Tema 3: Arquitecturas de los Sistemas Distribuidos

Sistemas Distribuidos

2022-2023

T3 Arquitecturas

Concepto

La arquitectura de un SD se refiere a:

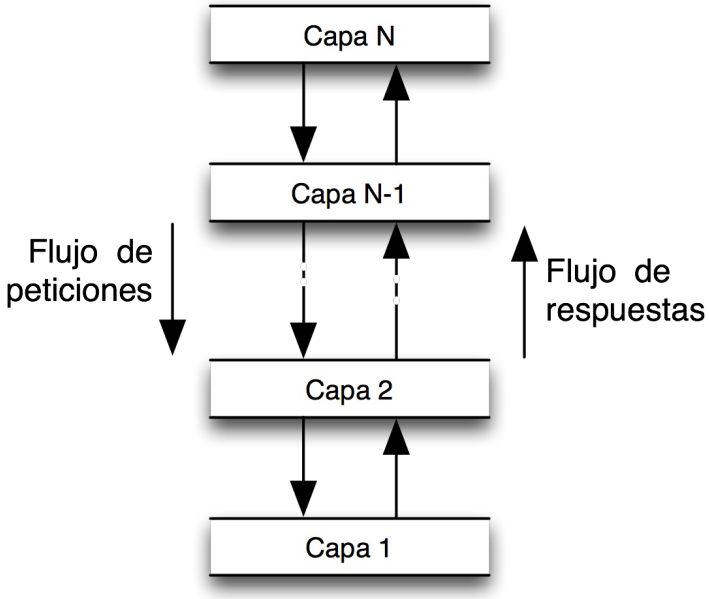
- La ubicación de los componentes del SD sobre la red subyacente
- La relación e interacción entre ellos

Estilos de arquitectura:

- Arquitectura en capas
- Arquitecturas basadas en objetos
- Arquitecturas basadas en eventos
- Arquitecturas centradas en datos

Arquitectura en capas

Cada capa puede invocar sólo a la que está debajo.



El estándar ISO/OSI es un ejemplo de arquitectura en capas.

Arquitectura en capas

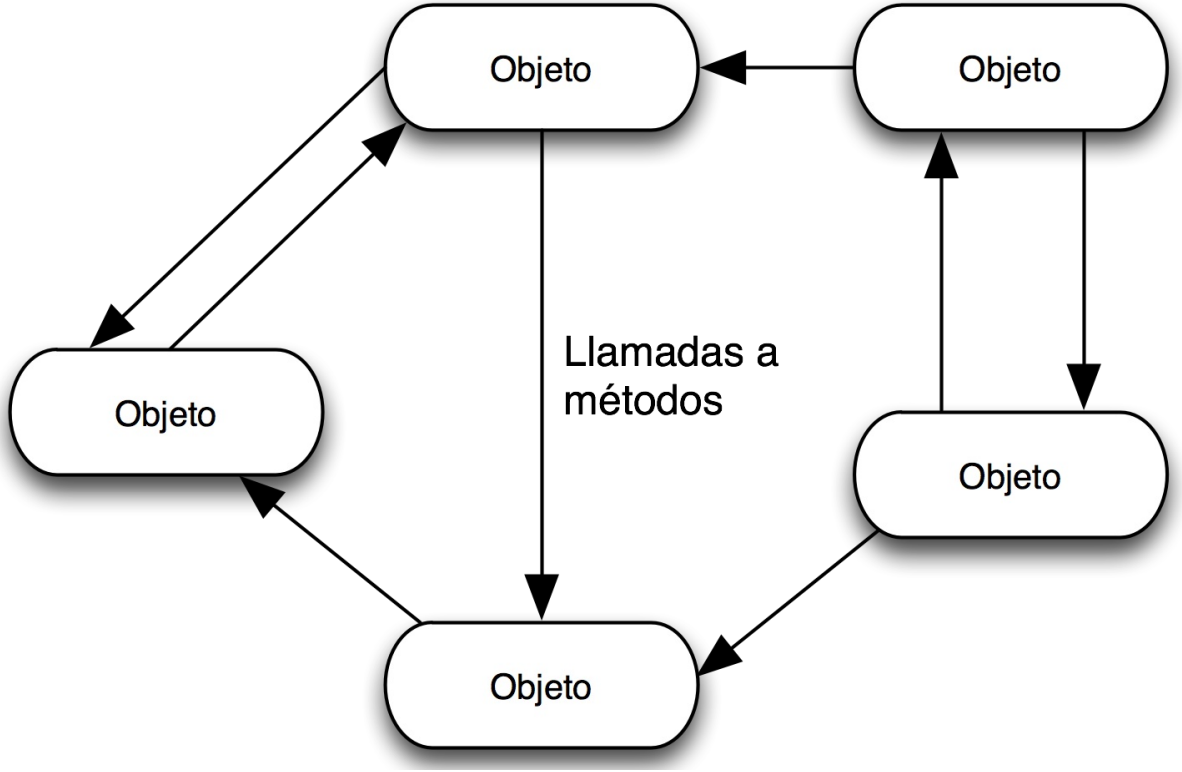
El *software* típicamente se estructura en capas:

- Bajo nivel: cercano a la arquitectura máquina
- Alto nivel: cercano al usuario

Esto mismo es posible hacerlo en un SD, pero aparecen más niveles:

- Nivel de plataforma
  - Hardware del computador
  - Más el operativo que corre en él
- Nivel de *middleware*
- Nivel de aplicación y servicios

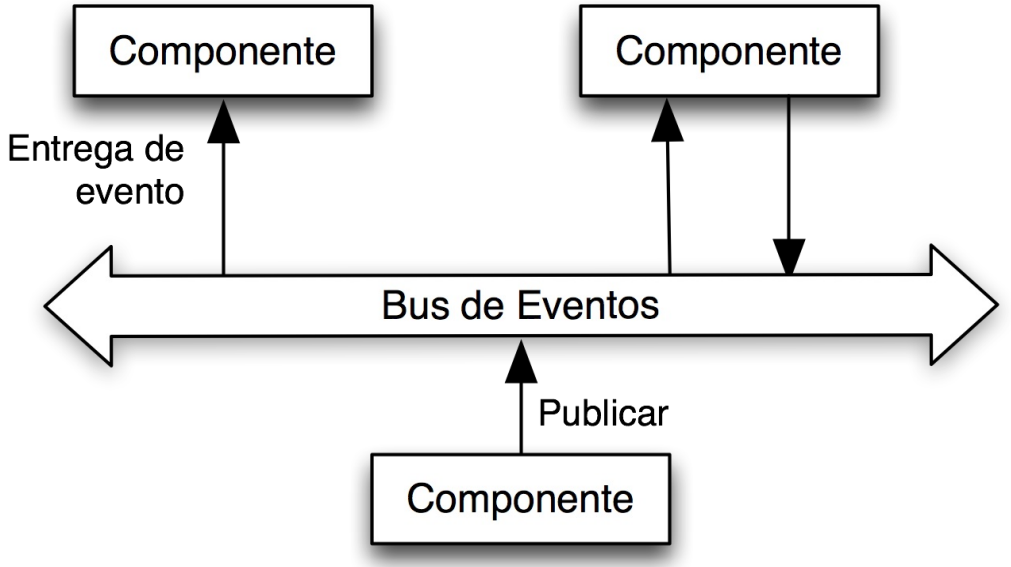
Arquitecturas basadas en objetos



Se trata de componentes que interactúan mediante RPC.

Sigue el paradigma cliente-servidor.

Arquitecturas basadas en eventos



- Los componentes se comunican mediante *eventos* que pueden llevar asociados datos.
- Están *débilmente acoplados*
- Ejemplo típico: sistemas *publisher/subscriber* (editor/subscriptor)
- Normalmente apoyadas en MOM

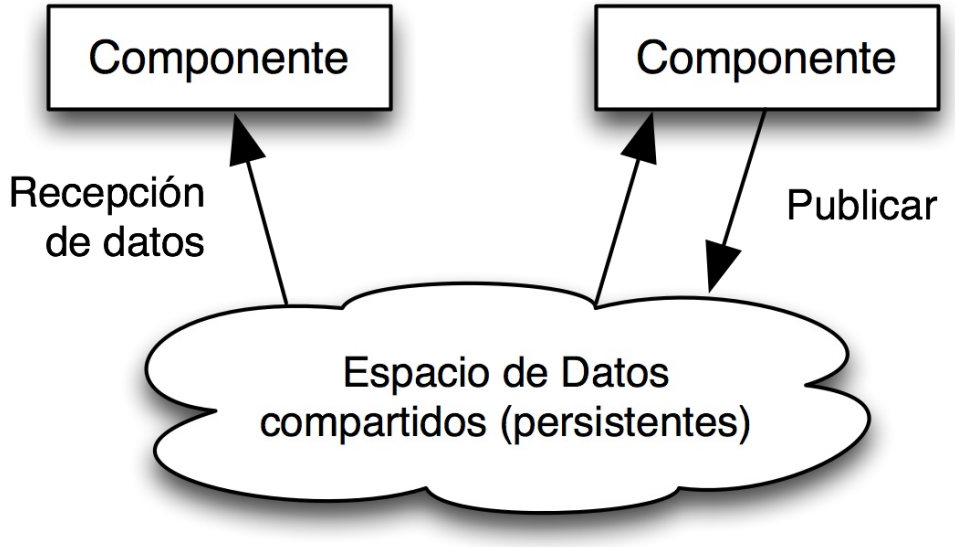
Arquitecturas centradas en datos

- Los componentes se comunican a través de un repositorio compartido.
- Por ejemplo:
  - Comunicación a través de ficheros NFS
  - Web
  - Web services

Si se combina con eventos, dan lugar a *espacios de datos compartidos*.

- Aún más desacoplados
- Se evita el *polling* en los componentes que necesitan los datos

Arquitecturas con espacios de datos compartidos



Muchos ejemplos de ficheros "en la nube" (iCloud, Google Drive, Dropbox, Box, SugarSync, etc.)

Bases de datos con capacidad de notificación (redis, CouchDB, MongoDB, etc.)

Arquitecturas de sistema

Atendiendo a dónde se ubican sus componentes:

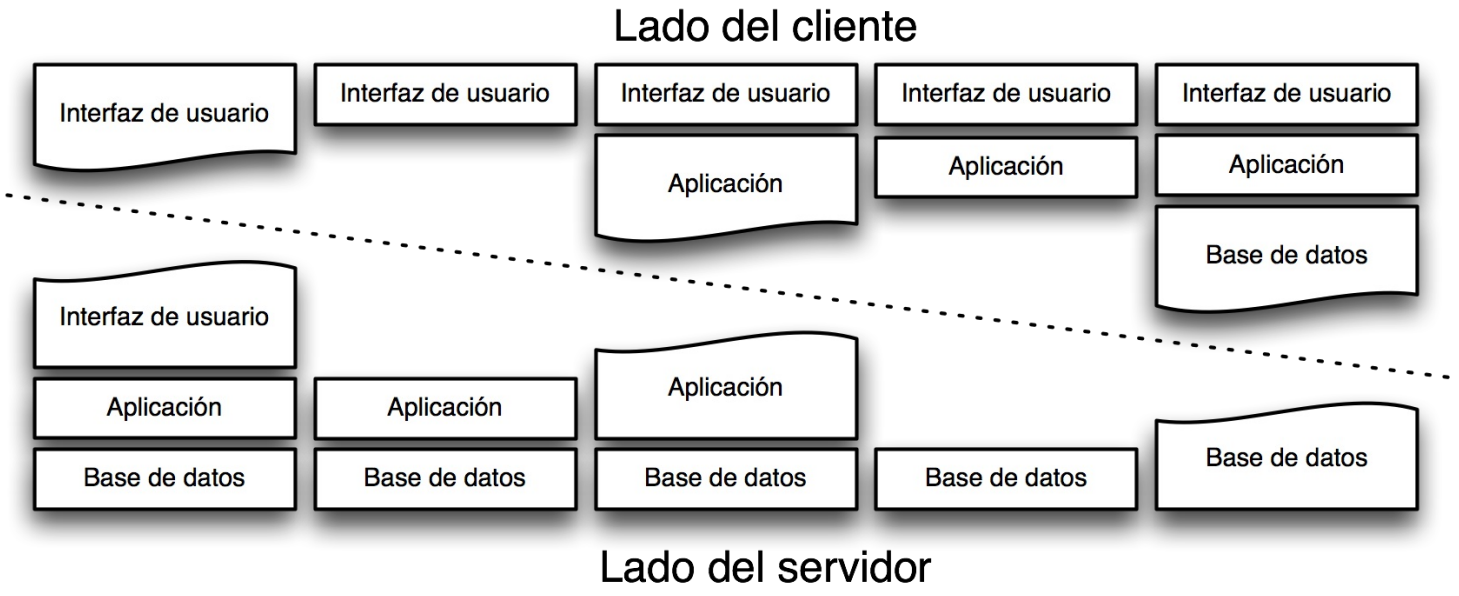
- Arquitecturas centralizadas
  - Cliente-servidor
  - 3 niveles (navegador, servidor, base de datos)
  - Multinivel
- Arquitecturas descentralizadas
  - Arquitecturas p2p estructuradas
  - Arquitecturas p2p desestructuradas

Cliente-servidor

El modelo más antiguo y aún en uso, pero:

- La división de qué funcionalidad va en los clientes y cuál en los servidores no está clara. Hay varias capas, como mínimo:
  - Interfaz de usuario
  - Lógica de la aplicación
  - Almacenamiento persistente (base de datos)
- Las capas se implementan en un número de *niveles*
- Cliente-servidor es el caso con 2 niveles. Varias posibilidades
- Arquitectura multinivel: cada nivel un proceso.

Distribución de niveles entre cliente y servidor



Arquitecturas descentralizadas

- En lugar de separar capas en procesos (*verticalmente*), se hace lo contrario.
- Una capa se implementa distribuida entre varios procesos (*horizontalmente*)
- Todos los procesos implementan todas las capas, pero trabajando sobre diferentes datos.
- Son todos iguales entre sí, e intercambiables.

Una arquitectura descentralizada crea una red virtual superpuesta "encima" de la red física (*overlay network*)

- Los nodos son los procesos
- Los enlaces son sus intercomunicaciones

Arquitecturas descentralizadas

Estructuradas

- La *overlay network* se construye con algoritmos deterministas.
- Los datos se mapean a nodos de forma determinista.
- Por ejemplo: *tabla hash distribuída* DHT (Chord, Kademlia, etc.)

Desestructuradas

- La red superpuesta se construye con algoritmos que usan aleatoriedad.
- Los datos se distribuyen aleatoriamente entre sus nodos.
- Las búsquedas son problemáticas (*flood*)

Arquitecturas descentralizadas

Híbridas

- Mayormente aleatorias
- Nodos "especiales" (*superpeers*):
  - Contienen índices para mejorar las búsquedas
  - Actúan como *brokers* (intermediarios)
- Ejamplo: *Bittorrent*