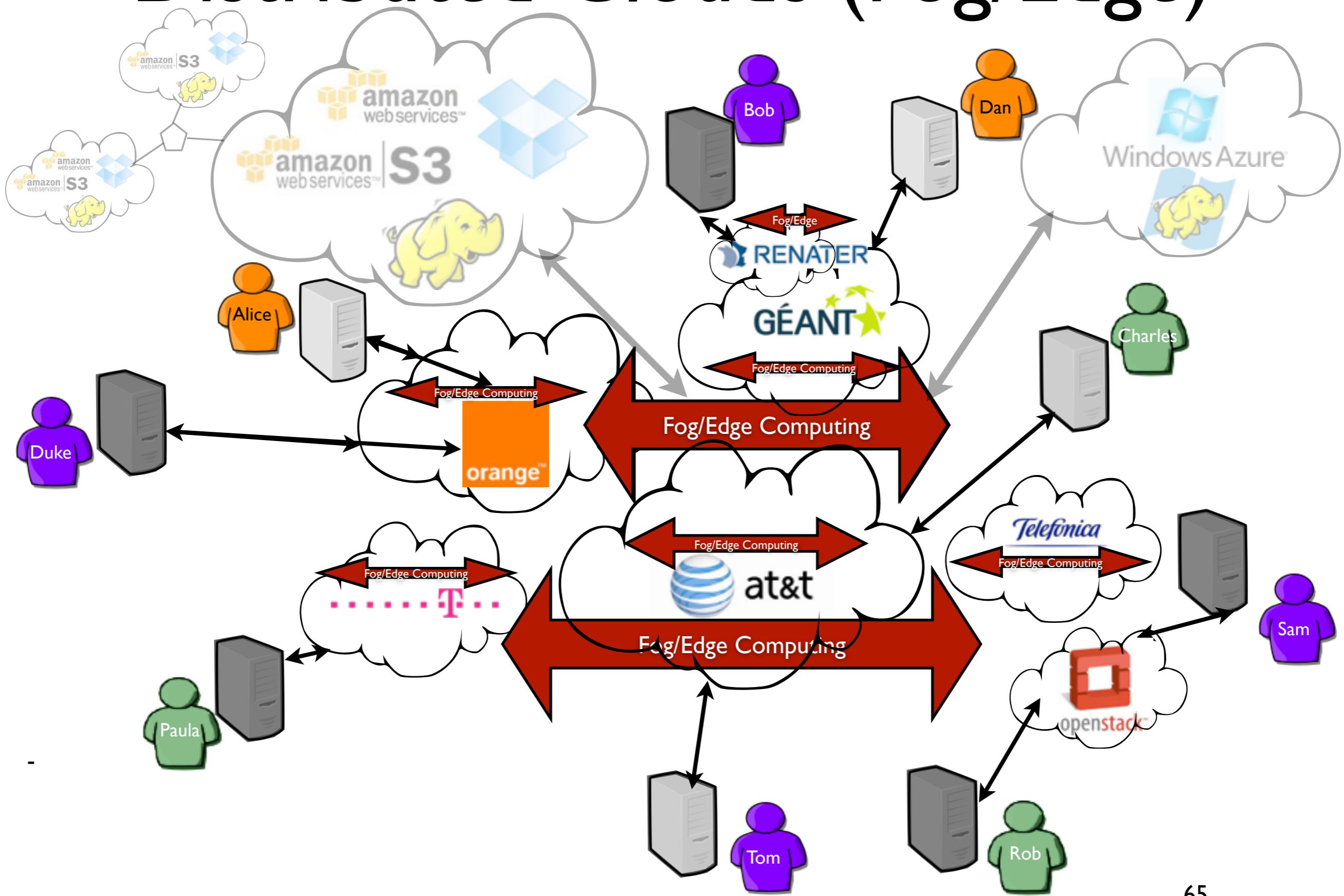
# The cloud in reality

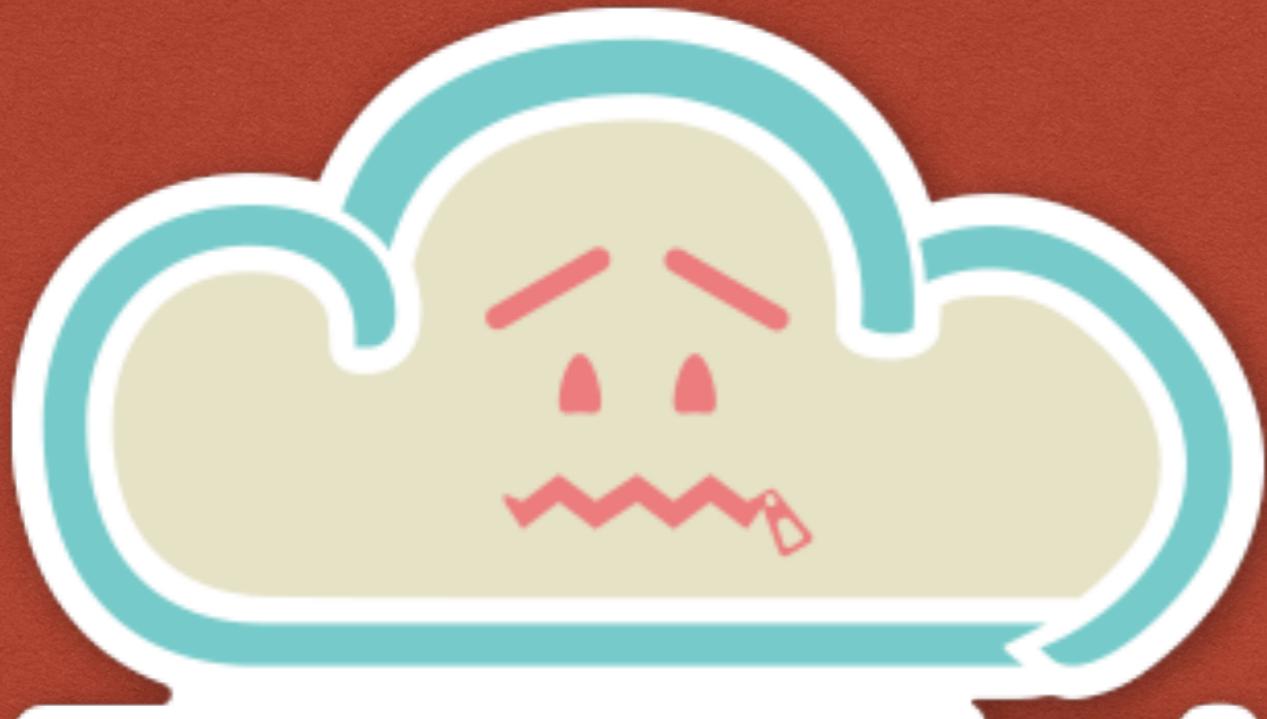


# Distributed Clouds (Fog/Edge)



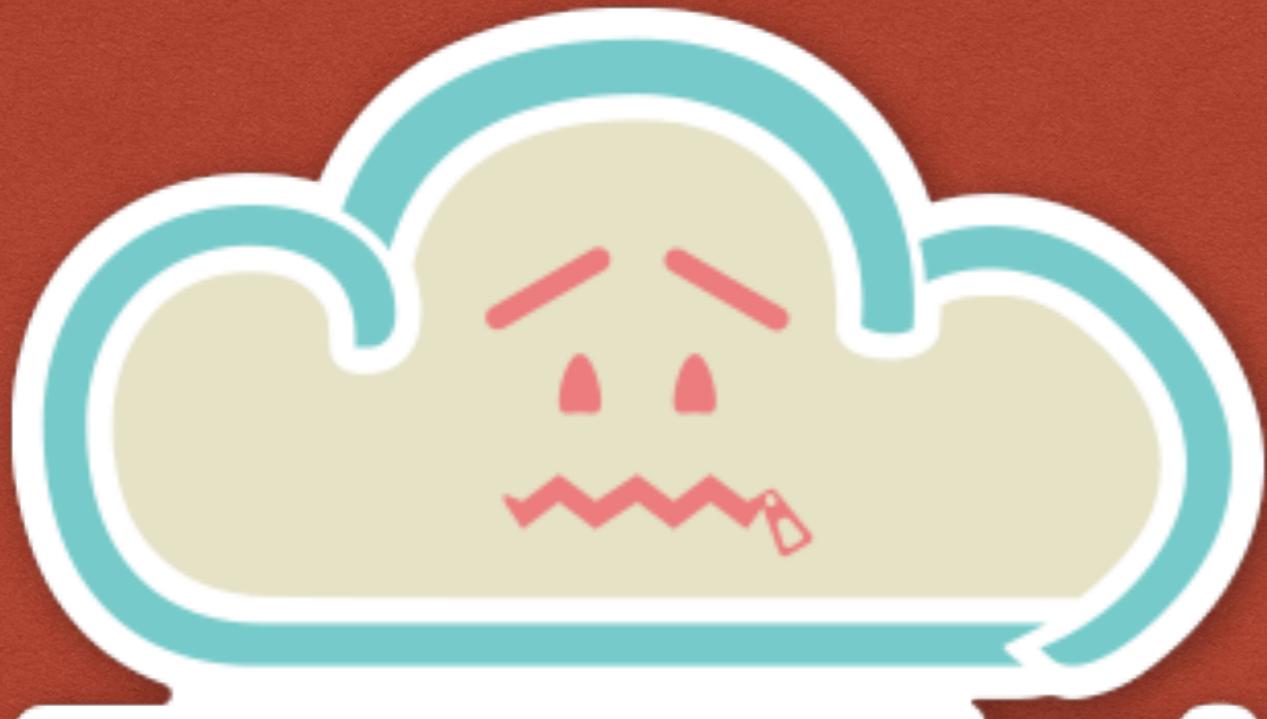
# Distributed Clouds (Fog/Edge)





**There is no cloud**  
it's just someone else's computer

Clouds hide the infrastructure...  
....by adding more layers !

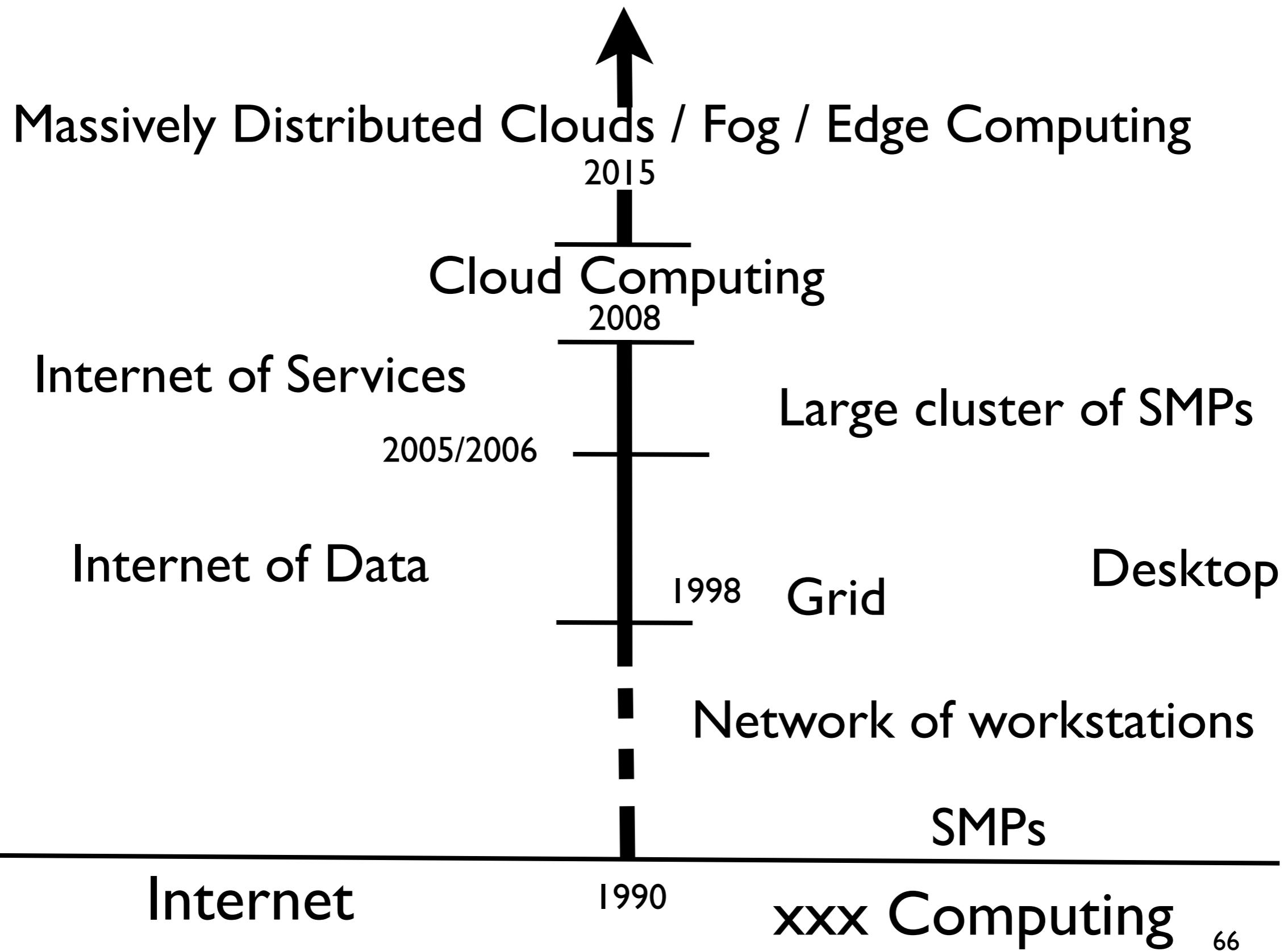


**There is no cloud**  
it's just someone else's computer

and someone else's network

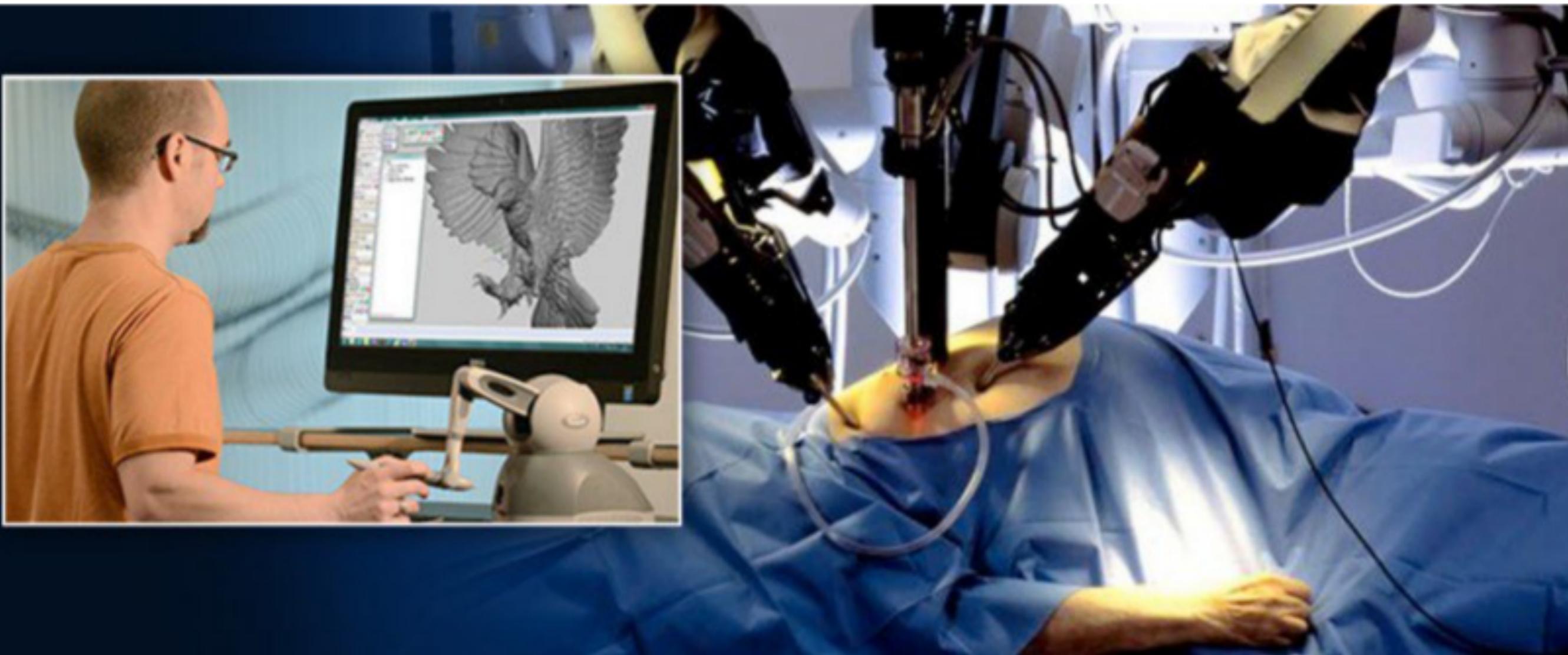
Clouds hide the infrastructure...  
....by adding more layers !

# What's next?

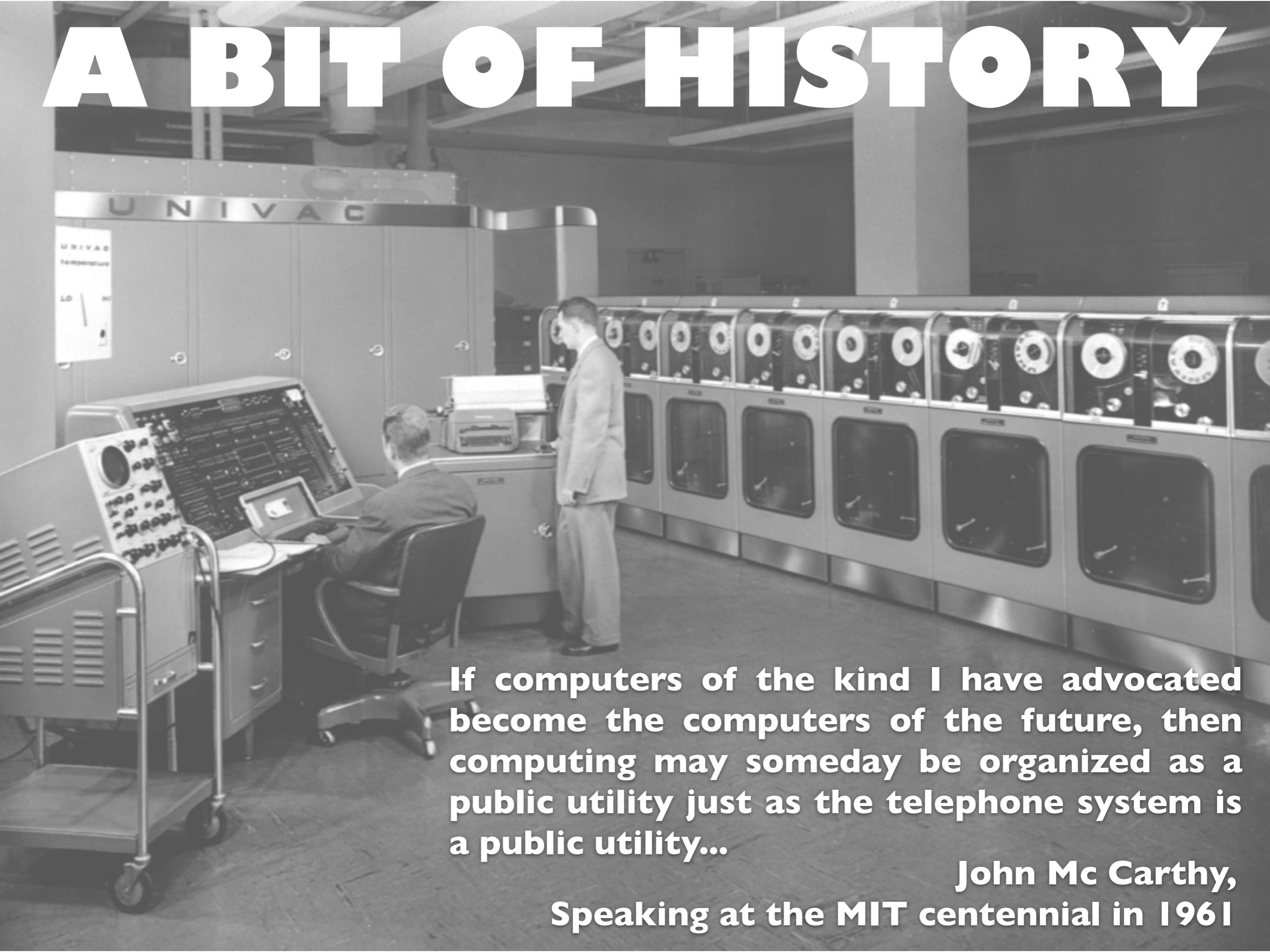


# Internet of Skills/Tactile Internet

- ability to deliver physical experiences remotely



# A BIT OF HISTORY



If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility...

John Mc Carthy,  
Speaking at the MIT centennial in 1961

# Thanks

## Utility

~~Cloud Computing technology is changing every day~~

How developers should develop new applications to benefit from geographically distributed infrastructures.

How to locate hardware/software components?

...

Do not hesitate to push the boundaries



<http://beyondtheclouds.github.io/>

[adrien.lebre@inria.fr](mailto:adrien.lebre@inria.fr)

# Bibliography

- [Bar03] **Xen and the art of virtualization.** By Barham et al. ACM SIGOPS Operating Systems Review, 2003
- [Hiro13] **SimGrid VM: Virtual Machine Support for a Simulation Framework of Distributed Systems.** By Hirofuchi et al. In IEEE Transactions of Cloud Computing, Dec 2015
- [Her10] **Cluster-Wide Context Switch of Virtualized Jobs.** By Hermenier et al., In Proceedings of the International Workshop on Virtualization Technologies in Distributed Computing, June 2010.
- [Her13] **BtrPlace: A Flexible Consolidation Manager for Highly Available Applications.** By Hermenier et al. In IEEE Transactions on Dependable and Secure Computing, Oct 2013
- [Dej11] **Resource Provisioning of Web Applications in Heterogeneous Clouds.** By Dejun et al. In Proceedings of the 2nd Usenix Conference on Web Application Development, June 2011
- [NGuyen17] **Virtual Machine Boot Time Model**  
By Nguyen et al. In Proceedings of the IEEE EuroMicro PDP conference, Mars 2017.

# Bibliography

- **Miscellaneous**
  - **OpenMosix, openSSI and Kerrighed: A comparative Study.** By Lottiaux et al. In Proceedings of the International Conference on Cluster Computing and the Grid, 2005
  - **The grid: blueprint for a new computing infrastructure**  
By Ian Foster and Carl Kesselman, 2004
  - **BOINC: A System for Public-Resource Computing and Storage**  
By Anderson. In Proceedings of the IEEE International Workshop on Grid Computing 2004
  - **Above the Clouds: A Berkeley View of Cloud Computing**  
University of California at Berkeley, Technical Report No. UCB/EECS-2009-28, Feb 2009