

Universidad Simón Bolívar

Departamento de Computación y T.I

Sistemas de operación III

CI-4822

Computación Ubicua

Una nueva técnica de computación distribuida

Prof. Yudith Cardinale

Sep - Dic 2011

Contenido

- ◆ Conceptos de GRIDS. Qué ofrecen?
- ◆ Desafíos
- ◆ Razones que motivaron la creación de GRIDS
- ◆ Posibles aplicaciones
- ◆ Tipos de usuarios y su descripción
- ◆ Arquitectura
- ◆ Peer to peer computing
- ◆ Ejemplos

Computación Ubicua

- ◆ ¿Dónde aplica el concepto de ubicuidad?
 - ◆ Se ofrece la misma imagen o entorno desde cualquiera de los lugares de trabajo (la misma interfaz “está en todas partes”)
 - ◆ Los procesos se ejecutan en alguna plataforma elegida por el sistema (el hardware “está en todas partes”)
- ◆ ¿Se puede hablar de subclases de Sistemas Distribuidos? Proponemos dos subclases, **Grids** **Computacionales** y **Sistemas P2P**:
 - ◆ No necesariamente disjuntas
 - ◆ Sistemas conformados por sistemas

Computación Ubicua

- ◆ **Grids Computacionales:** se presupone que hay control sobre el acceso a los servidores.
 - ◆ Rendimiento
 - ◆ Transparencia
 - ◆ Tolerancia a fallas
 - ◆ Seguridad
- ◆ **Sistemas P2P:** se presupone que los nodos participantes no tienen control sobre el acceso a los mismos.
 - ◆ Cooperación
 - ◆ Independencia de localidad
 - ◆ Rodeo de controles
 - ◆ Anonimato

Grids – Teragrid Backplane

NSF TeraGrid Extensible Terascale Facility



Grid Computacional: qué es?

- ◆ Otros términos usados: Metasistemas, Metacomputación, Sistemas de Metacomputación
- ◆ Infraestructura de hardware y software que provee alto desempeño y alta disponibilidad

Grid Computacional: qué es?

- ◆ Ambientes que integran, a través de redes de alta velocidad recursos heterogéneos complejos tales como: supercomputadores, grandes bases de datos, etc.
- ◆ Mecanismo para que los usuarios puedan usar recursos distribuidos geográficamente de forma transparente, creando la ilusión de un sistema de computación integrado. Esta ilusión la provee el middleware.

Grid Computacional: qué es?

- ◆ Los GRIDS hacen posible el compartir recursos, en forma coordinada, dentro de organizaciones dinámicas e interinstitucionales (organizaciones virtuales).
- ◆ La infraestructura debe proveer alto **desempeño y alta disponibilidad**

Grids y SD

- ◆ Como un SD los grids computacionales deben integrar recursos de capacidades variadas, conectados por redes no confiables, localizadas en dominios administrativos distintos. No obstante, la necesidad de **alto rendimiento** puede requerir de modelos de programación e interfaces radicalmente diferentes.

GRIDS y Organizaciones Virtuales

- La tecnología de GRID y en particular las organizaciones virtuales (VO) cambiaron radicalmente la forma cómo se resuelven los problemas.
- Las VO hacen posible la existencia de **grupos diversos de organizaciones y/o individuos** para **compartir recursos en forma controlada**, de forma tal que los miembros puedan **colaborar para lograr una meta común**.

GRIDS y Organizaciones Virtuales

- El compartir recursos es condicional: cada propietario coloca sus recursos a la disposición de otros, sujeto a restricciones sobre dónde, cuándo y qué se puede hacer.
- El usuario también puede imponer restricciones sobre los recursos que desea.

GRIDS y Organizaciones Virtuales

- El compartir puede combinarse y se pueden estar usando a la vez varios recursos pertenecientes a distintas organizaciones.
- El mismo recurso puede estarse usando de distintas formas por distintas organizaciones virtuales.

Grids Computacionales: qué ofrecen?

- ◆ Colaboración más efectiva, agrupando co-investigadores en el mismo espacio virtual.
- ◆ Incremento de la capacidad de cómputo local.
- ◆ Productividad mejorada a través de un ambiente de programación considerablemente más simple.

Grids Computacionales: qué ofrecen?

- ◆ Ahorro, los recursos en cuestión pueden ser muy costosos.
- ◆ Espacio de objetos (Archivos) persistentes compartido.
- ◆ Ejecución remota transparente.
- ◆ Meta-aplicaciones

Grids Computacionales: aspectos que los caracterizan

- **Múltiples dominios administrativos y autonomía:** los recursos de un Grid están geográficamente dispersos entre múltiples dominios administrativos y pertenecen a diferentes organizaciones. Es importante respetar la autonomía de los propietarios de los recursos en lo que respecta a la gestión de estos últimos y sus políticas de uso.
- **Heterogeneidad:** Un Grid involucra una gran variedad de recursos que son heterogéneos por naturaleza y abarcan un amplio rango de tecnologías.

Grids Computacionales: aspectos que los caracterizan

- **Escalabilidad:** Un Grid debe crecer en un rango que va desde pocos recursos, hasta millones. Este crecimiento provoca problemas de desempeño. Como resultado, las aplicaciones que necesitan de un gran número de recursos deben diseñarse para tolerar posibles problemas de latencia y ancho de banda.
- **Dinamismo:** Las características de los recursos (carga, prioridad, disponibilidad, etc.) pueden cambiar en el tiempo. Tanto los administradores de recursos como las aplicaciones, deben ir al paso de este comportamiento dinámico y usar los recursos y servicios disponibles de una forma eficiente y efectiva.

Razones que motivaron su creación

- ◆ Los datos presentan un gran desafío:
 - ◆ Los detectores del Laboratorio Europeo de Partículas Físicas, producirían para el año 2005 varios *petabytes* de datos por año, un millón de veces la capacidad de un computador de escritorio promedio.
 - ◆ Analizar estos datos pudiera requerir de alrededor de 20 teraflops/seg de poder computacional. Para el 2004 el supercomputador más rápido ofrecía 280600 Gflops. Hoy en día 1026000 Tflops

Razones que motivaron su creación

◆ Supercomputadores

◆ Clusters

- ◆ Surgen en 1980
- ◆ Los más rápidos, en el año 2000, fueron 8000 procesadores RS/6000 (ASCI White System) en el laboratorio nacional Lawrence Livermore.
- ◆ Ofrecen una mejora considerable en poder de cómputo, pero es un conjunto de computadores dedicados, ubicados en un solo lugar.
- ◆ Existen razones financieras y técnicas que imponen límites a cuán grandes deben ser estos sistemas.

Razones que motivaron su creación

◆ Clusters

- ◆ En el 2000, el sistema ASCI White costó 110 millones de dolares y necesitó de un nuevo edificio (512 servidores RS 6000 IBM).

◆ Computación en Internet

- ◆ Existen alrededor de 400 millones de PC's en el mundo, muchas tan poderosas como los supercomputadores de los 90.
- ◆ La mayoría de estos PC's tienen mucho tiempo ocioso.

Razones que motivaron su creación

◆ Computación en Internet

- ◆ La computación en Internet busca explotar los ciclos ociosos de workstations y PC's de modo de crear un sistema poderoso de computación distribuida.
- ◆ En 1985 Miron Livny mostró que la mayoría de las estaciones de trabajo permanecían ociosas y propuso el sistema **Condor** para utilizar ciclos libres de CPU. Es efectivo a pequeña escala: Universidades.

Razones que motivaron su creación

◆ Computación en Internet

- ◆ En 1997 apareció **Entropía** (Scott Kurowski). Usa ciclos ociosos de computadores, alrededor del mundo, para resolver problemas de interés científico. En 2 años creció hasta llegar a 30000 computadoras que proveen una rapidez de cómputo por encima de un teraflop/seg. Usando este sistema se identificó el número primo más grande hasta ahora conocido.

Razones que motivaron su creación

◆ Computación en Internet

- ◆ El próximo paso lo representa David Anderson con su proyecto SETI@home. Los ciclos ociosos se utilizan para analizar datos del telescopio de Arecibo buscando signos que indiquen la presencia de inteligencia extraterrestre. Abarca alrededor del millón de PC's.

Razones que motivaron su creación

- ◆ **Computational GRIDS** [Foster y Kesselman, 1998]:
 - ◆ La computación en Internet es sólo un caso especial de algo mucho más poderoso: la necesidad de distintas organizaciones (ambito científico) que buscan metas comunes de compartir recursos.
 - ◆ El web y el correo electrónico ofrecen mecanismos básicos que permiten a estos grupos trabajar juntos.
 - ◆ Lo ideal es agrupar datos, computadores, sensores y otros recursos en un único laboratorio virtual.

Razones que motivaron su creación

◆ Computational GRIDS

- ◆ La tecnología GRID persigue este propósito, ofreciendo protocolos, servicios y kits para el desarrollo de software, necesarios para crear un ambiente que permita compartir recursos a gran escala.
- ◆ Los primeros conceptos se exploraron en 1995 (experimento I-WAY): se usaron redes de alta velocidad para conectar 17 sites en Norteamérica.
- ◆ Hoy en día existen muchos proyectos de Investigación encargados de desarrollar las tecnologías para crear GRIDS en varias comunidades y disciplinas científicas.

Aplicaciones

- ◆ *Supercomputación distribuida*: usan las GRIDS para agrupar recursos que permitan resolver problemas que no pueden ser resueltos por un solo computador. Ejemplos: simulación interactiva distribuida (entrenamiento y planificación militar), simulación de procesos físicos complejos (cosmología, modelamiento del clima, etc.)

Aplicaciones

- ◆ *High-Throughput computing*: la idea es planificar un gran número de tareas independientes o débilmente acopladas (poca sincronización) para aprovechar ciclos de CPUs ociosos. La naturaleza independiente de las tareas conduce a diferentes métodos para resolver el problema. Ejem: Problemas de **criptografía**, se uso durante la etapa crucial en el diseño de los procesadores **AMD K6 y K7, CONDOR**.

Aplicaciones

- ♦ *Computación por demanda*: satisfacen requerimientos de recursos a corto plazo. Se trata de recursos a los que no se tiene acceso de manera local. Entre los recursos tenemos: BD, sensores, software, cómputo. Ejem: **NEOS** y **NetSolve**, envían a servidores remotos cálculos que necesitan mucho poder de cómputo o un software especializado.

Aplicaciones

- ♦ *Computación con un manejo intensivo de datos*: sintetizan información a partir de datos en repositorios distribuidos, librerías digitales y bases de datos. El proceso de síntesis es intensivo desde el punto de vista computacional y comunicacional.

Ejem: Experimentos Físicos, Fotografías Astronómicas.

Aplicaciones

- ♦ *Computación colaborativa*: permite interacción hombre-hombre, generalmente las aplicaciones están estructuradas en términos de un espacio virtual compartido.

Usuarios

Usuarios finales

Desarrolladores de aplicaciones

Desarrolladores de herramientas

Desarrolladores del metasisistema

Administradores del sistema

Usuarios

◆ Desarrolladores:

- ◆ Propósito: diseñar e implementar los servicios básicos (protocolo del metasisistema).
- ◆ Usan los servicios de los sistemas locales.
- ◆ Se preocupan por la simplicidad local, la conectividad y la seguridad.

Usuarios

- ◆ **Desarrolladores de herramientas:**
 - ◆ Propósito: implementar herramientas, compiladores, librerías, etc. que implementan los modelos de programación y servicios usados por los desarrolladores de aplicaciones.
 - ◆ Usan los servicios del metasisistema
 - ◆ Se preocupan por la adaptabilidad, el desempeño y la seguridad

Usuarios

- ◆ **Desarrolladores de herramientas:**
 - ◆ Ejem: Netsolve, MPI
 - ◆ Los desarrolladores de herramientas deben usar los servicios básicos para programar implementaciones eficientes de modelos de programación que utilizarán los programadores de aplicaciones.

Usuarios

◆ Desarrolladores de herramientas:

- ◆ Modelos de programación secuenciales: proveen abstracciones como subrutinas
- ◆ Modelos de programación paralelo: threads, variables de condición, pase de mensajes, etc.
- ◆ No hay consenso sobre cuál sería el modelo de programación apropiado en GRID: Shared memory, message passing, RPC, orientación por objeto, agentes.

Usuarios

- ◆ **Desarrolladores de aplicaciones:**
 - ◆ Propósito: desarrollar aplicaciones.
 - ◆ Usan los modelos de programación (tanto secuencial como paralelo) y las herramientas
 - ◆ Se preocupan por la facilidad de uso y el desempeño.

Usuarios

◆ Usuarios finales:

- ◆ Propósito: resolver problemas
- ◆ Usan las aplicaciones (paquetes), que a su vez usan los recursos y servicios del GRID.
- ◆ Les importa la transparencia y el desempeño.

Usuarios

- ◆ **Administradores del sistema:**
 - ◆ Propósito: administrar los recursos del metasisistema (administración distribuida)
 - ◆ Entre los problemas que encuentran se tienen: manejar comunidades de usuarios, lograr un equilibrio entre las políticas locales y las necesidades del metasisistema.

Usuarios

- ◆ **Administradores del sistema:**
 - ◆ Usan herramientas de administración.
 - ◆ Surgen nuevas actividades tales como: monitoreo; establecer políticas en situaciones donde el conjunto de usuarios puede ser muy largo variando en forma dinámica; negociar políticas con otros sites y usuarios, mecanismos de accounting y pago, etc.

Tipos de Grid

**Grids de
cómputo**

**Acceso
compartido a
sistemas de
computación
de alto rendimiento**

Grids de servicio

**Grids
de
datos**

**Acceso compartido
a bases de datos
y sistemas de
archivos**

**Acceso compartido
a software y otros
recursos
computacionales**

Peer-to-Peer computing: Cómputo intensivo descentralizado

- Sistemas y aplicaciones que emplean recursos distribuidos para realizar funciones críticas de manera descentralizada
 - ◆ Cómputo distribuido
 - ◆ Almacenamiento y compartimiento de datos (intercambio de archivos)
 - ◆ Comunicación y colaboración para crear software
 - ◆ Conversación directa en línea
 - ◆ Servicios en plataformas heterogéneas

Peer-to-Peer computing: Se presupone que los nodos participantes no tienen control sobre el acceso a los mismos

- Qué ofrecen:
 - ◆ Cooperación
 - ◆ Independencia de localidad
 - ◆ Rodeo de controles
 - ◆ Anonimato

Peer-to-Peer computing: Ejemplos de sistemas

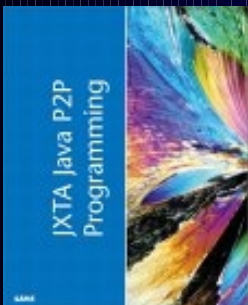
Sistemas para cómputo distribuido:



Proyecto académico para búsqueda de señales provenientes de otros mundos

MAGI

Producto para aplicaciones de negocios colaborativos (comercio basado en p2p) e incorpora múltiples tipos de dispositivos entre peers (PDAs, celulares, chips especializados, comunicación basada en eventos)



Proyecto de software abierto que provee servicios para p2p: encontrar peers, compartir archivos, encontrar contenido en sitios remotos, crear un grupo de peers, monitorear actividades y comunicación segura.

Peer-to-Peer computing: Ejemplos de sistemas

Sistemas para comunicación:

groove



Provee espacios virtuales compartidos para la interacción de pequeños grupos. Los usuarios que comparten un espacio pueden “chatear”, comunicarse por voz, enviar mensajes instantáneos, actualizar un calendario o plan de eventos, compartir un archivo, etc.

Sistemas para compartir información:

- ◆ Intercambio de música mp3. Creció considerablemente a punto de que fueron demandados por compañías editoras de discos... y perdieron.



- ◆ Servidor centralizado para mantenimiento de información de ubicación

El intercambio de información se hace directamente entre clientes

Peer-to-Peer computing: Ejemplos de sistemas

● Sistemas para compartir información: (cont.)



Intercambio de archivos, principalmente mp3. Incluyeron secretamente un módulo de cómputo, hecho por Brilliant Digital para hacer cómputo distribuido.

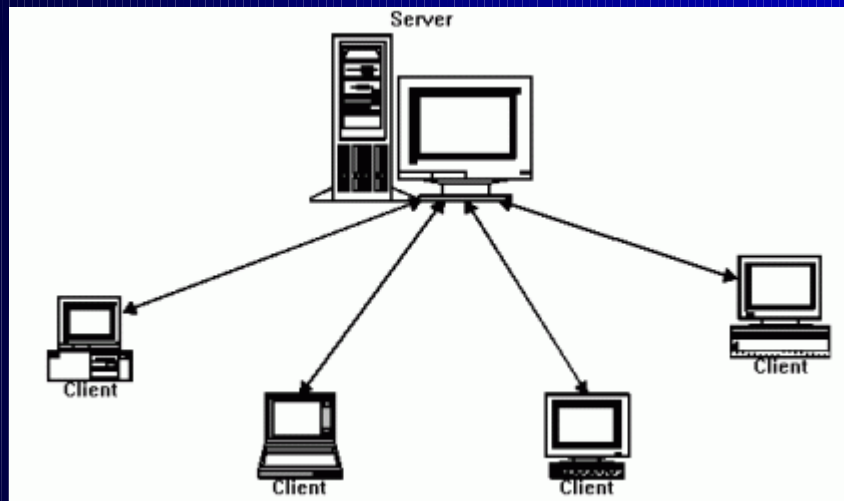


gnutella.com

- ◆ Compartimiento, búsqueda y copiado de archivos entre usuarios en Internet en forma descentralizada
- ◆ Los clientes son servidores al mismo tiempo.
- ◆ Aprende sobre los nodos conectados al vecino
- ◆ Descubrimiento de la red (ping y pong)

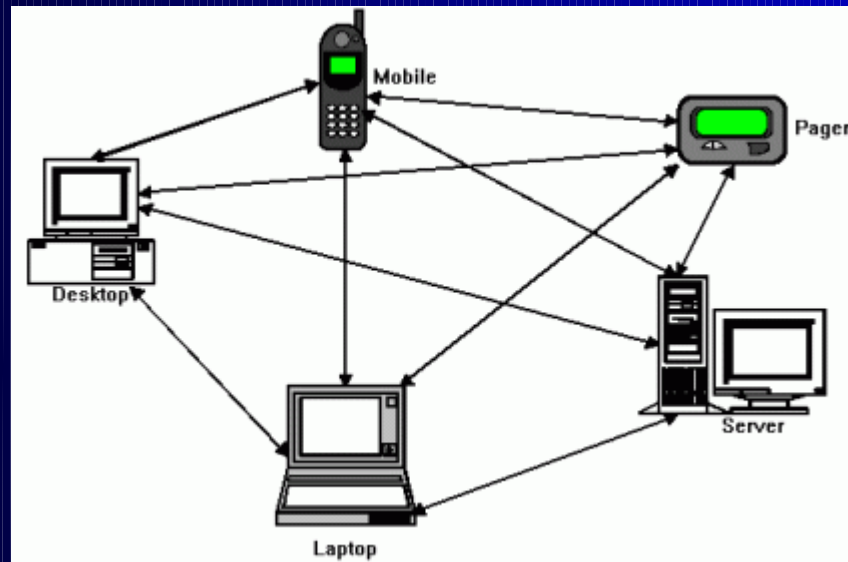
Modelo Cliente/Servidor

- Los clientes se comunican con el servidor
- Pueden comunicarse entre sí *a través* del servidor



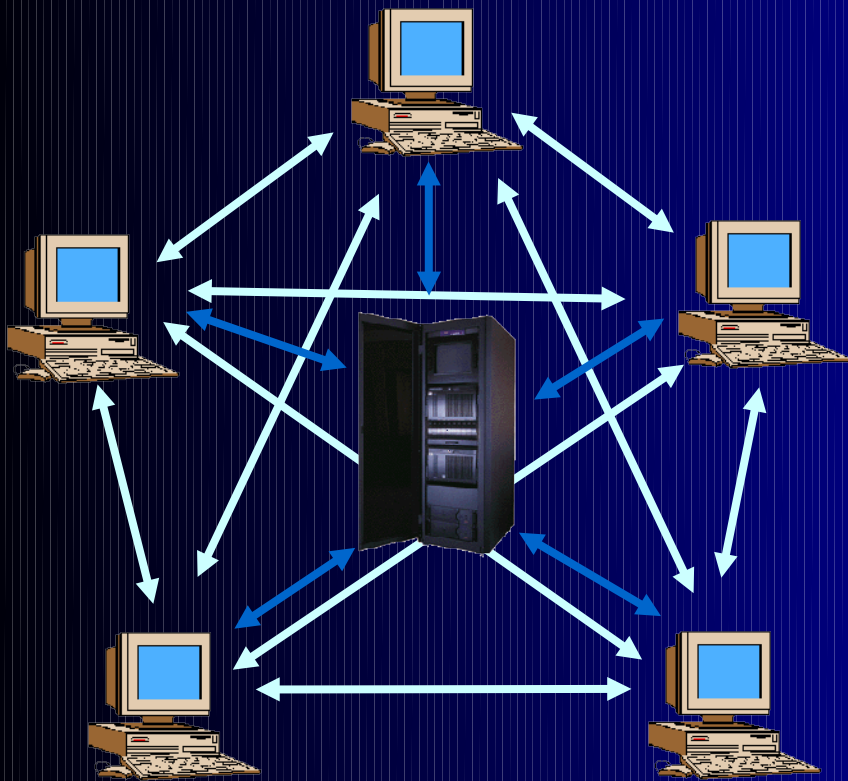
Modelo P2P (punto-a-punto)

- Los clientes se comunican directamente
- Pueden consultar previamente a un servidor con propósitos de localización

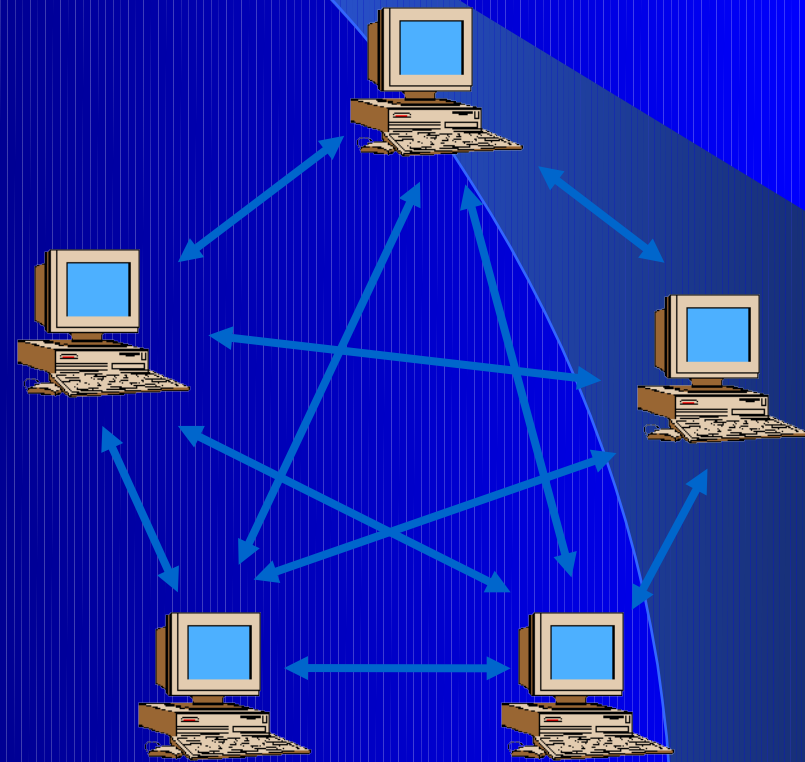


Variantes de P2P

Control central (ej. Napster)



Distribuido (ej. Gnutella)



Espacio de Aplicaciones P2P

- ◆ Las aplicaciones P2P se pueden definir en tres ejes principales:
 - **Descentralización**: puede haber un control centralizado (seti) o ser completamente independiente de un servidor central (freenet)
 - **Cooperación**: puede no haber comunicación entre pares (seti), basarse en la cooperación continua entre pares (freenet) o un esquema intermedio (napster)
 - **Indirección**: puede haber cooperación P2P directamente entre máquinas cliente (napster) o indirectamente, entre servidores (almacenamiento distribuido)

Consideraciones

◆ Interoperabilidad

- ◆ Protocolos
- ◆ Formato de datos

◆ Seguridad

- ◆ Confidencialidad
- ◆ Presencia de cortafuegos, NAT, etc.

◆ Descentralización

- ◆ Independencia del DNS
- ◆ Compartir recursos sin intermediación

◆ Identificación

- ◆ Nombres para los “peers”

Interoperabilidad: protocolos

- Hay muchos de ellos, ejemplos:
 - **HTTP** (Hypertext Transfer Protocol): protocolo para solicitud de información, usado en el Web
 - **SOAP** (Simple Object Access Protocol): protocolo para invocar código usando XML sobre HTTP
 - **GIOP** (General Inter ORB Protocol): protocolo utilizado en CORBA para comunicar ORBs
 - **RTP** (Real-Time Transport Protocol): protocolo para intercambio de datos a tiempo real, como video y audio
 - **Gnutella Protocol**: descubrimiento usado por Gnutella

Interoperabilidad: datos

- ♦ **HTML**: conjunto de etiquetas y reglas para definir documentos de hipertexto
- ♦ **XML**: conjunto de reglas para definir formatos de datos estructurados
- ♦ **Java bytecode**: se pueden ejecutar applets en casi cualquier plataforma
- ♦ **¿DOC?**: estamos hasta la coronilla de que nos envíen documentos en formato DOC

Seguridad: autenticación y confidencialidad

- **SSL** (Secure Socket Layer): protocolo para establecer conexiones de sockets seguras
- **HTTPS**: simplemente HTTP sobre SSL
- **PKI** (Public Key Infrastructure): sistema de cifrado de clave pública que se apoya en certificados digitales
- **SSH** (Secure Shell): programa para utilizar un shell en un computador remoto y para hacer túneles seguros

Seguridad: cortafuegos y NAT

- El modelo cliente-servidor ha dado origen a mecanismos que permiten la conexión “hacia afuera” a todos pero limitan la conexión “hacia adentro”
- NAT
 - ♦ Se origina por la carencia de direcciones IP
 - ♦ Impide el acceso a máquinas internas, a menos que haya NAT hacia adentro
- Cortafuegos
 - ♦ Se origina por la necesidad de protegerse de ataques
 - ♦ Intencionalmente se evita el acceso a máquinas internas, con excepción de ciertos servicios
- Se necesitan mecanismos de conexión a servicios que utilicen la conexión TCP hacia afuera

Descentralización: independencia del DNS

- Los servicios de acceso a Internet (por discado o por conexión de banda ancha) en general no incluyen un nombre DNS
- De hecho, los servicios de acceso a internet suelen asignar números IP que cambian continuamente
- Hay servicios de DNS dinámico (solución parcial)
- Se necesitan mecanismos de identificación (*naming*) alternativos, que no se basen en la asociación DNS-IP
- Para asociar un nombre con un destino (IP, Puerto) hace falta
 - Servidor de nombres
 - Mecanismo de descubrimiento

Descentralización: participar sin intermediación

- No necesariamente se cuenta con servidores con DNS oficial
- Posibilidad: establecer redes que se tejen extendiendo las conexiones con vecinos
- Se necesitan protocolos de descubrimiento, tanto de nodos como de otros objetos (archivos)
- Se puede tejer una red de servidores de nombres alternativa al DNS