Resumen unidad II. Modelos y Tipos de modelos

Modelos fisicos:

Representacion del hardware subyacente de un SD.

Existen 3 generaciones de SD:

- *Primeros*: surgieron entre los 70 y 80 en respuesta a las emergentes LAN (Ethernet). De 10 a 100 nodos conectados por medio de una LAN. Eran relativamente homogeneos.
- *Escala INTERNET*: surgieron en los 90 en respuesta al crecimiento de internet. Nodos interconectados en una red de redes. Completamente heterogeneo.
- *Contemporaneos*: aparicion de la computacion movil (comps personales, moviles, etc). Computacion en la nube.

Modelos arquitectonicos:

En termino de tareas computacionales y comunicacion. Inlcuye ordenadores individuales o conjuntos y sus interconexiones. Abstrae y simplifica las funciones de los componentes individuales y considera posteriormente: la ubicacion y las relaciones entre componentes.

```
aplicacion y servicios
    | Middleware
                          <-
        | sistema operativo   | plataforma
            | hardware        |
                          <-
```

Plataforma: nivel de hardware y capa baja de software. Proporcionan servicios a las capas superiores y proporcionan una interfaz de programacion del sistema.

Middleware: Es una capa de software que abstrae y enmascara la heterogeneidad de un sistema. Proporciona bloques utiles para la construccion de software que pueda trabajar con otros SD.

Roles y responsabilidades

Consideraciones:

- Particion de datos
- Uso de cache para datos en clientes y proxys
- El uso de Codigo y agentes moviles
- Requisitos para aadir o eliminar dispositivos.
- No hay tiempo global
- Toda la comunicacion se realiza por mensajes

Modelo cliente servidor

La mas importante y mas utilizada historicamente. Un cliente hace peticiones a un servidor. Los servidores pueden hacerse peticiones entre si, por lo que un servidor puede ser cliente y servidor al mismo tiempo.

La mayoria de servicios de internet son clientes del servicio DNS (Domain Name Service) que traduce nombres de dominio a direcciones de red.

Servicios proporcionados por multiples servidores

- *Particion de datos*: cada servidor web administrta su propio conjunto de recursos y datos.
- *Replicacion*: Se utiliza para aumentar las prestaciones, disponibilidad y tolerancia a fallos de un servicio. Multiples copias de los datos en varios servidores (mirrors).

Servidores proxy y caches

Cache: es un almacen de objetos de datos uytilizados recientemente. Cuando un cliente necesita un objeto, la cache le proporciona una copia en vez de realizar la busqueda completa. Si su copia no esta actualizada, entonces ira a buscarla y guardara una copia local.

Servidores proxy: proporcionan cache compartida de recursos web a las maquinas cliente. Su proposito es incrementar la disponibilidad de datos y prestaciones. Tambien se pueden utilizar para acceder a otros servidores evadiendo un cortafuegos (firewall).

Peer to peer

En esta arquitectura Todos los proceosos desempenan tareas semejantes cooperativamente, por ende no existte distincion entre clientes y servidores. La eliminacino del proceso servidor reduce los retardos de comunicacion al acceder a objetos locales.

Una aplicacion peer to peer puede implementarse con capas de middleware para realizar notificacion de eventos y comunicacion a grupos para indicar a todos los procesos que la aplicacion sufrio cambios. Esto proporciona mejor interactividad que con una arquitectura cliente-servidor.

Variaciones del modelo cliente-servidor

Dependiendo de las consideraciones como:

- El uso de codigo movil (codigo que se descarga del servidor y ejecuta en el cliente, como javascript).
- Las necesidades de los usuarios que cuentas con recursos de hardware bajos.
- El requisito de anadir o eliminar dispositivos moviles.

Se desprenden variaciones de este modelo.

- **Codigo movil**: el ejemplo mas conocido son los applets. Codigo que luego se descarga y ejecuta en el navegador del cliente, pero este codigo esta almacenado en un servidor web. Proporciona una respuesta interactiva imposible de lograr con el modelo tradicional.
- **Agentes moviles**: son programas de software que pueden moverse de un nodo a otro dentro de una red distribuida , llevando consigo información, código y recursos, y ejecutando tareas en diferentes nodos. Estos agentes pueden mejorar la eficiencia y el rendimiento del sistema distribuido al aprovechar la capacidad de procesamiento y los recursos disponibles en diferentes nodos. Tambien son una amenaza potencial a la seguridad (ej: un gusano); ademas, sus tareas pueden ser realizadas por otros medios (como crawlers web). Por estot si aplicabilidad es limitada.

- **Clientes ligeros**: En lugar de ejecutar aplicaciones y almacenar datos en el propio dispositivo, los clientes ligeros se conectan a un servidor central a través de una red y acceden a las aplicaciones y datos alojados en ese servidor. El servidor realiza el procesamiento pesado y envía los resultados al cliente ligero para su visualización y uso. El principal inconveniente se encuentra en aplicaciones graficas fuertemente interactivas (como CAD o videojuegos).
- **Enlace espontaneo**: Se refiere a la capacidad de los dispositivos o clientes en un sistema distribuido para establecer conexiones de red de forma automática y sin intervención manual. Permite que los dispositivos se descubran y se conecten entre sí sin la necesidad de configuraciones manuales o intervención humana. Esto facilita la comunicación y la colaboración entre los dispositivos en el sistema distribuido.

Para usuarios moviles se presentan otras cuestiones:

- Conectividad limitada.
- Seguridad y privacidad.

Requisitos para el diseno de arquitecturas distribuidas

- *Prestaciones*: capacidad de respuesta (Responsiveness), productividad (Throughput), balance de cargas.
- *Calidad del servicio*: fiabilidad, tolerancia a fallos, seguridad y prestaciones.

Modelos fundamentales

Son descripciones abstractas que se utilizan para analizar y comprender las propiedades y características comunes a todos los sistemas distribuidos. Estos modelos proporcionan una base teórica para el diseño, desarrollo y análisis de sistemas distribuidos.

Modelo de interaccion

Este modelo se centra en las interacciones entre los procesos en un sistema distribuido . Examina cómo se comunican y coordinan los procesos, y cómo se establecen límites temporales en el sistema distribuido

Factores que afectan a los procesos de un SD

Prestaciones del canal de comunicacion

- *Latencia*: retardo enttre el envio de un mensaje y su recepcion. Incluye el retarde de llegada al destino del mensaje, retardo de accerder a la red, tiempo empleado por el sistema operativo del emisor(depende de su carga).
- *Ancho de banda*: cantidad total de informacion que puede transmitirse en un tiempo dado.
- *Fluctuacion* o **Jitter**: variacionde tiempo invertitdo en completar la entrega del mensaje. Se requiere un flujo constante, en especial en multimedia.

Relojes y eventos de temporizacion

Cada computador de un SD tiene su propio reloj interno. Los relojes presentan derivas con respecto al tiempo perfecto y sus tasas de deriva difieren de una a otra.

Tasa de deriva: o tambien clock drift rate, es la proporcion por la cual un reloj difiere del reloj de referencia prefecto. A partir de un tiempo los relojes de un SD comenzaran a variar significativamente a menos que se apliquen correcciones.

Variantes del modelo de interaccion

Tenemos dos modelos simples que parten de posiciones extremas y opuestas: el primero tiene en cuenta una fuerte restricción sobre el tiempo; en el segundo no se hace ninguna presuposición.

1. *SD Sincronos*: El tiempo de ejecución de cada etapa tiene ciertos límites inferior y superior conocidos. Cada mensaje se recibe en un tiempo limitado conocido. Cada proceso cuenta con un reloj local cuya deriva es conocida.
2. *SD Asincronos*: un sistema distribuido asíncrono es aquel en que no existen limitaciones la velocidad de procesamiento, los retardos de transmision o las tasas de deriva de los relojes. Estte es el modelo de internet.

Ordenamiento de eventos

Como los relojes de un sistema distribuido no pueden sincronizarse de manera perfecta, Lamport (1987) propuso un modelo de tiempo lógico que pudiera utilizarse para ordenar los eventos de procesos que se ejecutan en computadores diferentes de un sistema distribuido.
Cada nodo o componente del sistema mantiene un contador local que se incrementa cada vez que ocurre un evento en ese nodo. Cuando un nodo envía un

mensaje a otro nodo, incluye su valor de tiempo en el mensaje. Al recibir el mensaje, el nodo receptor actualiza su contador local al máximo entre su valor actual y el valor de tiempo recibido. De esta manera, se establece un orden parcial entre los eventos en diferentes nodos.

Modelo de fallo

Este modelo describe los posibles fallos que pueden ocurrir en los procesos y canales de comunicación en un sistema distribuido . Ayuda a especificar y comprender los diferentes tipos de fallos y cómo afectan al sistema. Define comunicación fiable y procesos correctos.

Tipos de fallos

Fallos por omision

Casos en los que los procesos o canales no consiguen realizar sus funciones.

Tipos:

- *Omision de procesos*: fracaso o ruptura del procesamiento. El proceso esta parado y no puede continuear ejecutandose. La rotura de un proceso se denomina como fallo-parada(*fail-stop*). Para detectar estos fallos se requiere del uso de timeouts.
- *Omision de comunicaciones*: cuando el canal no consigue mandar el mensaje del buffer de salida al buffer de mensajes entrantes. La causa suele ser la fata de espacio en el buffer de recepcion de alguna pasarela intermedia, o por un error de red, detectable mediante el checksum de los mensajes.
- *Omision de envio*: perdida de mensajes entre el proceso emisor y bufer de mensajes de salida.
- *Omision de recepcion*: perdidad de mensajes entre el bufer de mensajes entrantes y el proceso receptor.
- *Omision del canal*: la perdida de mensajes entre los bufers de salida y entrada de mensajes.

Fallos arbitrarios

Se emplea para describir la peor semántica de fallo posible, en la que puede ocurrir cualquier tipo de error.

Un fallo arbitrario es aquel en el que se omiten pasos para el procesamiento o se realizan pasos no intencionados de procesamiento. En consecuencia los fallos arbitrarios en los procesos no pueden detectarse observando si el proceso responde a las invocaciones.

Ejemplos

- Contenido corrupto del mensaje.
- Mensaje inexistente.
- Mensajes duplicados.

Fallos de temporizacion y fiabilidad

Fallos de temporizacion: se aplican en sistemas sincronos. Sucede cuando el reloj local de un proceso excede el limite de su tasa de deriva sobre el tiempo real; cuando se excede el tiempo de computo de algun paso; la transmision de un mensaje toma demasiado tiempo.

Comunicacion fiable: se define en terminos de validez (el mensaje llegara eventualmente al destino) e integridad (el mensaje recibido sera identico y no se entregaran mensajes duplicados).

Modelo de seguridad

Este modelo se ocupa de las posibles amenazas y vulnerabilidades en un sistema distribuido . Examina cómo se pueden proteger los procesos y canales de comunicación contra ataques y garantizar la confidencialidad, integridad y disponibilidad de los datos.

Proteccion de objetos

La seguridad de un sistema distribuido puede lograrse asegurando los procesos, los canales empleados y los objetos que encapsulan contra el acceso no autorizado.

Los mensajes están expuestos a un ataque dado que la red y el servicio de comunicación es abierto, de modo que cualquier par de procesos puedan interactuar.

El enemigo...

Es un adversario capaz de enviar o leer o copiar cualquier mensaje entre un par de procesos. Tales ataques pueden cometerse simplemente utilizando un computador conectado a una red.

Amenazas

- A procesos: Cualquier proceso diseñado para admitir peticiones puede recibir un mensaje de cualquier otro proceso, y podria no ser capaz de determinar la identidad del emisor.
- A Servidores: un servidor puede recibir muchas invocaciones, no necesariamente puede determinar la identidad que se halla tras una invocación particular.
- A Clientes: cuando un cliente recibe un resultado de un servidor, no necesariamente puede decir si la fuente es del servidor bueno o de un enemigo (*spoofing*).
- A los canales de comunicación: Un enemigo puede copiar, alterar o insertar mensajes según viajan a través de la red y sus pasarelas.

Canales seguros

Es un medio de transporte de datos que garantiza la confidencialidad, integridad y autenticidad de la información transmitida. Para crearlos se emplean la encriptcion y autenticacion.

Propiedades

- Cada proceso conoce bien la identidad del principal en cuya representación se ejecuta otro proceso.
- Asegura la privacidad y la integridad de los datos.
- Cada mensaje incluye un sello de carácter temporal para prevenir el reenvío o la reordenación de los mensajes.

Otras posibles amenazas

Ataque DoS: el enemigo interfiere con las actividades de los usuarios autorizados mediante un número excesivo de invocaciones sin sentido sobre servicios o la red, la que resulta en una sobrecarga de los recursos físicos (ancho de banda, capacidad de procesamiento del servidor).

Código Móvil: un código descargado del servidor y ejecutado en el cliente podría actuar como TroyanHorse.