Sistemas Operativos II

Multiprocesadores

Universidad Arturo Jauretche
Ingeniería Informática

Docentes:

Coordinador Ing. Jorge Osio

Ing. Eduardo Kunysz

Motivaciones

- Continuamente se requiere mayor potencia de computo:
  - Astrónomos → entender el universo
  - Biólogos → descifrar genoma
  - Ingenieros → simulaciones mas eficientes y con mayor detalle
- La solución siempre fue aumentar la velocidad del reloj.

Problemas de subir el clock

- Limite → velocidad de la luz.

  30 cm/nseg → vacío

  20 cm/nseg → cobre → a 10 GHz → 2cm

  para 100 GHz → 2mm

  para 1 THz → 100 micrones

  Nuevo problema → disipación de calor

  ↑Rapido => ↑calor genera

  ↓Tamaño => mas dificil extraer el calor

ución: Computadoras masivamente en para

Computación paralela (1)

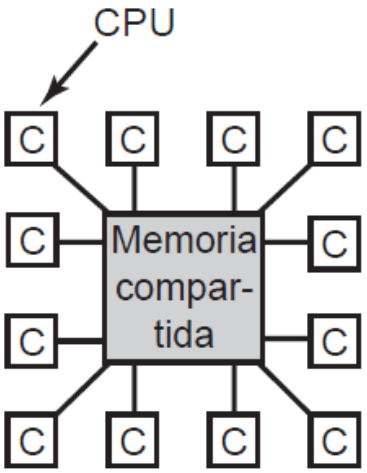
- Es fácil colocar 1 millos de computadoras en una habitación o alrededor del mundo.

¡Lo difícil es comunicarlas!

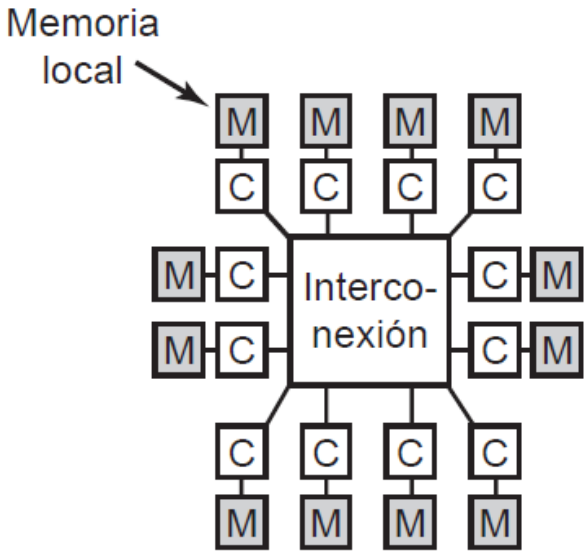
- Toda la comunicación se realiza mediante envío de mensajes.

Computación paralela (2)

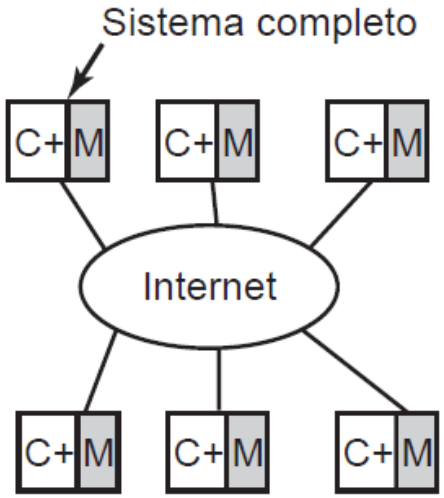
\+ Velocidad -



Multiprocesador con memoria compartida



Multicomputadora con paso de mensajes



Sistema distribuido de área amplia

Multiprocesador

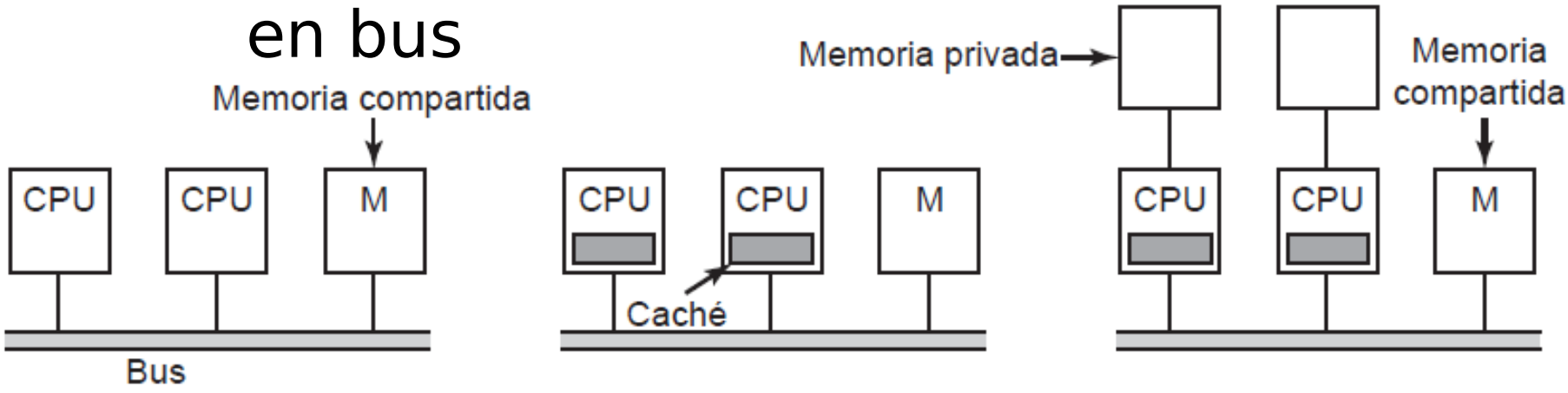
- Un Multiprocesador es un sistema de cómputo en el que dos o mas CPUs comparten todo el acceso a una RAM común.

Hardware

- Tienen la propiedad de que c/CPU puede direccionar toda la memoria.
- Se pueden diferenciar en:
  - **UMA:** Uniform Memory Access
  - **NUMA:** Non Uniform Memory Access

Multiprocesadores UMA (bus)

Tres Multiprocesadores basados en bus



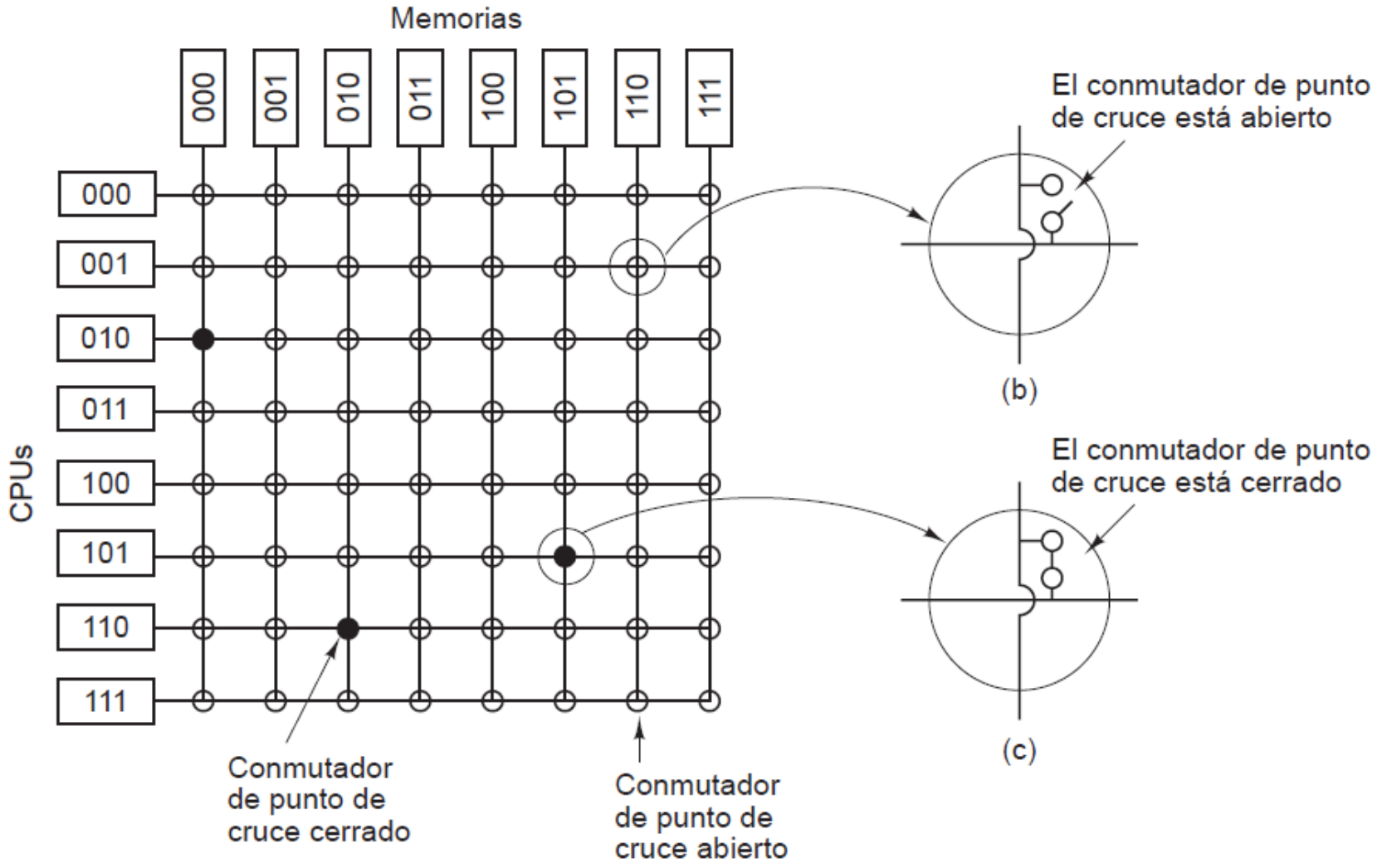
Sin cache

Con cache

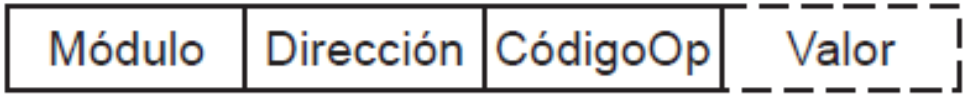
Con caché y memorias privadas

- Leer:
  - Comprueba si CPU busy
  - Coloca la dirección
- Si ocupado
  - Espera que BUS inactivo
- Problema mas de 32 o 64 CPUs
  - Solución: Cache

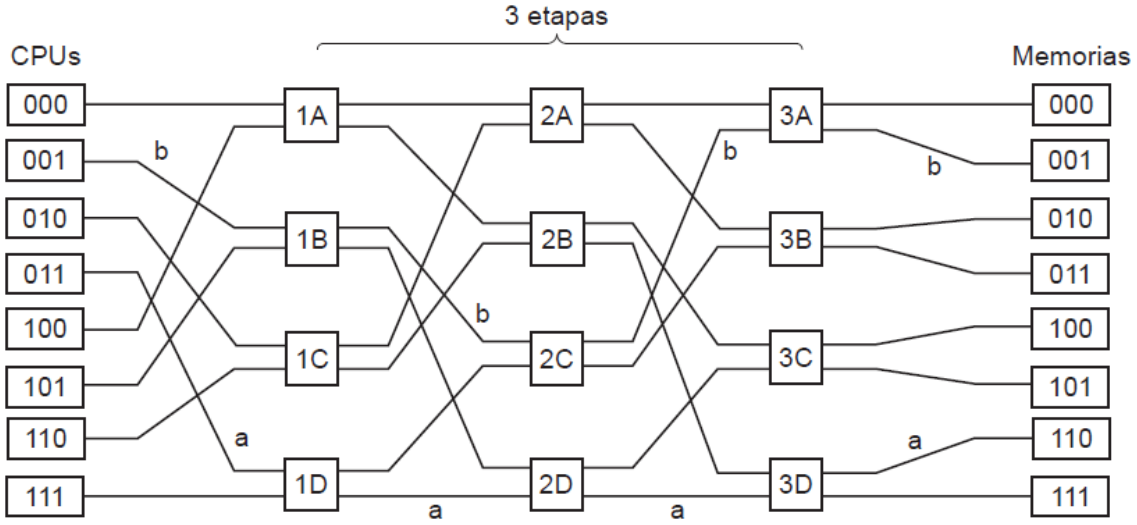
Multiprocesadores UMA interrupciones de barras cruzadas



Multiprocesadores UMA que utilizan redes de comunicación multietapa



onmutador de 2 x 2 con dos líneas de entrada y dos de salid

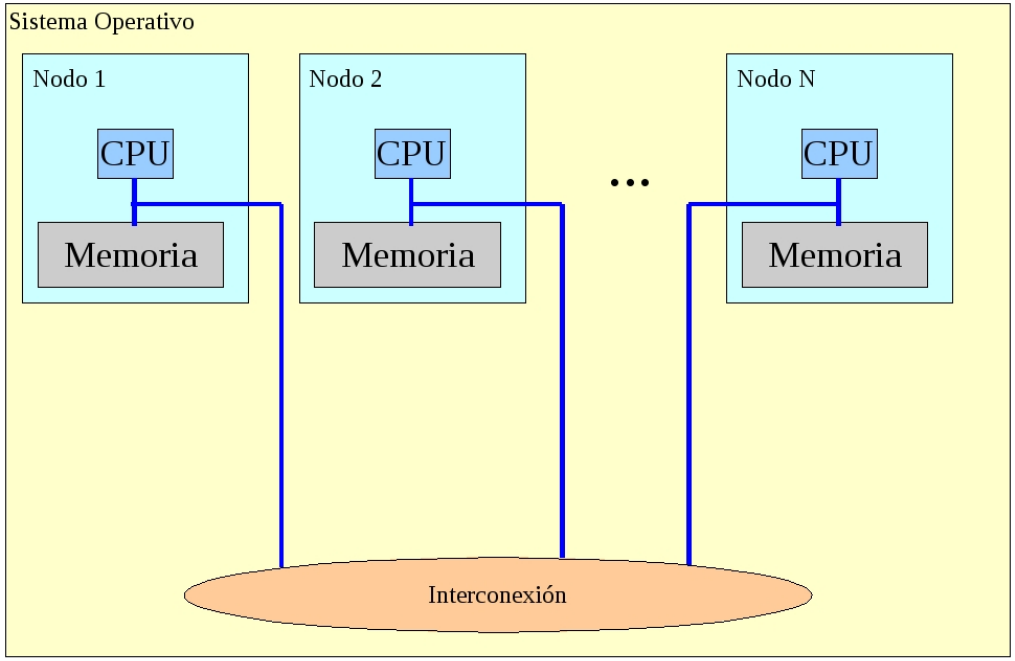


Red de conmutación omega

Las redes de interconexión multietapa son las más avanzadas entre las redes actuales de conmutación. En el caso de la red omega, para conectar N nodos se necesitan $\log_2 N$ etapas con N/s switches 2x2 cada una.

Multiprocesadores NUMA

- Cada procesador tiene su memoria local
- Puede acceder a la memoria de otro procesador de forma



NUMA

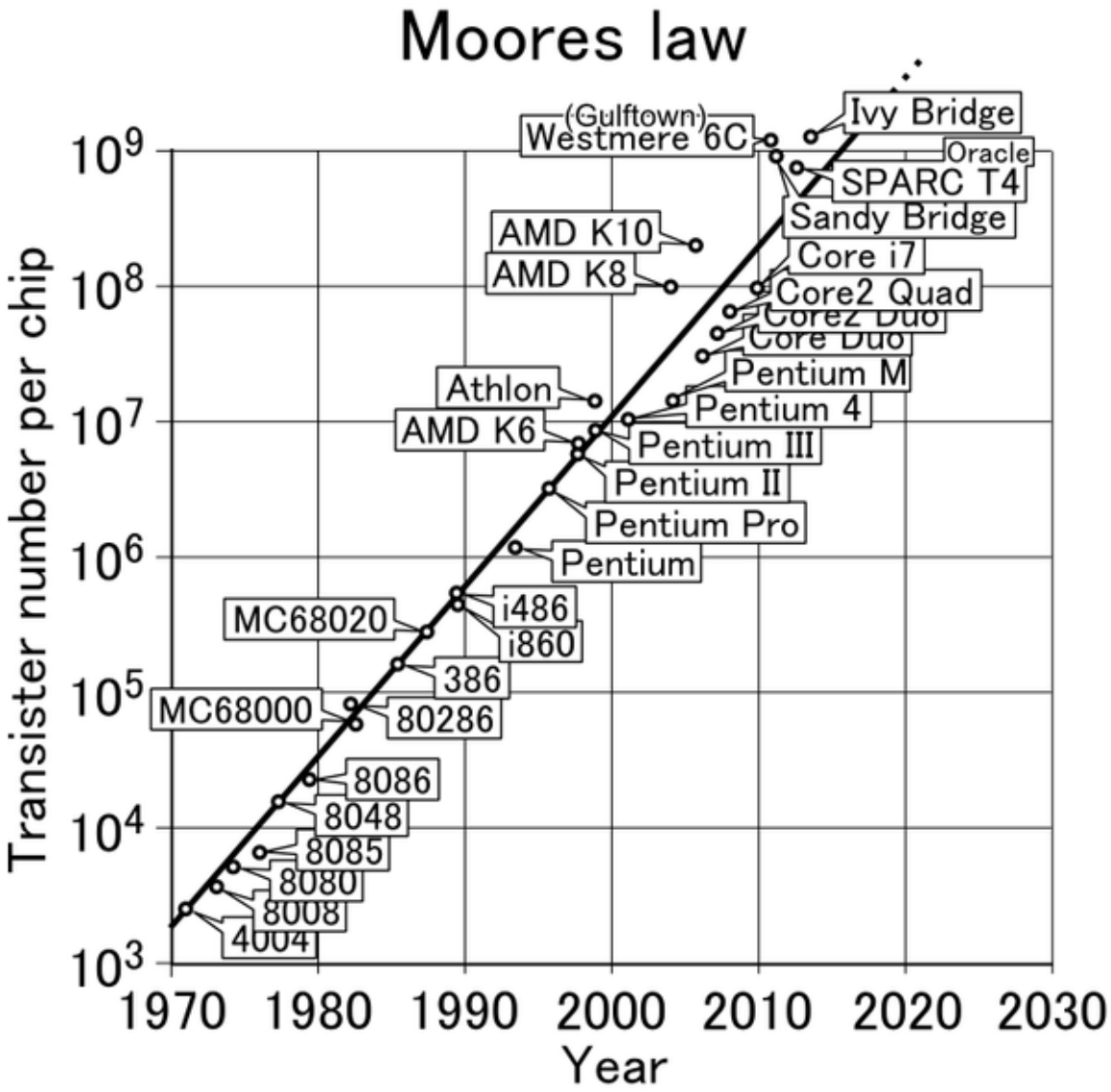
Multiprocesadores NUMA

- Las máquinas NUMA poseen tres características clave:
  1. Hay un solo espacio de direcciones visible para todas las CPUs.
  2. El acceso a la memoria remota es mediante instrucciones LOAD y STORE.
  3. El acceso a la memoria remota es más lento que el acceso a la memoria local.

Chips multinúcleo

Ley de Moore:

A mediada que mejora la tecnología los transistores se hacen cada vez mas pequeños.



Chips multinúcleo

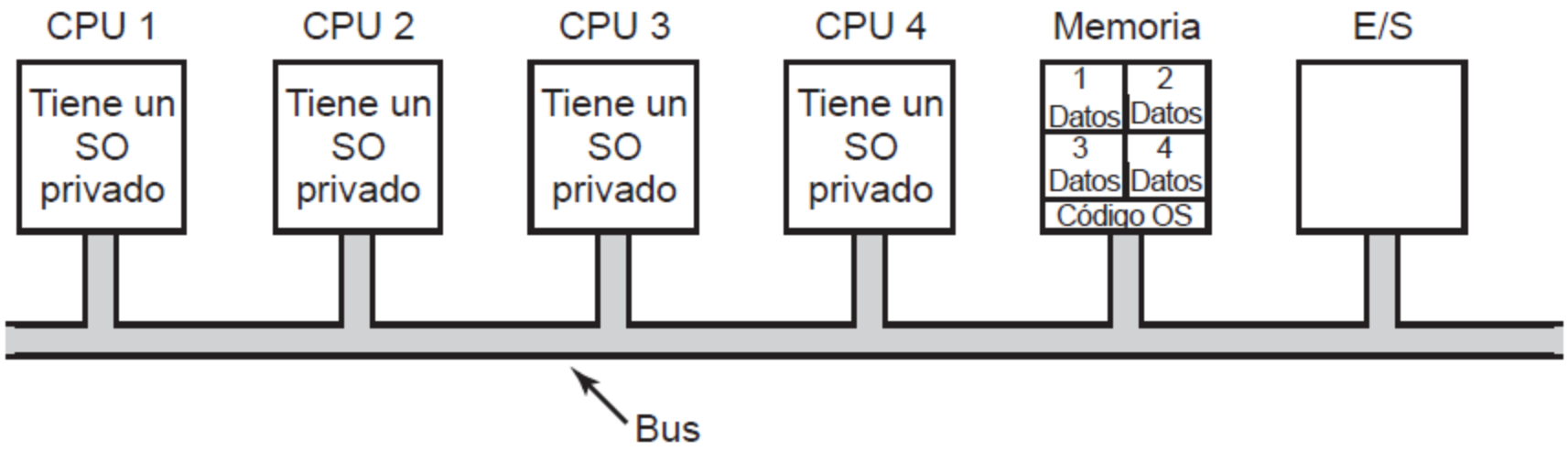
- ¿Como seguir mejorándolo?: Subir cache solo mejora de 95% a 99% -> Mas CPUs por pastilla
- No necesariamente comparten cache, pero siempre comparten memoria principal.
- **snooping :** La si un CPU modifica una palabra de cache, se modifica en todos, esto permite consistencia

Tipos de SO multiprocesadores

- Hay varias metodologías posibles:
  - Cada CPU tiene su propio SO
  - Multiprocesadores maestro esclavo
  - Multiprocesadores simétricos

Tipos de SO multiprocesador
Cada CPU tiene su propio SO

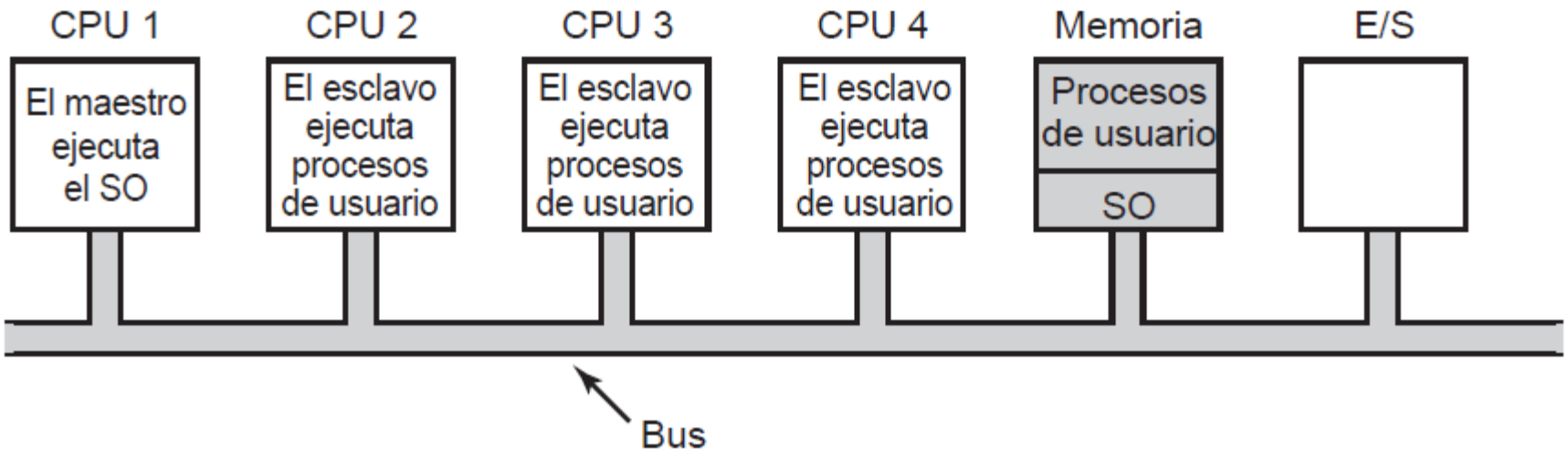
- Se divide estáticamente la memoria y su propia copia privada del SO.
- N CPUs como N computadoras independientes



Problemas de distribución de recursos: posiblemente se cargue mas un procesador que otro, desperdicio de disco por el uso de la cache

Tipos de SO multiprocesador
Multiprocesadores maestro-esclavo

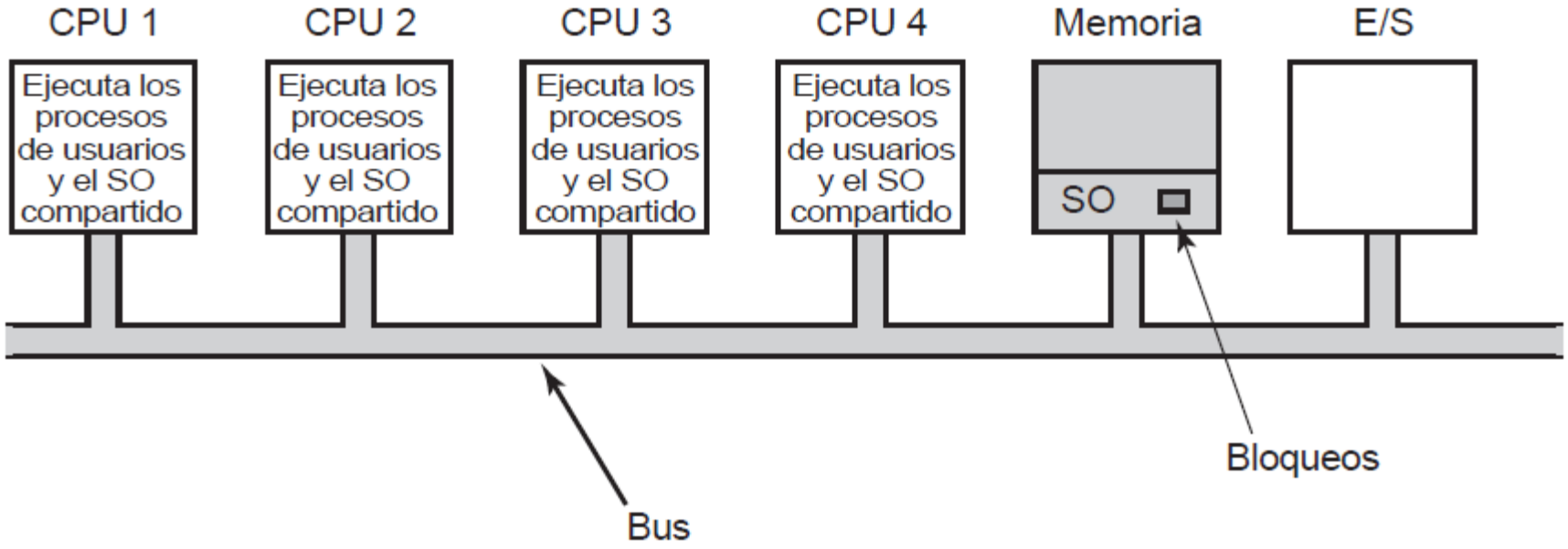
- El CPU1 tiene el SO, sus tablas y las llamadas al sistema.
- Mejor distribución de recursos, CPU1 administra, solo hay una cache de buffer.



Problemas: si hay muchos CPUs: CPU1 -> cuello de botella

Tipos de SO multiprocesador
Multiprocesadores simétricos (SMP)

- Hay una copia del SO en memoria, pero cualquier CPU puede ejecutarlo.
- Equilibra los procesos y la memoria en forma dinámica



Problemas: si dos o mas CPUs están ejecutando código del SO al mismo tiempo, problemas de ejecución, paginado, interbloqueo.
Solución: **MUTEX**

Multicomputadoras

- CPUs con acoplamiento fuerte que no comparten memoria
- Nombres:
  - **Cluster**
  - **COWS** (Cluste

  Of WorkStations

Multicomputadora: Hardware

- El nodo básico de una multicomputadora consiste en:
  - Una CPU, la memoria,
  - Una interfaz de red
  - Algunas veces un disco duro.
- El nodo puede estar empaquetado en un gabinete de PC estándar sin periféricos HMI

Multicomputadoras Granularidad

- El programa se divide en **N** procesos concurrentes que se ejecutan en **n** computadoras. Si **n<N** existirán varios procesos (time sharing) en cada computadora (canales externos vs. canales internos, no se puede compartir).
  - Definimos, para un proceso:

$$Granularidad = \frac{tiempo\ de\ cómputo}{tiempo\ de\ comunicación}$$

  - Granularidad gruesa: cada proceso tiene muchas instrucciones
  - Granularidad fina: pocas (incluso 1)

Los multiprocesadores funcionan mal en aplicaciones de granularidad fina debido el alto costo de las comunicaciones.

Multicomputadora: Hardware Tecnología de interconexión

- Cada nodo tiene **una tarjeta de interfaz de red**, de la cual salen uno o dos cables (o fibras).
- Estos cables se conectan a otros nodos o a switches.
- Como alternativa al diseño de un solo switch, los nodos pueden formar **un anillo** con dos cables que provienen de la tarjeta de interfaz de red: uno que entra al nodo por la izquierda y otro que sale del nodo por la derecha
- **Se pueden presentar diversas topologías:**

Multicomputadora: Hardware

Topologías de interconexión

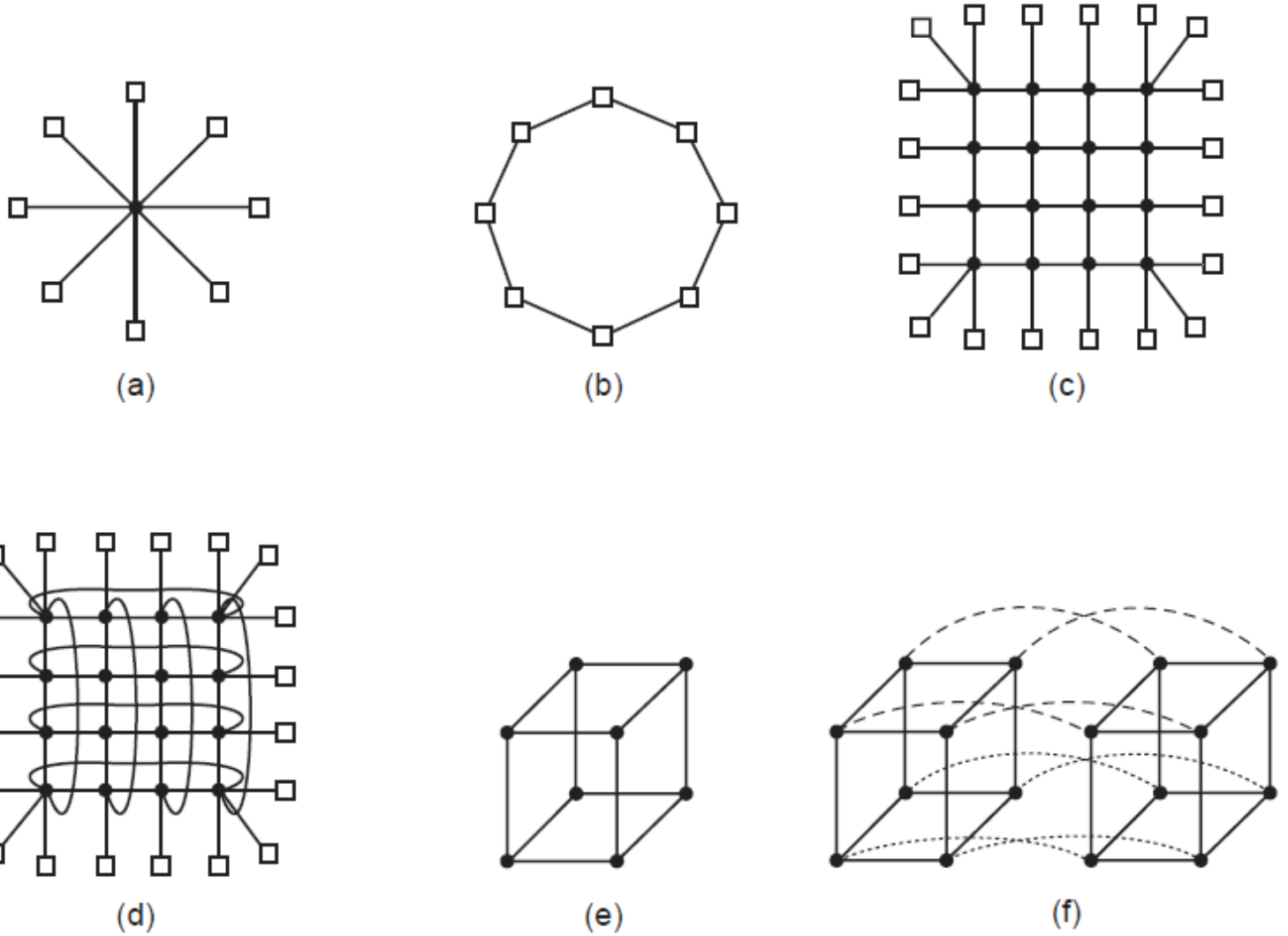


Figura 8-16. Varias topologías de interconexión. (a) Un solo switch. (b) Un anillo. (c) Una rejilla. (d) Un dobel toroide. (e) Un cubo. (f) Un hipercubo 4D.

Multicomputadoras
El modelo de programación

- Primitivas de comunicación: send, receive, broadcast, barrier.
- Mensajes bloqueantes (3 way) y no bloqueantes. PVM y MPI: bibliotecas de comunicaciones para C.
- Open Source: LAM-MPI y MPICH

```
#include <stdio.h>
#include <mpi.h>
int main(){
    int rank, size;
    MPI_Init( &argc, &argv );
    MPI_Comm_size( MPI_COMM_WORLD, &size );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    printf( "Hello world from process %d of %d\n", rank, size );
    MPI_Finalize();
    return 0;
}
```

Virtualización

Una máquina virtual (MV) es un duplicado de una máquina real, eficiente y aislado.

Gerald J. Popek and Robert P. Goldberg. Formal requirements for virtualizable third generation architectures. CACM, 17(7):413–421, 1974.

Virtualización

- Cada máquina virtual es idéntica al verdadero hardware, cada una puede ejecutar cualquier sistema operativo que se ejecute directamente sólo en el hardware.

- Distintas máquinas virtuales pueden (y con frecuencia lo hacen) ejecutar distintos sistemas operativos.

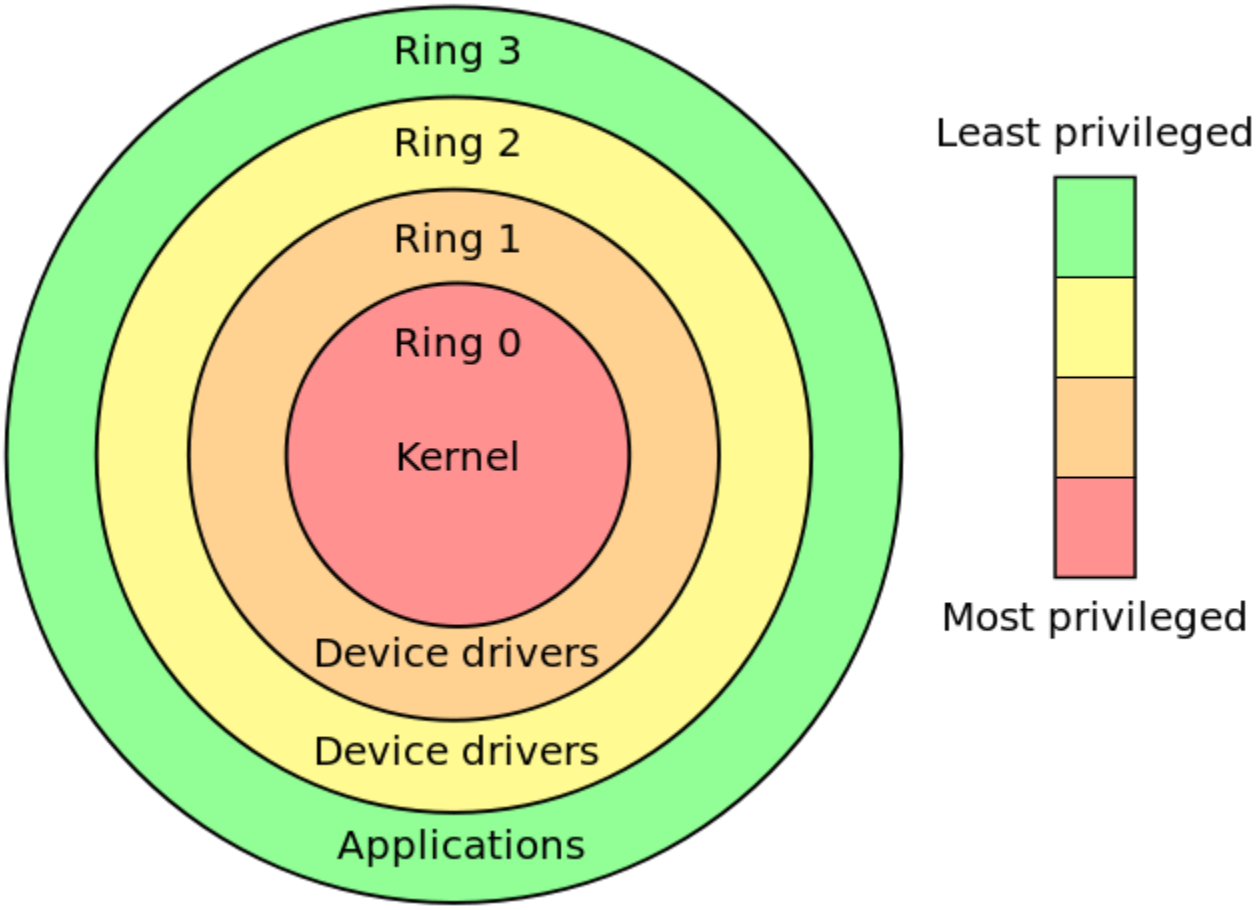
Maquinas virtuales

- **Duplicado**: La MV se debería comportar de forma idéntica a la máquina real, excepto por:
  - La existencia de menos recursos disponibles (incluso diferentes entre ejecuciones).
  - Diferencias de temporización al tratar con dispositivos.
- **Aislado**: Se pueden ejecutar varias MV sin interferencias.
- **Eficiente**: La MV debería ejecutarse a una velocidad cercana a la del HW real.
  - Requiere que la mayoría de las instrucciones se ejecuten directamente por el HW.

Anillos de seguridad

- El **modo de supervisor** es un flag de hardware que puede ser cambiado por código corriendo en el software de nivel de sistema.
- Determina si sería posible ejecutar operaciones de código máquina como la modificación de registros para varias tablas de descriptores, o llevar a cabo operaciones tales como deshabilitar las interrupciones.

Anillos de seguridad



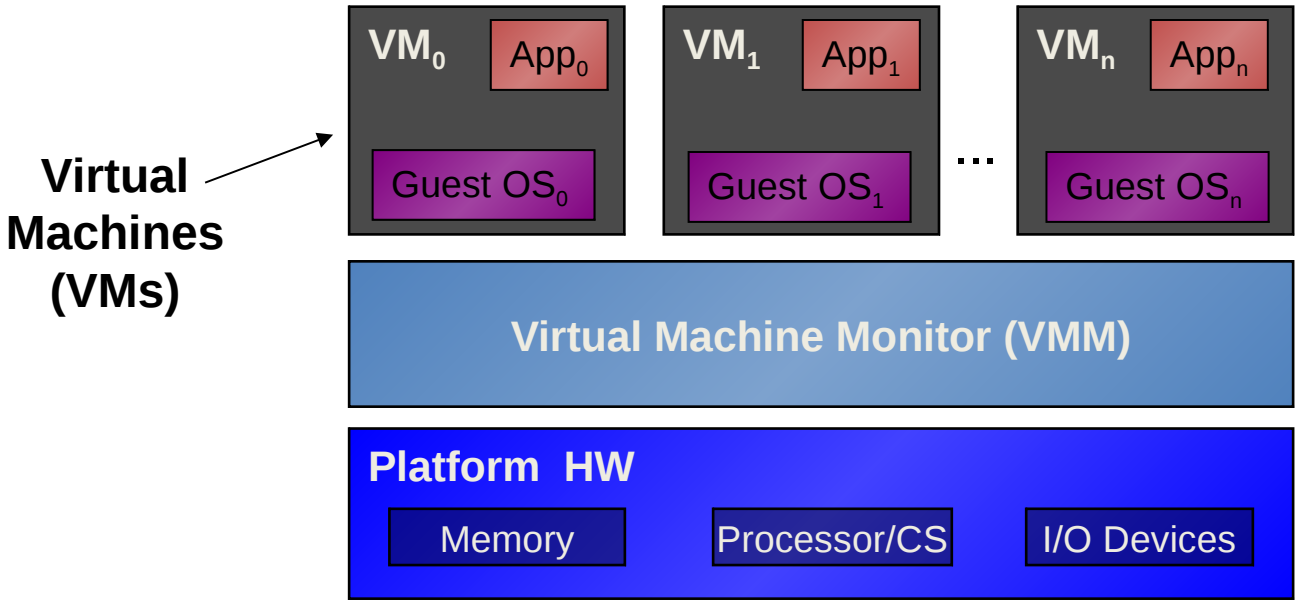
Virtualización: Hipervisor (Virtual Machine Monitor -> VMM)

- Programa que corre sobre el hardware real para implementar la máquina virtual.

- Control de recursos y planificación de huéspedes.

- Implicaciones:
  - MMV necesita ejecutarse en modo supervisor.
  - Software huésped en modo usuario.
  - Instrucciones privilegiadas de huéspedes implican traps.
  - MMV interpreta/emula instrucciones privilegiadas

Virtualización: Hipervisor



- VMM: Capa de software de sistema
  - Permite que se ejecuten varias MV sobre plataforma HW única
  - Permite ejecutar aplicaciones sin modificar

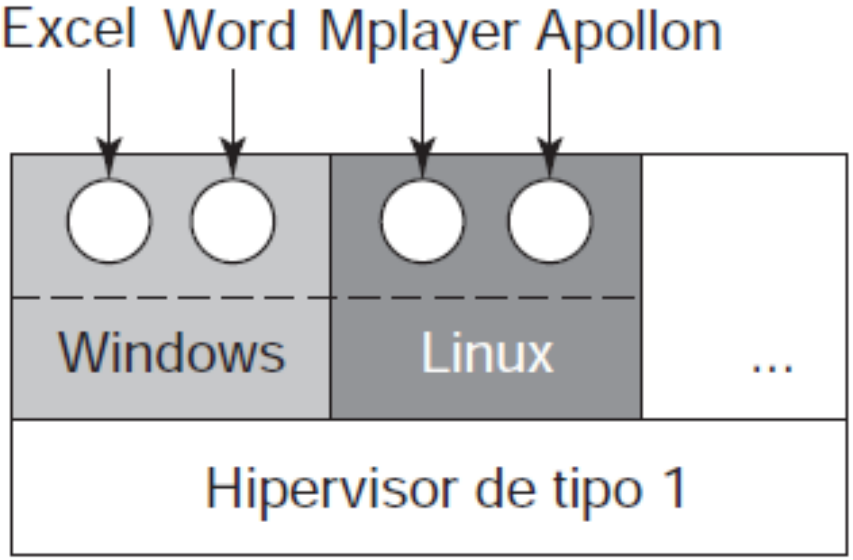
Virtualizacion Instrucciones

- **Instrucciones sensibles:** cada CPU con modo de kernel y modo de usuario tiene un conjunto de instrucciones que sólo se pueden ejecutar en modo de kernel (Ej: las operaciones E/S).
- **Instrucciones privilegiadas:** conjunto de instrucciones que producen una trampa (interrupción) si se ejecutan en modo de usuario.

Una máquina se puede virtualizar sólo si las instrucciones sensibles son un subconjunto de las instrucciones privilegiadas

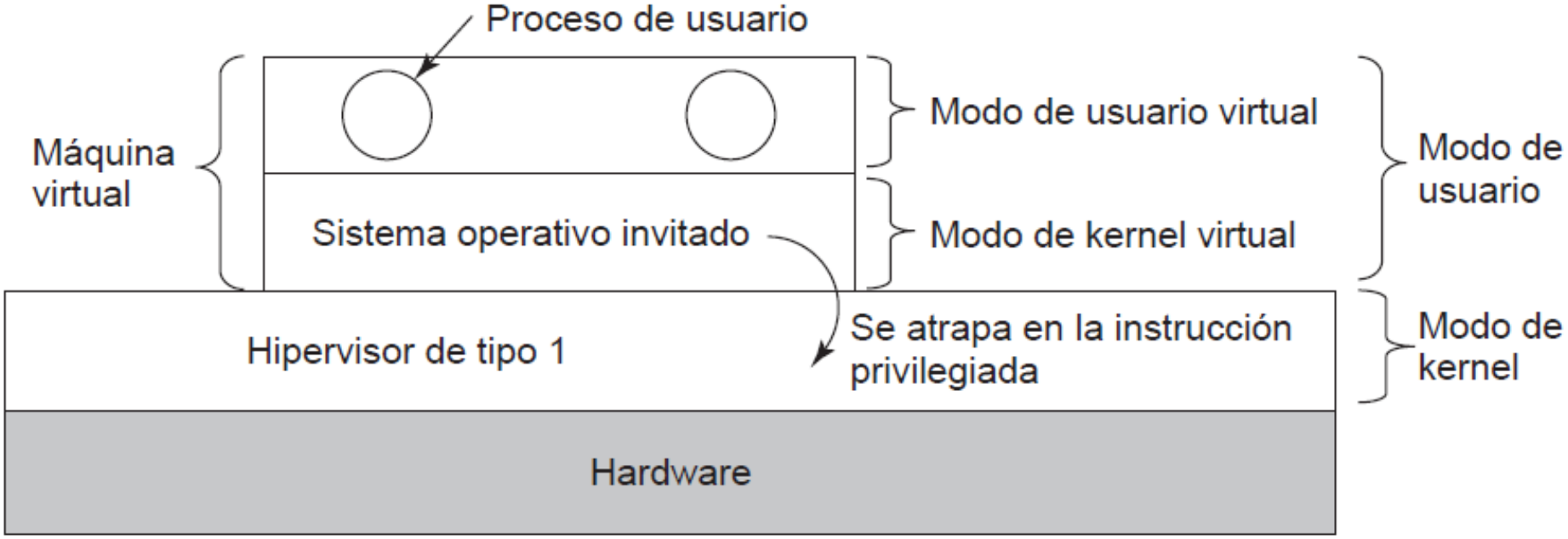
Tipos de virtualización

- ***Hipervisor del tipo 1 (full virtualization):*** El monitor se ejecuta directamente sobre el hardware y los huéspedes sobre el monito



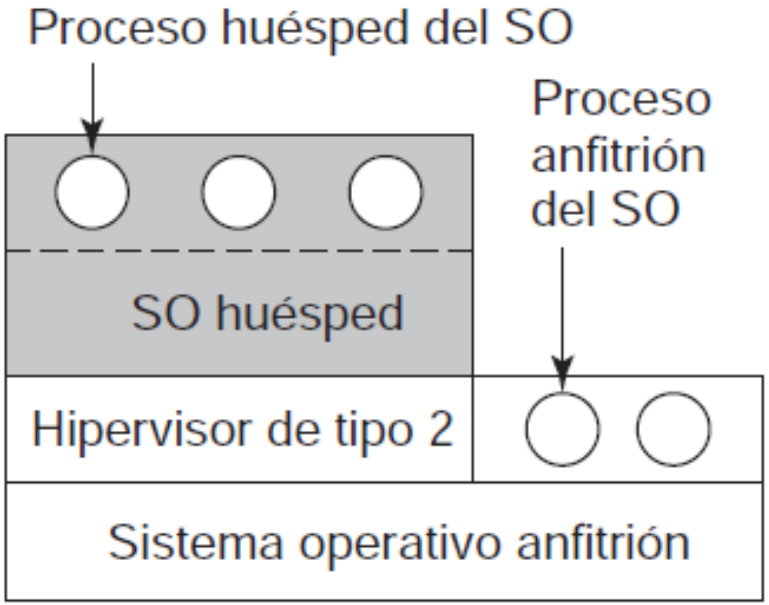
Tipos de virtualización

- Los **hipervisores de tipo 1** se ejecuta en modo de kernel
- La máquina virtual se ejecuta como un proceso de usuario en modo de usuario y, como tal, no puede ejecutar instrucciones sensibles
- Sistema operativo invitado piensa que se encuentra



Tipos de virtualización

- ***Hipervisor del tipo 2 (Indirecto)*** **:** El monitor se ejecuta sobre un sistema operativo y los huéspedes sobre el monitor.
  - Ejecución de sistemas operativos en instancias de la máquina virtual.
  - Menos eficiente (depende…ver siguiente).



Tipos de virtualización
Virtualizacion tipo 2

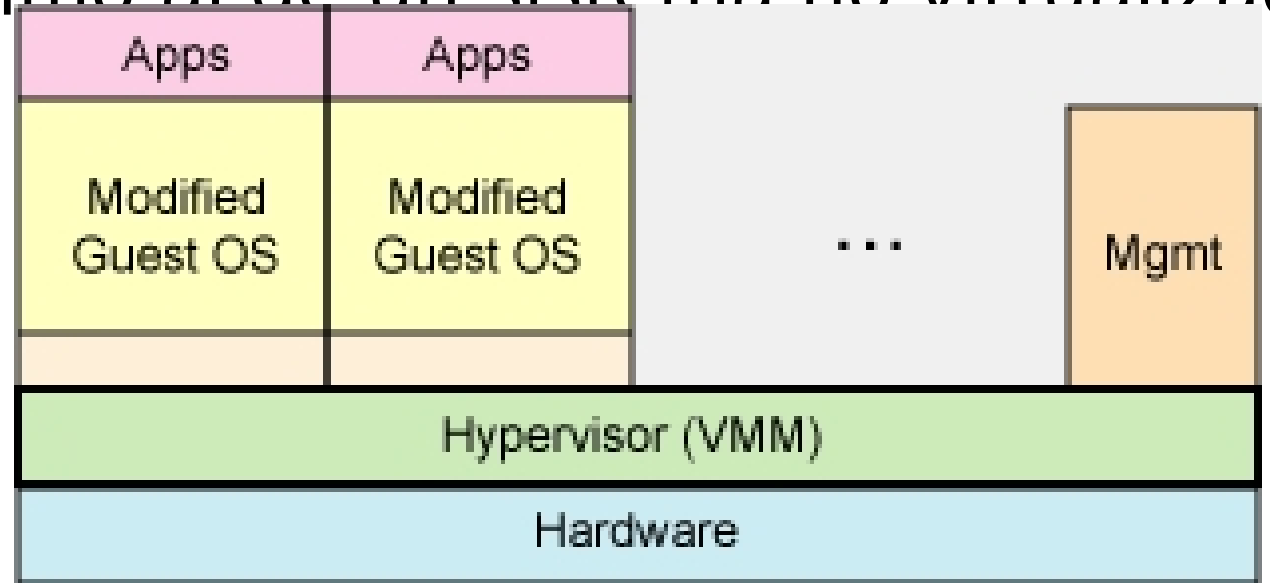
¿Cómo funciona?

1. Al ejecutar un programa binario -> busca **bloques básicos (BB)**
2. Se inspecciona el bloque básico para ver si contiene instrucciones sensibles.
3. Cada instrucción se sustituye con una llamada a un procedimiento.
4. Luego el BB se coloca en la cache y se ejecuta
5. A esta técnica se la conoce como:
   - **traducción binaria** -> mas eficiente que VT1 menos interrupciones

Bloques básicos: ejecuciones de instrucciones seguidas que terminan en un salto, una llamada, alguna instrucciones que modifica el contador de programa

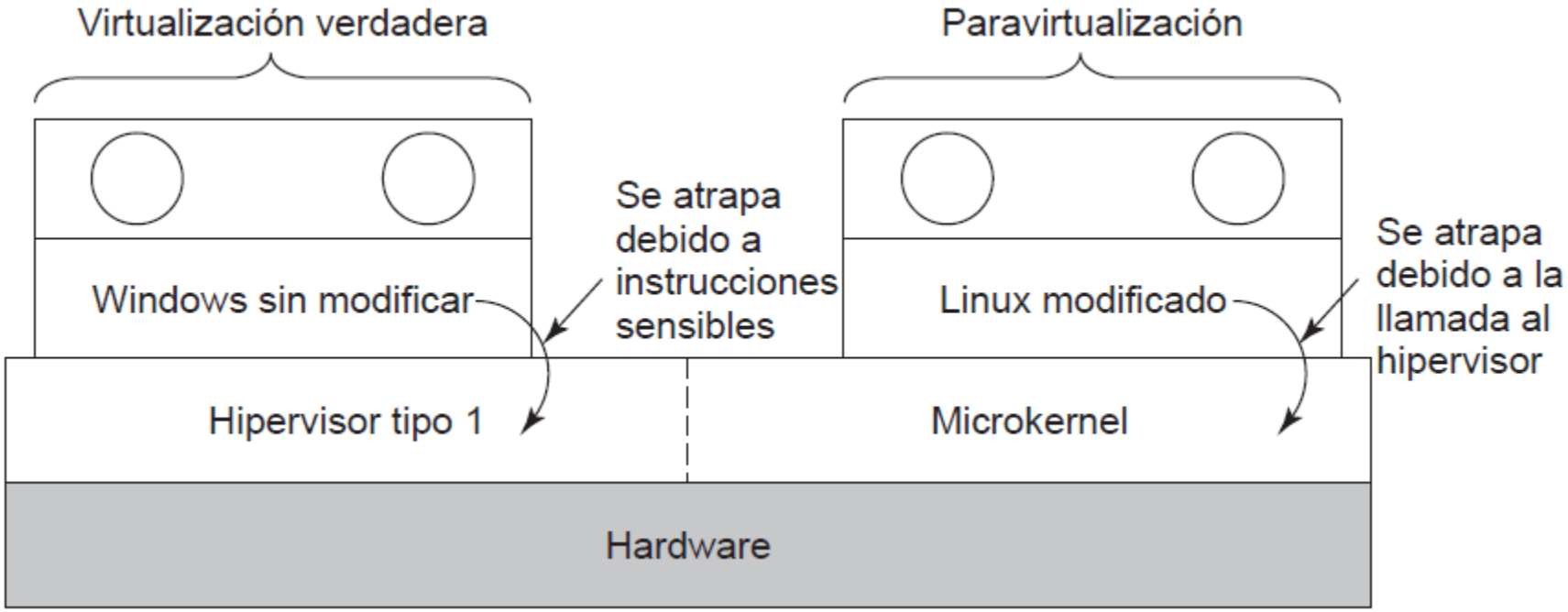
Tipos de virtualización

- ***Paravirtualización***: la paravirtualización precisa que los sistemas operativos alojados sean modificados por el hipervisor, lo que es una desventaja. Pero la paravirtualización ofrece un rendimiento próximo al de un sistema no virtualizado



Tipos de virtualización
Paravirtualización

- Se modifica el código fuente del SO invitado, de manera que en vez de ejecutar instrucciones sensibles, realiza llamadas al hipervisor.
- El hipervisor debe definir una API para SO invitado.
- El hipervisor pasa a ser un microkernel



Virtualización de la memoria

- Manejo de memoria virtual
  - Tabla de páginas (multinivel) en VM
  - Debe mapearse a tablas de páginas del hardware real
- Ej: SO invitado A crea tabla de páginas con la siguiente asignación
  - 5 -> 10, 8 -> 11, 2->12
- SO invitado B ahora crea tabla de páginas con la siguiente asignación
  - 4 -> 10, 5 -> 11, 6->12
- El VMM (hipervisor) debe mantener una tabla de **páginas oculta (shadow)** porque no puede darle los mismos marcos a dos VMs.

Virtualización de la memoria

- La creación de la tabla de páginas es una acción “sensible” (modifica MMU que genera trap), pero la posterior actualización de la tabla no es sensible. Posible solución: marcar las paginas de tabla como de solo lectura
- **Caso SO paravirtualizado:**
  - Se cambian las instrucciones de manejo de MMU por llamadas al hipervisor
  - Las actualizaciones sobre la tabla de páginas se puede hacer en modo batch, llamando al hipervisor luego que el SO invitado ha realizado todos los cambios.

Virtualización de E/S

- Cada VM piensa que tiene todo el hardware disponible para si
  - Ej: discos, impresoras etc
- Las operaciones de E/S son sensibles por lo que son tratadas por el hipervisor
- Para algunos recursos es preferible virtualizar el dispositivo
  - Discos representados como un archivo en el FS del SO anfitrión
  - Permite utilizar nuevo hardware sobre SO que no saben manejarlo
- DMA utiliza direcciones absolutas (físicas), por lo que deben ser traducidas por el hipervisor antes de efectivizarse el DMA.
- MMU de E/S virtualizada

Virtualización de la E/S

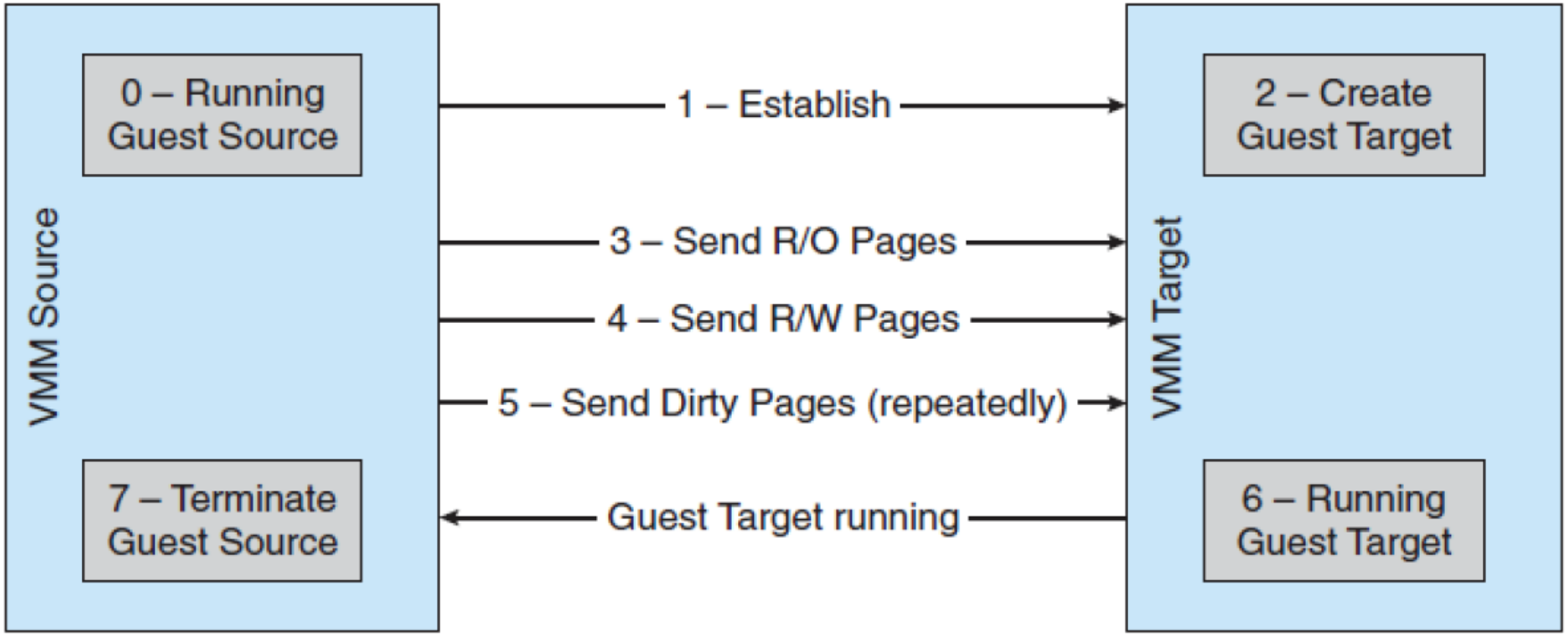
- Otro método es utilizar una de las máquinas virtuales para que refleje la E/S de todas las VM
- A esta VM a veces se le llama **dominio 0**
- Mayor facilidad para este esquema en sistemas paravirtualizados
- Hipervisores de tipo 2 pueden utilizar los drivers del SO anfitrión
- Hipervisores de tipo 1 pueden utilizar los drivers del dominio 0

Migración en vivo (Live Migration)

- Permite “mover” una VM de un host a otro de forma transparente para los usuarios de la VM
  - Debe permitir mover el estado (memoria) de la VM, sus conexiones IP, etc
  - No mueve estado del disco (se utilizan discos accesibles remotamente)
  - Implementado en hipervisores de tipo 1

Migración en vivo (Live Migration)

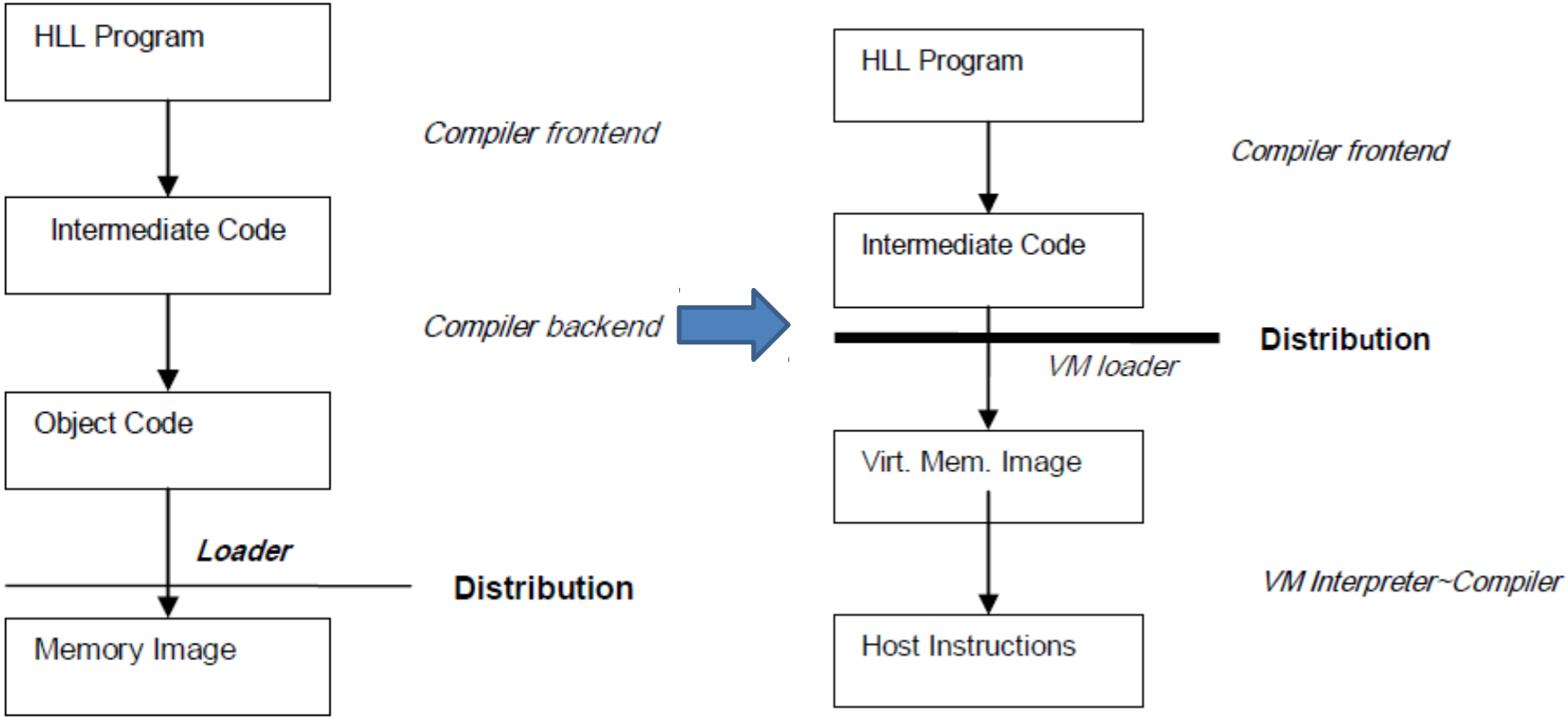
- Procedimiento:
  - Se crea VM en host destino
  - Se envían páginas de solo lectura
  - Se envían páginas de lectura-escritura
  - Mientras haya páginas modificadas, éstas se envían al host destino
  - Comienza a operar VM en nuevo host y termina de operar en host de origen



Virtualización de aplicaciones

- Emulación de plataforma
  - Permite crear máquinas virtuales con un hardware distinto del nativo
  - Lento, debe traducir todas las instrucciones del hardware destino al nativo, emular periféricos, etc
  - Ejemplo: QEMU
- Virtualización de aplicaciones
  - Aumento de portabilidad (diferentes SO y arquitecturas de hardware)
  - Ej: Java VM, .NET CLR

Virtualización de aplicaciones



Sistemas distribuidos

- Similares a las multicomputadoras en que cada nodo tiene:
  - Su propia memoria privada
  - Sin memoria física compartida
- Acoplamiento mas débil

Cada nodo es una computadora entera

Sistemas distribuidos

Elemento	Multiprocesador	Multicomputadora	Sistema distribuido
Configuración de nodo	CPU	CPU, RAM, interfaz de red	Computadora completa
Periféricos de nodo	Todos compartidos	Compartidos, excepto tal vez el disco	Conjunto completo por nodo
Ubicación	Mismo bastidor	Mismo cuarto	Posiblemente a nivel mundial
Comunicación entre nodos	RAM compartida	Interconexión dedicada	Red tradicional
Sistemas operativos	Uno, compartido	Varios, igual	Posiblemente todos distintos
Sistemas de archivos	Uno, compartido	Uno, compartido	Cada nodo tiene el suyo
Administración	Una organización	Una organización	Muchas organizaciones

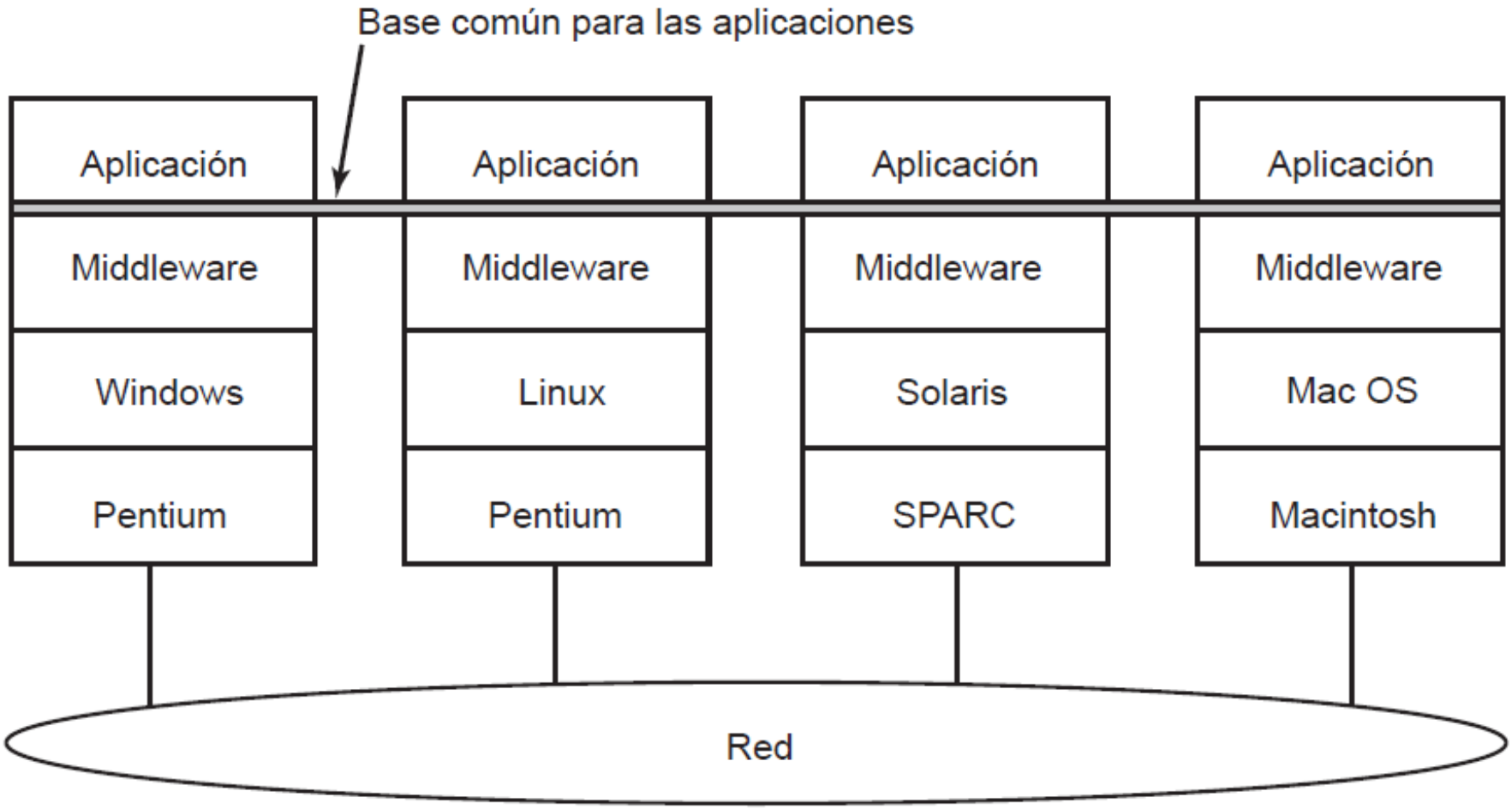
Middleware

- Las aplicaciones comunes de Internet incluyen el **acceso a computadoras remotas** (mediante el uso de *telnet*, *ssh* y *rlogin*).
- El acceso a **información remota** (mediante el uso de World Wide Web y FTP, el Protocolo de Transferencia de Archivos)
- **Comunicación de persona** a persona (mediante el correo electrónico y los programas de chat)

El problema con todas estas aplicaciones es que cada una tiene que reinventar la rueda.

- Una de las formas en que un sistema distribuido puede obtener cierta medida de uniformidad frente a los distintos sistemas operativos y el hardware subyacente es tener un nivel de software encima del sistema operativo: **middleware**

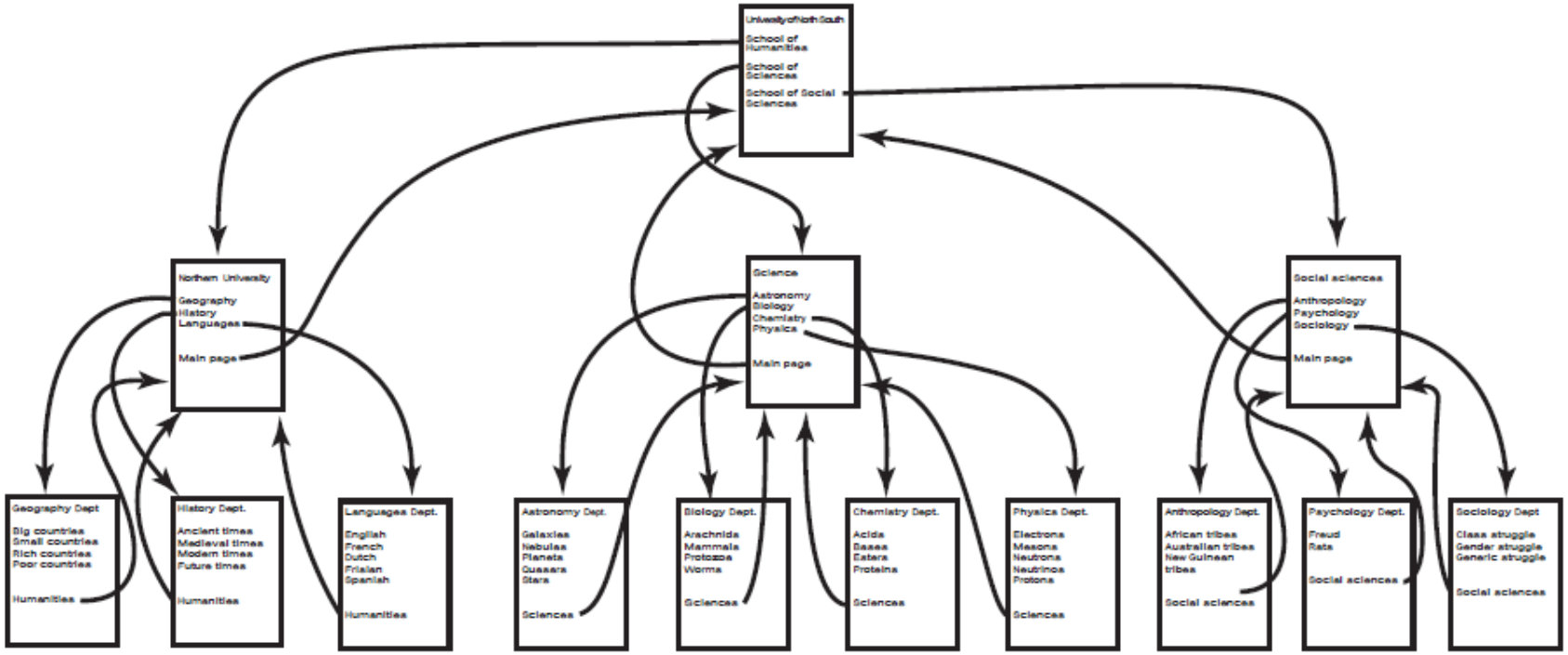
Middleware



En un sentido, el middleware es como el sistema operativo de un sistema distribuido

Middleware basado en documentos

- **World Wide Web**. Tim Berners-Lee inventó el servicio Web en 1989 en CERN, el Centro Europeo de Investigación de Física Nuclear
- Web es un gran grafo dirigido de documentos que pueden apuntar a otros documentos



La idea básica detrás de la Web es hacer que un sistema distribuido tenga la apariencia de una colección gigante de documentos hipervinculados

Middleware basado en sistemas de archivos

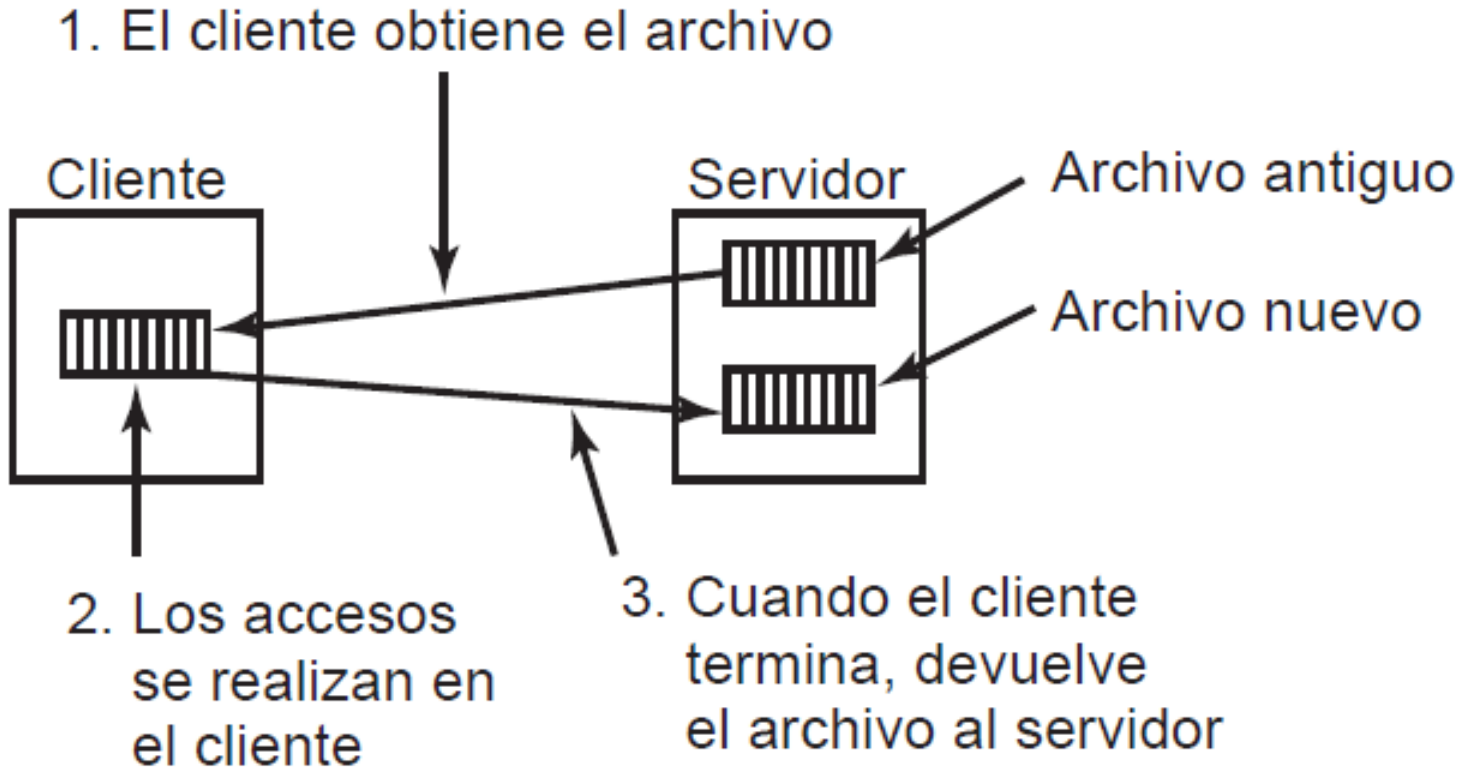
- Un segundo método sería hacer que un sistema distribuido tenga la apariencia de un gran sistema de archivos
- Para lograr la comunicación, un proceso tiene que escribir datos en un archivo y otros procesos tienen que leer los datos de vuelta.

- **Modelo de transferencia**
  - Modelo de envío/descarga
  - Modelo de acceso remoto

Middleware basado en sistemas de archivos

Modelo de envío/descarga:

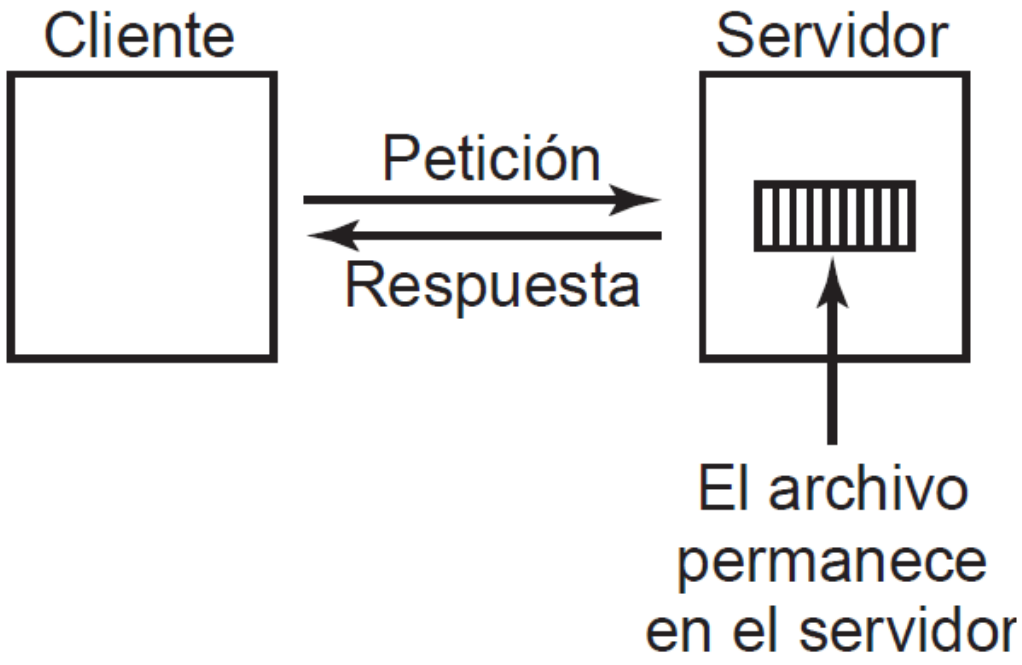
- Lectura -> Local (Mayor rendimiento)
- Escritura -> Local y luego se actualiza en el servidor



Middleware basado en sistemas de archivos

Modelo de acceso remoto:

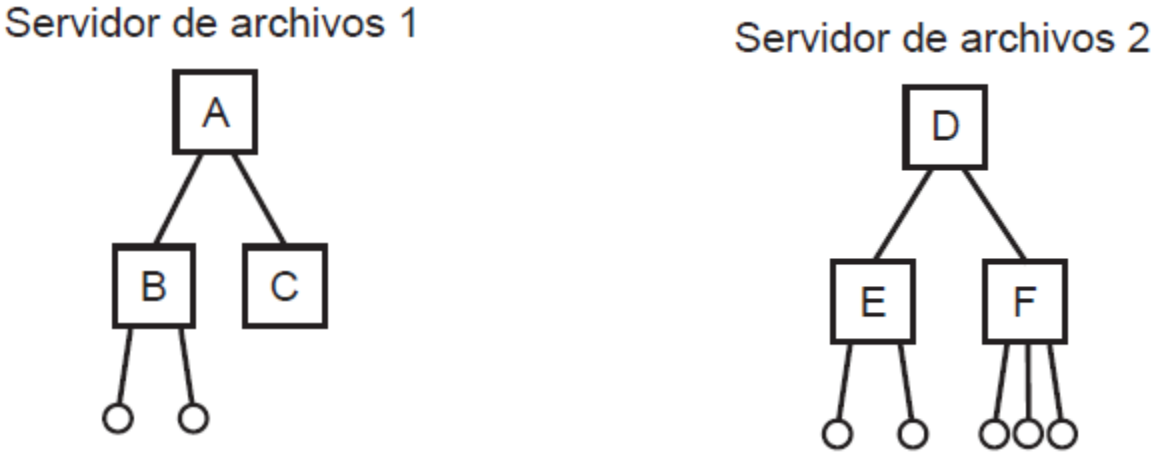
- Archivo -> siempre en el servidor
- Cliente -> envía comandos y hace el trabajo en el servid



Middleware basado en sistemas de archivos

Jerarquía de directorios

El problema es cómo ven los clientes la jerarquía de directorios

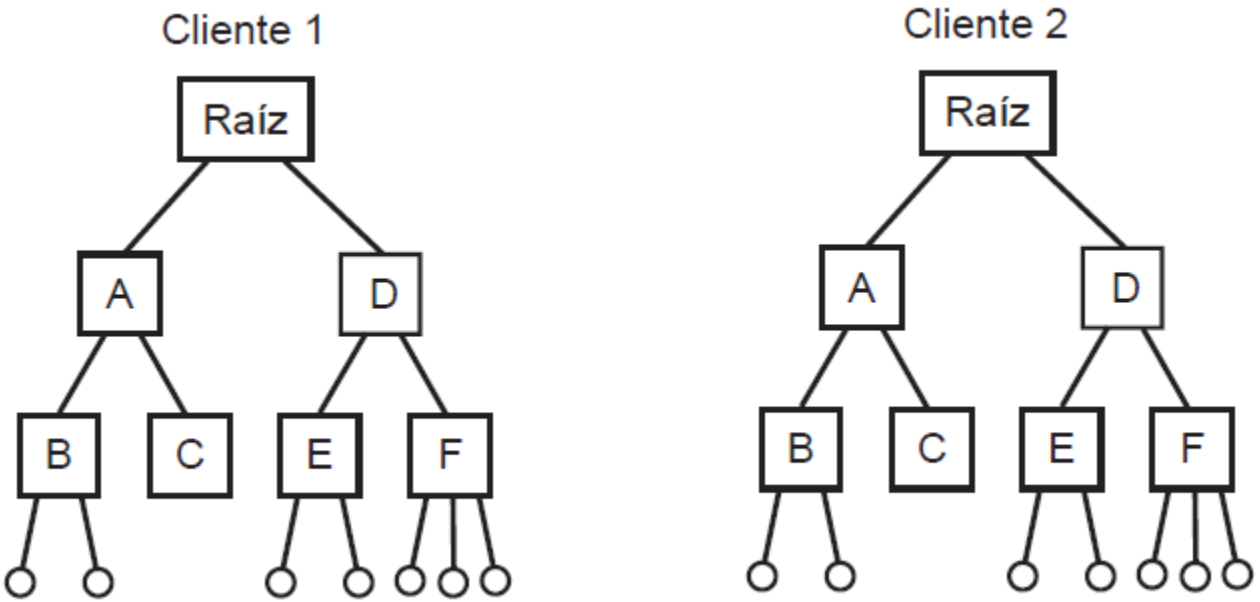


Dos servidores de archivos. Los cuadros son directorios y los círculos son archivos

Middleware basado en sistemas de archivos

Jerarquía de directorios

Los dos clientes ven el mismo árbol



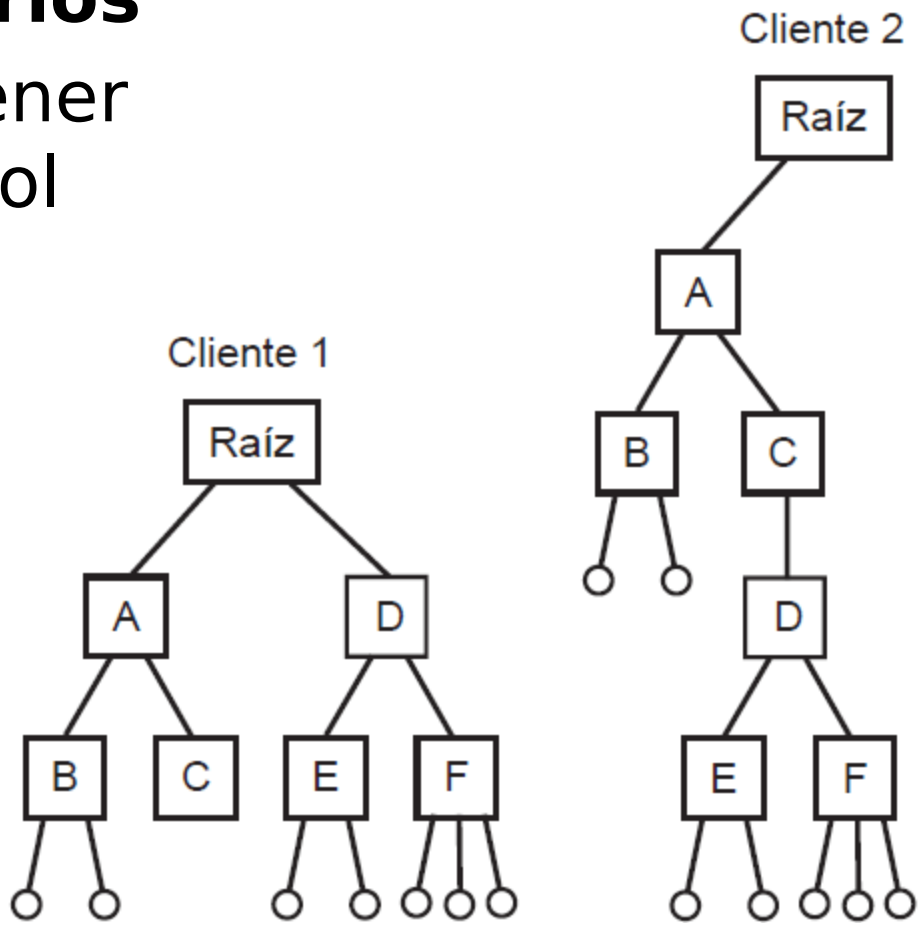
Middleware basado en sistemas de archivos

Jerarquía de directorios

Dos clientes pueden tener distintas vistas del árbol

Problemática:

- Direcciones relativas
- Direcciones absolutas



Middleware basado en sistemas de archivos

Transparencia de nomenclatura:

Transparencia de localización: el nombre de ruta no da ninguna pista sobre la ubicación del archivo.

Ej: */servidor1/dir1/dir2/x*

Indica que está en el servidor1 pero no donde está el servidor.

Independencia de ubicación: un sistema en el que se pueden mover archivos sin cambiar sus nombres

Tres metodos para nombrar archivos y directorios:

1. Usar la nomenclatura de máquina + ruta, **como *`/máquina/ruta`*** o ***`máquina:ruta`*.**
2. Montar los sistemas de archivos remotos en la jerarquía de archivos local.
3. Un solo espacio de nombres que sea igual para todas las máquinas.

Middleware basado en objetos

Objeto es una colección de variables que se incluyen con un conjunto de procedimientos de acceso -> **métodos**

Hasta acá nada nuevo!!

C++ y Java son orientados a objeto a nivel de lenguaje en vez de **objetos en tiempo de ejecución.**

Middleware basado en objetos

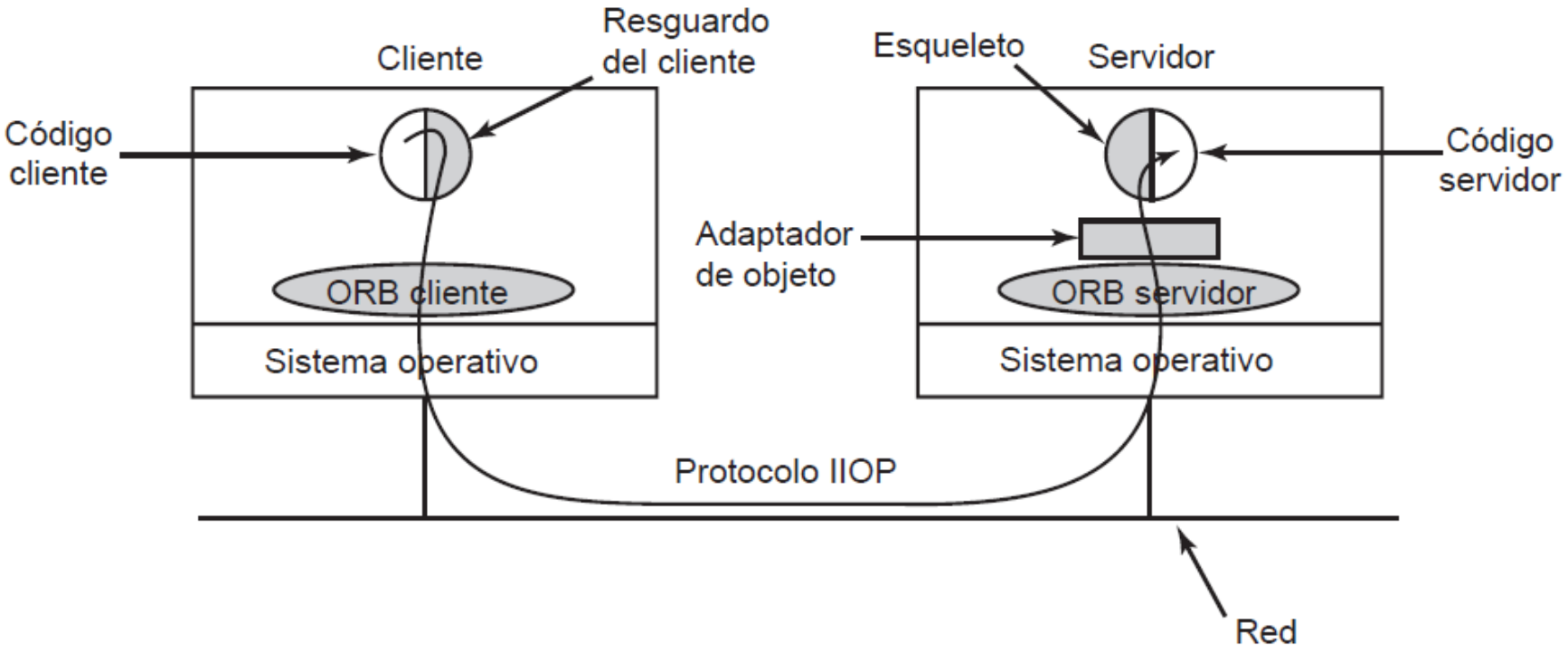
- **CORBA** (Common Object Request Broker Architecture, Arquitectura común de intermediarios en peticiones a objetos) estandarizado por OMG (Object Managment Group)
- Sistema **cliente-servidor**: clientes pueden invocar objetos en máquinas servidores.
- Sistemas heterogéneos (HW /SW / SO).
- Para que un cliente invoque a un servidor se usa: **ORB**s (Intermediarios en peticiones a objetos).
- Objetos se definen en el lenguaje IDL (Interfaz Definition Language).

Middleware basado en objetos

CORBA

Para comunicarse a través de Internet utiliza el protocolo **IIOP** (Internet InterOrb Protocol)

Para utilizar objetos no escritos para CORBA se utiliza **adaptador de objeto**



roblema: si hay muchos accesos a objetos en servidor

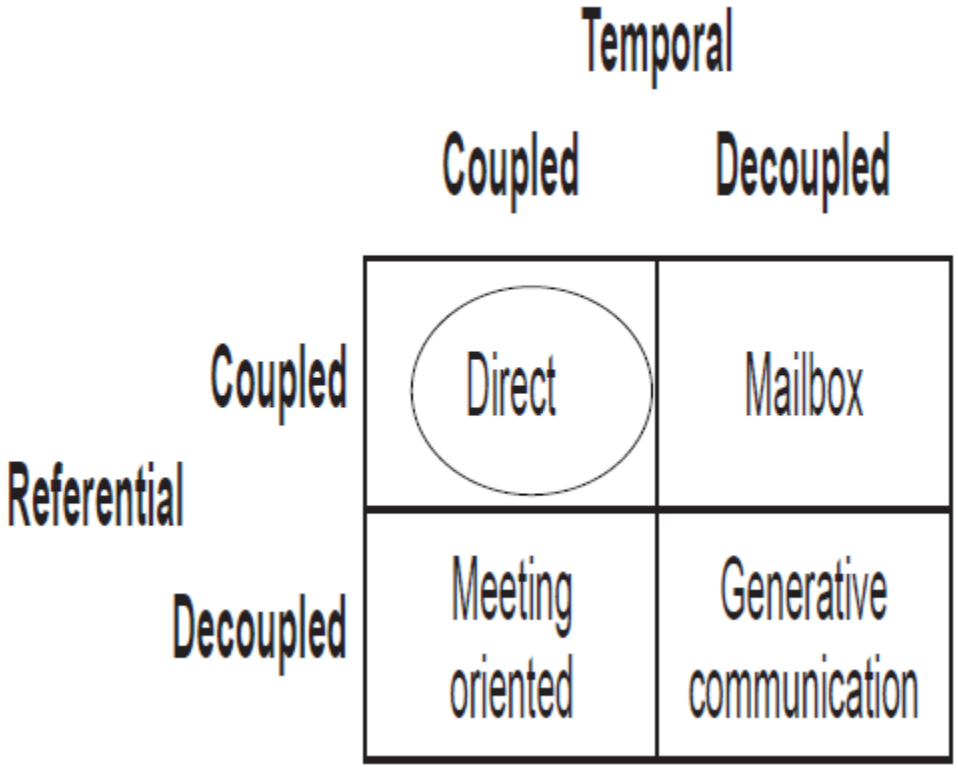
Middleware basado en coordinación

- Coordinación: maneja la comunicación y cooperación entre procesos
- El enfoque recae en cómo ocurre la Coordinación entre procesos.
- **Taxonomía:**
  - **Acoplamiento referencial**: tiene que ver con la forma de hacer referencia explícita en la comunicación. Por ejemplo, un proceso puede comunicarse con otro sólo si conoce su nombre o identificac

— **Acoplamiento temporal:** se refiere a que los procesos que se comunican deben estar funcionando.

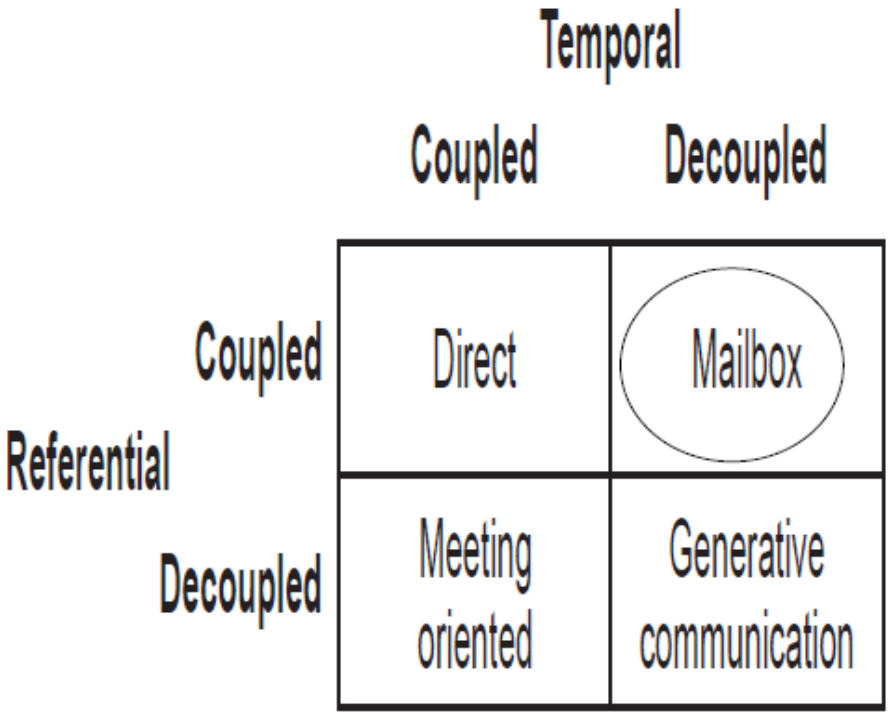
Referential \ Temporal	Coupled	Decoupled
Coupled	Direct	Mailbox
Decoupled	Meeting oriented	Generative communication

Middleware basado en coordinación



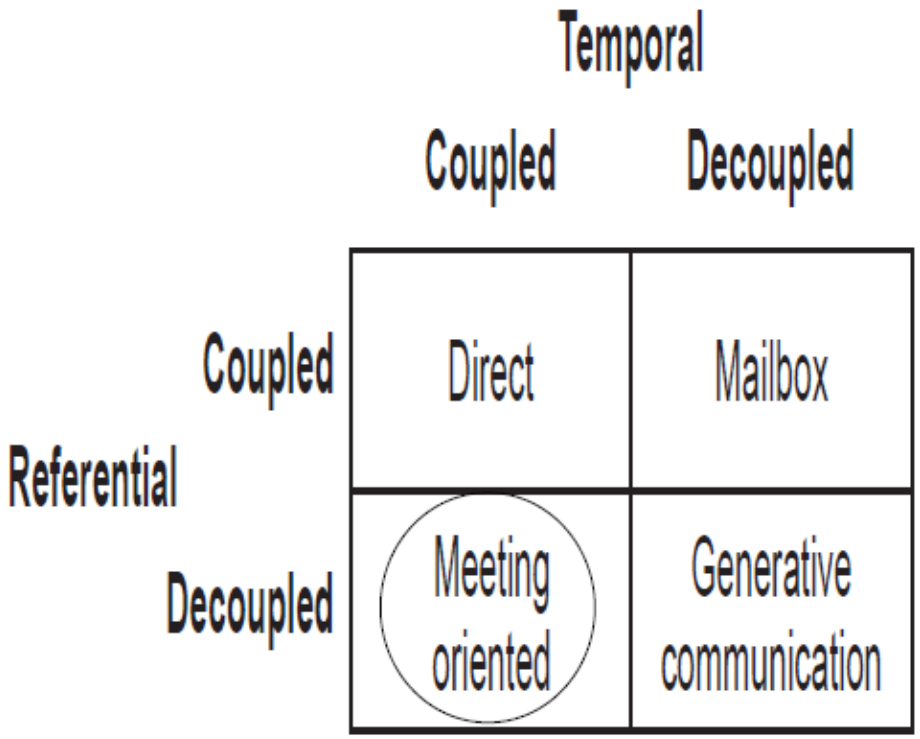
- **Coordinación directa**: los procesos deben estar funcionando y se conocen.

Middleware basado en coordinación



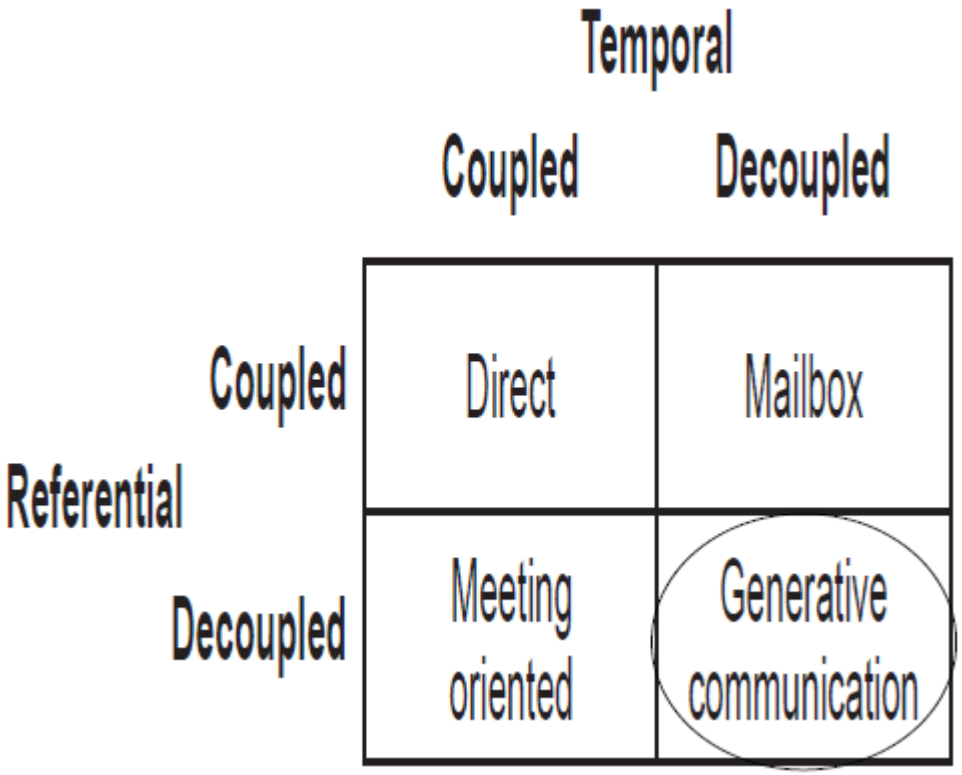
- **Coordinación de Buzon de Correo:** no se requiere que los procesos que se están comunicando funcionen al mismo tiempo para que funcione la comunicación. Ésta ocurre colocando los mensajes en un buzón de correo (posiblemente compartido).

Middleware basado en coordinación



- **Coordinación orientada a reunión:** los procesos no se conocen entre sí. Existe un concepto de reunión en la cual los procesos se agrupan temporalmente para coordinar sus actividades. Según este modelo los procesos sí deben estarse ejecutando al mismo tiempo.

Middleware basado en coordinación



- **Comunicación Generativa:** (Linda) un conjunto de procesos independientes utilizan un espacio de datos persistente, compartido, conformado por tuplas.
- Las **tuplas** son registros de datos etiquetados compuestos por varios campos de entrada.
- Las etiquetas sirven para distinguir entre tuplas que representan diferentes clases de información.

Fin primera parte

GRIDS

Grid

- Suponga millones de computadoras repartidas en todo el mundo.
  - Nada nuevo.
- Suponga esas computadoras conectadas a internet.
  - Nada nuevo.
- Ahora suponga una “herramienta mágica” que haga que todas ellas se vean como una única supercomputadora.
  - Este concepto SI es nuevo ! !

Idea de Grid

- Los sistemas y aplicaciones Grid deben:
  - Integrar
  - Virtualizar
  - Manejar
- Recursos y servicios a través de diferentes ubicaciones geográficas y servir a diferentes usuarios.

Grid - Analogía

- El término Grid se toma de *Electrical Power Grid* (red de distribución de energía eléctrica).
- Cuando conectamos algún dispositivo eléctrico a la red de alimentación no nos preocupamos por saber si la fuente es:
  - Energía hidráulica.
  - Energía eólica.
  - Una central nuclear.
- Nivel de conocimiento más alto (no se conocen, o no es necesario, las capas subyacentes).

Grid - Analogía

- **Red eléctrica**
- Infraestructura:
  - Une las diferentes centrales generadoras y los hogares a través de subestaciones, transformadores y líneas de transmisión.
- Es transparente:
  - No necesita preocuparse de donde viene o como es generada la energía.

- **Grid**
- Infraestructura:
  - Une diferentes recursos de cómputo como ser PCs, workstations, servidores, almacenamiento y provee la manera de acceder a ellos.
- Será transparente:
  - No necesita preocuparse acerca de qué computadora atenderá su requerimiento como así tampoco de qué lugar vienen o son almacenados los datos

Grid - Analogía

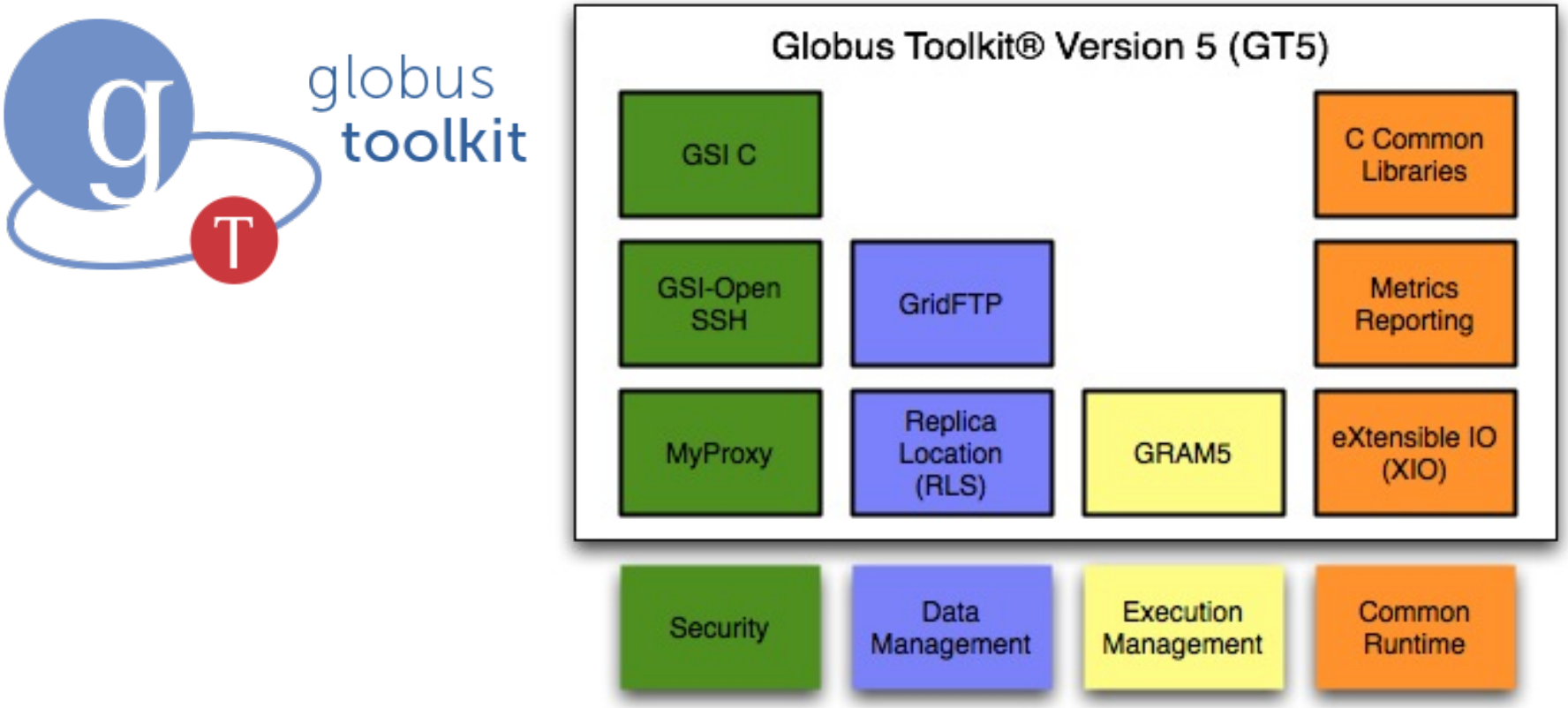
- **Red eléctrica**
- Es penetrante:
  - Está disponible prácticamente en cualquier lado y simplemente se puede acceder a la energía a través de un toma estándar.
- Es un servicio:
  - Se solicita energía, se accede y se paga por ella.

- **Grid**
- Será penetrante:
  - Se podrá acceder a los recursos de cómputo practicamente desde cualquier lugar y desde cualquier plataforma: desktop, laptop, tablets, telefonos móbiles.
- Es un servicio:
  - Se solicitan recursos de cómputo o de almacenamiento, se accede y se paga por ellos aunque es posible hacer uso bajo políticas de acceso

Grid - Middleware

Globus Toolkit:

Provee un marco de trabajo para que los usuarios compartan computadoras, archivos y otros recursos de una manera flexible y segura, sin sacrificar la autonomía local



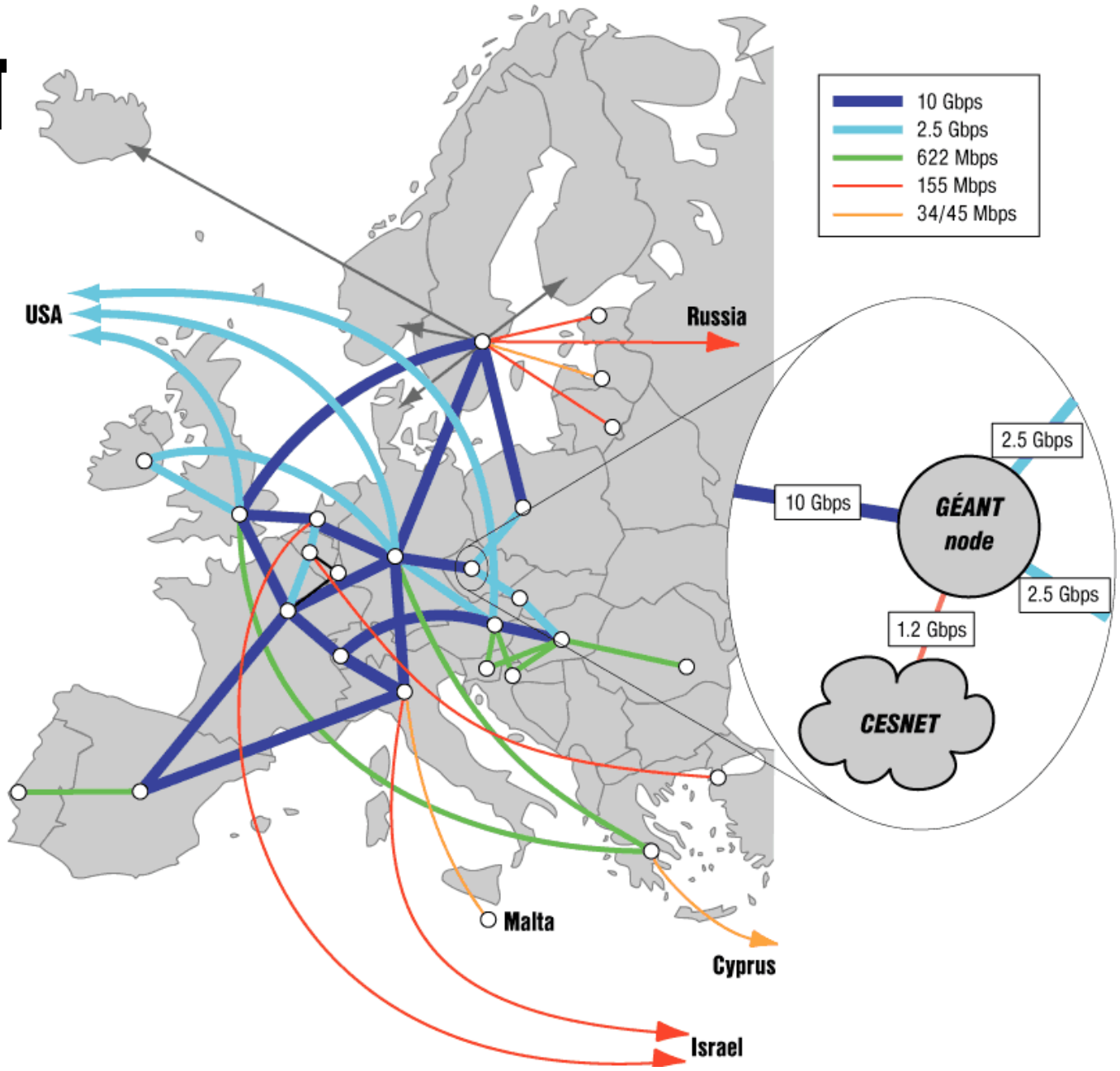
Infraestructura de Grid

- Modelo de capas (infraestructura de red, globus software foundation, middleware, servicios de alto nivel).
- Una infraestructura grid ofrece una capa común para poder integrar plataformas de computo no compatibles por medio de la definición de un conjunto consistente de interfaces para acceder y gestionar recursos compartidos.
- Los servicios grid incluyen, entre otros, descubrimiento y monitorización de recursos, asignación y gestión de recursos, infraestructura de seguridad y transferencia de ficheros.

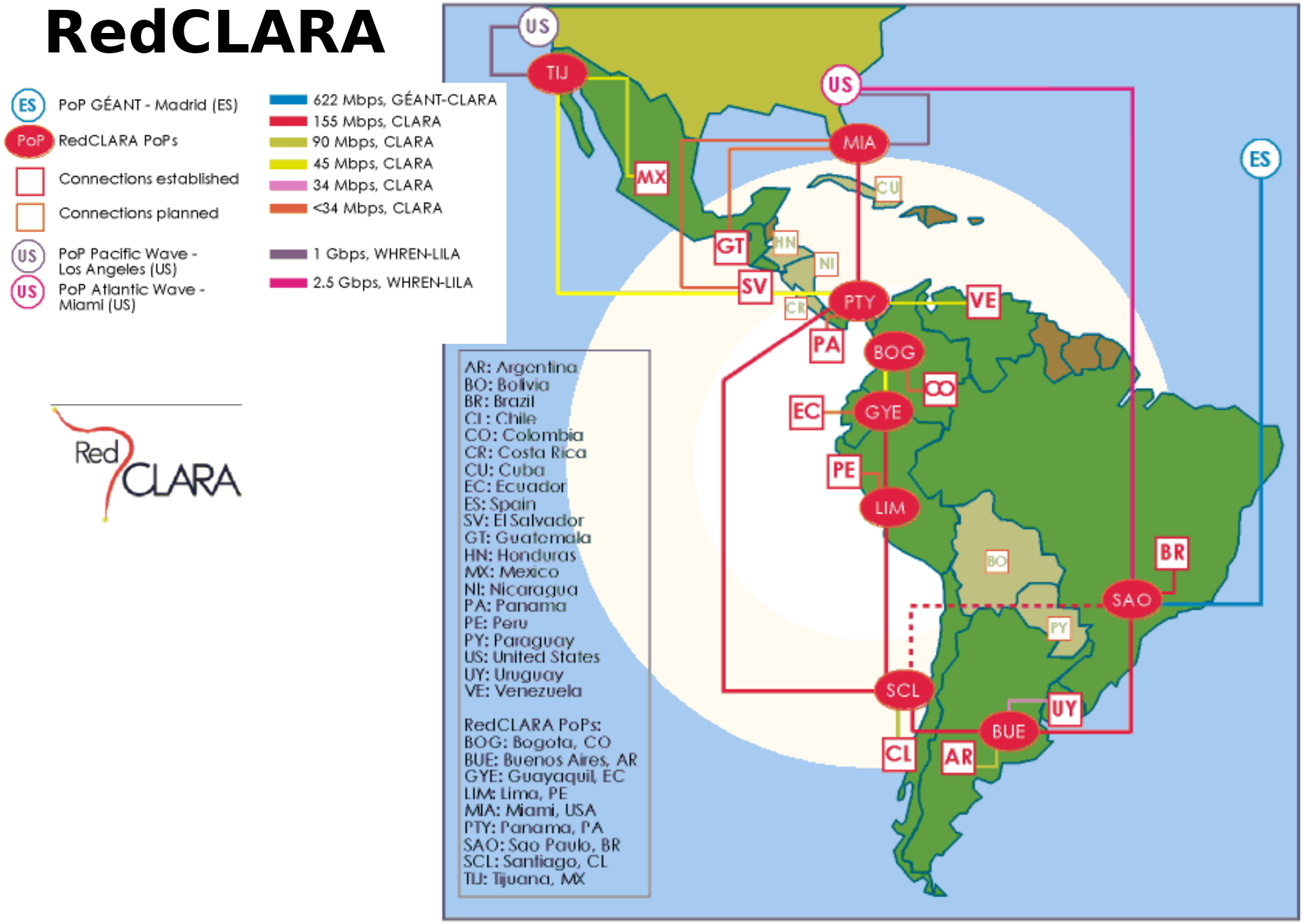
Infraestructura de red

- La infraestructura red es subyacente e indispensable para que funcione cualquier infraestructura grid.
- Por ejemplo:
  - GEANT en Europa.
  - RedCLARA en América Latina.

GÉANT


10 Gbps
2.5 Gbps
622 Mbps
155 Mbps
34/45 Mbps
USA
Russia
Malta
Cyprus
Israel
10 Gbps
2.5 Gbps
GÉANT node
2.5 Gbps
1.2 Gbps
CESNET

RedCLARA


ES PoP GÉANT - Madrid (ES)
PoP RedCLARA PoPs
Connections established
Connections planned
US PoP Pacific Wave - Los Angeles (US)
US PoP Atlantic Wave - Miami (US)
622 Mbps, GÉANT-CLARA
155 Mbps, CLARA
90 Mbps, CLARA
45 Mbps, CLARA
34 Mbps, CLARA
<34 Mbps, CLARA
1 Gbps, WHREN-LILA
2.5 Gbps, WHREN-LILA
AR: Argentina
BO: Bolivia
BR: Brazil
CL: Chile
CO: Colombia
CR: Costa Rica
CU: Cuba
EC: Ecuador
ES: Spain
SV: El Salvador
GT: Guatemala
HN: Honduras
MX: Mexico
NI: Nicaragua
PA: Panama
PE: Peru
PY: Paraguay
US: United States
UY: Uruguay
VE: Venezuela
RedCLARA PoPs:
BOG: Bogota, CO
BUE: Buenos Aires, AR
GYE: Guayaquil, EC
LIM: Lima, PE
MIA: Miami, USA
PTY: Panama, PA
SAO: Sao Paulo, BR
SCL: Santiago, CL
TIJ: Tijuana, MX
US
TIJ
US
MIA
ES
MX
CU
GT
HN
NI
SV
CR
PTY
VE
PA
BOG
EC
GYE
PE
LIM
BO
BR
SAO
PY
SCL
UY
BUE
CL
AR

Links sobre Grids

Research Papers from Globus Alliance Members

http://toolkit.globus.org/alliance/publications/papers.php

GÉANT

http://www.geant.net/Pages/default.aspx

RedCLARA

http://www.redclara.net/index.php?lang=es

Cloud Computing

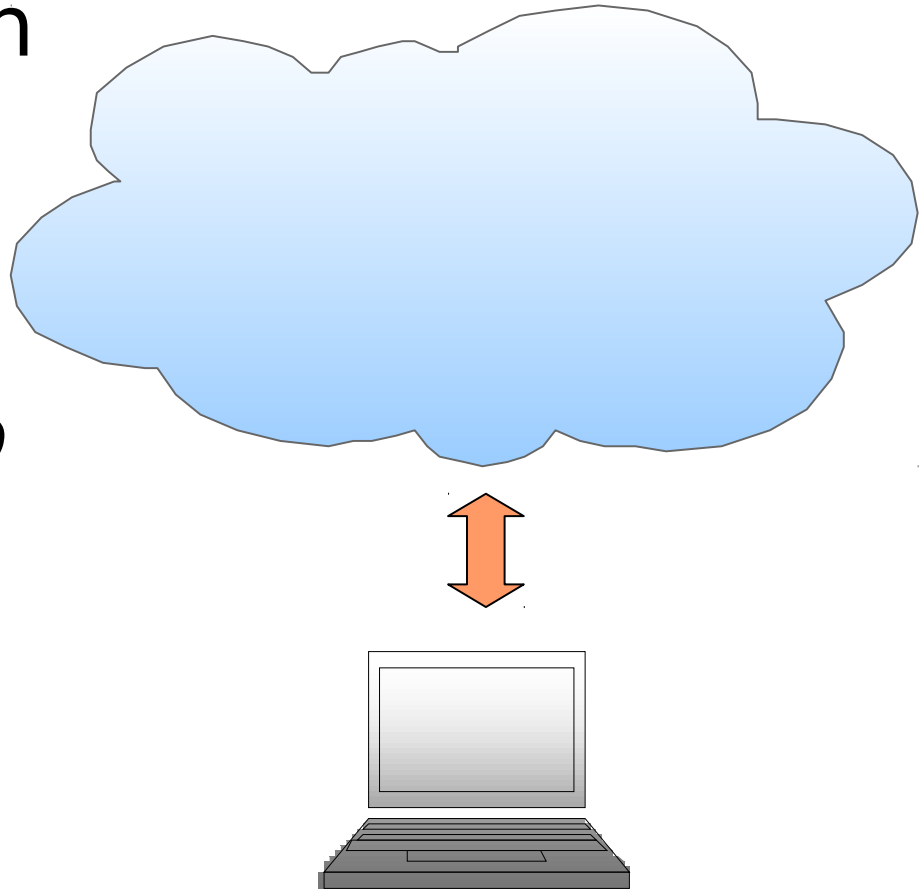
Computación en la nube

Es un modelo para habilitar el acceso a un conjunto de servicios computacionales (ej: redes, servidores, almacenamiento, aplicaciones y servicios) de manera conveniente y por demanda, que pueden ser rápidamente aprovisionados y liberados con un esfuerzo administrativo y una interacción con el proveedor de servicio mínimo.

National Institute of Estándar and technology (NIST)

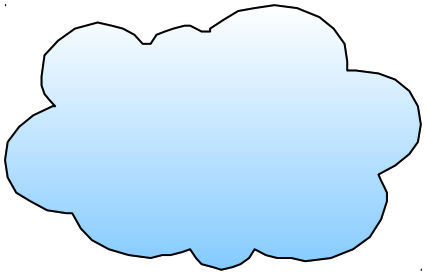
Cloud Computing

Es un paradigma en el cual las capacidades de cómputo y cálculo son ofrecidas *como un servicio*, permitiendo a los usuarios hacer uso de las mismas a través de Internet.



Cloud Computing

¿Es Grid computing?



Cloud -Internet Abstracción de detalles

- *Cloud computing* es un *paradigma*
- Su infraestructura podría ser un Grid
- Los usuarios hacen uso del cómputo y del almacenamiento sin conocer detalles de la infraestructura

Cloud Computing

Ventajas

- Es económico.
- Mejor utilización de los recursos.
- No hay exigencias al cliente en cuanto a ubicación o capacidades.

Desventajas

- Pérdida de control de datos sensibles (para los bancos).
- 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 a la disponibilidad de acceso a Internet

Cloud Computing - Capas

Software como servicio (SaaS)

- Se encuentra en la capa más alta y caracteriza una aplicación completa ofrecida como un servicio.
- Una sola instancia del software que corre en la infraestructura del proveedor y sirve a múltiples organizaciones de clientes
- Accesibles a través de un navegador web

Cloud Computing - Capas

Plataforma como servicio (PaaS)

- Es la encapsulación de una abstracción de un ambiente de desarrollo y el empaquetamiento de una serie de módulos o complementos que proporcionan, normalmente, una funcionalidad horizontal
- APIs preconfiguradas y listas para integrarse sobre una tecnología concreta de desarrollo
- Pueden dar servicio a todas las fases del ciclo de desarrollo y pruebas del software

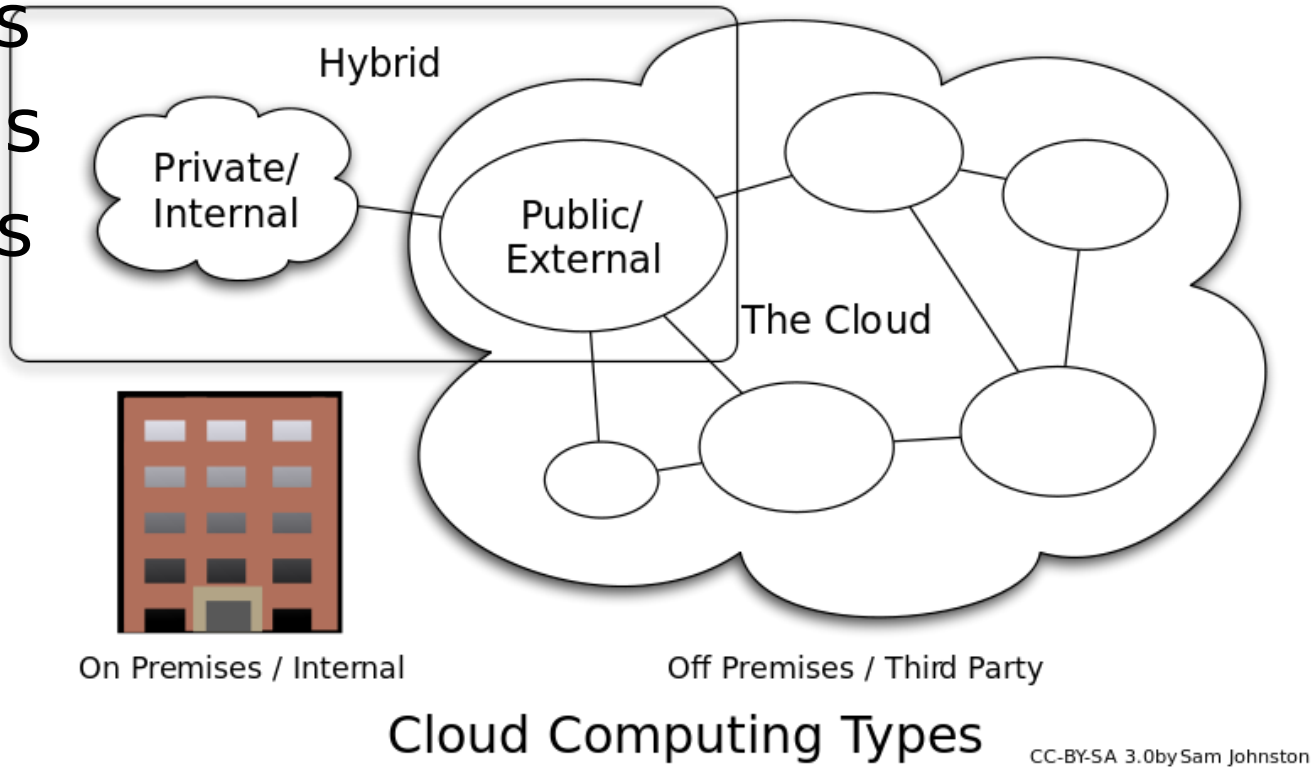
Cloud Computing - Capas

Infraestructura como servicio (IaaS)

- Se encuentra en la capa inferior y es un medio de entregar almacenamiento básico y capacidades de cómputo como servicios estandarizados en la red
- Servidores, sistemas de almacenamiento, conexiones, enrutadores, y otros sistemas

Tipos de nubes

- Las nubes pueden ser:
  - Publicas
  - Privadas
  - Hibridas



Tipos de nubes

- **Nube pública** es una nube computacional mantenida y gestionada por terceras personas no vinculadas con la organización. En este tipo de nubes tanto los datos como los procesos de varios clientes se mezclan en los servidores

- **Nubes privadas** están en una infraestructura bajo demanda gestionada para un solo cliente que controla qué aplicaciones debe ejecutarse y dónde. Son propietarios del servidor, red, y disco y pueden decidir qué usuarios están autorizados a utilizar la infraestructura

- **Híbridas** combinan los modelos de nubes públicas y privadas. Usted es propietario de unas partes y comparte otras, aunque de una manera controlada.

Cloud Computing: PROVEEDORES

SaaS, dSaaS, IaaS

- Amazon (Amazon Web Services, Simple Storage Services, Amazon EC2).

El primero en ofrecer Cloud Computing

Cloud Computing: PROVEEDORES

PaaS, SaaS

- Google (Google Apps Engine, Google Apps).

Gratis hasta:
500MB
5 millones páginas vistas por mes
Usa Python

Cloud Computing: PROVEEDORES

- IBM

IaaS, PaaS

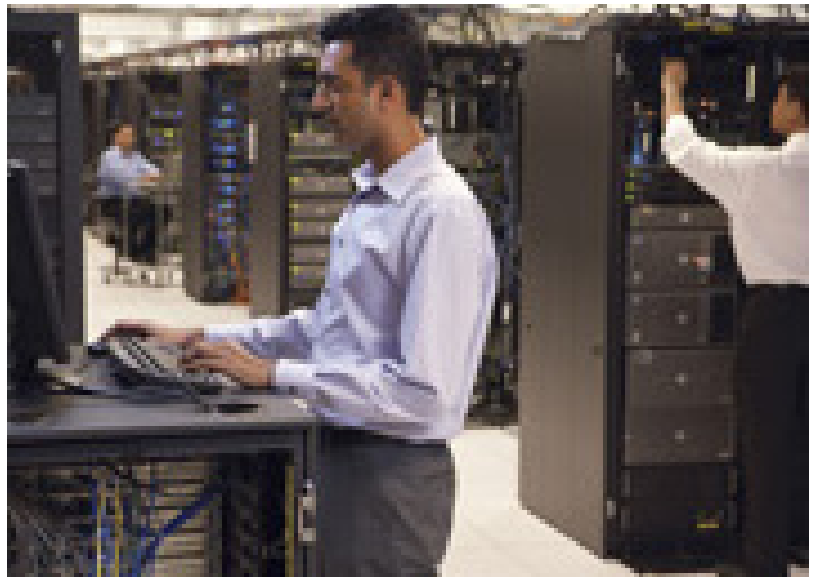
-IBM Blue Cloud Con Linux y Software Hadoop (basado en Nutch y Google's MapReduce),
-IBM IT Factory

IBM Blue Cloud en Alemania

Cloud Computing: PROVEEDORES

IaaS

- DELL



-Dell Cloud Computing Solution

Cloud Computing: PROVEEDORES

IaaS

- SUN

-Sun Grid Engine 6.2

Hasta 63.000 núcleos

Cloud Computing: PROVEEDORES

IaaS

- HP, Intel, Yahoo

Construcción de 6 Cloud

Con servidores HP (de 125 a 500), entre 1000 a 4000 núcleos cada uno corriendo Apache Hadoop.

Infocomm Development Authority of Singapore

University of Illinois at Urbana-Champaign

Karlsruhe Institute of Technology (KIT) in Germany

Cloud Computing: PROVEEDORES

- ORACLE

IaaS, PaaS,SaaS

-Asociado con
Amazon
AWS
Para el uso de
Oracle Software y
Oracle DataBase

Cloud Computing: PROVEEDORES

- Microsoft

Software + Servicios

SaaS

Microsoft Online Services

"suite online de productividad para las empresas en el 2009"

Live Mesh

Con Live Desktop SaaS

Cloud Computing: CLIENTES

- Universidades (ej: **Arizona State University**)
- General Electric.
- L'Oreal.
- Procter & Gamble.
- Valeo.

Bibliografía

- Cap 8, Tanembaum “Sistemas Operativos modernos
- The NIST Definition of Cloud Computing