DEPARTAMENTO DE

INGENIERÍA INFORMÁTICA

UNIVERSIDAD DE SANTIAGO DE CHILE

DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

ARQUITECTURAS DISTRIBUIDAS

13169

SISTEMAS DISTRIBUIDOS

Estilos Arquitectónicos

- Se formulan en términos de componentes, la forma en que los componentes interactúan entre sí, el intercambio de datos entre los componentes y en cómo se configuran juntos en un sistema.

- Un **componente** es una unidad modular con las interfaces requeridas bien definidas.

  - Es reemplazable dentro de su ambiente

- El **conector** un mecanismo que media la comunicación, coordinación o cooperación entre componentes.

Estilos Arquitectónicos

- Por medio de **componentes** y **conectores** se logran varias configuraciones, la cuales se clasifican en estilos arquitectónicos.

Los más importantes para SD

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

Arquitectura en Capas

- Componentes organizados a modo de capas.
- Capa de nivel L_i se le permite llamar a componentes de la capa subyacente en nivel L_{i-1}. No puede llamar a componentes de otras capas.
- Ampliamente utilizado en redes.
- Las peticiones se mueven hacia abajo en la jerarquía y los resultados se mueven hacia arriba.

Arquitectura Basada en Objetos

- Organización libre.
- Cada objeto es un componente.
- Se conectan a través de un mecanismo de llamadas a procedimientos remotos (RPC).
- Arquitectura cliente – servidor.

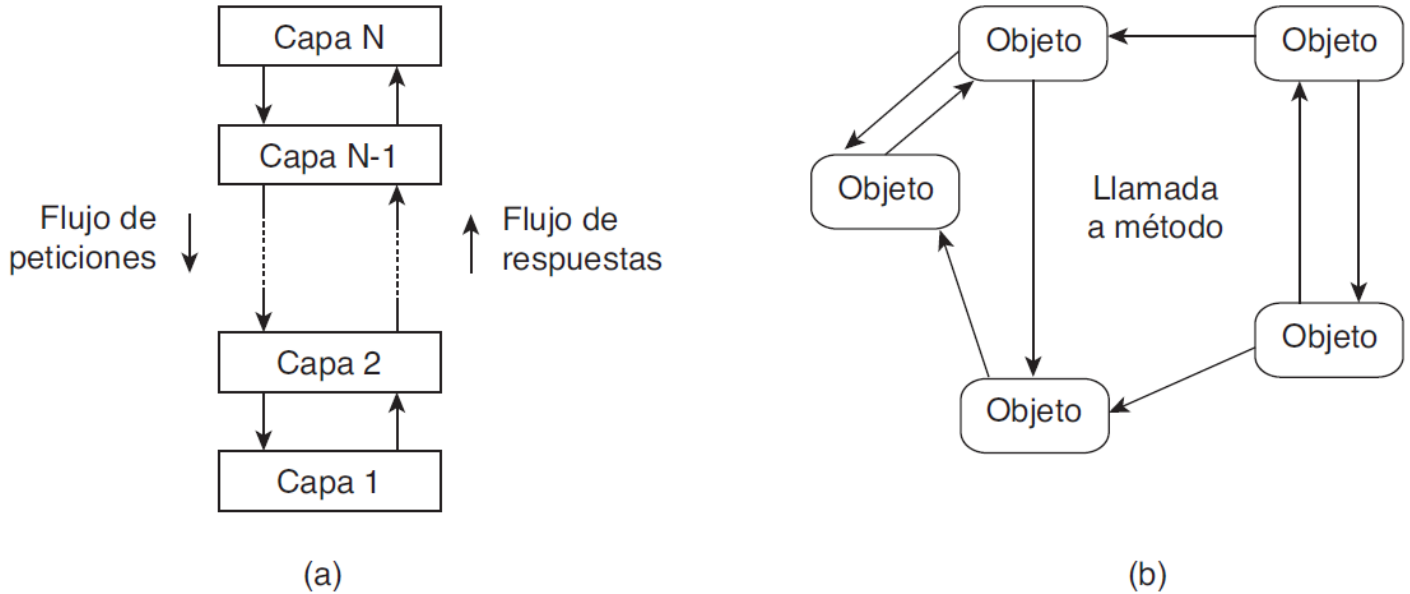


Figura 2-1. Estilos arquitectónicos (a) en capas, y (b) basado en objetos.

Arquitectura Centrada en Datos

- Procesos se comunican a través de un repositorio común de datos.
- Las aplicaciones en red se basan en un sistema de archivos distribuidos compartidos donde casi todas las comunicaciones se realizan a través de archivos.
- SD basados en la web : los procesos se comunican a través de servicios de datos compartidos basados en la web.
- Transforma el diseño del **centro de datos,** poniendo los datos como elemento central.

Arquitectura Basada en Eventos

- Comunicación generada a través de la propagación de eventos.
- **Sistemas de publicación – suscripción**.
- Sistema desacoplado, no es necesario que un proceso referencie a otro explícitamente.
- **Referencialmente desacoplado**.

Espacio de Datos Compartidos

- Combinación de arquitectura **basada en eventos** y arquitectura **centrada en datos**.
- Los procesos están desacoplados en el tiempo. **No es necesario que ambos estén activos para realizar la comunicación.**
- No necesariamente se utiliza una referencia explicita, como en el caso de los archivos.

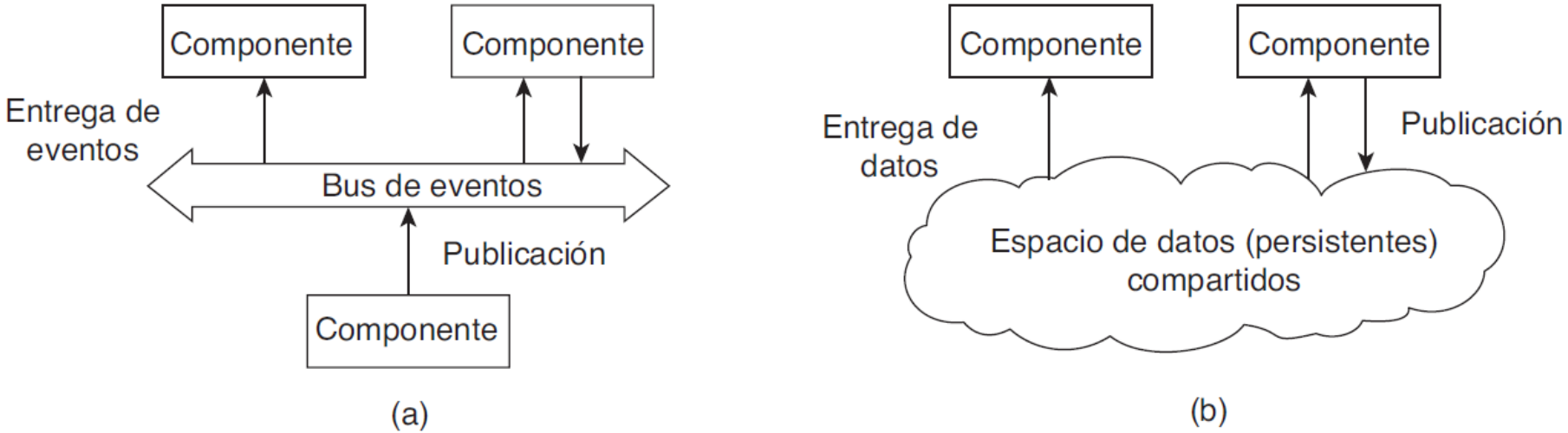


Figura 2-2. Estilo arquitectónico (a) basado en eventos, y (b) espacio de datos compartidos.

Arquitectura de Sistemas: Arquitectura centralizada

- ***Clientes*** que requieren servicios de los ***servidores.***

- Modelo básico de cliente-servidor.

- **Servidor:** Implementa un servicio específico.
- **Cliente:** Solicita un servicio a un servidor.

- Se envía una petición y se espera por la respuesta.

Arquitectura de Sistemas: Arquitectura centralizada

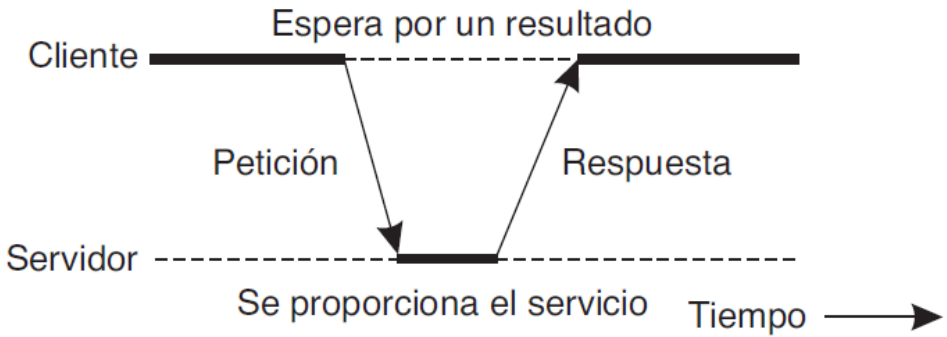


Figura 2-3. Interacción general entre un cliente y un servidor.

Arquitectura de Sistemas: Arquitectura centralizada

- La comunicación puede implementarse mediante un protocolo simple no orientado a conexión cuando la red es confiable.
  - Redes de área local.
- Este protocolo tiene la ventaja de ser eficiente, mientras los mensajes no se pierdan ni corrompan.
- ¿Qué hacer cuando no llega la respuesta?
  - Cliente reenvía la petición.
  - Problema → El cliente no sabe si el mensaje original se perdió o la transmisión de respuesta falló.

Arquitectura de Sistemas: Arquitectura centralizada

- Operación **idempotente** → Operación que puede repetirse sin ocasionar daño.
- Dado que hay operaciones idempotentes y otras que no, no hay una única solución al problema planteado.
- Alternativa: Utilizar un protocolo confiable orientado a conexión.
- Redes de área amplia.
- El cliente establece una conexión con el servidor para enviar la petición y recibir la respuesta.
- Problema → Establecer e interrumpir la conexión es costoso.

Arquitectura Centralizada: Aplicación de Capas

- Se distinguen 3 capas:
  1. El nivel de interfaz de usuario.
  2. El nivel de procesamiento.
  3. El nivel de datos.
- Nivel de usuario → Contiene todo para la administración visual e interacción del usuario.
- Nivel de procesamiento → Contiene las aplicaciones.
- Nivel de datos
  - Administra los datos reales sobre los que se trabaja.
  - Mantiene la **persistencia** entre diferentes aplicaciones.

Arquitectura Centralizada: Aplicación de Capas

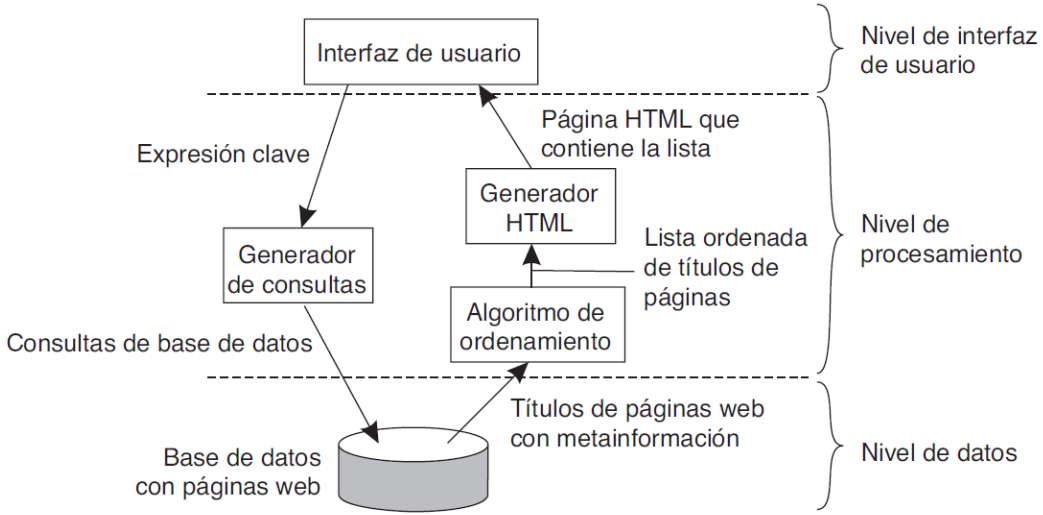


Figura 2-4. Organización simplificada, en tres capas diferentes, de un motor de búsqueda en internet.

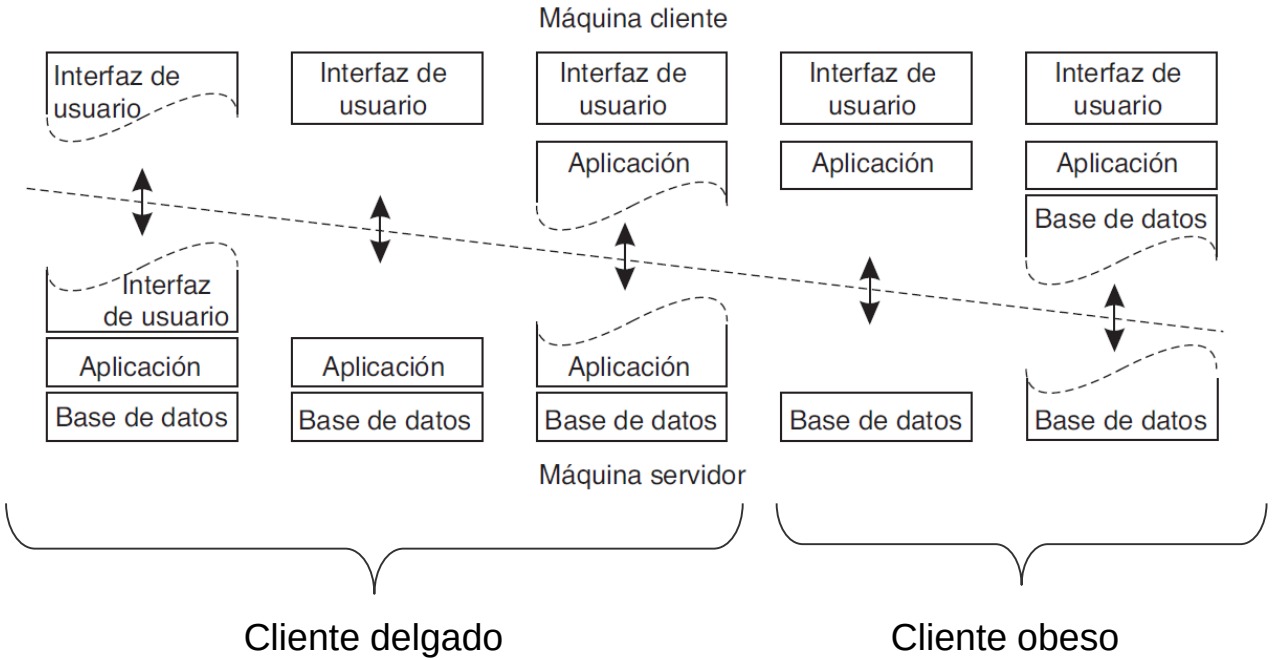
Arquitectura Centralizada: Multiniveles

- Los niveles lógicos proponen cierta cantidad de posibilidades para la distribución física de una aplicación cliente-servidor.
  - Cliente que contiene aplicaciones a nivel de interfaz.
  - Servidor contiene la implementación del procesamiento y los datos.

  } Lo más simple

- El servidor maneja todo y el cliente es básicamente un terminal simple.
- Es efectivo distribuir los programas en capas de aplicación a través de diferentes máquinas.

Arquitectura Centralizada: Multiniveles → Arquitectura de 2 Capas



Arquitectura Centralizada: Multiniveles

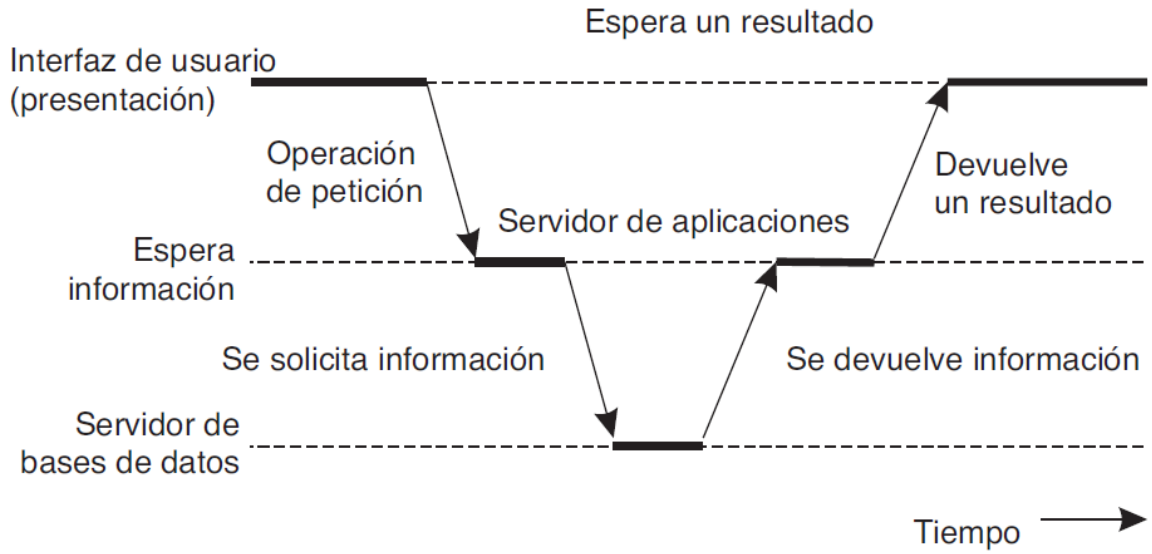


Figura 2-6. Ejemplo de un servidor que actúa como cliente.

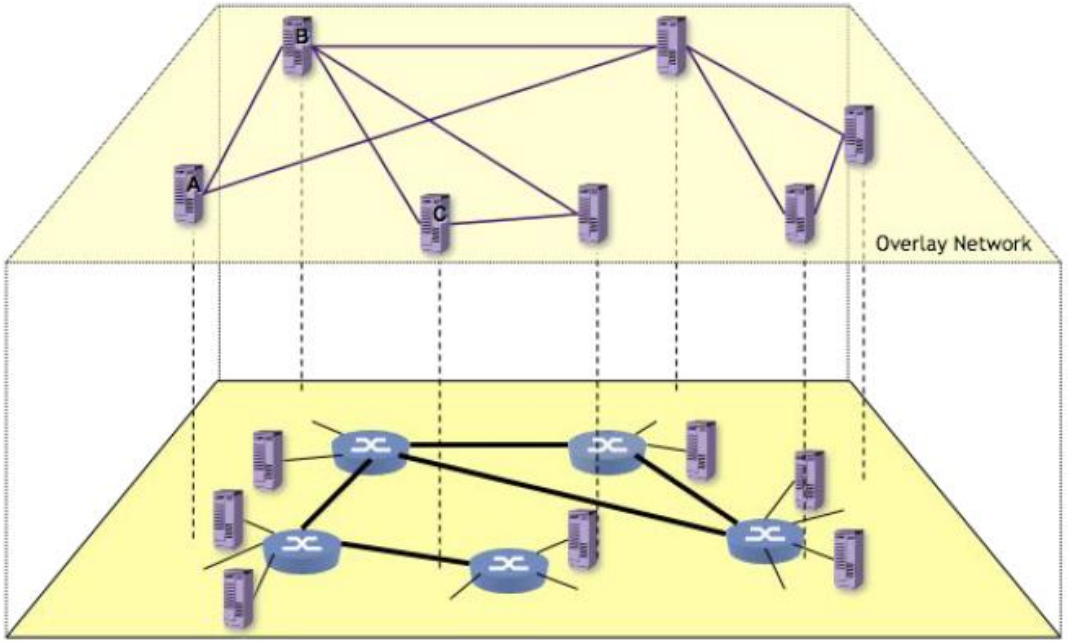
Arquitectura Descentralizada

- Aplicaciones cliente servidor son consecuencia directa de dividir aplicaciones.
- Los niveles corresponden a la organización lógica.
- **Distribución vertical**
  - Distribuir lógicamente los componentes de la aplicación.
- **Distribución horizontal**
  - Partes lógicas equivalentes, pero cada una con su propio conjunto de datos.

Arquitectura Descentralizada: Sistema Peer to Peer (P2P)

- Todos los procesos son iguales.
- **Interacción simétrica.**
  - Cada proceso actúa como cliente y como servidor al mismo tiempo.
- Esta interacción simétrica permite la evolución de la red, en cuanto a la organización de los procesos. A esto se le conoce como **red superpuesta**.
- Los procesos no se comunican directamente.
- Envían mensajes a través de los **canales de comunicación** disponibles.
- Los sistemas punto a punto se dividen en **estructurados** y **no estructurados**.

Arquitectura Descentralizada: Red Superpuesta (Overlay Network)



Arquitectura Descentralizada: Sistema P2P Estructurado

- La red superpuesta se construye a través de un proceso determinista.
- **Tabla Hach Distribuida (DHT).**
  - Nodos del sistema y elementos de dato se les asigna una llave.
  - Se busca mapear cada llave de elementos de datos hacia el identificador de un nodo, basándose en cierta distancia métrica.
- Por ejemplo, el sistema **Chord** y el sistema **Pastry**.

Sistema P2P Estructurado: Chord

- Organización lógica de los nodos en forma de anillo.
- Elementos de datos con llave *k* se mapea hacia el nodo con *id* más pequeño (*id* ≥ *k*).
- El nodo se conoce como sucesor de la llave *k*.

Sistema P2P Estructurado: Chord

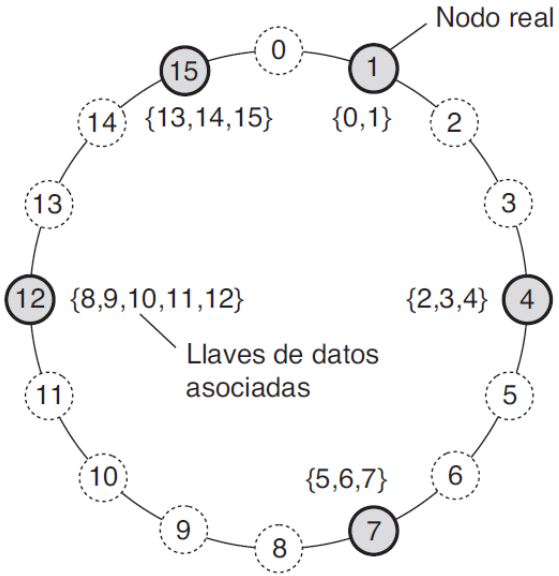
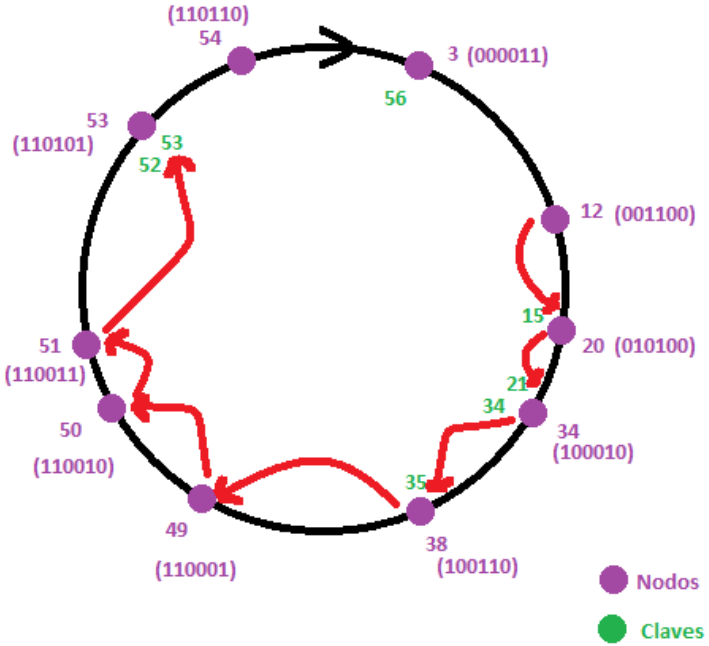


Figura 2-7. Mapeo de elementos de datos hacia nodos organizados en Chord.

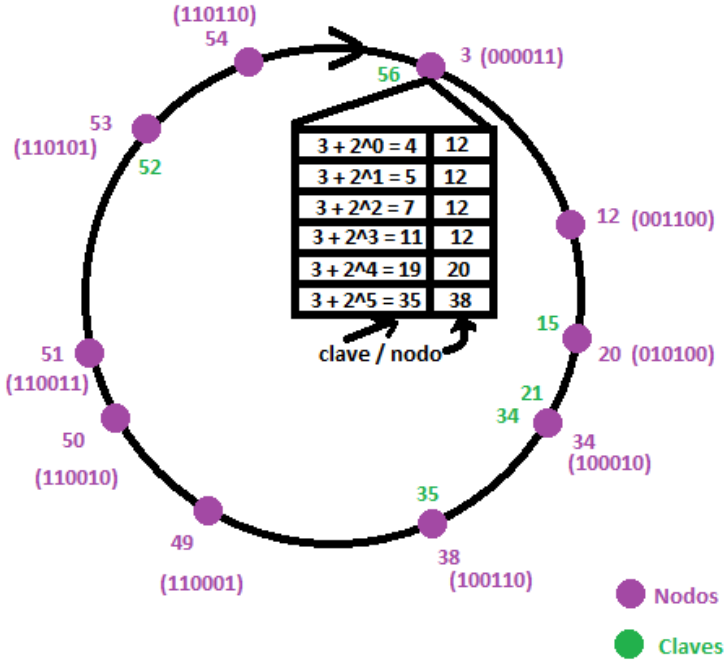
Sistema P2P Estructurado: Búsqueda Lineal en Chord



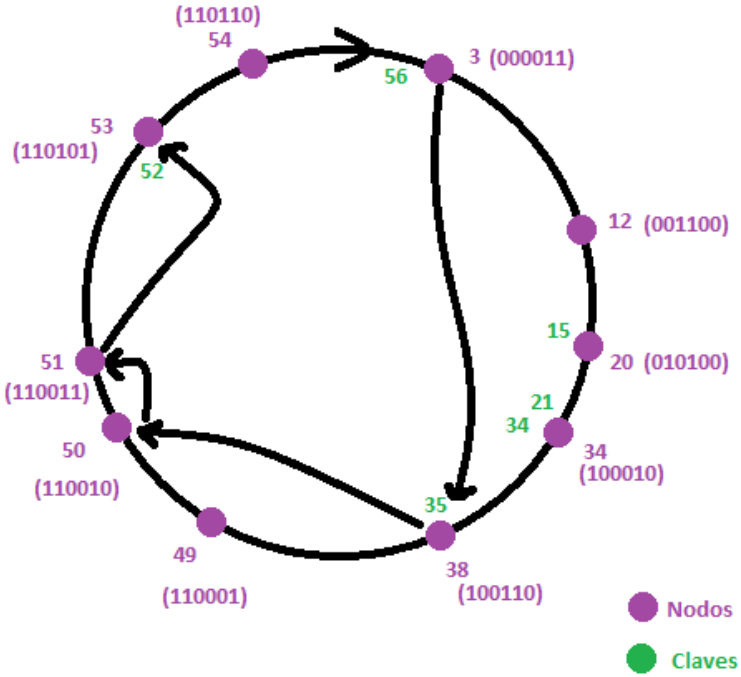
Sistema P2P Estructurado: Finger Tables en Chord

- La búsqueda secuencial puede ser costosa. En el peor de los casos, si cada elemento de dato está en un nodo distinto, entonces se hacen **n** saltos.
- Cada nodo tiene tablas con tantas filas como número de bits tiene el identificador.
- Cada fila almacena un valor, que es el nodo que contiene la clave del elemento de dato.
- Se construye a partir de *node key* + 2^i , donde *i* es la posición en la fila.

Sistema P2P Estructurado: Finger Tables en Chord



Sistema P2P Estructurado: Finger Tables en Chord



Sistema P2P Estructurado: Finger Tables en Chord

- Chord está diseñado para ser altamente escalable. En particular, si **n** es el número de nodos de la red, su coste es proporcional a **log(n)**.

- Es escalable porque solo depende del número de bits del que se compone un identificador. Si queremos más nodos, simplemente asignamos identificadores más largos.

- Es eficiente, porque hace búsquedas en un orden log(n), ya que en cada salto se puede reducir a la mitad el número de saltos que quedan por hacer.

Sistema P2P Estructurado: Pastry

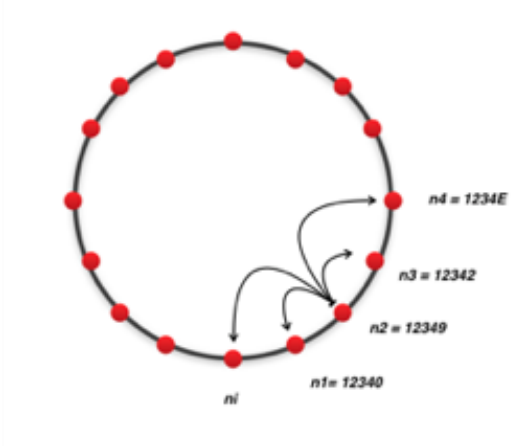
- Creación de una red en forma de anillo con usuarios independientes y son localizados mediante tablas de hojas, tabla de vecinos y tabla de encaminamiento que posee cada nodo.
- A cada nodo se le asigna un identificador único mediante función de hash.
- Las claves de los recursos serán almacenadas por los nodos cuyo identificador sea numéricamente más cercano a estas.
- Cada nodo de la red Pastry almacena referencias a otros nodos en las tablas mencionadas.

Sistema P2P Estructurado: Encaminamiento en Pastry

- Para encaminar una clave k, el nodo de la red Pastry hace uso de las referencias contenidas en su tabla de encaminamiento y conjunto de hojas.
- Primero se consulta la **tabla de hojas**.
  - Si la clave *k* está en la tabla de hojas, la consulta se envía al nodo con identificador numéricamente más cercano.
  - Si no se encuentra la clave *k*, se emplea la tabla de enrutamiento.

Sistema P2P Estructurado: Encaminamiento en Pastry

- La **tabla de hojas** contiene los nodos cuyo identificador es cercano al nodo local.
- Normalmente aparecen los ocho nodos mayores y los ocho nodos menores.



Sistema P2P Estructurado: Encaminamiento en Pastry

- La **tabla de encaminamiento** contiene *i* filas, con rango entre 0 y $\log_2(n)=b$, siendo *n* la cantidad de nodos en la red; y *j* columnas con rango entre 0 y 2^b-1.

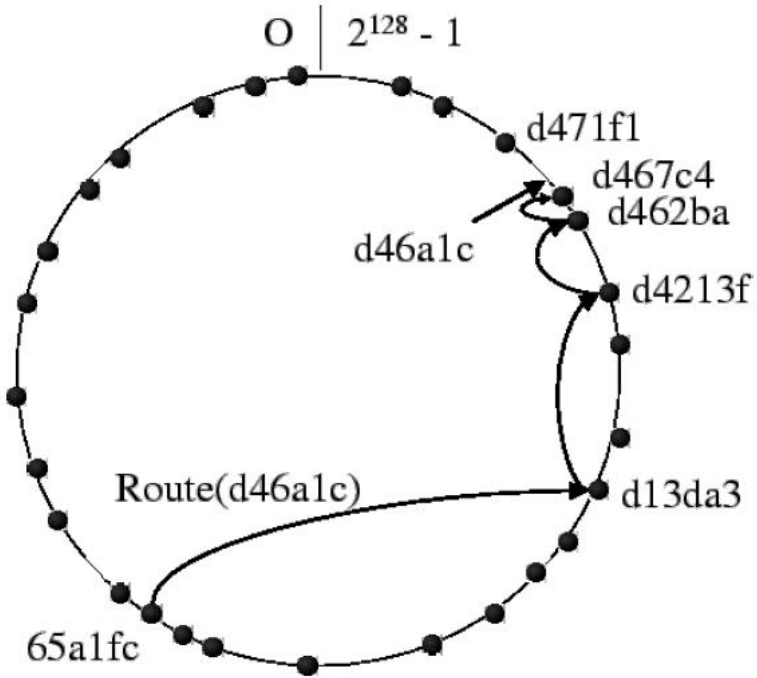
- Cada fila *i* tiene un prefijo común en todas las columnas.

Fila	Tabla de Encaminamiento						
0	0	1	2	3	4	…	F
1	10	11	12	13	14	…	1F
2	120	121	122	123	124	…	2F
3	1230	1231	1232	1233	1234	…	123F

Sistema P2P Estructurado: Encaminamiento en Pastry

- Para buscar un dato con llave *k* se emplea la tabla de encaminamiento.
- Se busca un nodo que comparta el prefijo más largo respecto a la llave *k* del dato buscado.
- Dado que cada vez el prefijo irá aumentando, este protocolo converge.
- La **tabla de vecinos** no se utiliza directamente en el algoritmo, pero es utilizada para iniciar y actualizar la tabla de encaminamiento.
  - Mantiene los nodos más cercanos en cuanto a la distancia espacial.
  - Acorde a dirección IP entregada por programas externos.

Sistema P2P Estructurado: Encaminamiento en Pastry



Sistema P2P Estructurado: Super Peers

- En sistemas P2P no estructurados localizar elementos de datos importantes puede resultar problemático en la medida que la red crece.
- La razón es sencilla, no hay una manera determinista de enrutar una solicitud.
- La solución es tener **super peers**. Peers, es decir, tienen conexión punto a punto, pero se conocen entre si y mantienen una cantidad de peers conectados a ellos.

Sistema P2P Estructurado: Super Peers

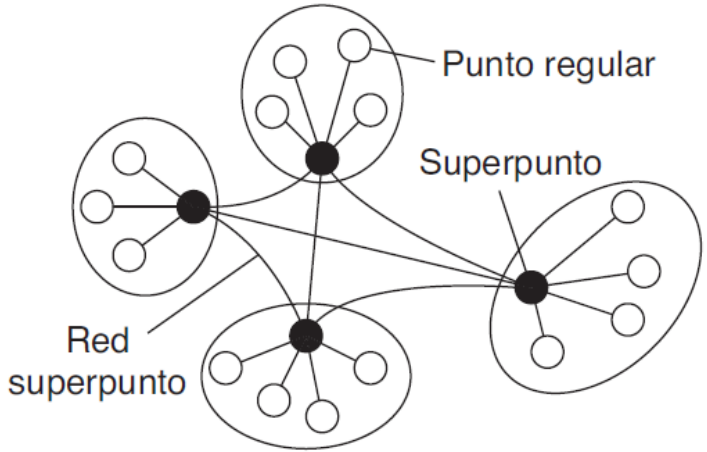
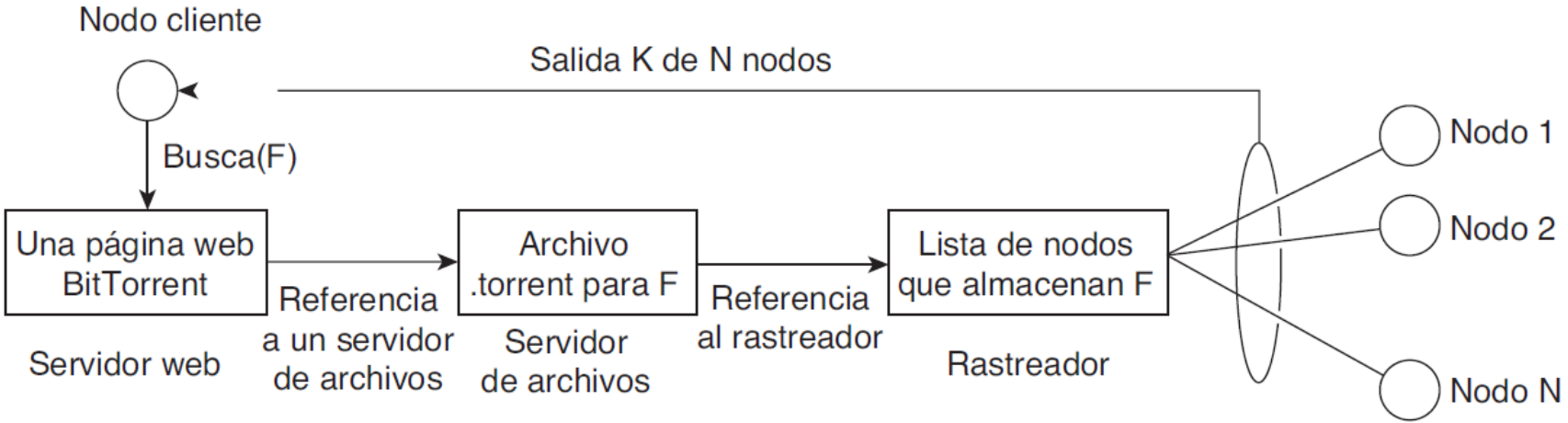


Figura 2-12. Organización jerárquica de nodos en una red de superpunto.

Sistema P2P Estructurado: Sistema Híbrido

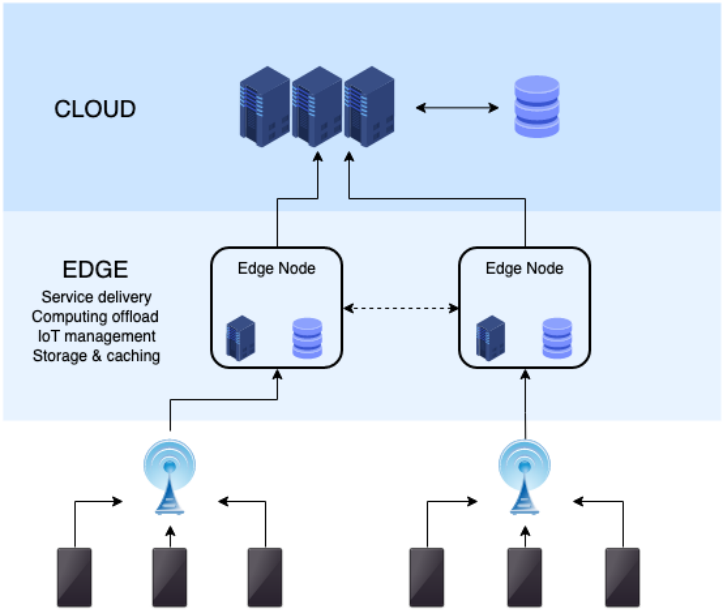
- Con la idea de super peers podemos tener sistemas híbridos, es decir, no son centralizados ni descentralizados.
- Se les denomina semicentralizadas.
- Fuertemente utilizado en **sistemas distribuidos en colaboración**.
  - Inician en una lógica cliente-servidor.
  - Una vez que se logra armar la red, se puede utilizar un sistema descentralizado en colaboración.

Sistema P2P Estructurado: Sistema Híbrido



Sistema P2P Estructurado: Sistema Híbrido

- **Sistemas de servidores al borde.**
- Ofrece baja latencia más cerca de las solicitudes.
- Utilizada cuando se necesita procesamiento de datos en tiempo real.
- Utilizado para el **IoT**.



Caso de Estudio: Bitorrent

- Bitorrent es un protocolo diseñado para el intercambio de archivos en redes P2P.
- Se utilizan servidores que almacenan la información de los archivos compartidos.
- Divide el archivo original en partes y utiliza un hash criptográfico para procurar la consistencia entre las partes y nodos que la contienen.
- Asegura replicación sin necesidad de un servidor central.
- Cuantos más usuarios, mayor probabilidad de encontrar la pieza buscada, reduciendo el ancho de banda utilizado.
- Proporciona fuentes de descarga transitorias, reduciendo la dependencia del distribuidor original.

Caso de Estudio: Componentes de la Red Bitorrent

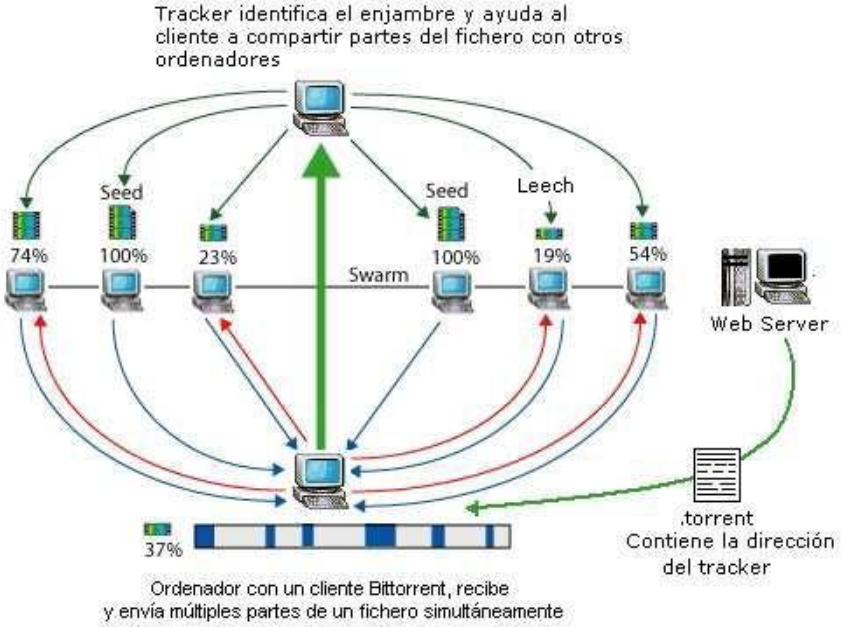
- Leechers (sanguijuelas): Se denomina así a todos los usuarios que están en la red descargando el archivo pero que todavía no tienen el archivo completo. También se llama a quienes descargan archivos pero no los comparten.

- Seeders (semillas): Son los usuarios de la red que poseen el archivo completo.

- Trackers (rastreadores): Un rastreador de BitTorrent es un servidor especial que contiene la información necesaria para que los pares se conecten unos con otros. Inicialmente es la única forma de localizar qué usuarios contienen el archivo que se quiere descargar.

- Swarm (enjambre): El enjambre son los usuarios en general que el rastreador se encarga de buscar. El nombre es debido a la similitud con las abejas y su comportamiento; en esta analogía, el rastreador es el panal de abejas, el enjambre de abejas son los usuarios y la miel es el torrent con el contenido.

Caso de Estudio: Bitorrent



Caso de Estudio: Bitorrent

- Para elegir la siguiente pieza a descargar se tienen 2 algoritmos:

- **Las piezas más raras primero**

  - Se mantiene información sobre la cantidad de Peers que tienen una pieza y con eso se define el conjunto de piezas más raras.
  - La siguiente pieza a descargar se selecciona de manera aleatoria.

- **Algoritmo de bloqueo**

  - Se utiliza para decidir el siguiente nodo con el cual interactuar.
  - Se penaliza a los “free riders”, usuarios que no suben datos y sólo descargan.

Caso de Estudio: Red P2P Spotify

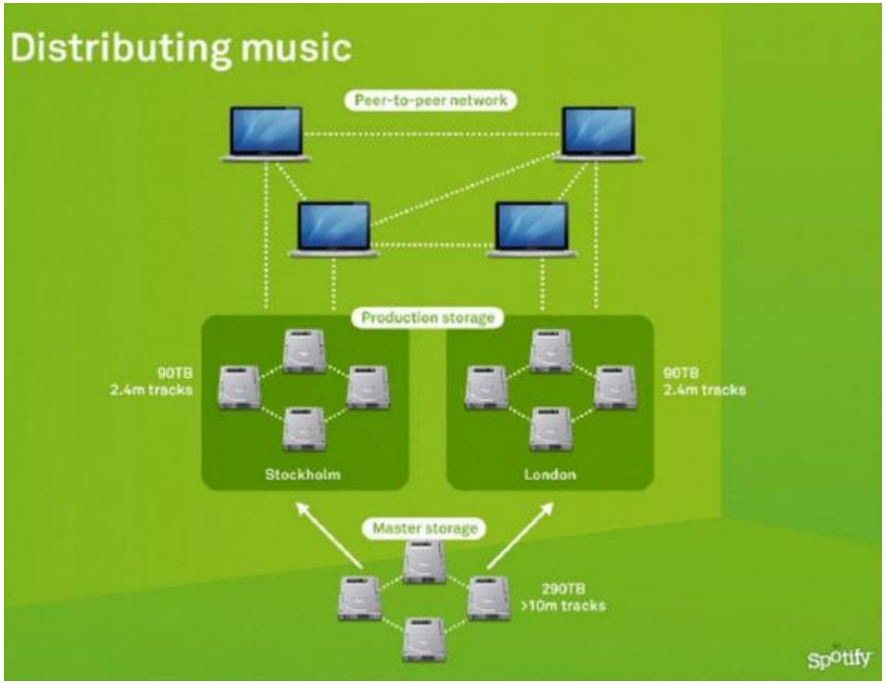
- Una característica poco conocida de Spotify es que utilizó una red P2P hasta mediados del año 2014.

- Cuando la aplicación buscaba una canción, ocurrían 3 cosas:

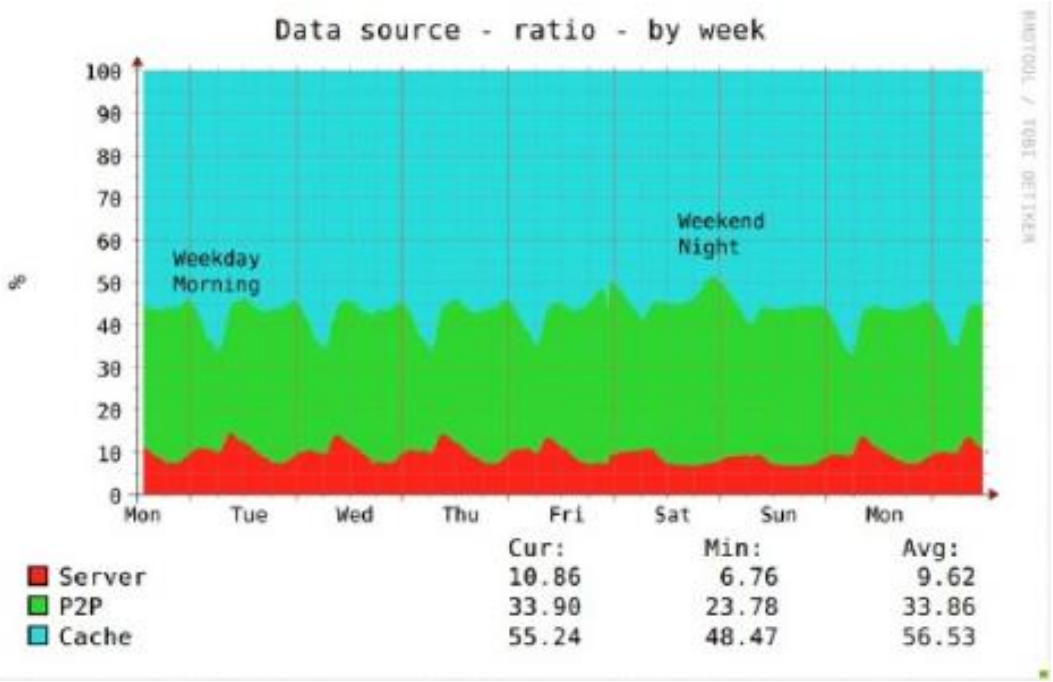
  1. Se buscaba la canción en **memoria caché**.
  2. Se hacía una búsqueda entre los dispositivos cercanos, que tuvieran en funcionamiento la aplicación y que tuvieran una **réplica** (completa o partes) de la canción.
  3. Lo último era buscar la canción en los servidores de Spotify.

- De esta forma, los servidores de Sportify eran utilizados sólo en un 8.8% de las consultas.

Caso de Estudio: Red P2P Spotify

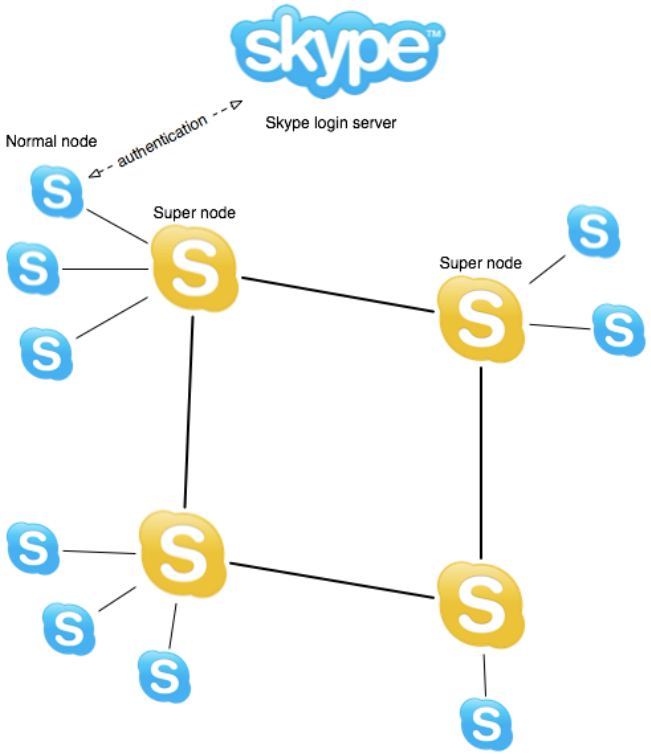


Caso de Estudio: Red P2P Spotify



Caso de Estudio: Skype

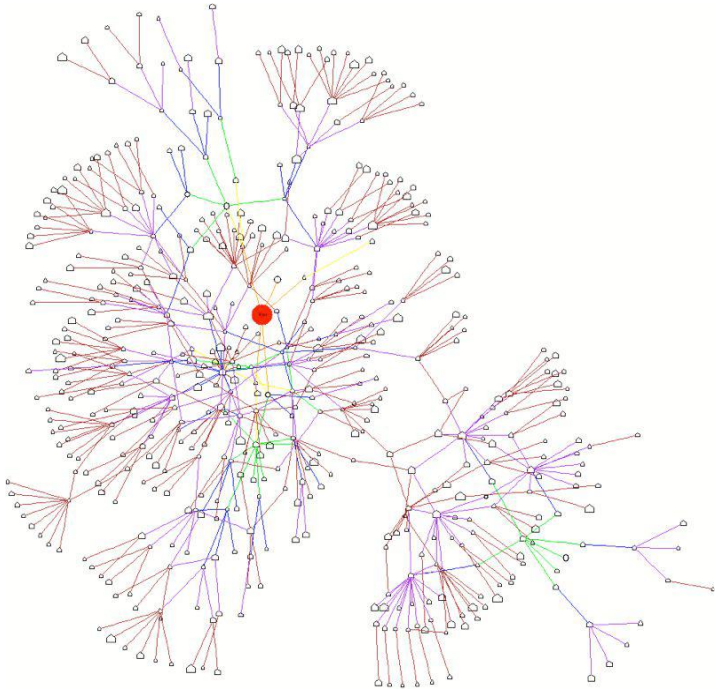
- Primera aplicación de telefonía P2P.
- Reconoce 2 tipos de máquinas: super nodes y normal nodes.
- Logra la conexión a través de la conexión entre super nodes. Una vez establecida, se genera una conexión directa entre los dos nodos normales.



Arquitectura Descentralizada: Sistema P2P No Estructurada

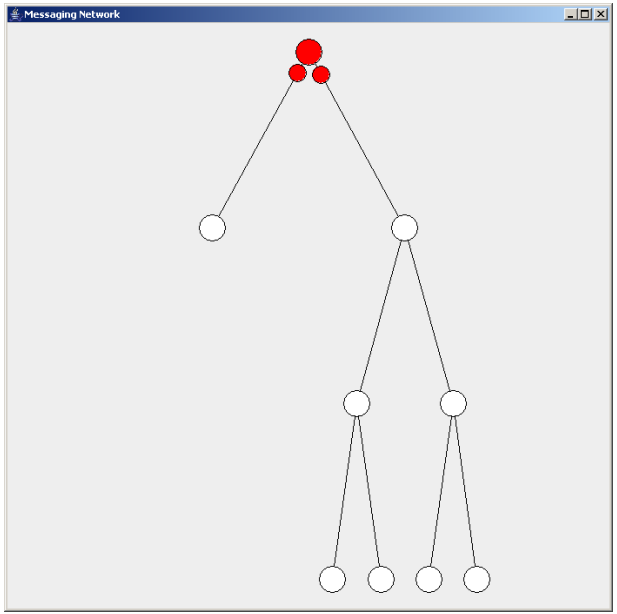
- Son las más versátiles al no requerir un gestionamiento central.
- La estructura se genera de manera aleatoria, manteniendo conocimiento local sobre los vecinos y peers conectados.
- El gran problema es cómo buscar un dato.
    - Se basa en inundación o **flooding.**

Arquitectura Descentralizada: Sistema P2P No Estructurada



Arquitectura Descentralizada: Sistema P2P No Estructurada

- La búsqueda de un dato se hace por **búsqueda en anchura**.
- El mensaje es enviado a todos los participantes de la red.
- Poco eficiente.
- Consume gran parte de la capacidad de la red en una sola consulta.
- La búsqueda puede ser muy lenta en una red grande.
- Si la red tiene loops, duplica el mensaje.



Arquitectura Descentralizada: Sistema P2P No Estructurada

- **Búsqueda en profundidad iterativa**
- Es una mejora de la búsqueda en profundidad.
- Se inicia con una profundidad limitada, que aumenta en cada iteración hasta alcanzar *d*, la profundidad del nodo objetivo de menor profundidad.

Arquitectura Descentralizada: Sistema P2P No Estructurada

- **Búsqueda en profundidad dirigida.**
- Se elige un subconjunto de los vecinos y se envía el mensaje.
- Trata de adivinar qué vecino es más probable que responda.
  - Random.
  - Resultado con menos saltos.
  - Más mensajes recibidos.
  - Más vecinos.
  - Menor latencia.
  - Entre otros.

Arquitectura Descentralizada: Sistema P2P No Estructurada

- **Random Walks**
- Mensaje enviado a un vecino aleatorio con tiempo de vida máximo.
- Se evalúa de manera aleatoria el siguiente salto.
- Ciego y sin memoria.
- **K-Random Walks**
- Múltiples random walks paralelos.
- Disminuye latencia.

Caso de Estudio: Gnutella

- Protocolo de red de distribución entre pares sin un servidor central, por lo que es robusta.

- Es una red P2P “pura”, y su funcionamiento consta de 3 partes:

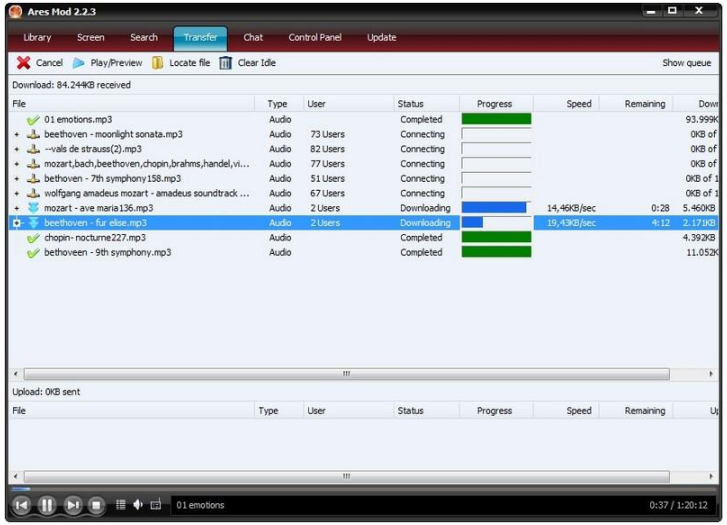
1. **Entrada:** Un nuevo nodo se conecta a la red. Se tiene una lista con nodos que se espera estén siempre conectados y se selecciona un nodo de forma aleatoria
2. **Búsquedas:** Se hace mediante inundación, procurando evitar reenvíos infinitos y bucles.
3. **Descargas:** Se realiza descarga directa desde los nodos que respondieron tener el fichero.

- La inundación es la debilidad de este protocolo.

- El algoritmo de búsqueda no garantiza que se encuentre el archivo, aún cuando esté dentro de la red.

Caso de Estudio: Gnutella

- Este protocolo es utilizado por diversas aplicaciones de intercambio de archivos:
  - FrostWire.
  - iMesh.
  - Phex.
  - BearShare.
  - Ares Galaxy.



Arquitectura Pervasiva: Mobile Adhoc Networks (MANET)

- Redes móviles punto a punto.
- Es una red auto-organizada.
- No tiene una topología definida, lo que favorece la movilidad.
- Las conexiones se realizan a través de WIFI, 3G/4G o Bluethoot.
- Redes locales limitadas al alcance del medio de comunicación.

Mobile Adhoc Networks (MANET): Escenarios

- Militar.
- Vehículos (VANETS).
- Emergencias: Rescates.

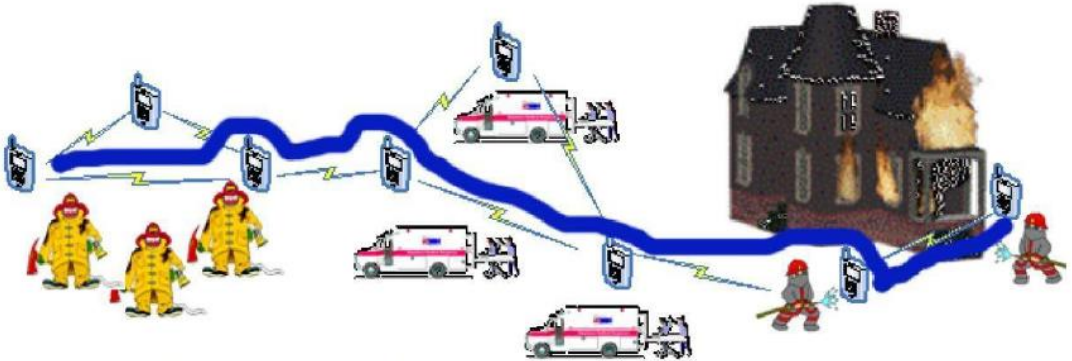
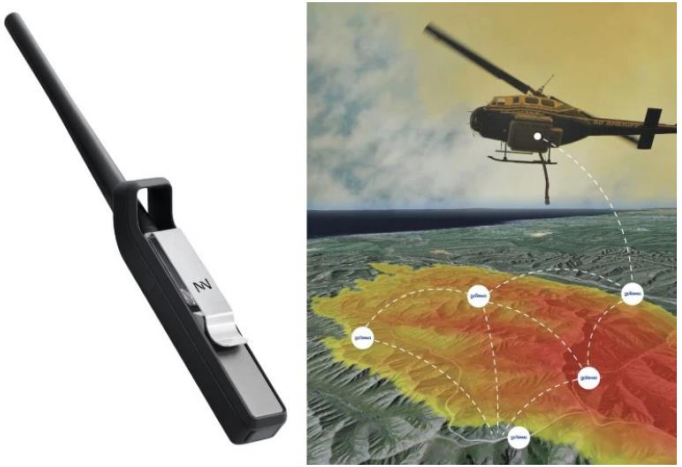


Figure 1 Multi hop communication in MANET

Caso de Estudio: GoTenna

- Empresa dedicada a la construcción de dispositivos de comunicación descentralizados y fuera de red.



DEPARTAMENTO DE
INGENIERÍA
INFORMÁTICA
UNIVERSIDAD DE SANTIAGO DE CHILE

Departamento de Ingeniería Informática
Universidad de Santiago de Chile

¿CONSULTAS?