DEPARTAMENTO DE

INGENIERÍA INFORMÁTICA

UNIVERSIDAD DE SANTIAGO DE CHILE

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

PROCESOS, HEBRAS Y VIRTUALIZACIÓN

13169

SISTEMAS DISTRIBUIDOS

Procesos

- Concepto de Sistemas Operativos.
- Un proceso es la instancia de un programa en ejecución.
  - Un programa no es un proceso hasta que no se ejecuta.
- Los SO basan gran parte de sus funciones en el manejo y planificación de procesos.
  - Mantiene una tabla con los procesos del sistema e información relevante.
- El uso de procesos en Sistemas Distribuidos, conjunto a programación multihilos, permite traslapar la comunicación y el procesamiento de las aplicaciones.

Procesos

- Los SO aseguran que los procesos independientes no afecten a otros procesos.
- Existe una fuerte interacción entre los SO y el Hardware de la máquina.
  - El SO reserva recursos (segmentos, periféricos, etc).
  - Uso de la CPU para ejecutar el proceso.
  - Manejo de interrupciones.
- Interacción **transparente**, pero **costosa**.
- Los Procesos son importantes a la hora de hablar de Sistemas Distribuidos, sin embargo, es mejor hablar de Hebras (Threads).

Hebras

- Al igual que un proceso, las hebras ejecutan su propio segmento de código.
- A diferencia de procesos, no se hace un esfuerzo por transparentar la concurrencia si aquello resulta en una degradación del rendimiento.
  - En general, el trabajar con hilos produce un menor **overhead** del sistema.
- Los hilos y, en particular, una aplicación multihilos produce una ganancia en el rendimiento de la aplicación y permite modulizar las operaciones del proceso.

Hebras en Sistemas no Distribuidos

- La principal ganancia de utilizar múltiples hilos es el no bloqueo de un proceso ante una llamada al sistema.
- Consideremos el trabajo con una hoja de calculo.
  - Si tuviera un solo hilo, cualquier modificación sobre una celda que implique grandes series de cálculo bloquearía todo el proceso.
  - Como alternativa, se trabaja con un hilo para procesador los datos, otro para ingresar datos e inclusive otro que mantenga copias y respaldos en caso de fallas.

Hebras en Sistemas no Distribuidos

- Otra ventaja de la tecnología multihilos es que hace posible explotar el paralelismo cuando ejecutamos el programa dentro de una máquina con múltiples procesadores.
    - A cada hilo se le asigna una CPU diferente y se permite el uso de memoria compartida para comunicación e interacción.
- En general las aplicaciones grandes son construidas como una interacción de distintos programas cooperativos.
- La cooperación se implementa mediante mecanismos de comunicación interprocesos.
    - En general la comunicación se implementa mediante pipes, colas de mensajes y segmentos de memoria compartidos.
    - **Problema:** Excesivo cambio de contexto.

Hebras en Sistemas no Distribuidos

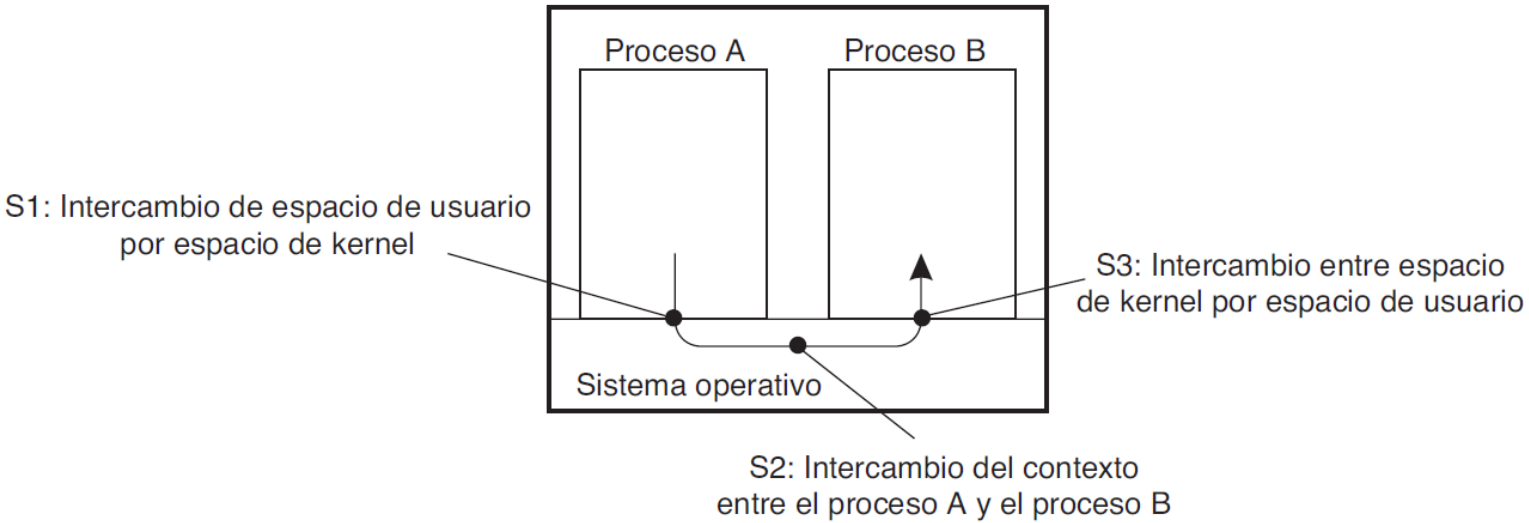

Figura 3-1. Intercambio de contexto como resultado de IPC.

Hebras en Sistemas no Distribuidos: Implementación

- En general se implementa un paquete de hilos.
  - Pthread
  - OpenMP
  - MPI
- Operaciones para crear, destruir y sincronizar mediante mutex y variables de condición.
- Implementación en dos modos:
  - **Modo Usuario.**
  - **Modo Kernel.**

Hebras en Sistemas no Distribuidos: Implementación

- **Hebras modo Usuario**
  - Bajo costo de creación y destrucción.
  - Cambios de contexto poco costoso (almacenamiento de registros).
  - Hilos a nivel de usuario son una “ilusión”. Si se bloquea un hilo se bloquea todo el proceso.
- **Hebras modo Kernel**
  - Creación, destrucción y otras actividades requieren uso del kernel, lo cual implica una llamada al sistema.
  - Cambios de contexto costoso.

Hebras en Sistemas no Distribuidos: Implementación

- Forma híbrida entre hebras modo usuario y hebras modo kernel.

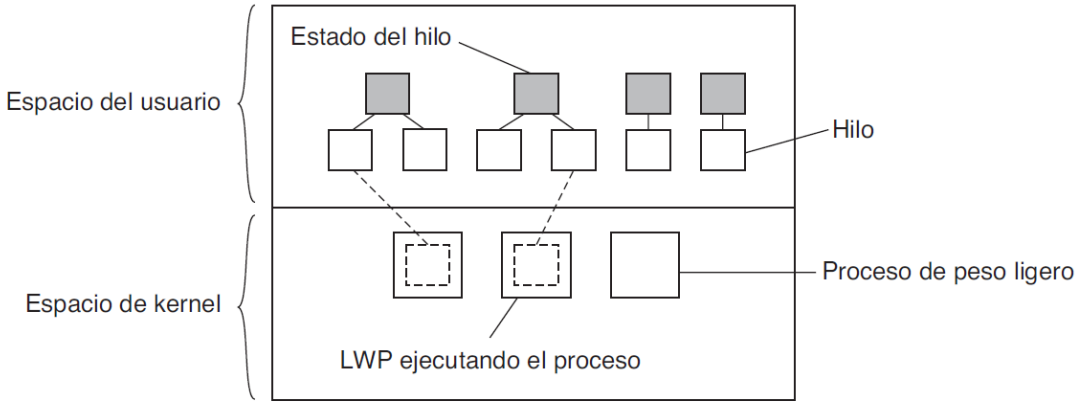

Figura 3-2. Combinación de procesos a nivel de kernel de peso ligero e hilos de nivel usuario.

Hebras en Sistemas Distribuidos

- Una propiedad importante de los hilos es que pueden proporcionar un medio conveniente para permitir llamadas de bloqueo de sistema sin bloquear todo el proceso que ejecuta el hilo.
- Gracias a esto, es más fácil expresar la comunicación mediante múltiples conexiones lógicas al mismo tiempo.
- Veamos un acercamiento mediante clientes y servidores multihilos.

Hebras en Sistemas Distribuidos: Clientes

- Los SD que operan en redes de área amplia necesitan conciliar grandes tiempos de propagación de mensajes interproceso para establecer un alto grado de transparencia.
- En WAN, los ciclos tienen retrasos de cientos de milisegundos, o incluso de segundos en algunos casos.
- La forma común de ocultar la latencia es iniciar la comunicación y proceder de inmediato con otra cosa.

Hebras en Sistemas Distribuidos: Clientes

- Un ejemplo común: Los navegadores Web.
- Página web consta de un archivo HTML, con texto plano, imágenes e iconos, etc.
- El navegador debe configurar una conexión TCP/IP, leer los datos de entrada y pasarlos hacia la componente de visualización.
    - Configurar la conexión y leer datos de entrada es bloqueante.
    - Al considerar transporte de grandes volúmenes de datos, se tiene la desventaja que el tiempo necesario para completar cada operación puede ser relativamente largo.

Hebras en Sistemas Distribuidos: Clientes

- El navegador inicia la recuperación del archivo y luego la despliega.
- Para ocultar latencias tanto como sea posible, los navegadores despliegan los datos mientras van llegando.
- Mientras el texto se pone a disposición del usuario, el navegador continúa con la recuperación de otros archivos que conforman la página, como las imágenes por ejemplo.
- El usuario no necesita esperar a que se recupere todo el archivo y puede navegar en la medida que llegan los datos.

Hebras en Sistemas Distribuidos: Clientes

- En efecto el navegador se comporta como una aplicación multihilos, separado las tareas de conexión y recuperación de la información en múltiples hilos.
- Otro beneficio es abrir múltiples conexiones en paralelo.
- En muchos casos, los servidores están replicados en distintas máquinas, donde cada servidor tiene exactamente la misma información.
- Se pueden generar conexiones en paralelo y traer distintas partes de la página en paralelo.
    - Se obtiene una disminución en los tiempos de carga del usuario.
    - Disminuye el flujo de un solo servidor.

Hebras en Sistemas Distribuidos: Servidor

- El principal uso de las hebras se hace del lado del servidor.
- Simplifica la implementación de los servidores a nivel de código.
- Hace sencillo el desarrollo de servidores que explotan el paralelismo para lograr un alto rendimiento.

Hebras en Sistemas Distribuidos: Servidor

Consideremos un **Servidor de Archivos**

- Dicho servidor se tiene que bloquear en espera del disco.
- El servidor espera consultas sobre un archivo, para procesarla y enviar la respuesta de regreso.
- Se puede implementar un **modelo dispatcher/worker.**
  - También se le llama **broker** al dispatcher.

Hebras en Sistemas Distribuidos: Servidor

- Trabajador realiza una lectura bloqueante en el servidor.
- Bloque la hebra hasta que se recuperan los datos del disco.
- Si se suspende, otra hebra es seleccionada.

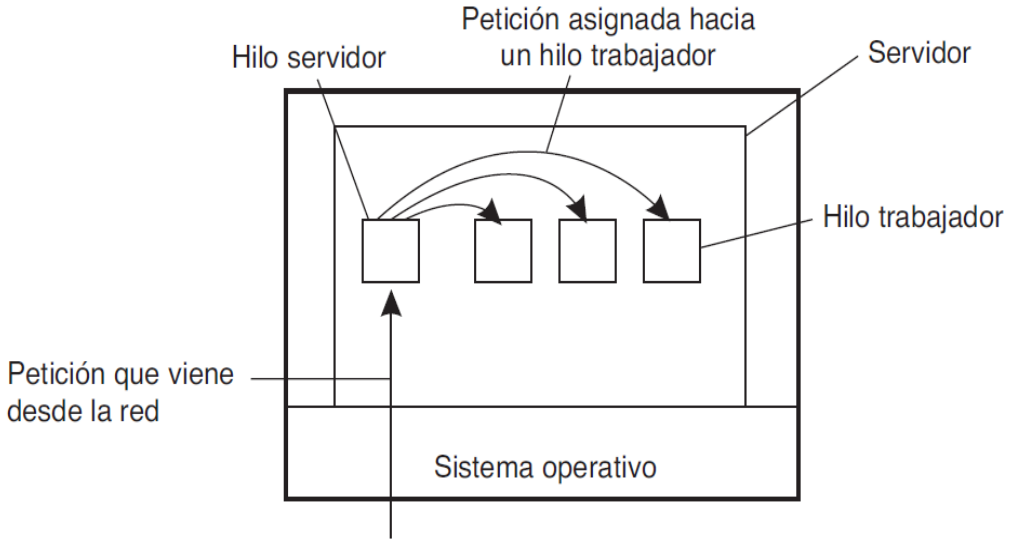

Figura 3-3. Servidor multihilos organizado en un modelo servidor/trabajador.

Resumiendo

- Las hebras permiten conservar la idea del procesamiento secuencial y sus llamadas bloqueantes, a la vez que permite explotar paralelismo.
- Las llamadas bloqueantes hacen más fácil la programación y el paralelismo permite aumentar el rendimiento.
- Sirven tanto a nivel de cliente como de servidor.
- Mejora **tiempos de espera** y **tiempos de procesamiento**.

Virtualización

- Los procesos e hilos permiten hacer más cosas al mismo tiempo.
- Permiten construir parte de programas que parecen ejecutarse en forma simultánea.
    - Independiente de la cantidad de CPUs, se genera una “ilusión” de paralelismo gracias al rápido intercambio entre hilos.
- La separación entre tener una sola CPU y pretender que existen más unidades de procesamiento se puede extender a otros recursos.
- **Virtualización de recursos.**

Virtualización: Sistemas Distribuidos

- Cada sistema distribuido ofrece una interfaz hacia un nivel de software más alto.
    - Distintos tipos de interfaces, abarcando desde instrucciones básicas hasta una colección de interfaces.
    - Sistemas de middleware.
- En su esencia, la virtualización trata con la extensión o el reemplazo de una interfaz existente de modo que imite el comportamiento de otro sistema.

Virtualización: Sistemas Distribuidos

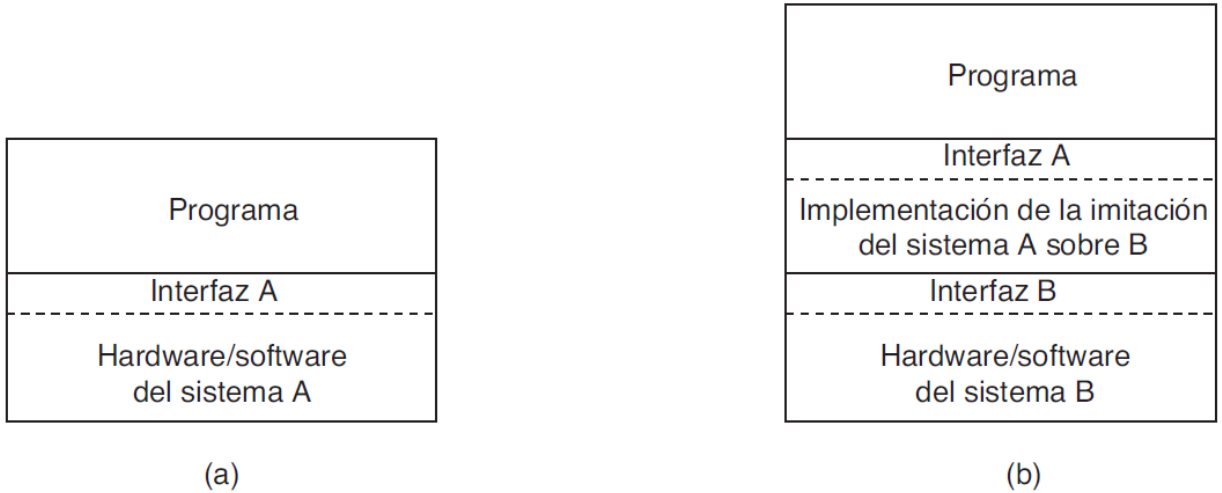

Figura 3-5. (a) Organización general entre un programa, la interfaz y el sistema. (b) Organización general entre el sistema de virtualización A sobre el sistema B.

Virtualización: Sistemas Distribuidos

Virtualización es la creación a través de software de una versión virtual de algún recurso tecnológico, como puede ser una plataforma de hardware, un sistema operativo, un dispositivo de almacenamiento u otros recursos de red.

Virtualización: Sistemas Distribuidos

- La virtualización se introdujo fuertemente durante la década de los 70.
- Principalmente, permitía heredar software (**legacy software**) para ser ejecutado sobre el costoso hardware.
- Se tenía capacidad de virtualizar aplicaciones o, inclusive, Sistemas Operativos completos.
- Cuando los recursos computacionales se masifican y, por tanto, se vuelven más baratos, la virtualización decae.

Virtualización: Sistemas Distribuidos

- En la década de los 90 gana nuevamente terreno.
- El desarrollo de las nuevas tecnologías impacta en los cambios frecuentes a nivel de Hardware y Software a bajo nivel.
- Software y Hardware de alto nivel de abstracción (middleware y aplicaciones) son muchos más estables.
    - **Problema**: Software heredado no se puede mantener al mismo ritmo de las plataformas.
- Virtualización ayuda aportando interfaces heredadas hacia nuevas plataformas.

Virtualización: Sistemas Distribuidos

- Las redes son, hoy por hoy, completamente masivas.
- Esto implica que los administradores de sistemas manejan una colección de servidores y aplicaciones muy grande y heterogéneos.
- Se espera que los recursos sean de fácil acceso para las aplicaciones.
- La virtualización ayuda, ya que cada aplicación se puede ejecutar en su propia máquina virtual, incluyendo bibliotecas relacionadas y el SO particular.

Virtualización: Arquitectura de Máquinas Virtuales

En general, los sistemas de cómputo ofrecen cuatro tipos de interfaz distinto:

1. Una interfaz entre el hardware y el software, constituido por **instrucciones de máquina** que se pueden invocar desde cualquier programa.

2. Una interfaz entre el hardware y el software, constituida por instrucciones máquina que se pueden invocar solamente desde programas privilegiados, tales como los sistemas operativos.

3. Una interfaz que consta de **llamadas de sistema** como las que ofrece un sistema operativo.

4. Una interfaz que consta de llamadas a bibliotecas, las cuales forman, por lo general, lo que conocemos como **interfaz de programación de aplicaciones (API**, por sus siglas en inglés). En muchos casos, las llamadas de sistema ya mencionadas están ocultas por una API.

Virtualización: Arquitectura de Máquinas Virtuales

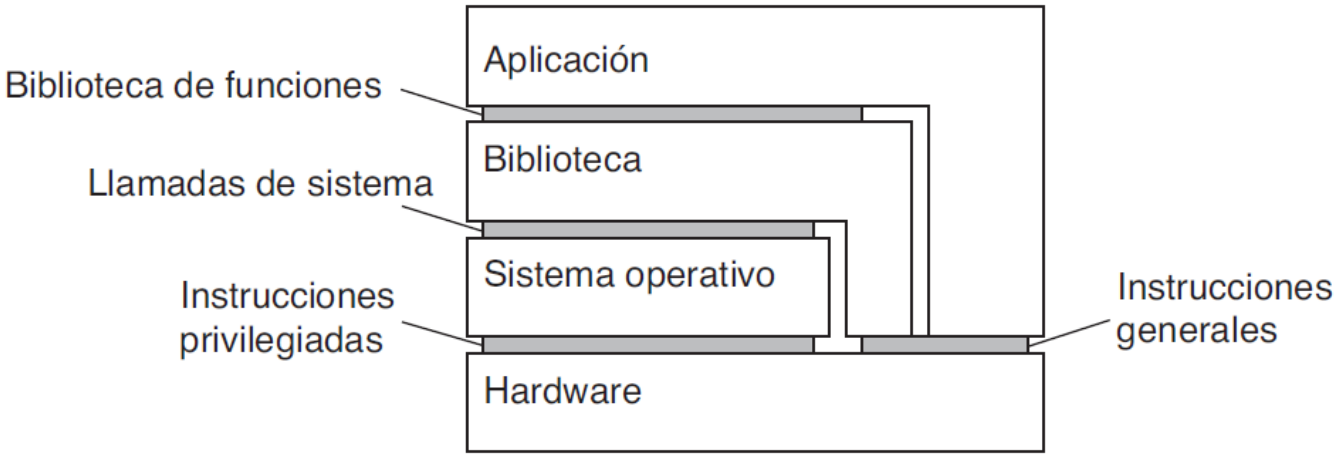

Figura 3-6. Distintas interfaces ofrecidas por los sistemas de cómputo.

Virtualización: Arquitectura de Máquinas Virtuales

Máquina virtual de proceso → La MV imita el comportamiento de las llamadas del sistema (JRE por ejemplo).

Monitor de la máquina virtual (VMM) → La MV cubre por completo el HW. Posibilidad de tener distintos SO ejecutándose de forma concurrente.

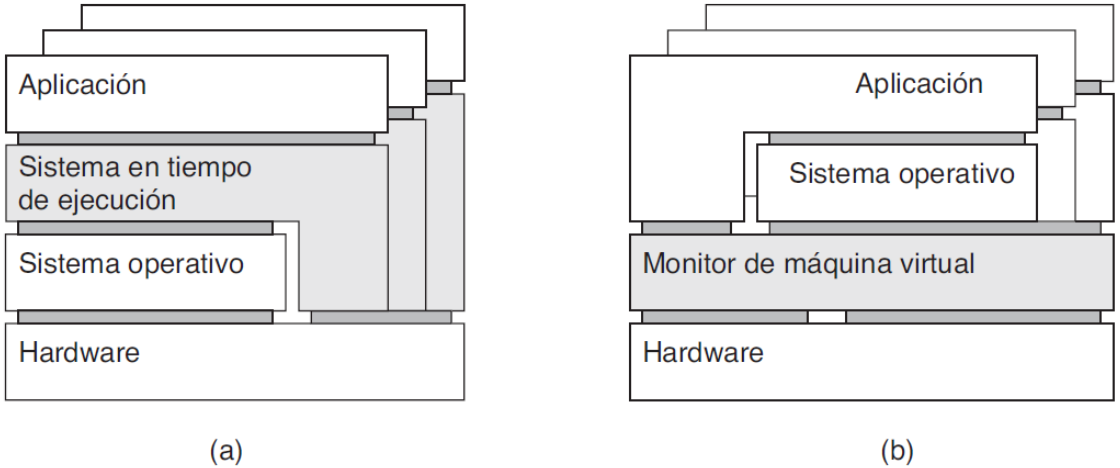

Figura 3-7. (a) Un proceso en una máquina virtual, con múltiples instancias de (aplicación, tiempo de ejecución) combinaciones. (b) Un monitoreo en una máquina virtual, con múltiples instancias de (aplicaciones, sistema operativo) combinaciones.

Virtualización: Arquitectura de Máquinas Virtuales

- Las VMM son cada vez más importantes en el contexto de la confiabilidad y la seguridad para los sistemas distribuidos.
  - Permiten el aislamiento de toda una aplicación y de su ambiente.
- Una falla ocasionada por un error o un ataque a la seguridad no afectará a una máquina en su totalidad. Además, la portabilidad mejora de manera importante
  - Las VMM proporcionan un desacoplamiento posterior entre hardware y software, lo cual permite mover un ambiente completo desde una máquina a otra.

Virtualización: VirtualBox

- Software de virtualización para arquitecturas x86/amd64.
- Permite la instalación de SO adicionales.
  - Se conocen como “sistemas invitados” dentro de un SO “anfitrión”.
- Permite la ejecución de VM de manera remota.

Virtualización: VirtualBox

Virtualización: Cloud Computing

- La virtualización es la base para el Cloud Computing.
  - Paradigma que permite ofrecer servicios de computación a través de una red, la cual es usualmente internet.
- Disponibilidad por demanda de recursos del sistema.
  - Especialmente en datos y capacidad de cómputo.
- Se ofrece en tres posibles modalidades:

1. Software como Servicio (SaaS).
2. Plataforma como Servicio (Paas).
3. Infraestructura como Servicio (Iaas).

Virtualización: Cloud Computing

El Cloud Computing ofrece distintas características:

- Autoservicio bajo demanda.
- Acceso amplio y ubicuo a toda la red.
- Ubicación transparente y agrupación de recursos .
- Rápida elasticidad → Acorde a la demanda.
- Servicio monitoreado → Ofrece estadísticas de uso.
- Autorreparable.
- Disponibilidad de información → No requiere copias en dispositivos físicos propios.
- Rendimiento.
- Seguridad → Mejora por tener todo centralizado.
- Mantenimiento → Es más sencillo al no necesitar instalar aplicaciones en el computador de cada usuario.

Virtualización: Cloud Computing

Software como Servicio

- Se encuentra en la capa más alta.
- Caracteriza una aplicación completa ofrecida como un servicio bajo demanda.
- **Multitenencia** → Una sola instancia de la aplicación corre en la infraestructura del proveedor que sirve a múltiples organizaciones de clientes.
- Las aplicaciones se acceden a través de un navegador Web.

Virtualización: Cloud Computing

Software como Servicio: Google Drive

- Servicio de alojamiento de archivos.
- De los más conocidos del mundo.
- Unificado con otros servicios como Gmail y Google +.
- Ofrece servicios distintos acorde al espacio máximo de almacenamiento.

Virtualización: Cloud Computing

Plataforma como Servicio

- Se encuentra en la capa del medio.
- Encapsulación de una abstracción de un medio ambiente de desarrollo.
    - Por ejemplo, un sistema Linux, un servidor Web y un ambiente Ruby.
- Sirve para todas las fases del proyecto, tanto para el desarrollo como testeo de la aplicación.

Virtualización: Cloud Computing

Plataforma como Servicio: Google App Engine

- Servicio de alojamiento web de Google.
- Presta servicio en dos módulos:
  - Google App Standard Environment: Solo tiene posibilidad de programación con C#, Java, Python, PHP y GO. No permite accesos a disco usando estos lenguajes, a cambio, los costes son más económicos.
  - Google App Flexible Environment : Puede implementar cualquier lenguaje de programación mediante contenedores Docker, permitiendo mayor control sobre la aplicación. Los costes son proporcionales al consumo de CPU.

Virtualización: Cloud Computing

Infraestructura como Servicio

- Se encuentra en la capa inferior.
- Entrega almacenamiento básico y capacidades de cómputo como servicio estandarizado.
- A través de la virtualización, se logra manejar tipos específicos de carga de trabajo.

Virtualización: Cloud Computing

Infraestructura como Servicio: Amazon Web Services

- Provee plataformas por demanda para computo en la nube.
- Servicio “pay-as-you-go”.
- Las máquinas virtuales de AWS emulan CPUs, GPUs, RAM, SSD, un SO, entre otros elementos que selecciona el usuario.
- Es la plataforma de mayor uso (33%) en la nube, seguida de Microsoft Azure (18%) y Google Cloud (18%).

Virtualización: Cloud Computing

Ventajas

- Integración probada de servicios Red.
- Prestación de servicios a nivel mundial.
- Permite prescindir de instalar cualquier software de manera local.
- Portabilidad de la información.
- Implementación rápida y con menos riesgos.
- Actualizaciones automáticas.
- Contribuye el uso eficiente de energía.

Virtualización: Cloud Computing

Desventajas

- La centralización de las aplicaciones y almacenamiento origina una interdependencia de los proveedores de servicio.
- Disponibilidad sujeta a disponibilidad de internet.
- Riesgo de “monopolio” de servicios.
- Servicios altamente especializados toman mucho tiempo en estar disponibles en la red.
- Seguridad. La información circula de distintos modos y cada uno de ellos es un foco de inseguridad.
- Escalabilidad a largo plazo. Muchas aplicaciones sobre la misma infraestructura.

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Departamento de Ingeniería Informática
Universidad de Santiago de Chile

¿CONSULTAS?