Sistemas Distribuidos

Dr. Víctor J. Sosa S.
vjsosa@tamps.cinvestav.mx

Página del curso:
http://www.tamps.cinvestav.mx/~vjsosa/clases/sd/

Bibliografía

• **Distributed Systems: Concepts and Design**
G. Coulouris, J. Dollimore y T. Kindberg; Addison-Wesley

• **Distributed Operating Systems**
A. S. Tanenbaum; Prentice-Hall

• **Distributed Systems: Principles and Paradigms**
A. S. Tanenbaum y M. Van Steen; Prentice-Hall

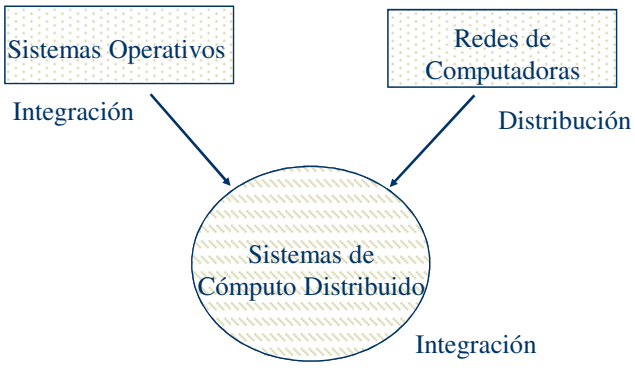
• **Distributed Operating Systems: Concepts & Practice**
D. L. Galli; Prentice-Hall

Distributed Systems: An Algorithmic Approach.
Sukumar Ghos. Chapman & Hall/CRC, Taylor & Francis Group, LLC

Contenido

- Motivación
- Definición de Sistema Distribuido (SD).
- Razones para distribuir.
- Aspectos importantes a considerar.
- Problemas comunes
- Retos
- Ejemplos

Motivación



Integración ≠ Centralización

¿Qué es un SD?

No es algo trivial de definir…....

- Leslie Lamport una vez dijo: "***A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable***".

- "Síntomas" de distribución:
  - Geografía: Sus componentes suelen estar en sitios diferentes
  - Multiproceso (concurrencia): El hardware permite el progreso simultáneo de varias actividades (varias CPUs, con memoria local, etc.).
  - Interconexión: Permite la comunicación entre las actividades.
  - Compartición: Uso compartido de recursos, información, etc.
  - Tolerancia a fallos: Busca soluciones resistentes en caso de fallo (nota: las comunicaciones también pueden fallar).

SD: Definiciones

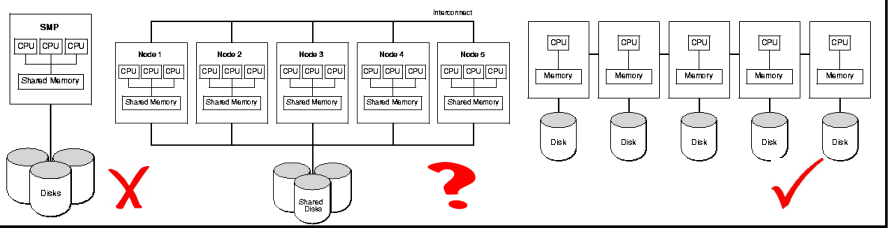
- "Un sistema en el cual componentes conectados a través de una red de computadoras se comunican y coordinan sus acciones mediante el intercambio de mensajes"[Coulouris]. *Principales características*: concurrencia de componentes, ausencia de reloj global e independencia de fallos en sus componentes.

- Un sistema distribuido es una colección de computadoras independientes que dan la apariencia al usuario de ser una computadora única" [Tanenbaum]

SD: Nuestra perspectiva

• Conjunto de procesadores sin memoria común conectados por una red

- Sistema débilmente acoplado
- No existe un reloj común
- Dispositivos de E/S asociados a cada procesador
- Fallos independientes de componentes del SD
- Carácter heterogéneo
- Buscan un objetivo común



Razones para Distribuir

- **Distribución funcional**: las computadoras tienen diferencias funcionales
  - Cliente / Servidor
  - Host / Terminal
  - Recaudación de datos / procesamiento de datos
  - Compartir recursos para funciones específicas
- **Distribución inherente al dominio de la aplicación**
  - Cajas registradoras y sistemas de inventario para cadenas de supermercados
  - Soporte para trabajo colaborativo
- **Balanceo de carga**: asignar tareas a procesadores tal que todo el desempeño del sistema sea optimizado.

Razones para Distribuir

- **Incremento del poder de procesamiento**: procesadores independientes trabajan con la misma tarea
  - Sistemas distribuidos conformados por varias microcomputadoras pueden tener poder de procesamiento que difícilmente una supercomputadora tendrá.
- **Tolerancia a fallos**: evitar un solo punto de fallo.
- **Económicos**: colecciones de microprocesadores ofrecen una mejor cuota precio/desempeño que grandes mainframes.
  - Mainframes: 10 veces mas rápidos, 1000 veces más caros

Razones para Distribuir

- **Conveniencia**:
  - Cuestiones organizacionales, se requiere compartir datos y recursos entre usuarios.
  - Mejora la comunicación persona-a-persona.
  - Permitir autonomía local y promover la evolución de los sistemas y los cambios en los requerimientos del usuario.
  - Diferentes computadoras con diferentes capacidades pueden ser compartidas entre usuarios.

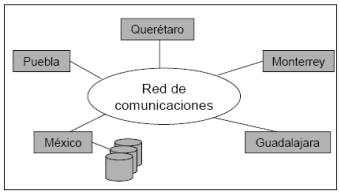
Dificultades

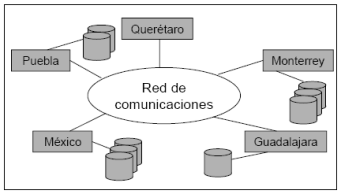
- Lograr un acceso eficiente, presumiblemente óptimo.
- Transformar e integrar datos de diferentes tipos de nodos.
- Distribuir datos de una manera óptima.
- Controlar el acceso a datos.
- Soportar la recuperación de errores en todo el sistema distribuido de manera eficiente y segura.
- Asegurar que los sistemas locales y globales permanezcan como una imagen fiel del mundo real.

Problemas comunes en las aplicaciones distribuidas

- Elección de líder
- Exclusión mutua
- Sincronización del tiempo
- Estado global
- Gestión de réplicas

Cómputo Centralizado vs Distribuido





Sistema Distribuido vs Sistemas Paralelos

- Algunos autores indican que los SD son subconjunto de los SP, sin embargo no es algo que todos acepten. En ambos tipos de sistemas se pueden emular cosas como paso de mensajes y memoria compartida

- ¿Porqué un sistema no debiera ser paralelo en su totalidad?
  - Escalabilidad
  - Modularidad y heterogeneidad
  - Datos compartidos (bases de datos distribuidas)
  - Recursos compartidos
  - Estructura geográfica
  - Confiabilidad (resistencia a fallos)
  - Bajo costo

Sistema distribuido heterogéneo

Un sistema distribuido puede estar formado por multitud de elementos conectados por redes LAN o WAN:

– Terminales X y *Network Computers.*
– PCs y estaciones de trabajo.
– Sistemas portátiles (redes móviles: WiFi, UMTS,GSM, WAP, ...).
– Minicomputadores.
– Supercomputadores.
– Multiprocesadores con memoria compartida.
– Servidores especializados (de almacenamiento, de impresión, ...).
– Sistemas empotrados (una cámara, un frigorífico, ...).

Ventajas de los Sistemas Distribuidos

- Economía: Buena relación rendimiento/coste
  - Gracias a avances en tecnología de microprocesadores y de comunicaciones
- Alto rendimiento: Procesamiento paralelo.
- Soporte de aplicaciones inherentemente distribuidas.
  - Por ejemplo: empresa distribuida geográficamente
- Capacidad de crecimiento: Escalabilidad.
- Fiabilidad y disponibilidad: Tolerancia a fallos.
- Carácter abierto y heterogéneo:
  - Necesidad de estándares de interoperabilidad.
- Compartir recursos y datos.

Desventajas de los Sistemas Distribuidos

- Necesidad de un nuevo tipo de software:
  - Más complejo.
  - No hay un acuerdo estándar sobre cómo debe ser.
- Dependencia de infraestructura subyacente, e.g. la red de interconexión que introduce nuevos problemas:
  - Pérdida de mensajes y saturación.
  - Latencia puede provocar que al recibir un dato ya esté obsoleto.
- Seguridad y confidencialidad

Aplicaciones de los Sistemas Distribuidos

- Entornos de empresa: redes corporativas e *intranets*.
  - Sustituyen a los clásicos *mainframes*
- Entornos que requieren procesamiento paralelo.
  - Sustituyen a costosos *supercomputadores*
- Servicios con alta disponibilidad y rendimiento.
- Sistemas distribuidos de gestión de bases de datos.
- Aplicaciones multimedia.
- Sistemas industriales distribuidos y aplicaciones de control.
- Internet es un enorme sistema distribuido.

Objetivos de un Sistema Distribuido

En general el desarrollo de sistemas distribuidos intenta poner solución a los siguientes objetivos:

– Transparencia.
– Fiabilidad.
– Rendimiento.
– Capacidad de crecimiento.
– Flexibilidad.
– Seguridad.

Transparencia

Existen varios perfiles de transparencia:

– **Acceso**: Manera de acceder a recurso local igual que a remoto.
– **Posición**: Se accede a los recursos sin conocer su localización.
– **Migración**: Recursos pueden migrar sin afectar a los usuarios.
– **Concurrencia**: Acceso concurrente no afecta a los usuarios.
– **Replicación**: La existencia de réplicas no afecta a los usuarios.
– **Fallos**: La ocurrencia de fallos no afecta a los usuarios.
– **Crecimiento**: El crecimiento del sistema no afecta a los usuarios.
– **Heterogeneidad**:Carácter heterogéneo no afecta a los usuarios.

¿Es buena tanta transparencia?
– A veces el usuario precisa conocer cómo es el sistema subyacente

Fiabilidad

Fiabilidad como **disponibilidad**:

- Teóricamente: OR-lógico de sus componentes.
- En ciertos casos: AND-lógico de varios componentes.
- Mecanismos: redundancia y evitar componentes críticos.

Fiabilidad como **coherencia**:

- Se dificulta con *caching* y redundancia

La fiabilidad está relacionada con la seguridad (otro objetivo).

Rendimiento

Rendimiento para un **servicio multiusuario**:

- Objetivo: Rendimiento no peor que un sistema centralizado

Rendimiento para la **ejecución paralela** de aplicaciones:

- Objetivo: Rendimiento proporcional a procesadores empleados

Factores:

• Uso de esquemas de *caching*

- Intentar que muchos accesos se hagan localmente

• Uso de esquemas de replicación

- Reparto de carga entre componentes replicados

• En ambos casos: Coste de mantener la coherencia

Capacidad de crecimiento

Diseño de un sistema distribuido **debe evitar** "cuellos de botella":

- Componentes centralizados
- Tablas centralizadas
- Algoritmos centralizados

Características deseables en un algoritmo distribuido:

- Ninguna máquina tiene información completa del estado del sistema
- Las decisiones se basan sólo en información disponible localmente
- El fallo de una máquina no debe invalidar el algoritmo
- No debe asumir la existencia de un reloj global

Flexibilidad

SOD debe ser adaptable:

- facilidad para incorporar cambios y extensiones al sistema

Uso preferible de arquitectura microkernel

Importancia de **sistemas abiertos**:

- Sus interfaces y protocolos deberían ser públicos.
- Contrario a *"tecnología propietaria"*.
- Uso de estándares siempre que sea posible.
- Disponibilidad de su código fuente (libremente o no).
- Regulación por parte de un colectivo (usuarios u organizaciones) y no por particulares (fabricantes).

Componentes de un Sistema Distribuido

El desarrollo de un sistema distribuido complejo requiere el uso de las siguientes funciones y servicios:

– Servicios de comunicación.
– Sistemas de archivos.
– Servicio de nombres.
– Servicios de sincronización y coordinación.
– Memoria compartida distribuida.
– Gestión de procesos.
– Servicio de seguridad.

Servicios de comunicación

• Modelos de interacción:
  – Cliente/servidor (2-niveles, 3-niveles o *n*-niveles)
  – *Peer-to-peer*: Equilibrio de roles.
  – Intermediarios: *Proxy*, *Dispacher*, *Caches*, ...
  – Comunicación en grupo (*Multicast*)
  – Código móvil.

• Tecnologías de comunicación:
  – Paso de mensajes: *sockets.*
  – Llamada a procedimientos remotos (RPC).
  – Invocación de métodos remotos (RMI).
  – Tecnologías de objetos distribuidos: CORBA, DCOM, EJB
  - Servicios Web

Sistemas de Archivos Distribuidos

• Sistema de archivos para sistema distribuido

• Gestiona distintos dispositivos en diferentes nodos ofreciendo a usuarios la misma visión que un SA centralizado

• Permite que usuarios compartan información de forma transparente

• *Caching* y replicación

Servicio de nombres

Identificación y localización de recursos en el entorno distribuido.

Comprende:

– Servicio de nombres (páginas blancas): DNS, COS-Naming (CORBA).
– Servicio de directorio (páginas amarillas): X.500, LDAP, Active Directory de Windows, UDDI (Web Services).

Estrategias de resolución de nombres

Arquitectura de los servicios.

– Almacenamiento intermedio: *caching*.
– Replicación y coherencia.

Servicios de Sincronización y Coordinación

Comprende los conceptos de:

- Tiempo en entornos distribuidos: Sincronización de relojes y relojes lógicos.
- Concurrencia y Paralelismo: Exclusión mutua e interbloqueos.
- Algoritmos distribuidos: Elección de líder, coordinación, ...
- Transacciones: Propiedades ACID, modelos de *commit*/*rollback*.

Afecta a otros servicios:

- Nombrado e identificación.
- Seguridad y fiabilidad.
- Comunicaciones.
- ...

Memoria Compartida Distribuida (DSM)

Concepto:

- Memoria físicamente privada pero lógicamente compartida.

Estrategias de implementación:

- Basada en páginas.
- Basada en variables compartidas.
- Basada en objetos.

Modelos de coherecia

Gestión de Procesos

- Estrategias de asignación de procesadores
- Planificación de procesos:
  - Planificación interna.
  - Planificación global.
- Migración de procesos
  - Equilibrado de carga.
  - Aprovechamiento de máquinas inactivas.

Servicio de Seguridad

Tipología de los ataques:

- Privacidad y confidencialidad.
- Autenticación (*spoofing*).
- Denegación de servicio.

Modelos y herramientas de seguridad:

- Cifrado: clave pública (RSA) y privada (DES).
- Protocolos de seguridad: IPsec, SSL.
- Certificados y firmas digitales: X.509.
- Elementos de seguridad: Firewalls.

Entornos de seguridad: p. ej. Kerberos.

Sistemas Operativos Distribuidos (SOD)

Definición: *Un sistema operativo distribuido ejecuta sobre un sistema distribuido haciendo creer a los usuarios que se trata de un sistema centralizado*

– *single system view* o uniprocesador virtual

Esconde el carácter distribuido del sistema:

– No hay acuerdo general si esto es siempre adecuado

Es fácil de decir pero no de hacer

– Cada sistema alcanza hasta cierto punto esta meta

Los fracasos pueden generar frustraciones en los usuarios:

– "*Un sistema distribuido es aquél en el que no puedes trabajar con tu máquina por el fallo de otra máquina que ni siquiera sabías que existía*" (Lamport)

Clasificación de los Sistemas Operativos

• Sistemas operativos para multiprocesadores con memoria compartida (SMP):

– Software **fuertemente** acoplado

– sobre Hardware **fuertemente** acoplado

• Sistema operativo de red:

– Software **débilmente** acoplado

– sobre Hardware **débilmente** acoplado

• Sistema operativo distribuido (SOD):

– Software **fuertemente** acoplado

– sobre Hardware **débilmente** acoplado

Sistemas Operativos para SMPs

Arquitecturas de varios procesadores con memoria compartida de acceso uniforme Características:

- *"Ligeras"* variaciones sobre versiones tradicionales.
- Sólo hay una copia del sistema operativo.
- Concurrencia se traduce en paralelismo real.
- Comercialmente aceptados (Linux, WinNT, Solaris, AIX, ...).
- Plantea retos para: la ejecución del núcleo en varios procesadores (llamadas al sistema concurrentes) , los mecanismos de sincronización (*spin-locks)*, optimización y planificación (*afinidad al procesador)*, ...

Sistemas Operativos de Red

Definición: [Cho97]
Red de computadoras débilmente acopladas en las que no existe un control externo directo sobre el hardware/software de cada computadora para la compartición de recursos.

Características:

- No dan la visión de uniprocesador virtual (máquinas independientes).
- Cada una ejecuta una copia de sistema operativo (posiblemente distinto).
- Sistema operativo convencional + utilidades de red.
- Protocolos de comunicación para intercambio de recursos y acceso a servicios de alto nivel.
- Desde *rcp*/*rlogin* hasta *Open Network Computing* (ONC) de Sun.

Sistemas Operativos Distribuidos (SOD)

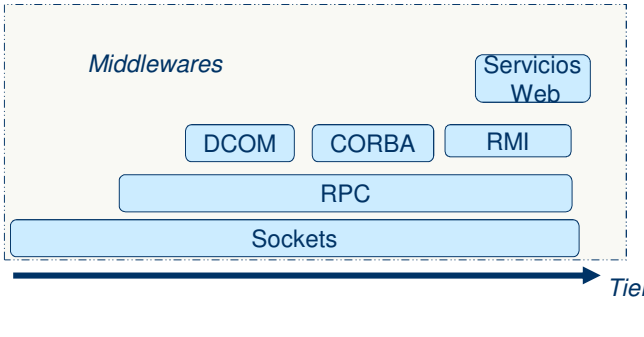
- Una copia del SO en cada procesador
- Necesidad de desarrollar nuevos conceptos
- Algunos ejemplos de esta problemática específica:
  - ¿Cómo lograr exclusión mutua sin memoria compartida?
  - ¿Cómo tratar los interbloqueos sin un estado global?
  - Planificación de procesos: Cada copia del sistema operativo tiene su cola de planificación (migración de procesos).
  - ¿Cómo crear un árbol de archivos único?
  - Implicaciones de la falta de reloj único, la presencia de fallos o la heterogeneidad.

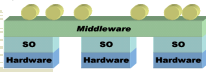
Evolución de los SOD

- Primeros SO de red:
  - Incluir servicios de red en SO convencional
  - Ejemplo: UNIX 4BSD (≈1980)
- Paulatina incorporación de más funcionalidad:
  - ONC de Sun (≈ 1985): incluye NFS, RPC, NIS
- Primeros SOD:
  - Nuevos SO pero basados en arquitecturas monolíticas
  - Ejemplo: Sprite de la Universidad de Berkeley (≈ 1988)
- SOD basados en microkernel. Ejemplos:
  - Mach de CMU (≈ 1986)
  - Amoeba diseñado por Tanenbaum (≈ 1984)
  - Chorus de INRIA en Francia (≈ 1988)
- Tendencia actual: Entornos distribuidos ---> *Middleware*

Middleware

Middleware:
– Capa de software que ejecuta sobre el sistema operativo local ofreciendo unos servicios distribuidos estandarizados.
– Sistema abierto independiente del fabricante.
– No depende del hardware y sistema operativo subyacente.
Ejemplos:





SD: Retos

- 1. Heterogeneidad de:
  - **Infraestructura** de la red **subyacente**,
  - Computadoras **hardware** y **software** (ej. Sistemas operativos, comparar sockets UNIX y llamadas Winsock),
  - **Lenguajes** de programación (en particular, representación de datos).
  - Algunas **metodologías**
    - Middleware (ej. CORBA): trasparencia de red, heterogeneidad de hardware y software y lenguajes de programación.
    - Código móvil (ej. JAVA): transparencia desde el hardware, software y heterogeneidad de lenguajes de programación mediante el concepto de máquina virtual.
- 2. Apertura
  - Asegura la extensibilidad y mantenibilidad del sistema
    - Adherencia a interfaces estándar
- 3. Seguridad
  - Privacidad
  - Autentificación
  - Disponibilidad
  - Etc.

SD: Retos

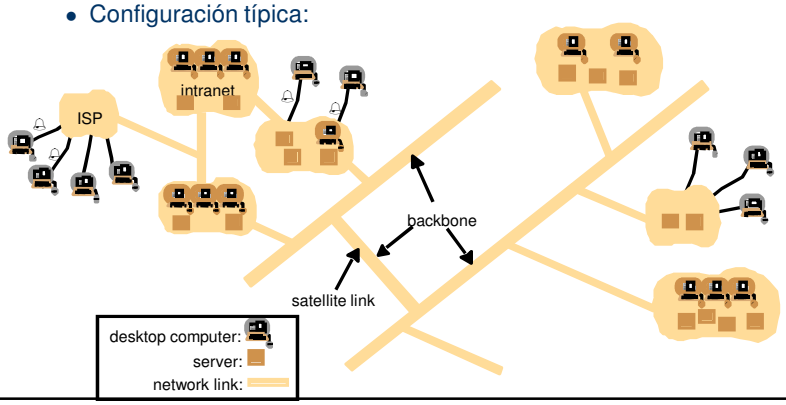
- 5. Manejo de fallas
  - Detección (puede ser imposible)
  - Enmascarar
    - Retrasmisión
    - Redundancia en almacenamiento de datos
  - Tolerancia
    - Manejo de excepciones (ej. Pausas en esperas de respuestas de la web)
  - Redundancia
    - encaminadores redundantes en la red
    - Replicación de tablas de nombres en múltiples dominios de servidores de nombres
- 6. Concurrencia
  - Planificación consistente de hilos concurrentes (con lo que se mantiene la dependencia, ej. En transacciones concurrentes)
  - Se evitan problemas de deadlocks y livelocks.

SD: Retos

- 7. Transparencia: ocultamiento de la heterogeneidad y distribución natural de los sistemas, lo cual aparece ante el usuario como un solo sistema.
  - Clasificación de la trasparencia (de acuerdo al ISO)
    - Acceso: a recursos locales y remotos utilizando las mismas operaciones
    - Localización: acceso sin conocer la ubicación de los recursos
      - Ej. Direcciones URLs, e-mails.
    - Concurrencia: permite a varios procesos operar concurrentemente utilizando recursos compartidos de manera consistente
    - Replicación: utiliza recursos replicados como si fueran una sola instancia
    - Fallas: permite que los programas completen sus tareas a pesar de fallas ej. Retransmisión de e-mails
    - Movilidad: permite mover recursos
    - Desempeño: adopción de los sistemas para variar situaciones de carga sin que el usuario lo perciba
    - Escalamiento: permite que el sistema y las aplicaciones se expandan sin necesidad de cambiar estructuras o algoritmos.

Ejemplos

- 1. La Internet
  - Redes heterogéneas de computadoras y aplicaciones
  - Implementación mediante la pila de protocolos de Internet
  - Configuración típica:

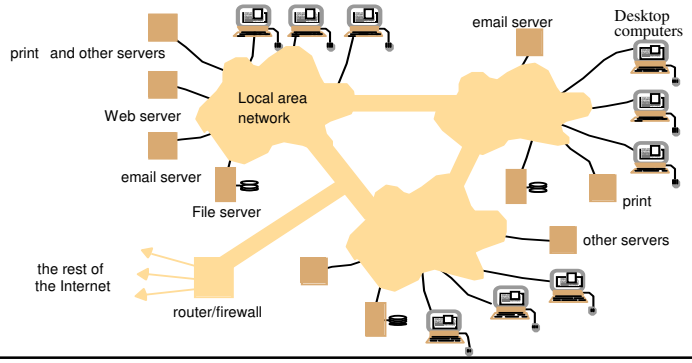


Ejemplos

- 2. Sistemas Multimedia Distribuidos
  - Frecuentemente utilizan la infraestructura de Internet
  - Características
    - Fuentes de datos y recipientes heterogéneos que necesitan ser sincronizados en tiempo real
      - Video
      - Audio
      - Texto
    - Frecuentemente: Servicios distribuidos
      - Multidifusión
    - Ejemplos
      - Herramientas de tele-educación
      - Video conferencias
      - Video y audio en demanda

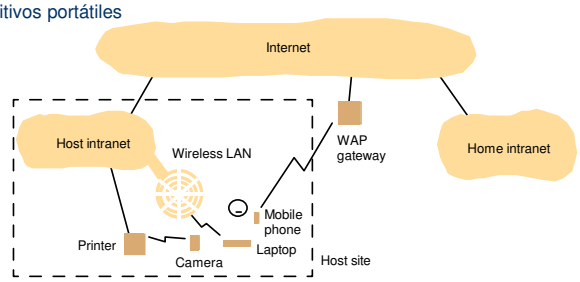
Ejemplos

- 3. Intranets
  - Redes localmente administradas
  - Generalmente propietarias (ej. Red del campus universitario)
  - Interfaces con la Internet
    - Cortafuegos
  - Proporciona servicios interna y externamente.



Ejemplos

- 4. Sistemas de cómputo móvil
  - Sistemas de telefonía celular (ej. GSM, UMTS, LTE)
    - Recursos que se comparten
      - Radio frecuencias
      - Tiempos de transmisión en una frecuencia (UMTS: multiplexado)
      - El móvil en movimiento
    - Computadoras portátiles
      - LANs inalámbricas
      - Dispositivos smartphones, tablets, handheld, PDAs, etc.
      - Dispositivos portátiles



Ejemplos

- 5. Sistemas embebidos
  - Una cafetera enlazada a la red
  - Sistemas de control de aviación
    - Sistemas de control de vuelo en aeropuertos
  - Sistemas de control de automóviles
    - Automóviles Mercedes S-Klasse equipados con 50+ procesadores embebidos autónomos
    - Conectados mediante canales Lan propietarios
  - Electrodomésticos
  - Equipos de audio HiFi

Ejemplos

- 6. Sistemas de telefonía
  - Ejemplos
    - POTS
    - ISDN
    - Redes inteligentes
    - Redes inteligentes avanzadas
  - Compartición de recursos
    - Redes
    - Administración
    - Teléfonos
- 7. Administración de la red
  - Administración de recurso de la red
  - Estado: estatus de los recursos y conexiones
  - Ejemplo
    - SNMP

Ejemplos

- 8. Sistemas de Archivos en Red
  - Arquitectura para acceder a sistemas de archivos a través de la red
  - Ejemplos conocidos
    - Network File System (NFS), originalmente desarrollado por SUN Microsystems para soportar acceso remoto en un contexto UNIX
- 9. El World Wide Web
  - Arquitectura cliente/servidor abierta implementada sobre la Internet
  - Compartición de recursos
    - Información, identificada únicamente mediante Localizador Uniforme de Recursos (URL)
    - Variantes: Webs basadas en intranets.

Ejemplos

10. Cluster

- Tipo de sistema distribuido muy popular
- Dedicado a una tarea específica
  - Computación paralela
  - Servicios escalables y de alta disponibilidad
- Ejemplo: Google usa un clúster con 6000 procesadores
- Sistema homogéneo basado en componentes estándar
- Gestión de procesos más coordinada que en SD general
- Seguridad sólo requerida si está expuesto al “exterior”

Ejemplos

10. Sistemas P2P

- Sistemas cuyo rol es similar
- Gnutella, Napster, Skype

11. Cloud Computing

- Basado en Virtualización y SOA
- Se paga por lo que se consume
- Públicos, Privados e Híbridos
- IaaS, PaaS. SaaS