Diseño Y Aplicaciones de Sistemas Distribuidos

Introducción

Joan Vila

DISCA / UPV

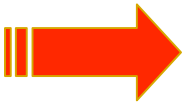
Departament d'Informàtica de Sistemes i Computadors

Universitat Politècnica de València

DISCA
DEPARTAMENT D'INFORMÀTICA DE SISTEMES I COMPUTADORS

Introducción

- **Indice**



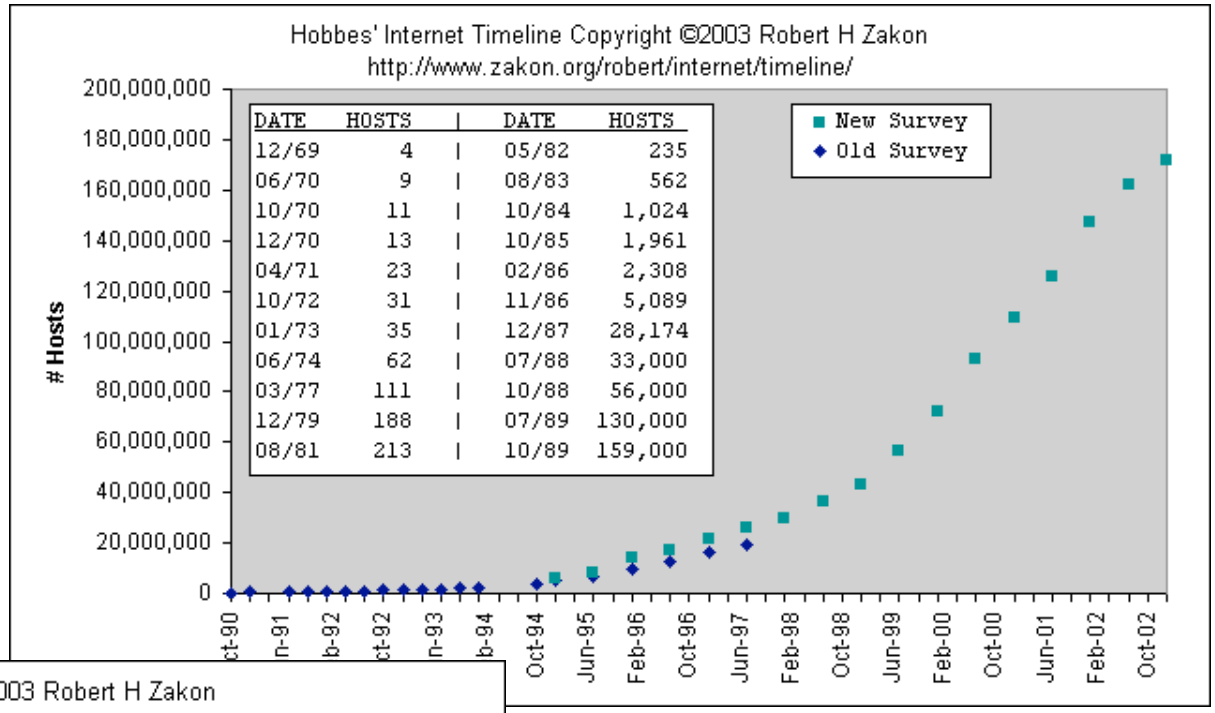
  - Conceptos básicos
  - Modelos de computación distribuida
    - Cliente-servidor
    - Movilidad de código
    - Grupos
  - Arquitecturas para aplicaciones distribuidas
    - Microkernels
    - Middleware
    - Servicios web

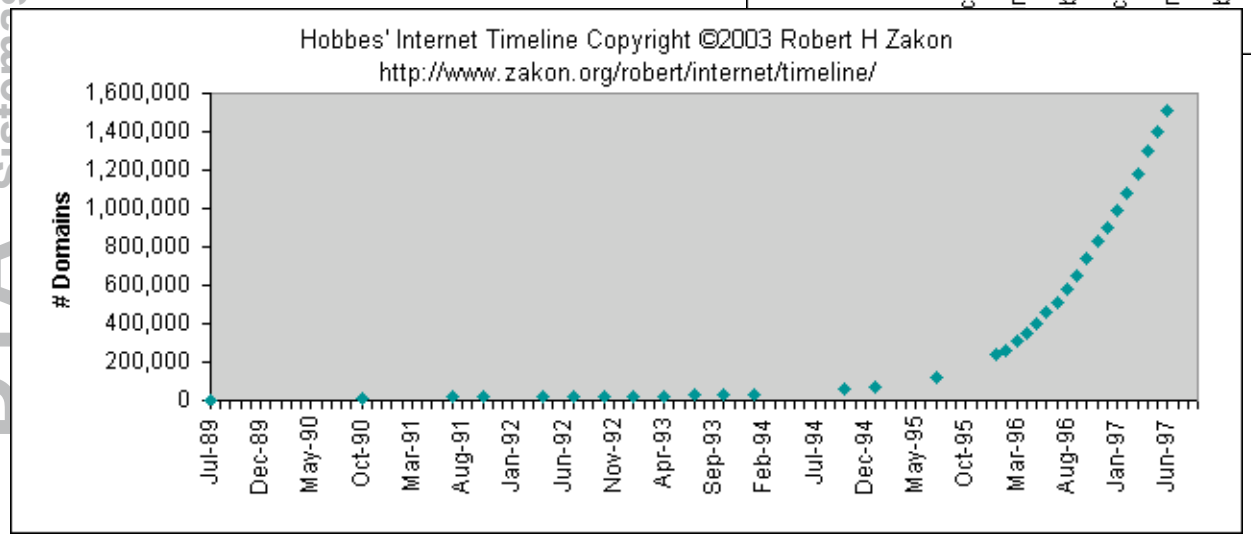
Conceptos básicos

- Sistema distribuido
  - Conjunto de computadores **débilmente acoplados** (independientes) interconectados a través de una red que colaboran con el fin de realizar una tarea.
    - Es un sistema **heterógeneo** (arquitecturas, sistemas operativos, protocolos, ... diferentes)
    - Diferentes tipos de redes: LAN's, WAN's, wifi's, …
    - Tendencia a computadores pequeños y muy potentes interconectados

Conceptos básicos

- Crecimiento
  - Marcado por la web...



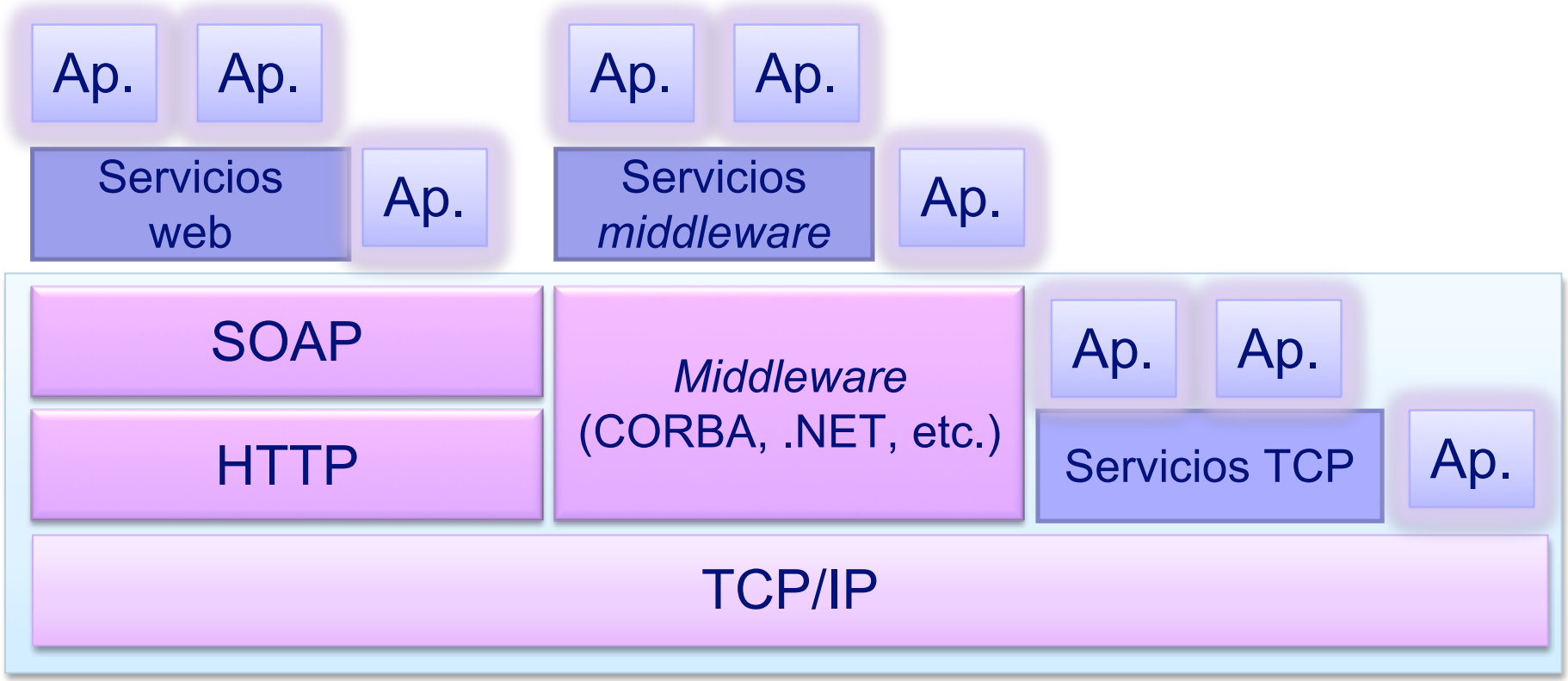


Conceptos básicos

- Computación distribuida
  - Computación que se ejecuta sobre un sistema distribuido. Los programas son bastante **autónomos** y acceden eventualmente a la red.
  - Pueden clasificarse en tres niveles:
    - **Servicios de red**: capa de servicios "estándar" en Internet proporcionado por un servidor (o red de servidores) como HTTP, DNS, FTP, SMTP. Es la mitad servidora del "cliente-servidor" y sirve para construir servicios y aplicaciones de red.
    - **Servicios Web**: conjunto de servicios ofertados en la web sobre el protocolo HTTP, tales como motores búsqueda de Información (Google), servicios de directorio, cambio de moneda, sms,..... Utilizan SOAP y XML como forma de especificarse. Algunos son "de pago".
    - **Aplicaciones de red**: aplicaciones específicas desarrolladas por usuarios finales, como aplicaciones de comercio electrónico, *chats*, juegos de red, etc... Utilizan diferentes tipos de *middleware* como sustrato de comunicaciones.

Conceptos básicos

- Protocolos y niveles de aplicaciones distribuidas

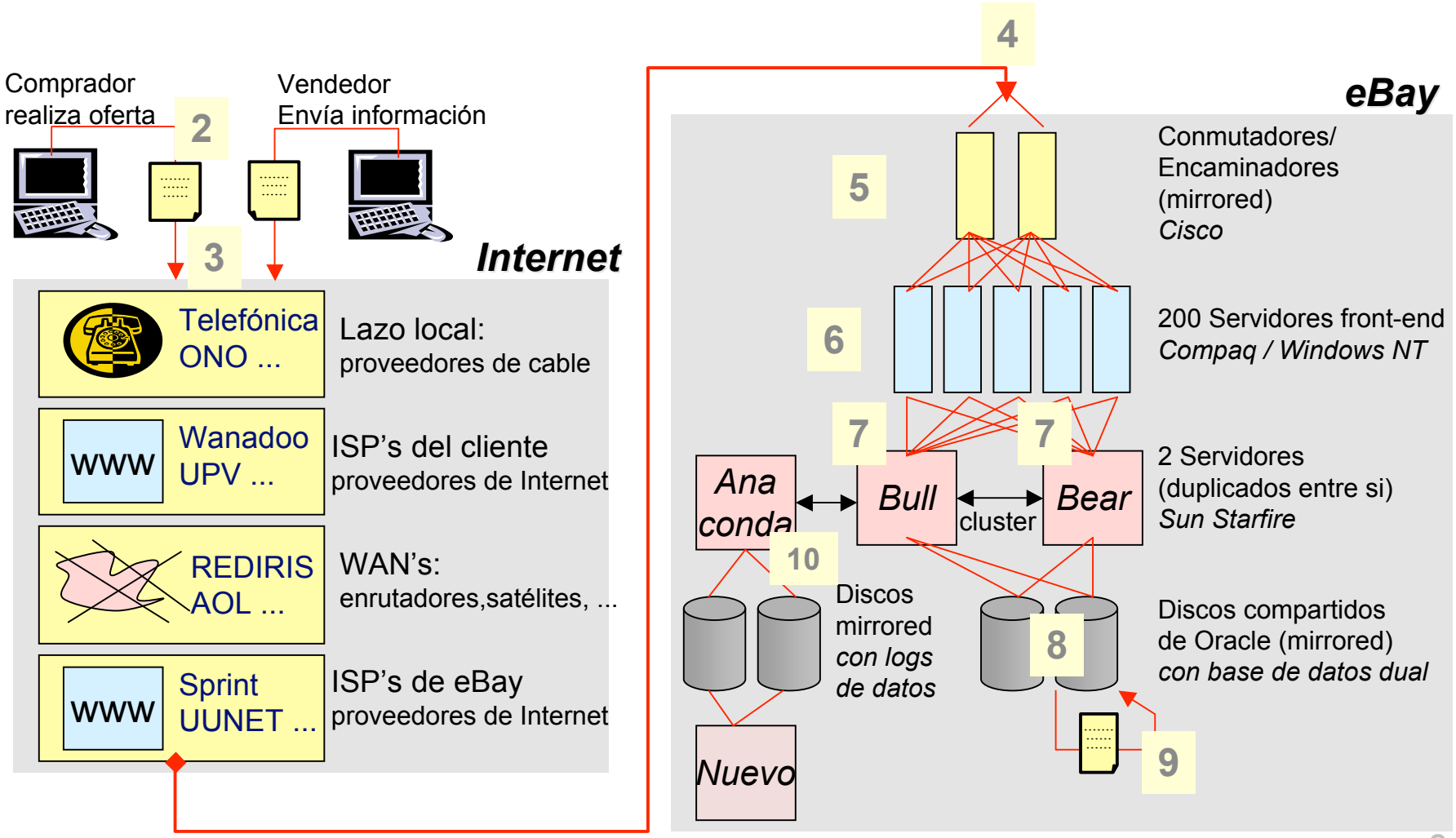


Conceptos básicos

- Tipos de aplicaciones distribuidas
  - Aplicaciones en WANs
    - **Aplicaciones web**: venta de artículos, comercio electrónico, accesos a bases de datos
    - **Transacciones distribuidas**: sistemas bancarios
    - **Aplicaciones multimedia**: videoconferencia, video bajo demanda
  - Aplicaciones en LANs
    - **Aplicaciones industriales**: control de procesos, robótica. Switched Ethernet está sustituyendo a los buses de campo.
      - Topologías en estrella y anillo. Configuraciones con anillos redundantes.
    - **Sistemas empotrados y vehículos**: Switched Ethernet se ha convertido en estándarde aviónica (ARINC 664).

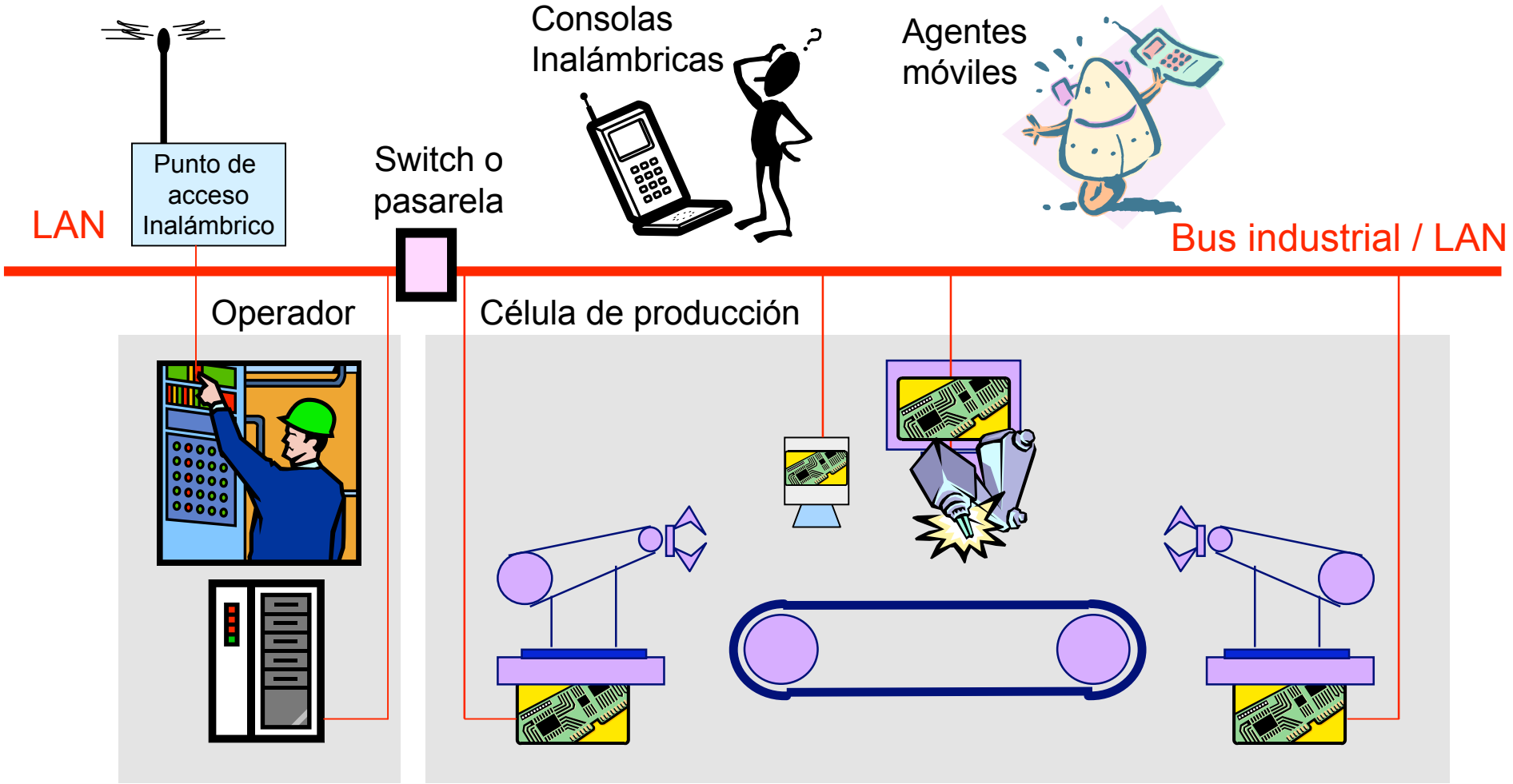
Conceptos básicos

- Taxonomía de una aplicación en una WAN



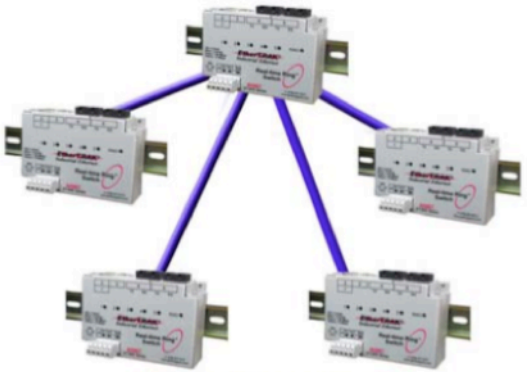
Conceptos básicos

- Taxonomía de una aplicación en una LAN

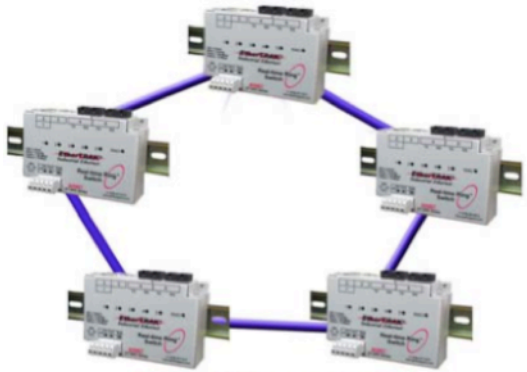


Conceptos básicos

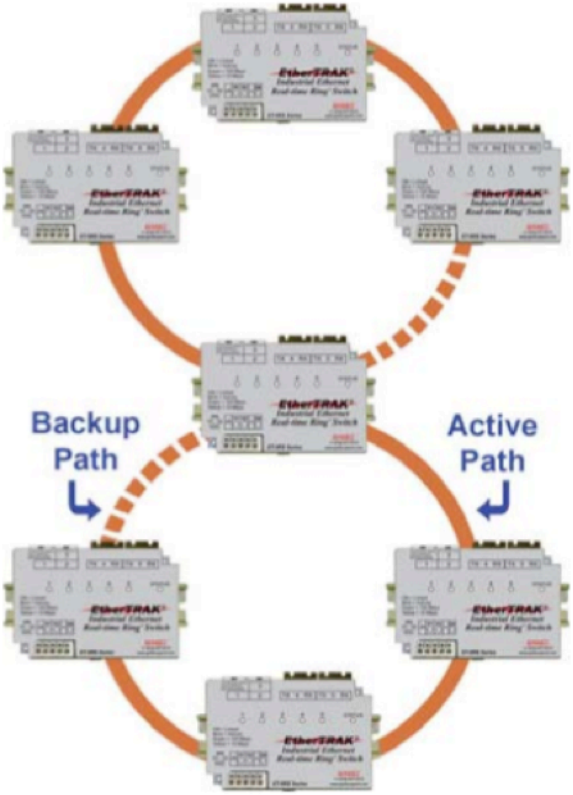
- Topologías de Switched Ethernet en LANs industriales



Star Topology



Ring Topology



Conceptos básicos

- Ventajas de la computación distribuida
  - Internet permite acceso universal a información y recursos
  - Acceder y compartir recursos remotos
    - Se pueden compartir **recursos físicos** (p.e. Impresoras, discos,...) **o información** (p.e. páginas web o bases de datos)
    - Compartir recursos permite **reducir costes y aumentar su utilización**
  - Se puede repartir la carga y tolerar fallos
    - La carga muchas veces se reparte de forma geográfica: la información puede **replicarse** total o parcialmente por zonas y cada zona estar atendida por un servidor diferente
    - Los recursos críticos pueden **replicarse**.

Conceptos básicos

- Problemática
  - Heterogeneidad
  - Búsqueda de servicios e información
  - Congestión de la red y congestión de recursos (calidad de servicio)
  - Seguridad
  - Asincronía
  - Fallos parciales

Conceptos básicos

- Heterogeneidad
  - Los componentes pueden estar construidos con tecnologías diferentes: arquitectura, s.o., redes y protocolos, lenguajes de programación.
  - **Soluciones**: interoperabilidad, sistemas abiertos
    - **Interoperabilidad**: integrar sistemas y componentes heterogéneos en un "todo" sin costosos desarrollos ad-hoc. Normalmente se logra a través de protocolos y estándares.
    - **Sistemas abiertos**: permiten ampliar su funcionalidad con nuevos componentes a través de interfaces claras.
- Búsqueda de servicios e información
  - Cómo encontrar un servicio ofertado en la red o una determinada información.
  - **Soluciones**: servicios de nombres (catálogos de servicios), buscadores.

Conceptos básicos

- Congestión de la red y de recursos
  - Los recursos compartidos pueden convertirse en cuellos de botella. Un servicio que funciona bien con 500 clientes, ¿Funcionará bien con 500.000?
  - Los recursos multimedia consumen muchos recursos y congestionan la red.
  - **Congestión de recursos**: escalabilidad, descentralización, replicación
    - *Escalabilidad*: las prestaciones no deben degradarse conforme aumenta el número de clientes.
    - La escalabilidad se basa en la *descentralización*: evitar soluciones donde existe un componente que interviene en el procesamiento de todas las peticiones. De ahí la famosa frase “Los ciclos de servidor son de oro”.
    - La descentralización requiere un reparto de la carga que normalmente implica *replicación* total o parcial de datos o recursos.

Conceptos básicos

- Congestión de la red y de recursos
  - **Congestión de la red**: autonomía de los nodos, gestión de la Calidad de Servicio (QoS: Quality of Service)
    - *Autonomía* de los nodos: los nodos deben minimizar la dependencia la red. Ello les permitirá seguir funcionando ante eventuales fallos de la misma:
    - La autonomía requiere *caching/replicación* de datos: si se mantiene una memoria local con los datos más frecuentemente utilizados, se reducirá el número de accesos a la red.
    - *Calidad de Servicio (QoS)*: técnicas de gestión de la red y contratos para garantizar un ancho de banda y unos retrasos máximos acotados

Conceptos básicos

- Seguridad
  - Las redes de difusión son proclives a la intromisión de intrusos que pueden fisgonear la información y utilizar recursos sin autorización.
  - La autentificación de las partes es crítica en las transacciones.
  - El código migrado es un virus en potencia.
  - **Soluciones**: criptografía, autentificación de clientes
    - La *criptografía* permite cifrar la información para evitar el fisgoneo. Las *firmas digitales* permiten autentificar a clientes, proveedores de servicios, fuentes de código migrado.

Conceptos basicos

- Fallos parciales
  - Las aplicaciones no fallan por completo, sino que determinados componentes pueden dejar de operar y otros seguir funcionando (independencia en el modo de fallo).
  - Los fallos parciales pueden ser una desventaja más que una ventaja pues las caídas de nodos pueden provocar inconsistencias.
    - Sistemas AND: sistemas en los que con que falle un componente, la aplicación deja de funcionar.
    - Sistemas OR: pueden seguir funcionando con que funcione al menos algún componente.
  - **Solución**: descentralización y redundancia.

Conceptos basicos

- Asincronía
  - La comunicación y las acciones de cómputo no están guiadas por un único reloj global. Cada nodo tiene su reloj y puede haber derivas entre ellos.
  - Esto dificulta la sincronización entre los componentes de un sistema distribuido y la determinación de la relación “antes/después” entre eventos remotos.
  - **Solución**: protocolos de sincronización de relojes y utilización del concepto de causalidad.

Conceptos básicos

- ¿Transparencia?
  - En un principio se pensó que ocultar al usuario el hecho de la distribución de componentes o incluso su replicación era la base para estructurar un sistema distribuido.
  - Hoy en día esto se ha revelado imposible, aunque principios como la **uniformidad de nombre** o la **uniformidad de acceso** se consideren deseables.

Introducción

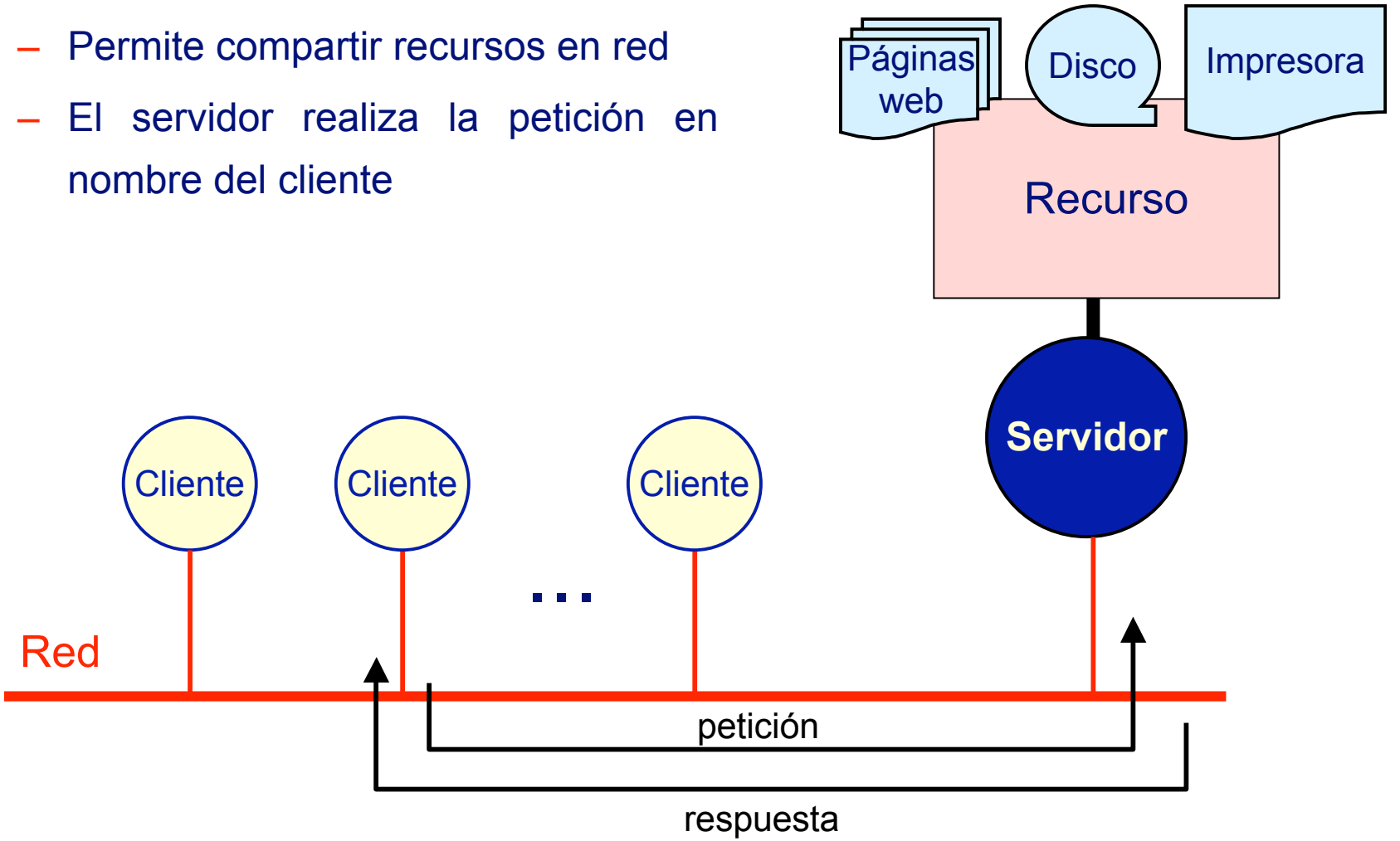
- **Indice**
  - Conceptos básicos
  - Modelos de computación distribuida
    - Cliente-servidor
    - Movilidad de código
    - Grupos
  - Arquitecturas para aplicaciones distribuidas
    - Microkernels
    - Middleware
    - Servicios web

Modelos de computación distribuida

- Paradigmas de computación
  - Cliente-Servidor
  - Grupos
  - Movilidad de código

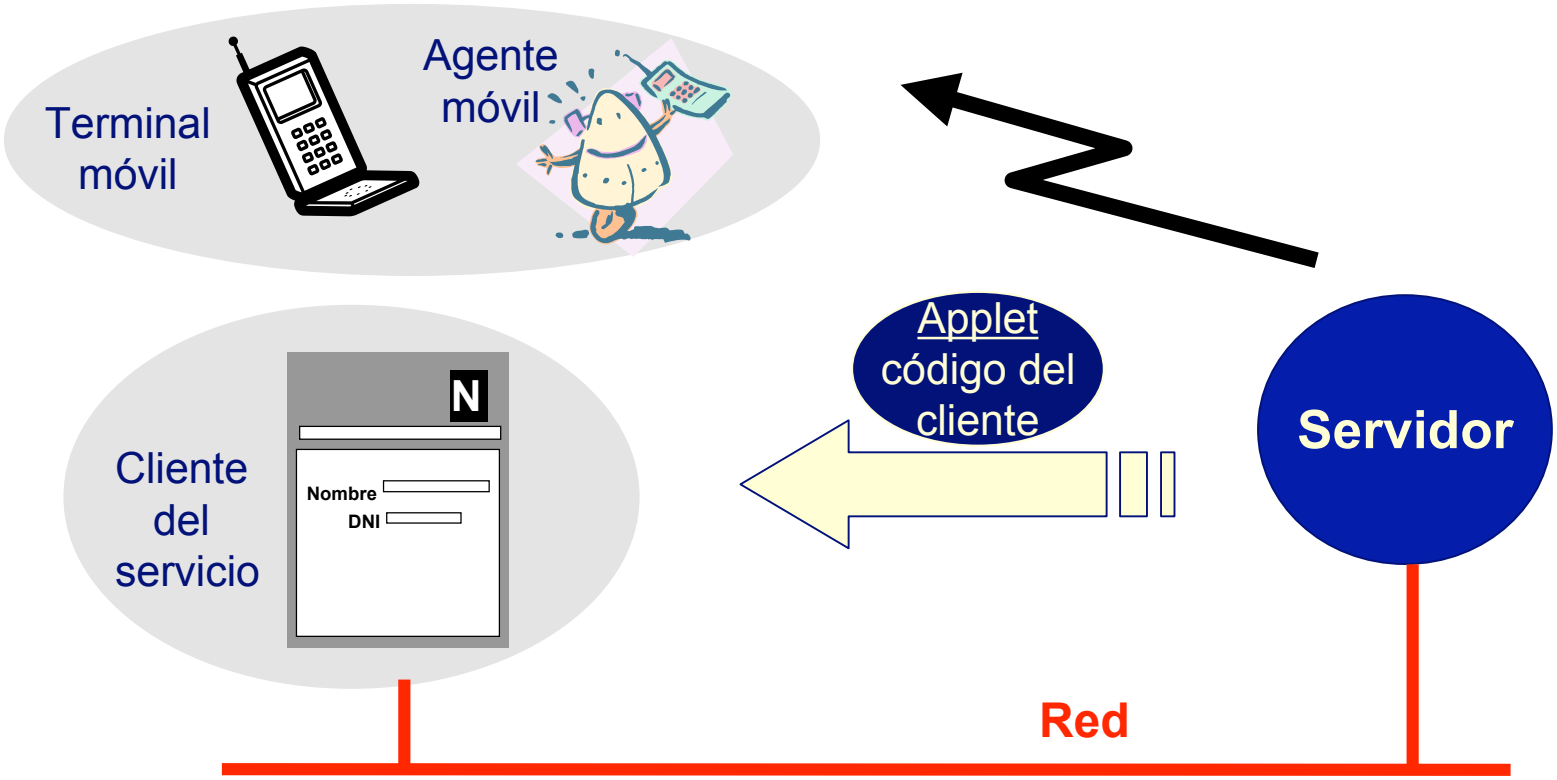
Cliente-servidor

- Modelo cliente-servidor
  - Permite compartir recursos en red
  - El servidor realiza la petición en nombre del cliente



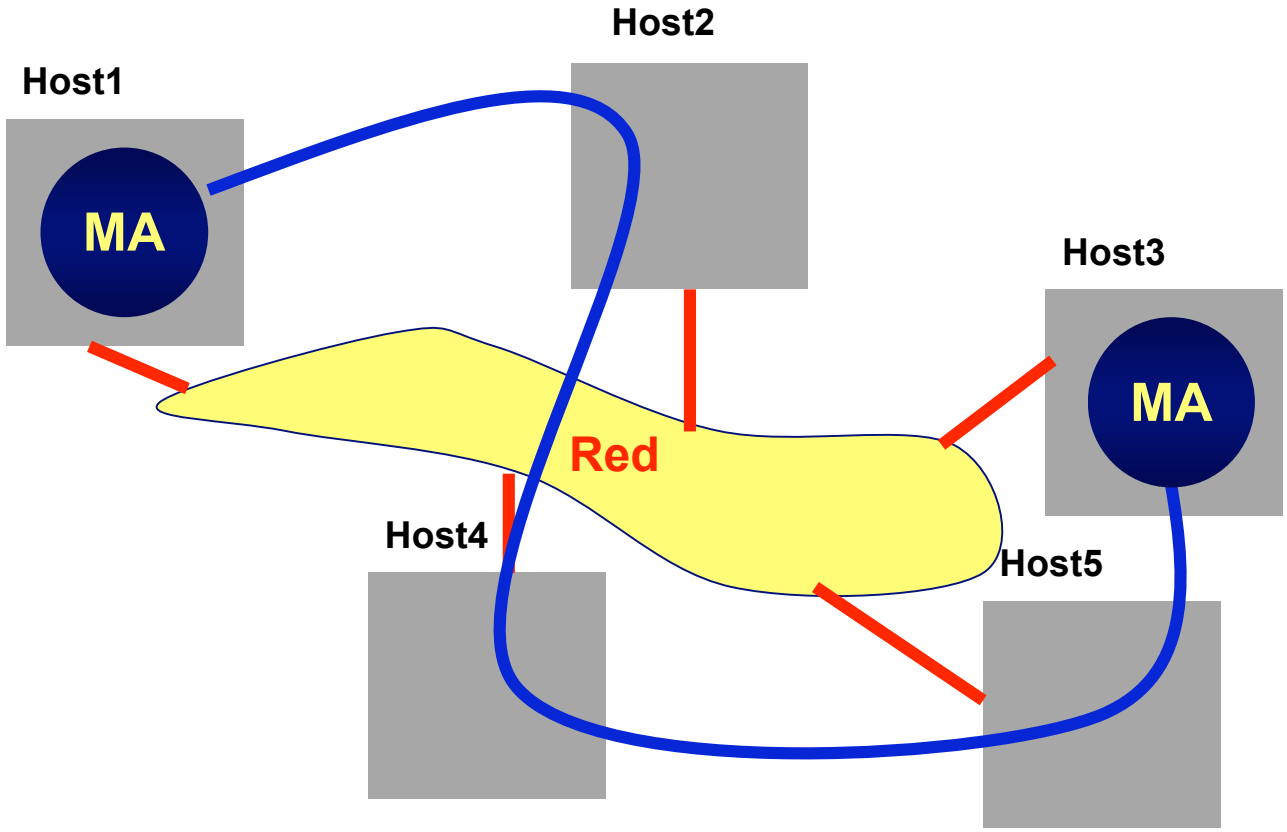
Movilidad de código

- Movilidad de código (i)
  - El servidor envía al cliente el código del servicio
  - Ejemplo: applets, teléfonos móviles



Movilidad de código

- Movilidad de código (ii)
  - Agentes móviles:

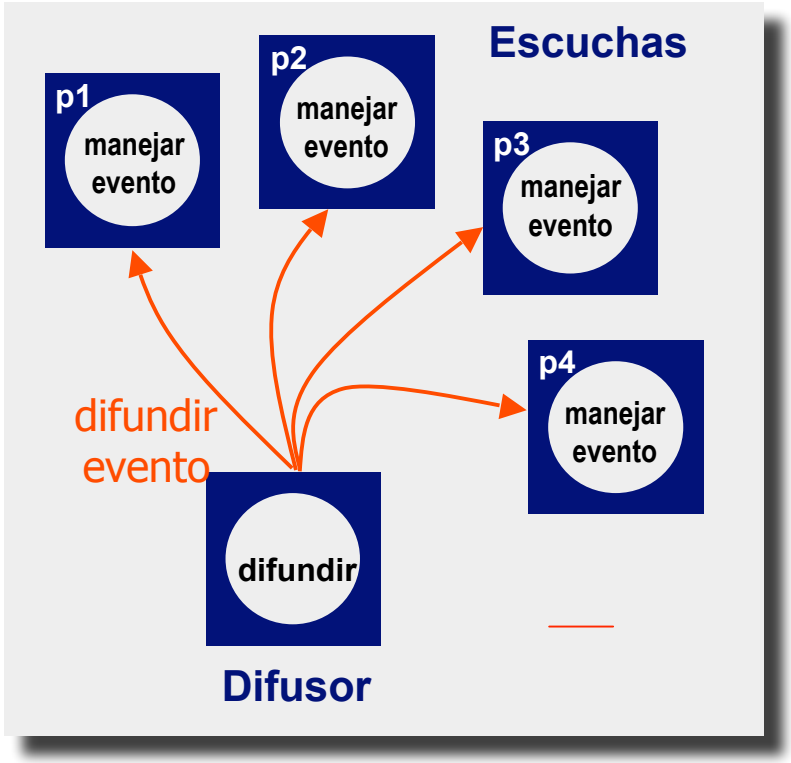


Movilidad de código

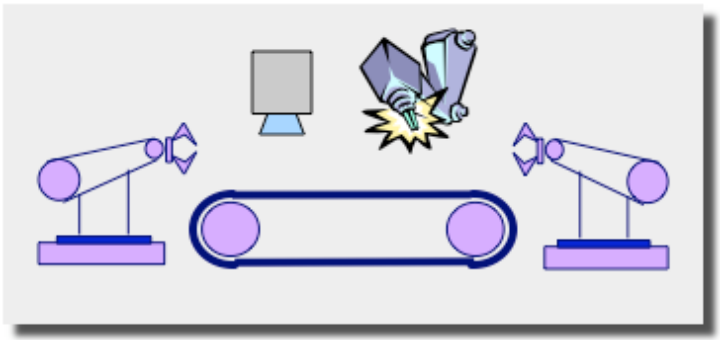
- Movilidad de código
  - **Clientes web (applets)**: la parte cliente de la aplicación es un navegador web que puede programarse para ser cliente de cualquier aplicación.
    - No hay necesidad de instalación previa del programa del cliente.
    - Reducir la comunicación, realizando el cómputo en la máquina donde se migra en lugar del servidor.
  - **Agentes móviles**: agentes "inteligentes" que pueden desplazarse por la red, ejecutándose en los distintos nodos.
    - El programa escoge cuando y donde migrar. Se transporta asímismo a otra máquina y activa y suspende su ejecución a voluntad.
    - Objetivos: recolección de datos, búsqueda de información, etc...
      - Comercio electrónico
      - Procesamiento paralelo
  - Problemas:
    - Seguridad: son virus en potencia (gusanos).
    - Ejecución en un entorno heterogéneo. Precisan ejecutarse en máquina virtual.

Grupos

- Grupos
  - **Comunicación reactiva:** sigue el patrón difusor / escucha. Se utiliza para notificar eventos.



Célula de producción



Robocup

Introducción

- **Indice**
  - Conceptos básicos
  - Modelos de computación distribuida
    - Cliente-servidor
    - Movilidad de código
    - Grupos
  - Arquitecturas para aplicaciones distribuidas
    - Microkernels
    - Middleware
    - Servicios web

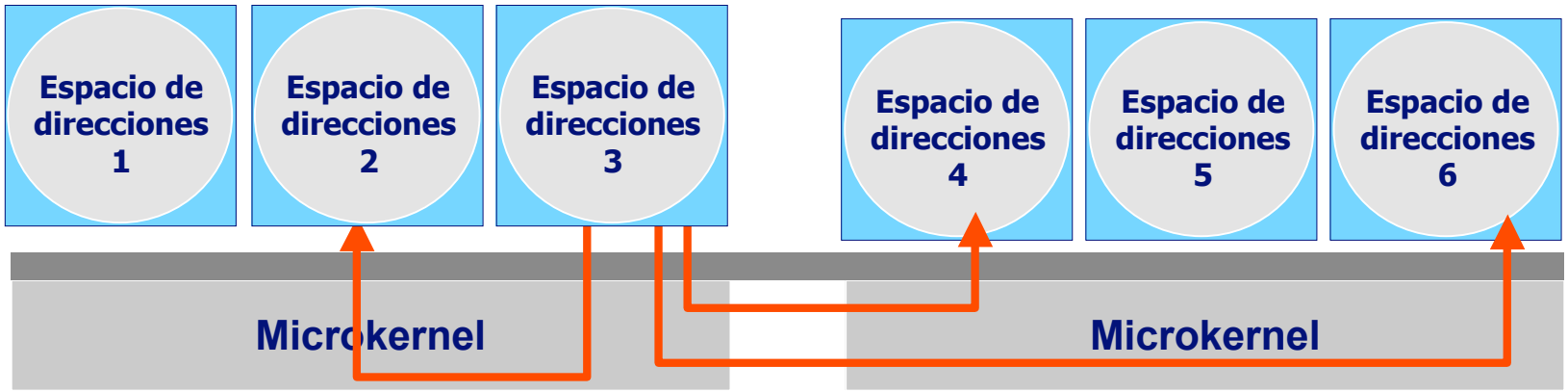
Introducción

- Evolución histórica

  - Soporte a nivel de núcleo
    - **Kernels y microkernels**
      - Ej: RT-Mach, Clouds, ...
  - Soporte a nivel de usuario
    - **Middleware**
      - Ej: CORBA, Java-RMI
  - Soporte a nivel de web
    - **Servicios web**
      - Ej: XML-RPC, SOAP

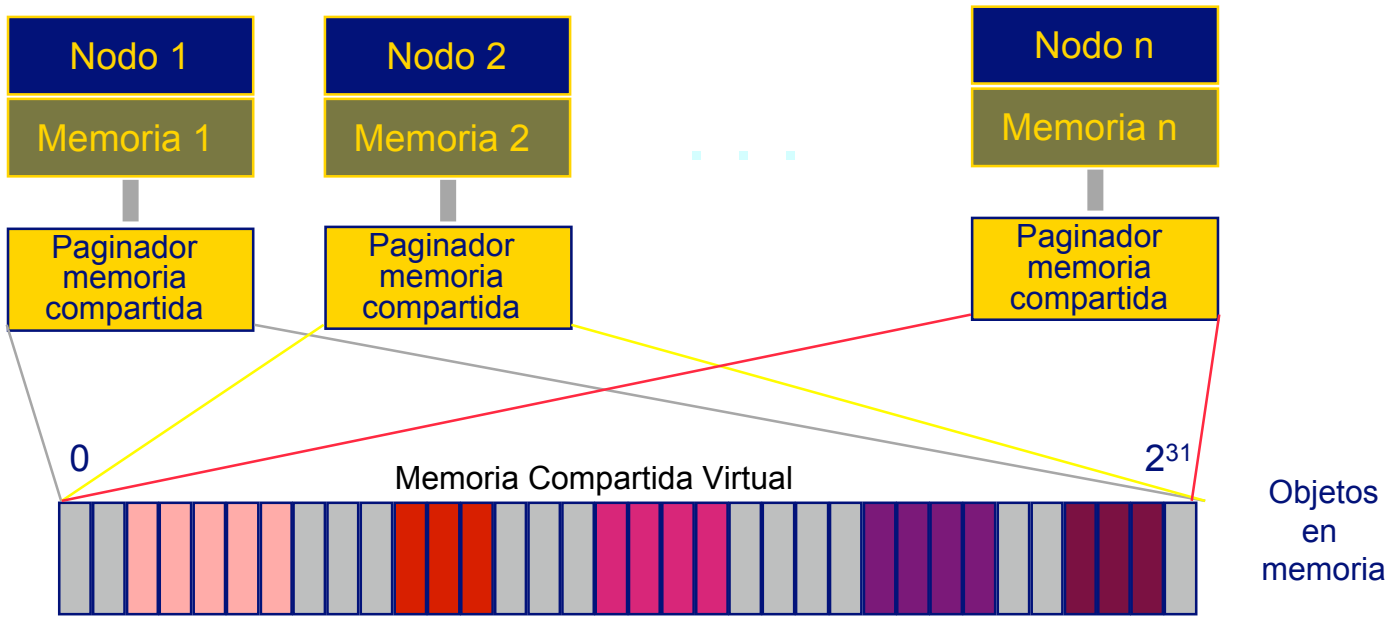
Microkernels

- Kernels y microkernels
  - El kernel proporciona una funcionalidad mínima:
    - **Espacio de direcciones:** memoria lógica para ubicar los componentes de una aplicación distribuida
    - **Mecanismo de comunicación** entre espacios de direcciones
    - **Hilo de ejecución** dentro de un espacio de direcciones
    - **Mecanismo protección.**
  - Cualquier servicio del sistema operativo se implementa fuera del núcleo, como un componete más:
    - Servicio de ficheros, Servicio de nombres, ...
  - **Aspecto clave:** comunicación uniforme entre espacios de direcciones locales y remotos



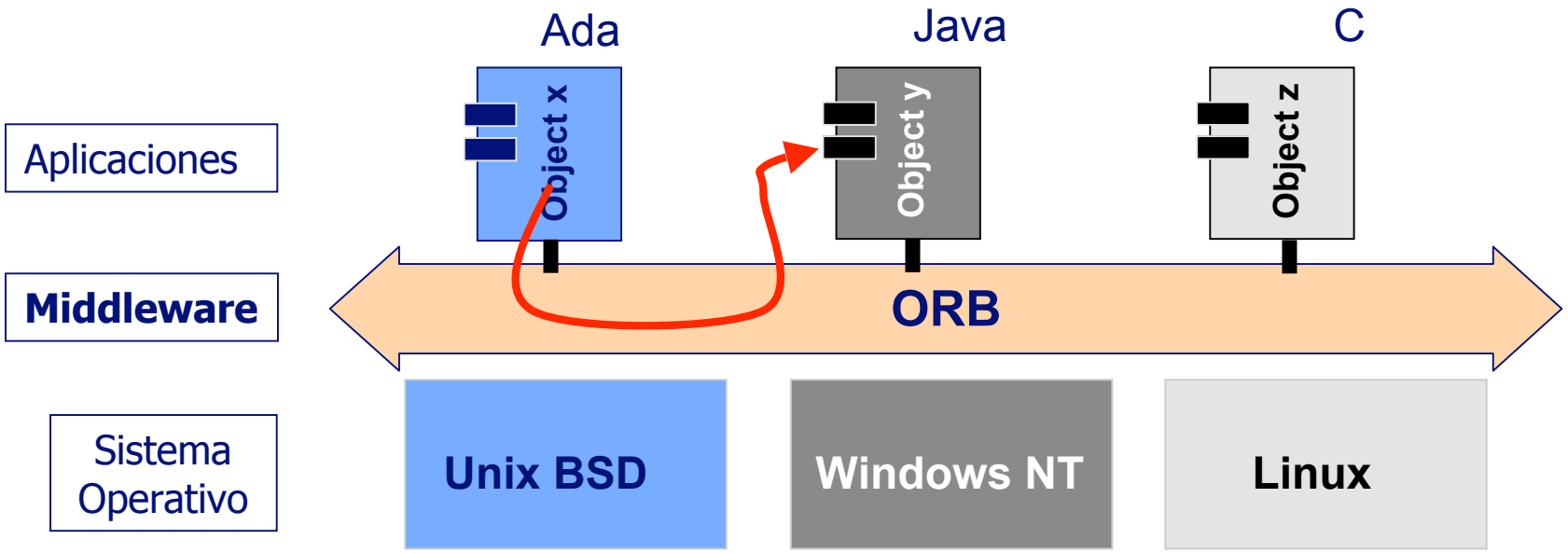
Microkernels

- Memoria compartida distribuida
  - Espacio de direcciones lineal en memoria virtual al que tienen acceso todos los procesadores.
  - Las memorias locales actúan como una *cache* de la memoria virtual: la página física que da soporte a una página lógica puede residir en cualquier procesador.
  - Los fallos de página pueden ser servidos por un paginador local o remoto.



Middleware

- Middleware
  - Bus software a nivel de usuario que proporciona **interoperabilidad** entre objetos en un entorno distribuido heterogéneo.
  - Asume el **modelo de objetos** e invocaciones a métodos remotos



Middleware

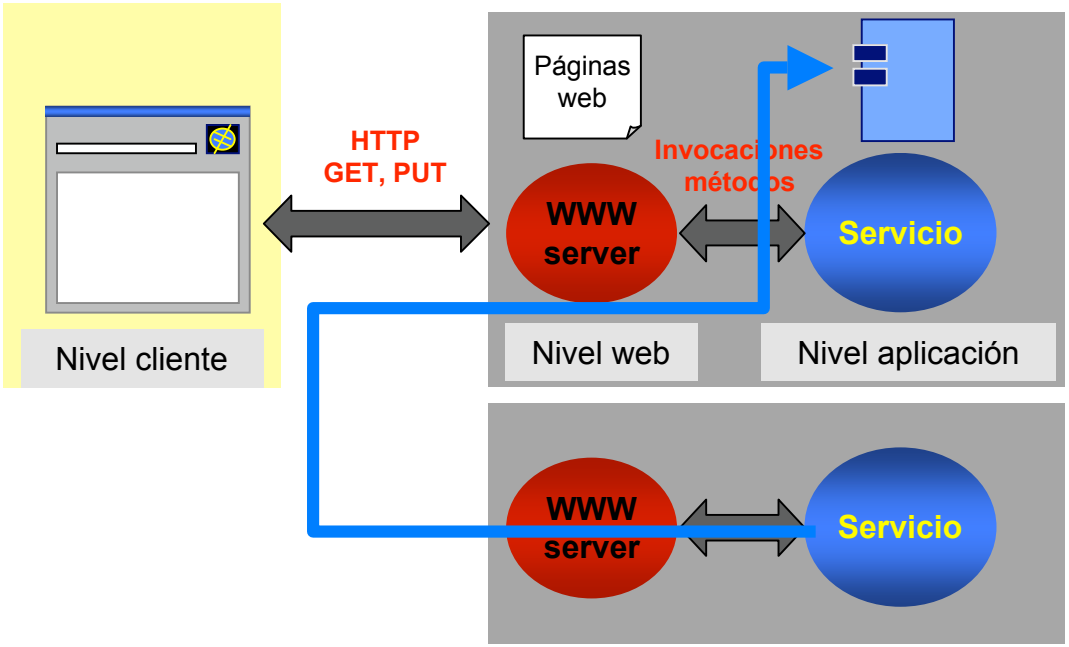
- Middleware

  Existen, fundamentalmente, 3 proyectos Middleware importantes:

  - **CORBA**
    - **Es un estándar del consorcio de fabricantes OMG**
    - **Genérico:** no cubre los requerimiento de aplicaciones multimedia.
  - **Java-RMI**
    - **Independencia de plataforma:** la máquina virtual Java se ejecuta sobre muchas plataformas
    - **Protocolos e interfaces genéricos**: el protocolo RMI es propietario y no permite comunicar, en principio, con sistemas no Java. Los interfaces se definen en Java. Para solucionar esto existen clases CORBA.
  - **DCOM (Microsoft), .NET**
    - **Independencia de plataforma:** restringido a Microsoft.
    - **Interfaces y protocolos estándar**: utiliza IDL y la tecnología del proyecto DCE (anterior a CORBA) del consorcio OSF. Existen puentes DCOM/CORBA

Servicios web

- Servicios web
  - Son objetos o rutinas en Internet disponibles a programas CGI en Perl, PHP,... (no tienen interfaz de usuario como pág. web necesariamente).
    - **Ejemplos**: servicio de fax, servicio de SMS,..
  - Se basan en transformar invocaciones a objetos en peticiones HTTP (GET, PUT) y utilizan XML como lenguaje de intercambio de datos.
    - **Protocolos**: XML-RPC SOAP



Arquitecturas

- Comparación
  - Microkernel
    - Aplicable a **sistemas homogéneos, cerrados**
    - Poco compatibles / interoperables con otros sistemas
    - Protocolos de comunicación propietarios o TCP/IP
    - El soporte para tiempo real es fácil de introducir
  - Middleware
    - Aplicable a **sistemas heterogéneos, abiertos o cerrados**
    - Interoperabilidad es uno de los objetivos fundamentales
    - Protocolos de comunicación estandarizados (IIOP)
    - Modelo de tiempo real de sistemas orientados a objetos. Descansa sobre las facilidades
  - Servicios Web
    - Aplicable a **sistemas heterogéneos, abiertos**
    - Protocolo de comunicación basado en web (muy disponible)