Sistemas distribuidos

Grado en Ingeniería Informática

Tema 2.1: Modelos de Sistema

Departamento de Ingeniería Informática
Universidad de Cádiz

Escuela Superior de Ingeniería
Dpto. de Ingeniería Informática

Versión 2.0

Indice

Sección 1 | Introducción

Conceptos previos

- Un sistema software es distribuido cuando se ejecuta repartido entre varios ordenadores conectados en red.
- Razones por las que se desea distribuir un sistema software:
  - Pocas aplicaciones trabajan aisladas; la mayoría necesita intercambiar datos e interactuar con aplicaciones externas.
  - El abaratamiento e incremento de potencia de los ordenadores personales.
  - Internet permite que ciertas partes de las aplicaciones se distribuyan en las máquinas de los usuarios finales, mientras que la funcionalidad que implementa la lógica del negocio se ejecute en los ordenadores del proveedor de los servicios.

Definiciones

Modelo

Descripción abstracta simplificada pero consistente de cada aspecto relevante del diseño de un SD.

Tipos de modelos

- **Arquitectónicos**: Relación entre los componentes del sistema. Ej.: modelo cliente-servidor.
- **Fundamentales**: Descripción más formal de las propiedades que son comunes en todos los modelos arquitectónicos:
  - **Modelo de interacción**: prestaciones y dificultad de poner límites temporales en un SD. Ej: entrega de mensajes.
  - **Modelo de fallos**: especificación precisa de los fallos que se pueden producir en los procesos y canales de comunicación.
  - **Modelo de seguridad**: posibles amenazas para los procesos y canales de comunicación.

Dificultades y amenazas para SD (I)

Modos de utilización muy variables

Las partes componentes de los sistemas pueden:

- Estar sujetas a grandes variaciones en la carga de trabajo.
- Estar desconectadas o deficientemente conectadas.
- Tener requisitos especiales para comunicaciones.

Amplio rango de entornos

- Hardware.
- Sistemas operativos.
- Redes heterogéneas.

Dificultades y amenazas para SD (II)

Problemas internos

- Relojes no sincronizados.
- Actualizaciones conflictivas de datos.
- Fallos en hardware y software.

Amenazas externas

- Ataques a la integridad y secreto de datos.
- Denegación de servicio.

Sección 2 | Modelos Arquitectónicos

Introducción

- Un modelo arquitectónico simplifica y abstrae, inicialmente, las funciones de los componentes individuales y posteriormente considera:
  - Ubicación de los componentes en la red de computadores.
  - Interrelaciones entre los componentes.
- Los procesos pueden clasificarse en:
  - Servidores.
  - Clientes.
  - Iguales.
- Se pueden construir otros sistemas dinámicos (variantes de cliente-servidor):
  - Mover código de un proceso a otro: un proceso delega tareas en otro.
  - Añadir o eliminar dispositivos móviles sin incidencias.

Capas de software (I)

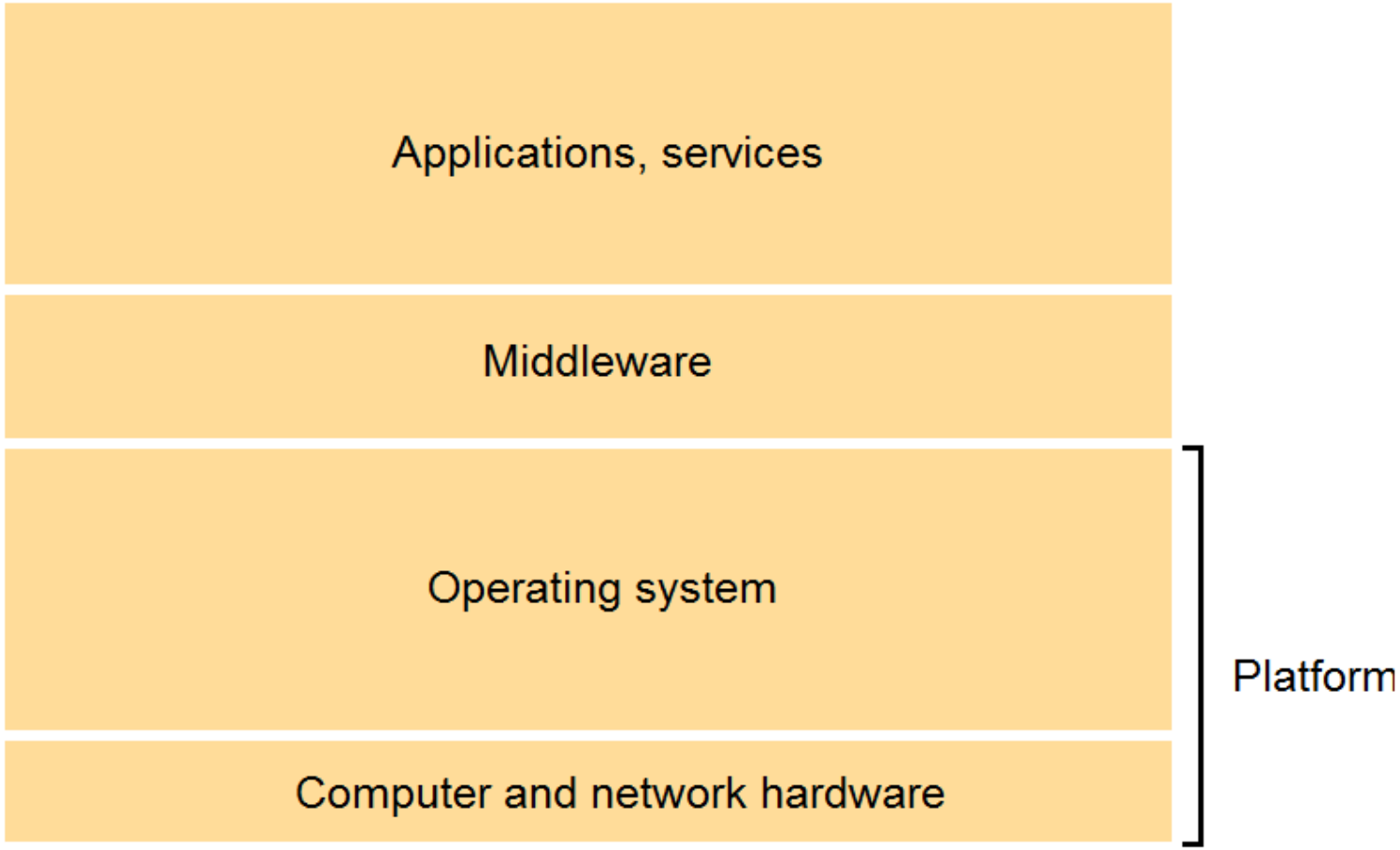
Plataforma

- Nivel de hardware y capas más bajas de software.
- Estas capas proporcionan servicios a las que están por encima de ellas (implementadas independientemente en cada computador).

Middleware

- Enmascara la heterogeneidad.
- Proporciona un modelo de programación conveniente.
- Se representa mediante procesos u objetos en un conjunto de computadores que interactúan entre sí.
- Proporciona bloques útiles para la construcción de componentes software que puedan trabajar con otros.
- Mejora el nivel de las actividades de comunicación.
- Proporcionan servicios para su uso en los programas de aplicación.

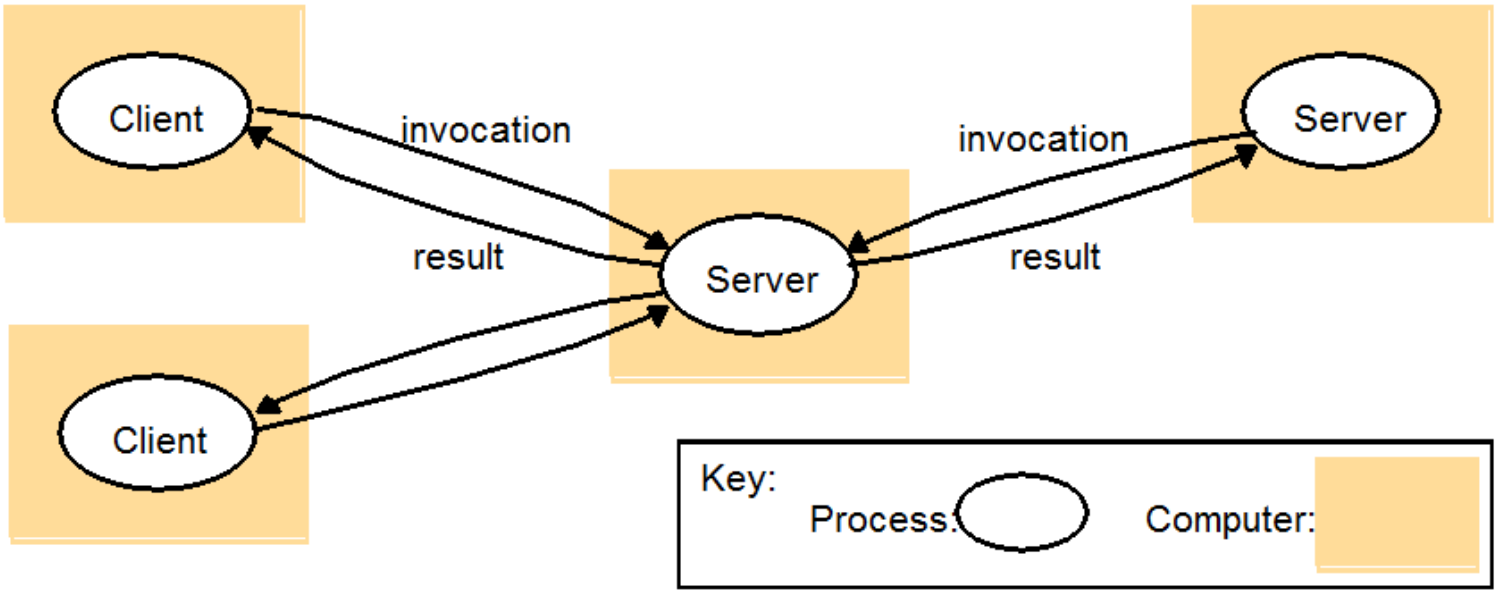
Capas de software (II)



Arquitecturas de sistema (I)

Modelo cliente-servidor

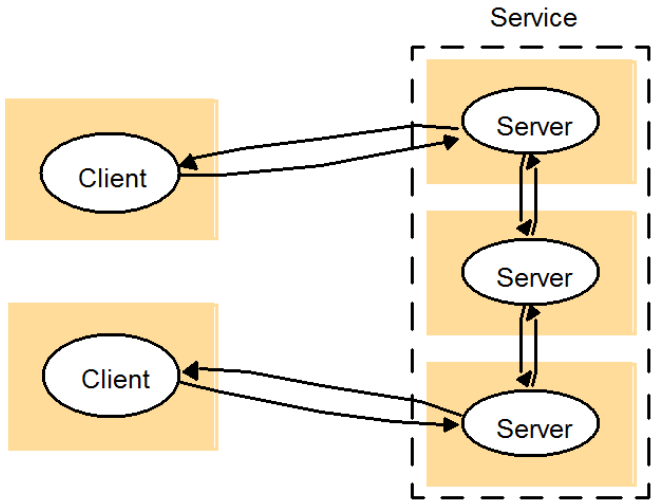
- Una de las más utilizadas.
- Estructura sencilla donde interaccionan los procesos clientes con los procesos servidores individuales, para acceder a los recursos compartidos.
- Los servidores pueden ser también clientes de otros servidores.



Arquitecturas de sistema (II)

Servicios proporcionados por múltiples servidores

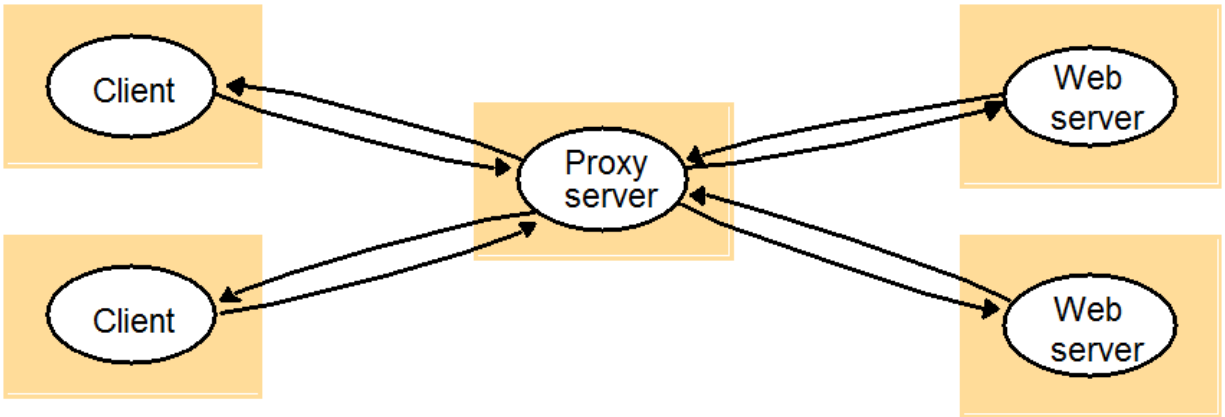
- Los servicios pueden implementarse como distintos procesos de servidor en computadores separados interaccionando para proporcionar un servicio a los procesos clientes.
- Los servidores pueden:
  - Dividir el conjunto de objetos en los que está basado el servicio y distribuírselos entre ellos mismos.
  - Mantener copias replicadas de ellos en varias máquinas.



Arquitecturas de sistema (III)

Servidores proxy y cachés

- Una caché es un almacén de objetos utilizados recientemente.
- Se encuentra más próximo que los objetos en sí.
- Al recibir un objeto nuevo en un computador se añade a la caché.
- Cuando se necesita un objeto, el servicio comprueba la caché y proporciona al proceso cliente el objeto de una copia actualizada.
- La caché puede estar en cada cliente o en un servidor proxy (compartido desde varios clientes).



Arquitecturas de sistema (IV)

Proceso "de igual a igual" (*peer to peer*)

- Todos los procesos desempeñan tareas semejantes.
- Interactúan cooperativamente como iguales para realizar una actividad distribuida o cómputo sin distinción entre clientes y servidores.
- En general, n procesos parejos podrán interactuar entre ellos.
- La eliminación del proceso servidor reduce los retardos de comunicación entre procesos (al acceder a objetos locales).

Tareas

1. Describa e ilustre la arquitectura cliente-servidor de una de las principales aplicaciones de Internet (Web, email o netnews).

Variaciones en el modelo de cliente-servidor (I)

Factores

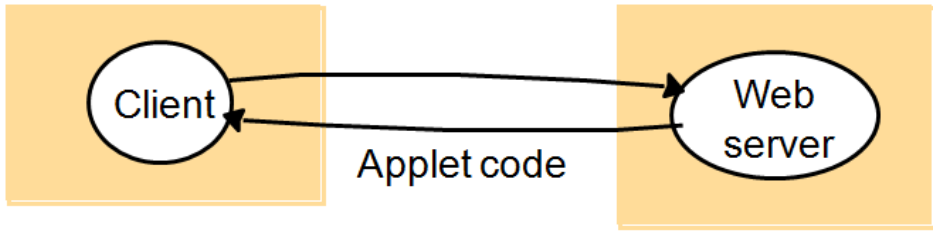
- Uso de código móvil y agentes móviles.
- Necesidad de computadores de bajo coste y con recursos hardware limitados.
- Requisito de añadir o eliminar dispositivos móviles.

Variaciones en el modelo de cliente-servidor (II)

Código móvil

- Los *applets* son el ejemplo más conocido de código móvil.
- El usuario ejecuta un navegador y selecciona un enlace con un applet (código almacenado en un servidor web). Este código se descargará en el navegador donde se ejecutará.
- Ventaja: proporciona una buena respuesta interactiva.

a) La petición del cliente origina la descarga del código del applet



b) El cliente interactúa con el applet

Variaciones en el modelo de cliente-servidor (III)

Agente móvil

- Es un programa en ejecución (código y datos) que se traslada de un computador a otro en la red realizando una tarea para alguien.
- Puede hacer muchas solicitudes a los recursos locales de los sitios que visita.
- Útil para instalar y mantener software en los ordenadores de una empresa.

Variaciones en el modelo de cliente-servidor (IV)

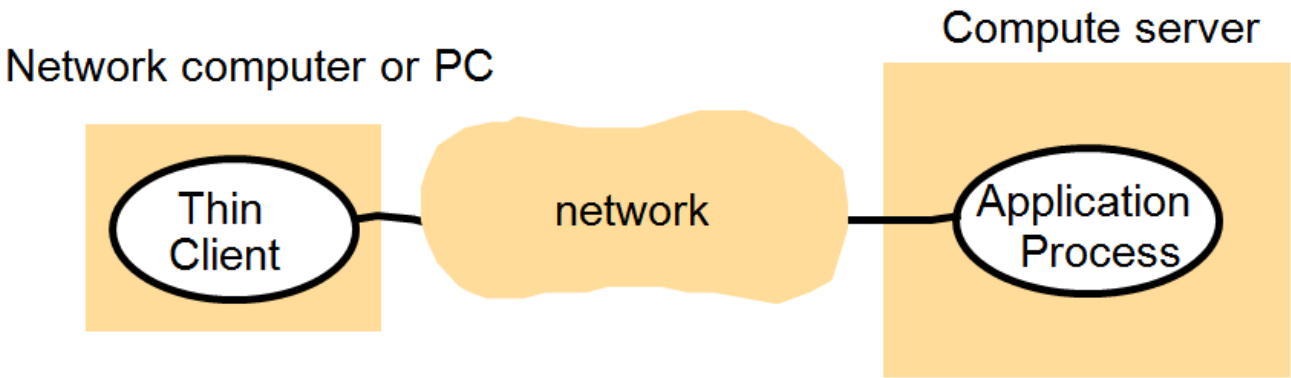
Computadora de red

- Descarga su sistema operativo y cualquier aplicación software que necesite el usuario desde un servidor de archivos remoto.
- Las aplicaciones se lanzan localmente.
- Los archivos se gestionan desde un servidor de archivos remoto.
- Se pueden ejecutar aplicaciones en red (navegadores web).
- Los usuarios pueden migrar de un computador de red a otro.
- Se podría utilizar un disco como caché de archivos de programa y datos.

Variaciones en el modelo de cliente-servidor (V)

Cliente ligero

- Capa de aplicación que soporta una interfaz de usuario basada en ventanas sobre un computador local de usuario mientras se ejecutan programas de aplicación en un computador remoto.
- Las aplicaciones se ejecutan en un servidor de cálculo (multiprocesador o *cluster*) con capacidad para ejecutar un gran número de aplicaciones simultáneamente.
- Inconveniente: latencias de red y sistema operativo (actividades gráficas interactivas).



Variaciones en el modelo de cliente-servidor (VI)

Dispositivo móvil y enlace espontáneo a red

- Los usuarios desplazan sus dispositivos móviles entre los entornos de red y se benefician de los servicios locales y remotos.
- Las características de la conexión a red espontánea son:
  - Conexión fácil a la red local.
  - Intregración fácil con servicios locales.
- Problemas para los usuarios móviles:
  - Conectividad limitada.
  - Seguridad y privacidad.

Tareas

2. Dé dos ejemplos de aplicaciones donde sea beneficioso emplear código móvil.

Requisitos de diseño para arquitecturas distribuidas (I)

Temas de prestaciones

- Capacidad de respuesta.
- Productividad: rapidez a la que se realiza el trabajo computacional.
- Balance de cargas computacionales.

Calidad de servicio

- Fiabilidad.
- Seguridad.
- Prestaciones.
- Facilidad de adaptación.

Requisitos de diseño para arquitecturas distribuidas (II)

Uso de caché y de replicación

Aspectos de fiabilidad

- Corrección.
- Tolerancia a fallos (introduciendo redundancia).
- Seguridad.

Sección 3 | Modelo de Interacción

Modelo de Interacción

- Hay limitaciones debidas a la comunicación
- Es imposible predecir el retraso con el que llega un mensaje
- No hay un tiempo global a todo el sistema
- La ejecución es 'no determinista' y difícil de depurar

Algoritmo Distribuido: Definición de los pasos que hay que llevar a cabo por cada uno de los procesos del sistema, incluyendo los mensajes de transmisión entre ellos

Prestaciones del canal de comunicación

- Latencia: Retardo entre el envío de un mensaje y su recepción
- Ancho de banda: Información que puede transmitirse en un intervalo de tiempo
- Fluctuación (jitter): Variación del tiempo invertido en repartir una serie de mensajes

Relojes y eventos de tiempo

- Cada computador tiene su propio reloj interno (reloj local): Puede usarse en procesos locales para marcas de tiempo
- Tasa de deriva de reloj (clock drift rate): Evolución de la diferencia entre un reloj local y un reloj de referencia "perfecto"
- Receptores GPS
- Network Time Protocol (NTP)
- Mecanismos de ordenación de eventos
- Dos tipos de modelo de interacción: Síncrono y asíncrono

Modelos Síncronos

Conocimiento de características temporales:

- El tiempo de ejecución de cada etapa de un proceso tiene ciertos límites inferior y superior conocidos
- Cada mensaje transmitido sobre un canal se recibe en un tiempo límite conocido
- Cada proceso tiene un reloj local cuya tasa de deriva sobre el tiempo de referencia tiene un límite conocido
- A nivel teórico, podemos establecer unos límites para tener una idea aproximada de cómo se comportará el sistema
- A nivel práctico, es imposible garantizar esos límites siempre: Aunque a veces se pueden utilizar, por ejemplo como timeout

Modelos Asíncronos

- No hay limitaciones en cuanto a
  - Velocidad de procesamiento
  - Retardos en la transmisión de mensajes
  - Tasas de deriva de los relojes
- Los sistemas distribuidos reales suelen ser asíncronos (p.ej Internet)
- Una solución válida para un sistema asíncrono lo es también para uno síncrono

Sección 4 | Modelos de Fallo

Modelos de Fallo

- Estudio de las causas posibles de fallo: Para poder comprender sus consecuencias
- Tipo de fallo según la entidad
  - Fallos de proceso
  - Fallos de comunicación
- Tipo de fallo según el problema:
  - Fallos por omisión : No se consigue realizar una acción que se debería poder hacer, o en el tiempo en el que se debería poder hacer (s. síncrono)
  - Fallos arbitrarios (bizantinos): Errores de cualquier tipo, fuera del esquema de mensajes

Tareas

3. Dé dos ejemplos de modelos de fallos que encuentre en internet.

Bibliografía

Coulouris, G.; Dollimore, J.; Kindberg, T.
Distributed Systems: Concepts and Design (5ª ed.)
Addison-Wesley, 2012.
(Trad. al castellano: Sistemas distribuidos: conceptos y diseño, 3ª ed., Pearson 2001)