Paradigmas/Modelos de SD (3)

Sistemas Distribuidos
I.T.I. Sistemas (2005-06)
© César Llamas Bello
Universidad de Valladolid

Abstracciones, Paradigmas

- Abstraciones: ocultan detalles
- Paradigmas (Modelos): Patron, ejemplo.

Level of abstraction

high

Object space, collaborative applications
Network services, object request broker, mobile agent
Remote procedure call, remote method invocation
Message services Client-server, peer-to-peer
Message passing

low

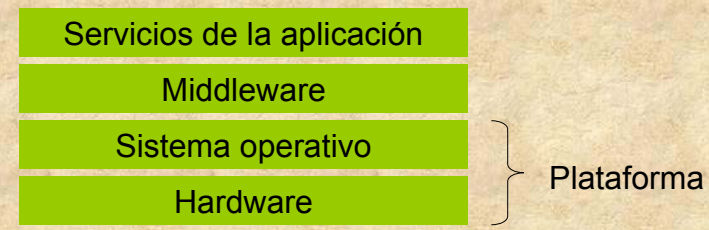
Introducción

- El modelo arquitectónico describe:
  - Interacciones entre componentes
  - Enlace con la plataforma de red: "simplifica y abstrae las funciones y roles de los componentes individuales"
- El enfoque CDK es un poco confuso
  - Seguiremos el enfoque de Liu, aunque veremos antes algún concepto interesante.

Íntroducción

- Modelos de requisitos no funcionales
  - Modelo de interacción
  - Modelo de fallo
  - Modelo de seguridad

Capas de soft



- Componentes importantes:
  - Plataforma
  - Middleware

Capas de soft

- Plataforma
  - Contiene los servicios propios de cada computadora concreta
  - Depende del hardware y del S.O.

Middleware

- Permite enmascarar la heterogeneidad
- Puede dar un modelo y una interfaz de programación utilizable
- Puede soportar abstracciones como:
  - Llamadas a procedimientos remotos (RPC)
  - Comunicación en grupo (JXTA)
  - Eventos, replicación, servicios multimedia, etc... (IBM MS, Lotus N. ...)

Middleware

- ¿Qué forma tiene el middleware? :
  - Bibliotecas adicionales:
    - Procedimientos remotos (RPC)
    - Objetos remotos (RMI, CORBA, JXTA)
  - Herramientas de programación
    - Lenguajes de definición de interfaces (IDL) + Compiladores para ellos
  - Servicios básicos de ayuda
    - servicios de nombres, para buscar objetos
    - de notificación de eventos, (WebSphere M. Broker)
    - De control de transacciones, etc.

.

Middleware

- ¿Qué limitaciones impone?
  - Se incrementa la complejidad arquitectónica:
    - Hay más niveles
  - Hay que aprender más herramientas
  - Se pierde el control de bajo nivel sobre los modos de fallo
  - Se depende de terceras partes,
  - ...

Ejemplo de aplicación

- Sistema de subastas públicas
  - Se inicia una sesión de subastas anunciando un objeto a subastar.
  - Se admiten pujas
  - Al final de la subasta, el subastador anuncia el resultado.

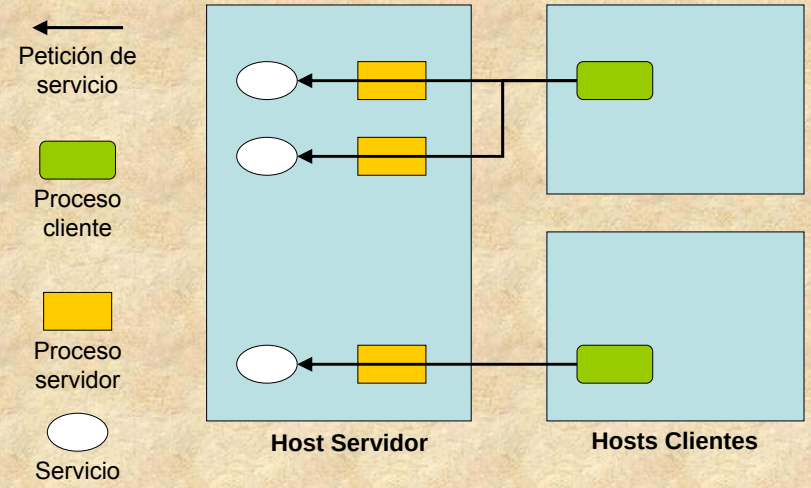
Paso de mensajes

- Partes:
  emisor y receptor, que intercambian roles.
- Cada mensaje pertenece a un catalogo de primitivas de un protocolo que comparten emisor y receptor.
  - Las operaciones básicas son envía() y recibe()
  - En caso de conexión hay también conecta() y desconecta()
  - /* no confundir con primitivas del protocolo */
- El API de sockets da soporte a este paradigma.

Cliente-Servidor

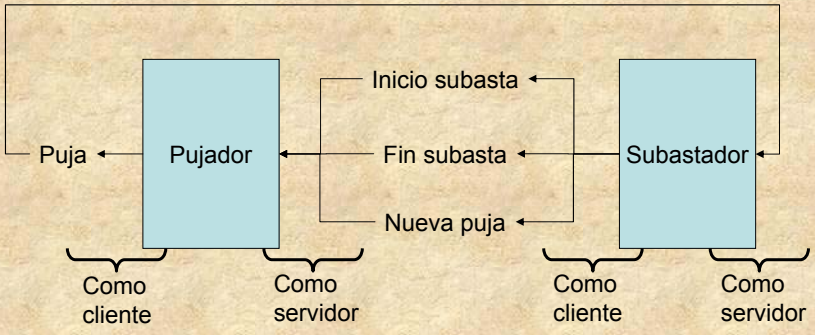
- Cada interlocutor asume un rol:
  - El proceso servidor proporciona un servicio.
  - El proceso cliente requiere el servicio: emite peticiones y espera respuestas del servicio.
  - Estos roles vienen asignados en el protocolo.
- La sincronización es sencilla, al estilo de la invocación de un procedimiento
  - De echo C-S es la base de RPC.
- Es la base de muchos servicios de red: HTTP, FTP, DNS, finger, gopher, smtp, …

Cliente-Servidor



Cliente-Servidor (ej.)

- Cada participante asume el papel de cliente y de servidor (aunque **sobre interfaces diversas**).



Peer-to-Peer

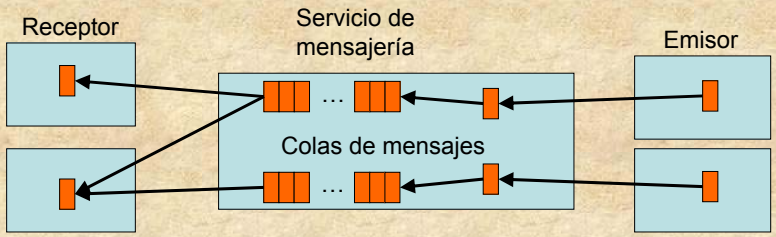
- Los procesos participantes juegan papeles iguales, con habilidades y responsabilidades equivalentes.
  - Cada uno puede emitir y recibir peticiones, *sobre la misma interfaz de interacción*.
  - C-S es más apropiado para servicios centralizados
  - P2P es más apropiado para entornos colaborativos y conexiones punto a punto.

Peer-to-Peer

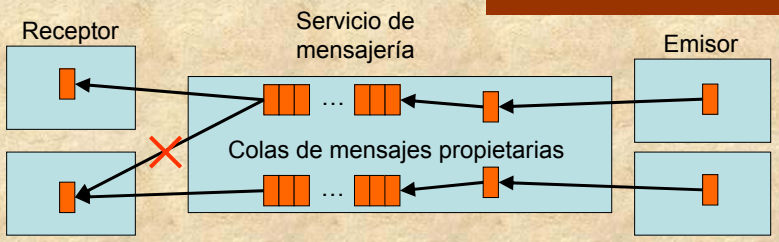
- Ejemplo: Napster, para intercambio de archivos.
  - Utiliza el modelo C-S para directorio, y
  - P2P entre usuarios.
- Se puede implementar sobre muchos tipos de plataforma, y además:
  - JXTA y Jabber.
- Según Liu: subastador ↔ pujadores.
  - Se parece sospechosamente a C-S.
- Pero … mejor:
  - pujadores comunicándose entre ellos las ofertas.
  - Hay que buscar nuevos algoritmos

Sistemas basados en mensajes

- *Elaboración* del paradigma básico del paso de mensajes.
  - Point to Point Message Model
  - Publish/Subscribe Message Model
    Tablón de anuncios

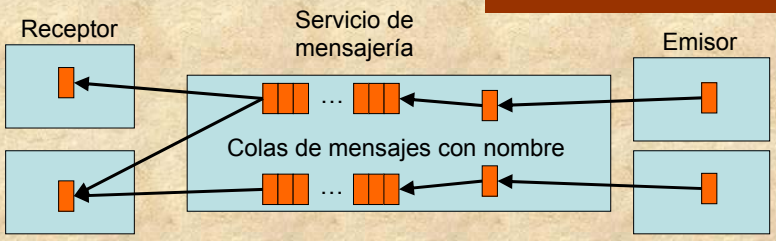


Punto a punto



- Cada receptor tiene una cola propia.
  - Liu se hace un lio
- El receptor puede fijar una política de acceso: urgencia, interés, …

Publish/Subscribe



- Cada cola de mensaje se crea con un 'nombre' para cierto fin.
- Los receptores registran su interés en las colas de mensajes, que reciben eventos de notificación de cambios.
- Es un modelo de multicast basado en tablón de anuncios.

Pub/subs

- Ej. Subastas:
  - El subastador crea una subasta (cola de mensajes) y registra su interés en cierto tipo de evento:
    - 'nueva puja'
  - Los pujadores registran su interés en cierto tipo de evento de la subasta:
    - En este caso 'apertura de pujas', 'cierre de pujas' y 'nueva puja'
  - Cada vez que se incluye un mensaje, en la cola se envía un evento de 'nueva puja'.

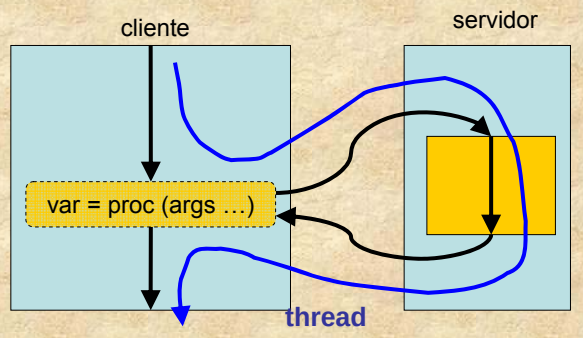
Mensajería

- Los middleware suelen permitir diversos modelos, incluyendo el síncrono.
- Suelen ciertos servicios que deben estar activos en los sistemas participantes.
- Java Messaging service es un ejemplo de servicio de mensajería.
  - Otros conocidos son IBM MQ series y MS MQ.
  - Todos ellos se asientan sobre **servicios de notificación de mensajes**.

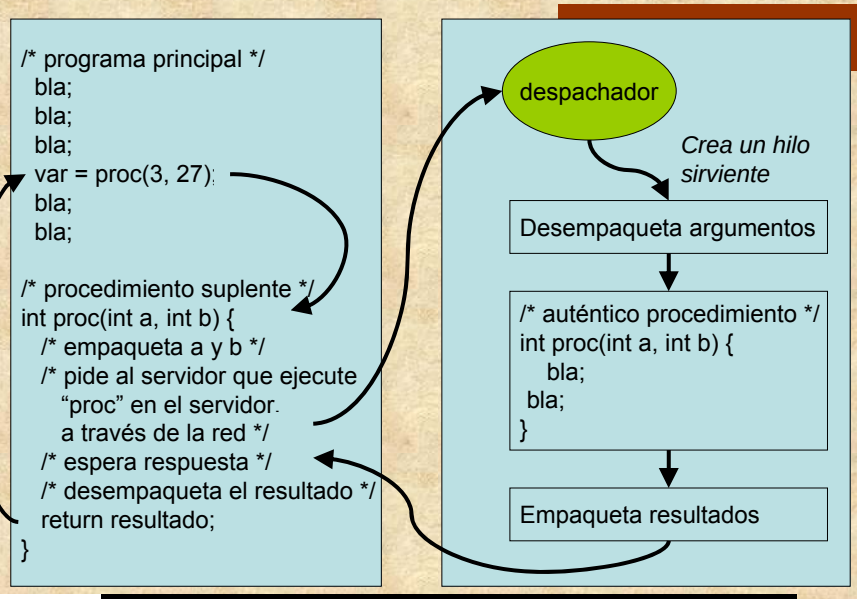
RPC

- Lleva el modelo cliente-servidor al nivel de programación, donde se basa en el mecanismo de invocación de procedimientos locales.
- Aquí el detalle está en que:
  - El código del procedimiento reside en otra máquina
  - … y reside en el contexto de ejecución de otro proceso.
- Es un modelo orientado a acciones, más que orientado a datos.

RPC



- El middleware de RPC se encarga de empaquetar y desempaquetar los argumentos y el valor devuelto.
- También se ocupa de la red y de cumplir con el modelo síncrono de invocación de procedimiento.



RPC

- Estándares:
  - El original es Sun RPC (1982)
  - ONC RPC adoptó el estándar de Sun.
    - Apollo domain, HP, …
  - DCE RPC (de Open Group)
  - Una última versión es XML-RPC que ha evolucionado en SOAP (Simple Object Access Protocol)
- Ejemplo:
  - El servidor ofrece una interfaz con registro y pujas.
  - Los clientes ofrecen una interfaz con: anuncio de subasta, nueva puja, fin de sesión. Ver (D.14)

Objetos distribuidos

- Los objetos reciben mensajes, que disparan métodos de servicio.
- Existen dos aproximaciones:
  - Invocación de métodos remotos, sobre objetos remotos. (RMI)
  - Invocación de métodos remotos sobre un agente intermediario de objetos. (ORB) (mediador de objetos)

RMI

- Invocación de métodos remotos (lo veremos más adelante).

  "Cada clase define un servicio"

- Los procesos pueden manejar *referencias remotas* a objetos remotos.
- El ejemplo es muy similar al caso de antes.

ORB

- Los ORB actúan como buses de mediación que:
  - Implementan el mecanismo de direccionamiento de objetos
  - Dan aspecto de objetos a los procesos conectados a él.
    Permite conectar objetos de servicio en lenguajes muy diversos (Fortran, Cobol, C, …)
  - Y unifican dicho aspecto bajo un :
    protocolo de mediación.
- CORBA (de OMG) sigue esta aproximación (cap. 10). Existen puentes RMI-CORBA (bajo IIOP)

Componentes

- Objetos → ☺ → componentes (¿?)
- Surge un componente cuando
  - El propio software se convierte en clase,
  - se le dota de cierta interfaz
  - y un un ciclo de existencia.
  - De esta forma se convierte en un artefacto utilizable.
- Componentes distribuidos viviendo en su servidor de aplicaciones.
- Ejemplos:
  - DCOM, Enterprise Java Beans, WebServices…

Espacios de objetos

- Generalización de los espacios de tuplas (LINDA)
- Una implementación es JavaSpaces, que es la base de JINI (suena como Genie)
- Un espacio de objetos es una entidad común a un conjunto de procesos distribuidos, con primitivas para:
  - **Depositar** objetos (en principio JavaBeans)
  - **Leer** objetos
  - **Tomar** objetos
  - **Establecer eventos de notificación** sobre cambios en objetos.

Espacios de objetos

- El espacio de objetos se convierte en:
  - Un canal de comunicación entre procesos.
  - Un mecanismo de sincronización entre procesos.

- **Genie** is the English term for the Arabic "jinni | جن". In pre-Islamic Arabian mythology and in Islam, a jinni (also "djinni" or "djini") is a member of the jinn (or "djinn"), a race of creatures. The word "jinn" literally means anything which has the connotation of concealment, invisibility, seclusion and remoteness.

Espacios de objetos

- Tiene un aspecto similar a un bulletin board, en cuanto a que permite un multicast efectivo.
- Pero la interfaz es mucho más flexible

Agentes móviles

- Migración de procesos.
- Exige plataformas compatibles en ejecución.
- Cada entorno de ejecución es diferente
  - Los agentes toman ventaja de este entorno y reducen las comunicaciones.
  - También realizan tareas de seguridad
  - Y plantean problemas de seguridad.

Servicios de red

- Federaciones de servicios de red:
  - Servicio de nombres básico (paginas blancas)
  - Servicio de paginas amarillas
  - Servicio de adherencia a la federación.
- Es una implementación de sistemas basados en agentes.
- Ejemplos:
  - WSDP y JINI

Groupware

- Aproximación muy especializada.
  - Aproximación basada en mensajes "todos con todos".
    - Lotus Quickplace
  - Aproximación de whiteboard "Tablero de dibujo común"
    - Blackboard???
    - SMART,
    - Netmeeting

Pros y Cons

- Middleware (véase D.9)
- Abstraction vs. Overhead (sobrecarga)
  - Runtime support → tiempo extra y recursos extra.
- Scalability
  - Es más simple con aprox. Más abstractas.
- Platform independency
  - Ej. Java RMI vs CORBA vs DCOM vs .NET …

Puntos de interés

- Elegir una herramienta visto desde S.E. …
  - Madurez y estabilidad de la herramienta
  - Tolerancia a fallos provista por la herramienta
  - Disponibilidad de las herramientas de desarrollo
  - Mantenibilidad
  - Reutilización de código.
  - ….y
  - Facilidad de aprendizaje y
  - Gestionabilidad del proceso generado.

Resumen

- Modelos vistos
  - Paso de mensajes
  - Cliente servidor
  - P2P
  - Sistemas de mensajes
  - RPC y RMI
  - Espacios de objetos
  - Agentes móviles
  - Servicios de Red
  - Aplicaciones colaborativas.
- Trade-offs