Hype in the Cloud, Stacks in the Ground
Andata e ritorno per il Cloud Computing

Davide Salomoni
INFN CNAF
10/5/2013

Cloud computing

Definizione

- La definizione classica di riferimento è quella del National Institute of Standards and Technology (NIST) USA (http://goo.gl/eBGBk)

- In sintesi il Cloud computing si occupa di:

> Fornitura di tecnologia di informazione e comunicazione (ICT) come servizio

5 caratteristiche del Cloud

- **Self-service, on-demand**
  - Il cliente chiede autonomamente ciò che gli serve, quando gli serve e lo ottiene
- **Accesso attraverso la rete**
  - Assume che una rete (Internet o intranet) sia disponibile, normalmente a banda larga
- **Pool di risorse**
  - L'utente non si preoccupa di conoscere i dettagli delle risorse, che sono gestiti dai Cloud resource provider
- **Elasticità**
  - Il servizio Cloud può scalare rapidamente come dimensioni a seconda delle necessità del cliente
- **Pagamento a consumo**
  - Il cliente paga solo per ciò che usa

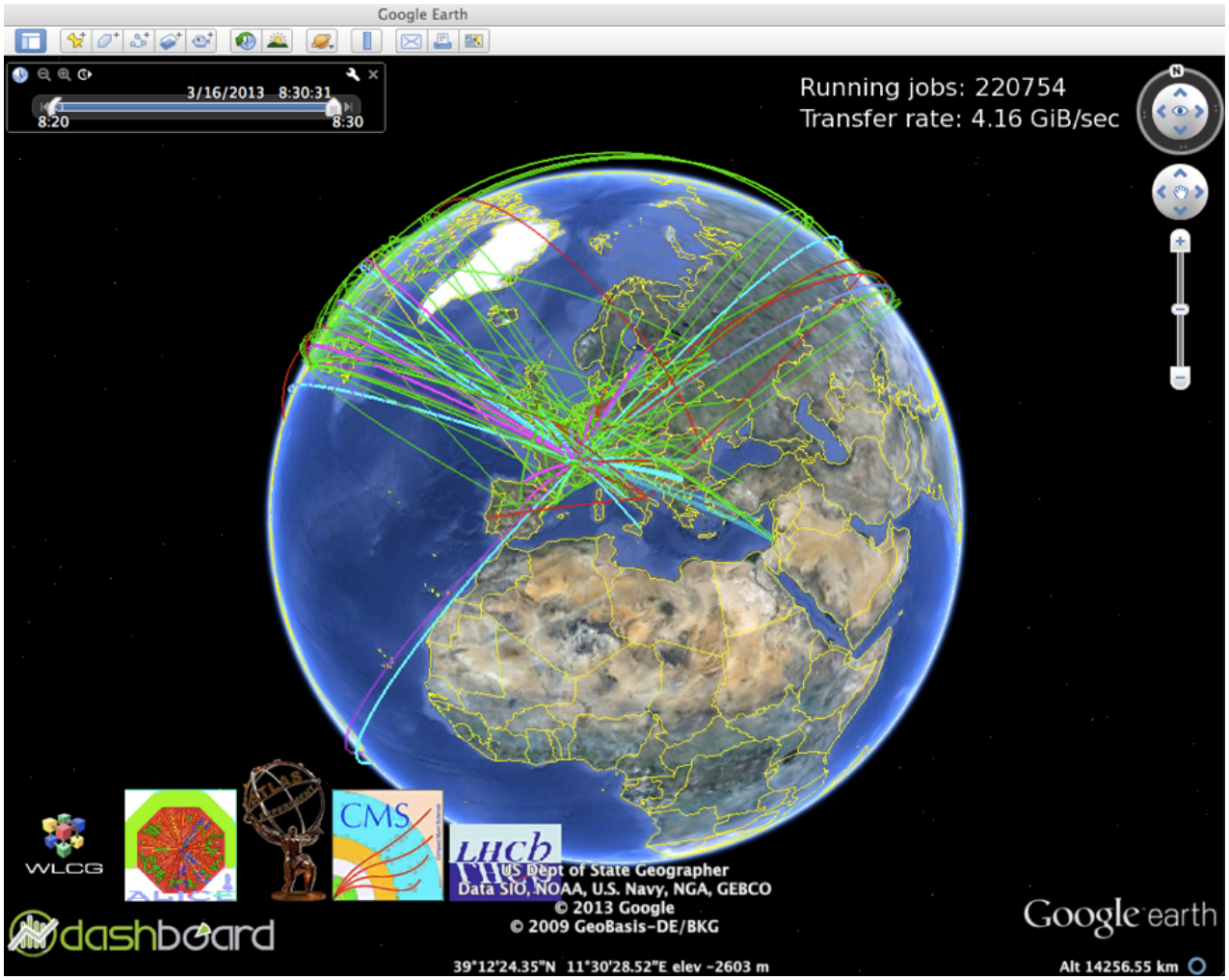
Una analogia: l'autonoleggio

- Self-service, on-demand
  - Prenotazione telefonica oppure online
- Rete
  - Estesa rete di autonoleggi in tutto il mondo
- Pool di risorse
  - Pensa l'autonoleggio a gestire il parco vetture e ad acquisire le macchine che servono
- Elasticità
  - Il numero di auto disponibili normalmente varia a seconda della richiesta
- Pagamento a consumo
  - Il cliente paga per il tempo in cui usa l'auto (e non pensa ad assicurazione, gomme, etc.)

Fonte: http://goo.gl/cEa8M

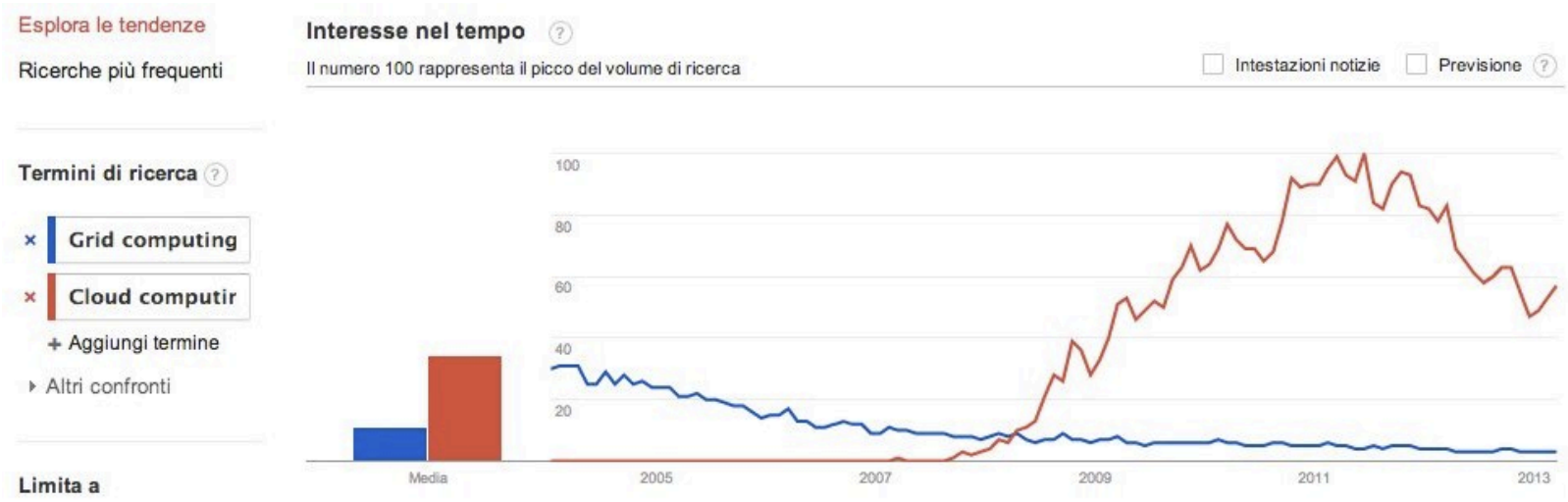
Che cosa c'è di nuovo?

- La fornitura di risorse in modo distribuito, in forme *simili*, avviene da molti anni
- Un esempio di successo nel mondo scientifico è rappresentato dal **Grid Computing**. Cf. http://goo.gl/i5Rkt:
  - Stato in tempo reale dei job di calcolo (Computing) e del trasferimento dati (Storage, Network) legati agli esperimenti LHC al CERN



Tuttavia…

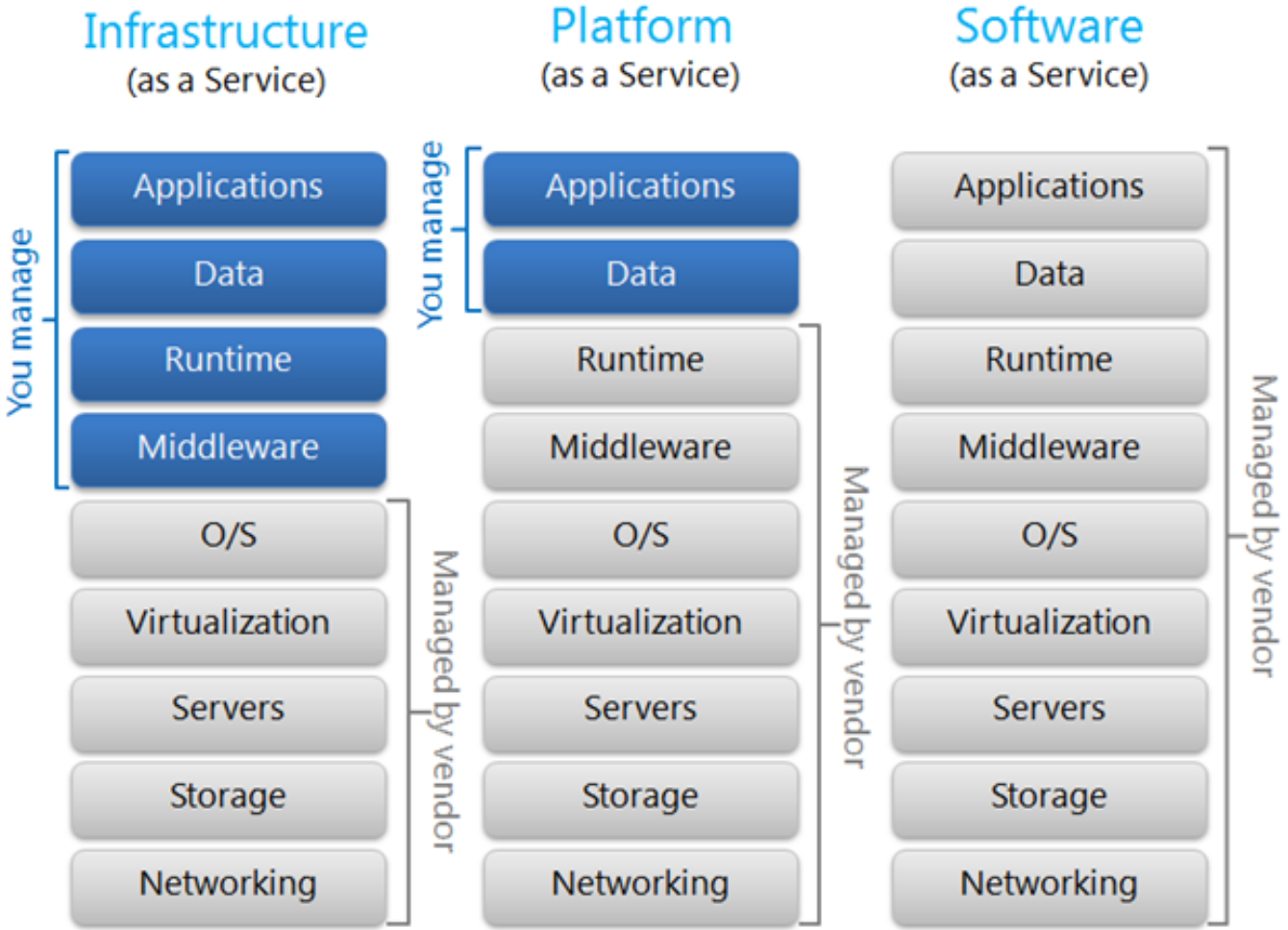
- Il Grid Computing, di enorme successo nel suo campo, non ha mai avuto significativa diffusione al di fuori di grandi collaborazioni scientifiche
- I trends (http://www.google.com/trends/): Grid Computing vs. Cloud Computing



Il focus sul “service”

- Abbiamo visto che nella definizione di Cloud computing il *servizio* nei confronti del cliente è parte essenziale.
- Il Cloud computing si può modellare infatti intorno a *servizi* legati principalmente a
  - Infrastruttura (IaaS → Infrastructure as a Service)
  - Piattaforma (PaaS → Platform as a Service)
  - Software (SaaS → Software as a Service)

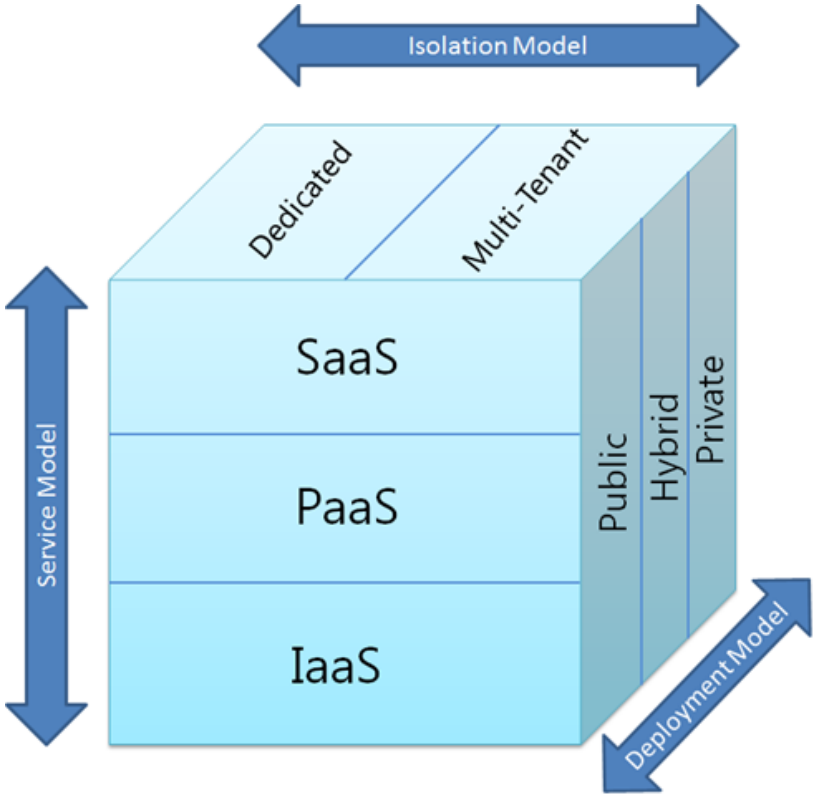
Chi fa cosa?



Fonte: http://goo.gl/1jmkR

Aggiungiamo dimensioni

- Oltre i modelli di *servizio*, parti importanti per definire e capire il Cloud computing sono i modelli di:
  - *deployment* (dove distribuisco i servizi)
  - *isolamento* (come isolo i servizi)



Fonte: http://goo.gl/1jmkR

Deployment: i “tipi di Cloud”

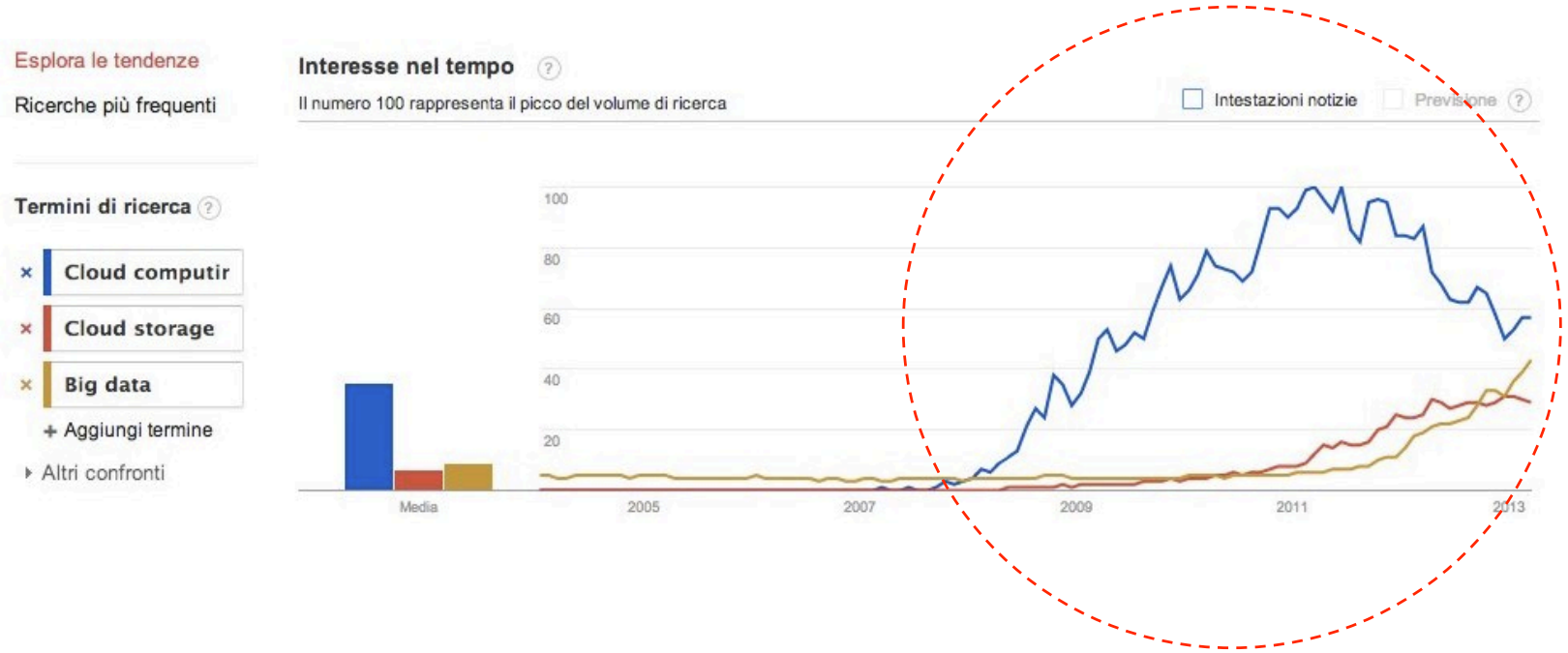
- **Cloud privata:**
  - L’infrastruttura viene fornita per un *uso* esclusivo da parte di una singola organizzazione. La gestione, l’operazione, la proprietà, la dislocazione della Cloud privata tuttavia può essere anche indipendente dall’organizzazione che la usa.
- **Cloud di comunità (Community Cloud):**
  - L’infrastruttura è disponibile ad una comunità di organizzazioni che hanno uno scopo comune (ad esempio missione, requisiti di sicurezza, conformità a regole comuni, etc.)
- **Cloud pubblica:**
  - L’infrastruttura è disponibile in generale al pubblico. La gestione può essere pubblica o privata. La dislocazione è presso il fornitore di servizi.
- **Cloud ibrida:**
  - L’infrastruttura è una combinazione di due o più infrastrutture Cloud (private, di comunità o pubbliche) che sono collegate in modo da garantire forme di portabilità ad esempio di dati o applicazioni.

Isolamento

- I modelli di isolamento nel Cloud (spesso ignorati) sono importanti e si dividono in:
  - Infrastrutture dedicate
  - Infrastrutture “multi-tenant” (con diversi [tipi di] clienti)
- Il tipo di isolamento è importante per molti aspetti, come:
  - Segmentazione delle risorse
  - Protezione dei dati
  - Sicurezza delle applicazioni
  - Auditing
  - Disaster recovery

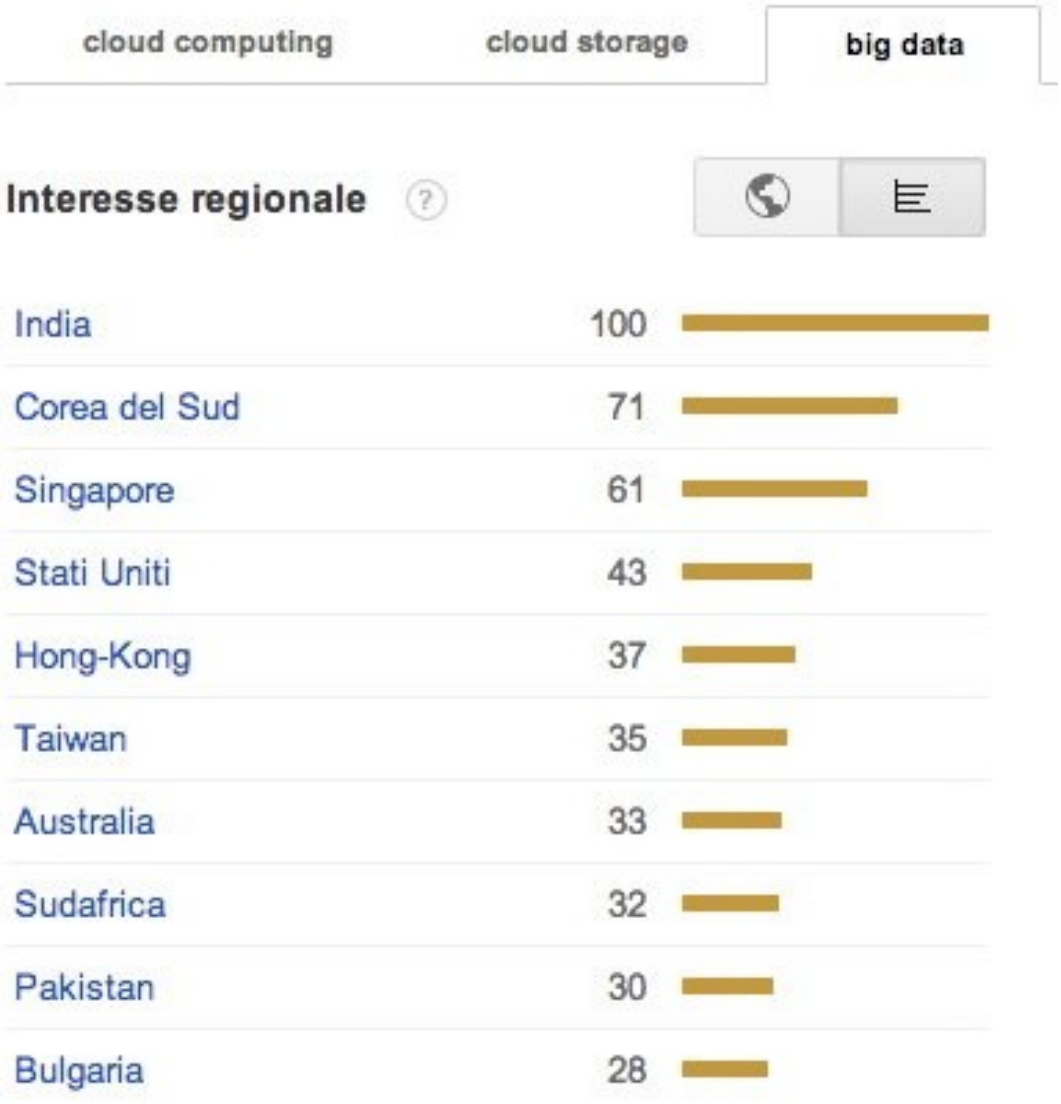
Ma la moda è già passata?

- Cf. i trend di Google per i termini “Cloud computing”, “Cloud storage”, “Big data”



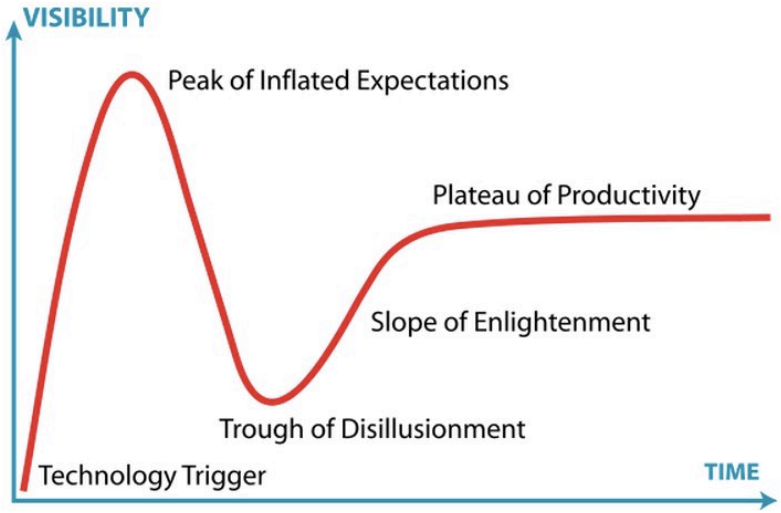
(Intermezzo sui Big data)

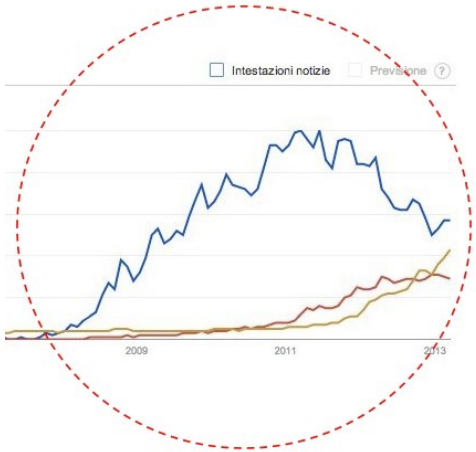
- Questa è la distribuzione geografica riportata da Google per ricerche molto popolari come “Big data”
- Gartner prevede che entro il 2015 il 20% delle principali organizzazioni mondiali avrà sull’**infrastruttura dell’informazione** la stessa attenzione strategica della gestione delle applicazioni (cf. http://goo.gl/ghSk7)



Dunque il Cloud è una moda?

- In molte tecnologie si passa attraverso varie fasi di “hype” (di moda). Possiamo distinguere (cf. http://goo.gl/1qaC) spesso tra:
  - Inizio della tecnologia
  - Picco di aspettative
  - Disillusione (aspettative fallite)
  - Crescita di maturazione
  - Altopiano di produttività

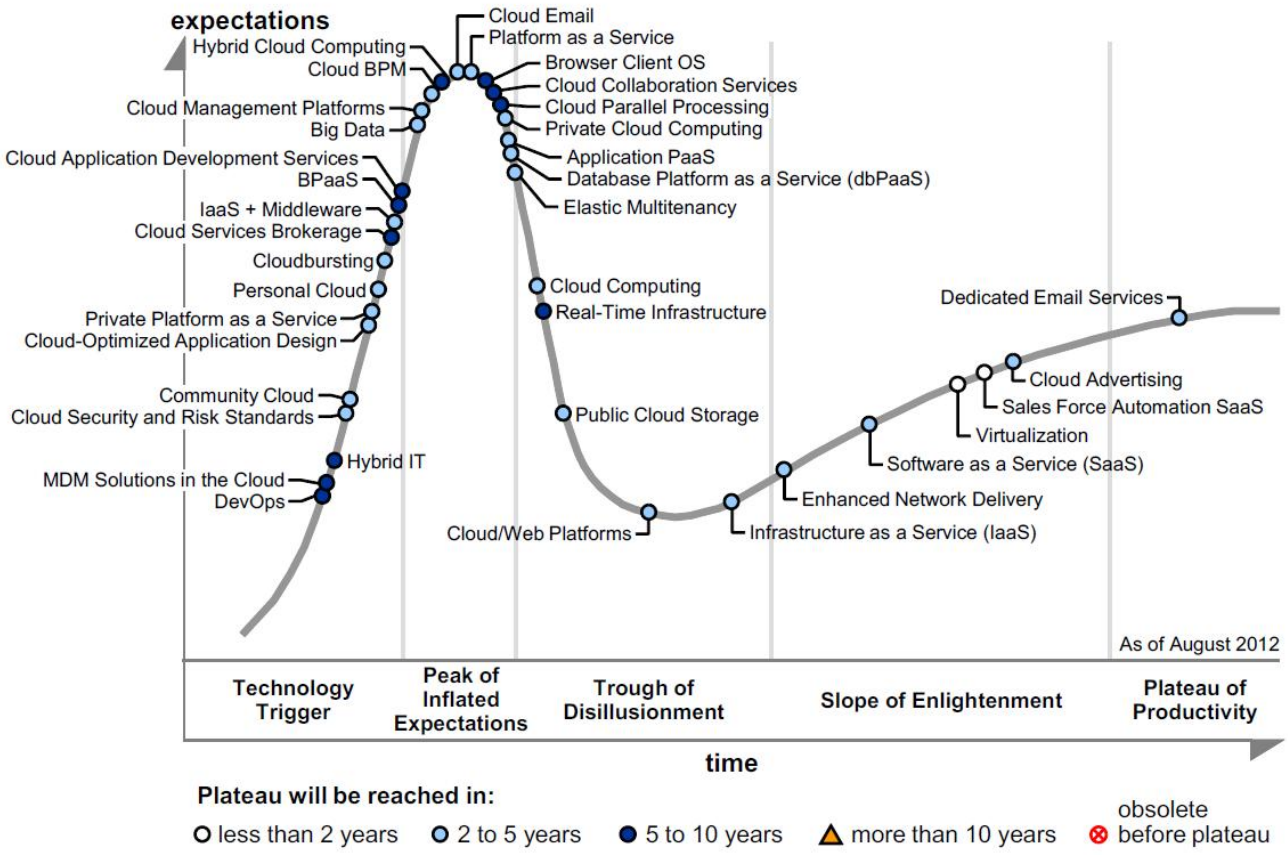




Dove siamo nella moda?

- Forbes, citando Gartner (cf. http://goo.gl/4r1AM), ha riportato in agosto 2012 la sua stima della maturità delle tecnologie associate al Cloud

Figure 1. Hype Cycle for Cloud Computing, 2012



Source: Gartner (August 2012)

A maturazione…

- Entro i prossimi due anni:
  - Virtualizzazione
- Tra due e cinque anni:
  - Big data, public, private o community Cloud, pubblicità nella Cloud, PaaS
- Tra cinque e dieci anni:
  - Hybrid Clouds, real-time infrastructures, paradigma DevOps

Figure 2. Priority Matrix for Cloud Computing, 2012

benefit	years to mainstream adoption			
	less than 2 years	2 to 5 years	5 to 10 years	more than 10 years
transformational	Virtualization	Big Data Cloud Advertising Cloud Computing Community Cloud Platform as a Service	DevOps Hybrid Cloud Computing Real-Time Infrastructure	
high		Application PaaS Cloud BPM Cloud Management Platforms Cloud Security and Risk Standards Cloud/Web Platforms Cloudbursting Cloud-Optimized Application Design Elastic Multitenancy Enhanced Network Delivery Infrastructure as a Service (IaaS) Personal Cloud Private Cloud Computing	Cloud Application Development Services Cloud Parallel Processing Cloud Services Brokerage Hybrid IT	
moderate	Sales Force Automation SaaS	Cloud Email Database Platform as a Service (dbPaaS) IaaS + Middleware Private Platform as a Service Public Cloud Storage Software as a Service (SaaS)	BPaaS Cloud Collaboration Services MDM Solutions in the Cloud	
low		Dedicated Email Services	Browser Client OS	

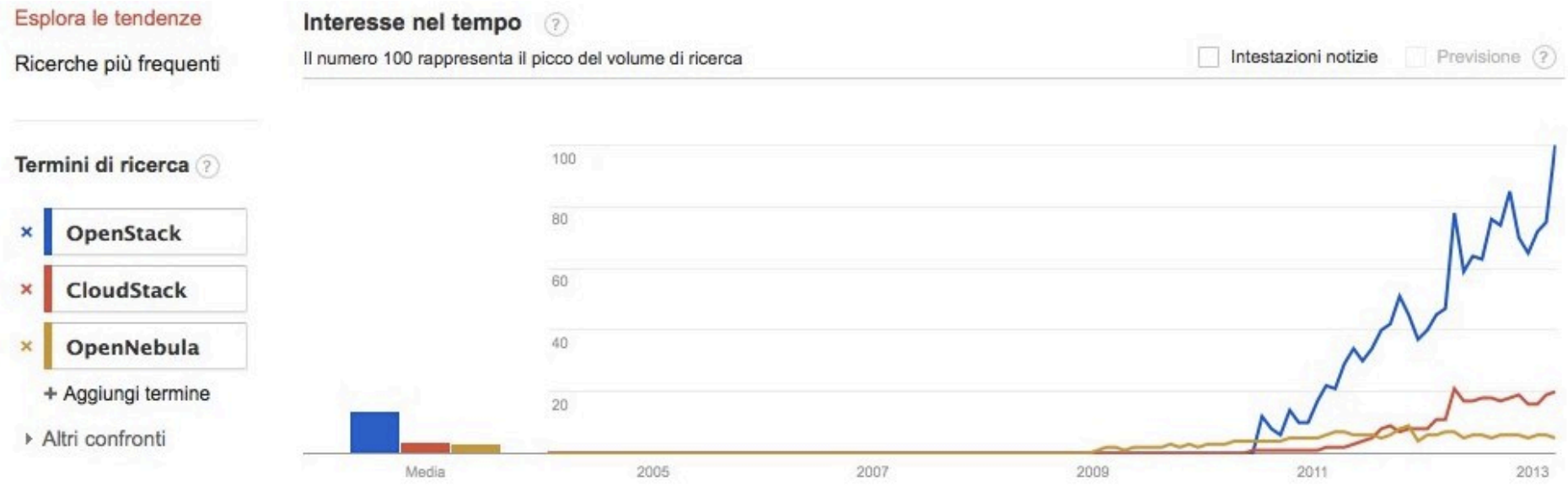
As of August 2012

Source: Gartner (August 2012)

Gli stack Cloud IaaS più usati

Ancora sui trend

- Abbiamo visto l'importanza degli standard (*de jure* o *de facto*) nei casi d'uso. Limitandoci per ora agli stack Cloud di tipo aperto, questi i Google trends per OpenStack, CloudStack, OpenNebula:



OpenStack

- Alcuni punti importanti:
  - È un prodotto open source, con dipendenze di tipo open source e che può essere eseguito su piattaforme interamente open source (ad es. Linux)
  - Ha un forte supporto da parte dell'industria
    - Ad es. Rackspace, Intel, Cisco, Juniper, NetApp, HP, DELL, VMware, AT&T, IBM, Canonical, SUSE, RedHat, Yahoo!
  - In forte crescita in termini di funzionalità e di sviluppatori (cf. http://goo.gl/lBHzn per una comparazione con OpenNebula, CloudStack, Eucalyptus)
  - Ha un disegno architetturale aperto e modulare, principalmente sviluppato in Python
  - Ha una governance interna ben definita che non è in mano a nessun singolo ente o impresa
  - Interopera con altri sistemi di Cloud computing pubblici o privati
    - Ad es. VMware ESXi, Microsoft Hyper-V, Amazon EC2

E OpenStack vs. VMware?

- Un buon articolo si trova in http://goo.gl/hsrwN
- Possiamo distinguere alcune categorie di comparazione:
  - Design
  - Features
  - Use cases
  - Value
  - Performance

OS vs. VMware: Design

- ESXi è un'ottima base per prodotti come vSphere e vCloud, con eccellente documentazione
  - Tuttavia, il sistema è chiuso e la sua evoluzione è totalmente dipendente da VMware.
  - Attenzione a questo punto: è assai spiacevole dover tracciare le fortune o sfortune, le scelte o le non scelte di una ditta per verificare le proprie strategie.
- OpenStack è più nuovo, con deployment e architettura più complesse di quella di VMware.
  - Ma nessuno controlla singolarmente il suo sviluppo e ha un grande supporto di industria e sviluppatori

OS vs. VMware: Features

- Migrazione
  - VMware: vMotion, richiede uno storage condiviso
  - OS: live migration (storage condiviso) oppure block migration (non richiede storage condiviso)
- Scheduling
  - VMware: DRS (resource scheduling) e DPM (power management) per il consolidamento delle risorse
    - Sono feature molto attraenti. Tuttavia DRS non è controllabile in quanto a pesi e a dimensione temporale (es. la CPU è alta durante un backup notturno e DRS sposta live la VM, senza reale motivo)
  - OS: scheduling per compute e per volumi. Questi scheduler sono modificabili attraverso JSON. OpenStack non è però qui flessibile come VMware perché ad es. non può spostare automaticamente le macchine “live”.

OS vs. VMware: Features

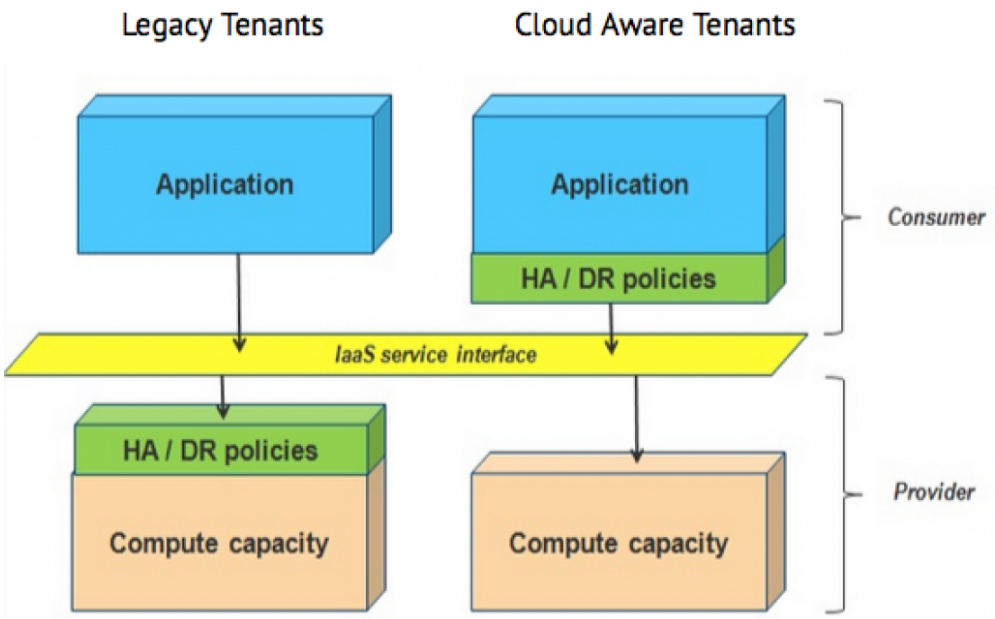
- HA
  - VMware: spawn di una VM su un host differente su la VM o l'hypervisor hanno problemi
  - OpenStack: non c'è supporto diretto. E' implementabile esternamente. Vedi anche le slides seguenti.
- Fault tolerance
  - VMware: controlla lo stato di una VM e ne fa il mirroring continuo su un ESXi secondario. Tuttavia funziona solo in caso di failure dell'hypervisor, duplica le risorse richieste come RAM, disco, CPU, bandwidth e una sola vCPU può essere protetta in questo modo.
  - OpenStack: non c'è una feature simile e neanche il progetto di inserirla, sia per motivi architetturali (vedi dopo) sia perché KVM non supporta l'instruction mirroring.

OS vs. VMware: casi d'uso

- Qui si gioca buona parte di questa comparazione: le feature sono importanti ma devono essere mappate su casi d'uso reali.
- In generale, è bene distinguere tra "virtualizzazione" e "Cloud computing".
- I casi d'uso sono alla fine legati alle applicazioni. Ma quali sono i requirement di una applicazione "di tipo Cloud", o che "possa andare su Cloud"?

La mia applicazione è “cloud-friendly”?

- Applicazioni “cloud-aware”:
  - Distribuite
  - Stateless
  - Fail-over in the app
  - Scaling in the app
- Applicazioni “legacy”:
  - Client-server
  - Monolitiche, senza scalabilità orizzontale
  - Fail-over nell’infrastruttura
  - Scaling nell’infrastruttura



Fonte: VMware

Una analogia: i cagnolini e le mucche

- Le applicazioni “legacy” vengono curate come animali domestici, sono uniche e spesso non rimpiazzabili
- Le applicazioni “cloud” vengono trattate come mucche in una stalla. Ad esempio quando una mucca si ammala la sostituiamo con una delle tante altre (funzionalmente uguali) che abbiamo a disposizione.

Fonte: http://goo.gl/Gx0ly

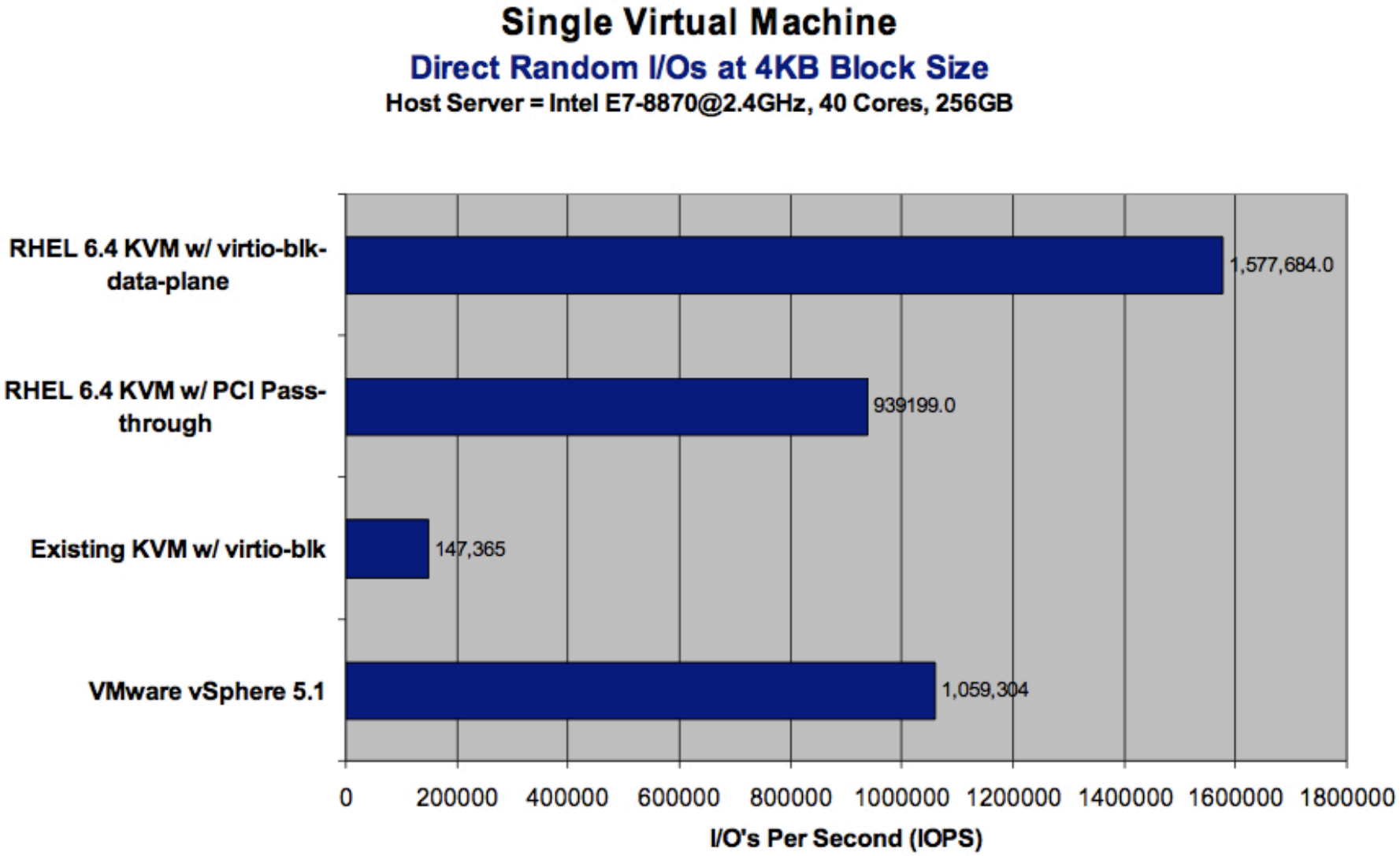
Dai casi d'uso alla Cloud

- "Applications are being built to control and manipulate the infrastructure" (L.Moorman, presidente di Rackspace, http://goo.gl/LQtGu)
- La applicazioni "di tipo Cloud" devono essere largamente indipendenti dall'infrastruttura per poter scalare orizzontalmente (e per essere indipendenti dal fornitore di servizi Cloud)
- Questo implica che, nel Cloud, molte delle features presenti nella suite VMware non sono necessariamente così utili come sembrano
  - Ad esempio, nelle app "legacy" HA è gestito dell'infrastruttura. Nelle app "cloud" HA è gestito autonomamente.
  - "Ultimately, VMware today is not about cloud, it is about datacenter automation. It is not about infrastructure "as a service", it is about virtualization offerings focused at very specific enterprise pain points." – Boris Renski, Mirantis (http://goo.gl/ho6CZ)

E le performance?

- Un'area tipicamente scarsa per tecnologie di virtualizzazione è l'I/O
- Da un paper su "KVM Virtualized I/O Performance" di IBM e RedHat (http://goo.gl/tgmed), Febbraio 2013:
  - KVM (con RedHat 6.4), utilizzato da OpenStack, gestisce *per una singola VM* fino a 1.2 milioni di IOPS con richieste random di 8KB, sostanzialmente in linea con i valori dell'hardware non virtualizzato
  - Questo è circa il 50% in più di quanto dichiara vSphere 5.1 (cf. http://goo.gl/SliZf)

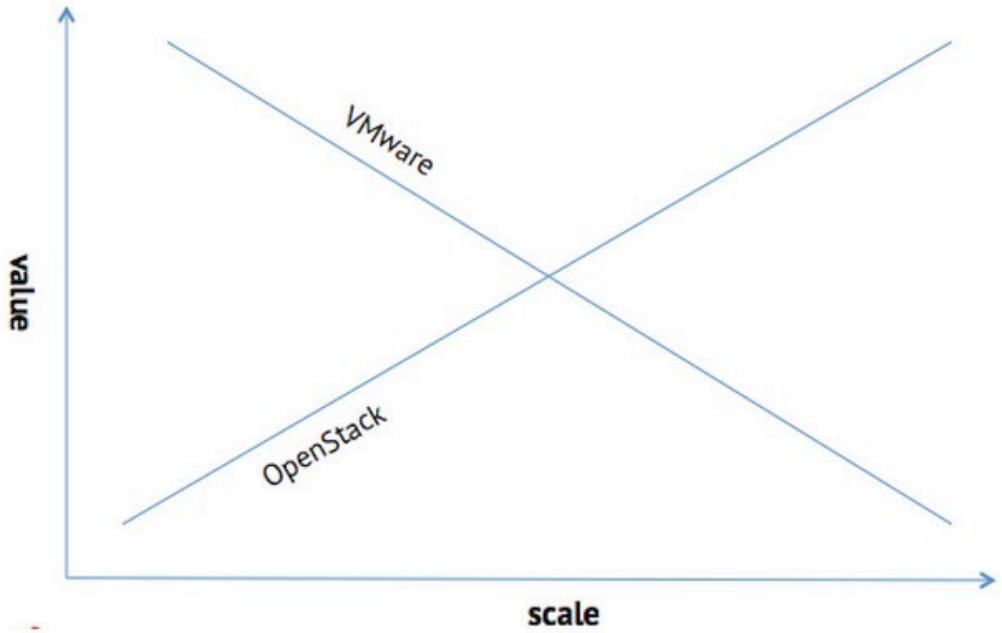
Comparazione di I/O



Fonte: http://goo.gl/tgmed

Concludendo, OpenStack vs. VMware

- OpenStack è gratis ma richiede uno sforzo significativo per la corretta ingegnerizzazione
  - Questo sia perché è più nuovo e sia perché supporta un numero molto alto di scenari applicativi e di installazioni
- VMware costa ma è più facile da utilizzare, anche grazie a un uso maggiore di interfacce grafiche. Tuttavia molte feature sono teoricamente interessanti ma la loro reale applicabilità è da verificare
  - Un problema significativo è la applicabilità in ambito “Cloud” che si estenda oltre il singolo data center e soprattutto il rischio strategico
- Il “Cloud costing” è un argomento complesso (definito a volte “dark art”) e che richiede molto tempo; non ne parlo qui in ulteriore dettaglio.



OpenStack, recap

- Alcuni punti importanti di OpenStack (http://www.openstack.org/):
  - È un prodotto open source, con dipendenze di tipo open source e che può essere eseguito su piattaforme interamente open source (ad es. Linux)
  - Ha un forte supporto da parte dell'industria
    - Ad es. Rackspace, Intel, Cisco, Juniper, NetApp, HP, DELL, VMware, AT&T, IBM, Canonical, SUSE, RedHat, Yahoo!
  - In forte crescita in termini di funzionalità e di sviluppatori (cf. http://goo.gl/lBHzn per una comparazione con OpenNebula, CloudStack, Eucalyptus)
  - Ha un disegno architetturale aperto e modulare, principalmente sviuppato in Python
  - Ha una governance interna ben definita che non è in mano a nessun singolo ente o impresa
  - Interopera con altri sistemi di Cloud computing pubblici o privati
    - Ad es. VMware ESXi, Microsoft Hyper-V, Amazon EC2

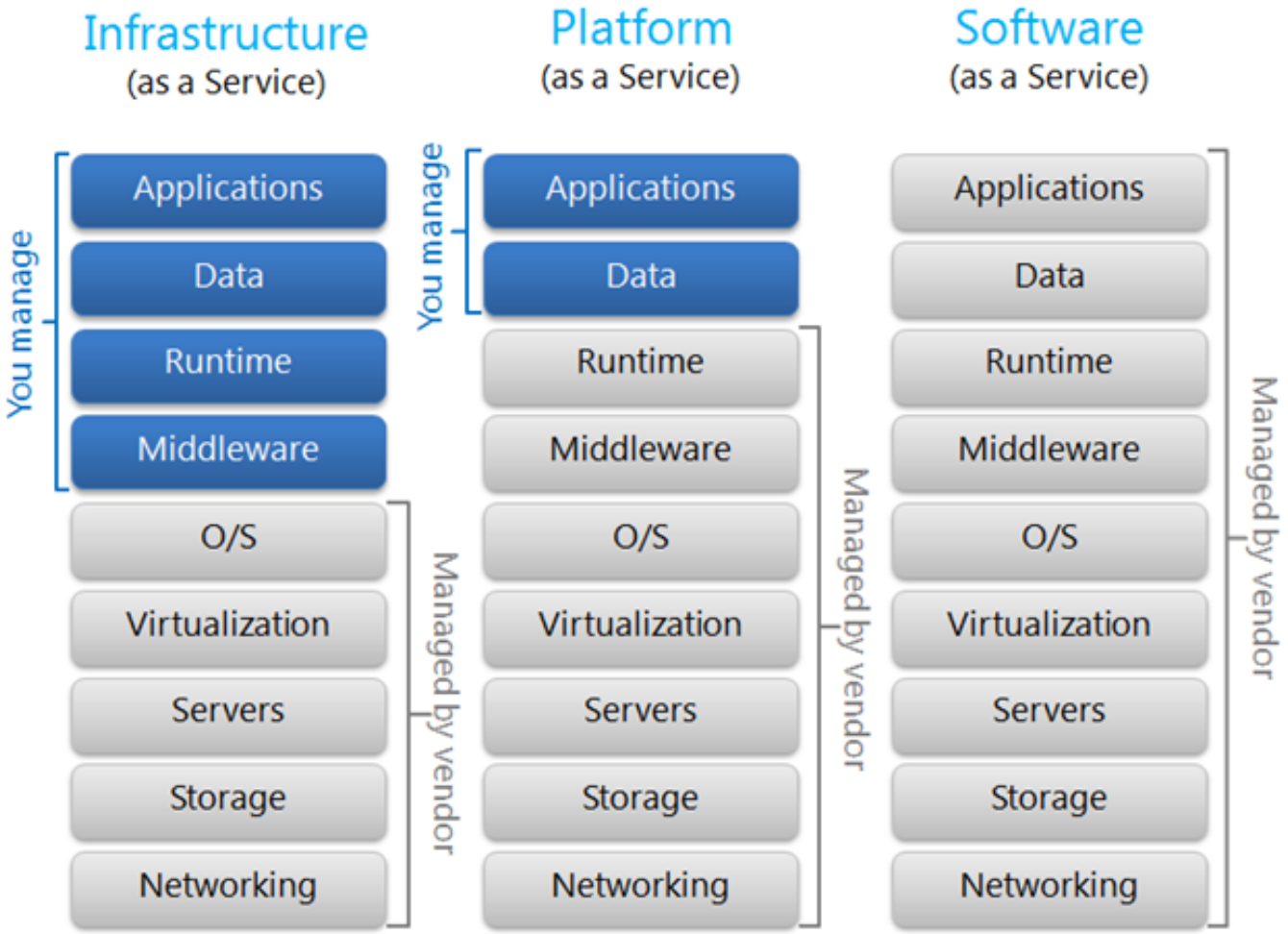
Qualche numero

- Alla fine del 2012 (cf. http://goo.gl/d6vG8):
  - 6.695 membri della comunità di OpenStack in 87 paesi
  - Più di 550 sviluppatori sui vari progetti che compongono OpenStack
  - Più di 300.000 download di OpenStack dai repository centrali
  - Sponsorizzato da 155 industrie/compagnie
  - Solo nel 2012 sono stati lanciati 48 user groups in 33 paesi
  - La partecipazione al Design Summit di OpenStack in autunno 2012 è aumentata di tre volte rispetto al Design Summit di primavera
  - La comunità di OpenStack nei social media è stimata essere circa sei volte quella del prodotto concorrente open source più prossimo

Open Source

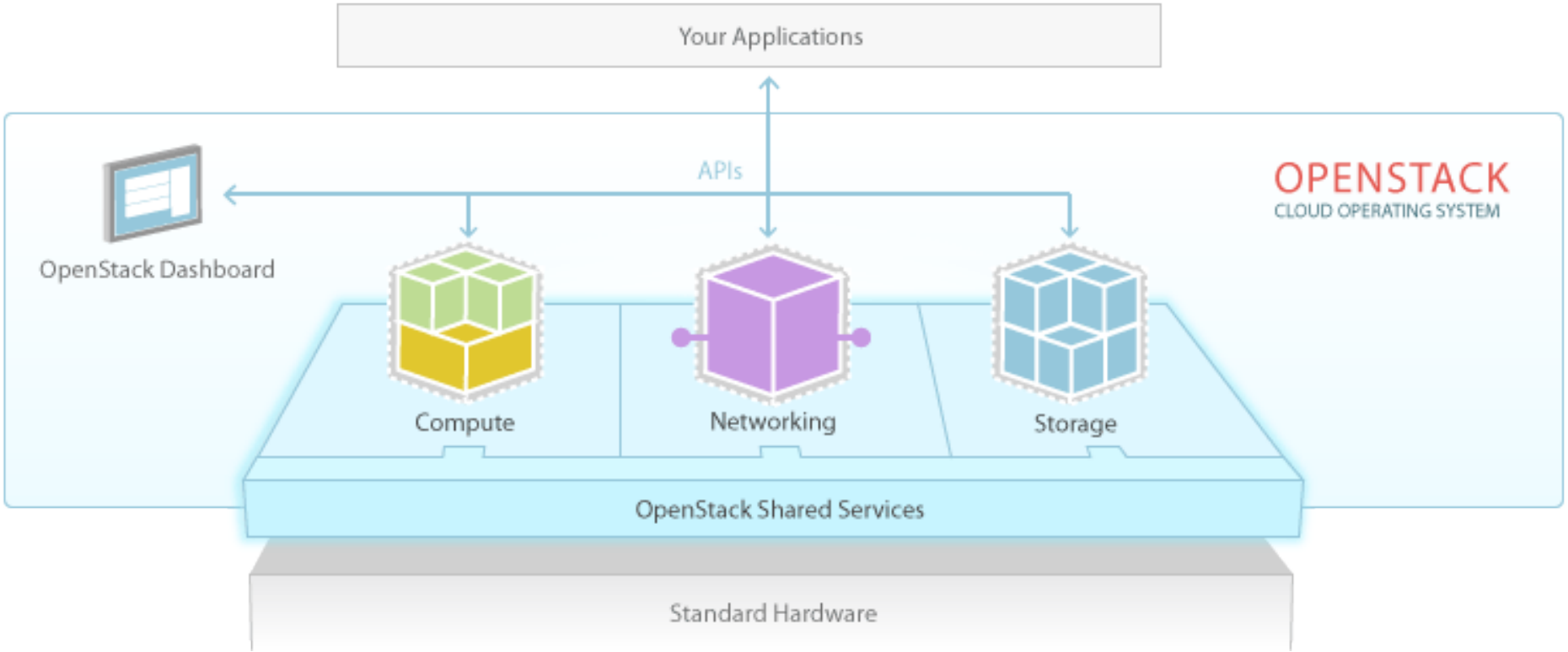
- Per definizione, è disponibile il *codice sorgente* del prodotto (OpenStack, ma ad esempio anche Linux, Apache web server, Firefox, Calibre, etc.)
  - Analisi di feature, sicurezza, trasparenza, contributi allo sviluppo
- Da “Raccomandazioni e proposte sull’utilizzo del Cloud computing nella pubblica amministrazione”, giugno 2012, http://goo.gl/h7OI8:
  - “Le amministrazioni dovrebbero selezionare fornitori di **servizi cloud conformi agli standard e alle altre caratteristiche tecnologiche che garantiscano portabilità e interoperabilità dei servizi erogati**. L’infrastruttura di un fornitore di servizi cloud deve garantire che i servizi cloud possano essere trasferiti su piattaforme di fornitori differenti ovvero possano eventualmente essere riportati all’interno dell’organizzazione cliente con il minimo di impatto, così da evitare il rischio di legarsi ad un unico cloud provider (il cosiddetto vendor lock-in).”
- Modifica dell’art. 68 del CAD attraverso DL 18/10/2012, convertito in legge il 17/12/2012, cf. http://goo.gl/a9DC1 (art. 9 bis):
  - “**Ove** dalla valutazione comparativa di tipo tecnico ed economico, secondo i criteri di cui al comma 1-bis, **risulti motivatamente l’impossibilita’ di accedere a soluzioni già disponibili all’interno della pubblica amministrazione, o a software liberi o a codici sorgente aperto, adeguati alle esigenze da soddisfare**, è consentita l’acquisizione di programmi informatici di tipo proprietario mediante ricorso a licenza d’uso.”

OpenStack: piattaforma IaaS



Fonte: http://goo.gl/1jmkR

Visione d'insieme



Fonte: OpenStack

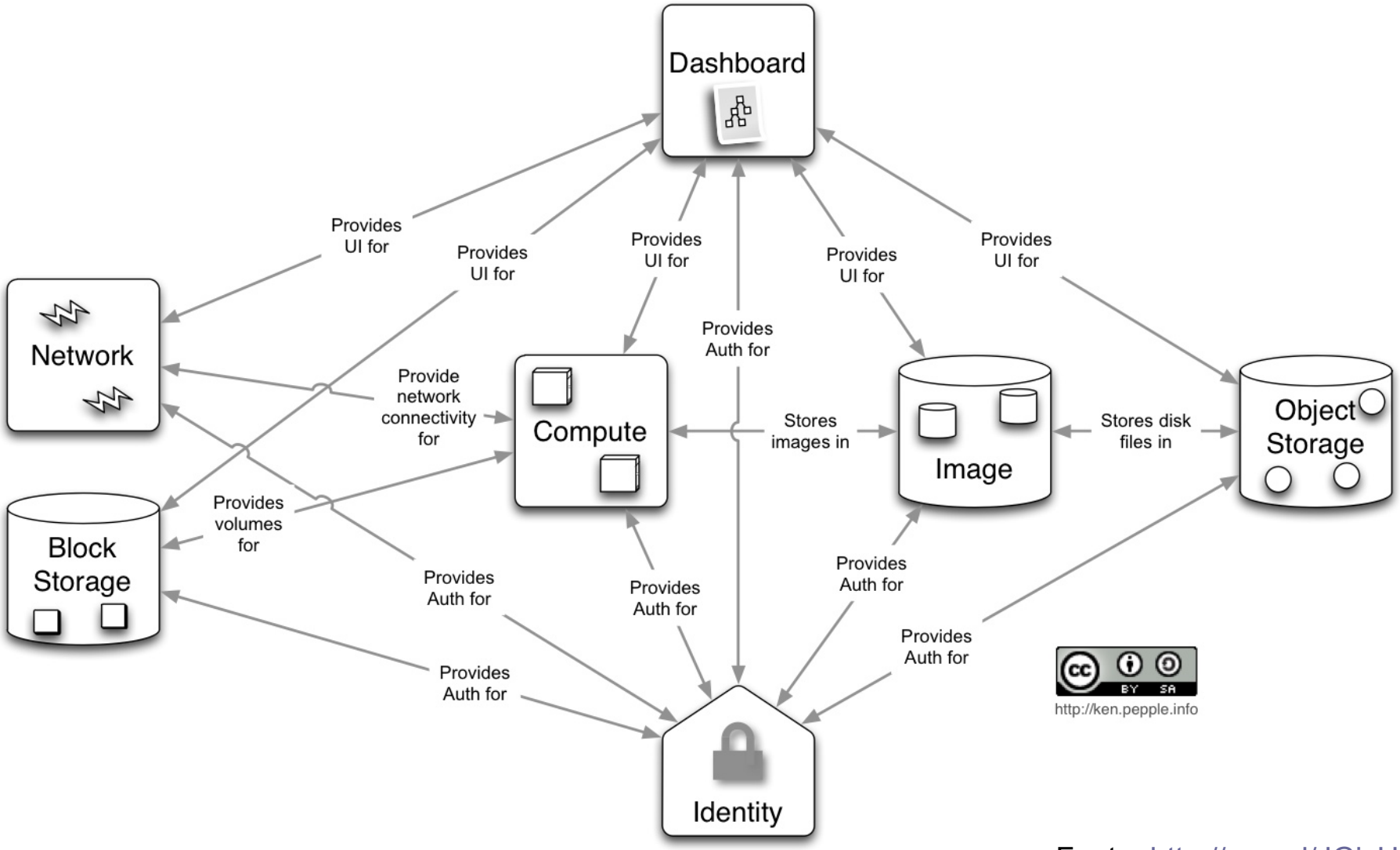
I nomi delle release

- In OpenStack ogni *major release* del software ha un nome in codice
  - Il ciclo di rilascio delle major release è attualmente di 6 mesi (http://goo.gl/gMRhb)
  - La versione attuale, rilasciata ad aprile 2013, è chiamata **Grizzly**. Questa è la settima release di OpenStack (http://goo.gl/MlPbu).
  - La prossima versione, chiamata *Havana*, è prevista uscire a metà di ottobre 2013.
- La maggior parte della informazioni che seguono sono basate su OpenStack *Folsom*, rilasciata a settembre 2012
  - Verranno comunque fornite osservazioni su Grizzly ove applicabile
  - NB: Folsom è disponibile in EPEL, Grizzly (ancora) no

I nomi dei componenti

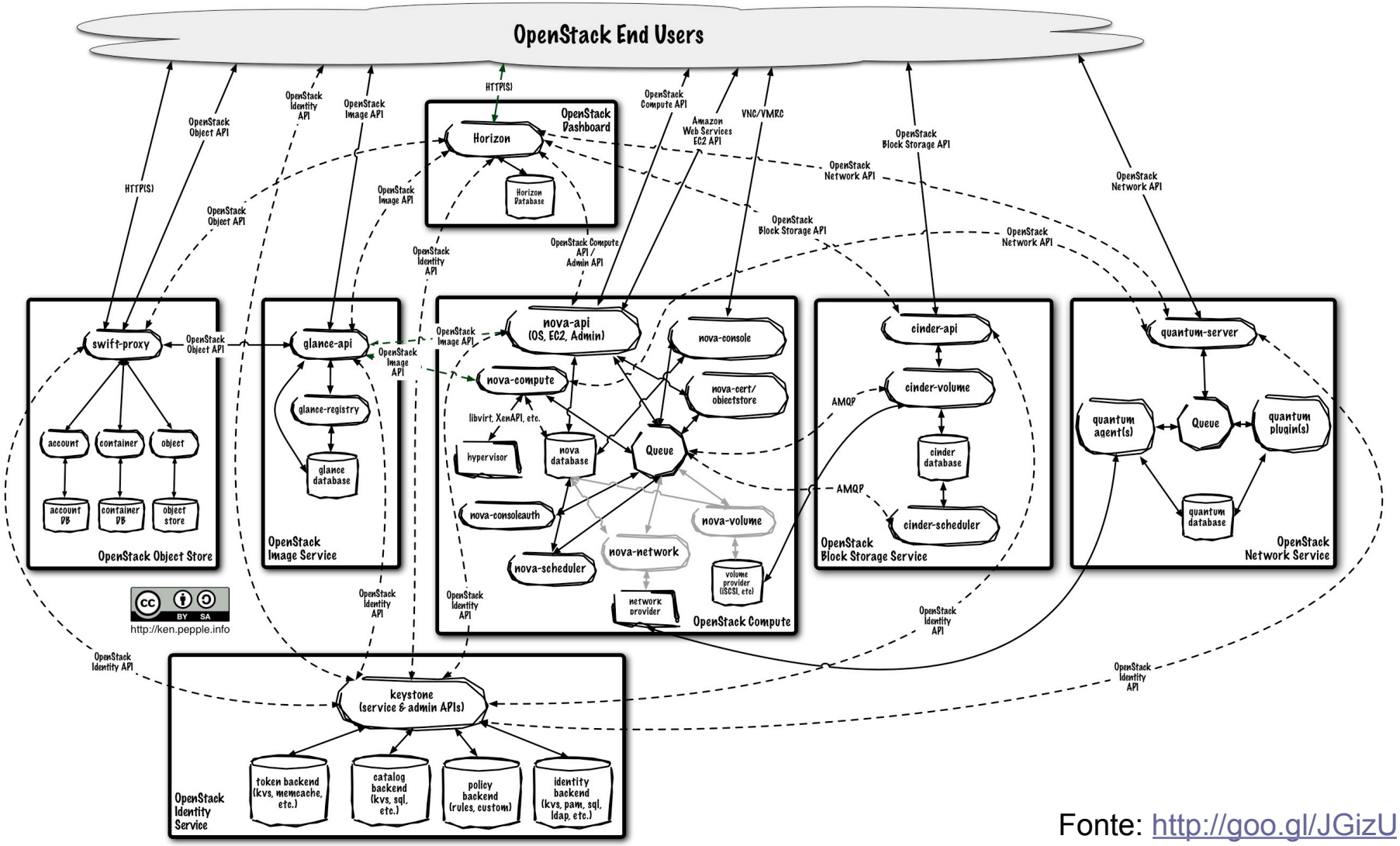
- Ogni componente funzionale (modulo) di OpenStack ha un nome in codice. Questi sono i moduli presenti nelle release Folsom e Grizzly:
  - Dashboard (web interface) → **Horizon**
  - Servizio di immagini (catalogo, repository) → **Glance**
  - Compute (server virtuali) → **Nova**
  - Network (gestione della rete) → **Quantum**
  - Identità (autenticazione, autorizzazione) → **Keystone**
  - Block storage (gestione dei volumi) → **Cinder**
  - Object store (gestione di files) → **Swift**

L'architettura a moduli



Fonte: http://goo.gl/JGizU

Il diavolo sta nei dettagli…



Fonte: http://goo.gl/JGizU

Ma in sintesi…

- Gli utenti finali interagiscono con i servizi attraverso una interfaccia web comune, oppure attraverso API specifiche di ogni servizio
- Tutti i servizi hanno una autenticazione comune
- I singoli servizi interagiscono tra di loro attraverso le rispettive API pubbliche

Comparato ad Amazon Cloud

- Diversi dei servizi di OpenStack sono concettualmente simili a quelli forniti dal servizio Cloud di Amazon

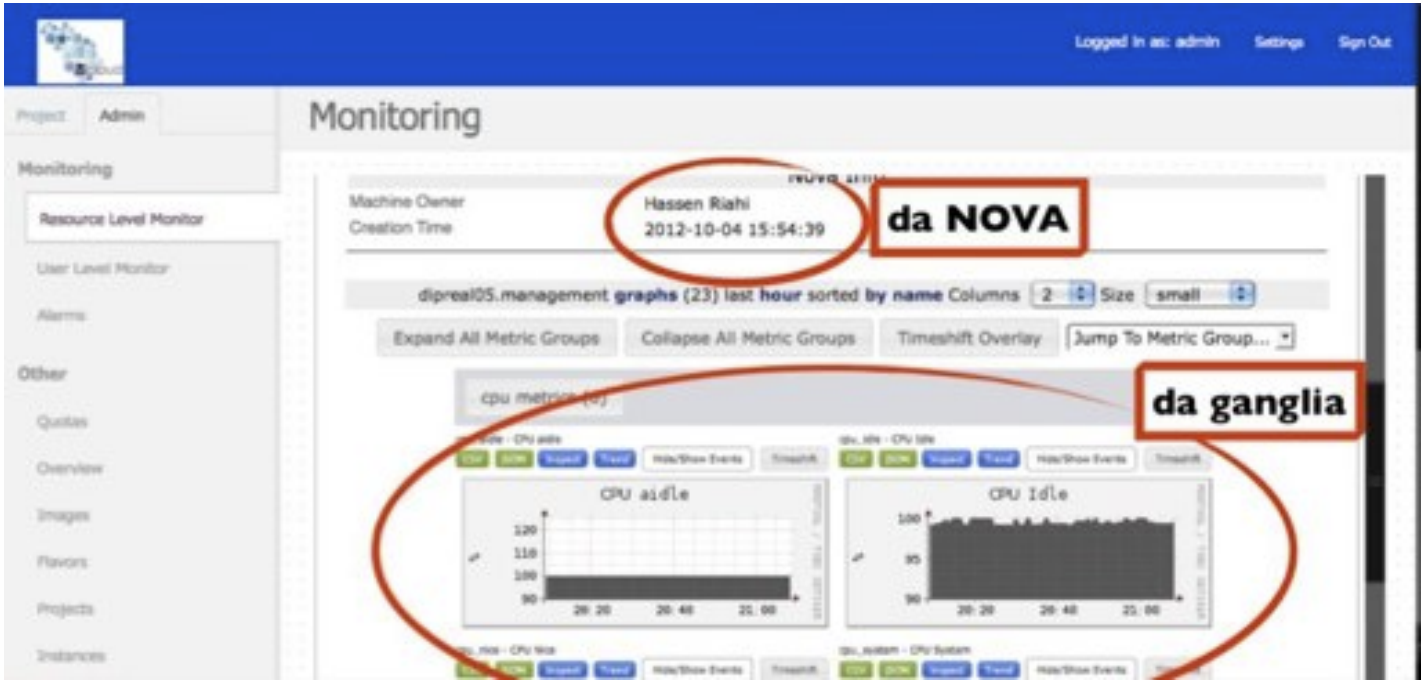
- **Nova** (la parte “compute”) corrisponde ad Amazon EC2 (“Elastic Compute Cloud”, http://goo.gl/2r8X)
  - Esistono API in OpenStack che consentono di interoperare direttamente con EC2 (http://goo.gl/lxh9T)
- **Swift** (object store) corrisponde ad Amazon S3 (“Simple Storage Service”, http://goo.gl/ailE)
- **Glance** (image service) corrisponde al catalogo AMI di Amazon
- **Cinder** (volumi, o block storage) corrisponde ad Amazon EBS (“Elastic Block Storage”, http://goo.gl/Q7Fb)

Horizon (Dashboard)

- Horizon è il modulo che fornisce l'interfaccia utente via Web per amministratori e per utenti finali
  - È scritta in Django (http://goo.gl/WWtun), un framework per lo sviluppo in Python di applicazioni web
    - La personalizzazione è resa possibile dalla separazione tra la parte di presentazione e quella di interfacciamento con il resto di OpenStack.
  - Usa Apache attraverso il modulo `mod_wsgi`
- Nota: non è indispensabile usare Horizon come dashboard
  - Si può senza problemi sviluppare un proprio pannello di controllo (magari solo per gli utenti) che comunichi con le API di OpenStack
  - Scelta adottata da alcuni fornitori di servizi Cloud

Esempio di dashboard

- La figura seguente mostra una dashboard OpenStack base alla quale sono stati aggiunti componenti legati a monitoring (con Ganglia) ed allarmistica (con Nagios)



Glance (gestione immagini)

- È un servizio autonomo (può essere installato anche indipendentemente da OpenStack) che fornisce un catalogo per la memorizzazione e la gestione di immagini virtuali
- È diviso in tre blocchi fondamentali:
  - Le API
  - Un database (spesso MySQL o SQlite)
  - Il registro delle immagini. Questo può essere (come nei diagrammi precedenti) Swift, ma anche un altro tipo di servizio di memorizzazione delle immagini. In MCloud abbiamo usato GlusterFS (http://goo.gl/pfyg)
- Supporta immagini di diversi formati
  - Raw, AMI, VHD (Hyper-V), VDI (VirtualBox), qcow2 (QEMU/KVM), VMDK (VMware), OVF (VMware, altri)

Nova (compute) – 1

- È un framework per la fornitura e la gestione su larga scala di istanze virtuali.
- Supporta diversi tipi di hypervisor (cf. http://goo.gl/mVdjG per dettagli)
  - XenServer, KVM, QEMU, LXC, ESXi, Hyper-V
- Utilizza un database per la memorizzazione delle configurazioni e degli stati a run-time dell'infrastruttura
  - Ad esempio quali tipi di istanza sono disponibili, quali istanze sono in uso, i progetti, gli utenti, le reti, etc. Normalmente si usa MySQL o PostgreSQL.
- All'interno di nova, una parte di cloud controller si occupa dell'orchestrazione della comunicazione tra i vari componenti
  - Utilizzando un sistema di messaggistica basato su code che sfrutta l'Advanced Message Queuing Protocol (AMQP, http://goo.gl/WmxSO). L'implementazione di AMQP può essere data ad esempio da prodotti come Apache Qpid, RabbitMQ o ZeroMQ.

Nova (compute) – 2

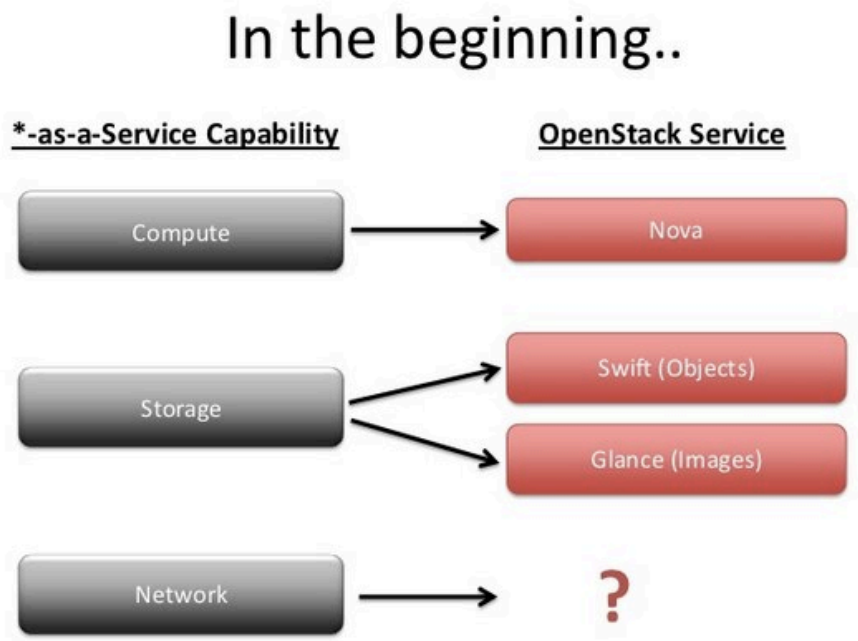
- La parte chiamata nova-schedule si occupa di prendere una richiesta di allocazione dalla coda delle richieste e determinare su quale host fisico deve essere eseguita
  - Concettualmente semplice ma i dettagli possono essere complessi
- Altri servizi di nova si occupano della gestione dell'accesso alle console delle macchine virtuali
- Esiste una parte opzionale per la gestione dei volumi di disco chiamata `nova-volume`
  - E' possibile usarlo direttamente, oppure l'alternativa è utilizzare il componente dedicato chiamato *Cinder*
- Esiste una parte opzionale per la gestione della rete chiamata `nova-network`
  - Da Grizzly in poi esiste un componente dedicato chiamato *Quantum*

Quantum (rete)

- In versioni precedenti a Folsom di OpenStack, la parte rete era gestita direttamente da nova, attraverso la parte `nova-network`
  - `nova-network` è tuttora supportato in Folsom
- A partire da Folsom, è disponibile il componente Quantum che implementa il concetto di "network connectivity as a service"
  - "Servizio" tra interfacce (per esempio interfacce virtuali) gestite da Nova
  - Come altri componenti in OpenStack, è composto da plug-in per la massima configurabilità. In particolare il caso d'uso è di poter creare delle topologie di rete complesse, come sistemi web multi-tier, usare L2-in-L3, QoS end-to-end, interconnessione di data center trasparente, etc.
  - Tuttavia questo avviene a fronte di una notevole potenziale complessità (e Quantum è un componente relativamente giovane)

Perché Quantum?

- Pre-Folsom, la rete è un sotto-componente di Nova
- Problemi:
  - Limitazioni dovute al modo in cui la rete è implementata in nova-network
  - Assenza di controllo della topologia di rete, impossibilità di gestire servizi avanzati di rete (ad esempio Dynamic Virtual Networks)
- Cf. http://goo.gl/dU59Y



VLANs are Great!
- Stone Age Man

Keystone (identità)

- Keystone implementa un unico componente di autenticazione per i diversi moduli di OpenStack. Fornisce autorizzazioni per credenziali di log-in multiple.
  - Identità: validazione delle credenziali per utenti, tenant e ruoli
  - Token: validazione e gestione dei token per l'autenticazione
  - Catalogo dei servizi
  - Gestione delle policy di autorizzazione
- Supporta diversi back-end come identity provider
  - Non ancora SAML in Folsom
  - E' comque prevista l'espansione per proxying verso servizi esterni come oAuth, SAML e openID
    - Lavoro di integrazione SAML in Keystone è stato fatto dall'Università di Camerino

Cinder (block storage)

- In versioni precedenti a Folsom di OpenStack, la parte di block storage era gestita direttamente da nova, attraverso la parte `nova-volume`
  - `nova-volume` è tuttora supportato in Folsom
- Si tratta di storage persistente
  - I volumi definiti possono essere collegati alle macchine virtuali
  - È possibile la definizione di tipi di volumi, snapshot, statistiche
  - Non viene garantita alta affidabilità, demandata al file system sottostante
  - Esistono diversi tipi di plug-in
- Come in nova, esiste uno scheduler (cinder-scheduler) per definire il posto ottimale in cui istanziare un volume di dati
- Come per Quantum, Cinder è un componente relativamente giovane

Swift (Object store)

- Swift implementa un distributed storage system simile a Amazon S3
  - Supporta direttamente le API S3 di Amazon, gestisce quote, controllo accessi e si interfaccia a diversi sistemi di storage
  - Non è:
    - Un file system (non si può montare sulle VM)
    - Pensato per memorizzare dei DB live o delle gerarchie di file
- È pensato per scalabilità, alta affidabilità e gestione di elevata concorrenza nelle transazioni
  - Es. il San Diego Supercomputing Center ha una installazione basata su Swift con oltre 5 PB di spazio disco (http://goo.gl/jmxJK)

Perché Swift

- Applicabile come storage solution a diversi livelli:
  - Service provider
  - Applicazioni
  - Imprese (es. per archivi online, backup)
- Cf. http://goo.gl/9AGH4

3 Use-Cases for OpenStack Swift

Da Folsom a Grizzly (1)

- Alcuni punti indicati qui. Cf. http://goo.gl/Gip5P per una lista completa.
- **Swift**:
  - Supporto a Static Large Objects (SLO) → upload di molti oggetti concorrentemente e in seguito download come singolo oggetto (l'utente definisce in un manifesto le caratteristiche degli oggetti)
- **Nova**:
  - Supporto di Celle (zone "geografiche" di disponibilità delle risorse; importante per partizionamento in gruppi logici per load balancing e distribuzione del carico)
  - Introduzione di una common Cloud API per il supporto di bare metal (senza l'hypervisor)
- **Horizon**:
  - Migrazione dalla dashboard
  - Upload di immagini in glance dalla dashboard
- **Keystone**:
  - Pluggable authentication (attualmente con qualche caveat)

Da Folsom a Grizzly (2)

- **Quantum**:
  - Security groups (in Folsom sostanzialmente si usano `iptables` in nova); importante ad esempio per packet filtering
  - IPv6 and IPv4 support
  - Load-balancing as a Service
- **Cinder**:
  - Block storage backup to Swift (importante per disaster recovery)

- Esistono inoltre diversi progetti non ancora ufficialmente inseriti nella roadmap ma potenzialmente molto importanti:
  - Ceilometer (http://goo.gl/aPOI8) per il supporto a sistemi di billing
  - Heat (http://goo.gl/7j8fv) per descrivere applicazioni cloud e loro dipendenze utilizzando la Amazon AWS Cloud Formation API

Installazione, alcuni riferimenti

- Tool per la valutazione (non per produzione!): **DevStack** (http://goo.gl/YjyNq) e in particolare cf. una guida per Grizzly (http://goo.gl/VZDLD).
- Per gli amanti della virtualizzazione estrema: cf. “OpenStack on OpenStack” (**nested virtualization** e interessanti configurazioni di rete), http://goo.gl/ECERj.
- Grizzly è disponibile **su Ubuntu** 13.04 e 12.04, cf. http://goo.gl/JENP8.
- RedHat (uno dei maggiori contributor a Grizzly) ha recentemente rilasciato **RDO** (OpenStack for RedHat Enterprise Linux, Fedora, etc.) – cf. http://goo.gl/e8Q1z.

Conclusione

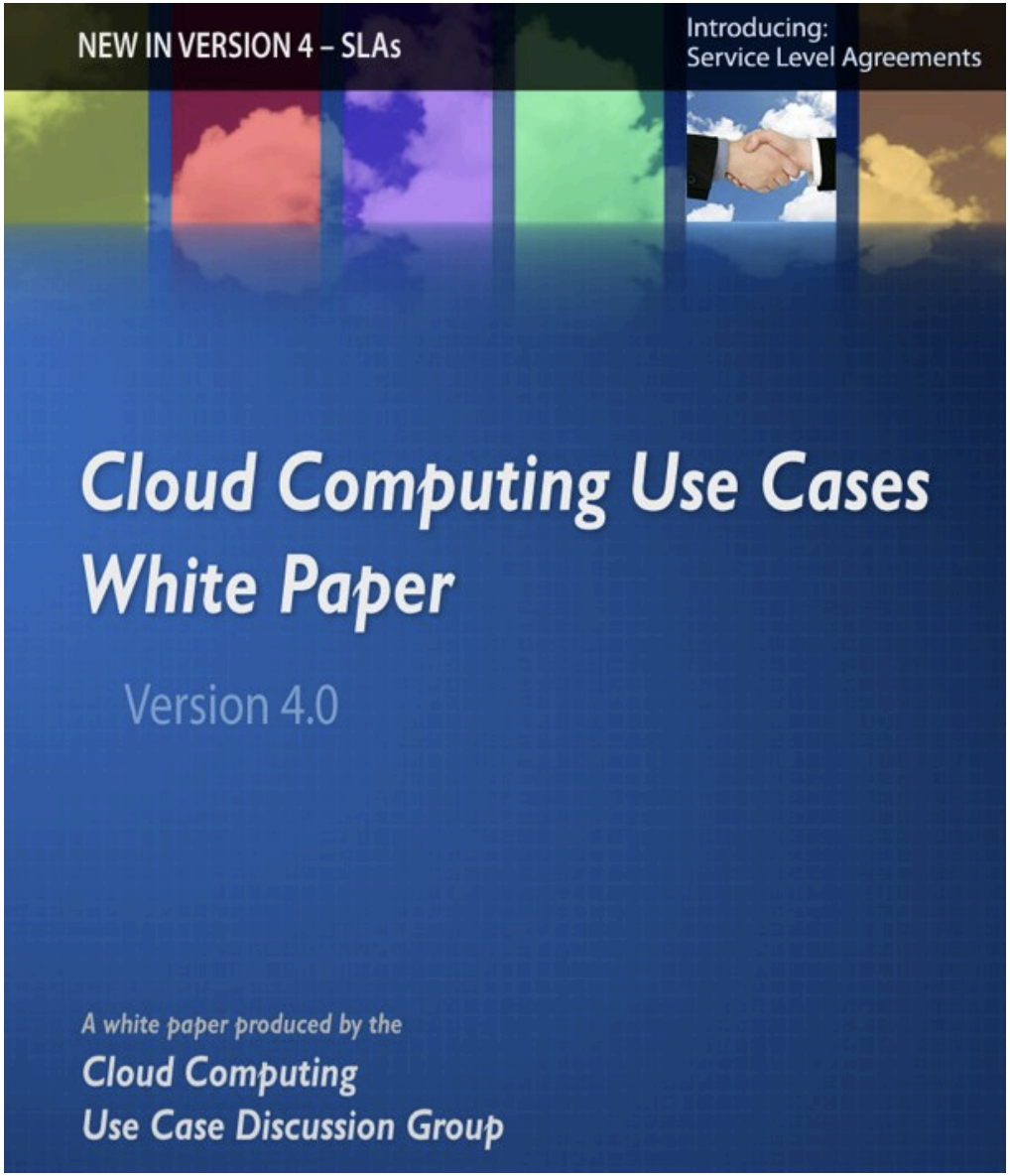
- OpenStack è un framework open source dall'architettura modulare, con sviluppo e supporto in forte crescita e può costituire una importante scelta strategica per il futuro.
- La ingegnerizzazione / configurazione di OpenStack è relativamente complessa a causa dei molti scenari applicativi supportati, a causa della "gioventù" del prodotto e a causa del fatto che a differenza di altri stack non tutto è gestibile da interfacce grafiche.
- OpenStack presenta diverse opportunità come base per la creazione e lo sviluppo di applicazioni avanzate grazie alle sue possibilità di adattamento
  - Questo è importante per progetti di reale Cloud Computing, più che di virtualizzazione di un data center
  - Può inoltre aprire importanti prospettive per PMI tecnologiche ad esempio legate al supporto e all'adattamento di OpenStack

Back-up Slides

Casi d'uso del Cloud

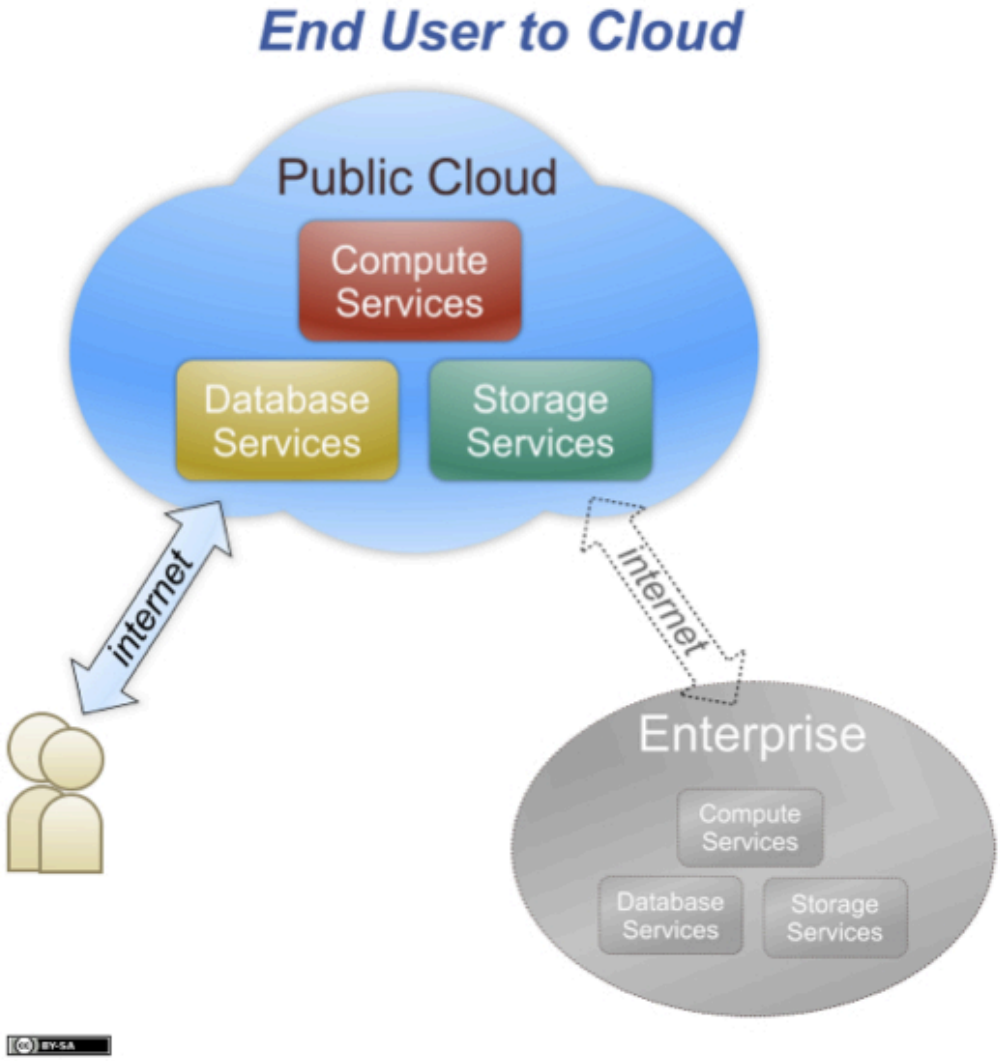
Un buon testo di riferimento

- http://goo.gl/1jrqK
- Sette casi principali:
  - Utente finale → Cloud
  - Impresa → Cloud → utente finale
  - Impresa → Cloud
  - Impresa → Cloud → impresa
  - Cloud privata
  - Cambiamento di fornitori Cloud
  - Cloud ibride



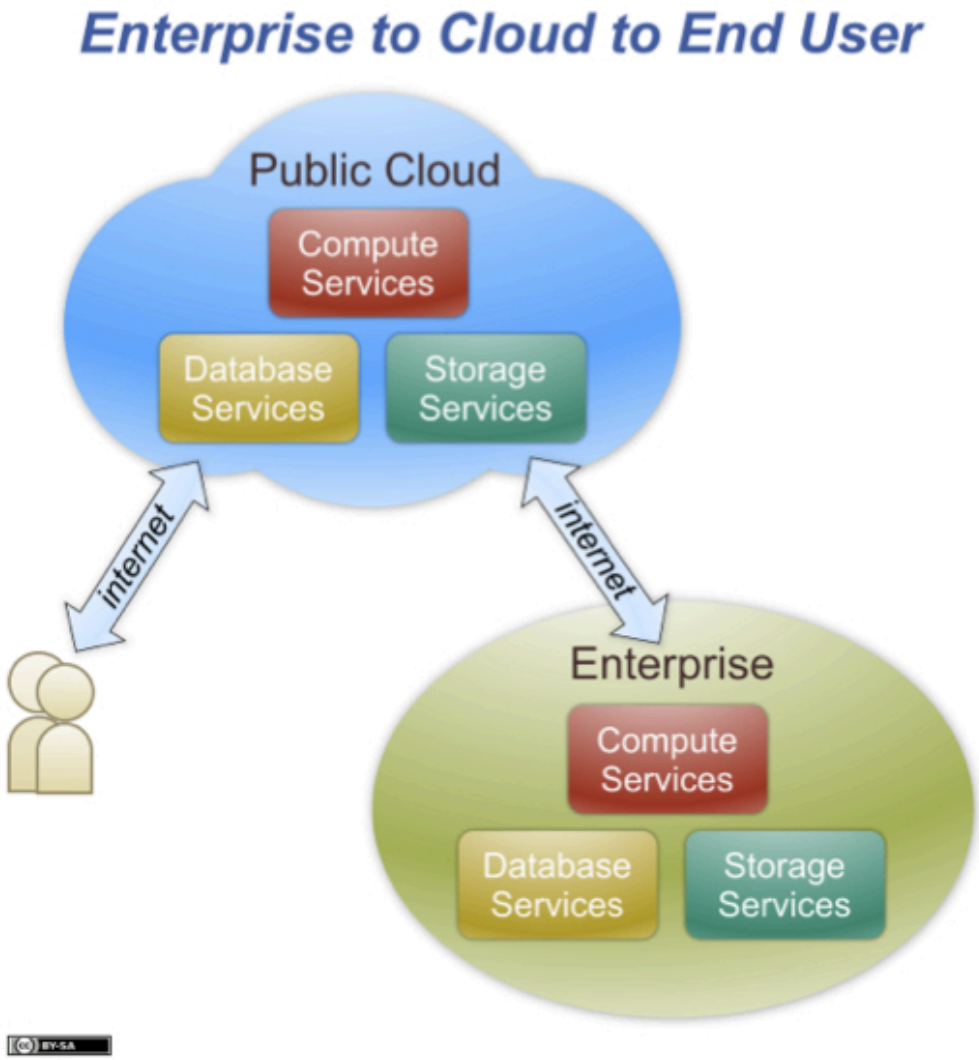
Utente → Cloud

- Accedo a dati o applicazioni nella Cloud (es. email, social networks)
- Punti chiave:
  - Identità
  - Open client
  - SLA non tanto importanti



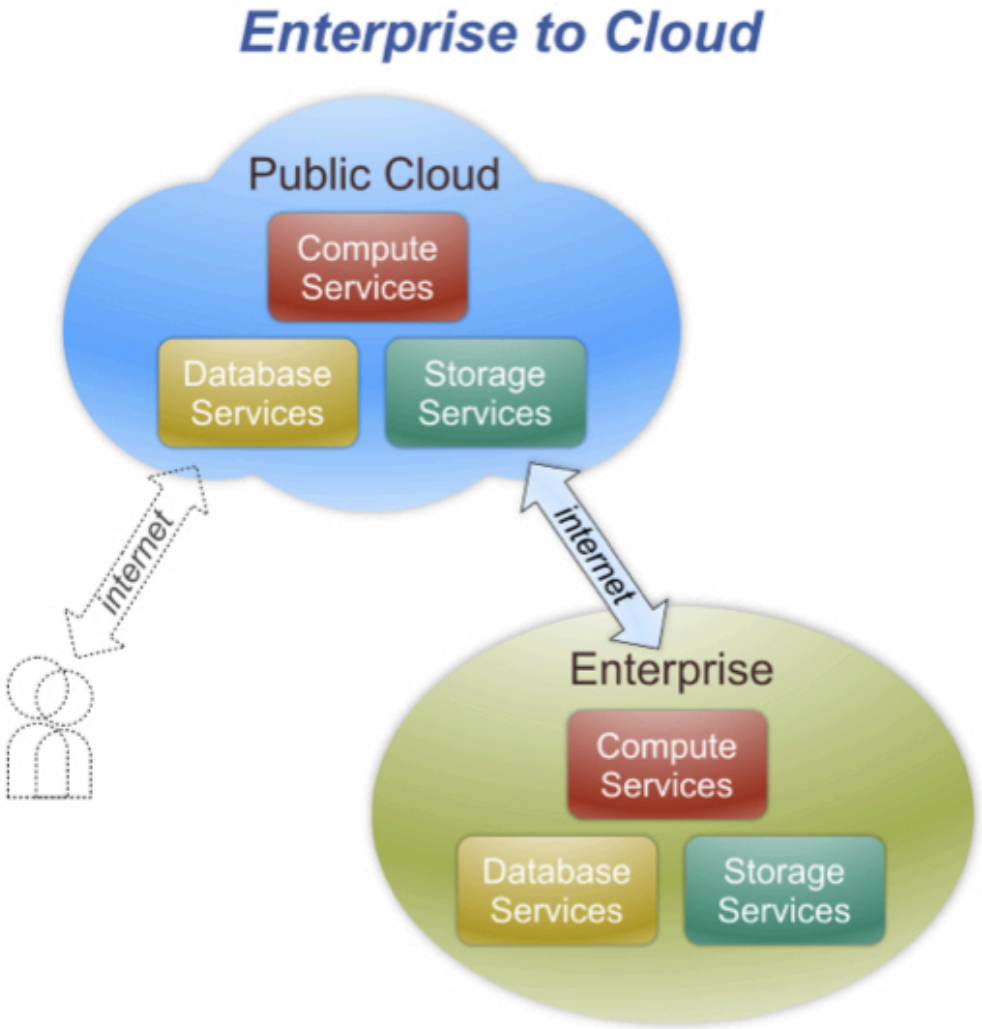
Impresa → Cloud → utente

- Un'impresa usa la Cloud per fornire servizi ad utenti
- Punti chiave: anche
  - Identità → spesso federata (es. OpenID)
  - Dislocazione dei dati (es. per questioni legali)
  - Controllo dei costi (monitoring, accounting)
  - Sicurezza
  - API comuni (per portabilità)
  - SLA
  - Dati / applicazioni federate
  - Data retention / data destruction



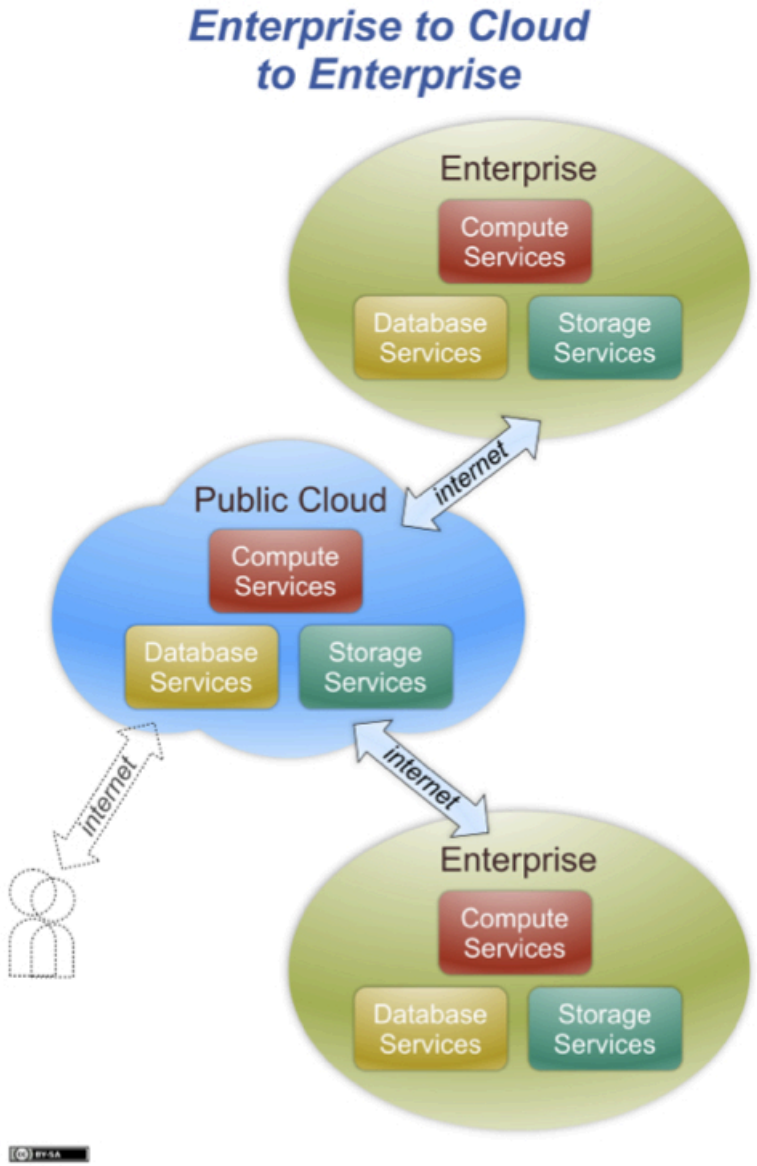
Impresa → Cloud

- Un'impresa usa la Cloud per i propri processi interni
- Punti chiave:
  - Supplemento di storage (es. per back-up)
  - "Cloud bursting", creazione di VM per periodi di picco
  - Uso della Cloud per alcune applicazioni (email, calendario, etc.)
  - Portabilità, uso di standard, possibilità di spostamento di VM e dati per evitare di legarsi a un unico fornitore



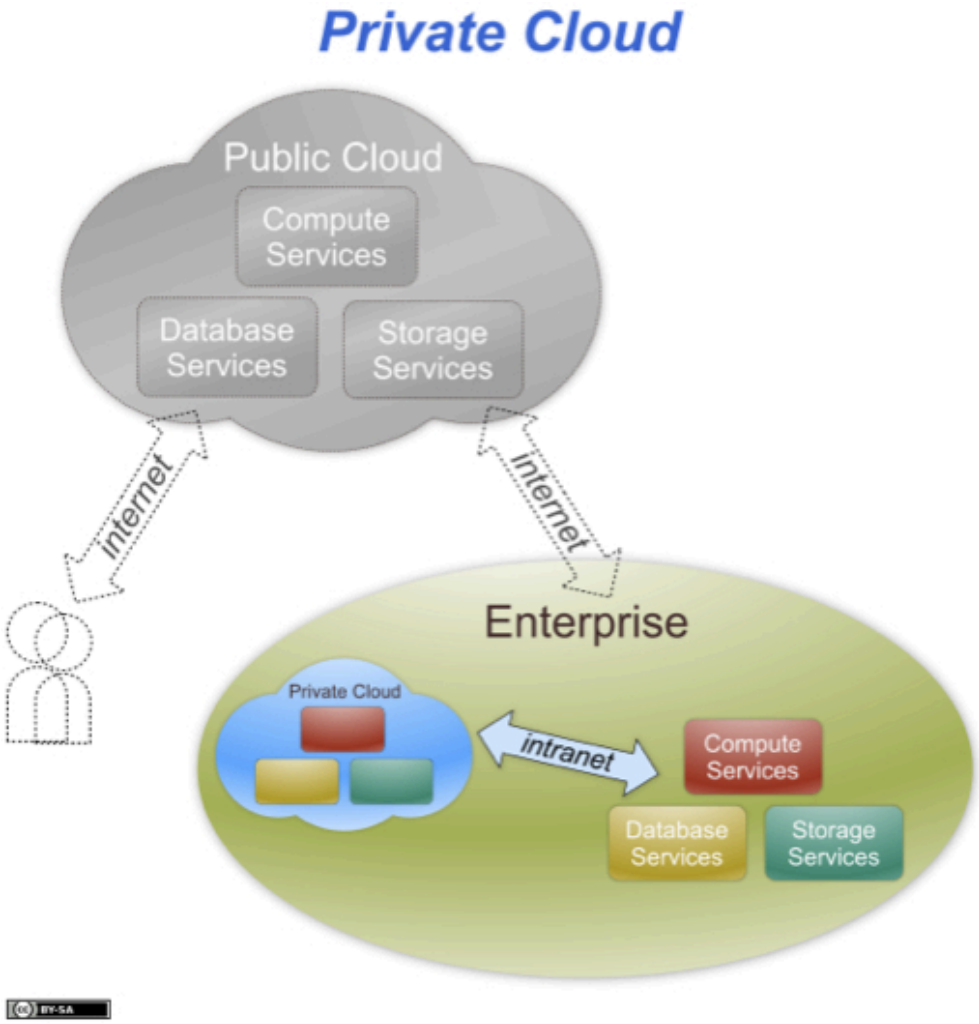
Impresa → Cloud → Impresa

- Due imprese utilizzano la stessa Cloud
- Punti chiave:
  - Le applicazioni nelle due imprese possono interoperare (esempio tipico è una catena di approvvigionamento)
  - Consistenza e concorrenza nelle transazioni
  - Interoperabilità attraverso uso di standard



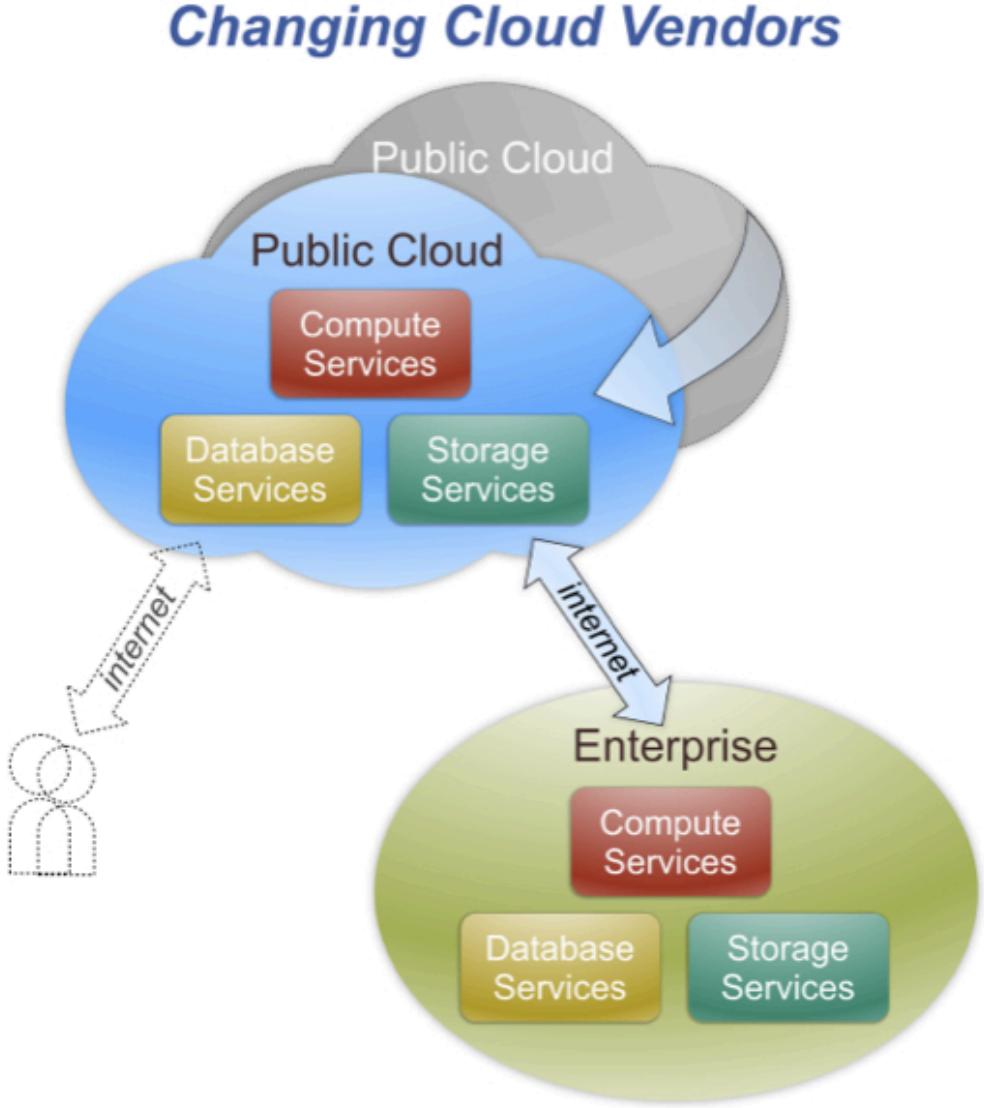
Cloud privata

- Una cloud all'interno di un'impresa



Cambiamento di fornitori

- Ad esempio si vuole cambiare un fornitore Cloud oppure aggiungerne un altro
- Punti chiave:
  - Ovviamente standardizzazione.



Cloud ibride

- Ad esempio uso di Cloud sia pubbliche che private
- Punto essenziale:
  - Per l'utente finale questo caso d'uso non deve essere diverso dal caso Utente → Cloud. L'utente finale ignora (vuole ignorare) i dettagli.
- Le Cloud di comunità possono essere considerate un sottoinsieme delle Cloud ibride