

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Université Aboubakr Belkaïd – Tlemcen – Faculté des Sciences
Département d'Informatique



Mémoire de fin d'études
Présenté pour l'obtention du diplôme de MASTER en : Informatique
Spécialité : Réseaux et Systèmes Distribués (R.S.D.)

Par :

Didi Souheyla

,

Cherifi Sabah

Thème

Mise au point d'une application de téléchargement et de communication en Peer-to-Peer

Soutenu publiquement, le 02/06/ 2016, devant le jury composé de :

Mme la Présidente : Labraoui Nabila

Examineur : Belhocine Amine

Examineur : Benaïssa Med Samir

Encadreur : Didi Fedoua

Année universitaire : 2015-2016

Remerciements

Avant tout, nous tenons à remercier Dieu le tout puissant qui nous a donné la force, la patience et la volonté d'accomplir ce modeste travail.

Nous remercions en cette occasion tout le corps professoral et administratif du département d'informatique de l'université de Tlemcen pour la richesse et la qualité de leur enseignement et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Nous tenons très particulièrement à exprimer notre gratitude à Mme. DIDI Fedoua pour avoir acceptée de suivre notre mémoire. Et pour ses précieux conseils et sa présence encourageante, son encadrement de qualité et ses orientations ficelées tout au long de notre travail.

Nos remerciements vont aussi aux membres du jury, Mme Labraoui, messieurs Belhocine et Benaisa pour nous avoir fait l'honneur d'examiner et évaluer notre travail. Qu'ils trouvent ici l'expression de notre infini respect.

Et bien évidemment nous remercions nos familles et nos amis qui par leurs prières et leurs encouragements, nous avons pu surmonter tous les obstacles.

Dédicaces

Je dédie cet humble travail avec grand amour, sincérité et fierté : A mes parents, source de tendresse, de noblesse et d'affection.

A mon frère et à ma sœur, en témoignage de la fraternité, avec mes souhaits de bonheur, de santé et de succès. Et à tous les membres de ma famille.

A tous mes enseignants, tous mes amis, et à tous ceux qui compulseront ce modeste travail.



Didi Souheyta

Dédicaces

Louange à Allah le Miséricorde

Merci à mon dieu de m'avoir donné la volonté et le courage pour la réalisation de ce travail.

Avec un énorme plaisir, un cœur ouvert et une immense joie, que je dédie mon travail :

À mes parents :

Grâce à leurs tendres encouragements et leurs grands sacrifices, ils ont pu créer le climat affectueux et propice à la poursuite de mes études.

Aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux.

Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

À mes chers frères, Fouad et Moustafa, à ma nièce Assil et sa maman, à mes oncles et mes tantes, à tous mes cousins et cousines.

À mes ami(e)s pour leur soutien moral durant l'élaboration de ce travail et surtout Tahraoui Hayet et ma chère Benledghem Rafika pour son aide, ses conseils et son encouragement et à tous mes camarades de promotion.

À mes enseignants de la filière Informatique, et surtout les enseignants de la spécialité Réseaux et Systèmes Distribués.

À tous ceux et toutes celles qui m'ont soutenu de près ou de loin durant cette année.

À tous qui me sont chers.

SABAH

Table des matières

| | |
|---|----|
| Table des matières | 1 |
| Introduction générale | 3 |
| Chapitre 1 : Généralités sur le Peer-to-Peer..... | 4 |
| 1. Introduction..... | 5 |
| 2. Historique..... | 5 |
| 3. Définition du Peer-to-Peer | 6 |
| 4. Les caractéristiques du Peer-to-Peer | 6 |
| 5. Les Objectifs du modèle Peer-to-Peer | 7 |
| 6. Les Problèmes du Peer-to-Peer..... | 7 |
| A. Premier problème : trouver l'adresse..... | 7 |
| B. Deuxième problème : s'adapter aux caractéristiques de l'ADSL | 8 |
| C. Troisième problème : optimiser les protocoles | 8 |
| 7. Pourquoi utiliser des applications P2P ?..... | 8 |
| 8. Le P2P est-il infallible ? | 9 |
| A. Les problèmes de comportement des utilisateurs..... | 9 |
| B. Les problèmes de comportement des providers et autres entreprises | 9 |
| C. Le problème des droits d'auteurs..... | 10 |
| 9. P2P versus client/serveur | 10 |
| 10. Les Champs d'application des réseaux P2P..... | 11 |
| 11. Conclusion..... | 12 |
| Chapitre 2: Les Architectures Peer to Peer | 13 |
| 1. Introduction..... | 13 |
| 2. Les systèmes Peer-to-Peer non structurés..... | 13 |
| A. Peer-to-Peer centralisé | 14 |
| B. Peer-to-Peer décentralisé « purs »..... | 17 |
| C. P2P hybrides..... | 23 |
| D. Le téléchargement multiple (réseau eDonkey et logiciels de partage eDonkey, eMule, mlDonkey) | 24 |
| E. Le logiciel Freenet, la liberté à l'état pur | 25 |
| F. Le logiciel BitTorrent | 26 |
| 3. Bilan des systèmes P2P non structurés | 27 |
| 4. Les systèmes Peer-to-Peer Structurés | 27 |
| A. Table de Hachage Distribuée (DHT Distributed hashed table) | 28 |

| | |
|--|----|
| B. Le système chord..... | 29 |
| C. Le système Can (Content Adressable Network) | 32 |
| D. Le système Pastry | 35 |
| E. Le système Kademlia | 37 |
| 5. Bilan des systèmes P2P structurés | 39 |
| 6. Bilan des architectures Peer-to-Peer | 39 |
| 7. Les logiciels p2p les plus courants | 40 |
| 8. L'avenir du p2p | 41 |
| 9. Conclusion | 42 |
| Chapitre3 : Présentation de notre Application..... | 43 |
| 1. Introduction..... | 43 |
| 2. La Modélisation de notre application | 43 |
| 3. Les Diagrammes | 43 |
| 4. Dimensionnement des couches matérielles | 45 |
| 5. Caractéristiques de l'application..... | 46 |
| 6. Architecture | 46 |
| 7. Interface de l'application | 46 |
| 8. Fonctionnement | 50 |
| 9. Les messages échangés | 51 |
| 10. Les package utilisés..... | 52 |
| 11. Conclusion..... | 53 |
| Conclusion générale et Perspectives..... | 54 |
| Bibliographie | 56 |
| Liste de figures..... | 58 |
| Liste des tableaux..... | 59 |

Introduction générale

En quelques années seulement, internet a connu un développement incommensurable, offrant aux utilisateurs du globe un moyen rapide et efficace pour partager des informations. La montée du nombre de transfert de fichiers est aujourd'hui considérable et la taille des fichiers transférés a également été multipliée par dix ces dernières années. De nos jours, Il n'est plus rare de voir des échanges de plusieurs Giga octets de données sur internet. Cet accroissement des échanges a mis à jour les limites du modèle client/serveur qui se fait vieux et l'engouement pour ce nouvel arrivé qu'est le Peer to Peer. Malgré une augmentation considérable de leur puissance de traitement, les serveurs ont de plus en plus de peine à répondre aux demandes de tous les clients et on s'est orienté de plus en plus vers des architectures N-tiers pour tenter de répartir la charge, mais l'avènement du "Peer-to-Peer" représente aujourd'hui de manière indéniable une partie considérable des échanges sur internet. Leur développement est lié en majeure partie à l'augmentation des capacités matériels (capacité de stockage, bande passante) des internautes et intéressent de plus en plus les grandes entreprises. Des sociétés comme RedHat utilisent aujourd'hui les Peer-to-Peer pour distribuer leurs produits afin de soulager leurs serveurs (la taille des distributions linux atteignant plusieurs Giga de données). Les Peer-to-Peer ne se limitent plus aujourd'hui au partage de fichiers et de nombreuses applications, reposant sur le même principe, ont vu le jour comme par exemple des logiciels de téléphonie sur ip, ou encore de travail collaboratif. Le fait d'utiliser des architectures décentralisées ou semi décentralisées a conduit inévitablement à l'apparition de nombreux problèmes de sécurités pour les utilisateurs. De plus, les protocoles d'échange de fichiers sont souvent considérés comme une violation des droits de propriété intellectuelle et subissent donc des attaques incessantes des sociétés représentant les ayant droits. Ces difficultés sont à l'origine du développement significatif des Peer-to-Peer.

C'est donc pour toutes ces raisons que notre intérêt s'est porté sur cette thématique en plein essor (le Peer-to-Peer), et donc le but de ce PFE est la mise au point d'une application Peer to Peer totalement décentralisé qui fait du téléchargement de fichiers, de la vidéo conférence et du chat.

Ce mémoire est par conséquent divisé en 3 chapitres :

Le 1^{er} chapitre s'attaque à ce qu'on entend par "Peer-to-Peer" et traite de généralités sur cette thématique, en particulier les problèmes rencontrés tout au long de sa genèse, il traite des différences entre le modèle client/serveur et le modèle Peer-to-Peer, en citant quelques-uns de ses avantages et inconvénients.

Le 2^{ème} chapitre met l'accent sur les nombreuses solutions apportées par l'état de l'art à cette problématique, ainsi que leur mode de fonctionnement.

Le 3^{ème} chapitre est réservé à l'implémentation de notre application, citant respectivement l'environnement de travail, les objectifs à atteindre initialement posés, et le plan qui a été suivi afin d'aboutir à un produit fonctionnel.

Et pour finir, une conclusion générale avec des perspectives et un bref passage qui parle de l'avenir du Peer-to-Peer a été ajouté.

Chapitre 1 : Généralités sur le Peer-to-Peer

1. Introduction

Le Peer-to-Peer (réseau d'égal à égal), fait appel à un principe qui est le fondement même de tout dispositif informatique relié en réseau : celui du partage des ressources et de la communication à distance entre pairs. Il convient ici, avant de se lancer dans le cœur du sujet, de connaître quelques notions de base, ayant trait au p2p, comme ses objectifs quand il a été mis en service et les problèmes rencontrés tout au long de son évolution.

Et enfin nous citerons brièvement quelques champs d'application de ce dernier.

2. Historique

La hausse de la taille des fichiers échangés, la baisse des prix des offres des FAI et celle des matériels et périphériques informatiques sont les facteurs technologiques et commerciaux qui ont permis l'émergence du P2P. Voici les dates clés de cette ascension fulgurante [1] [2]:

Années 60 :

À l'origine Internet était axé sur l'échange d'informations entre universités (ARPANET - 1969) mais inexploitable car non transportable pour une utilisation privée.

1979 :

Deux étudiants développent le système Usenet, basé sur le protocole UUCP (Unix-to-Unix Protocol), permettant ainsi l'échange de nouvelles (News en anglais) via un modèle de communication P2P. Chaque serveur Usenet se connecte à un serveur pair et échange avec ce dernier les informations dont il dispose : émission des news postées par ses utilisateurs et réception des news disponibles sur le serveur pair.

En quelques mois, le Système nommé par abus de langage réseau Usenet s'enrichit de centaines de milliers de serveurs qui se synchronisent par échange de pair à pair.

Mai 1999 :

Naissance du premier logiciel de téléchargement utilisé à grande échelle : Napster. A la base il est le fruit de la réflexion du jeune Shawn Fanning (dix-huit ans) qui décide de créer un logiciel lui permettant d'échanger de la musique en comité restreint avec ses

amis. Très vite, Napster devient le premier véritable phénomène Peer-to-Peer global.

Décembre 1999 :

Les sociétés musicales prennent peur et entament des procédures pour fermer Napster. amis. Très vite, Napster devient le premier véritable phénomène Peer-to-Peer global.

Début 2000 :

Naissance de Kazaa. Offre la possibilité nouvelle de pouvoir télécharger les morceaux d'un même fichier depuis plusieurs sources afin d'accroître la vitesse de réception.

2003 :

Apparition de eMule et de Donkey : nouvelle façon de télécharger les fichiers. Il s'agit de la technique de "fractionnement de fichier" : à peine un téléchargement est commencé que la partie récupérée est disponible pour l'envoi. Cette méthode permet d'augmenter encore la vitesse de transfert de fichiers. Elle est encore aujourd'hui très populaire.

Fin 2005 :

Afin de faire face aux majors qui entament de plus en plus de procédures à l'encontre des internautes qui téléchargent, une nouvelle génération de clients et réseaux P2P a vu le jour. Il s'agit de réseaux totalement cryptés et anonymes techniquement quasi-impossible à contrôler.

3. Définition du Peer-to-Peer

Le P2P est une relation d'échanges réciproques qui unit directement deux acteurs de même statut (des pairs, donc). A l'origine elle désigne une technologie d'échange de fichiers entre internautes, permettant à deux ordinateurs reliés à Internet de communiquer directement l'un avec l'autre sans passer par un serveur central. Au-delà, elle désigne le modèle économique qui y est associé qui permet une mise en commun à une vaste échelle de données ou de capacités, débouchant sur une réduction spectaculaire des coûts. Napster a été le modèle le plus spectaculaire de la réussite technologique du P2P et de son échec économique. Les logiciels Peer-to-Peer actuels sont essentiellement axés sur le téléchargement, comme par exemple Emule ou ares [3].

4. Les caractéristiques du Peer-to-Peer

Les principales caractéristiques des différentes solutions proposées dans l'état de l'art sont [4]:

- La localisation des fichiers dans un environnement distribué
- Des métadonnées ou index du réseau P2P.

- La libre circulation des fichiers entre systèmes.
- Un mode de communication standard (TCP et HTTP).
- Des capacités de connexion variables suivant les modèles.
- Des échanges d'informations non sécurisés.
- Possibilité d'avoir des pairs non sûrs.
- Aucun Peer n'a de vue globale du système.

5. Les Objectifs du modèle Peer-to-Peer

Le modèle P2P étant très général, des applications de nature très différentes peuvent l'utiliser, les objectifs sont donc variés [5] :

- Partage et réduction des coûts entre les différents pairs.
- Fiabilité et passage à l'échelle, l'absence d'élément centralisé pour l'échange des données permet d'accroître la fiabilité en supprimant tout point central de panne et d'améliorer le passage à l'échelle en évitant les goulots d'étranglement.
- Agrégation des ressources et interopérabilité, en mettant en commun des ressources individuelles comme de la puissance de calcul ou de l'espace de stockage.
- Accroissement de l'autonomie en l'absence d'une autorité centrale, il est de la responsabilité de chacun de partager ou non des fichiers.
- Anonymat pouvant être assuré par certaines applications, en utilisant par exemple des algorithmes de routage qui rendent quasiment impossible le pistage d'une requête.
- Communication ad-hoc et collaborative.

6. Les Problèmes du Peer-to-Peer

Le modèle p2p pour fonctionner correctement doit répondre à un certain nombre de problèmes qui sont résumés dans ce qui suit [6] :

A. Premier problème : trouver l'adresse

Pour obtenir un fichier, il faut trouver une personne dont l'ordinateur contient le fichier recherché et qui accepte la connexion pour le partager. Le premier problème est donc de trouver « l'adresse IP » d'un ordinateur sur lequel le fichier recherché est accessible.

B. Deuxième problème : s'adapter aux caractéristiques de l'ADSL

Le développement de l'ADSL a posé un nouveau problème pour le fonctionnement des réseaux pair à pair. En effet, le débit fourni par les systèmes ADSL est asymétrique. Puisque les utilisateurs, qui sont les « clients », n'envoient que des requêtes de faible poids, le débit ascendant « upload » sera faible, tandis qu'à l'inverse, pour recevoir les grosses quantités d'informations que les serveurs envoient aux clients en guise de réponse, le trafic descendant « download » sera dix à vingt fois plus élevé. Cette asymétrie va à l'encontre du fonctionnement des réseaux pair à pair, puisque ce faible débit d'upload freine l'ensemble du réseau : aussi rapide que soit sa vitesse de connexion en download, un ordinateur ne peut pas recevoir un fichier à un débit plus rapide que celui imposé par l'ordinateur qui lui envoie les données !

C. Troisième problème : optimiser les protocoles

Le protocole d'un réseau englobe toutes les règles organisant les échanges de messages entre pairs. Toutes les activités sont ainsi encadrées : recherche des adresses IP, gestion et contrôle des fragments de fichiers échangés, etc.

Mais la mise en place de ces protocoles génère un trafic d'informations très abondant sur le réseau pair à pair : ainsi, dans le réseau Gnutella (version 0.4), le volume des données liées au protocole était aussi important que celui des fichiers échangés. À l'origine, le protocole HTTP (qui permet la navigation sur internet) a été optimisé. Le trafic lié à ce protocole ne représente que quelques pour cent du trafic des données. À l'heure actuelle, les protocoles pair-à-pair, eux, sont loin d'être optimisés. L'optimisation de ces protocoles est donc un thème crucial de recherche qui permettrait d'améliorer de façon considérable le fonctionnement global du réseau.

7. Pourquoi utiliser des applications P2P ?

- Très grande scalabilité : Jusqu'à des millions de pairs.
- Faible coût : Exploite les ressources existantes.
- Autonomie : Libre d'entrer et sortir, et le contrôle de ses propres données (confidentialité).
- Tolérance aux pannes : Pas de point singulier de panne.
- Disponibilité : Grâce à la redondance et la réplication de données

- Distribution de la charge : Exécution parallèle.
- Résistance : Fort contre les attaques de déni de service et les censures.
- Le Peer-to-Peer est pris en charge actuellement par Windows et est très facile à mettre en place en tant que réseau domestique. [7]

8. Le P2P est-il infaillible ?

Le Peer-to-Peer n'a pas que des avantages voici les inconvénients les plus pertinents [7]:

A. Les problèmes de comportement des utilisateurs

Les systèmes d'échanges de fichiers Peer-to-Peer peuvent facilement souffrir de l'anarchie générale. L'anonymat aidant, les membres sont parfois tentés d'avoir des comportements malveillants.

Sans un minimum de contrôle, on peut voir apparaître des activités comme la diffusion de **virus** ou le **freeloading**.

B. Les problèmes de comportement des providers et autres entreprises

Une atteinte à la vie privée ?

Notre vie privée n'est pas nécessairement protégée lorsque nous utilisons des outils Peer-to-Peer. En effet, les adresses IP des utilisateurs peuvent être récupérées lorsque le logiciel est centralisé. Lorsque celui-ci est décentralisé, les utilisateurs sont identifiables par leur fournisseur d'accès. Aux USA, un fournisseur d'accès n'a pas hésité à menacer tous ses clients qui utilisaient Gnutella de résiliation de contrat. Autrement dit, le FAI est allé regarder en détails ce que faisaient ses clients sur le Net.

Le projet Freenet, avec la cryptographie et la virtualisation du stockage paraît être le réseau le mieux armé pour la protection de la vie privée.

La pollution des réseaux

Sans même annoncer d'ultimatum, plusieurs entreprises proposent de polluer les réseaux d'échange de musique entre particuliers en y incorporant des fichiers de moindre qualité ou incorrects. L'objectif est de rendre ces réseaux gratuits et anarchiques moins attractifs.

La propagande

Nul n'est sans savoir que les systèmes Peer-to-Peer sont pour la plupart envahis par les bannières publicitaires même si à présent, des versions allégées des logiciels sont proposées sans publicités.

Mais la propagande va plus loin : l'installation de certains logiciels tels que Kazaa est accompagnée de l'installation de logiciels espions (spyware).

Officiellement, les informations recueillies par ces logiciels ne sont pas vendues mais on peut penser que les spywares listent les sites Web consultés par les utilisateurs pour mieux cibler les publicités qu'ils peuvent recevoir.

C. Le problème des droits d'auteurs

Napster doit autant sa popularité à son logiciel qu'à ses péripéties judiciaires. La société a été poursuivie en justice par les principaux majors de l'industrie musicale pour avoir incité ses utilisateurs à échanger des fichiers protégés par le copyright.

9. P2P versus client/serveur

Comme le montre le tableau suivant, il existe des différences significatives entre le modèle p2p et le modèle client/serveur, Il convient de préciser que la plupart des réseaux domestiques d'aujourd'hui sont de type Peer-to-Peer, car pour la plupart d'entre eux l'objectif est d'effectuer des tâches de base, mais importantes, telles que le partage d'une connexion Internet ou les jeux multi-joueurs, il n'y a aucune raison pour un utilisateur d'envisager un réseau client / serveur [8].



| Critère |  Modèle Client-serveur |  Modèle P2P |
|--------------------------------|--|---|
| Gestion | Supervisé | Auto-organisé |
| Présence | Permanente | Ad Hoc |
| Accès aux ressources | Recherche | Découverte |
| Organisation | Hiérarchique | Distribuée |
| Mobilité | Statique | Mobile |
| Disponibilité | Dépendante du serveur | Indépendante des pairs |
| Nommage | DNS | Indépendante |
| Modèle de programmation | RPC | Asynchrone |
| Sécurité | Attaques plus simples | Attaques difficiles |

Tableau 1.1 : Peer-to-Peer versus Client/serveur [8]

10. Les Champs d'application des réseaux P2P

Le modèle P2P n'est pas adapté pour toutes les utilisations, Nous allons citer quelques-unes de ses utilisations. On peut ainsi citer [9] :

Calcul distribué :

Consiste à utiliser les machines connectées à l'internet pour faire des petites portions d'un grand calcul, en exploitant les ressources (CPU, mémoire....) inutilisées des PC en réseau en vue d'accroître le potentiel réseau, comme exemple le projet :

- Seti@homeSearch for Extra Terrestrial Intelligence).
- Moteur de recherche
- Stockage en cluster
- Amazon DynamoDB

Le Partage de fichiers :

Le partage de fichiers constitue l'application la plus répandue actuellement du réseau P2P. Les internautes peuvent partager leurs fichiers et ainsi télécharger ces fichiers en ayant recours à des logiciels gratuits pour la plus part comme BitTorrent, Napster, Gnutella, Kazaa, eDonkey...

Le Partage privé :

Si un utilisateur voulait partager des fichiers volumineux (par exemple des photos de vacances) avec ses proches. Il devait jusqu'à présent les télécharger sur son site web où son hébergeur pouvait limiter son espace. Pour ce modèle, il est également assez difficile de restreindre l'accès aux photos à quelques utilisateurs uniquement. Ces fonctionnalités peuvent être assurées par des applications P2P telles que Qnext ou TribalWeb.

La Diffusion multiple :

Dans le cas où des fichiers doivent être diffusés auprès d'un grand nombre de personnes (par exemple diffusion d'une émission vidéo ou audio), une autre application P2P est adaptée : BitTorrent. Le protocole Bit Torrent découpe ainsi un fichier en plusieurs morceaux. Chaque utilisateur voulant télécharger le fichier va contacter le « tracker » pour obtenir la liste de toutes les machines ayant déjà téléchargé un des morceaux du fichier. Plus un fichier est téléchargé et plus il y a donc de sources disponibles.

L'inconvénient de Bit Torrent est que les fichiers ayant une certaine ancienneté ou une popularité faible vont avoir moins de sources voire même une disparition du réseau.

Le Streaming (P2P TV) :

Le streaming P2P est le fait de regarder en direct des flux produits et/ou relayés par d'autres pairs du réseau afin d'éviter ou du moins de diminuer la congestion qui pourrait se produire sur les serveurs de téléchargement, tout se déroule entre les personnes qui veulent accéder au fichier, Swarmplayer, qui permet de lire des vidéos en streaming en utilisant Bittorrent, s'annonce comme une vraie révolution dans le domaine.

La Communication et la collaboration :

Les applications P2P sont très adaptées pour toutes les applications nécessitant une communication entre pairs. Cette communication peut être audio (Skype) ou par simple message texte (chat et plateforme Jabber) ou permettre aux utilisateurs de jouer ensemble (jeux multi joueurs) ou même de travailler ensemble (plateforme de travail collaboratif GROOVE).

La Sauvegarde coopérative :

La sauvegarde croisée est une application intéressante du pair à pair qui consiste à sauvegarder ses données sur l'espace disque disponible des autres. Pour assurer un équilibre, le système doit procéder par échanges : si A stocke une quantité x pour B, alors il pourra aussi stocker x chez B. (Freenet, LeanOnMe).

11. Conclusion

Nous avons présenté dans ce chapitre les réseaux p2p qui sont basés sur l'égalité des nœuds d'un réseau, où tous les éléments du réseau jouent le rôle de client et serveur en même temps. Cette caractéristique de base a permis aux systèmes p2p d'être impliqués dans plusieurs types d'applications distribuées tel que le calcul distribué, la diffusion de contenu, le stockage distribué, l'échange et la communication. Les systèmes p2p souffrent cependant de plusieurs problèmes liés au réseau puis au comportement des utilisateurs. Des solutions ont vu le jour classées en deux grandes familles, qu'on abordera dans le chapitre suivant.

Chapitre 2: Les Architectures Peer to Peer

1. Introduction

Bénéficiant d'une notoriété se développant exponentiellement de par sa facilité d'accès et d'utilisation, le Peer-to-Peer représente la plus grande source de téléchargements dans le monde. Mais devant le grand nombre et la diversité des réseaux de partage, il n'est pas facile de s'y retrouver. Dans ce chapitre, état de l'art des solutions les plus pertinentes, nous allons présenter les principes de fonctionnement des réseaux d'échange de fichiers les plus populaires, ayant solutionné les problèmes cités dans le chapitre 1, ensuite un panorama non exhaustif des meilleurs logiciels d'échange est dressé.

Les réseaux Peer-to-Peer ont été classés en deux catégories : les p2p structurés et les p2p non structurés.

2. Les systèmes Peer-to-Peer non structurés

Pour surmonter le 1^{er} problème du chapitre 1, les réseaux pair-à-pair non structurés ont été conçus. Dans ce type de réseaux, chaque pair est indépendant et la propagation des requêtes se fait par inondation en interrogeant tous les pairs du réseau jusqu'à l'obtention du pair concerné. Les mécanismes sont simples et faciles à mettre en place, cependant les performances lors de la recherche d'un pair sur le réseau sont souvent limitées. Comme le montre la **Figure 2.1** ce modèle peut être classifié à son tour en trois sous catégories :

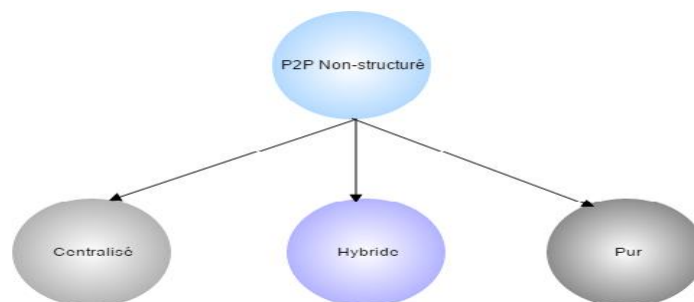


Figure 2.1 : Classification des systèmes P2P non structurés

A. Peer-to-Peer centralisé [6]

La première génération d'applications de partage de fichiers comprenait un serveur central qui servait d'index. Les exemples les plus connus sont Napster et Audiogalaxy.

a. Napster

L'appellation Napster est évocatrice d'un service pair-à-pair spécialisé dans le partage de fichiers son (au format MP3) qui a largement contribué à développer la technique P2P sur le marché grand public et même professionnel. Or, toute l'originalité de ce réseau, devenu désormais commercial, vient du fait qu'il a adopté une architecture centralisée.

Mais un tel mécanisme ne permet « que » le téléchargement de fichier « MP3 » pas les autres types de fichiers. Et donc il faut déguiser les fichiers avec WRAPSTER ou les renommer si besoin est de partager autres chose [11].

Ci-dessous un petit aperçu de l'interface graphique de Napster :

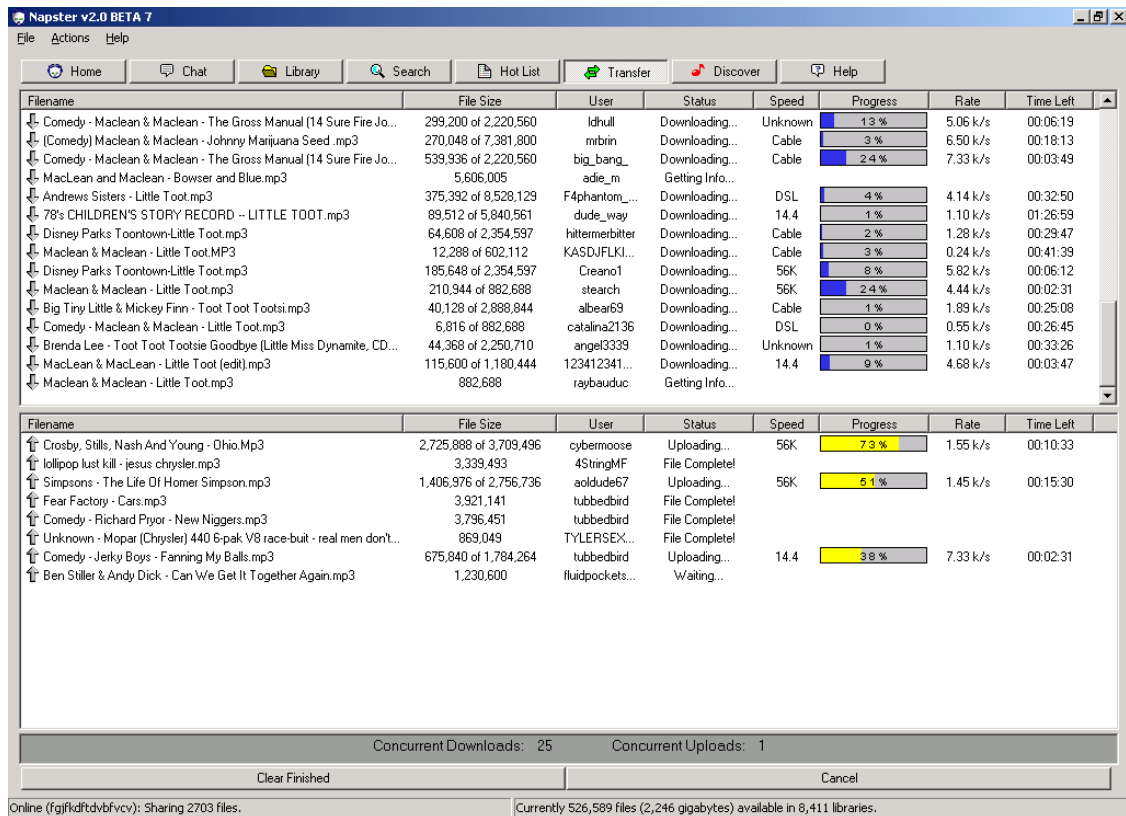


Figure 2.2 : Le logiciel Napster [6]

Dans ce type de système, quand un internaute veut rejoindre le réseau il doit s'enregistrer auprès du serveur central. Et Lorsqu'il souhaite partager un fichier, il le

déclare au serveur central. Celui-ci répertorie alors son adresse IP.

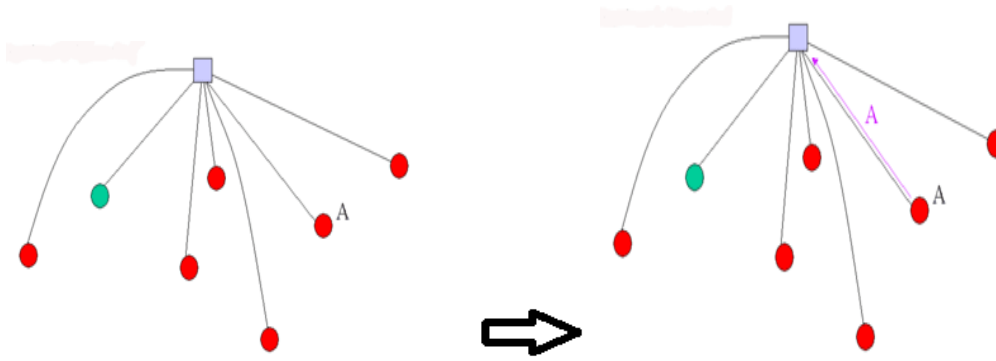


Figure 2.3 : Partage de fichiers Napster [6]

Tout autre internaute qui souhaite obtenir ce fichier interroge l'index central, comme il interrogerait un moteur de recherche.

L'index central lui communique une adresse IP.

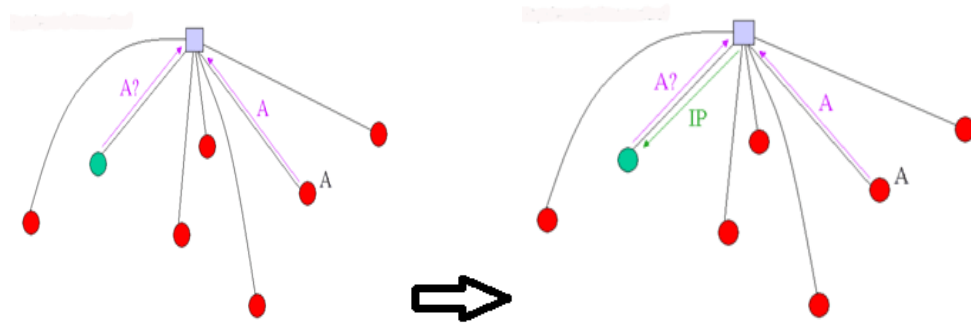


Figure 2.4 : Recherche de fichiers Napster [6]

Ensuite, le logiciel de partage lui permet de se connecter directement sur l'ordinateur qui propose le fichier.

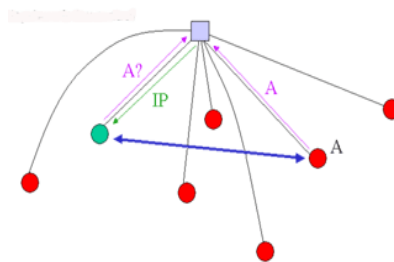


Figure 2.5 : Téléchargement de fichiers Napster [6]

Le fichier en lui-même ne transite pas par le serveur central, qui fonctionne simplement comme un annuaire : une fois l'adresse IP trouvée, les ordinateurs peuvent se connecter directement entre eux.

Mais il ne s'agit pas non plus d'un réseau totalement « pair à pair », dans la mesure où sans le serveur central, le réseau ne peut pas fonctionner.

i. Avantages [9]

- Simplicité : Pas de soucis de connexion au bon serveur.
- La recherche de document est facilitée : Le serveur maintient un index des ressources.
- Trafic réseau réduit : Les pairs ne communiquent entre eux que s'ils ont quelque chose à échanger.

ii. Inconvénients [9]

- Au niveau de la sécurité : une architecture P2P centralisée s'avère particulièrement vulnérable. Elle ne propose qu'une seule porte d'entrée, ce qui constitue le talon d'Achille de tout le réseau. Il suffirait effectivement de bloquer ce serveur pour déconnecter tous les utilisateurs et stopper le fonctionnement de l'ensemble du réseau.
- Autre chose, le fait de passer à travers une architecture centralisée, où il faut s'enregistrer pour pouvoir y accéder, ne garantit bien évidemment aucun anonymat. Le service connaît l'adresse IP de votre machine et le type de fichiers que vous téléchargez.
- Enfin, l'échange de fichiers numériques à grande échelle à travers Internet (musique, vidéo, photo, etc.) Entraîne bien souvent le non-respect des protections intellectuelles. Les œuvres placées sous copyright qui circulent sur les réseaux P2P sont légion et on se retrouve ainsi en face d'une gigantesque organisation de piratage.

-Remarque :

-Il est possible de mettre en place des clusters de serveurs ou alors de serveurs répliqués afin de mieux répartir les demandes de connexions et donc de limiter la chute de bande passante **Figure 2.6**. Chaque serveur peut avoir accès aux informations des clients connectés sur les autres.

L'accès aux données partagées est donc totalement transparent pour les utilisateurs [21].

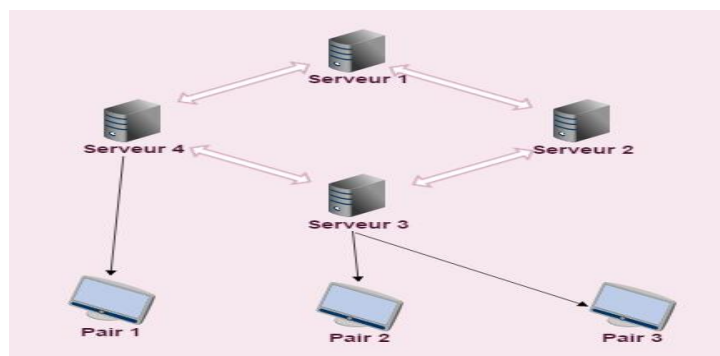


Figure 2.6 : Une amélioration du Peer-to-Peer centralisé

B. Peer-to-Peer décentralisé « purs »

Dans cette deuxième génération, le serveur central n'est plus nécessaire à la connexion. Chaque internaute appelé « Servent » indexe lui-même ses propres fichiers. Ceux qui sont à la recherche d'un fichier interrogent, de proche en proche, tous les ordinateurs du réseau. L'exemple le plus connu est le réseau Gnutella.

a. Gnutella

A la différence des applications de première génération, le réseau Gnutella (version 0.4) lui est un véritable réseau pair à pair, puisqu'on peut se passer totalement d'entité centrale [6].

i. Types de requête

| Types | Description | information |
|-----------------|--|---|
| Ping | Utilisé pour découvrir les autres servents sur le réseau. Un servent qui reçoit un Ping doit répondre avec un ou plusieurs Pong. | Vide. |
| Pong | Réponse à un Ping. | @IP+ N° de port (Informations sur les fichiers partagés et leurs tailles.) |
| Query | Message pour la recherche de fichier sur le réseau. Un servent qui reçoit un Query répondra par un QueryHit s'il trouve une correspondance avec un ou plusieurs des fichiers qu'il partage et les critères de recherche. | Bande passante minimum demandée (Critères de recherche.) |
| QueryHit | réponse à un Query si on possède le fichier. | @IP+N° de port +Bande passante (Nombre de réponses + descripteur) |
| Get | téléchargement des données. | |
| Push | Demande de téléchargement pour les pairs derrière un firewall. <u>Remarque :</u> Si les deux servents sont derrière un firewall, le téléchargement est impossible. | @IP du pair + index du fichier demandé (@IP+N° de port, ou envoyer le fichier.) |
| Bye | « Optionnel » informe qu'il va quitter le réseau, ainsi que les raisons qui le pousse à partir. | |

Tableau 2.1 : Les requêtes Gnutella [9]

ii. Fonctionnement de Gnutella

Un nouveau pair souhaite se connecter au réseau Gnutella

- Le premier problème à résoudre est de trouver, lors de la première connexion, d'autres internautes qui participent au partage.

Dès que l'on rencontre un ordinateur connecté au réseau, celui-ci peut communiquer les adresses IP d'autres participants. Pour trouver ce premier internaute, on utilise les « Gwebcache ». Ce sont des ordinateurs du réseau Gnutella qui fonctionnent également comme des serveurs web. Ainsi, ils sont répertoriés par les moteurs de recherche : il suffit donc de taper « Gwebcache » sur l'un de ses moteurs pour obtenir des listes d'ordinateurs connectés [6].

Gnutella2 GWebCaches
Updated 2013-06-24T17:04:00-05:00

| Score | URL | Version | IP Address | Hosts | Δ Age | URLs | Status |
|-------|---|----------------------|----------------|-------------|-------|-------------|---------------|
| 16.7 | http://dkac.trillinux.org/dkac/dkac.php | DKAC/Enticing-Enumon | 69.163.250.145 | 27/32 (84%) | 92 | 8/20 (40%) | Working |
| 12.4 | http://gwc.dyndns.info:28960/gwc.php | Cachechu 1.5 | 99.102.200.13 | 23/30 (77%) | 489 | 6/12 (50%) | Working |
| 12.2 | http://cache.ce3c.be/ | Cachechu 1.4 | 83.169.18.110 | 23/30 (77%) | 175 | 7/15 (47%) | Working |
| 11.0 | http://cache2.bazookanetworks.com/g2/bazooka.php | Bazooka 0.3.7b | 69.163.248.145 | 13/14 (93%) | 1158 | 0 | Low URL Count |
| 8.8 | http://htmlhell.com/ | Boa | 75.119.198.84 | 11/12 (92%) | 321 | 6/16 (38%) | Working |
| 7.7 | http://gweb.dwbo.nl/ | Cachechu 1.5 | 82.201.113.125 | 25/61 (41%) | 1386 | 10/17 (59%) | Working |
| 7.5 | http://cache.w3-hidden.cc/ | Cachechu 1.5 | 193.107.16.79 | 17/30 (57%) | 836 | 8/16 (50%) | Working |
| 6.8 | http://www.omhub.ru/mibew/view/ | Cachechu 1.5 | 193.169.52.69 | 24/57 (42%) | 1409 | 9/16 (56%) | Working |
| 5.6 | http://cache.hxdsecurity.com/gwc/ | Cachechu 1.5 | 87.106.101.7 | 15/30 (50%) | 655 | 1/13 (8%) | Working |

Figure 2.7 : Gwebcaches Gnutella

Chaque internaute ne peut se connecter qu'à un nombre limité d'autres participants. Ainsi, de proche en proche, se constitue ce qu'on appelle un « réseau logique ». C'est un « overlay network » qui se greffe sur le réseau global.

- Le nouveau pair émet un message Ping par Broadcast ou bien par l'intermédiaire d'une liste de pairs connus.
- P2 reçoit la requête Ping du nouveau pair.

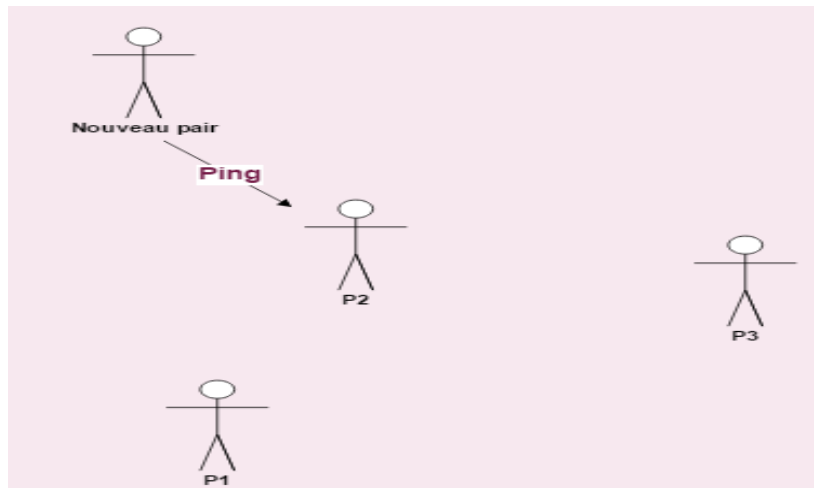


Figure 2.8 : Connexion d'un Peer Gnutella

- P2 répond par un Pong et transfère le Ping du nouveau pair à ses voisins « P1 et P3 ».

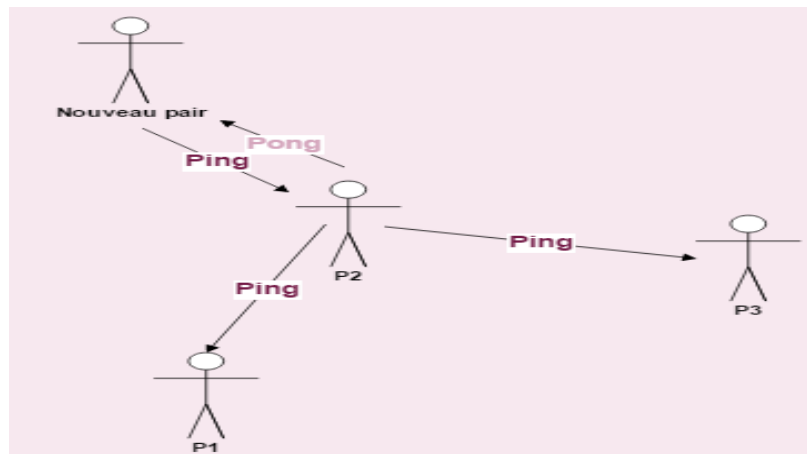


Figure 2.9: Routage des messages Ping Gnutella

- P1 et P3 transfèrent les Pong à P2 qui les transmet à son tour au nouveau pair.

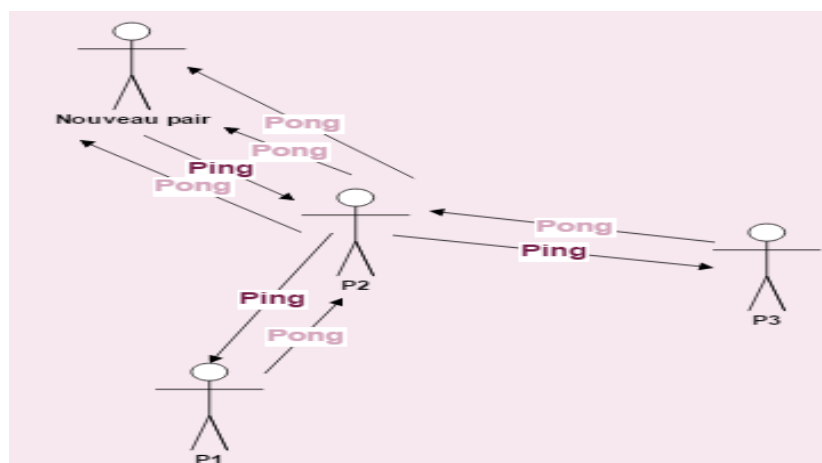


Figure 2.10 : Routage des messages Pongs Gnutella

Si le nombre maximum TTL (*Time To Live* en anglais) n'a pas été atteint alors P1 et P3 répéterons les mêmes opérations avec leurs voisins.

Après insertion le nouveau pair veut télécharger le fichier (Fichier.mp3)

- le nouveau pair envoie la requête QUERY à ses voisins et donc a (P2).

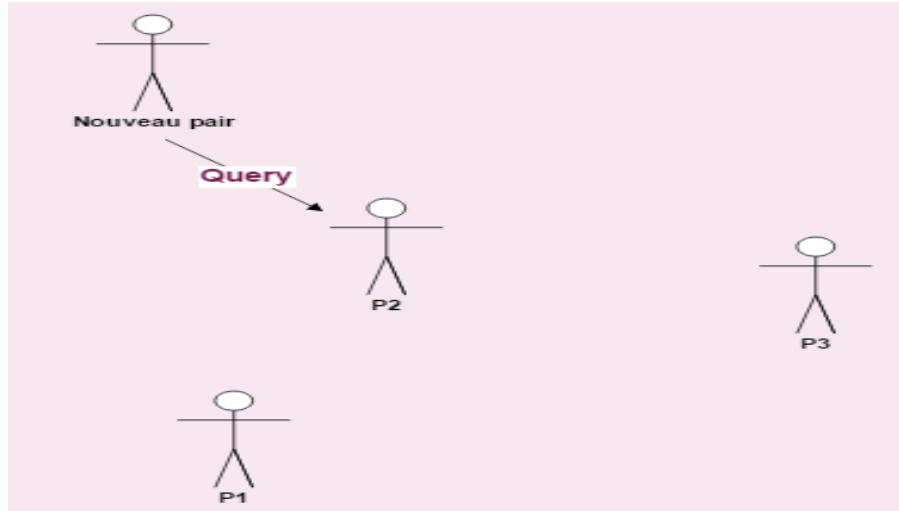


Figure 2.11 : Recherche de données (Query) Gnutella

- P2 ne détient pas le fichier alors il diffuse la requête Query a ces voisins (P1 et P3).

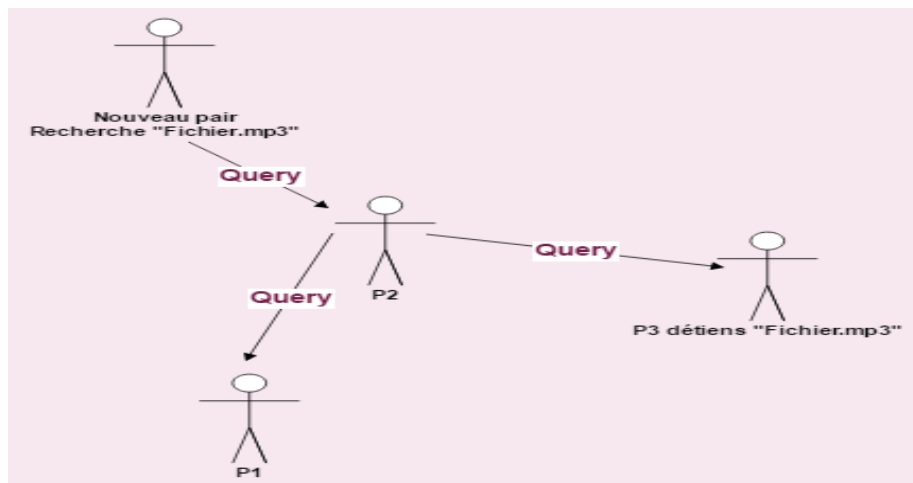


Figure 2.12 : Routage des messages Query Gnutella

- P3 détient le fichier alors il envoie la requête QueryHit à son voisin qui la fait suivre au nouveau pair.

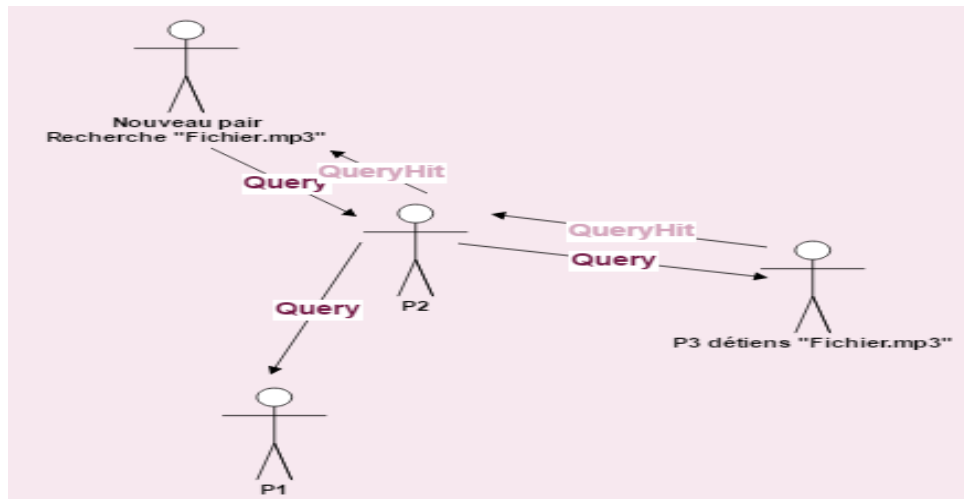


Figure 2.13 : Routage des messages QueryHit Gnutella

- Le nouveau pair contacte P3 et le téléchargement peut s'effectuer.

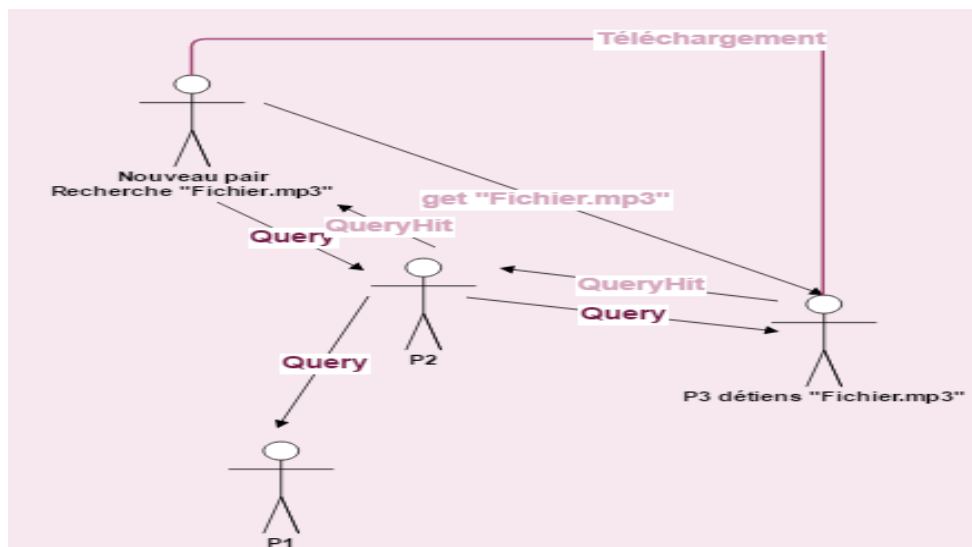


Figure 2.14 : Echange de fichiers Gnutella

- ✓ En pratique, quand on recherche un fichier, c'est sur ce réseau logique qu'on diffuse sa requête. Celle-ci est d'abord adressée aux ordinateurs voisins, puis peu à peu propagée jusqu'à trouver un ordinateur possédant le fichier en question. Un champ TTL (*Time To Live* en anglais) est associé au message de recherche pour comptabiliser le nombre de retransmissions restantes.
- ✓ La réponse, constituée par l'adresse IP de cet ordinateur, suit le même chemin en sens inverse. Ce système garantit donc une certaine forme d'anonymat : celui qui détient le fichier ne connaît pas l'identité de celui qui le lui demande.

Le nouveau pair souhaite quitter le réseau

- Le logiciel client enregistre la liste des voisins du nouveau pair.

Le nouveau pair quitte involontairement le réseau

- le réseau est divisé en plusieurs parties.

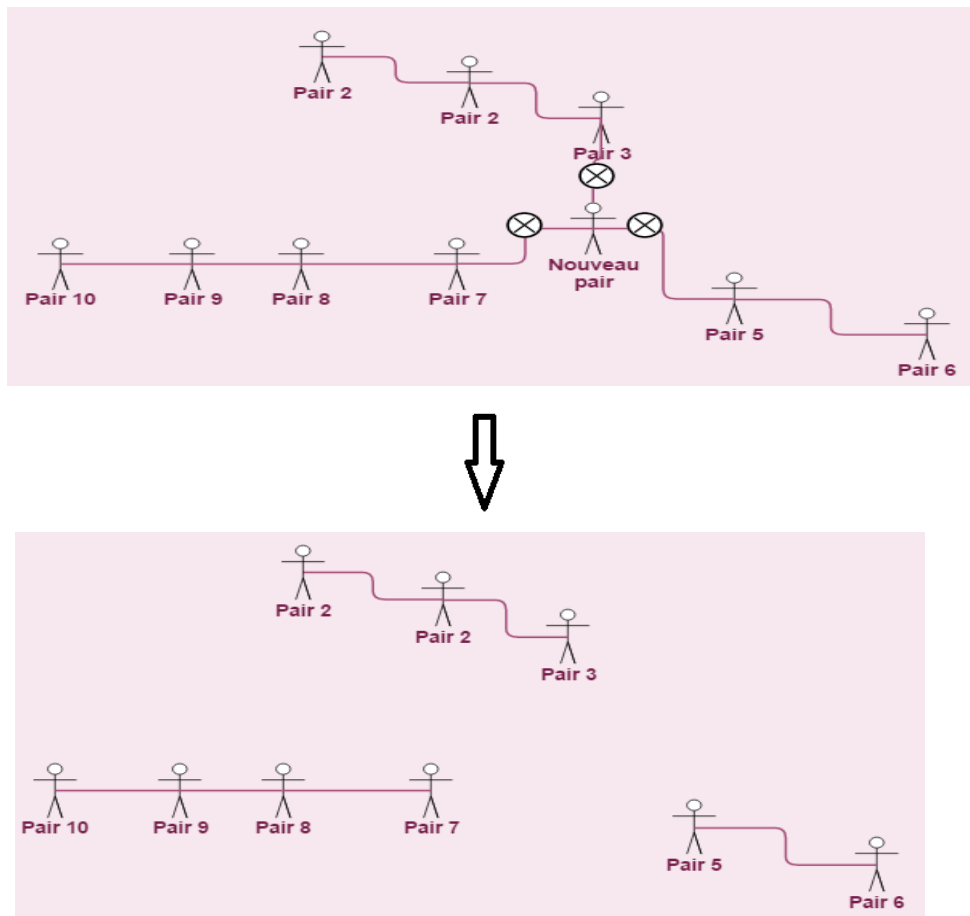


Figure 2.15 : Panne d'un Peer

- ✓ Cette deuxième génération a bien vite déperî, car le réseau était constitué à la fois d'ordinateurs connectés en haut débit et d'ordinateurs connectés par modem. Du coup, ces derniers imposaient leur vitesse de connexion à l'ensemble du réseau, et la propagation des données se trouvait fortement ralentie.

iii. Avantages [4]

- La taille d'un tel réseau est théoriquement infinie. IL n'y a pas de contraintes sur les ressources d'un serveur central.
- Anonymat.
- Tolérance aux pannes (grand nombre de nœuds pouvant répliquer les mêmes données)
- Adaptabilité (connexion et déconnexion des pairs sans conséquences)

iv. Inconvénients [4]

- Réseau rapidement inondé par des Ping et des Pong.
- Supporte mal une forte montée en charge du nombre d'utilisateurs.

- Message Push continuellement envoyé si pas de réponse.
- Gros consommateur de Bande passante.
- Pas de garantie de succès, ni d'estimation de la durée des requêtes.
- Pas de sécurité, ni de réputation (pas de notion de qualité des pairs, ni des données fournies).
- Son mécanisme de recherche présente une limite, fixée par la valeur initiale du TTL. Ainsi une requête peut être stoppée par une expiration de TTL sans avoir parcouru l'intégralité du réseau et retourner une réponse négative. Seulement 25% des requêtes aboutissent.

C. P2P hybrides

Cette troisième génération opère une distinction entre deux niveaux de pairs : ceux qui ont une connexion haut débit et ceux qui ont une connexion par modem.

Les ordinateurs disposant d'une connexion par modem se relient à un ordinateur ayant une connexion haut débit, ce dernier devient dès lors un « super pair ». Chaque super pair indexe alors les fichiers des pairs bas débits qui lui sont rattachés, comme le faisait autrefois le serveur central des réseaux de première génération [6].

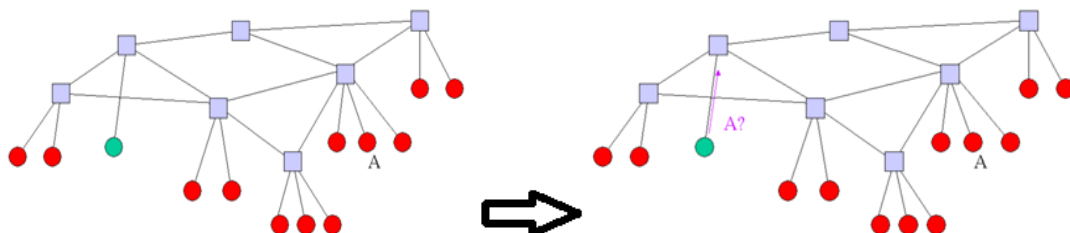


Figure 2.16 : Le modèle hybride [6]

Entre deux supers pairs en revanche, le système continue à fonctionner comme les réseaux de seconde génération. Mais la propagation des données est plus rapide, puisqu'elle n'utilise plus que les connexions haut débit.

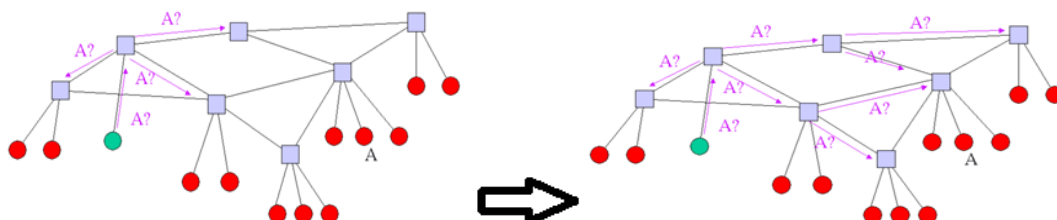


Figure 2.17 : Recherche de fichiers dans le modèle hybride [6]

Une fois l'adresse IP communiquée à l'ordinateur d'origine, une connexion directe s'établit entre les deux pairs, quel que soit leur niveau.

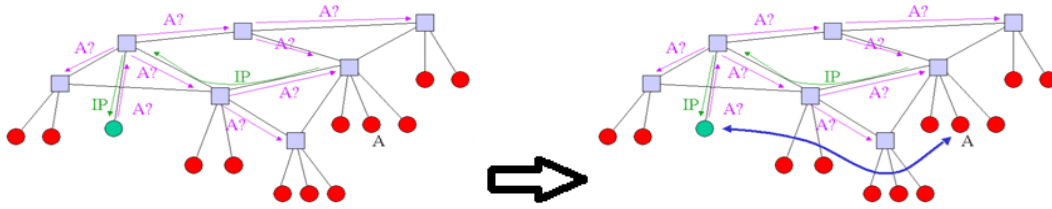


Figure 2.18 : Téléchargement de fichiers dans le modèle hybride [6]

Il s'agit donc d'une solution hybride entre les réseaux de première et de seconde génération. Le réseau FastTrack, associé au logiciel de partage Kazaa, constitue un bon exemple de réseau pair à pair de troisième génération.

- ✓ Pour s'adapter à l'environnement hostile et au trafic asymétrique cité dans le second problème du chapitre 1, le système non structuré représenté par excellence par eDonkey introduit le téléchargement multiple.

D. Le téléchargement multiple (réseau eDonkey et logiciels de partage eDonkey, eMule, mlDonkey) [6]

Pour compenser le caractère asymétrique des connexions internet, des internautes ont eu l'idée de rendre possible le téléchargement à partir de plusieurs fichiers, et donc de plusieurs ordinateurs. Le fichier est « découpé » en petites parties et peut ensuite être téléchargé morceau par morceau à partir de plusieurs pairs.

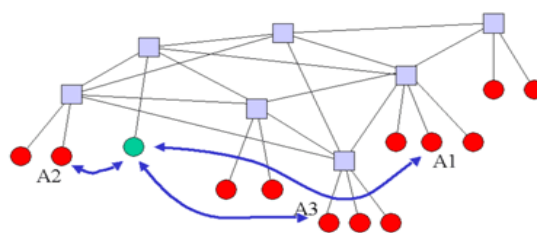


Figure 2.19 : Le téléchargement multiple [6]

Le trafic descendant se nourrit donc de plusieurs trafics ascendants. Évidemment, cette solution n'est que partielle : elle ne peut pas permettre à tous les ordinateurs d'augmenter en même temps leur vitesse de téléchargement, puisqu'au total, les capacités cumulées de download et d'upload du réseau restent déséquilibrées. Par ailleurs cette fragmentation en tranches favorise le partage partiel du fichier, un coup de pouce pour la coopération.

a. Développer la coopération [6]

En quoi consiste la coopération ? Pendant qu'un internaute est en train de télécharger un fichier, son ordinateur peut immédiatement redistribuer les fragments déjà obtenus à d'autres utilisateurs : les différents clients s'entraident donc automatiquement. Par contre, lorsqu'un internaute a fini de télécharger un fichier, il a souvent tendance à l'effacer du dossier partagé ou bien à se déconnecter du réseau pair à pair.

Ce comportement est doublement préjudiciable pour le fonctionnement du réseau : d'une part, c'est une source de téléchargement en moins pour les autres utilisateurs désireux d'acquérir ce fichier, d'autre part, c'est autant de capacité d'upload indisponible pour compenser l'asymétrie. Pour inciter à la coopération, les réseaux pair à pair ont donc mis en place un système de points, qui marchent sur le modèle du bonus/malus : les données fournies en upload permettent d'acquérir des points donnant droit à des capacités de download supplémentaires.

Le protocole BitTorrent, présenté un peu plus loin, propose une solution élégante au problème en utilisant le principe d'« un prêté pour un rendu » : il faut redonner des morceaux du fichier pour pouvoir en acquérir de nouveaux plus rapidement.

b. Gérer un réseau dynamique [6]

L'un des enjeux fondamentaux des réseaux pair à pair, c'est de gérer la « dynamicité » du réseau. En effet, il ne faut pas oublier que le réseau n'est pas figé : à chaque instant, de nouveaux utilisateurs font leur apparition, tandis que d'autres se déconnectent.

Avant les téléchargements multiples, les échanges se faisaient uniquement entre deux pairs : l'inconvénient résidait dans le fait que si le donneur se déconnectait en cours de transfert, le demandeur devait trouver une autre source et recommencer tout le téléchargement depuis le début.

Grâce au système des donneurs multiples, les échanges sont devenus bien plus ergonomiques, puisque de nouveaux pairs peuvent prendre le relais des pairs qui se déconnectent. Un fichier peut ainsi rester disponible dans le réseau alors même que plus personne ne le propose dans son intégralité ! Le téléchargement multiple induit ainsi une persistance des fichiers dans le temps, ce qui est extrêmement utile pour le partage.

E. Le logiciel Freenet, la liberté à l'état pur [10]

Nom pour le moins révélateur, Freenet caractérise le réseau fer de lance pour la liberté d'expression et d'information « totale » en s'appuyant sur l'architecture décentralisée

axée sur l'anonymat. Aucune donnée n'est transmise directement. Elle est cryptée afin de dissimuler son contenu (seul le pair téléchargeant le fichier pourra lire le contenu) et transite par différents intermédiaires, histoire de compliquer le plus possible l'identification des utilisateurs, le risque zéro n'existant pas. Chaque donnée, cryptée par une clé unique est stockée sur chaque pair dans un cache local, dont il faudra définir sa taille en fonction de la capacité du disque dur du pair concerné. Lors de la recherche d'un fichier, une requête est adressée aux pairs ainsi qu'une requête pour la clé de décryptage.

En contrepartie, l'exigence en bande passante est assez élevée. Ce qui limite davantage le champ d'utilisation de Freenet aux fichiers peu volumineux comme les morceaux mp3, les documents et les images, il permet également d'héberger et de publier des sites Internet, de communiquer instantanément et offre également la gestion du courrier électronique. Un moyen de préserver une grande liberté et de lutter contre la censure. C'est certainement ce qui explique l'engouement de la communauté d'eMule en ce qui concerne le réseau Freenet, puisque le plugin FreeMule permet d'associer la meilleure des techniques d'anonymat implémentées dans le réseau Freenet avec le réseau eDonkey. Et les deux réseaux, comme tous les différents réseaux p2p (gnutella, BitTorrent, eDonkey, FastTrack, etc.) restent incompatibles entre eux...

a. Avantages

- Réplication intrinsèque.
- Autonomie de stockage.
- Autonomie partielle du routage (dépendance entre clés stockées et requêtes traitées).

F. Le logiciel BitTorrent [6]

Le logiciel BitTorrent, est l'une des solutions au 3^{ème} problème cité dans le chap1. Il se concentre sur le téléchargement des fichiers. L'indexation des fichiers est prise en charge par un autre protocole. Il reprend le principe du téléchargement multiple, mais il permet également de développer la coopération en utilisant le principe «d'un prêté pour un rendu » : pour cela, un algorithme d'incitation à la coopération inspiré de la théorie des jeux est utilisé. Chaque pair BitTorrent, en tant que serveur, donne la priorité aux clients qui lui ont récemment envoyé des morceaux du fichier. Ainsi un pair, en tant que client, a intérêt à donner ce qu'il possède pour recevoir le plus souvent possible des morceaux de fichier.

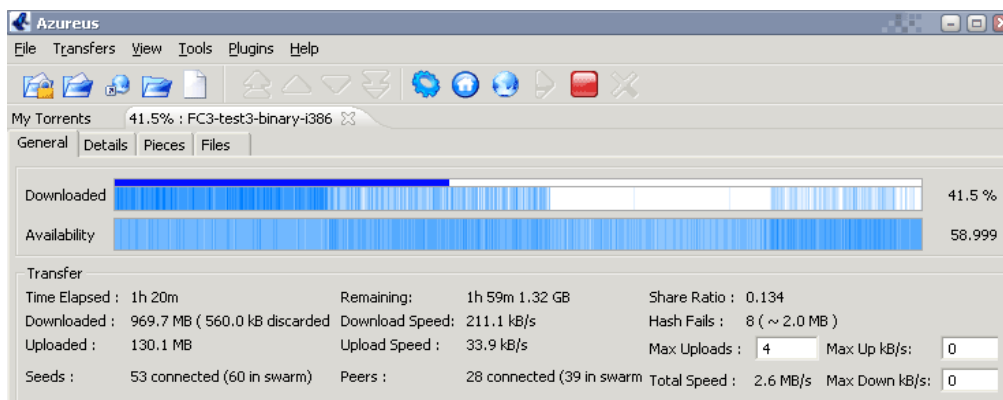


Figure 2.20 : Téléchargement avec BitTorrent [6]

À l'instar du réseau eDonkey, BitTorrent utilise la capacité «d'upload » des ordinateurs connectés pour redistribuer le fichier et ainsi désengorger le serveur. Des fichiers de taille importante peuvent donc être distribués par de petits serveurs.

Le logiciel BitTorrent est notamment utilisé pour distribuer certaines versions de Linux. Certaines sociétés éditrices de jeux l'utilisent également pour la diffusion de leurs logiciels et de leurs versions d'essai.

3. Bilan des systèmes P2P non structurés

| | |
|----------|---|
| Napster | -Type de fichier : MP3 -Anonymat : ✗ - Cout de la recherche : O(1) |
| Gnutella | -Type de fichier : tout type de fichier -Anonymat : ✗ - Cout de la recherche : O(N) |
| FreeNet | -Type de recherche : égalité -Anonymat : ✓ -Coût de recherche et insertion : O (Log(n)) |

Tableau 2.2 : Bilan des systèmes P2P non structurés

D'après le tableau ci-dessus, on voit clairement que Gnutella est le plus simple, en étant efficace, donc le rapport qualité/complexité est le meilleur des trois, par contre Freenet est plus complexe, mais redouble d'efficacité puisqu'il gère le téléchargement fragmenté d'un même fichier depuis plusieurs pairs, pour augmenter le débit.

4. Les systèmes Peer-to-Peer Structurés

Pour résoudre les problèmes qui se sont posés dans les systèmes présentés précédemment, les systèmes pairs à pairs structurés ont fait leur apparition.

L'idée est de pouvoir rechercher de façon efficace les données et d'être résistant aux défaillances et à la volatilité des pairs.

Ces systèmes fournissent un routage efficace d'un point à un autre du système (Nul besoin d'inonder le système), et sont basés sur une notion de connaissance locale : un nœud n'a pas de connaissance globale, mais il peut se rapprocher de la donnée recherchée.

Le principe est d'utiliser une fonction de hachage « h », et ensuite à stocker, de façon totalement distribuée, l'information (Nom, adresse IP) pour chaque ressource, qui associe au nom d'une ressource l'adresse IP du pair qui possède la ressource.

La plupart des identifiants de nœuds ne correspondent pas à des nœuds réels. On définit donc une fonction successeur(k) qui retourne l'identifiant du premier nœud réel $\geq k$ modulo 2^m .

L'information sur la ressource nom est alors stockée sur le nœud d'identifiant successeur(h(nom)). L'index est ainsi globalement réparti aléatoirement sur plusieurs nœuds.

La topologie adoptée change d'un système P2P à un autre. Par exemple, CAN se base sur « un espace en N dimensions », Chord « un anneau », et Tapestry a une topologie dédiée. L'idée de la topologie est d'avoir

- un faible degré pour une mise à jour rapide malgré la volatilité des nœuds
- un faible diamètre pour une recherche performante

Il faut donc trouver un compromis, et Chord obtient à la fois un degré et un diamètre en $\log(N)$ comme nous allons voir [23].

A. Table de Hachage Distribuée (DHT Distributed hash table)

Le principe des tables de hachage distribuées consiste à utiliser une fonction de hachage « h », que l'on peut appliquer à la fois aux adresses IP des pairs et aux chaînes ASCII identifiant une ressource disponible sur le réseau. Le résultat est un nombre aléatoire codé sur « m bits ». « m » est choisi suffisamment grand pour que la probabilité d'une collision soit suffisamment petite. Typiquement $m = 160$, et on dispose donc de 2^{160} identificateurs [23].

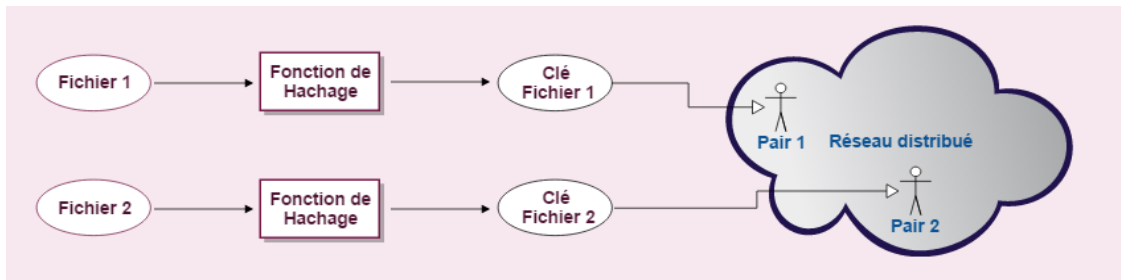


Figure 2.21 : Table de hachage distribuée (DHT)

B. Le système chord

Chord utilise une topologie en anneau. Il a pour particularité de disposer d'algorithmes d'une complexité d'au plus $O(\log(N))$ requêtes pour trouver une information dans un anneau de N éléments en utilisant le principe des tables de hachage distribuées. Chaque pair maintient une liste de successeurs [12].

Remarque : [14]

L'ajout d'un nouveau nœud repose sur la combinaison d'opérations élémentaires

-Nœud Bootstrapping :

Un nœud d'amorçage, également connu en tant qu'hôte rendez - vous, est un nœud dans un réseau de recouvrement qui fournit des informations de configuration initiale aux nouveaux nœuds afin qu'ils puissent joindre avec succès le réseau de recouvrement [13].

– **N.join (N')** : Le nœud N annonce au nœud N' qu'il rentre dans le réseau et lui demande de lui fournir son successeur.

– **N.stabilize ()** : lancé périodiquement, permet à N et à son successeur de vérifier qu'ils forment un couple correct (il n'y a pas de nouveaux nœuds entre les 2).

– **N.maj-entrées ()** : lancé périodiquement, permet d'initialiser la table de routage pour les nouveaux nœuds ou de la mettre à jour pour les nœuds existants.

– **N.test-predecesseur ()** : lancé périodiquement, vérifie que le prédécesseur est toujours là.

– La fonction de hachage utilisée est : SHA-1. L'identifiant d'un nœud est obtenu en hachant l'adresse IP, et la clé d'une ressource est obtenue en hachant son nom [23].

a. Fonctionnement de Chord

-Un nouveau pair souhaite se connecter au réseau Chord dont la capacité des pairs supportés est « 64 ».

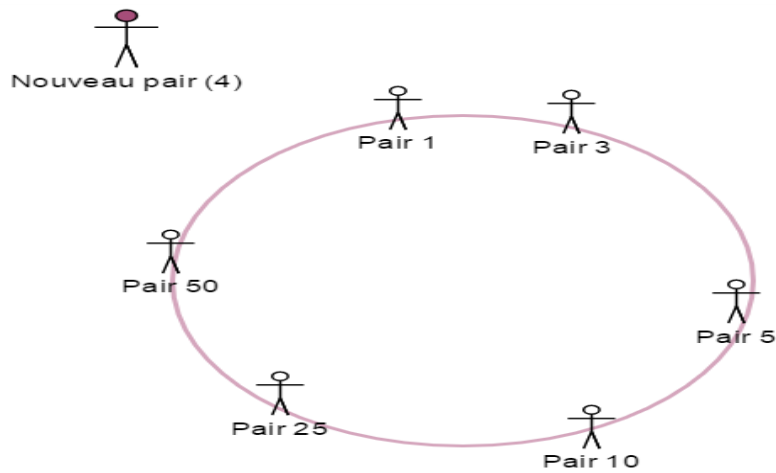


Figure 2.22 : Bootstrap chord

- Le nouveau pair utilise le Bootstrap pour trouver son successeur « Pair 5 » et lui envoie la requête JOIN.

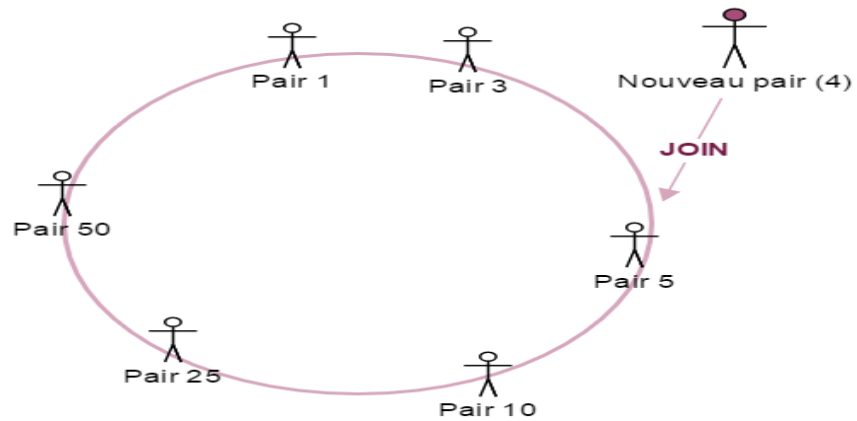


Figure 2.23 : Connexion d'un peer chord

- Le nouveau pair prend place entre « Pair 1-Pair 5 » puis exécute la fonction STABILIZE ().

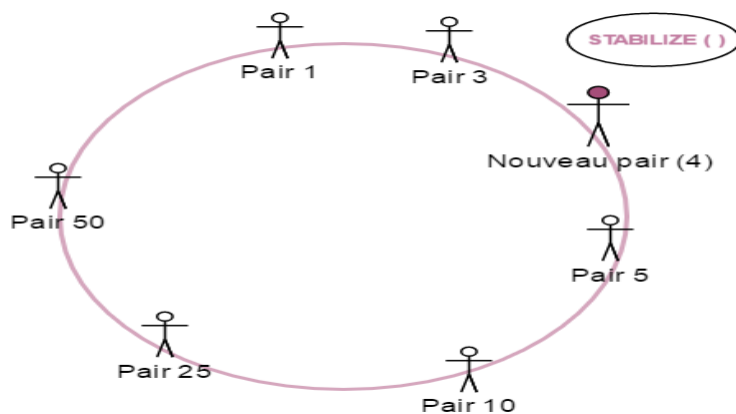


Figure 2.24 : Stabilisation d'un peer Chord

-Après insertion, le nouveau pair veut télécharger le fichier (Fichier.mp3) qui se trouve

en possession du Pair 3 :

- le nouveau pair crée sa table de routage

Table de Routage du nouveau pair

| | |
|-------------------------------|-------------------------------------|
| <i>Pair 4 + 2⁰</i> | 4+1=5 <10 → pair successeur= Pair10 |
| <i>Pair 4 + 2¹</i> | 4+2=6 → Pair10 |
| <i>Pair 4 + 2²</i> | 4+4=8 → Pair10 |
| <i>Pair 4 + 2³</i> | 4+8=12 → Pair25 |
| <i>Pair 4 + 2⁴</i> | 4+16=20 → Pair25 |
| <i>Pair 4 + 2⁵</i> | 4+32=36 → Pair50 |

Tableau 2.3 : Table de routage du nouveau nœud

- le Pair 50 est le pair le plus proche de la clé et qui ne la dépasse pas donc ce dernier crée sa table de routage de la même manière.

Table de Routage du pair 50

| | |
|--------------------------------|--|
| <i>Pair 50 + 2⁰</i> | Pair 1 |
| <i>Pair 50 + 2¹</i> | Pair 1 |
| <i>Pair 50 + 2²</i> | Pair 1 |
| <i>Pair 50 + 2³</i> | Pair 1 |
| <i>Pair 50 + 2⁴</i> | 50+16=66 > 2 ⁶ → 66 mod 64 = 2 (Pair 3) |
| <i>Pair 50 + 2⁵</i> | 50+32=82 → 82 mod 64 = 18 (Pair 25) |

Tableau 2.4 : Table de routage du nœud 50

- Le Pair 1 est le pair le plus proche du fichier et qui ne le dépasse pas alors il vérifie si son successeur possède le fichier, et dans notre cas « c'est oui ». Et ainsi le téléchargement pourra s'effectuer.

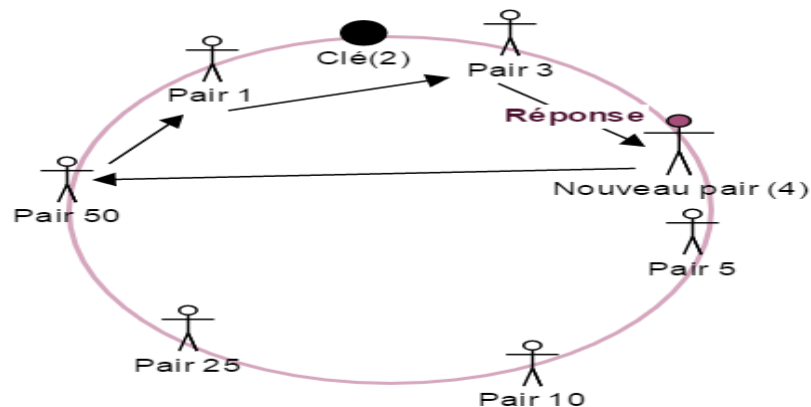


Figure 2.25 : Téléchargement de fichiers Chord

-Le nouveau pair veut quitter le réseau :

- il remet ses clés à son successeur et il informe également son prédécesseur pour que celui-ci puisse changer son pointeur successeur. Le processus de stabilisation met à jour les tables de repérage comme pour le cas de l'ajout [19].

-Le nouveau pair tombe en panne :

- les pairs contenant le « nouveau pair » dans leur table de repérage doivent pouvoir retrouver le successeur de celui-ci. Notamment, le prédécesseur du « nouveau pair » n'a plus de successeur valide.

L'idée consiste à conserver, en plus de son successeur immédiat, une liste des « r » successeurs directs. Cela permet à un pair de sauter par-dessus « r - 1 » pairs contigus défectueux et de continuer sa participation sur le cercle.

Remarque :

La probabilité que « r » pairs soient simultanément en panne est « p^r », « p » étant la probabilité qu'un pair soit en panne.

C. Le système Can (Content Adressable Network) [12]

CAN est un protocole de réseau pair-à-pair structuré s'appuyant sur une topologie à N dimensions. Les ressources sont réparties dans une zone multidimensionnelle, délimitée par des coordonnées. Chaque pair s'occupe d'une zone. Supposons que nous sommes dans un repère en deux dimensions, alors chaque pair s'occupe de gérer une zone définie par deux coordonnées minimales et maximales tout en maintenant à jour la liste de ses voisins (nord, est, sud, ouest). Cette zone contient les ressources que le pair partage. Un exemple de réseau CAN en deux dimensions est illustré sur la **Figure 2.26**.

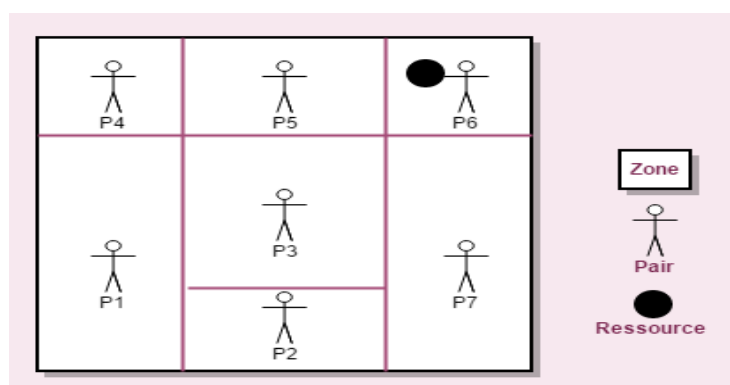


Figure 2.26 : Protocole Can

a. Fonctionnement de Can

-Un nouveau pair veut rejoindre le réseau Can :

- A l'aide du bootstrap, le Nouveau pair trouve un pair déjà existant sur le réseau.

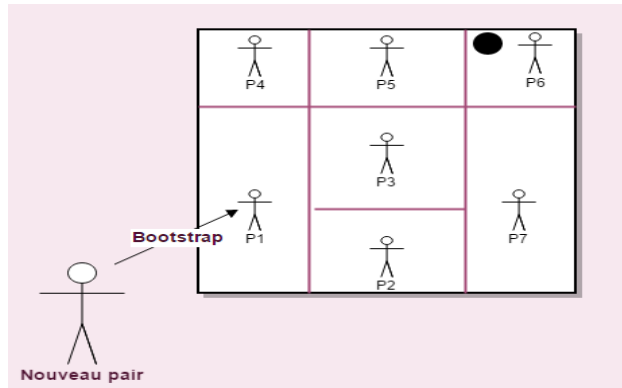


Figure 2.27 : BootStrap Can

- le Nouveau pair choisi des coordonnées (x, y) aléatoirement.
- Par le mécanisme de routage, le Nouveau pair trouve le pair qui gère ces coordonnées sur le réseau.

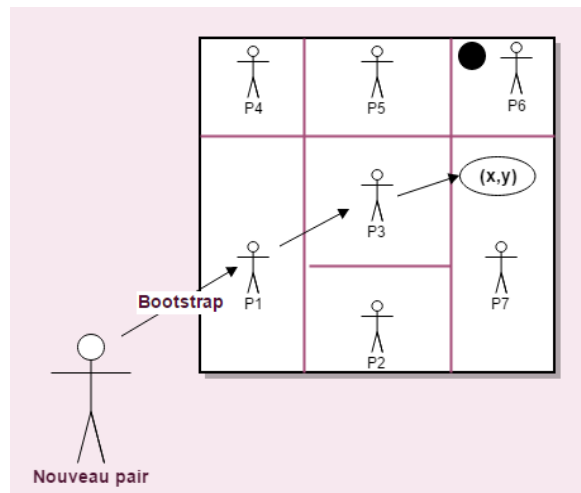


Figure 2.28 : Routage des messages Can

- La zone contenant ces coordonnées est scindée en 2.

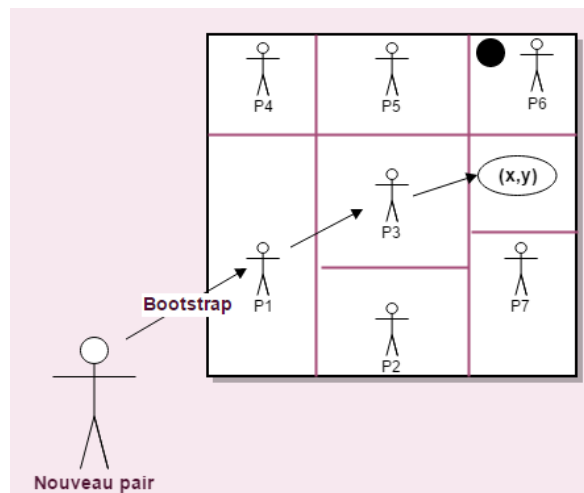


Figure 2.29: Connexion d'un Peer Can

- Distribution des ressources et mise à jour des voisins.

- Les voisins du pair dont la zone va être scindée doivent être prévenus de l'arrivée du nouveau pair.

-P1 cherche une ressource :

- P1 parcourt les différentes dimensions : en partant de la première, si les coordonnées sont présentes sur cette dimension il s'arrête là.

Sinon le message est routé vers le pair voisin se trouvant le plus proche de la coordonnée cherchée sur l'axe courant.

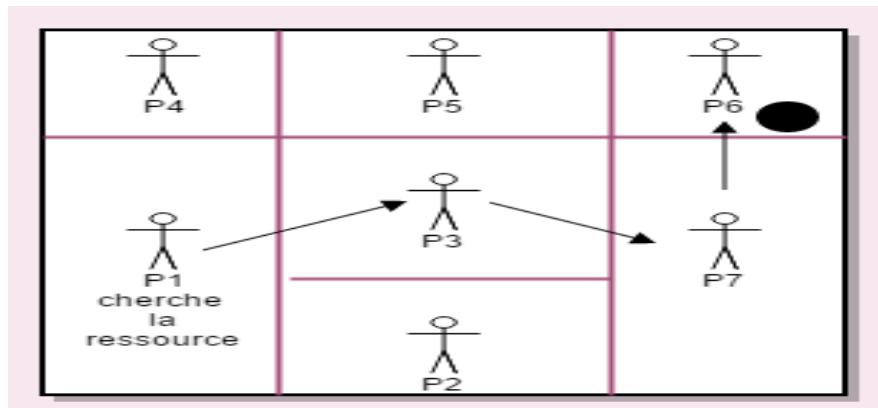


Figure 2.30 : Recherche de fichier Can

-Si un pair « P » veut quitter proprement le réseau, sans perte d'informations :

- P notifie ses voisins de son départ.
- Sa zone est fusionnée avec le pair dont il est issu lors de la dernière de zone.
- Les ressources de P sont maintenant sous la responsabilité du nouveau propriétaire de la zone de P.

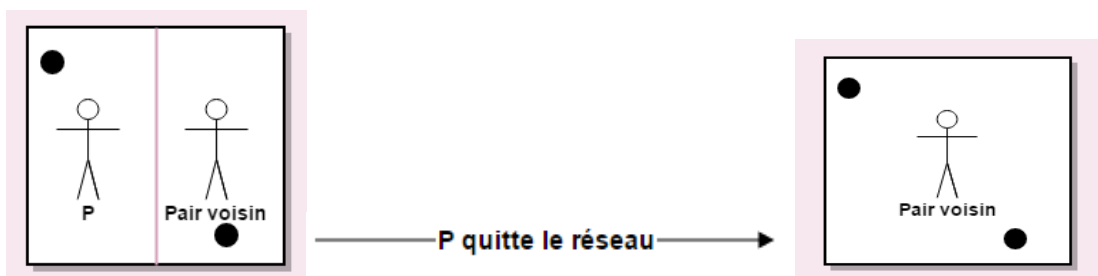


Figure 2.31 : Départ d'un Peer Can

-Si un nœud subi un dysfonctionnement, et quitte le réseau sans préavis, le message périodique de mise à jour découvrira ce départ ou cette panne et le mécanisme de prise de contrôle sera automatiquement déclenché par les nœuds qui découvrent la panne.

Un compte à rebours sera lancé, quand le délai d'un des nœuds expirera, ce dernier enverra un message TAKEOVER avec des informations sur sa propre zone à tous les voisins du nœud défaillant.

Le nœud qui reçoit ce message annule automatiquement son propre compte à rebours si

le volume de sa propre zone est supérieur au volume de la zone contenu dans le message.

D. Le système Pastry [15]

Pastry est un protocole de routage et de localisation dans un réseau P2P basé sur une table de hachage distribuée. Chaque nœud membre possède un identifiant unique de 128 bits qui est le fruit d'une fonction de hachage appliquée par exemple à son adresse IP. Les deux opérations de base définies dans Pastry sont :

- **store (key, value)** : permet de sauvegarder la ressource « value » dans le pair dont l'identifiant est le plus proche numériquement de sa clé « key ».
- **look-up (key)** : permet de récupérer la ressource associée à la clé « key ». Cette requête sera routée vers le pair dont l'identifiant est le plus proche numériquement de la clé « Key ». Soit ce dernier sauvegarde réellement la ressource demandée qui sera donc transmise directement au nœud-client, soit il contient les paramètres de localisation (l'adresse IP et le numéro de port) de son hébergeur.

La table d'état

Chaque pair dans Pastry maintient une table d'état contenant trois catégories d'entrées.

-Dans la **première catégorie**, on trouve l'ensemble des voisins virtuels du pair considéré en termes de distance numérique entre les identifiants des nœuds. Cet ensemble est subdivisé en deux sous-parties : l'une contenant les identifiants numériquement plus petits, l'autre, les identifiants numériquement plus grands.

-La **seconde catégorie** est la table de routage du pair. Cette table est un tableau bidimensionnel dont les colonnes représentent un des éléments de la base choisie (de 0 à F pour l'hexadécimale) et les lignes, la taille du préfixe commun entre l'identifiant courant et celui considéré. Plus formellement, la cellule définie par $(l=i, c= j)$ contient l'identifiant d'un pair possédant un préfixe commun de i digits exactement avec l'identifiant de référence et dont le $(i+1)$ ème digit est égal au (j) ème digit de l'alphabet utilisé. Dans le cas où il n'y a pas un identifiant accomplissant ces deux conditions, la cellule considérée reste vide.

-La **troisième catégorie** de la table d'état contient l'ensemble des voisins réels du nœud. La métrique choisie pour évaluer la distance réelle entre pairs est le nombre de sauts IP.

La **Figure 2.32** présente un exemple simplifié de table d'état d'un nœud Pastry. Dans cet exemple, la longueur d'un identifiant est de 16 bits représentés en base 4 (les digits

permis sont 0, 1, 2 et 3). Par exemple, la deuxième colonne de la cinquième ligne (soit $l = 4$) possède une entrée pour un pair possédant un préfixe commun de 4 digits, et un cinquième digit égal à 1.

| Pair 10233102 | | | |
|------------------|----------|----------|----------|
| Voisins virtuels | | | |
| 10233033 | 10233021 | 10233120 | 10233122 |
| 10233001 | 10233000 | 10233230 | 10233232 |
| Table de routage | | | |
| 02212102 | 1 | 22301203 | 31203203 |
| 0 | 11301233 | 12230203 | 13021022 |
| 10031203 | 10132102 | 2 | 10323302 |
| 10200230 | 10211302 | 10222302 | 3 |
| 10230322 | 10231000 | 10232121 | 3 |
| 10233001 | 1 | 10233232 | |
| 0 | | 10233120 | |
| | | 2 | |
| Voisins réels | | | |
| 13021022 | 10200230 | 11301233 | 31301233 |
| 02212102 | 22301203 | 31203203 | 33213321 |

Figure 2.32 : La table d'état d'un nœud Pastry d'identifiant 10233102 (base4) [15]

Le routage

Le principe du routage de Pastry est le suivant : lorsqu'une requête munie d'une clé K arrive dans un pair n , celui-ci vérifie tout d'abord si la clé est comprise dans l'ensemble de ses voisins virtuels. Si c'est le cas, la requête est routée vers son voisin virtuel qui a l'identifiant numérique le plus proche de la clé K . Sinon, le pair n détermine m , le nombre de chiffres communs à son identifiant et à K , consulte sa table de routage pour trouver un pair qui présente $m+1$ chiffres en commun et lui fait suivre la requête. Si aucun pair ne correspond, le pair n cherche, dans toute sa table d'états, un autre pair ayant m chiffres en commun et dont l'identifiant est le plus proche de K .

Le message est arrivé à destination lorsqu'aucune entrée de la table d'état ne permet de se rapprocher plus de la clé K .

Entrée d'un nœud dans le réseau

Lorsqu'un nœud désire entrer dans le réseau, il envoie, à un pair d'entrée, une requête d'entrée ayant comme destination son propre identifiant. Le pair d'entrée transmet à l'émetteur de la requête sa table d'état, si nécessaire met à jour sa propre table et route ensuite la requête normalement.

Chaque pair qui reçoit par la suite la requête d'entrée fait de même. Finalement, le nœud voulant entrer dans le réseau P2P crée sa table d'état en fonction des réponses qu'il a

reçues et la distribue à tous les nœuds contenus dans sa table d'état. Cette distribution finale de l'état d'un nœud entrant permet aux nœuds déjà membres du réseau de mettre à jour leur table d'état sans besoin d'émettre une requête.

E. Le système Kademlia [16]

Kademlia appelé aussi « kad » est un protocole P2P qui se base sur une table de hachage distribuée. Créé en l'an 2002. Il spécifie la structure du réseau overlay et la manière dont se passe l'échange d'informations. Les nœuds du réseau Kademlia communiquent entre eux via le protocole UDP.

-Chaque nœud « *nodeID* » et information (généralement un mot-clef) appelé « *value* » est affilié à un identifiant unique, codé sur 160 bits, Ces identifiants sont utilisés par Kademlia pour rechercher des informations indexées.

-Un nœud est responsable d'un certain nombre de couples « key, value » dont les identifiants « key » sont proches de son « nodeID ».

-La distance « *d* » entre deux identifiants *x* et *y* est définie par la fonction XOR (ou exclusif) entre ces derniers : $d(x, y) = x \oplus y$ [20].

-Avec cette méthode d'indexation, retrouver une information revient à chercher un nœud dont l'identifiant est proche. Pour ce faire, Kademlia considère chaque nœud comme une feuille d'un arbre binaire. La position de chaque nœud est déterminée par le plus petit préfixe unique de son *nodeID*.

-Ce protocole s'assure que chaque nœud possède au moins un contact dans chaque sous arbre. Avec cette garantie, chaque nœud peut trouver tout autre nœud dont l'identifiant est différent du sien.

- La majorité des clients Kad sont qualifiées d'hybrides. Car ils supportent à la fois le réseau Kad et le réseau eDonkey2000 (*ed2k* en abrégé). Ces utilisateurs sont aussi connectés aux serveurs *ed2k* qui sont utilisés pour trouver des points d'entrées sur le réseau Kad.

-Kademlia utilise quatre procédures, chacune prenant en paramètre un identifiant ID codé sur 160 bits :

– La procédure **Ping** interroge un nœud pour savoir s'il est connecté au réseau.

– **Store** ordonne à un nœud de stocker un couple <key, value> dans le but de le récupérer ultérieurement.

– **Find-node** retourne une liste de triplets contenant l'adresse IP, le numéro de port UDP et l'identifiant *nodeID* des nœuds connus pour être les plus proches de l'identifiant ciblé.

- **Find-value** : même que **Find-node**, mais si le destinataire a la clé demandée dans son magasin, il retourne juste la valeur stockée.

Fonctionnement de Kademia :

- Un nœud « N » recherche un autre nœud du réseau :

- « N » lance la requête « FIND_NODE » en interrogeant les α nœuds se trouvant dans ses propres k-seaux et qui sont proches de la clé désirée.
- Les nœuds contactés répondront à la requête de « N » en donnant à partir de leur propres k-seaux, la liste des nœuds les plus proches de la clé.
- « N » met à jour la liste des résultats obtenus, en gardant les « k » meilleurs qui répondent aux critères de recherche.
- Et rebelote, « N » enverra encore et encore la même requête jusqu'à obtention de l'emplacement du nœud désiré.

- Un nœud « N » veut rejoindre le réseau pour cela il suit le schéma suivant :

- « N » a recours au bootstrap pour trouver un nœud déjà existant sur le réseau.
- « N » calcule aléatoirement un ID censé ne pas être déjà attribué à un autre nœud.
- Le nœud trouvé grace au bootstrap insère « N » dans l'un de ces k-seaux.
- « N » fait alors un « FIND_NODE » de son propre ID.

-Un nœud « N » souhaite localiser une ressource :

- « N » lance la requête « FIND_VALUE » en interrogeant les α nœuds se trouvant dans ses propres k-seaux et qui sont proches de la clé désirée.
- Les nœuds contactés répondront à la requête de « N » en donnant « s'ils ne possèdent pas la ressource » la liste des nœuds les plus proches de la clé à partir de leur propres k-seaux.
- Si un nœud contacté par « N » possède la ressource, ce dernier enverra alors la « value » à « N ».

-Un nœud « N » tombe en panne et quitte le réseau sans en informer les autres :

- Grace aux messages périodiques de type « Ping » les autres nœuds détecteront l'absence de « N ».

5. Bilan des systèmes P2P structurés

| | |
|----------|---|
| Chord | <ul style="list-style-type: none"> -Topologie en Anneau -Type de recherche : égalité -Coût de la recherche : $O(\log(n))$ -Coût de la mise à jour : $O(\log(n))$ -Coût de l'ajout d'un nœud : $O(\log_2(n))$ - Tous les sauts n'ont pas forcément le même prix -Type de fichier : tout type de fichier -Anonymat : ✓ |
| Can | <ul style="list-style-type: none"> -Topologie en D-dimension -Coût de la recherche : $O(\log(n))$ -Type de fichier : tout type de fichier -Anonymat : ✓ |
| Pastry | <ul style="list-style-type: none"> -Topologie en Anneau -Coût de la recherche : $O(\log_2^b(N))$ -Type de fichier : tout type de fichier -Anonymat : ✓ |
| Kademlia | <ul style="list-style-type: none"> -Topologie en Arborescence -Coût de la recherche : $O(\log_2^b(N))$ -Type de fichier : tout type de fichier -Anonymat : ✓ |

Tableau 2.5 : Bilan des systèmes P2P structurés

D'après le tableau comparatif, on voit que le moins efficace en termes de coût de recherche, qui est le paramètre le plus important pour un pair est le système Can.

Bien que chacun soit redoutable dans une application et inefficace dans une autre, donc le choix du système à utiliser dépendra fortement de l'utilisation que nous voulons en faire.

6. Bilan des architectures Peer-to-Peer

| | <i>Non structuré</i> | <i>Structuré</i> |
|--------------------------|----------------------|------------------|
| -Autonomie | Forte | Faible |
| -Efficacité | Faible | Forte |
| -Qualité de réponse | Faible | Forte |
| -Expressivité du langage | Forte | Faible |
| -Tolérance aux pannes | Forte | Forte |

Tableau 2.6 : Bilan des architectures P2P [17]

Le bilan montre clairement que l'architecture structurée et hybride ont une qualité de réponse forte, mais souffrent chacune de problèmes comme l'autonomie et la tolérance aux

pannes, ce qui fait que il n'y a pas encore de système parfait absolu et c'est pour cette raison que la recherche dans ce domaine est encore très active.

7. Les logiciels p2p les plus courants

Nous ne détaillerons volontairement pas les logiciels de p2p qui restent trop nombreux. Néanmoins, parmi les plus connus et utilisés nous pouvons en citer quelques-uns.

LimeWire

Logiciel p2p développé en Java en s'appuyant sur le réseau distribué Gnutella. A noter que plusieurs limitations ont été mises en place pour empêcher le partage de fichiers sans licence.

Frostwire

FrostWire est un logiciel libre en Java permettant l'échange de fichiers via le protocole BitTorrent. Il était au départ basé sur le logiciel Limewire.

Jusqu'en 2011 FrostWire était compatible avec le réseau Gnutella mais depuis cette année-là (saisie judiciaire de Limewire l'année précédente) il est devenu un client BitTorrent.

uTorrent [22]

uTorrent est un client Peer-to-Peer très léger compatible avec le protocole Bit-Torrent, un protocole de téléchargement des plus populaires, conçu pour la distribution de fichiers à très haut débit. uTorrent est très utilisé car il est très léger et extrêmement facile à utiliser.

eMule

Emule s'est imposé comme un logiciel d'échange et de partage incontournable s'appuyant pour sa part sur le réseau distribué « Kademlia » ainsi que sur « eDonkey ». Très apprécié par les internautes, de nombreux forks et Mod d'eMule existent (xMule, eMule Xtreme, eMule ZZUL, etc.).

Soulseek

Soulseek est un logiciel libre de p2p s'appuyant sur deux serveurs centraux, l'un supportant le client original et la version 156, l'autre supportant le nouveau réseau (fonctionnant avec les clients 157 et QT) et regroupant de multiples salons de discussions par centre d'intérêt (Rock, Jazz, Reggae, Soul, etc). Très conviviale, Soulseek s'axe essentiellement sur l'échange de production musicale personnelle et met l'accent sur une communauté conviviale de musiciens.

8. L'avenir du p2p [18]

Si beaucoup imaginent la fin du p2p à grand renfort de décision de justice sous la pression des majors, celui-ci étant systématiquement associé au piratage, beaucoup seront surpris de constater que le p2p est en passe de connaître une nouvelle ère.

En informatique, difficile de prédire l'avenir, mais on peut voir quelques tendances :

- **Les P2P payants** : Actuellement, le contenu de tous les réseaux P2P est accessibles gratuitement. Beaucoup d'entreprises essaient de mettre en place un système de P2P qui permettrait de rapporter de l'argent, mais le modèle économique est difficile à trouver, et cela pose de nombreux problèmes techniques.
- **Les P2P anonymes** : Dans les réseaux P2P classiques, il est possible de savoir que telle adresse IP possède tel fichier. Cela permet donc d'identifier théoriquement qui distribue ou télécharge un fichier. Les P2P anonymes sont différents : ils fonctionnent de telle manière que vous ne savez pas qui distribue un fichier (utilisation d'intermédiaires), et il est impossible de savoir ce que vous téléchargez (tout est chiffré). Pour le moment, ces réseaux sont expérimentaux et possèdent de nombreux inconvénients (lenteur, complexité, faible nombre de participants, mauvaise résistance aux attaques, etc.). Citons : *Freenet, Mute, GnuNet ...*

Certains P2P non-anonymes (comme BitTorrent) peuvent fonctionner avec des surcouches (*Tor, I2P ...*) qui permettent de les rendre un peu plus anonymes, mais au prix de pertes de performances importantes.

- **Les P2P privés** : Ce système consiste à créer des réseaux P2P privés, uniquement entre amis. Vous ne pouvez accéder à un réseau P2P privé que sur invitation. Chacun peut créer un réseau privé, et se relier à d'autres réseaux privés. Toute personne extérieure au réseau P2P privé ne peut pas voir ce qu'il contient, ni ce que les personnes échangent. Ils sont encore assez peu développés, mais ils fonctionnent bien et ne possèdent pas les inconvénients des P2P anonymes.

- **L'utilisation des codes correcteurs d'erreur** : Actuellement, les réseaux P2P utilisent des codes détecteurs d'erreur (tel que SHA-1 ou MD5), mais pas de code correcteurs d'erreur (tels que Reed-Solomon).

Les codes correcteurs d'erreur permettent de reconstituer des parties manquantes ou endommagées d'un fichier. Cela permettrait en théorie de terminer plus rapidement les téléchargements (Il est possible de reconstituer les parties manquantes du fichier, même si personne d'actuellement connecté ne possède ces bouts).

Microsoft travaille d'ailleurs sur ce genre de réseaux avec *Avalanche*, un système P2P similaire à BitTorrent mais utilisant des codes similaires aux codes Reed-Solomon.

9. Conclusion

On a ainsi vu comment le P2P est redoutable dans le partage de fichiers, utilisant diverses techniques algorithmiques pour l'accès au fichier et également en utilisant sur un même fichier des techniques de partage équitables afin d'assurer la persistance des fichiers dans le temps et un téléchargement rapide et fiable.

Ainsi nous distinguons trois sous-modèles qui sont : les modèles centralisés, hybride et pur.

Après la présentation des caractéristiques de certains protocoles et la description de leurs principes de fonctionnement, nous décrivons dans le chapitre suivant les étapes que nous avons suivies afin d'aboutir à une application de partage de fichier et de communication entre pairs, simple et efficace.

Chapitre 3 : Présentation de notre Application

1. Introduction

Dans ce chapitre, nous allons présenter et détailler les différents cas d'utilisation de notre application avec des diagrammes, nous présenterons les outils que nous avons utilisé avant de commencer l'élaboration de notre projet, enfin nous présenterons notre application, via des captures d'écran des interfaces graphiques nombreuses et variées, expliquant ainsi toutes les possibilités offertes par notre application.

2. La Modélisation de notre application

Pour notre application, il existe un seul type d'utilisateur. L'utilisateur est toute personne physique, à ce titre, il peut :

- Voir tous les utilisateurs connectés.
- Télécharger n'importe quel type de fichier.
- Partager des fichiers.
- Consulter la table des fichiers partagés par les autres pairs.
- Chatter.
- Faire de la vidéo conférence.

3. Les Diagrammes

Afin de représenter les différentes fonctionnalités et utilisations possible de notre application, nous allons présenter une série de diagrammes, en l'occurrence un diagramme des cas d'utilisation, un diagramme de classes et des diagrammes de séquence, pour mettre à jour toute sa force et nous faciliter l'étape implémentation, qui répondra aux objectifs posés au début.

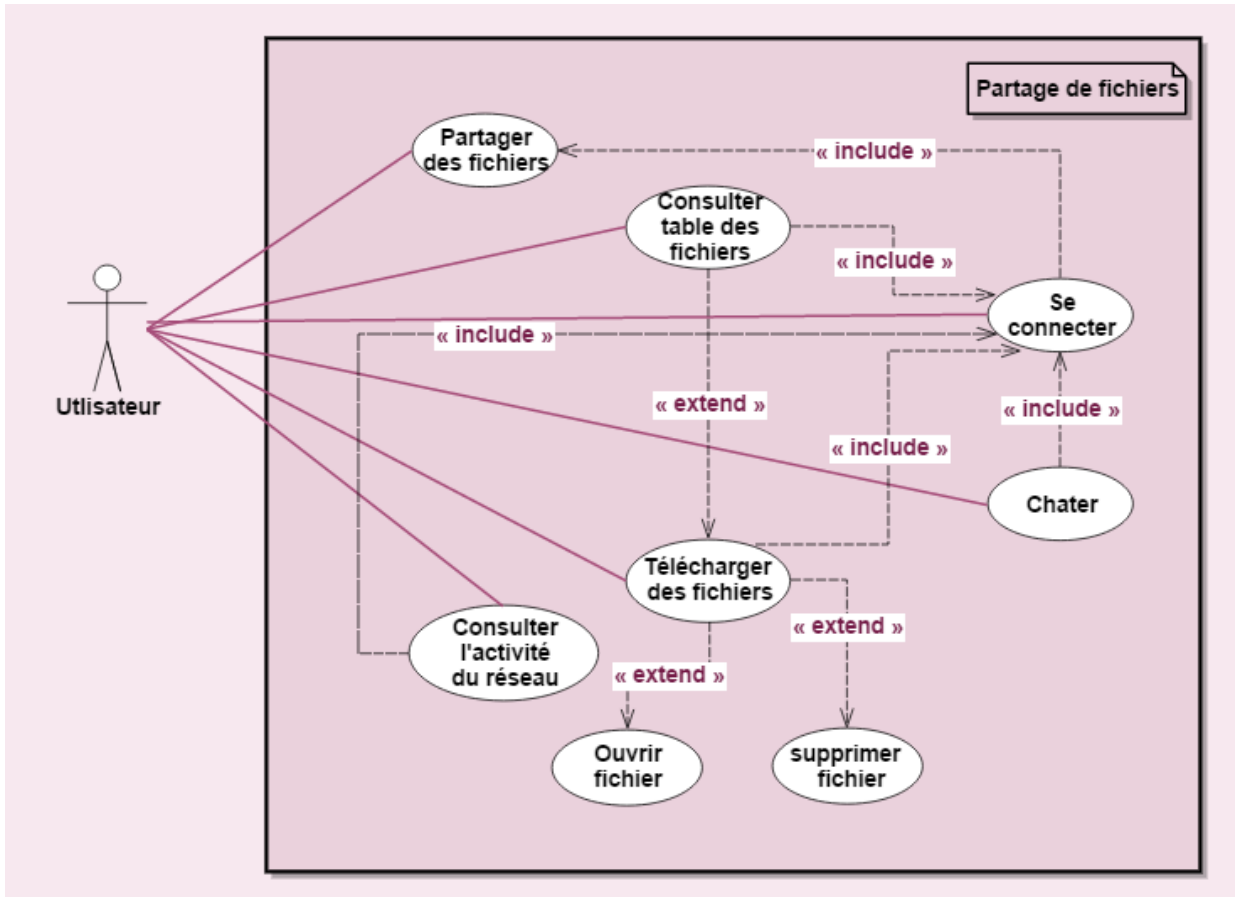


Figure 3.1 : Diagramme de cas d'utilisation

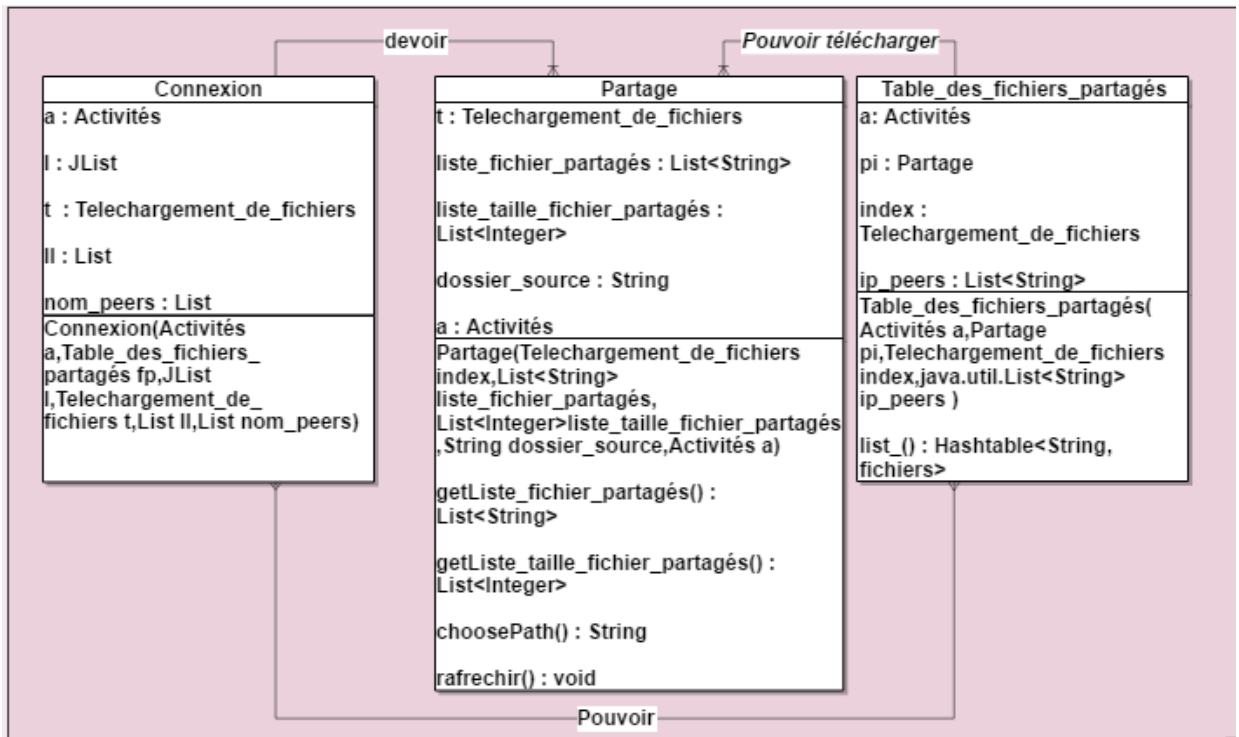


Figure 3.2 : Diagramme de classes

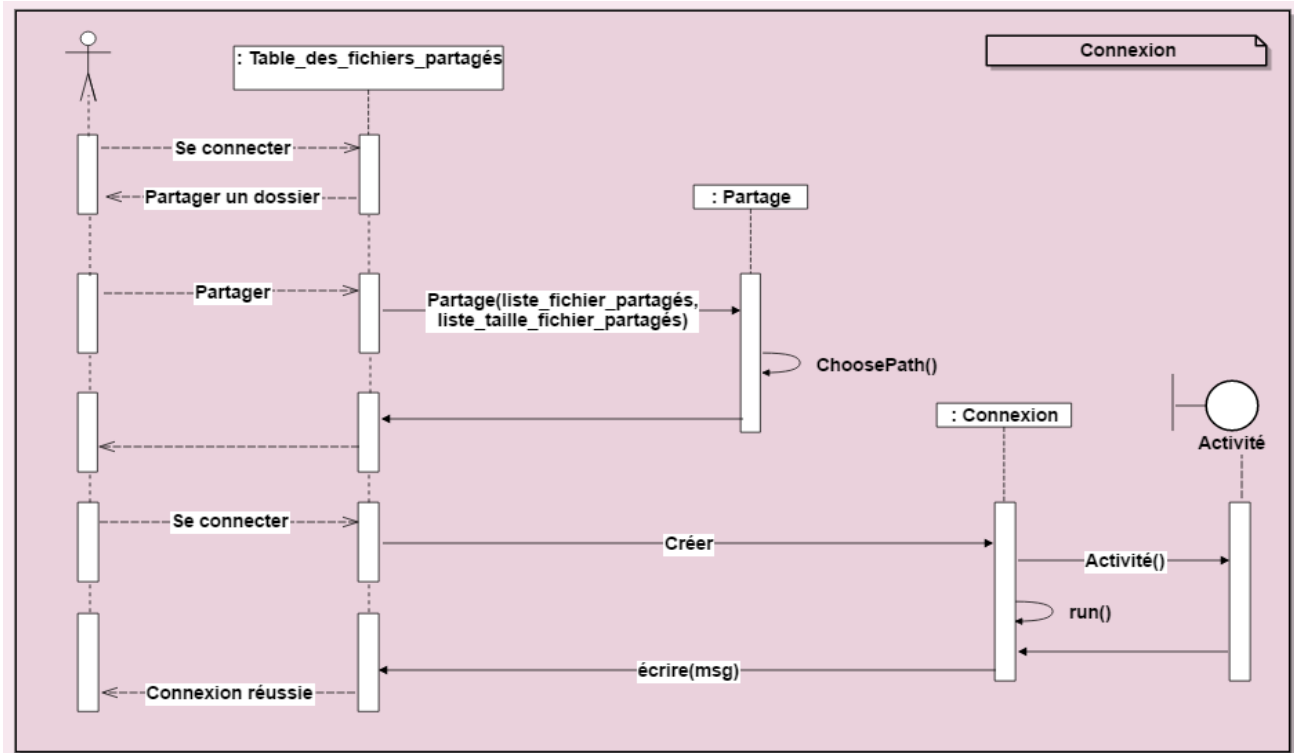


Figure 3.3 : Diagramme de séquence du cas d'utilisation <<Connexion>>

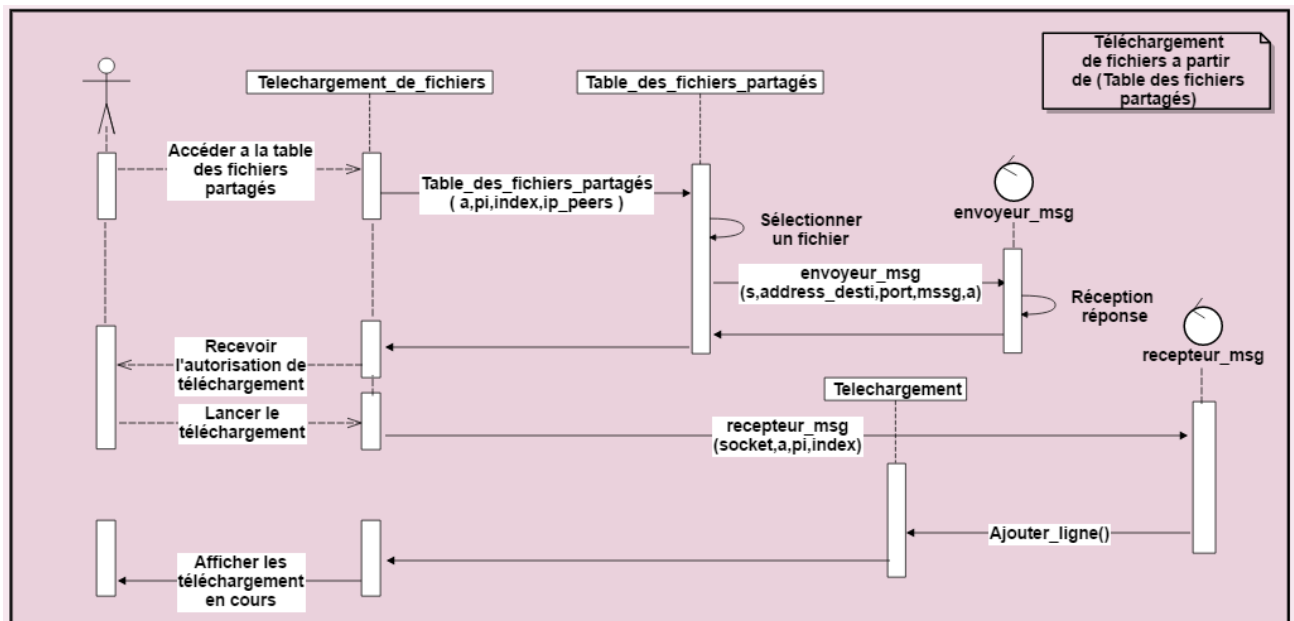


Figure 3.4 : Diagramme de séquence du cas d'utilisation

<<Téléchargement de fichiers>>

4. Dimensionnement des couches matérielles

Pour mettre sur pied notre application nous avons utilisé le matériel suivant :

-Trois PC (SE: windows 8.1/ RAM: 4go).

-IDE Netbeans 7.4 (JDK: V1.7.0_60/ JRE 7/Serveur web apache Tomcat Ou glassfish).

-switch.

5. Caractéristiques de l'application

1. Le téléchargement comprend 2 acteurs étant : le pair qui envoie et celui qui réceptionne.
2. La recherche de fichier se fait par mots clés.
3. Recherche de fichier déterministe.
4. Utilisation du protocole UDP pour la découverte du réseau, et le protocole TCP pour le reste.
5. Téléchargement simultané dans les deux sens supporté.

6. Architecture

La topologie de notre réseau dans laquelle nous avons testé notre solution est la suivante :

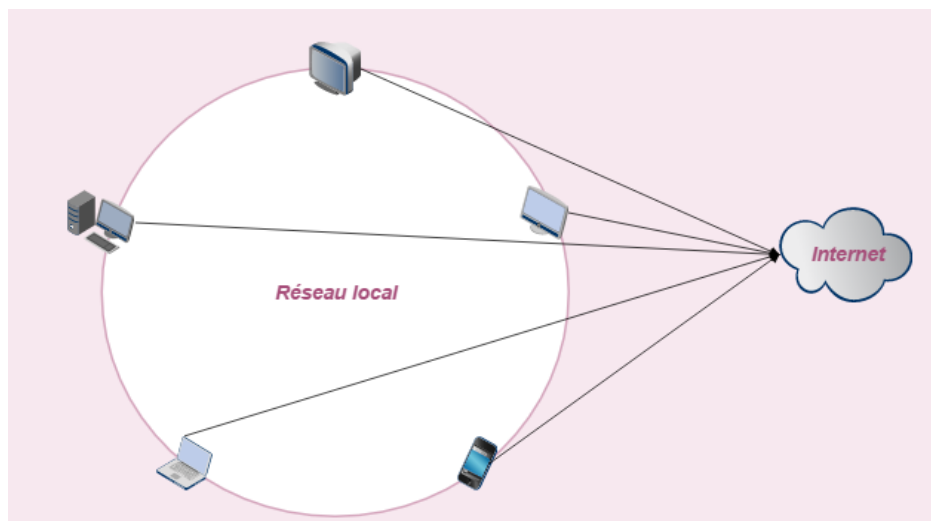


Figure 3.5 : Topologie du réseau test

7. Interface de l'application

L'interface de l'application comprend 7 onglets :

- **INFORMATIONS :**

Cet onglet affiche les caractéristiques du pc hôte :

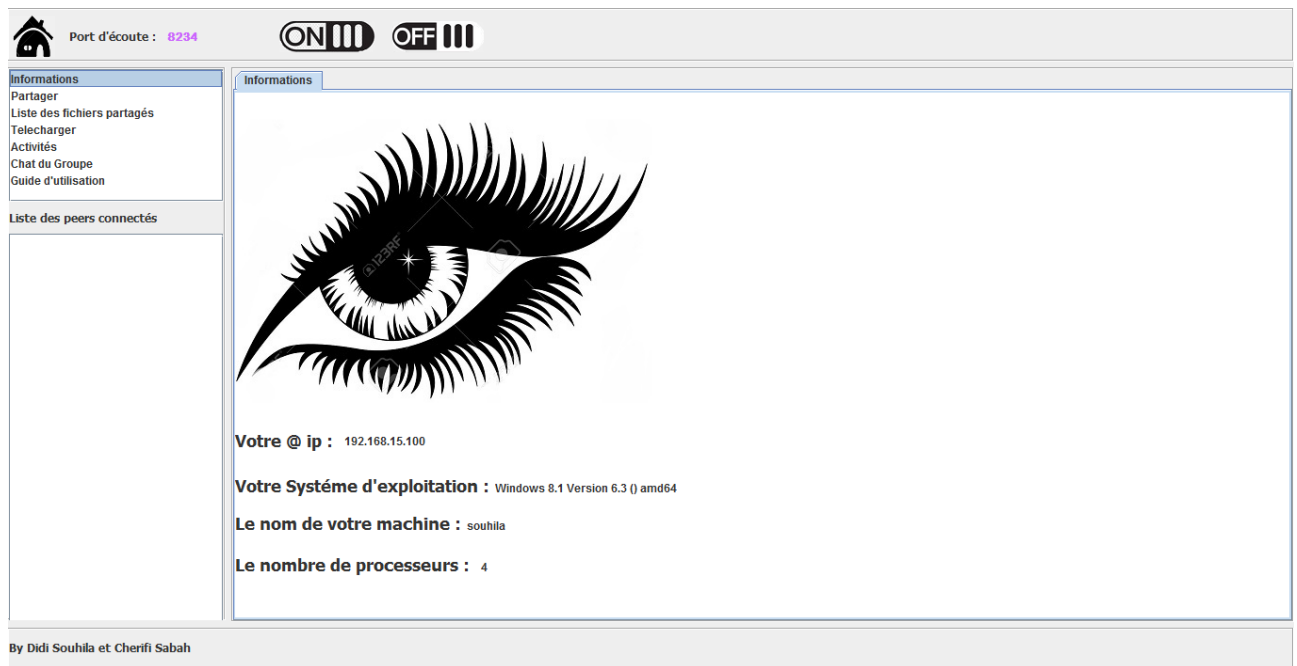


Figure 3.6 : Onglet Informations

- **PARTAGER :**

Dans cet onglet on peut choisir le dossier que l'on partage.

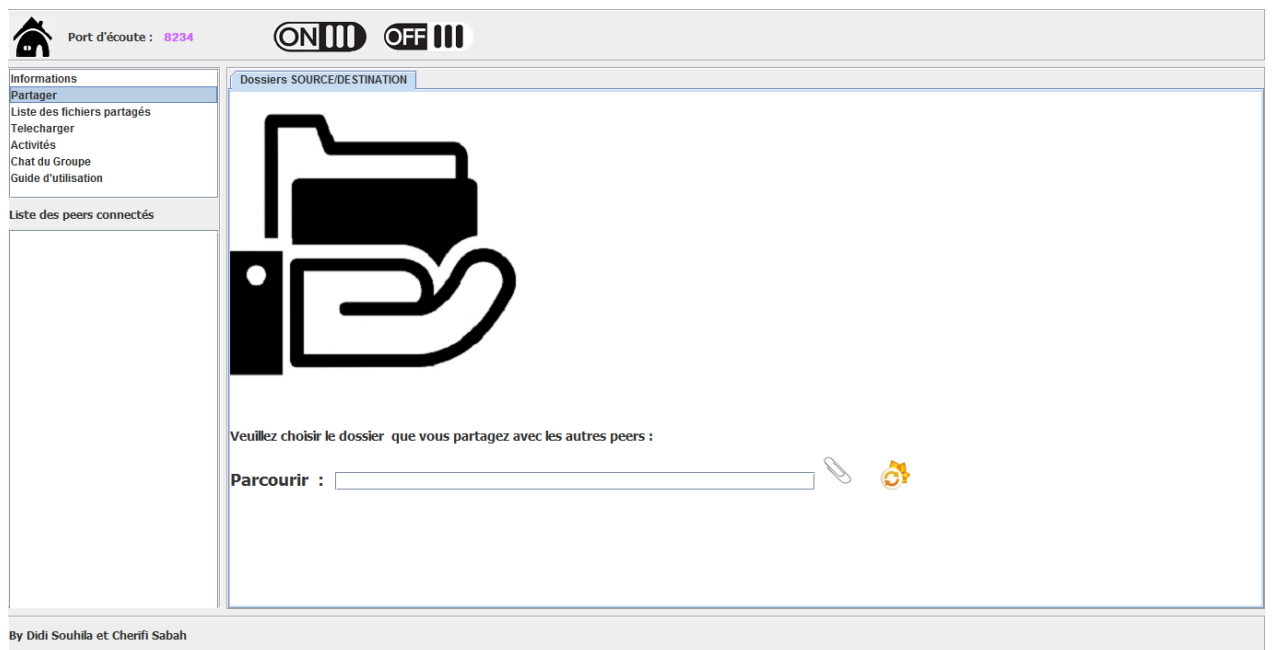


Figure 3.7 : Onglet Partager

- **LISTE DES FICHIERS PARTAGES :**

Ici sont affichés les fichiers partagés par tous les autres pairs.



Figure 3.8 : Onglet Liste des fichiers partagés

- **TELECHARGER :**

Cet onglet nous affiche les fichiers téléchargés, et nous donne la possibilité de supprimer, d'ouvrir, ou d'accéder au répertoire qui contient l'un de ces fichiers reçus.

On peut également télécharger en tapant dans la barre de recherche un mot clé.

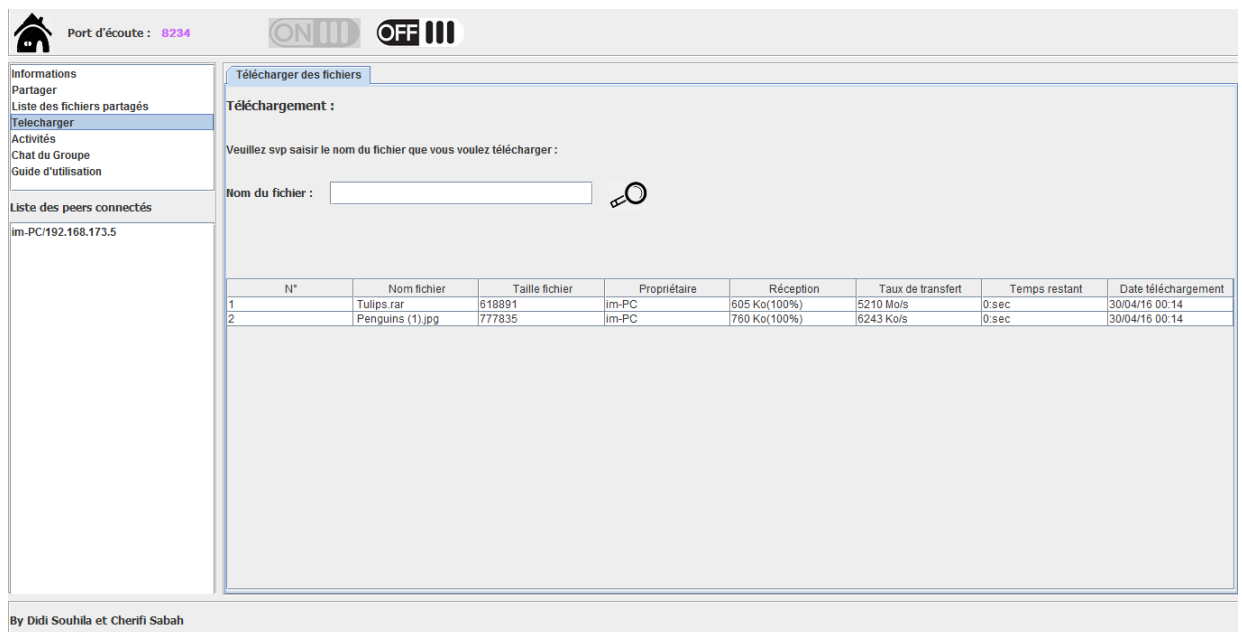


Figure 3.9 : Onglet Télécharger

- **ACTIVITES :**

Affiche l'activité du réseau en outre « l'inscription au groupe multicast, les différents messages envoyés et reçus, et les datagrammes envoyés et reçus ».

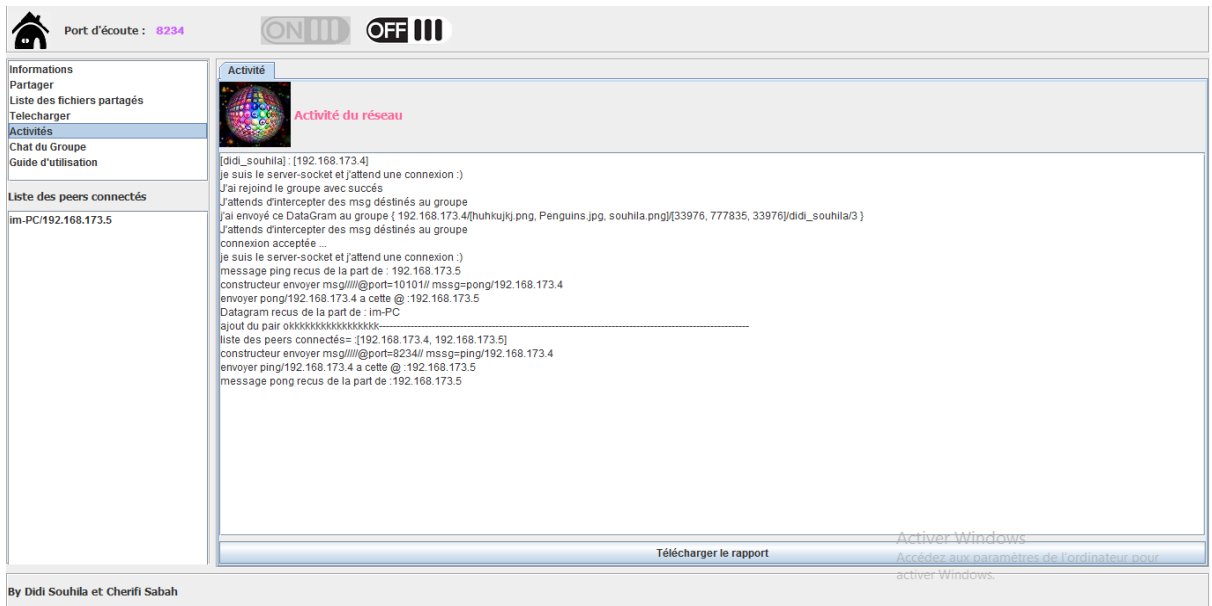


Figure 3.10 : Onglet Activités

- **CHAT :**

Dans cet onglet il est possible d'envoyer des messages aux utilisateurs de l'application, et démarrer une conversation visuelle si besoin est, entre deux ou plusieurs pairs.

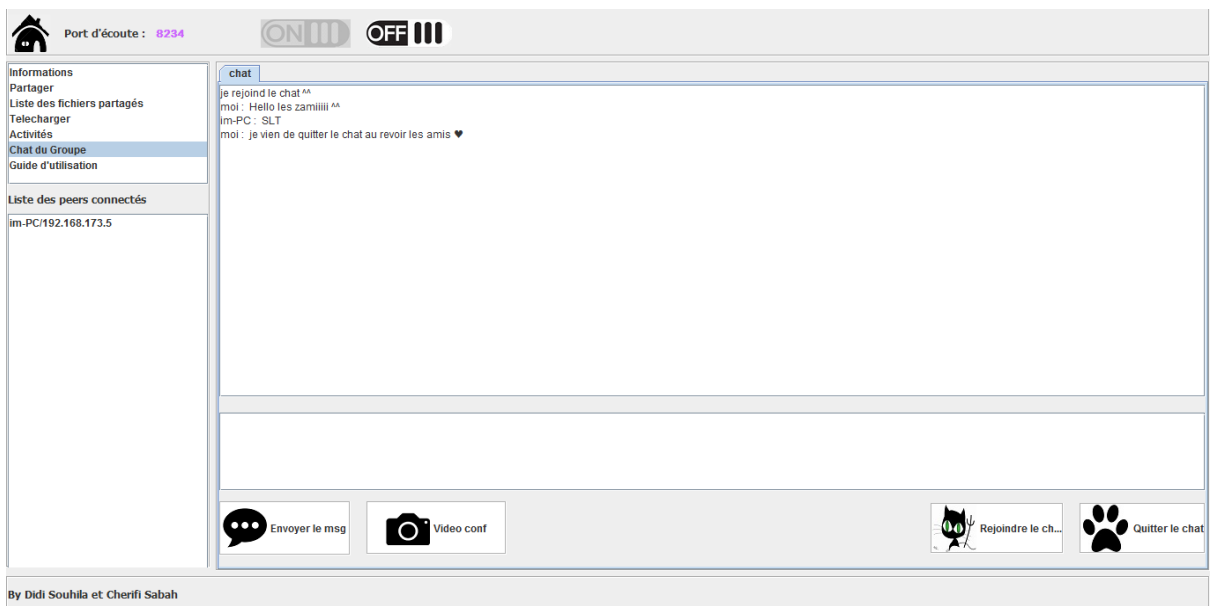


Figure 3.11 : Onglet Chat

- **GUIDE D'UTILISATION :**

Comme son nom l'indique, c'est un petit manuel d'utilisation de l'application.

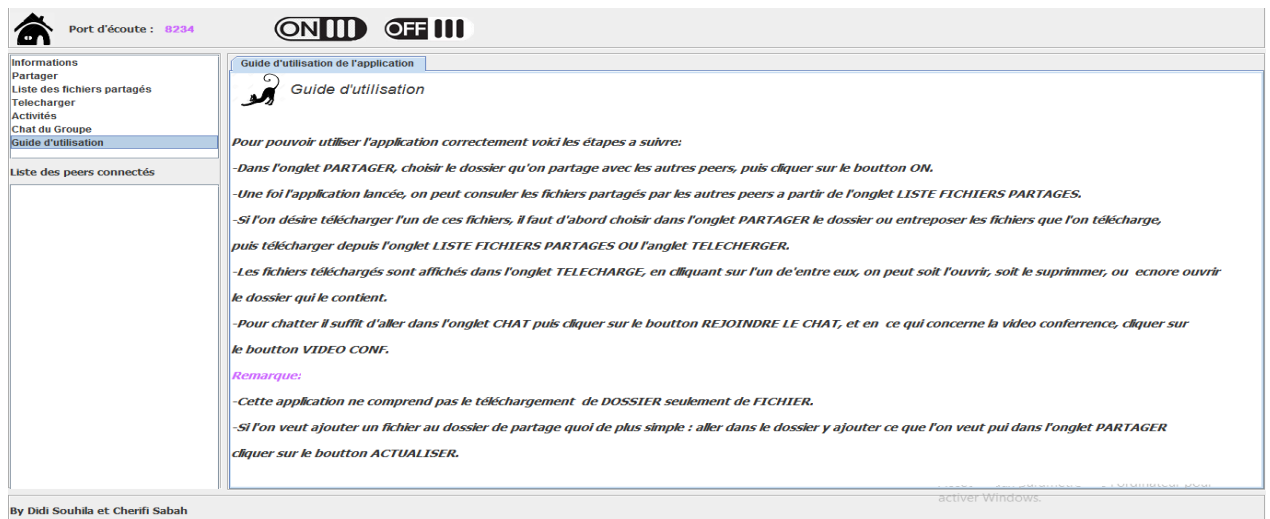


Figure 3.12 : Onglet Guide d'utilisation

8. Fonctionnement

Une fois le bouton « ON » lancé :

- **Etape 1**

Inscription au groupe multicast dont l'adresse est « 228.5.6.7 » à l'aide de la classe « Connexion.java ».

- **Etape 2 :**

Lancement du socket d'écoute sur le port 8234 afin d'intercepter les messages qui circulent sur le réseau à l'aide de la classe « Serveur_socket.java ».

- **Etape 3 :**

Envoi de datagramme toutes les quelques secondes afin que les autres pairs connectés nous trouvent, et ce grâce à la classe « envoyeur-datagram.java ».

- ✓ A l'interception d'un datagramme, un message Ping est envoyé à l'aide de la classe « envoyeur-msg.java » et un Pong est reçu en guise de réponse à partir de la classe « receteur_msg ».
- ✓ Le téléchargement se fait par échange de message : envoyer un message RECHERCHE à l'aide de la classe « envoyeur-msg.java » et recevoir un autre message REPONCE-RECH contenant une réponse positive ou négative à l'aide de la classe « receteur-msg.java ».

- ✓ La vidéoconférence utilise une API WebRTC « Communications Web en temps réel » qui permet la vidéo Peer-to-Peer, et l'audio dans un navigateur web sans plugins, ni cadres ou applications nécessaires.
- ✓ A la déconnection un message BYE est envoyé grâce à la classe « `envoyeur_msg.java` » à tous les pairs et tous les sockets sont fermés.

9. Les messages échangés

- **Ping (@ IP) :**

A la réception d'un datagramme contenant une adresse IP qui n'a pas été enregistré dans la liste IP-PEERS, l'IP est stockée dans la liste et un message Ping est envoyé à destination de cet @ IP.

Le nombre de Ping envoyés équivaut au nombre de pairs connectés.

- **Pong (@ IP) :**

A la réception d'un Ping, un Pong est envoyé en guise de réponse.

Le nombre de Pong envoyés équivaut au nombre de Ping reçus.

- **Recherche_fp (Nom fichier, @IP, port) :**

Après avoir cliqué (Dans l'onglet « LISTE DES FICHIERS PARTAGES ») sur un fichier à télécharger, Un message de recherche est envoyé à l'@ IP concerné pour avoir une validation de téléchargement.

- **Rep_rech_fp (existe ou pas, @IP, NOM pc, port) :**

A la réception d'un message RECHERCHE_FP, si le pc dispose du fichier il envoie un message Rep_rech positif et sinon il envoie un message rep_rech négatif.

- **Recherche (Nom fichier, @IP, port) :**

Après avoir choisi et valider (Dans l'onglet « TELECHARGER ») un fichier à télécharger, un message de recherche est envoyé à l'@ IP concerné pour avoir une validation de téléchargement.

- **Rep_rech (existe ou pas, @IP, NOM pc, port) :**

A la réception d'un message RECHERCHE, si le pc dispose du fichier il envoie un message Rep_rech positif et sinon il envoie un message rep_rech négatif.

- **Bye :**

Une fois avoir quitté l'application un message BYE est envoyé à tous les pairs afin qu'ils suppriment de leur liste IP-PEERS l'@ IP ayant quitté le groupe.

10. Les packages utilisés

Ci-dessous, sont listés tous les packages java nécessaires à l'implémentation de notre application, dont le package pour les adresses multicast,....

| | |
|------------------|---|
| JAVA.NET | <ul style="list-style-type: none"> ✓ java.net.InetAddress ✓ java.net.MulticastSocket ✓ java.net.ServerSocket ✓ java.net.Socket ✓ java.net.DatagramPacket ✓ java.net.DatagramSocket |
| JAVA.UTIL | <ul style="list-style-type: none"> ✓ java.util.Vector ✓ java.util.Date ✓ java.util.ArrayList ✓ java.util.Hashtable ✓ java.util.List |
| JAVA.IO | <ul style="list-style-type: none"> ✓ java.io.DataOutputStream ✓ java.io.Serializable ✓ java.io.File ✓ java.io.FileOutputStream ✓ java.io.BufferedInputStream ✓ java.io.DataInputStream ✓ java.io.ObjectInputStream |

Tableau 3.1 : Liste des packages

11. Conclusion

Dans ce chapitre, nous avons détaillé l'acheminement de notre travail de fin d'études, la façon dont nous l'avons mené et les résultats obtenus sous forme de captures d'écran des différentes interfaces que nous avons développées. L'application développée en Java répond aux objectifs initialement posés, mais peut être améliorée avec un peu plus de temps imparti. Nous avons indiqué les bibliothèques Java que nous avons utilisées et nécessaires à l'application. Cependant nous avons découvert l'existence d'une bibliothèque nouvellement apparue en Java, facilitant le développement d'applications réseau pour des groupes, chose qui pourrait être utile pour les futurs étudiants intéressés par cette thématique.

Conclusion générale et Perspectives

Dans ce PFE, nous nous sommes intéressées à la thématique du Peer to Peer qui est en train de prendre de plus en plus d'ampleur, supplantant le modèle client/serveur qui a fait ses preuves, mais sature suite à la montée en puissance du nombre de téléchargements et de la quantité phénoménale d'informations échangées. L'avènement du p2p s'est introduit dans toutes sortes d'applications, qui jusqu'alors peinaient à fonctionner correctement avec le modèle client/serveur. Bien que ce dernier soit facile d'utilisation, a bien marché pendant longtemps, et que le p2p comporte encore plein de verrous, cependant la tendance du p2p est claire et son utilisation est arrivée aux sociétés privées, signe de sa réussite. Pour mener à bien notre projet de fin d'études, s'arquant sur la mise au point d'une application de téléchargement et de communication en Peer to Peer, permettant en premier lieu de faire des téléchargements de tout type de fichiers, ensuite de faire du chat et de la vidéoconférence entre pairs, sans passer par des serveurs centraux, donc en mode complètement décentralisé. L'application écrite en java, le langage portable présent sur toutes les plateformes informatiques existantes actuellement, a donné de bons résultats, mais peut encore être améliorée. Dans ce travail de fin d'études, nous avons procédé par étapes, comprenant d'abord cette thématique et ses verrous, puis les systèmes qui chacun a apporté un plus au précédent, nous avons donc fait un état de l'art pour mettre à jour les inconvénients et avantages des solutions existantes. Ce qui nous a inspiré par la suite, dans le choix de notre application pour une solution totalement décentralisée, mais nous n'avons réussi à la tester qu'en topologie locale et avec un nombre de pairs restreints. Dans l'avance de notre travail, on s'est vu confrontées à plusieurs problèmes, dont le plus dur fut la vidéoconférence, car la capture du flux vidéo en direct et son envoi vers un groupe de pairs n'est pas gérée par les bibliothèques java les mieux documentées et testées. Du coup ce point nous a maintenues bloquées longtemps, cependant cela nous a permis d'en apprendre encore plus en termes de programmation distribuée. Ce fut une expérience enrichissante et bénéfique, bien que frustrante car le temps nous était compté. Nous pensons avoir atteint l'objectif posé au tout début et espérons que les prochaines promotions pourront profiter de cet humble travail comme base pour faire plus. Des perspectives restent cependant à accomplir comme :

1. Faire des tests à grande échelle ;

2. Faire des tests via internet ;
3. Ajouter d'autres services comme la ToIP, les jeux réseau, le Cloud en Peer to Peer

Bibliographie

- [1] «Histoire d'Internet,» [En ligne]. Available: <http://histoire-internet.vincaria.net/post/histoire/internet/1997/Peer-to-peer>. [Accès le 03 05 2016].
- [2] O. abdou, «Historique - Naissance du P2P,» [En ligne]. Available: <http://webcache.googleusercontent.com/search?q=cache:http://morebus.free.fr/siteMemoire/historique02.html>. [Accès le 03 05 2016].
- [3] «Journal du Net - Encyclopedie,» [En ligne]. Available: http://www.journaldunet.com/encyclopedie/php/commun/imprimer_definition.php?f_id_definition=50. [Accès le 03 05 2016].
- [4] G. Feltin, G. Doyen et O. Festor, «Les protocoles Peer-to-Peer, leur utilisation et leur détection,» [En ligne]. Available: <https://hal.inria.fr/inria-00099498/file/A03-R-543.pdf>. [Accès le 03 05 2016].
- [5] D. S. Milojicic, «Peer-to-Peer Computing,» [En ligne]. Available: <http://www.hpl.hp.com/techreports/2002/HPL-2002-57R1.pdf>. [Accès le 03 05 2016].
- [6] O. Lausecker et L. Viennot, «Interstices - Les réseaux de pair à pair,» [En ligne]. Available: https://interstices.info/jcms/c_8622/les-reseaux-de-pair-a-pair. [Accès le 03 05 2016].
- [7] N. BUDAN, S. VAUBOURG et B. TEDESCHI, «Les réseaux peer-to-peer Fonctionnement, exemples, limites,» [En ligne]. Available: <http://www-igm.univ-mlv.fr/~duris/NTREZO/20022003/Peer-to-peer.pdf>. [Accès le 03 05 2016].
- [8] H. HAFI, «Protocole pour la sécurité des réseaux sans fil peer to peer,» [En ligne]. Available: <http://bu.univ-ouargla.dz/Houda-HAFI.pdf?idthese=830>. [Accès le 03 05 2016].
- [9] «P2P,» [En ligne]. Available: <http://www.capit.net/technique/p2p.aspx>. [Accès le 03 05 2016].
- [10] M. D.C, «Mieux comprendre le peer-to-peer (p2p),» [En ligne]. Available: <http://www.generation-nt.com/peer-to-peer-peer2peer-p2p-partage-fichiers-internet-article-40573-7.html>. [Accès le 03 05 2016].
- [11] A. Benoit, «Napster ,» [En ligne]. Available: <http://wiki.p2pfr.com/p2p/napster>. [Accès le 03 05 2016].
- [12] K. N. Fanny, P. Laurent et T. Alexandre, «Réseaux pair-à-pair structurés,» [En ligne]. Available: <http://deptinfo.unice.fr/twiki/pub/Minfo/TER/rapport-6.pdf>. [Accès le 03 05 2016].
- [13] «Bootstrapping-wikipedia,» [En ligne]. Available: <https://en.wikipedia.org/wiki/Bootstrapping>.
- [14] B. DEFUDE, «Recherche d'informations à grande échelle dans des architectures Peer-to-Peer,» [En ligne]. Available: <http://www-inf.int-evry.fr/~defude/P2P/P2PDefude.pdf>. [Accès le 03 05 2016].

- [15] H. Jarraya, «Un système de sauvegarde P2P sécurisé s'appuyant sur une architecture AAA,» [En ligne]. Available: https://portail.telecom-bretagne.eu/publi/public/download.jsp?id_publication=5335. [Accès le 03 05 2016].
- [16] «kademlia-wikipedia,» [En ligne]. Available: <https://en.wikipedia.org/wiki/Kademlia>. [Accès le 03 05 2016].
- [17] B. DEFUDE, «Gestion de données dans les systèmes pair-à-pair,» [En ligne]. Available: <http://www-poleia.lip6.fr/~doucet/CoursBDRA/BDRA-Cours-2008-09.pdf>. [Accès le 03 05 2016].
- [18] «Comprendre l'ordinateur,» [En ligne]. Available: <http://sebsauvage.net/comprendre/p2p/>. [Accès le 03 05 2016].
- [19] D. VODISLAV, «Architectures distribuées P2P,» 03 05 2016. [En ligne]. Available: <https://depinfo.u-cergy.fr/~vodislav/Master/M2Integration/1112/fichiers/p2p.pdf>.
- [20] P. Maymounkov et D. Mzières, «kademlia,» [En ligne]. Available: <https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf>. [Accès le 03 05 2016].
- [21] B. T. TIENDREBEOGO, «Systeme Dynamique et Reparti de Nomme a Indirections Multiples pour les Communications dans l'Internet.,» [En ligne]. Available: https://tel.archives-ouvertes.fr/file/index/docid/841591/filename/TIENDREBEOGO_TELESPHORE_2013.pdf. [Accès le 03 05 2016].
- [22] « μ Torrent,» [En ligne]. Available: <http://www.commentcamarche.net/download/telecharger-196-utorrent>. [Accès le 03 05 2016].
- [23] A. Benoit, «Réseaux pair-à-pair ,» [En ligne]. Available: <http://graal.ens-lyon.fr/~abenoit/reso05/cours/2-p2p.pdf>. [Accès le 03 05 2016].

Liste de figures

| | |
|---|----|
| Figure 2.1 : Classification des systèmes P2P non structurés | 14 |
| Figure 2.2 : Le logiciel Napster | 14 |
| Figure 2.3 : Partage de fichiers Napster | 15 |
| Figure 2.4 : Recherche de fichiers Napster | 15 |
| Figure 2.5 : Téléchargement de fichiers Napster | 15 |
| Figure 2.6 : Une amélioration du Peer-to-Peer centralisé..... | 16 |
| Figure 2.7 : Gwebcaches Gnutella..... | 18 |
| Figure 2.8 : Connexion d'un Peer Gnutella | 19 |
| Figure 2.9 : Routage des messages Ping Gnutella..... | 19 |
| Figure 2.10 : Routage des messages Pongs Gnutella..... | 19 |
| Figure 2.11 : Recherche de données (Query) Gnutella..... | 20 |
| Figure 2.12 : Routage des messages Query Gnutella | 20 |
| Figure 2.13 : Routage des messages QueryHit Gnutella | 21 |
| Figure 2.14 : Echange de fichiers Gnutella..... | 21 |
| Figure 2.15 : Panne d'un Peer..... | 22 |
| Figure 2.16 : Le modèle hybride | 23 |
| Figure 2.17 : Recherche de fichiers dans le modèle hybride | 23 |
| Figure 2.18 : Téléchargement de fichiers dans le modèle hybride | 24 |
| Figure 2.19 : Le téléchargement multiple | 24 |
| Figure 2.20 : Téléchargement avec BitTorrent | 27 |
| Figure 2.21 : Table de hachage distribuée (DHT) | 29 |
| Figure 2.22 : Bootstrap chord | 30 |
| Figure 2.23 : Connexion d'un peer chord | 30 |
| Figure 2.24 : Stabilisation d'un peer Chord..... | 30 |
| Figure 2.25 : Téléchargement de fichiers Chord | 31 |
| Figure 2.26 : Protocole Can | 32 |
| Figure 2.27 : BootStrap Can | 33 |
| Figure 2.28 : Routage des messages Can..... | 33 |
| Figure 2.29 : Connexion d'un Peer Can | 33 |
| Figure 2.30 : Recherche de fichier Can | 34 |
| Figure 2.31 : Départ d'un Peer Can | 34 |
| Figure 3.1 : Diagramme de cas d'utilisation..... | 44 |
| Figure 3.2 : Diagramme de classes..... | 44 |
| Figure 3.3 : Diagramme de séquence du cas d'utilisation <<Connexion>>..... | 45 |
| Figure 3.4 : Diagramme de séquence du cas d'utilisation <<Téléchargement de fichiers>>..... | 45 |
| Figure 3.5 : Architecture réseau où a été déployée l'application..... | 46 |
| Figure 3.6 : Onglet Informations..... | 47 |
| Figure 3.7 : Onglet Partager..... | 47 |
| Figure 3.8 : Onglet Liste des fichiers partagés..... | 48 |
| Figure 3.9 : Onglet Télécharger..... | 48 |
| Figure 3.10 : Onglet Activités..... | 49 |
| Figure 3.11 : Onglet Chat..... | 49 |
| Figure 3.12 : Onglet Guide d'utilisation..... | 50 |

Liste des tableaux

| | |
|---|----|
| Tableau 1.1 : Peer-to-Peer versus Client/serveur..... | 11 |
| Tableau 2.1 : Les requêtes Gnutella | 17 |
| Tableau 2.2 : Bilan des systèmes P2P non structurés | 27 |
| Tableau 2.3 : Table de routage du nouveau nœud | 31 |
| Tableau 2.4 : Table de routage du nœud 50..... | 31 |
| Tableau 2.5 : Bilan des systèmes P2P structurés | 39 |
| Tableau 2.6 : Bilan des architectures P2P | 39 |
| Tableau 3.1 : Liste des packages..... | 52 |

Résumé

Un réseau P2P (pair à pair) est composé d'un ensemble d'entités (appelées pairs), partageant un ensemble de ressources, et jouant à la fois le rôle de serveur et de client. Dès son apparition, ce modèle a connu un grand succès et a été considéré comme une vraie alternative au modèle classique client-serveur. Parmi les nouvelles utilisations de ce modèle, on trouve les systèmes de sauvegarde distribuée qui permettent à un utilisateur de garder une copie de ses données dans les espaces de stockage d'autres utilisateurs afin de les restaurer en cas de crash de disque dur, d'erreurs de manipulation... Cependant, la sécurité, le comportement égoïste de quelques pairs et la qualité de service offerte restent toujours des défis dans ce type de réseau.

Dans ce mémoire, nous nous sommes intéressés au téléchargement de fichier et à la communication des pairs. Notre application développée en java, très simple à utiliser permet donc de télécharger n'importe quel type de fichier quel que soit sa taille et ce sans passer par un élément central, puis de communiquer via le chat ou la vidéoconférence en groupes

Abstract

A peer-to-peer (P2P) network is composed of a set of entities (called pairs) which share a set of resources and play simultaneously the roles of server and client. This model has been very successful since the beginning and has been considered as a real alternative to the classical client-server model. The distributed backup system is one of the new usages of this model. Their objective is to enable users to keep a copy of their data in the disk space of other users so restoration of them is possible after a hard disk crash, error of handling ... However, the security, the selfish behaviour of a few peers and the quality of service are still challenging topics in this type of network. In this brief, we looked at file download and communication between peers. Our simple application developed in java, allows us to download any file type regardless of its size and without a central element, after that its allows us to communicate via chat or videoconferencing.

ملخص

شبكة (الند للند) تتكون من مجموعة من الوحدات (وتسمى الأقران)، تتشارك في مجموعة من الموارد، و تلعب دور الخادم والعميل. منذ ظهورها، كان هذا النموذج ناجحا جدا، وقد تم اعتباره بديلا حقيقيا للنموذج خدمة العملاء الكلاسيكية. من بين الاستخدامات الجديدة لهذا النموذج، هناك الأنظمة الاحتياطية الموزعة التي تسمح للمستخدم بالحفاظ على نسخة من بياناتها في مساحات التخزين لمستخدمين آخرين لاستعادتها في حالة تحطم القرص الصلب، الأخطاء المعالجة... و فيما يخص الأمن، وسلوك الأناية من بعض الزملاء وجودة الخدمة لا تزال التحديات في هذا النوع من الشبكات. في هذا الموجز، اهتمينا بتحميل الملف، تطبيقنا بسيط يسمح بتحميل أي نوع ملف بغض النظر عن حجمه و بغير حاجة إلى عنصر مركزي.