Computación distribuida

Sergio Nesmachnow
(sergion@fing.edu.uy)

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

UNIVERSIDAD
DE LA REPÚBLICA
URUGUAY

Cloud Computing

CLOUD COMPUTING: CONCEPTOS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Cloud Computing es un término genérico que describe un paradigma de computación distribuida sobre Internet.
- Extiende el concepto de *utility computing*, que se basa en el suministro de recursos computacionales (procesamiento y almacenamiento) como un servicio a demanda, similar a otros productos públicos tradicionales (electricidad, agua, gas natural, etc).
- El modelo que cuenta con la ventaja de tener un costo nulo (o muy bajo) de adquisición de hardware; en cambio, los recursos computacionales son esencialmente alquilados y tienen costo.
- Clientes que realizan procesamiento de datos a gran escala o que están frente a un pico de demanda también pueden evitar los atrasos que resultarían de adquirir y ensamblar físicamente una gran cantidad de computadoras.

CLOUD COMPUTING: una “definición”

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

¿Qué es Cloud computing?

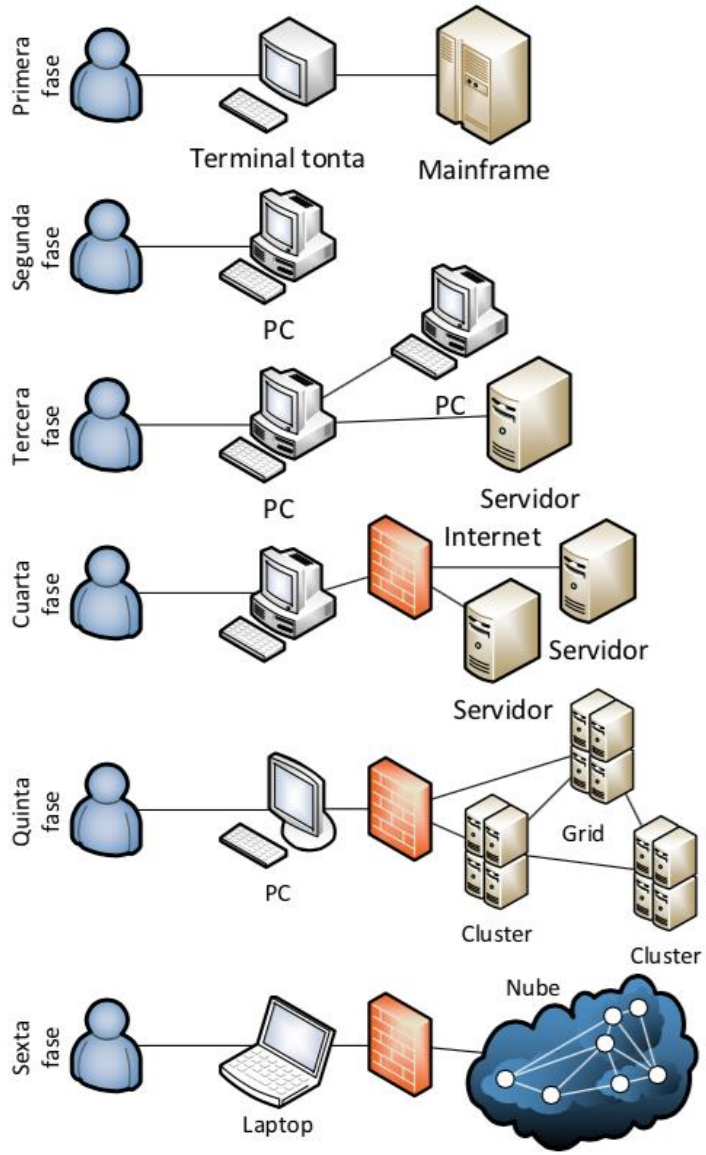
> “Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.”
>
> – National Institute of Standards and Technology, Information Technology Laboratory (2009).

CLOUD COMPUTING: CARACTERÍSTICAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Existe una colección de recursos de hardware integrados a través de una red, mediante un software específico, y con acceso a través de Internet (este conjunto integrado se denomina *plataforma*).
- Se utiliza Internet para las comunicaciones y para proveer servicios de hardware, software y conectividad (networking) a los clientes.
- Las plataformas ocultan la complejidad y los detalles de la infraestructura subyacente a los usuarios y proveen soporte para el desarrollo y ejecución de aplicaciones mediante simples interfaces gráficas y/o APIs.
- La plataforma provee servicios a demanda, que están siempre disponibles (ubicuos: anywhere, anytime, anyplace).
- Los usuarios pagan por la utilización de los servicios.
- La plataforma puede escalar elásticamente las capacidades y funcionalidades de los servicios provistos.

CLOUD COMPUTING: HISTORIA

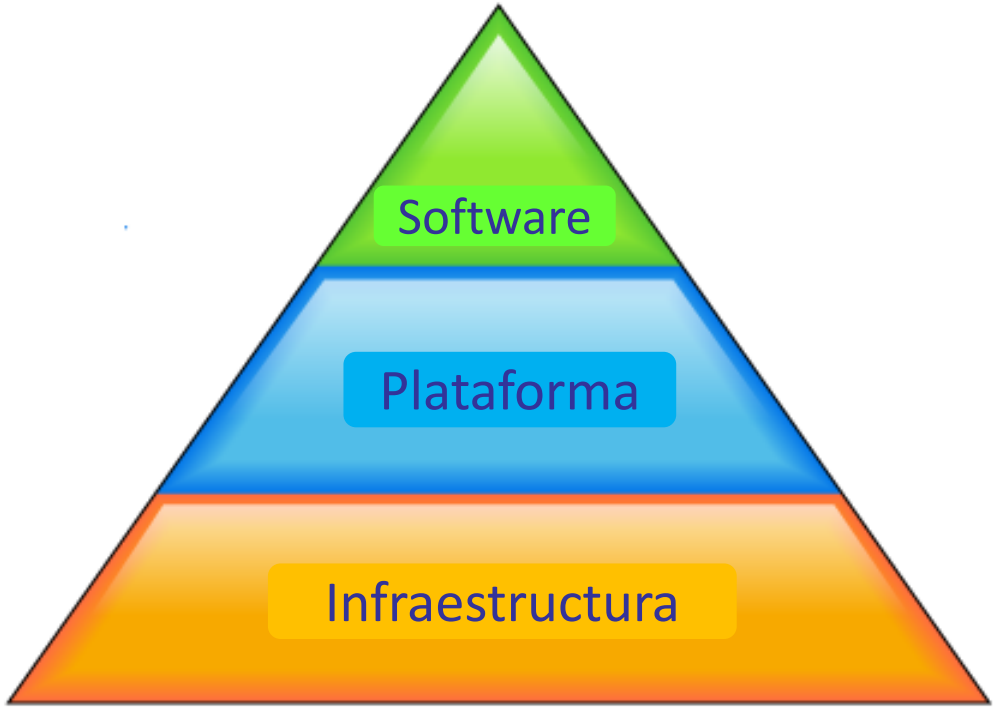


1. *Primera etapa*: Paradigma de computación centralizada basado en mainframes.
2. *Segunda etapa* : conexión de usuario a PC.
3. *Tercera etapa* : conexiones en red de área local.
4. *Cuarta etapa* : interconexión de computadoras en red a través de internet.
5. *Quinta etapa* : Grid y el uso compartido de procesamiento y cómputo.
6. *Sexta etapa* : Usuarios se conectan a la nube.

CLOUD COMPUTING: CAPAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Capas asociadas a los servicios
  - Infraestructura, plataforma y software



Clientes Cloud

Navegadores web, aplicaciones móviles, clientes livianos, emuladores de terminal, etc.

Software

SaaS

E-mail, desktop virtuales, comunicación y redes sociales, juegos, etc.

Plataforma

PaaS

Entorno de ejecución, bases de datos, servidor web, herramientas de desarrollo, etc.

Infraestructura

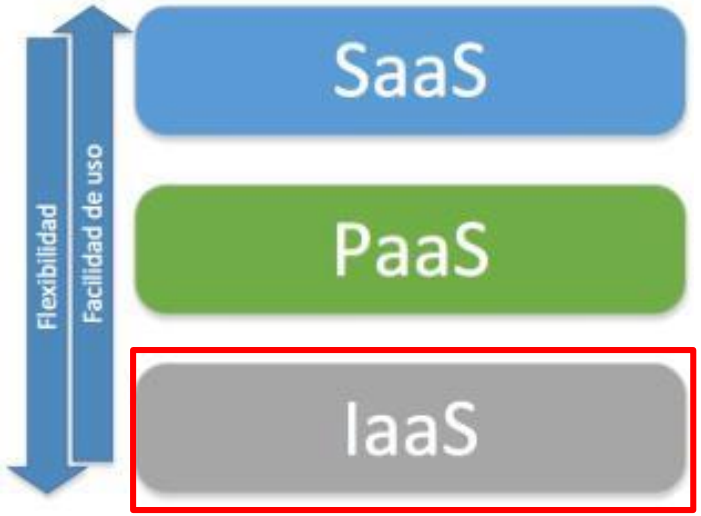
IaaS

Máquinas virtuales, servidores, almacenamiento, balance de cargas, red, etc.

CLOUD COMPUTING: CAPAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Infraestructura como servicio (Infrastructure as a Service, IaaS)
  - Capa inferior del cloud, es un medio de entregar almacenamiento y capacidades de cómputo como servicios estandarizados en la red.
  - Servidores, almacenamiento, conexiones, routers, y otros sistemas se concentran (por ejemplo a través de virtualización) para manejar tipos específicos de cargas de trabajo (procesamiento "batch", o aumento de servidor/ almacenamiento durante cargas pico).
  - Ejemplos comerciales de referencia: Amazon Web Services [servicios EC2 (cómputo) y S3 (almacenamiento)]; Azure y Joyent, que ofrecen hardware virtualizado a demanda, altamente escalable para diversas aplicaciones en múltiples lenguajes.



CLOUD COMPUTING: CAPAS

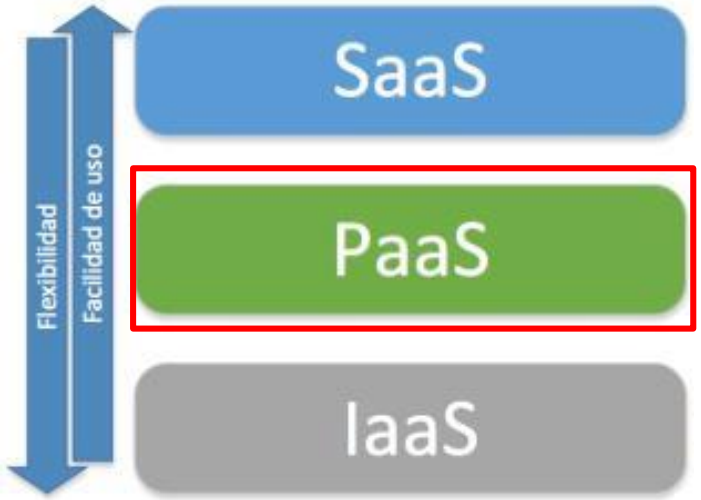
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Infraestructura como servicio (Infrastructure as a Service, IaaS)
  - Proveedores ofrecen máquinas virtuales (VMs), ejecutadas con un hipervisor (Xen, KVM, etc.).
  - Mediante un pool de hipervisores se puede manejar un gran número de VMs y brindar la capacidad de escalar (en ambos sentidos) de acuerdo a los requerimientos de los clientes de la infraestructura.
  - Otros servicios ofrecidos: bibliotecas de imágenes de discos para VMs, storage crudo y basado en archivos, firewalls, balance de cargas, direcciones IP, VLANs, y software preinstalado (bundles).
  - Los proveedores de IaaS brindan los recursos a demanda (on-demand) desde grandes pools instalados en data centers. Los clientes se conectan a trevés de Internet o de VPN dedicadas.
  - Para desarrollar sus aplicaciones, los clientes instalan (y luego mantienen) imágenes de SO y software de aplicaciones en la infraestructura cloud.
  - Costos reflejan el uso de recursos reservados y consumidos, *brokers* existen como agentes intermedios.

CLOUD COMPUTING: CAPAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Plataforma como servicio (Platform as a service, PaaS)
  - Encapsula una abstracción de un ambiente de desarrollo y empaqueta una serie de módulos que proporcionan una funcionalidad horizontal (persistencia de datos, autenticación, mensajería, etc.).
  - Ejemplo: un entorno conteniendo sistemas, componentes o APIs preconfiguradas y listas para integrarse sobre una tecnología concreta de desarrollo (servidor web en Linux y ambiente de programación Perl o Ruby).
  - PaaS puede dar servicio a todas las fases del ciclo de desarrollo de software, o especializarse en una particular (administración de contenidos).
  - Ejemplos comerciales: Google App Engine para aplicaciones Google, y Windows Azure para desarrollar aplicaciones en .NET, Java y PHP.
  - Los servicios PaaS otorgan flexibilidad, pero pueden estar restringidos por las capacidades del proveedor.



CLOUD COMPUTING: CAPAS

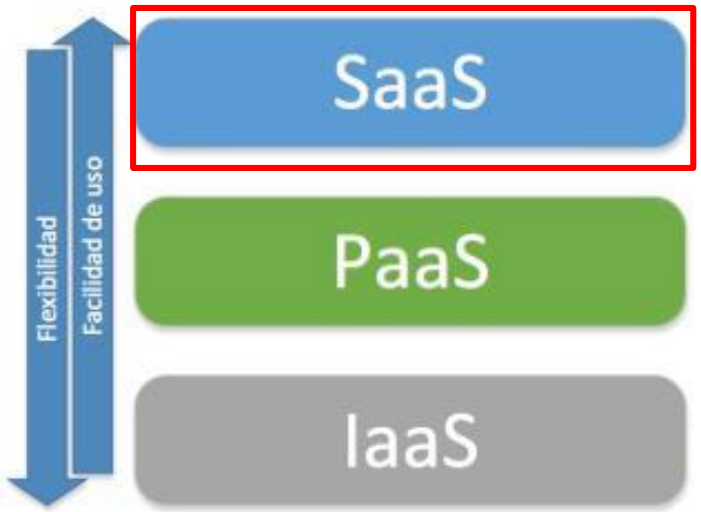
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Plataforma como servicio (Platform as a service, PaaS)
  - Varios tipos de proveedores, pero todos ofrecen hosting, un entorno de desarrollo de aplicaciones y servicios de escalabilidad y mantenimiento.
  - Tipos de servicios PaaS:
    - Desarrollo de adds-on y software personalizado: usando macrolenguajes de personalización de aplicaciones (Lotus Notes, Microsoft Word, etc.)
    - Entornos de desarrollo genéricos y stand-alone: no tienen dependencias técnicas, de licencia, o financieras con aplicaciones SaaS específicas.
    - Entornos de entrega de aplicaciones: no incluyen funcionalidades de desarrollo o testeo, que pueden proveerse fuera de línea. Los servicios en general se enfocan en seguridad y escalabilidad a demanda.
    - Plataforma abierta como servicio: no incluye hosting, pero provee software open source para ejecutar aplicaciones. Ejemplo: AppScale, que permite desplegar aplicaciones desarrolladas en Google App Engine en servidor del usuario. Otras plataformas dan más flexibilidad sobre lenguajes, bases de datos, sistemas operativos, etc.

CLOUD COMPUTING: CAPAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Software como servicio (Software as a Service, SaaS)
  - Aplicaciones completas ofrecidas como servicio a demanda, y en multitenencia: una única instancia del software ejecuta en la infraestructura del proveedor y sirve a múltiples organizaciones de clientes.
  - Ejemplo comercial de referencia: Salesforce.com, uno de los mejores ejemplos de cómputo en cloud durante muchos años, con sus aplicaciones para ventas (Sales Cloud), servicio al cliente (Service Cloud), colaboración empresarial (Chatter).
  - Otros ejemplos: Google Apps que ofrece servicios básicos de negocio (incluyendo e-mail, calendarios, documentos web, etc.), Microsoft Office 365, la plataforma MS Office como servicio, que incluye versiones online de la mayoría de las aplicaciones de la suite ofimática de Microsoft.

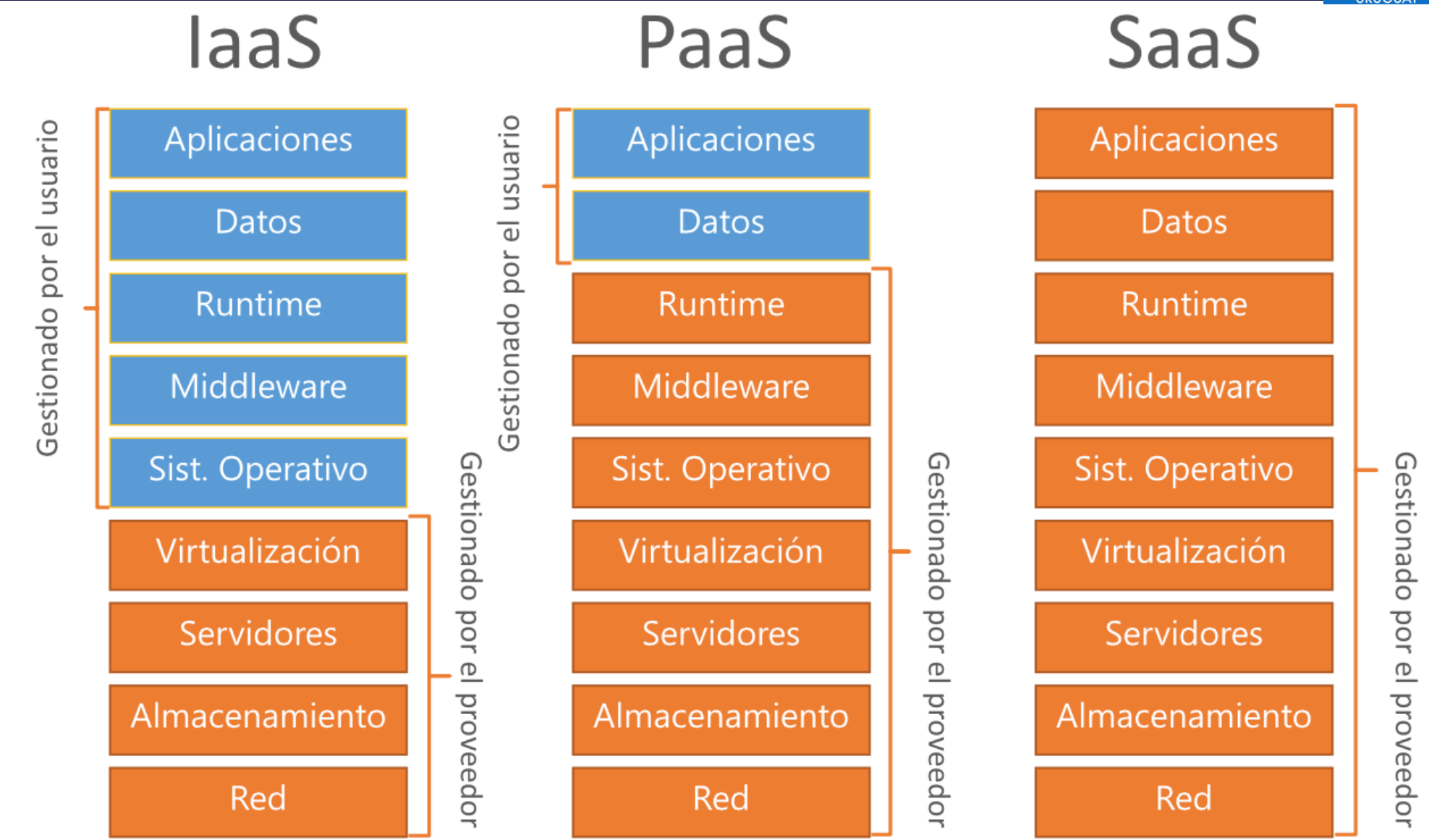


CLOUD COMPUTING: CAPAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Software como servicio (Software as a Service, SaaS)
  - Provee a los usuarios acceso a software de aplicación y bases de datos.
  - Los proveedores instalan, operan y mantienen (actualizan) el software, la plataforma de desarrollo y la infraestructura.
  - El proceso es totalmente transparente para el usuario, que no se ocupa de mantenimiento, soporte, o escalabilidad de las aplicaciones.
  - La multitenencia es vital para dar servicio a un gran número de usuarios.
  - El modelo de costos en general es por suscripciones (mensuales, anuales, etc.), con precios escalables si se agregan o eliminan en un período.
  - Es propuesto como un modelo de negocio con el potencial de reducir costos operativos al dejar el mantenimiento y soporte de hardware y software en el proveedor cloud.
  - Como contrapartida, una desventaja es que los datos de los usuarios deben residir (ser almacenados) en los servidores del proveedor cloud.

CLOUD COMPUTING: RESPONSABILIDADES



CLOUD COMPUTING: BENEFICIOS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Integración simple y rápida con otras aplicaciones (tradicionales o cloud), ya sean desarrolladas de manera interna o externa.
- Prestación de servicios con mayores capacidad de adaptación, recuperación de pérdida de datos y reducción de tiempos de inactividad.
- Permite prescindir de la instalación y administración de hardware, requiriendo una menor inversión inicial para un trabajo.
- Implementación rápida y con menos riesgos, incluso al requerir niveles considerables de personalización o integración.
- Actualizaciones automáticas de las aplicaciones, con poca (o ninguna) intervención del usuario.
- Contribuye al uso eficiente de la energía (requerida para el funcionamiento de la infraestructura), ofreciendo grandes reducciones de consumo al compararse con los datacenters tradicionales.

CLOUD COMPUTING: DESVENTAJAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- La centralización de las aplicaciones y el almacenamiento de los datos origina una interdependencia de los proveedores de servicios.
- La disponibilidad de las aplicaciones está sujeta al acceso a Internet.
- Datos "sensibles" residen fuera de sus propietarios; se puede generar un contexto de alta vulnerabilidad para el robo de información.
- La confiabilidad de los servicios depende de la robustez tecnológica y financiera de los proveedores. Servicios altamente especializados podrían tardar meses o años para que sean factibles de ser desplegados en la red.
- Muchas aplicaciones modifican continuamente sus interfaces, por lo cual el aprendizaje y su uso automático por otras aplicaciones es complejo.
- La información debe recorrer diferentes nodos para llegar a su destino; cada uno de ellos es un foco de inseguridad. Si se utilizan protocolos seguros (HTTPS u encriptado), la velocidad de transferencia disminuye debido a la sobrecarga.

CLOUD COMPUTING: DESVENTAJAS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- La escalabilidad a largo plazo es una incógnita. A medida que más usuarios empiecen a compartir la infraestructura de cloud, la sobrecarga en los servidores de los proveedores aumentará, y si la empresa no posee un esquema de crecimiento óptimo puede llevar a degradaciones en el servicio.
- Dependencia de proveedores para almacenamiento de datos y ejecución de aplicaciones.
  - Richard Stallman (2008): “cloud computing is simply a trap aimed at forcing more people to buy into locked, proprietary systems that would cost them more and more over time.” (“cloud computing es simplemente una trampa destinada a obligar a más gente a adquirir sistemas propietarios, bloqueados, que les costarán más y más conforme pase el tiempo.”)

CLOUD COMPUTING y HPC

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Cloud computing no fue concebido para brindar soporte a aplicaciones científicas que utilicen HPC.
- El paradigma de cloud computing se enfoca principalmente en soportar aplicaciones y servicios de propósito general.
- La infraestructura de hardware y la plataforma están optimizadas para aplicaciones basadas en transacciones de corta duración, e.g. peticiones de aplicaciones web.
- Este tipo de aplicaciones no requieren comunicación entre procesos.
- No se manejan servidores dedicados, los recursos de cómputo son instancias de cómputo virtualizadas.

CLOUD COMPUTING y HPC

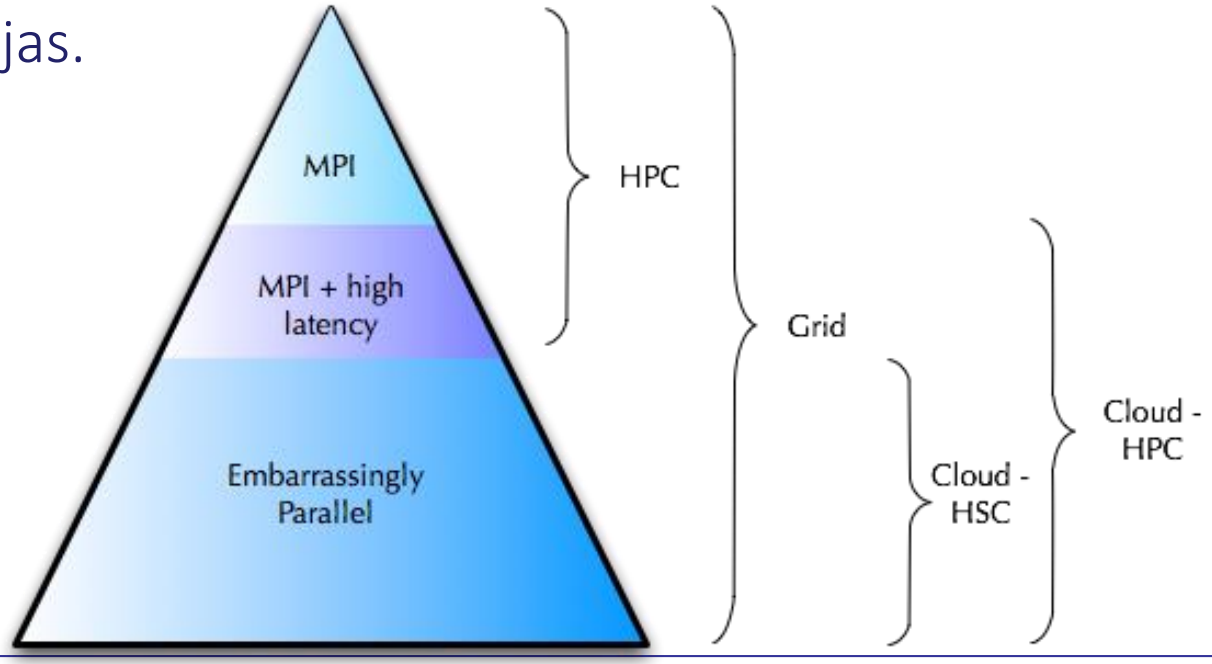
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Aunque cloud computing no es ni ha sido concebida para dar soporte a aplicaciones de HPC …
- Actualmente, el paradigma puede proveer el soporte para HPC.
- En general, sobre cloud pueden ejecutarse aplicaciones paralelas con esquema de particionamiento sencillo, y sin comunicaciones (embarrassingly parallel applications).
- Cloud computing provee un paradigma de computación diferente, denominado High Scalability Computing (HSC).
- Muchísimas (del orden de varios millones) unidades de procesamiento (virtuales), con poca capacidad, pero disponibles para usar a demanda.
- Por ejemplo: la unidad básica de Amazon EC2 (Amazon EC2 unit), tiene capacidad equivalente a una CPU correspondiente a un procesador Opteron o Xeon de 1.0-1.2 GHz, del año 2007.

CLOUD COMPUTING y HPC

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

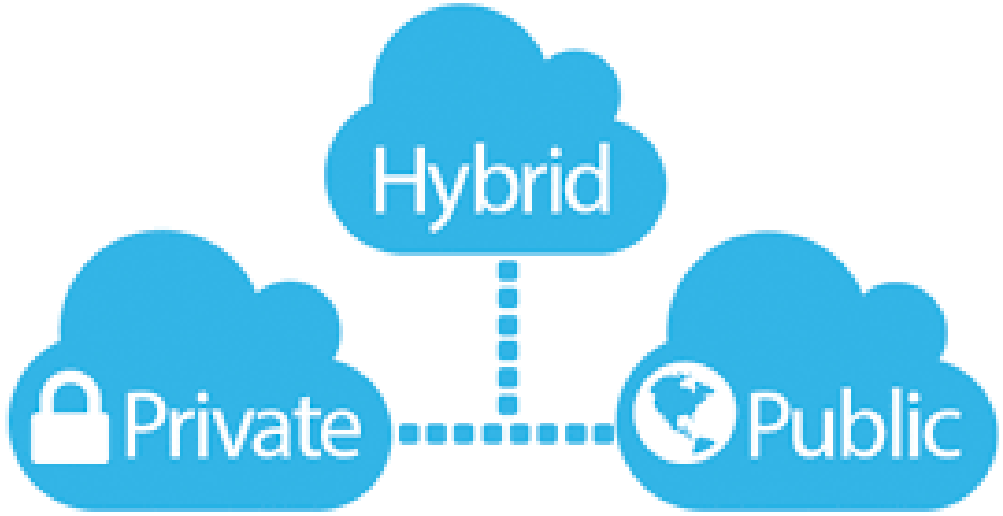
- Algunas infraestructuras cloud pueden dar soporte a aplicaciones previamente diseñadas para ejecutar en grid, o inclusive a programas MPI con requerimientos de comunicación bajos (altas latencias).
- Amazon EC2 ha mejorado su tecnología de comunicaciones, permitiendo la ejecución de aplicaciones basadas en pasaje de mensajes.
- Clouds con GPU o XeonPhi proveen poder de cómputo adicional para aplicaciones más complejas.



CLOUD COMPUTING y HPC

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Amazon Elastic Compute Cloud (EC2) ofrece instancias de cloud computing (IaaS) específicas para aplicaciones HPC y aplicaciones que requieren comunicación entre procesos.
- Características:
  - Desde 23 GB a 68 GB de memoria.
  - Desde 33.5 a 88 unidades de cómputo EC2. El desempeño de una unidad de cómputo EC2 equivale a un CPU Opteron o Xeon del 2007 con 1.0-1.2GHz.
  - SO Linux de 64bits y red 10 Gigabit Ethernet.
  - Desde 840 GB a 4 × 840 GB de almacenamiento.
  - Dos GPU NVIDIA Tesla M2050.
- Desarrollado por Amazon en conjunto con investigadores del Lawrence Berkeley National Laboratory.
- Costo a demanda: USD 2.10–2.40 hora (Linux, USA) y 2.60–3.00 (Windows, USA), o por reserva USD 2000 anuales (sin GPU).



UNIVERSIDAD DE LA REPÚBLICA URUGUAY

MODELOS para DESPLEGAR SISTEMAS CLOUD

CLOUD COMPUTING: MODELOS

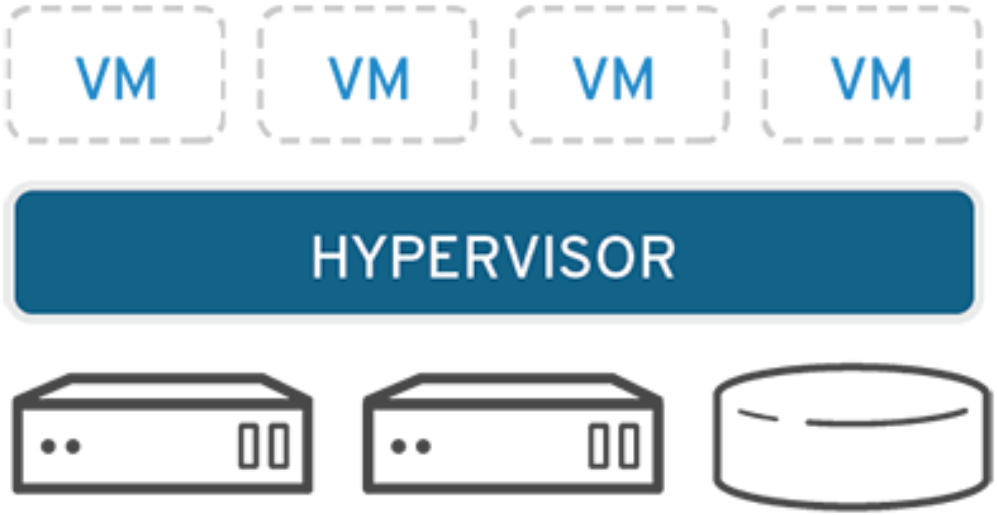
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Cloud privada: infraestructura operada por una única organización.
  - Mayor control sobre la infraestructura.
  - Mayor seguridad sobre los datos, el software y las comunicaciones.
  - Requiere inversión en recursos y esfuerzos de implementación.
  - Costos elevados, requerimientos de espacio, refrigeración, mantenimiento, actualización, etc. No se benefician de los aspectos más provechosos del modelo de cloud … "still have to buy, build, and manage, thus do not benefit from less hands-on management", … "[lacking] the economic model that makes cloud computing such an intriguing concept" (Gordon, 2009).
- Cloud pública: servicios ofrecidos a través de una red pública
  - Mayor escalabilidad y flexibilidad, usa datacenters accesibles via Internet
  - Aprovecha los beneficios del modelo de gestión no local.
  - Bajo (o nulo) control sobre la infraestructura.
  - Menor nivel de seguridad (datos, comunicaciones, etc.)

CLOUD COMPUTIN: MODELOS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Cloud híbrida: combina infraestructuras de cloud pública y privada. Intenta capturar las ventajas de ambos modelos
  - Los componentes permanecen como entidades separadas, pero conectadas para ofrecer diferentes opciones para desplegar aplicaciones.
  - Permite extender las capacidades de servicios cloud mediante agregación, integración o personalización con otros proveedores de servicios.
- Casos típicos de nubes híbridas:
  - Almacenar datos sensibles en nube local y utilizar servicios (inteligencia computacional, inteligencia de negocios) de la nube pública en modelo SaaS.
  - Cloud bursting: técnica para situaciones temporales como resolver picos de carga. Elasticidad y escalabilidad a demanda. El usuario/organización solamente paga por recursos adicionales cuando los necesita.
  - Se aplica sobre un modelo multiplatforma (cross-platform) con diferentes arquitecturas (x86-64, ARM, etc.) de modo transparente. Potenciados por sistemas en chip (system-on-chip) para servidores.



UNIVERSIDAD DE LA REPÚBLICA URUGUAY

TECNOLOGÍAS de VIRTUALIZACIÓN y SOFTWARE

TECNOLOGÍAS DE VIRTUALIZACIÓN Y SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- El paradigma de grid/cloud computing ha sido posible por la convergencia tecnológica en cuatro áreas:

  (1) la virtualización de hardware y los sistemas multinúcleo;

  (2) el desarrollo de los paradigmas de computación utilitaria (incluyendo modelos previos en cluster y grid);

  (3) el desarrollo de sistemas para programar aplicaciones distribuidas; y

  (4) la automatización de los datacenters.

TECNOLOGÍAS DE VIRTUALIZACIÓN Y SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Virtualizacion de datacenters para cloud computing
  - Las plataformas de cloud se construyen con hardware tradicional (arquitectura x86 o ARM) e interconexiones de red Gigabit Ethernet.
  - El diseño de datacenters enfatiza la relación entre costo y desempeño de los servicios. La performance no es el único objetivo de diseño, otros factores también juegan un rol importante (costo de dispositivos de almacenamiento, eficiencia energética, etc.).
  - Un datacenter estándar cuenta con unos mil servidores. Los costos se reparten en: 30% para la compra de nuevo equipamiento, 33% para el sistema de refrigeración, 18% en las fuente redundantes de energía (UPS), 9% en aire acondicionado y 7% en electricidad, iluminación, etc.
  - Para economizar se utilizan componentes de hardware de bajo costo, sistemas operativos, software de base y middleware de software libre o con muy bajos costos de licenciamiento.
  - Para lidiar con los elevados costos energéticos, se utilizan técnicas de planificación considerando eficiencia energética (regulación de voltaje, apagado de servidores, etc.).

TECNOLOGÍAS DE VIRTUALIZACIÓN Y SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Han posibilitado la construcción de plataformas para computación paralela y distribuida.

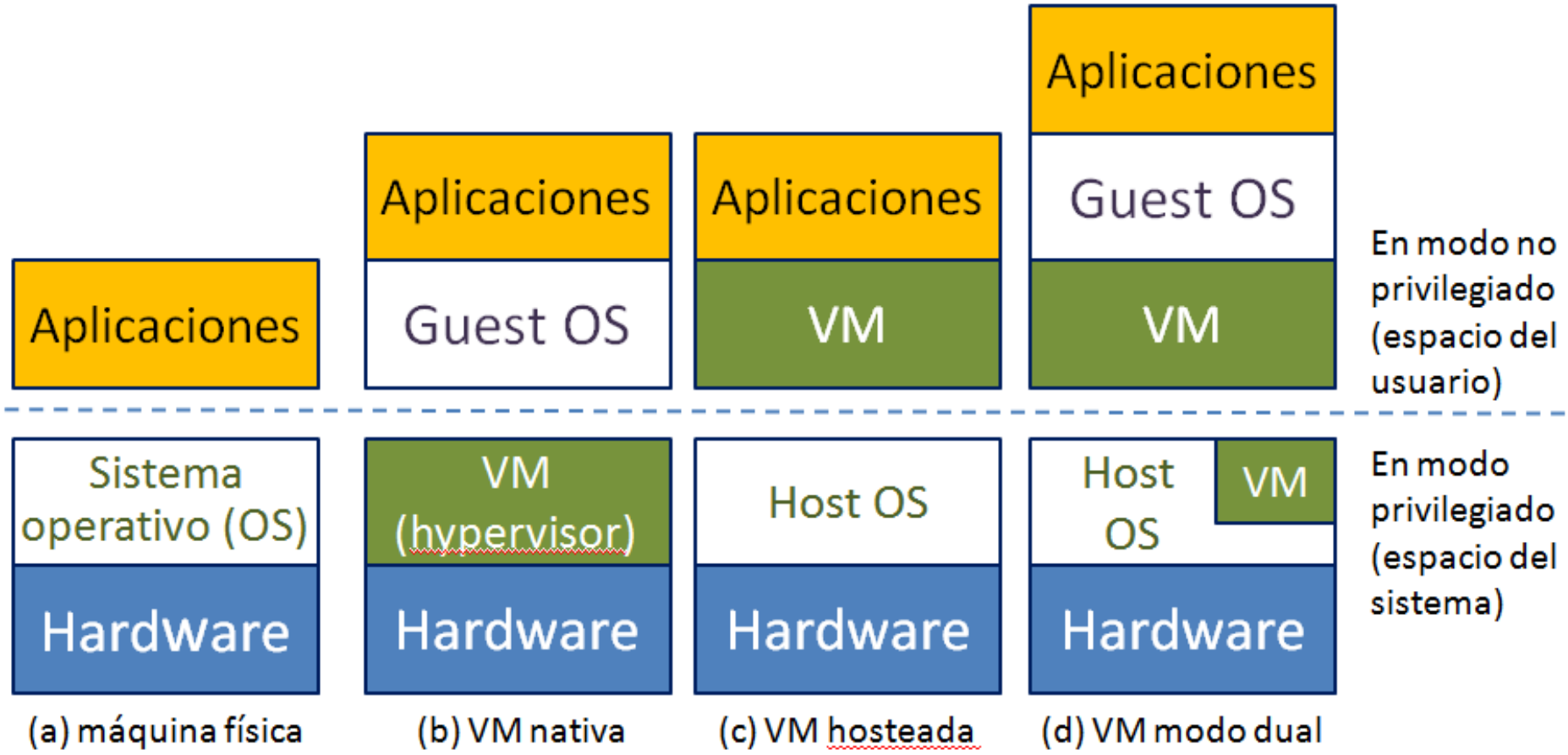
- *Máquinas virtuales y middleware para virtualización*
  - Un computador tradicional tiene una única imagen de sistema operativo, y por este motivo constituye un sistema rígido que acopla muy fuertemente a las aplicaciones con el hardware subyacente.
  - El paradigma de máquinas virtuales (Virtual machines, VMs) es una alternativa que provee flexibilidad para la utilización de recursos computacionales, aprovechando recursos de procesamiento ocioso a la vez que proporciona mayores niveles de manipulación de hardware y software y mejora la seguridad de las aplicaciones.

TECNOLOGÍAS DE VIRTUALIZACIÓN Y SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Soluciones de virtualización
  - Tradicional: las aplicaciones se ejecutan sobre un sistema operativo que gestiona a la propia máquina física.
  - VM nativa: se configuran VMs mediante un monitor (hypervisor), que es un middleware para gestión de las VMs ejecutando en modo privilegiado. Cada VM ejecuta un sistema operativo *guest*, potencialmente diferente al sistema operativo del host físico, y sobre este guest OS se ejecutan las aplicaciones de usuario.
  - VM hosteada: la máquina virtual ejecuta sobre el sistema operativo del equipo físico, en modo no privilegiado, sin requerir modificaciones al sistema operativo de base.
  - Modo dual: permite tener al gestor de máquinas virtuales en modo protegido y parte ejecutando en modo de usuario. Esta solución puede requerir modificar el sistema operativo de base.
- Estas estrategias proporcionan flexibilidad y portabilidad para las VMs, que pueden ejecutar de modo independiente al hardware subyacente.

TECNOLOGÍAS DE VIRTUALIZACIÓN Y SOFTWARE

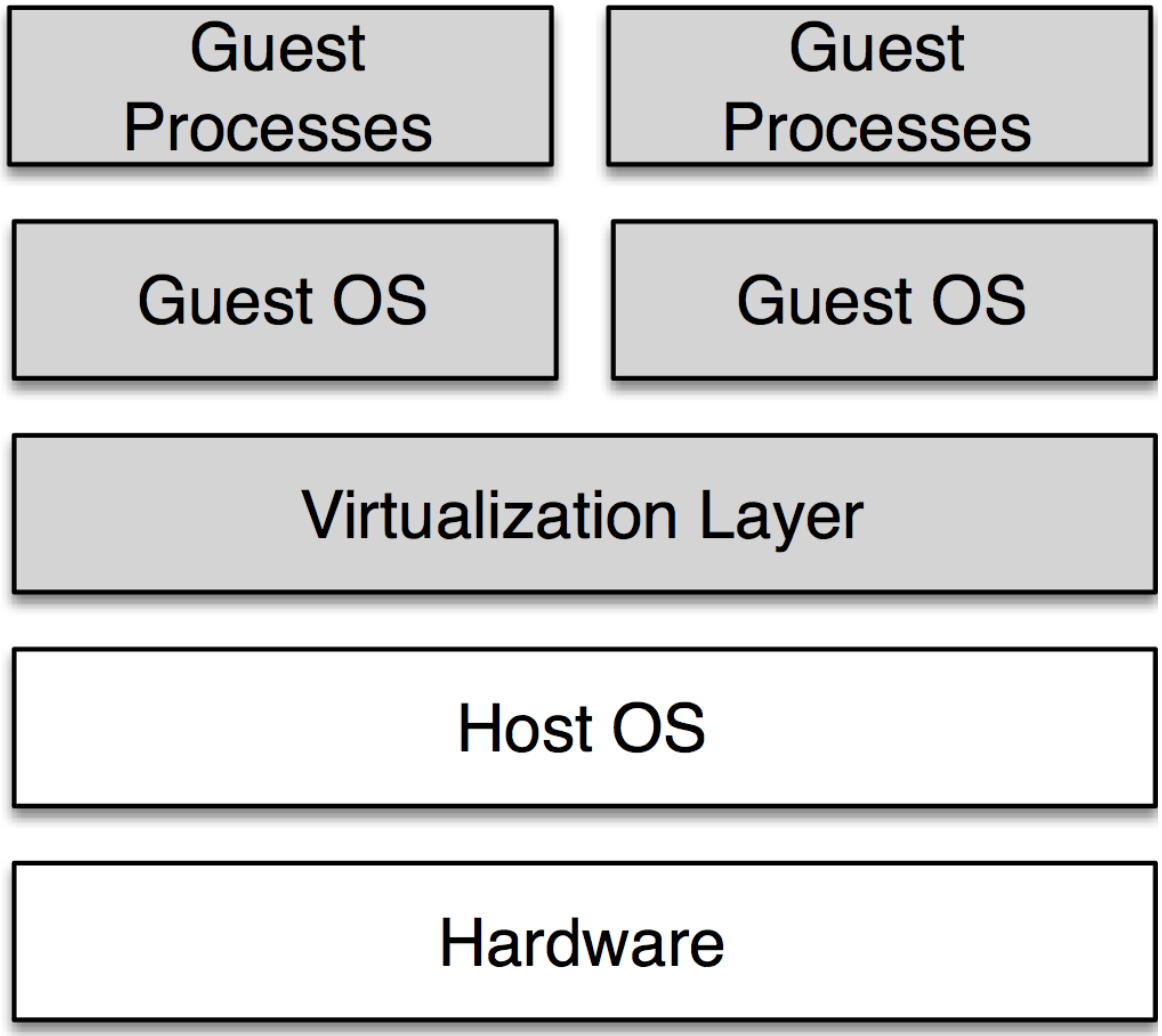


VIRTUALIZACIÓN BASADA en HIPERVISORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Soluciones basadas en hypervisores
  - Es una implementación del modelo de VM nativa que consiste en un sistema operativo guest trabajando sobre un sistema operativo de host que provee una abstracción total de la máquina virtual.
  - Cada máquina virtual tiene su propio OS que ejecuta completamente aislado del resto de las VMs.
  - Los beneficios del enfoque incluyen:
    - independencia del hardware.
    - aislamiento de las VMs.
    - posibilidad de ejecutar múltiples imágenes de OS en un único host.
  - Combinado con estrategias de puntos de control (checkpointing) y migración de procesos, las VMs pueden ser desplegadas y redesplegadas de acuerdo a las necesidades de los usuarios y la carga de las máquinas físicas.
  - Son populares actualmente para sistemas livianos y aplicaciones de usuario
  - Hypervisores más utilizados: XEN, VMWare, KVM

VIRTUALIZACIÓN BASADA en HIPERVISORES



XEN
VMWare
KVM

Hypervisor-based Virtualization

VIRTUALIZACIÓN BASADA en HIPERVISORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Soluciones basadas en hipervisores
  - Desventajas:
    - *Incrementan considerablemente la carga* de la infraestructura de base.
    - La abstracción completa de cada VM impone importantes overhead de procesamiento y manejo, que puede degradar notoriamente el desempeño, en especial para aplicaciones intensivas en entrada/salida y en comunicación de datos a través de la red.
    - El costo de alojar un OS completo por sobre del OS de base limita el número de instancias de VM que se pueden ejecutar en un único server, por los requerimientos de recursos (disco, memoria, CPU) del OS guest y de las aplicacíones que ejecutarán sobre él.
      - Aplicaciones usualmente demandan memoria y CPU, y posiblemente almacenamiento en el caso de procesar grandes volúmenes de datos.
      - OS guest demanda disco, memoria y CPU.
  - No son una alternativa recomendable para soluciones complejas o que requieren alta escalabilidad.

VIRTUALIZACIÓN BASADA en CONTENEDORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

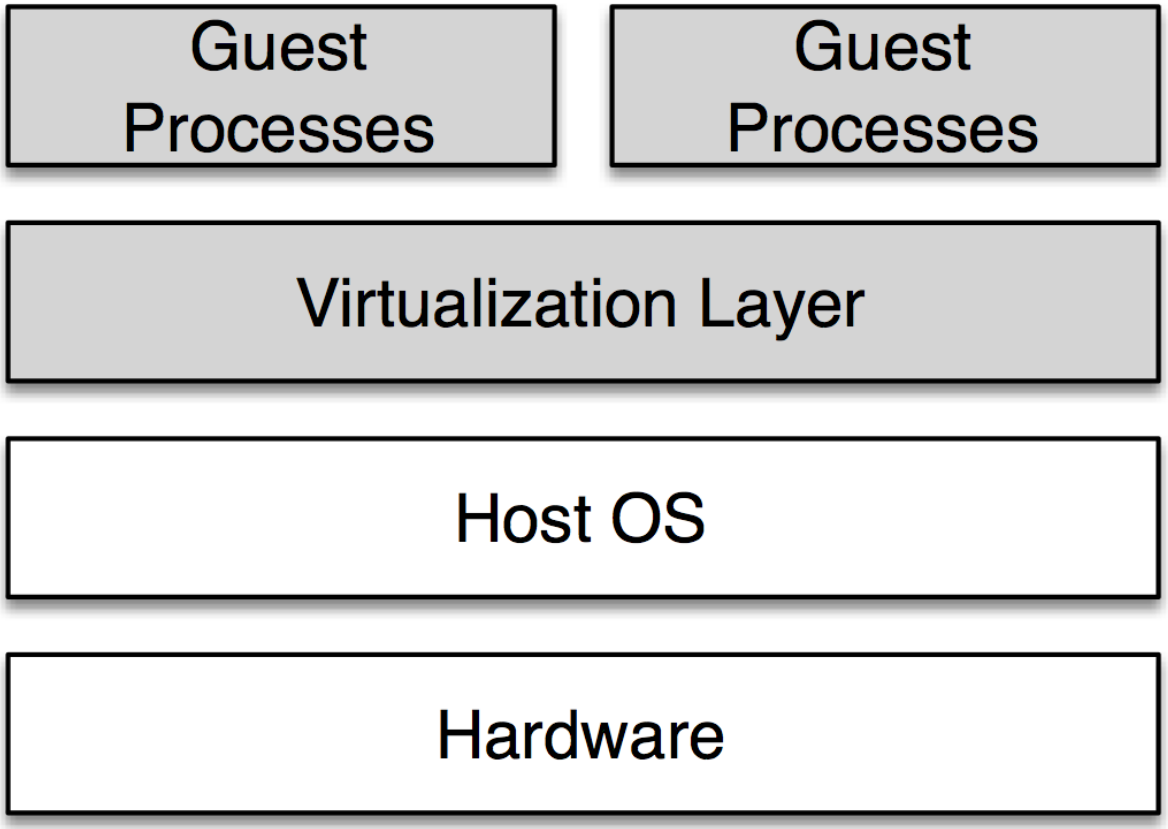
- Soluciones basadas en containers
  - Es una implementación del modelo de VM hosteada.
  - El enfoque se basa en las funcionalidades de los OS de particionar los recursos de la máquina física, creando múltiples instancias de espacios de usuario aislados (*containers*) sobre un único kernel del host.
  - Los containers trabajan a nivel del sistema OS y todas las VMs comparten un único kernel de base.
  - Proporciona menor nivel de aislamiento que las soluciones basadas en hypervisores, pero se han realizado avances recientes en mejorar los niveles de aislamiento para los containers.
  - Las primeras herramientas para containers (e.g. chroot) no proveían suficiente seguridad y aislamiento de recursos, pero actualmente existen opciones que implementan desde sandboxes básicos para procesos individuales a soluciones para sistemas completos.

VIRTUALIZACIÓN BASADA en CONTENEDORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Soluciones basadas en containers
  - El soporte para containers depende los servicios del Sistema operativo del host (BSD jails, Solaris Zones, Linux namespaces/cgroups), pero en general se basan en dos métodos:
    1. un mecanismo de namespaces que permite a diferentes procesos tener diferentes vistas del sistema.
    2. un método de manejo de recursos que restringe el uso de memoria, CPU e I/O para los diferentes containers.
  - En Linux:
    - Linux-Vserver, OpenVZ, Google Lmctfy y LXC.
    - Todos proveen manejo de bajo nivel de los containers, virtualización y manejo de namespaces y aislamiento.

VIRTUALIZACIÓN BASADA en CONTENEDORES



Container-based Virtualization

V-Server
OpenVZ
Lmctfy
LXC

VIRTUALIZACIÓN BASADA en CONTENEDORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Linux-VServer
  - Solución clásica de Linux, basada en herramientas clasicas (chroot, rlimit) para manejo básico de containers (aislamiento, manejo de recursos) y un patch del kernel para proveer aislamiento de procesos, uso de CPU y red.
  - Usa un espacio global de identificadores de proceso (PID) que maneja las vistas de cada proceso y limita las comunicaciones con procesos fuera del grupo asociado al container.
  - Como desventajas, VServer no permite migración, checkpointing o resumir containers, ya que Linux no admite relanzar procesos con el mismo PID.
- OpenVZ y LXC
  - Se basan en nuevas funcionalidades del kernel (namespaces y cgroups)
  - Namespaces para aislar los containers y garantizar acceso a su subconjunto de recursos. Cada namespace tiene su propio espacio de PID space, permitiendo checkpointed, migración y resumir procesos y containers.
  - También se usan para controlar las capacidades de red e IPC

VIRTUALIZACIÓN BASADA en CONTENEDORES

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- OpenVZ y LXC
  - Utilizan diferentes mecanismos de gestión de recursos
  - OpenVZ implementa su propio componente de gestión (User Beancounters), que controla el scheduling, quotas de disco, etc.
  - LXC utiliza cgroups de Linux, configurando namespaces de red y limitando uso de CPU y recursos.

- Son sistemas de bajo nivel que han sido extendidos para proveer funcionalidades más avanzadas (desplegar containers, migración avanzada)
  - Sobre OpenVZ: sistema de virtualización commercial Parallels Cloud Server.
  - Sobre LXC: Docker, manejador open-source para containers desarrollado por el proveedor PaaS dotCloud, incluye lógica para reducir el overhead de creación y administración de nuevas VMs.

ENTORNOS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Arquitectura orientada a servicios (SOA)
  - SOA es un paradigma de arquitectura de software aplicable al diseño e implementación de sistemas distribuidos.
  - Las principales funcionalidades del paradigma se relacionan con proveer facilidad y flexibilidad de integración con otros sistemas (en especial sistemas legados), permitiendo reducir los costos de implementación y lograr una adaptación ágil ante cambios en los procesos de negocio.
  - SOA define un conjunto de capas de software orientadas a proveer un entorno para el desarrollo de aplicaciones escalables, basadas en el paradigma de exposición e invocación de servicios (en general, servicios web), simplificando la interacción entre diferentes sistemas

ENTORNOS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Capas de SOA:
  - *Capa de aplicaciones básicas*, que incluye sistemas desarrollados bajo cualquier arquitectura o tecnología, geográficamente dispersos y bajo cualquier figura de propiedad;
  - *Capa de exposición de funcionalidades*, que expone funcionalidades de la capa de aplicaciones en forma de servicios (generalmente servicios web);
  - *Capa de integración de servicios*, que facilitan el intercambio de datos entre aplicaciones para procesos internos o en colaboración;
  - *Capa de composición de procesos*, que permite definir cada proceso en términos de las características de la lógica de la aplicación y sus necesidades, y que varían en función del problema a resolver;
  - *Capa de entrega*, donde los servicios son desplegados a los usuarios finales.
- El paradigma de SOA se aplica a la construcción de grids, clouds, grids de clouds, clouds de grids, clouds de clouds (conocidos como interclouds), y sistemas de sistemas en forma general.

ENTORNOS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Servicios web (web services)
  - Tecnología para la comunicación de datos entre aplicaciones distribuidas utilizando protocolos y estándares diseñados para tal fin. El modelo permite comunicar e interoperar a aplicaciones desarrolladas en diferentes lenguajes de programación y/o que ejecutan sobre diferentes plataformas de hardware y software. Como medio de comunicación se utilizan redes globales, en general Internet.
  - La interoperabilidad se consigue adoptando estándares abiertos, por ejemplo especificando completamente todas las características del servicio y su entorno mediante comunicaciones utilizando el Protocolo Simple de Acceso a Objetos (Simple Object Access Protocol, SOAP).
  - El entorno de aplicación toma las características de un *sistema operativo distribuido universal*, con capacidades totalmente distribuidas y comunicación con mensajes. El enfoque es poderoso, pero los resultados están condicionados a los acuerdos en partes clave del protocolo, que puede ser complejo de implementar por los softwares disponibles.

ENTORNOS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Representational State Transfer (REST)
  - REST es una arquitectura de software para computación distribuida sobre el protocolo HTTP, que ha sido exitosamente aplicada en los últimos años.
  - Adopta la simplicidad como principio universal para la comunicación y delega la mayor parte de los componentes complejos a la propia aplicación.
  - REST utiliza información mínima en el cabezal de los mensajes y es el cuerpo del mensaje quien transporta toda la información necesaria para la comunicación.
  - Como consecuencia de su simplicidad y agilidad, la arquitectura REST es más apropiada para entornos de tecnología "rápidos" sobre infraestructuras livianas, que permiten desarrollar servicios con un herramientas de desarrollo minimales.
  - Se obtiene así un paradigma de desarrollo ágil, escalable y que requiere de poco esfuerzo de desarrollo.
  - REST es una alternativa eficiente a SOAP, que provee una gran integración con HTTP, utilizándose en general un diseño de "XML sobre HTTP".

ENTORNOS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- CORBA y Java RMI
  - Common Object Request Broker Architecture (CORBA) es un estándar para la cooperación entre diversos componentes de software (escritos en múltiples lenguajes de programación) para facilitar el desarrollo de aplicaciones distribuidas en entornos heterogéneos.
  - En CORBA y Java, las entidades que participan en un sistema distribuido están relacionadas entre sí mediante invocación de procedimientos remotos (RPC) y la manera más simple de construir sistemas que combinan aplicaciones es ver a las entidades como objetos distribuidos.
  - En Java, este mecanismo se implementa mediante un programa tradicional donde las invocaciones a métodos están reemplazadas por invocaciones a métodos remotos (RMI).
  - CORBA soporta un modelo similar, con una sintaxis que refleja el estilo de interfaces entre entidades (objetos) de C++.

HERRAMIENTAS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Tres ejemplos de software de programación y gestión para sistemas cluster, grid y cloud de computación distribuida

- *Clusters: la biblioteca Message Passing Interface (MPI)*
  - MPI es una biblioteca de desarrollo de sistemas paralelos y distribuidos basada en el paradigma de pasaje de mensajes para la comunicación entre procesos.
  - Fue definido a mediados de la década de 1990 como un estándar para el desarrollo de este tipo de aplicaciones.
  - Aunque MPI fue originalmente concebido para el diseño de aplicaciones paralelas eficientes en infraestructuras cluster, es posible utilizarlo para combinar clusters, grids y sistemas P2P con servicios web mejorados y aplicaciones de computación utilitaria.

HERRAMIENTAS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Grid: Open Grid Services Architecture (OGSA) y Globus*
  - OGSA es un estándar para el uso público y general de servicios grid, definido para sacar el mayor provecho en la ejecución de sistemas distribuidos genéricos y aplicaciones de gran escala, que hacen uso intensivo de recursos y comparten grandes volúmenes de datos.
  - Las principales características de OGSA incluyen un entorno para la ejecución de aplicaciones distribuidas, servicios de autenticación de clave pública utilizando autoridades de certificación, manejo de confiabilidad y políticas de seguridad en el grid.
  - Globus es un middleware para grid (también se utiliza en sistemas cluster) que implementa el estándar OGSA para el manejo de recursos distribuidos. Globus provee componentes y APIs para descubrimiento de recursos, asignación de tareas a recursos y mecanismos para garantizar la seguridad mediante autenticación PKI multisitio, entre otras funcionalidades.

HERRAMIENTAS DE SOFTWARE

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Cloud: MapReduce y Hadoop*
  - MapReduce es un modelo de programación genérico, muy adaptable el procesamiento de grandes volúmenes de datos en infraestructuras cloud. Ha recibido gran atención recientemente como consecuencia del desarrollo de bibliotecas e implementaciones específicas orientadas al procesamiento web, búsquedas web y problemas genéricos de *big data*.
  - El usuario especifica una función Map para generar un conjunto de valores intermedios clave/valor a partir de un conjunto de datos de gran tamaño. A continuación, una función de reducción especificada por el usuario se aplica para combinar todos los valores intermedios con el mismo valor de clave.
  - Este modelo de paralelismo simple es altamente escalable y permite explotar el paralelismo a diferentes niveles. Una aplicación MapReduce tipica es capaz de manejar varios Terabytes de datos en cientos o miles de recursos de cómputo simultáneamente.

GRIDS vs CLOUDS

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Las fronteras entre sistemas grid y cloud son borrosas actualmente, en particular por la utilización de tecnologías de workflow que se aplican para coordinar u orquestar servicios que definen modelos de procesos de negocio transaccionales sobre grid y sobre clouds.
- Las principales diferencias que se siguen manteniendo están relacionadas con el manejo de los recursos:
  - Un sistema grid utiliza un modelo de recursos estáticos, mientras que los sistemas cloud enfatizan en sistemas elásticos, basados en virtualización.
  - Sin embargo, la creación de metasistemas que combinen las características de ambos paradigmas es posible.
  - Por ejemplo, un sistema grid que combine múltiples clouds puede proveer mejores funcionalidades que un sistema cloud puro, porque puede proveer soporte explícito para negociar la asignación de recursos.
  - Los metasistemas distribuidos (clouds de clouds, grids de clouds, cloud de grids, interclouds) proporcionan nuevos desafíos y oportunidades para construir arquitecturas novedosas.

DESEMPEÑO, TOLERANCIA A FALLOS, SEGURIDAD

DESEMPEÑO, TOLERANCIA, SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Métricas de desempeño*
  - En un sistema distribuido, el concepto de desempeño es complejo y multifacético, ya que depende de muchos factores que condicionan la eficiencia, incluyendo las características de la plataforma de ejecución, los modelos de programación, las bibliotecas y middlewares utilizados.
  - Paradigma de HPC: la eficiencia de una aplicación distribuida se mide en MIPS (millones de instrucciones por segundo) o TFlops (tera operaciones de punto flotante por segundo),
  - Sistema de HTC: la eficiencia se evalúa en TPS (transacciones por segundo).
  - Desde el punto de vista del usuario, en sistemas interactivos es importante considerar el tiempo de respuesta total, que está condicionado por la latencia de la red, la velocidad de acceso a los datos y/o a los dispositivos, el tiempo requerido para encontrar y asignar recursos, etc.
  - En sistemas distribuidos en Internet, la calidad de servicio también incluye otras métricas como la facilidad de uso, la disponibilidad y robustez, la seguridad, etc.

DESEMPEÑO

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Métricas de desempeño*
  - Al analizar el tiempo de ejecución de una aplicación paralela/distribuida, la métrica más comúnmente utilizada es el *speedup*, que representa la aceleración de la versión paralela/distribuida al compararla con una aplicación secuencial de referencia.
  - Formalmente, el speedup se define mediante $S_n = \frac{TS}{TP_n}$, siendo TS el tiempo de ejecución de la versión secuencial de la aplicación y TP_n el tiempo de la versión paralela/distribuida utilizando n recursos de cómputo.
  - El valor normalizado del speedup está dado por la métrica *eficiencia*, definida por $e_n = \frac{S_n}{n}$
  - El objetivo ideal al paralelizar es conseguir el speedup lineal ($S_n = n$), pero en general se obtienen valores de speedup sublineal como consecuencia de la existencia de partes no paralelizables en la aplicación y del overhead introducido por las comunicaciones y sincronizaciones.

DESEMPEÑO

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Escalabilidad*
  - El objetivo último de diseño de una aplicación distribuida no será resolver un problema de tamaño fijo o proveer servicio a una cantidad de usuarios fija.
  - El objetivo primordial es la *escalabilidad*, definida como la capacidad de resolver instancias más complejas de un problema o de atender un número creciente de usuarios utilizando un numero potencialmente mayor de recursos computacionales.
  - La flexibilidad (capacidad de escalar en ambos sentidos) es muy importante en sistemas distribuidos que utilizan (y pagan por) recursos computacionales de forma elástica. Un sistema flexible permitirá atender a un número potencialmente ilimitado de usuarios, y a la vez evita sobredimensionar la infraestructura requerida (con el consiguiente impacto económico).
  - Varias dimensiones de la escalabilidad son consideradas en sistemas paralelos y distribuidos …

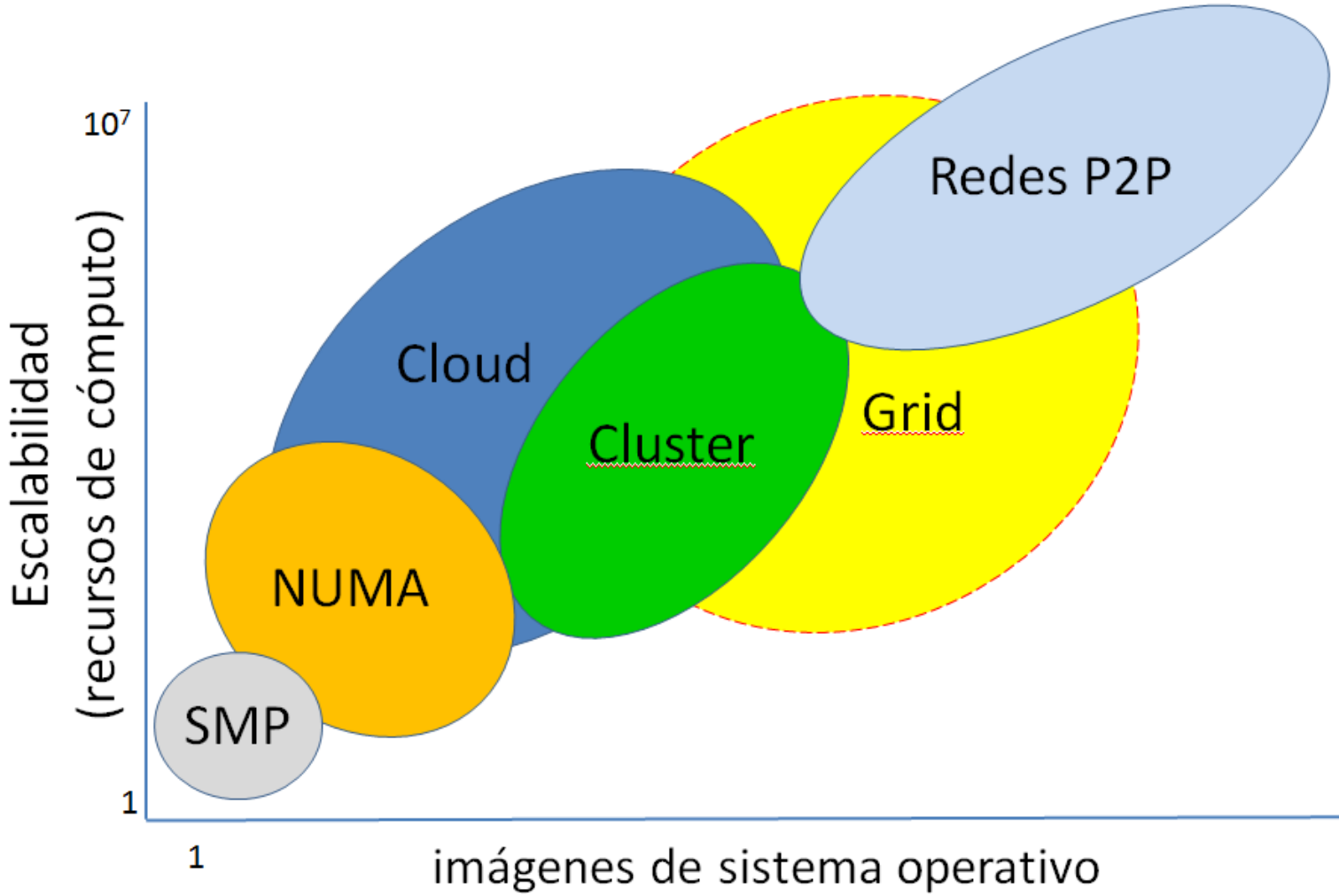
DESEMPEÑO

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- Varias dimensiones de la escalabilidad son consideradas en sistemas paralelos y distribuidos:
  - *Escalabilidad de tamaño de la infraestructura*: refiere a la capacidad de mejorar el desempeño, el throughput, y/o las funcionalidades al utilizar más recursos de cómputo.
  - *Escalabilidad del software*: refiere a la capacidad de realizar actualizaciones en el software de base, incorporar nuevos entornos de desarrollo amigables y portar las aplicaciones desarrolladas.
  - *Escalabilidad de la aplicación*: refiere a la capacidad de la aplicación de adaptar el tamaño y la escalabilidad del problema con el tamaño y la escalabilidad de la infraestructura computacional utilizada. Contar con una implementación flexible en ambos aspectos es la situación más deseable.
  - *Escalabilidad de la tecnología*: Refiere a la capacidad de un sistema distribuido de adaptarse a cambios en las tecnologías utilizadas, por ejemplo tecnologías de componentes y de red, capacidad de utilizar entornos heterogéneos, etc.

DESEMPEÑO

- Escalabilidad versus imágenes del sistema operativo



TOLERANCIA A FALLOS

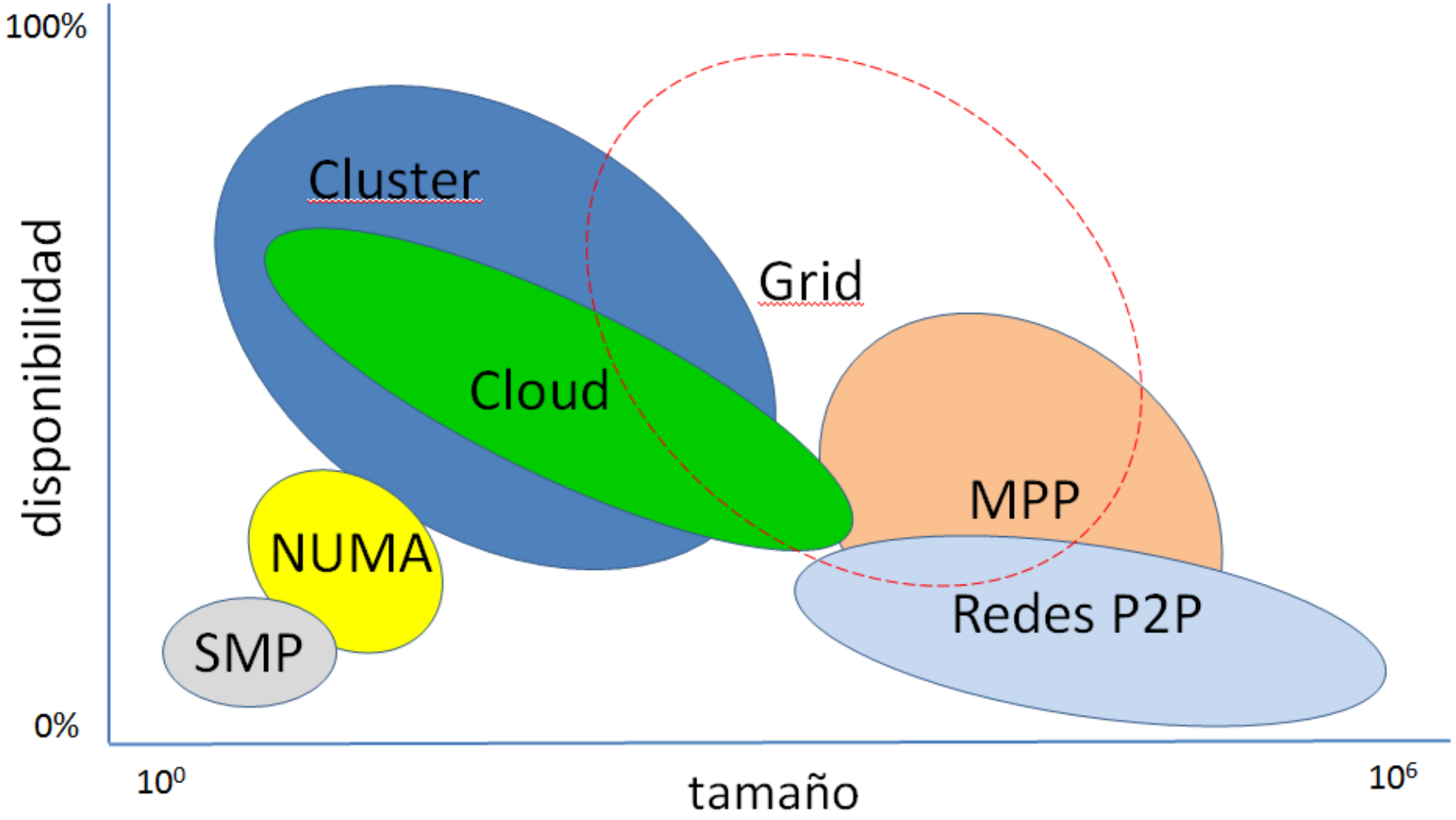
UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Disponibilidad de un sistema*
  - Se define tomando en cuenta su *tiempo medio entre fallos* (mean time to failure, MTTF) y su *tiempo medio de recuperación* (mean time to repair/recover, MTTR).
  - En una aplicación distribuida existen muchos componentes de hardware y software involucrados, cada uno con diferentes tasas de fallo.
  - Un fallo que puede ocasionar que el sistema deje de estar en estado operativo se denomina *punto único de fallo* (*single point of failure*, SPOF).
  - Un sistema distribuido debería tener muy pocos (idealmente ninguno) puntos únicos de fallo, y en caso de existir deben proveerse mecanismos de tolerancia y recuperación que permitan al sistema volver al estado operativo en el menor tiempo y de la forma más automática posible.
  - Estas características se consiguen agregando redundancia de hardware, incrementando la confiabilidad de los componentes de software, y aplicando metodologías de diseño que permitan analizar y verificar la mayor cantidad de eventos que puedan producir fallos.

DESEMPEÑO, TOLERANCIA, SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Disponibilidad de sistemas*



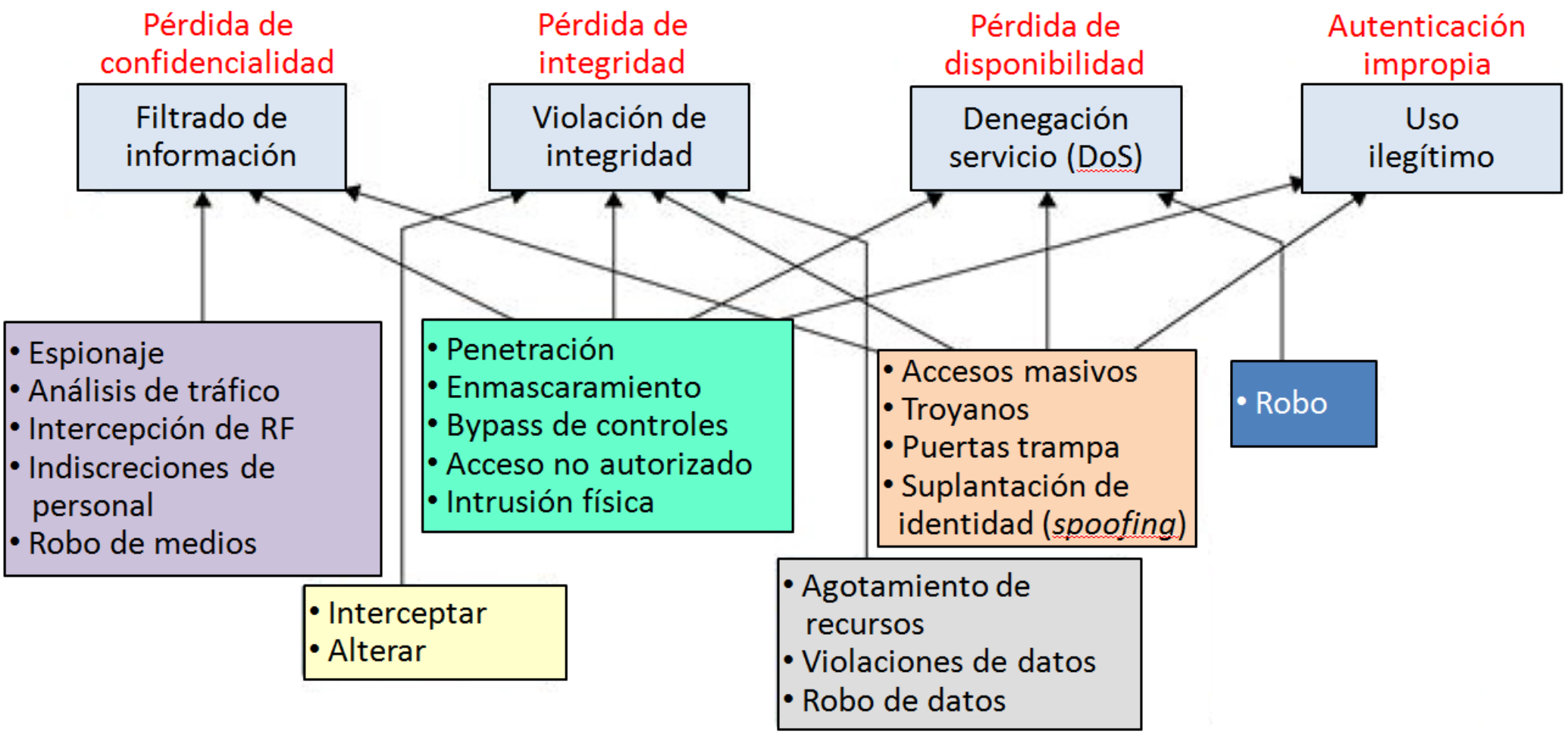
SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Seguridad e integridad de datos*
  - Los sistemas distribuidos demandan ciertos niveles básicos de seguridad y protección de datos para ser utilizados en aplicaciones realistas de la sociedad de información actual.

- *Seguridad de sistemas y redes*
  - Cuatro requisitos de seguridad se contemplan habitualmente para sistemas distribuidos: *confidencialidad*, *integridad*, *disponibilidad* y *veracidad de la autenticación*.
  - Diferentes tipos de ataques pueden conducir a pérdidas en algunas de estas características deseables

SEGURIDAD



Ataques sobre la seguridad de sistemas distribuidos

SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Protección de datos y derechos de autor*
  - El tema de protección de datos resulta especialmente importante cuando se trabaja sobre infraestructuras distribuidas.
  - Es conocida la fama de las redes P2P de cometer violaciones a la propiedad intelectual de material difundido en ellas (principalmente audio y video).
  - Los clientes regulares pueden compartir derechos de archivos con clientes no regulares o piratas, y la ausencia de mecanismos apropiados de seguridad han ocasionado que las redes P2P hayan sido prácticamente descartadas como base para los servicios de entrega de contenido para aplicaciones comerciales.

SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Tecnologías de defensa de sistemas distribuidos*
  - Tres generaciones de herramientas para mejorar la confiabilidad y disponibilidad de sistemas distribuidos
    1. Herramientas diseñadas para prevenir o evitar intrusiones, mediante políticas de control de acceso, sistemas de encriptado y mecanismos de privacidad. Los sistemas de control de acceso siempre son tan débiles como el enlace más débil de la red, y los intrusos pueden aprovechar puntos débiles en los servicios de proveedores externos o enlaces de conectividad.
    2. Sistemas enfocados en identificar los accesos no autorizados en una manera eficiente y rápida, para poder luego instrumentar mecanismos de defensa y contramedidas para evitar perder integridad en los datos. Las técnicas propuestas en esta generación se implementan hoy en día en firewalls, sistemas de detección de intrusos, servicios de infraestructura de clave pública, sistemas basados en reputación, etc.
    3. Las herramientas evolucionaron, incorporando técnicas de inteligencia computacional que permiten implementar respuestas más inteligentes a las intrusiones y a los ataques.

SEGURIDAD

UNIVERSIDAD DE LA REPÚBLICA URUGUAY

- *Infraestructuras para protección de datos*
  - Una infraestructura de seguridad es imprescindible para resguardar servicios web y servicios cloud disponibles públicamente a través de Internet. Al nivel del usuario es necesario implementar negociaciones de confianza (*trust negotiation*) y mecanismos de reputación pública entre múltiples usuarios e instituciones.
  - Al nivel de aplicación, deben establecerse precauciones de seguridad para contener ataques masivos de virus y gusanos, intrusiones y ataques de DoS distribuidos, piratería online y acceso no autorizados a datos disponibles.
  - En sistemas distribuidos globales, los mecanismos se separan entre proveedores y usuarios, bajo diferentes niveles de responsabilidad de acuerdo al paradigma utilizado:
    - Los proveedores son responsables de la disponibilidad de la plataforma, los usuarios son responsables de la confidencialidad, los proveedores de contenido son responsables de garantizar la integridad de los datos.
    - En modelos de cloud bajo paradigmas PaaS y SaaS, proveedores y usuarios son igualmente responsables de la integridad de datos y confidencialidad.