Peer-to-peer (Pair à pair)

IFT 6802

Par Laurent Magnin

Université de Montréal

Réseaux *Peer-to-Peer - P2P* (pair à pair)

©Laurent Magnin,

Largement inspiré par

« Communautés virtuelles: une analyse expérimentale du réseau *Peer-to-Peer* Gnutella » (Jean Vaucher, UdeM)

+ présentation « Chord » (Arnaud Dury, UdeM, Crim)

Université de Montréal

Communautés

« Social aggregates emerging from the Internet when enough people carry on public discussions long enough and with sufficient human feeling to form webs of personal relationships .» [Howard Rheingold]

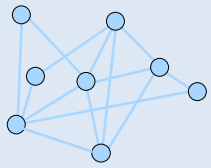
- **intérêts communs**
- **contexte partagé**
- **auto-organisation (*self-organization*)**
- **Individus (entités) autonomes**

Systèmes *Peer-to-Peer* (P2P) - Définition

- **Un réseau P2P est un ensemble de clients informatique partageant leurs données et/ou leurs ressources à travers un réseau de communications.**

Systèmes *Peer-to-Peer* (P2P)

- **Chaque participant est à la fois client et serveur («servent»)**
- **Complètement décentralisé :**
  - aucun contrôle/coordination centrale
  - aucune base de données centrale
  - le comportement global émerge des interactions locales
  - Tous les services et données sont accessibles par tous les «peers»
  - Les «peers» sont autonomes.
  - Les «peers» et leurs interconnexions ne sont pas fiables
- **«Modèle d'affaire»: chaque noeud contribue/paie par la mise à disposition de (quelques unes de) ses ressources**
- **Communautés en ligne**



Napster, La première génération P2P

- **Première version : mai 1999**
- **Destiné aux fichiers musicaux**
- **Un serveur centralisé sert d'index**
- **Provoque la colère de la RIAA**
- **Le serveur central sera fermé par décision juridique le 26 juillet 2000**
- **La fermeture de Napster s'accompagne de l'éclosion d'alternatives indépendantes d'un serveur central**

Gnutella, la deuxième génération

- **Pas de serveur centralisé, suite à l'expérience juridique Napster**
- **Architecture simple, robuste et assez « scalable »**
  - Les requêtes se propagent par inondation brutale du réseau
  - Le download se fait via HTTP
- **Système ouvert, indépendant des types de fichiers recherchés**
- **Très dynamique, pairs autonomes, auto-organisés**
- **Beaucoup de logiciel « OpenSource »**

Gnutella : historique

- **Développé en 14 jours par Nullsoft (winamp) dans le but de faciliter l'échange de recettes**
- **Publié sous «GNU General Public License» sur le serveur Web de Nullsoft**
- **Retiré quelques heures plus tard par AOL (propriétaire de Nullsoft)**
- **L'Internet était alors déjà infecté !**
- **Le protocole de communication de Gnutella a été «défini» à partir des versions téléchargées du serveur de Nullsoft (ré-ingénierie couronnée de succès…)**
- **Distribution rapide, surtout après la fermeture de Napster**

Gnutella : clients disponibles (avril 2004)

- **Cf. http://www.gnutella.com/**
- **Windows**
  - BearShare
  - Gnucleus
  - Morpheus
  - Swapper
  - XoloX Ultra
  - LimeWire
  - Phex
- **Linux/Unix**
  - Gtk-Gnutella
  - Mutella
  - Qtella
  - LimeWire
  - Phex
- **Macintosh**
  - LimeWire
  - Phex

Client LimeWire

- **Version commerciale + en OpenSource (http://www.limewire.org/)**
- ***"The LimeWire program will connect at startup, via the internet, to the LimeWire Gateway, a specialized intelligent Gnutella router, to maximize the user's viewable network space."***
- ***"Limewire is written in Java, and will run on Windows, Macintosh, Linux, Sun, and other computing platforms."***
- ***Pattern* de "façade", mais documentation périmée**

Client Phex

- **Projet *Open-Source* : http://phex.sourceforge.net/**
- ***"Phex is entirely based on William W. Wong's Furi", which "is a Gnutella protocol-compatible Java program that can participate in the Gnutella network. It is a full version program with a easy to use GUI interface that can perform most of the tasks of a Gnutella servant."***
- **Absence de documentation**
- **GUI et "core" imbriqués**

Protocole Gnutella : anonymat relatif

- **Requêtes ne contiennent pas l'identité**
- **Chaque « servent » connaît seulement les voisins auxquels il est directement connecté**
- **Le téléchargement des fichiers par HTTP révèle l'identité**

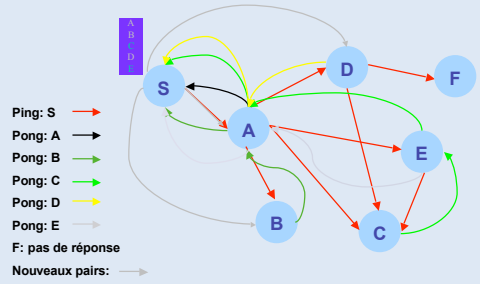
Réseau Gnutella

- **Programme Gnutella = serveur et client: « servent »**
- **Joindre le système : connexion à un hôte connu (e.g. gnutellahosts.com:6346)**
- **Maintenir le fonctionnement du système (connectivité): ping/pong continu**
- **Inondation: Requêtes sont envoyées aux voisins (souvent 4 ou 5) qui les acheminent à leur voisins; limitation par un TTL**
- **Pour chaque requête, les « servents » recherchent des fichiers locaux correspondants à la requête**
- **Transfert de fichiers avec HTTP (UDP « *out of band* »)**
- **Rencontre : connexion TCP** GNUTELLA CONNECT/x.x\n\n GNUTELLA OK\n\n

Protocole : types de message

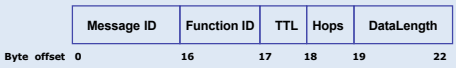
Type	*Description*	*Information*
Query	Requête de recherche de l'utilisateur	Vitesse, mots clés
Reply (QueryHit)	Réponses d'hôtes correspondantes à la requête	description de fichier, adresse hôte:port, largeur de bande de l'hôte, servent ID
Ping	requête pour des adresses d'hôtes	aucune
Pong	Réponse à Ping	hôte:port adresse, largeur de bande de l'hôte, nombre et taille totale de fichiers locaux
Push	Requête de charger un fichier d'un servent derrière firewall	servent ID, index du fichier demandé, hôte:port adresse

Rencontre de pairs avec Ping/Pong



En-tête de paquet (*Descriptor*)

- **MessageID : 16 octet id unique**
- **FunctionID : type de paquet**
- **RemainingTTL : nombre de fois que le paquet sera réexpédié (souvent 7)**
- **HopsTaken : TTL(0) = TTL(i) + Hops(i)**
- **DataLength : longueur des données suivant l'en-tête**



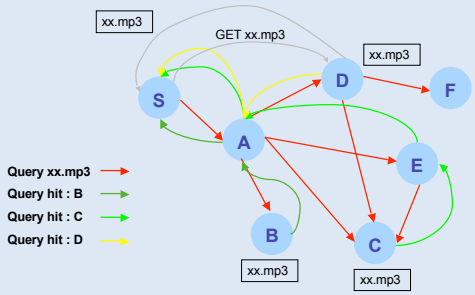
Messages Ping/Pong

- **En-tête de message:**

- **Ping (0x00) : en-tête avec donnée 0x00**
- **Pong (0x01): en-tête avec données :**

  - Port où l'hôte répondant accepte des connexions
  - Adresse IP de l'hôte répondant
  - Nombre de fichiers partagés
  - Nombre KBytes partagés

Requête: Query/QueryHit/GET



Query

- **Query (0x80):**

Minimum speed	Search criteria

Byte offset 0 1 2 ....

- Minimum speed : la largeur de bande minimale (en kbps) du/des «servent» répondant(s)
- Search criteria : une chaîne de caractère avec un zéro (i.e., 0x00) à la fin; la longueur maximale est limitée par "Payload length" dans l'en-tête.

QueryHit

QueryHit (0x81)

Number of hits	Port	IP address	Speed	Result set	Servent identifier

Byte offset 0 1 2 3 6 7 10 11 .... n n+16

- Number of hits: réponse
- Port: où le répondant accepte des connexions
- IP address: adresse du répondant
- Speed: vitesse (largeur de bande) du répondant
- Servent identifier: 16-byte mot identifiant le répondant (unique)
- Result set : résultats (*number of hits* entrées)

File index	File size	File name

Byte offset 0 3 4 7 8 .....

- File index: un numéro unique assigné au fichier par le répondant
- File size: taille du fichier (en bytes)
- File name: nom du fichier terminé par deux zéros (0x0000) terminated

Téléchargement de fichiers

- **«Out of band» avec HTTP simplifié**
- **Connexion à l'adresse IP donnée dans QueryHit**
- **Exemple:**

2468	4356789	Foobar.mp3\0x00\0x00
File index	File size	File name

```
GET /get/2468/xx.mp3/ HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0\r\n
User-Agent: Gnutella\r\n
\r\n
```

```
HTTP 200 OK\r\n
Server: Gnutella\r\n
Content-type: application/binary\r\n
Content-length: 4356789\r\n
\r\n
<data> ...
```


Gnutella Push : servent derrière firewall

- **Servent *A* reçoit un QueryHit du servent *B* qui est derrière un firewall et n'accepte que des connexions sur le port de Gnutella**
- ***A* envoie Push (0x40) à *B***

Servent identifier	File index	IP address	Port

- ***B* ouvre une connexion à l'adresse IP/port fourni dans le message Push et envoie:**

```
GIV <File index>:<Servent identifier>/<File name>\n\n
```

- **Quand A reçoit GIV il commence un téléchargement ordinaire par cette connexion:**

```
GET /get/<File index>/<File name>/ HTTP/1.0\r\n
Connection: Keep-Alive\r\n
Range: bytes=0\r\n
User-Agent: Gnutella\r\n
\r\n
```

- **Ne marche pas, si les deux servent se trouvent derrière des firewalls**

Kazaa, la troisième génération

- **Pas de serveur centralisé, ni d'inondation brutale**
- **Les noeuds les plus rapides sont élus « super-noeuds »**
- **Les supers-noeuds créent l'ossature du réseau de recherche d'informations**
- **Le download se fait en « multi-sources »**

L'infrastructure JXTA

- **http://www.jxta.org/**
- **Objectifs du project JXTA :**
  - *Interoperability - across different peer-to-peer systems and communities*
  - *Platform independence - multiple/diverse languages* [uniquement Java ?], *systems, and networks*
  - *Ubiquity - every device with a digital heartbeat*
- **Projet OpenSource supporté par Sun Microsystems**

Réseau *FreeNet*

- **http://freenet.sourceforge.net (code en Java)**
- ***"I worry about my child and the Internet all the time, even though she's too young to have logged on yet. Here's what I worry about. I worry that 10 or 15 years from now, she will come to me and say 'Daddy, where were you when they took freedom of the press away from the Internet?"*** **Mike Godwin, Electronic Frontier Foundation**
- ***"Freenet is free software designed to provide a way to publish and obtain information on the Internet without fear of censorship"***

Réseau *FreeNet*

- **Confidentialité :**
  - Des clients et des serveurs
  - Lors des requètes et lors des téléchargements
- **Se base sur la notion de *proxy* : les différents noeuds servent de relais entre-eux**
- ***"The anonymity that Freenet offers is really just obscurity in the fact that it is hard to prove that your node wasn't proxying the request for or insert of data on behalf of somebody else (who might also just have been proxying it)."***

Le réseau "Donkey"

- **http://www.edonkey2000-france.com/**
- **« Vous pouvez transférer n'importe quels types de fichiers. Il reprend automatiquement les transferts interrompus à partir d'autres sources. Il permet même de partager un ensemble de fichiers (collection) ce qui assure la récupération d'un album complet ou d'un fichier en plusieurs parties. Le chargement peut s'effectuer en parallèle à partir de plusieurs sources. Ceci permet d'assurer le chargement le plus rapide possible pour vous et le moins lourd pour ceux qui partagent puisque le chargement est réparti. »**
- **« Vous pouvez partager un fichier pendant que vous le charger chez quelqu'un d'autre. Ceci permet de distribuer aussi rapidement que possible un fichier peu présent sur le réseau. »**

Le réseau "Donkey" : les clients

- **Client officiel : eDonkey2000**
- **Client Open Source : mlDonkey**
  - http://savannah.nongnu.org/projects/mldonkey/
  - Écrit en Objective-Caml pour systèmes Unix
  - Séparation entre le moteur et l'interface graphique (Web, Telnet, application spécialisée)

Overnet, la quatrième génération

- **Client P2P récent: www.overnet.com**
- **Informations disponibles**
  - Each node is assigned a random 128bit ID. This ID serves as the node's address in the network. When a node wants to publish some small piece of data to the Overnet, the data is also assigned a unique 128bit ID. The node that has an ID closest to the data's ID is responsible for this piece of data. The responsible node is looked up in the network. Once found, the publishing node sends the responsible node the piece of data. This is publishing.
  - When a node wants to search for something it must determine the ID of the thing it is looking for. Then it finds the node that is responsible for that ID. Then it asks for the list of data pieces that correspond to that ID.

Overnet, la quatrième génération

- **Client P2P récent: www.overnet.com**
- **Informations disponibles**
  - Each node is assigned a random 128bit ID. This ID serves as the node's address in the network. When a node wants to publish some small piece of data to the Overnet, the data is also assigned a unique 128bit ID. The node that has an ID closest to the data's ID is responsible for this piece of data. The responsible node is looked up in the network. Once found, the publishing node sends the responsible node the piece of data. This is publishing.
  - When a node wants to search for something it must determine the ID of the thing it is looking for. Then it finds the node that is responsible for that ID. Then it asks for the list of data pieces that correspond to that ID.

Overnet, la quatrième génération

- **Problème : ne colle pas à la réalité, Overnet n'upload pas les fichiers qu'il partage au démarrage, et heureusement !**
- **Probablement une implantation de l'extension suggérée par Chord:**
  - Seule la table d'index est distribuée

Chord : index distribué

- **Développé par le MIT : http://www.pdos.lcs.mit.edu/chord/**
- ***"The Chord project aims to build scalable, robust distributed systems using peer-to-peer ideas. The basis for much of our work is the Chord distributed hash lookup primitive. Chord is completely decentralized and symmetric, and can find data using only log(N) messages, where N is the number of nodes in the system. Chord's lookup mechanism is provably robust in the face of frequent node failures and re-joins."***
- ***"One way that we use Chord is as the basis for the CFS (Cooperative File System) storage system. CFS allows anyone to publish and update their own file system, and provides read-only access to others."***

Distribution d'une table de hachage : Chord

- **Clés de hachage sur les fichiers d'un côté et les identifiants de noeuds de l'autre**
- **Les noeuds sont chainés sur un anneau**
- **La clé de fichier k est associée au noeud n de plus faible indice tel que: n>=k**
- **Pas de recherche par mots-clés dans la version actuelle**

Applications possibles

- **Réplication coopérative**
  - Assurer la charge moyenne au lieu de la charge maximale
- **Hébergement en temps partagé**
  - Informations disponibles tout le temps, serveur disponible sporadiquement
- **Index distribués**
  - Recherche d'informations P2P efficace
- **Recherche combinatoire à grande échelle**
  - Distribution des tâches

Overnet, hypothèses

- **Client se connecte:**
  - Calcule les clés de hash des noms des fichiers locaux partagés
  - Publie les noms de fichiers aux k plus proches ID (décompose les noms en mots, d'après les séparateurs)
- **Recherche**
  - Calcule les clé de hash sur les mots clés de la requête
  - Interroge autant de clients que de mots de la requête
  - Fusionne localement les réponses
  - Ne crée pas de clés pour les mots de moins de n lettres (n=3 ?)
- **Probable: re-publication périodique pour conserver la cohérence (join/leave)**

Projet à long terme : la cinquième génération

- **Hachâge + communautés + requêtes durables**
- **Création de communauté d'expertise à court terme**
  - Un client ayant effectué une requête récemment peut devenir la source d'informations idéale pour une requête similaire dans le futur proche
- **Création de communautés d'intérêts à long terme**
  - Des clients effectuant des requêtes similaires, ou possédant des fichiers similaires ont intérêts à se proposer l'un à l'autre leurs fichiers locaux
- **Architecture à base d'agents mobiles**
  - Requêtes de très longue durée, les utilisateurs sont représentés par des agents se déplaçant de noeud en noeud, vivant indépendemment du client

Usage commercial : BitTorrent

- **Protocole destiné au transfert de large fichiers (et non à leur découverte)**
- **Le transfert peut se faire en même temps que le téléchargement**
- **Des compagnies proposent ce mode de diffusion de leur propres fichiers**

Références P2P

- **J. Vaucher, G. Babin, P. Kropf, Th. Jouve: *Experimenting with Gnutella Communities*, Distributed Communities on the Web, Sydney, April 2002, LNCS 2468, Springer Berlin, pp. 85-99**
- **http://openp2p.com/**
- **http://www.zeropaid.com/**