Clase del día - 09/03/2021

**Objetivos de los sistemas distribuidos**

Como vimos la clase anterior los sistemas distribuidos tienen grandes ventajas sobre los sistemas centralizados.

Sin embargo, los sistemas distribuidos también tienen algunas desventajas que podemos resumir en su alta complejidad y costo. Por esta razón, es muy importante establecer claramente los objetivos de un sistema distribuido antes de su implementación.

En general, un sistema distribuido deberá cumplir los siguientes objetivos:

1. Facilidad en el acceso a los recursos.
2. Transparencia.
3. Apertura.
4. Escalabilidad.

**1. Facilidad en el acceso a los recursos**

Es de la mayor importancia en un sistema distribuido facilitar a los usuarios y a las aplicaciones el acceso a los recursos remotos. Entendemos como recurso el CPU, la memoria RAM, las unidades de almacenamiento, las impresoras, los DBMS, los archivos, o cualquier otra entidad lógica o física que preste un servicio en el sistema distribuido.

En un sistema distribuido se comparten los recursos por razones técnicas y por razones económicas.

En el primer caso, se comparten recursos por **razones técnicas** cuando tenemos procesos que ejecutan en forma distribuida utilizando datos que se encuentran distribuidos geográficamente, o bien, procesos que requieren la distribución del cálculo en diferentes CPUs, o procesos de facturación que envían la impresión de facturas a múltiples impresoras.

En el segundo caso, se comparten recursos por **razones económicas** debido a su alto costo.

Por ejemplo, la virtualización permite compartir los recursos de una computadora como son el CPU, la memoria, y las unidades de almacenamiento, creando entornos de ejecución llamados máquinas virtuales. La virtualización aumenta el porcentaje de utilización de los recursos de la computadora y con ello se obtiene un mayor beneficio dado el costo de los recursos.

Sin embargo, compartir recursos conlleva un compromiso en la seguridad, ya que será necesario implementar mecanismos de **comunicación segura** (SSL, TLS o HTTPS), esquemas para la confirmación de la identidad (**autenticación**) y esquemas de permisos para el acceso a los recursos (**autorización**).

**2. Transparencia**

La transparencia es la capacidad de un sistema distribuido de presentarse ante los usuarios y aplicaciones como una sola computadora.

Tipos de transparencia

Podemos dividir la transparencia de un sistema distribuido en siete categorías:

**2.1 Transparencia en el acceso a los datos**. Un sistema distribuido deberá proveer de una capa que permita a los usuario y aplicaciones acceder a los datos de manera estandarizada. Por ejemplo, un servicio que permita acceder a los archivos que residen en computadoras  con diferentes sistemas operativos mediante nombres estandarizados, independientemente del tipos de nomenclatura implementada en cada sistema operativo.

**2.2 Transparencia de ubicación**. En un sistema distribuido los usuarios acceden a los recursos independientemente de su localización física. Por ejemplo, una URL identifica un recurso en la Web de manera única. Así, https://m4gm.com/moodle/curso.txt es la URL del archivo "curso.txt" localizado en el directorio "moodle" de la computadora cuyo dominio es "m4gm.com". Adicionalmente, la URL indica el protocolo que utilizará para acceder el recurso, en este caso el protocolo es HTTPS.

**2.3 Transparencia de migración**. En alguno sistemas distribuidos es posible migrar recursos de un sitio a otro. Si la migración del recurso no afecta la forma en que se accede el recurso, se dice que el sistema soporta la transparencia de migración. Por ejemplo, si un sistema permite la migración transparente de procesos de una computadora a otra como una estrategia de tolerancia de fallas, los usuarios y procesos no se verán afectados ante la migración del proceso que acceden.

**2.4 Transparencia de re-ubicación**. La transparencia de re-ubicación se refiere a la capacidad del sistema distribuido de cambiar la ubicación de un recurso mientras está en uso, sin que el usuario que accede el recurso se vea afectado. Por ejemplo, en UNIX (Linux) para cambiar la ubicación de un proceso en ejecución, primero se le envía al proceso un signal SIGSTOP en la ubicación de origen, el proceso se migra a la ubicación de destino, se envía al proceso un signal SIGCONT en la ubicación de destino, entonces el proceso sigue ejecutando desde el punto en que se quedó.

**2.5 Transparencia de replicación**. La transparencia de replicación es la capacidad del sistema distribuido de ocultar la existencia de recursos replicados. Por ejemplo, la repilcación de los datos es una estrategia que permite aumentar la confiabilidad y la rendimiento en los sistemas distribuidos.

**2.6 Transparencia de concurrencia**. En una computadora todos los recursos son compartidos. La transparencia de concurrencia se refiere a la capacidad de un sistema de ocultar el hecho de que varios usuarios y procesos comparten los diferentes recursos. Por ejemplo, un sistema operativo multi-tarea oculta el hecho de que varios procesos utilizan de manera concurrente el CPU, la memoria, los discos duros, etc. Por otra parte, un sistema operativo multi-usuario oculta el hecho de que la computadora es utilizada por múltiples usuarios de manera concurrente.

**2.7 Transparencia ante fallas**. La transparencia ante fallas es la capacidad del sistema distribuido de ocultar una falla. Como vimos anteriormente, la distribución del procesamiento permite implementar sistemas tolerantes a fallas. Por ejemplo, en un sistema que se encuentra totalmente replicado, si el sistema principal falla entonces el usuario accederá de manera transparente a la réplica del sistema. Más adelante en el curso veremos cómo replicar un sistema completo en la nube.

**3. Apertura**

Un sistema abierto es aquel que ofrece servicios a través de reglas de sintaxis y semántica estándares.

Las reglas de sintaxis generalmente se definen mediante un **lenguaje de definición de interfaz**, en el cual se especifica los nombres de las operaciones del servicio, nombre y tipo de los parámetros, valores de retorno, posibles excepciones, entre otros elementos que sean de utilidad para automatizar la comunicación entre el cliente del servicio y el servidor..

La semántica (funcionalidad) de las operaciones de un servicio se define generalmente de manera informal utilizando lenguaje natural.

Los sistemas abiertos exhiben tres características que los hacen más populares que los sistemas propietarios, estas características son: interoperabilidad, portabilidad y extensibilidad.

La definición de las reglas de sintaxis permite que diferentes sistemas puedan interaccionar. A la capacidad de sistemas diferentes de trabajar de manera interactiva se le llama **interoperabilidad**. Por ejemplo, un servicio web escrito en Java, Python o en C# puede ser utilizado indistintamente por un cliente escrito en JavaScript, Java, Python, o C#.

La **portabilidad** (*cross-platform*)de un programa se refiere a la posibilidad de ejecutar el programa en diferentes plataformas sin la necesidad de hacer cambios al programa. Por ejemplo un programa escrito en Java puede ser ejecutado sin cambios en cualquier plataforma que tenga instalado el JRE (Java Runtime Environment). En 1995 Sun Microsystems explicó la portabilidad de los programas escritos en Java con la siguiente frase: "*[Write once, run everywhere](https://www.computerweekly.com/feature/Write-once-run-anywhere)*".

La **extensibilidad** se refiere a la capacidad de los sistemas de crecer mediante la incorporación de componentes fáciles de reemplazar y adaptar. Más adelante en el curso veremos cómo desarrollar sistemas extensibles mediante objetos de Java distribuidos.

**4. Escalabilidad**

La **escalabilidad**es la capacidad de un sistema de crecer sin reducir su calidad.

Un sistema puede escalar en tres aspectos principales: tamaño, geografía y administración.

4.1 Escalar en tamaño

Cuando un sistema requiere atender más usuarios o ejecutar procesos más demandantes, es necesario agregar más CPUs, más memoria, mas unidades de almacenamiento o incrementar el ancho de banda de la red. Es decir, el sistema requiere escalar en tamaño.

Sin embargo, un sistema centralizado solo puede crecer hasta alcanzar el número máximo de CPUs, la cantidad máxima de memoria RAM, el número máximo de controladores de disco duro y el máximo ancho de banda que puede ofertar el ISP (*Internet Service Provider*).

4.2 Escalar geográficamente

En la actualidad las empresas globales requieren operar sus sistemas en múltiples regiones geográficas. Si la empresa cuenta solamente con un sistema central, los usuarios tendrán que conectarse desde ubicaciones remotas por lo que se incrementará los tiempos de respuesta debido a la latencia de la red.

Entonces surge la necesidad de escalar geográficamente los sistemas, por tanto será necesario instalar servidores en diferentes ubicaciones estratégicamente localizadas con el fin de reducir los tiempos de respuesta. Generalmente se divide el mundo en regiones (América del Norte, América del Sur, Europa, Asia, África) y se instala un centro de datos en cada región. Si la región es de alta demanda (como es el caso de América del Norte y Europa) se suele instalar más centros de datos en la misma región.

4.3 Escalar la administración

Cuando un sistema crece en tamaño y geografía, también aumenta la complejidad en la administración del sistema.

Un sistema más grande implica más computadoras, más CPUs, más tarjetas de memoria RAM, más unidades de almacenamiento, más concentradores de red, en suma, más componentes que pueden fallar, más información que se tiene que respaldar, más usuarios, más permisos que controlar, etc. En resumen, para crecer el sistema se requiere escalar así mismo la administración.

**Técnicas de escalamiento**

Ahora veremos brevemente algunas técnicas utilizadas para escalar los sistemas.



1. Ocultar la latencia en las comunicaciones

La latencia en las comunicaciones es el tiempo que tarda un mensaje en ir del origen al destino. Existen múltiples factores que influyen en la latencia de las comunicaciones, como son el tamaño de los mensajes, la capacidad de los enrutadores, la distancia, la hora del día, la época del año, etc.

La latencia en las comunicaciones aumenta el tiempo de espera cuando se hace una petición a un servidor remoto.

Una estrategia que se utiliza para ocultar la latencia en las comunicaciones, es el uso de **peticiones asíncronas**.

Supongamos que una aplicación realiza una petición a un servidor cuándo el usuario presiona un botón, si la petición es sincrona el usuario debe esperar a que el servidor envíe la respuesta, ya que la aplicación no puede ejecutar otra tarea mientras espera.

Por otra parte, si la petición es asíncrona, la aplicación puede ejecutar otras tareas. Por ejemplo, en Android todas las peticiones que se realizan a los servidores deben ser asíncronas, lo cual garantiza que las aplicaciones siguen respondiendo al usuario mientras esperan la respuesta del servidor.

2. Distribución

Una técnica muy utilizada para escalar un sistema es la distribución. Para distribuir un sistema se divide en partes más pequeñas las cuales se ejecutan en diferentes servidores.

Por ejemplo, supongamos que una empresa tiene una plataforma de comercio electrónico en la Web, cuando la empresa comienza a tener operaciones globales surge la necesidad de escalar la plataforma de comercio electrónico, para ello se puede distribuir el sistema en distintos servidores.

3. Replicación

Otra técnica utilizada para escalar un sistema es la replicación de los procesos y de los datos.

Replicar los procesos en diferentes computadoras permite liberar de trabajo las computadoras más saturadas, es decir, balancear la carga en el sistema.

Replicar los datos en diferentes computadoras permite acceder a los datos más rápidamente, debido a que con ello se evitan los cuellos de botella en los servidores.

Para replicar los datos se puede utilizar caches que aprovechen la localidad espacial y temporal de los datos.

Por ejemplo, si un archivo se utiliza con frecuencia, es conveniente tener una copia en una cache local. En el caso de que el archivo sea modificado en el servidor, entonces éste enviará un mensaje de **invalidación de cache**, lo que significa que el archivo deberá ser eliminado de la cache local. Si posteriormente el cliente requiere el archivo, deberá solicitarlo al servidor y con ello contará con el archivo actualizado.

4. Elasticidad

Posiblemente la técnica más interesante para escalar un sistema es la elasticidad en la nube. La **elasticidad** es la adaptación a los cambios en la carga mediante aprovisionamiento y des-aprovisionamiento de recursos en forma automática.

Supongamos un servicio de *streaming*bajo demanda, como es el caso de Netflix. En este tipo de servicio la demanda crece los fines de semana y decrece los días entre semana. Si el proveedor de servicio no aprovisiona los recursos suficientes para atender la demanda del fin de semana, entonces muchos usuarios se quedarán sin servicio.

Por otra parte, si el proveedor del servicio aprovisiona los recursos necesarios para atender a sus usuarios el fin de semana, estos recursos estarán sub-utilizados los días entre semana, lo cual resulta en pérdidas económicas.

Entonces la solución es utilizar la posibilidad que les ofrece la nube para crecer y decrecer los recursos aprovisionados en forma automática.

Más adelante en el curso veremos cómo utilizar la elasticidad en la nube.

Actividades individuales a realizar

En esta actividad el alumno va a crear una máquina virtual con Ubuntu en la nube de Microsoft utilizando su cuenta de Azure for Students.

Es muy importante que el alumno elimine la máquina virtual una vez haya terminado de utilizarla, ya que mantener encendida una máquina virtual genera costo, lo que representa una disminución en el crédito que tiene el alumno como parte del programa Azure for Students.

**Creación de una máquina virtual con Ubuntu**  
Ingresar al portal de Azure en la siguiente URL:

[**https://azure.microsoft.com/es-mx/features/azure-portal/**](https://azure.microsoft.com/es-mx/features/azure-portal/)  
  
1. Dar click al botón "Iniciar sesión".  
  
2. En el portal de Azure seleccionar "Máquinas virtuales".  
  
3. Seleccionar la opción "+Agregar".  
  
4. Seleccionar la opción "+Virtual machine"  
  
5. Seleccionar el grupo de recursos o crear uno nuevo. Un grupo de recursos es similar a una carpeta dónde se pueden colocar los diferentes recursos de nube que se crean en Azure.  
  
6. Ingresar el nombre de la máquina virtual.  
  
7. Seleccionar la región dónde se creará la máquina virtual. Notar que el costo de la máquina virtual depende de la región.  
  
8. Seleccionar la imagen, en este caso vamos a seleccionar Ubuntu Server 18.04 LTS.  
  
9. Dar click en "Seleccionar tamaño" de la máquina virtual, en este caso vamos a seleccionar una máquina virtual con 1 GB de memoria RAM. Dar click en el botón "Seleccionar".  
  
10. En tipo de autenticación seleccionamos "Contraseña".  
  
11. Ingresamos el nombre del usuario, por ejemplo: ubuntu  
  
12. Ingresamos la contraseña y confirmamos la contraseña. La contraseña debe tener al menos 12 caracteres, debe al menos una letra minúscula, una letra mayúscula, un dígito y un carácter especial.  
  
13. En las "Reglas de puerto de entrada" se deberá dejar abierto el puerto 22 para utilizar SSH (la terminal de secure shell).  
  
14. Dar click en el botón "Siguiente: Discos>"  
  
15. Seleccionar el tipo de disco de sistema operativo, en este caso vamos a seleccionar HDD estándar.  
  
16. Dar click en el botón "Siguiente: Redes>"  
  
17. Dar click en el botón "Siguiente: Administración>"  
  
18. En el campo "Diagnóstico de arranque" seleccionar "Desactivado".  
  
19. Dar click en el botón "Revisar y crear".  
  
20. Dar click en el botón "Crear".  
  
21. Dar click a la campana de notificaciones (barra superior de la pantalla) para verificar que la maquina virtual se haya creado.  
  
22. Dar click en el botón "Ir al recurso". En la página de puede ver la direción IP pública de la máquina virtual. Esta dirección puede cambiar cad vez que se apague y se encienda la máquina virtual.  
  
23. Para conectarnos a la máquina virtual vamos a utilizar el programa putty.exe, el cual se puede encontrar en la siguiente URL:  [**https://www.putty.org/**](https://www.putty.org/)

Nota. Para enviar (o recibir) archivos desde Windows a la máquina virtual se puede utilizar el programa psftp.exe, el cual se descarga de la misma página donde se descarga putty.exe, ver: **[What is PSFTP?](https://puttygen.com/psftp)**

24. Ejecutar el programa putty.exe

25. Escribir la dirección IP de la máquina virtual en el campo "Host Name". Dar click en el botón "Open". Putty despliega una ventana de alerta de seguridad preguntando si la huella digital del servidor es correcta, dar click al botón "Si".  
  
26. Ingresar el nombre del usuario, en este caso: ubuntu (ver el paso 11). Ingresar la contraseña.  
  
  
**Detener una máquina virtual**  
Cuando una máquina virtual no se utiliza es conveniente detenerla con el fin de reducir el costo. Para detener una máquina virtual:  
  
1. Dar click en la opción "Detener" en el portal de Azure.  
2. Dar click en el botón "Aceptar".  
  
Esperar a que el estado de la máquina virtual sea "Desasignada".  
  
**Encender una máquina virtual**  
Para encender una máquina virtual  
  
1. Seleccionar la opción "Iniciar" en la página de la máquina virtual dentro del portal de Azure.  
  
Esperar a que el estado de la máquina virtual sea "En ejecución".  
  
**Eliminar una máquina virtual**  
Para eliminar una máquina virtual:  
  
1. Seleccionar la opción "Eliminar" en la página de la máquina virtual dentro del portal de Azure.  
2. Dar clieck en el botón "Aceptar".  
  
Los recursos asociados (discos, IP pública, interfaz de red, grupo de seguridad de red, etc.) no se eliminarán, para eliminarlos se deberá seleccionar cada recurso y eliminarlos manualmente.  
  
Para eliminar los recursos asociados a una máquina virtual previamente eliminada:  
  
1. Dar click al icono de "hamburguesa" (las tres líneas horizontales) localizado en la parte superior izquierda de la pantalla.  
2. Seleccionar "Todos los recursos".  
3. Seleccionar cada recursos (dar click en cada checkbox)  
4. Seleccionar "Eliminar".  
5. Verificar la lista de recursos a eliminar.  
6. Escribir la palabra Sí (con acento en la i).  
7. Dar click en el botón "Eliminar".