République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université A.MIRA-BEJAIA
Faculté des Sciences Exactes
Département d'Informatique

Mémoire

Présenté par

MOUALKIA Yamina

Pour l'obtention du diplôme de Magister

Filière : Informatique

Option : Cloud Computing

Thème

Performance et Optimisation des Architectures P2P pour les Applications des Réseaux Sociaux

Soutenu le : Devant le Jury composé de :

Nom et Prénom	Grade	Université d'origine	
Mr.AISSANI Djamil	Professeur	Univ.de Bejaia	**Président**
Mr. SEMCHEDINE Fouzi	M.C.A	Univ.de Sétif	**Rapporteur**
Mr.BOUKERRAM Abdellah	Professeur	Univ.de Bejaia	**Examinateur**
M^{me}.BOUALLOUCHE Louiza	Professeur	Univ.de Bejaia	**Examinatrice**
Mr. AMAD Mourad	M.C.B	Univ.de Bejaia	**Invité**

Année Universitaire : **2015-2016**

On estime à quelque deux milliards *le nombre d'êtres humains qui participent aux réseaux sociaux sur la planète. Des réseaux éminemment centralisés, dans lesquels nous ne sommes pas maîtres de grand-chose. Le scandale de la NSA comme la multiplication des modifications unilatérales des conditions d'utilisation de ces systèmes – Facebook a récemment décidé d'interdire aux utilisateurs de refuser d'apparaître dans les résultats de recherche – montrent combien cette centralisation est devenue un piège pour l'espionnage de masse", qu'il soit le fait des Etats ou d'entreprises privées comme les Gafa (**G**oogle, **A**pple, **F**acebook, **A**mazon…).*

professeur François Taïani

Remerciements

Je tiens tout d'abord à remercier ALLAH le tout puissant, qui m'a donné l'aide, la force , la volonté dans les moments difficiles et la patience d'accomplir ce modeste travail.

« Mon Seigneur ! Inspire-moi de Te remercier pour les bienfaits dont Tu m'as comblé »

Je souhaite ensuite exprimer ma gratitude et ma reconnaissance à mes promoteurs Dr AMAD Mourad et Dr SEMCHEDINE Fouzi , pour leur patience, leur modestie, le temps qu'il m'ont consacré, la qualité de leur encadrement, leurs encouragements et pour leurs précieux conseils et remarques.

J'exprime, aussi, mes vifs remerciements aux membres de jury de ma soutenance d'avoir pris le temps de lire et juger ce mémoire et le travail qu'il représente.

Une pensée particulière est adressée à tous mes collègues et amies de l'école doctorale Cloud Computing de l'université de A. Mira-Bejaia, spécialement ma chère pour leurs encouragements , les moments passés ensemble ont été très agréables.

Enfin, que tous ceux qui ont contribué de près ou de loin, par leurs encouragements et conseils à l'accomplissement de ce travail, trouvent ici l'expression de ma profonde reconnaissance

Dédicace

Je dédie ce modeste travail à ma bien-aimée mère, à mon cher père, Aucun hommage ne pourrait être à la hauteur de l'amour Dont ils ne cessent de me combler. Que dieu leur procure bonne santé et longue vie,

à mes adorables frères.et sœurs,

à toute ma famille,

à toutes mes amies et à mes collègues de l'école doctorale.

Résumé : Le P2P est une nouvelle technologie réseau et ses applications sont variées, nous citons a titre d'exemple : le multicast applicatif, la signalisation, le parallélisme, le partage de fichiers, la téléphonie IP, la messagerie instantanée, les moteurs de recherche et plus récemment les applications des réseaux sociaux. Le développement de nouvelles applications de groupe pour des fins culturelles ou commerciales est devenu indispensable et il suit une vitesse vertigineuse de sorte que les réseaux traditionnels (*client serveur*) sont dépassés. De nombreux développements sont actuellement en cours, particulièrement pour la gestion des utilisateurs et leurs données à grande échelle dans un environnement distribué.

Les réseaux sociaux comme Facebook ou Twitter sont aujourd'hui des applications très célèbres (*Le nombre d'utilisateur est en augmentation exponentielle*), la question de la confidentialité des données et l'anonymat des utilisateurs se pose sérieusement aujourd'hui. La nécessité de garder un certain contrôle de ses données sur le Net commence à devenir un sérieux problème qui pourra certainement être en grande partie réglé grâce à la décentralisation des données. Au lieu de poster une photo ou un commentaire (*donnée personnelle*) sur un serveur distant, ceux-ci sont hébergés sur l'ordinateur de l'utilisateur, ainsi, tout est sous contrôle. C'est tout simplement le principe du Peer-To-Peer, ou P2P. Néanmoins, la gestion d'un tel nombre d'utilisateurs est une tâche très complexe, voir très difficile. Par conséquent, l'aspect d'optimisation sur plusieurs critères (*espace mémoire, temps de calcul, temps de réponse, durée de vie des données,* …) est devenu un challenge ces dernières années.

Le but de ce mémoire est l'adaptation des architectures P2P pour les applications de groupes des réseaux sociaux. Notre architecture proposée vise à améliorer les performances (*coût de recherche, coût de maintenance*) par rapport aux autres architectures DOSNs en utilisant une architecture hiérarchique à deux niveaux ,capable de gérer le nombre important des utilisateurs d'une manière efficace.
Mots clés : Peer-To-Peer, Réseaux sociaux, décentralisation des données, Architecture P2P.

Abstract : P2P is a new network technology and its applications are varied, we cite as an example: the application multicast, signage, parallelism, file sharing, IP telephony, instant messaging, search engines and more recent applications of social networks. The development of new group of applications for cultural or commercial purposes has become indispensable and follows a dizzying speed so that traditional networks (*Client Server*) are exceeded. Many developments are underway, particularly for the management of users and their large-scale data in a distributed environment.

Social networks like Facebook or Twitter are now very famous applications (*The number of users is increasing exponentially*), the issue of data privacy and anonymity of users arises seriously today. The need to retain some control of their data on the Internet is starting to become a serious problem that can certainly be largely resolved through decentralization of data. Instead of posting a photo or comment (*personal information*) to a remote server, they are hosted on the user's computer and everything is under control. It's just the principle of Peer-To-Peer or P2P. Nevertheless, the management of such users is a very complex, very difficult to see. Therefore, the appearance of several optimization criteria (*memory space, computation time, response time, data lifetime,* ...) has become a challenge in recent years.

The objective of this thesis is the adaptation of P2P architectures for groups of social networking applications. Our proposed architecture is designed to improve performance (*lookup cost, maintenance cost*) compared to other DOSNs architectures, using a hierarchical architecture with two levels, able to handle the large number of users in an effective manner.
Keywords: Peer-To-Peer, Social Networks, decentralization of data, P2P Architecture.

الملخص : P2Pهي تقنية الشبكة الجديدة وتطبيقاتها متنوعة، نذكر على سبيل المثال: تطبيق الإرسال المتعدد، لافتات، والتوازي، تبادل الملفات، الاتصال عبر بروتوكول الإنترنت، والرسائل الفورية، ومحركات البحث والمزيد من التطبيقات الحديثة للشبكات الاجتماعية. لقد أصبح تطوير مجموعة جديدة من التطبيقات لأغراض ثقافية أو تجارية لا غنى عنه ويتبع بسرعة مذهلة بحيث يتم تجاوز الشبكات التقليدية (*الخادم العميل*). وتجري حاليا العديد من التطورات، لا سيما بالنسبة لإدارة المستخدمين والبيانات على نطاق واسع في بيئة موزعة.

الشبكات الاجتماعية مثل الفيسبوك أو تويتر هي الآن من التطبيقات الشهيرة جدا (*عدد المستخدمين يزداد أضعافا مضاعفة*)، ومسألة خصوصية البيانات وعدم الكشف عن هوية المستخدمين تبرز جديا اليوم. الحاجة إلى الاحتفاظ ببعض التحكم في البيانات الخاصة بهم على شبكة الإنترنت قد بدأ يصبح مشكلة خطيرة يمكن بالتأكيد أن تحل بشكل كبير من خلال اللامركزية في البيانات.

بدلا من نشر صورة أو تعليق (*المعلومات الشخصية*) إلى ملقم بعيد، يتم استضافتها على جهاز الكمبيوتر الخاص بالمستخدم وكل شيء تحت السيطرة. انها مجرد مبدأ الند للند P2P . ومع ذلك، فإن إدارة هؤلاء المستخدمين هي معقدة للغاية، وقد تكون صعبة جدا . وبالتالي، أصبح ظهور عدة معايير الأمثل (*مساحة الذاكرة، حساب الوقت، وزمن الاستجابة، وحياة البيانات، ...*) تحديا في السنوات الأخيرة. والهدف هذه الأطروحة هو تكييف أبنية P2P لمجموعات من تطبيقات الشبكات الاجتماعية. تم تصميم الهيكل المقترح من قبلنا لتحسين الأداء (*التكلفة البحث، تكاليف الصيانة*) بالمقارنة مع هياكل DOSNs أخرى، وذلك باستخدام بنية هرمية ذات مستويين والقادرة على التعامل مع عدد كبير من المستخدمين بطريقة فعالة.
كلمات البحث : الند للند، الشبكات الاجتماعية، لامركزية البيانات، هندسة نظام P2P

Abréviations

A

API	**A**pplication **P**rogramming **I**nterface

C

CAN	**C**ontent **A**dressable **N**etwork
CPU	**C**entral **P**rocessing **U**nit
CRS	**C**hoice **R**eplicas **S**et

D

DHT	**D**istributed **H**ash **T**able
DNS	**D**omain **N**ame **S**ystem
DoS	**D**enial **O**F **S**ervice
DOSN	**D**ecentralized **O**nline **S**ocial **N**etwork

F

FOAF	**F**riend **O**f **A** **F**riend
FTP	File Transfer Protocol

G

GUID	**G**lobal **U**nique **ID**entifier

H

HTTP	**H**yper **T**ext **T**ransfer **P**rotocol

I

ICQ	**I**nternet **C**hat **Q**uery
IP	**I**nternet **P**rotocol

M

MTTF	**M**ean **T**ime **T**o **F**ailure
MTTR	**M**ean **T**ime **T**o **R**ecover

N

NAT	Network Address Translation

O

OSI	**O**pen **S**ystems **I**nterconnection
OSN	**O**nline **S**ocial **N**etwork

P

P2P	**P**eer to Peer
P4L	P2P four Layers
PDA	**P**ersonal **D**igital **A**ssistant

PSN	**P**rofessional **S**ocial **N**etworks

S

S2S	**S**ensor **To S**ensor
SHA	**S**ecure **H**ash **A**lgorithm
SIC	**S**ervice d'**I**dentification de **C**onfiance
SMS	**S**hort **M**essage **S**ervice
SOAP	**S**imple **O**bject **A**ccess **P**rotocol

T

TCP	**T**ransmission **C**ontrol **P**rotocol
TCP	**T**ake **C**are **P**hase
TTL	**T**ime-**T**o-**L**ive

U

URL	**U**niform **R**esource **L**ocator

Table des matières

Chapitre 1 : État de l'art sur les Réseaux Pair à Pair (*Peer to Peer)*

Chapitre 2 : État de l'art sur les Réseaux Sociaux

Chapitre 3 : Modélisation des réseaux sociaux distribués : Etat de l'art

Chapitre 4 : Proposition d'une architecture P2P pour une application d'OSN

Chapitre 5 : Simulation et évaluation des Performances

Liste des figures

Liste des tableaux

Introduction Générale

Au cours des dernières années, les réseaux sociaux en ligne sont devenus de plus en plus populaire. Ces types d'applications ont revendiqué avec succès leur place parmi les services les plus célèbres sur Internet (*Au 31 mars 2015, Le leader Facebook l'un des plus célèbres plates-formes de réseaux sociaux en ligne compte 1,44 milliards d'utilisateurs actifs par mois*[1]). Concomitant avec cette formidable croissance des plates-formes de réseaux sociaux en ligne, des inquiétudes croissantes des utilisateurs sur leur vie privée et la protection de leurs données sont augmentées. Comme la gestion et le contrôle des données de l'utilisateur sont généralement centralisées, les fournisseurs d'accès aux OSNs ont aujourd'hui un privilège sans précédent pour accéder aux données privées de chaque utilisateur, ils peuvent obtenir un aperçu en profondeur sur les intérêts personnels de leurs utilisateurs, les opinions et les relations sociales, ce qui rend la fuite de la vie privée à grande échelle et l'utilisation abusive des données extrêmement possible, par exemple, LinkedIn a fui des millions de mots de passe de ses utilisateurs [**1**], et Facebook a passé des informations commerciales sensibles des utilisateurs au public sans leurs consentements [**2**]. Les limites posées par la nature centralisée des OSNs ont motivé la communauté de recherche pour développer des architectures OSN alternatives. Le thème principal de ces propositions est de donner aux utilisateurs d'OSN plus d'autonomie en termes de stockage (*les données sont répliquées*) et de contrôler les droits d'accès à leurs contenus. La plupart de ces propositions préconisent une architecture décentralisée peer-to-peer (*P2P est un sous-domaine des systèmes distribués, où chaque nœud peut être utilisé à la fois comme un client et un serveur*), où une infrastructure OSN est formée à travers la participation d'un ensemble d'utilisateurs autonomes qui collaborent avec les autres. En conséquence, les utilisateurs peuvent avoir plus de contrôle sur leur contenu où seront stockés et comment ils seront accessibles. Au lieu de poster une photo ou un commentaire (*donnée personnelle*) sur un serveur distant, ceux-ci sont hébergés sur l'ordinateur de l'utilisateur, ainsi, tout est sous contrôle. Ceci, à son tour, donne aux utilisateurs la liberté de participer à tout OSN sans la nécessité de migrer leurs données entre les différents systèmes.

[1] Source : JDN l'économie demain : http://www.journaldunet.com/ebusiness/le-net/nombre-d-utilisateurs-de-facebook-dans-le-monde.shtml

Cependant, une telle approche de décentralisation des OSNs (*DOSN*[2]) doit relever de nouveaux défis. En particulier, elle doit atteindre une haute disponibilité des données de chaque utilisateur avec un minimum d'overhead de réplication et sans assumer aucun stockage permanent en ligne. Il est nécessaire aussi de fournir des mécanismes pour le cryptage des données des utilisateurs, contrôle d'accès aux données, la synchronisation des répliques. En outre, elle doit permettre le passage à l'échelle pour les grands réseaux sociaux et être résiliente et adaptative pour mieux gérer l'overhead d'un grand nombre de participants réguliers et les attaques d'utilisateurs malveillants.

Récemment, plusieurs chercheurs ont proposé un large éventail d'approches P2P pour les DOSNs, qui diffèrent considérablement dans la manière dont le serveur central de données est substitué et comment l'utilisateur contrôle ses données. Cependant, chacun de ces systèmes souffre de plusieurs lacunes, notamment: d'un succès limité dans la fourniture d'une haute disponibilité des données (le *principale défi de la décentralisation)*, la non-considération des utilisateurs mobiles, et l'overhead élevé.

Le but de ce mémoire est l'adaptation des architectures P2P pour les applications de groupes des réseaux sociaux. En abordant les inconvénients des travaux existants notamment le problème de disponibilité, nous proposons une nouvelle architecture hiérarchique scalable qui adopte une stratégie de réplication avec un minimum d'overhead.

Ce mémoire est structuré en cinq chapitres :

Dans le premier chapitre, nous présentons les différents concepts liés aux réseaux P2P : définitions des concepts de base des réseaux P2P, leurs domaines d'applications, leurs caractéristiques, les différentes architectures ainsi que leurs problèmes courants.

Le deuxième chapitre comprend une étude de réseaux sociaux numériques :

Une définition des réseaux sociaux, leurs types, leurs intérêts, leurs enjeux et leurs perspectives

Le troisième chapitre comprend une étude, une classification et une comparaison des architectures DOSNs les plus intéressantes que nous avons repérées lors de notre étude bibliographique.

[2] DOSN : Decentralized Online Social Network

Le quatrième chapitre a été consacré la description et les détails de la nouvelle architecture DOSN que nous avons proposés.

Le dernier chapitre présente les résultats de simulations que nous avons réalisées afin d'étudier trois aspects de l'architecture proposée, à savoir : le coût de maintenance des répliques, le nombre de sauts et la taille de la table de raccourcis.

Enfin, nous terminerons ce mémoire par une conclusion générale et des perspectives envisagées.

Chapitre 1

État de l'art sur les Réseaux Pair à Pair (*Peer to Peer*)

1. Introduction

Ces dernières années, les systèmes pair à pair (*en anglais Peer-to-Peer ou bien P2P*) sont devenus de plus en plus populaires, cette popularité est dû aux caractéristiques avantageuses offertes par ces systèmes telles que: le passage à l'échelle, tolérance aux panne et le contrôle décentralisé, chaque dispositif peut jouer le rôle d'un serveur en offrant ces ressources aux autres nœuds, d'un client en consommant les ressources des autres nœuds (*Servent*).

Actuellement, la recherche considère ce modèle une vraie alternative au modèle classique client -serveur et contribue à de nombreux travaux dans ce domaine. Parmi ceux-ci, nous pouvons citer le partage des fichiers, le calcul distribué ainsi que les espaces collaboratifs.

Dans ce chapitre, nous présentons les réseaux P2P, nous commençons par définir précisément le concept peer-to-peer, puis nous citons quelques domaines d'applications de ces réseaux, ensuite nous décrivons leurs caractéristiques, et par la suite nous parlerons des différentes architectures du modèle et nous présentons pour chacun un exemple, à la fin nous citons les problèmes courants du P2P.

2. Réseaux Pair à Pair

2.1 Définition

Les systèmes pair à pair (*P2P, de l'anglais "peer-to-peer"*) sont composés d'un ensemble d'entités partageant un ensemble de ressources, et jouant à la fois le rôle de serveur et de client **[3]**. Chaque nœud peut ainsi télécharger des ressources à partir d'un autre nœud, tout en fournissant des ressources à un troisième nœud. Il permet une utilisation maximale de la puissance du réseau, une élimination des coûts d'infrastructure, et une exploitation du fort potentiel inactif en bordure de l'Internet. Il retient l'attention de la recherche, des développeurs et des investisseurs. Les pairs du réseau peuvent être de nature hétérogène : PC, PDA[3], Téléphone portable...

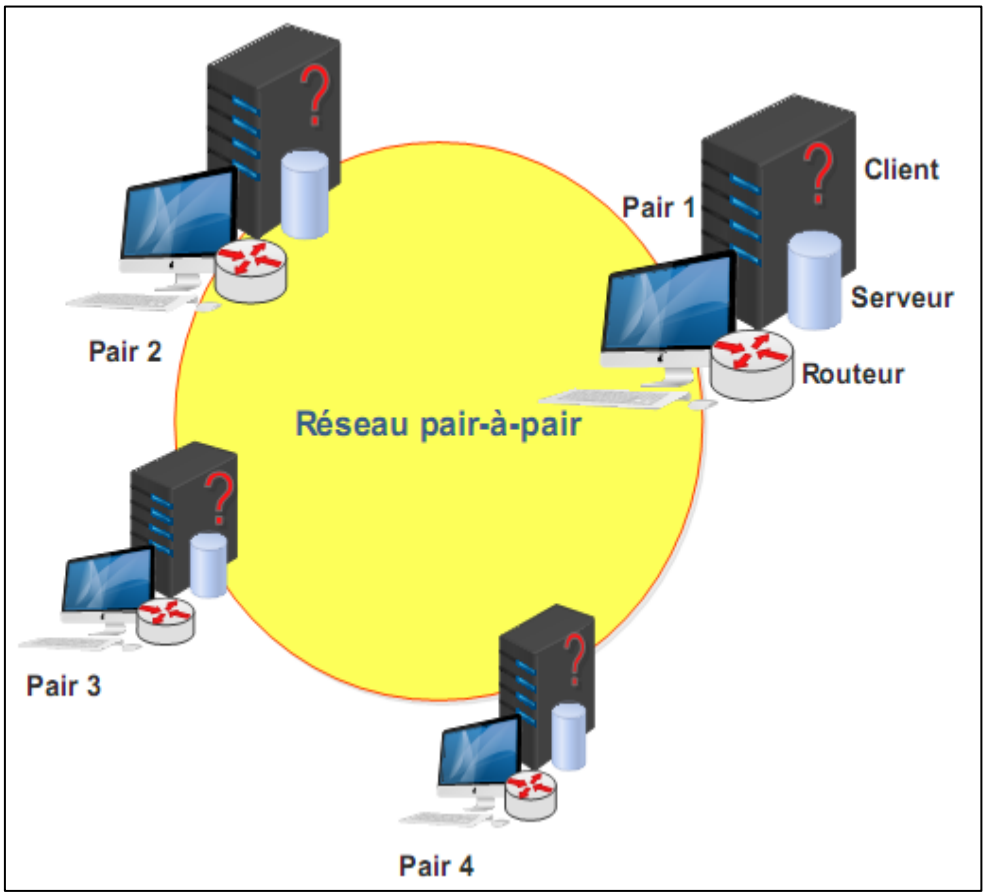


Figure 1. 1: Rôle d'un pair

Rôle d'un pair

Un pair dans un réseau p2p peut jouer différents rôles ; il peut être un:

[3] PDA : Personal Digital Assistant

Client: partie de pair permettant de demander des ressources du réseau pair-à-pair. **Serveur** : partie de pair permettant d'offrir des ressources au réseau pair-à-pair. **Routeur (*niveau applicatif*)**: partie de pair permettant de router les demandes (*ou requêtes*) de ressources.

2.2 Exemples d'Utilisations du P2P

Le modèle P2P n'est pas adapté pour toutes les utilisations. Par exemple une application d'e-commerce ne va pas déporter son module de prise de commande sur ses clients. Par contre, certaines applications peuvent tirer d'énormes avantages dans l'utilisation de cette technologie. Afin de mieux comprendre ce qu'est le P2P, nous allons citer quelques-unes de ses utilisations **[4]**. On peut ainsi citer :

2.2.1. Partage de fichiers

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**[4] ou **TribalWeb**[5].

2.2.2. 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 BitTorrent découpe ainsi un fichier en plusieurs morceaux. Chaque utilisateur téléchargent 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 complète du réseau.

2.2.3 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,

[4] http:// www.qnext.com
[5] http://www.gigatribe.com

le système doit procéder par échanges : si A stocke une quantité x pour B, alors il pourra aussi stocker x chez B. (*ex. Freenet, LeanOnMe*) **[4]**.

2.2.4 Communication et collaboration

Les applications P2P sont très adaptées pour toutes les applications nécessitant une communication entre pairs. Cette communication peut être audio (*Skype* [6]*)* ou par simple message texte (*chat et plateforme Jabber* [7]*)* ou permettre aux utilisateurs de jouer ensemble (*jeux multi joueurs*) ou même de travailler ensemble *(plateforme de travail collaboratif GROOVE*[8]).

2.2.5 Messagerie instantanée

Parmi les applications permises du Peer to Peer on retrouve la messagerie instantanée qui permet à deux utilisateurs du réseau de s'envoyer des messages sans passer par l'intermédiaire d'un serveur. On peut notamment citer : ICQ[9] , AOL[10] Instant Messenger (*messagerie/Chat*) et NetMeeting de Microsoft (*visioConférence en plus de la messagerie*).

2.2.6. Calcul distribué (*Distributed Computing*)

Le calcul distribué est probablement le domaine qui peut tirer les grands bénéfices d'une architecture P2P. Il 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'accroitre le potentiel réseau **[5]**, **[6]**.

Pour un problème qui peut être décomposable et parallèlisable , c'est-à-dire qu'il peut être diviser en plusieurs parties susceptibles d'être résolues d'une manière quasi- simultanée par plusieurs CPUs, il sera plus profitable de le faire sur une architecture P2P . Dans ce type d'application, les peers coopèrent ensemble pour la réalisation du traitement désiré, ce qui permettre l'utilisation des ressources des ordinateurs non exploitées des , telles que la puissance du processeur ou l'espace de stockage.

[6] http:// www.skype.com
[7] http:// www.jabber.org
[8] http://www.groove.net
[9] ICQ (Internet Chat Query) :https://www.icq.com
[10] AOL (America Online) :http://www.aim.com

Seti@Home (*Search for Extra Terrestrial Intelligence*) (*1999*) qui s'inscrit dans un projet américain de recherche d'intelligence extraterrestre est un exemple d'application P2P basé sur le traitement distribué. utilisant des ordinateurs branchés sur Internet.

2.2.7. Moteur de recherche WEB

InfraSearch, par exemple fonctionne de la même façon que la version initiale de Gnutella. Les nœuds qui veulent faire partie du réseau peuvent faire fonctionner un élément du logiciel dans leur base de données. Ils partagent alors les résultats de recherche en temps réel, retournant leur information à n'importe quel utilisateur faisant une requête via InfraSearch **[3]**. Ceci est différent des moteurs de recherche traditionnels, qui dispose de « spiders[11] » électroniques balayant le Web pour le contenu et qui stocke ces résultats dans une énorme base de données. Ce modèle garde souvent des résultats datant de quelques jours à quelques mois dans ses bases, alors que le contenu Web est constamment en train d'évoluer.

Les moteurs de recherche traditionnels ont aussi des problèmes de retour sur les «contenus dynamiques», des pages créées à la volée par l'e-commerce et autre société Web en fonction des résultats de recherche des visiteurs. Aussi, le logiciel Gnutella pourrait par exemple trouver un lien vers une page décrivant une configuration *Dell Computer* créée spécifiquement pour cette recherche, et Dell pourrait retourner directement le résultat. Les moteurs de recherche traditionnels ne peuvent pas pointer vers des liens de ce type, car ils sont limités à des pages statiques.

2.2.8. Transactions financières

Yahoo lance aux Etats-Unis fin 2001 un service de paiement en ligne directement de particulier à particulier. L'offre (*Yahoo! PayDirect*), en partenariat avec la banque HSBC[12] Holdings, en apportant l'ergonomie du peer-to-peer, autorise les paiements en ligne de personne à personne sans tenir compte des frontières. Quelle que soit la monnaie ou le moyen de paiement utilisé en réalité, un simple email permet d'ordonner un virement depuis le compte Yahoo! PayDirect préalablement ouvert par l'utilisateur chez HSBC **[3]**.

La sécurité des ressources doit être assurée lors de leur transit d'une machine à une autre, particulièrement sur des réseaux étendus. Cela passe par un système d'authentification,

[11] Spider: est un logiciel qui explore automatiquement le Web. Il est généralement conçu pour collecter les ressources (pages Web, images, vidéos, documents Word, PDF ou PostScript, etc.), afin de permettre à un moteur de recherche de les indexer.
[12] HSBC : Hong Kong & Shanghai Banking Corporation

d'une part et de chiffrement (*afin de contrer les attaques visant à s'approprier les données*) d'autre part.

2.2.9. Développement

Sun s'est fortement engagé dans le développement de la technologie P2P en "open source" (*logiciel libre*) en essayant d'en prendre le leadership avec le Projet JXTA. La plateforme de JXTA normalise la façon dont les peers (*nœuds*) se découvrent, annoncent des ressources, communiquent les uns avec les autres, et coopèrent les uns avec les autres pour constituer des groupes sécurisés.

La plateforme JXTA P2P permet aux développeurs d'applications de créer et de déployer des services interopérables. De son côté Microsoft avec .NET et le Visual Studio .NET facilitent le développement des applications pair à pair en offrant un modèle de programmation de client, le support pour les services Web, et des classes de gestion de réseau faciles à utiliser, tous dans un même environnement de développement. Mais une évolution importante et ayant de plus en plus de succès reste les Web Services. Il s'agit là de fournir et d'utiliser des services directement entre peers. C'est en fait une technologie permettant à des applications de dialoguer à travers Internet, et ceci indépendamment des plates-formes et des langages sur lesquelles elles reposent. Les Web Services s'appuient sur un ensemble de protocoles utilisant des méthodes d'invocations standardisés, comme SOAP[13].

3. Caractéristiques des systèmes P2P

Dans cette section, nous présentons les principales caractéristiques que présente le modèle peer-to-peer **[6]:**

- **Décentralisation:** le fait que chaque nœud gère ses propres ressources permet d'éviter la centralisation de contrôle. Un système P2P peut fonctionner sans avoir aucun besoin d'une administration centralisée ce qui permet d'éviter les goulets d'étranglements et d'augmenter la résistance du système face aux pannes et aux défaillances.
- **Passage à l'échelle:** il s'agit de faire coopérer un grand nombre de nœuds (jusqu'à des milliers ou des millions) pour partager leurs ressources tout en maintenant une bonne performance du système. Cela signifie qu'un système P2P doit offrir des méthodes bien adaptées avec un environnement dans lequel il y a un grand volume de données à partager,

[13] SOAP : Simple Object Access Protocol

un nombre important de messages à échanger entre un grand nombre de nœuds partageant leurs ressources via un réseau largement distribué.

- **Auto-organisation:** puisque les systèmes P2P sont souvent déployés sur l'Internet, la participation d'un nouveau nœud à un système P2P ne nécessite pas une infrastructure couteuse. Il suffit d'avoir un point d'accès à l'Internet et de connaitre un autre nœud déjà connecté pour rejoindre le système. Un système P2P doit être un environnement ouvert ; c'est-à-dire, un utilisateur sur un nœud doit être capable de connecter son nœud au système sans avoir besoin de contacter une personne ou de passer par une autorité centrale.
- **Autonomie des nœuds:** chaque nœud gère ses ressources d'une façon autonome. Il décide quelle partie de ses données à partager. Il peut se connecter ou/et se déconnecter à n'importe quel moment. Il possède également l'autonomie de gérer sa puissance de calcul et sa capacité de stockage.
- **Hétérogénéité :** à cause de l'autonomie de nœuds possédant des architectures matérielles et/ou logicielles hétérogènes, les systèmes P2P doivent posséder des techniques convenables pour résoudre les problèmes liés à l'hétérogénéité de ressources.
- **Dynamique**: à cause de l'autonomie de nœuds, chacun peut quitter le système à n'importe quel moment, ce qui fait disparaitre ses ressources du système. De nouvelles ressources peuvent être ajoutées au système lors de la connexion de nouveaux nœuds. Alors, à cause de l'instabilité de nœuds, les systèmes P2P doivent être capables de gérer un grand nombre de ressources fortement variables. La sortie d'un nœud du système (*ou la panne d'un nœud*) ne doit pas mettre le système en échec. Elle doit être tolérée et avoir un "petit" impact sur la performance de tout le système.

4. Client-serveur versus P2P

Le modèle P2P est une alternative au modèle client-serveur classique. En effet, dans le modèle client-serveur, les services sont centralisés. Les clients ne disposent pas d'informations mais sont capables d'aller en chercher sur les serveurs qui sont situés sur des machines puissantes afin de pouvoir servir plusieurs clients en même temps. Par conséquent, toute panne du serveur entraîne l'arrêt total du fonctionnement du système dans sa globalité **[7]**. En contraste, dans un modèle P2P, les pairs peuvent jouer en même temps le rôle de serveur et le rôle du client et sont à égalité de devoirs et de droits : chacun peut rendre accessible des ressources dont il dispose et exploiter des ressources fournies par d'autres et ceci sans aucun contrôle central ce qui rend le système flexible et tolérant aux pannes. Ainsi,

en cas de panne d'un pair, le dysfonctionnement reste localisé et le système continue tout de même à fonctionner.

Une autre différence contrastant les deux modèles vient du fait que les performances du modèle client-serveur se dégradent au fur et à mesure que le nombre d'utilisateurs augmente, alors qu'une des caractéristiques des réseaux P2P est que la qualité et la quantité des données disponibles augmentent à mesure que le nombre d'utilisateurs augmente. La valeur du réseau augmente donc avec sa popularité. Cependant, dans le modèle client-serveur, des solutions pour les problèmes relatifs au passage à large échelle sont bien connues, utiliser des machines plus puissantes par exemple mais ceci a un coût considérable.

Par contre, les systèmes P2P décentralisés ont souvent besoin des solutions algorithmiques pour résoudre les problèmes de passage à large échelle puisqu'il n'y a rien de centralisé pour ajouter plus des ressources informatiques. Ces algorithmes distribués ont tendance à être parmi les plus difficiles à développer car ils nécessitent des décisions locales à faire sur chaque pair, généralement avec peu de connaissance globales. D'un autre côté, le modèle client-serveur offre plus de sécurité que le modèle P2P. En effet, le modèle client-serveur permet l'identification des clients et a un contrôle sur les transactions qui se produisent sur le serveur ce qui n'est pas le cas du modèle P2P qui offre à chaque pair une autonomie significative sans possibilité de contrôle et lui permet de se connecter de manière intermittente avec des adresses IP variables. En conséquence, l'identification de pair est difficile pour reconnaître les membres qui auraient eu des comportements éventuellement malveillants. Finalement, il est important de préciser qu'il est essentiel qu'un nombre minimal des pairs soit disponible pour que le système P2P continue à fonctionner ce qui n'est pas le cas pour le modèle client-serveur qui continue à fonctionner aussi longtemps que le serveur conserve le service en cours d'exécution.

5. Répartition du trafic Internet

Entre 1993 et 2006, la répartition du trafic sur Internet a bien évolué avec l'accroissement sans partage du P2P sur le réseau. Ce graphique montre cette évolution en comparant le trafic Internet généré par le Web, les emails, le FTP et surtout le P2P **[8]**.

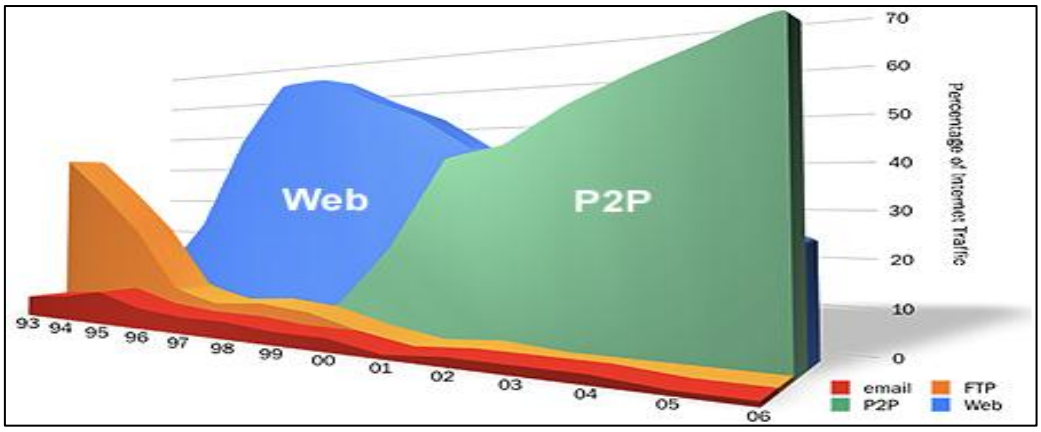


Figure 1. 2: Trafic global sur Internet (1993-2006)

Entre 50% et 65% du trafic de téléchargement (*download*) est un trafic P2P ,et entre 75% et 90% du trafic de téléversement (*upload*) est un trafic P2P.

6. Modèle en couches et le P2P

De point de vue couche réseau, les systèmes P2P sont situés au niveau de la couche 7du modèle OSI[14] (*couche application*).Ainsi, vu que les systèmes P2P assurent un routage de niveau applicatif, un réseau P2P peut être considéré comme un **réseau virtuel** (*ou overlay*) situé au-dessus du réseau physique **[9]**. En général, il existe deux types de nœuds dans un réseau: des terminaux (*nœuds sources et destinations des messages*) et des nœuds de routage de messages.

Les systèmes P2P peuvent être présentés par un modèle en couches par des sous-couches de la couche application :

- La Couche systèmes P2P (*exemples: Gnutella, DHT,…*) sert à localiser les pairs et/ou les ressources à télécharger et/ou distribuer les contenus des ressources.
- La Couche applications P2P qui sont :
  - ✓ Applications centrées **données** : partage de données (*fichiers mp3, autres fichiers,…*)
  - ✓ Applications centrées **utilisateurs** : Messagerie instantanée (*Skype, jeux,…*)
  - ✓ Applications centrées **calcul** : Les grilles de calcul global (*SETI@home, Groove,…*)

[14] OSI :Open Systems Interconnection

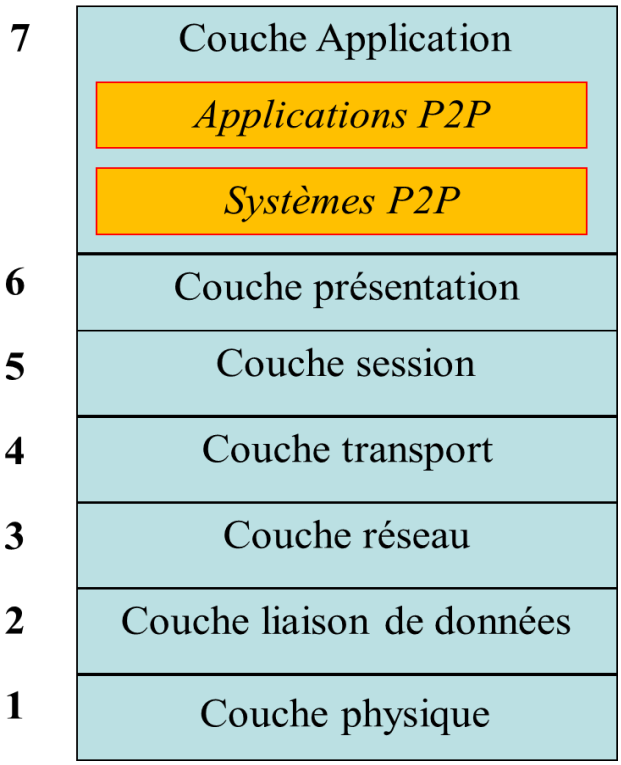


Figure 1. 3:Modèle en couche et le P2P

7. Classification des réseaux

Le P2P est une branche des systèmes distribués dans la classification des systèmes informatiques comme le montre la figure suivante.

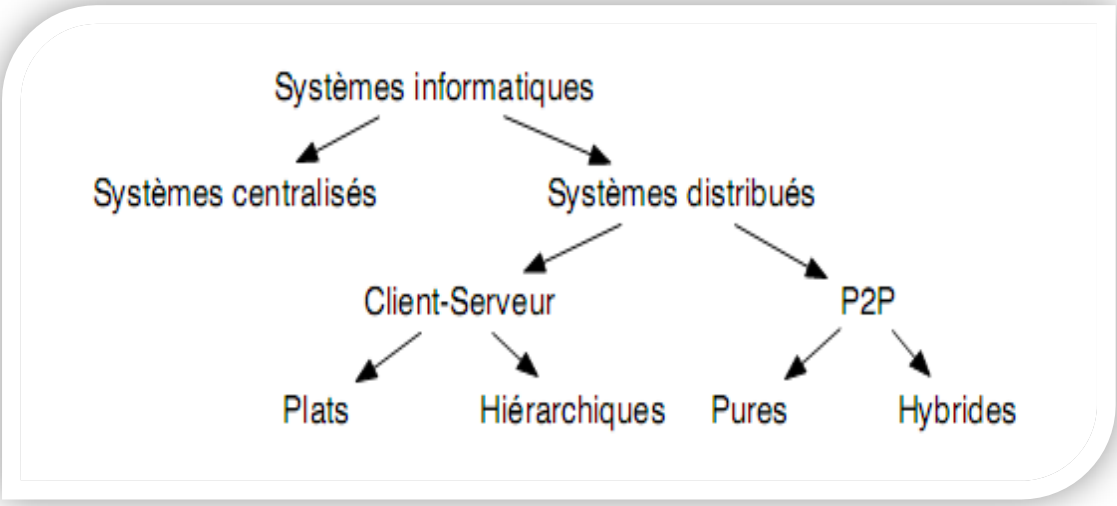


Figure 1. 4:Taxonomie des systèmes informatiques

Plusieurs classifications des réseaux P2P selon plusieurs critères ont été proposées dans la littérature, nous présentons les principales.

7.1. Selon le degré de structuration

Dans cette catégorie, on distingue deux types d'architectures :

7.1.1.Architecture structurée

Les réseaux structurés imposent des contraintes sur les endroits où sont hébergés les contenus. Ils utilisent des tables de hachage distribuées (*DHTs*) pour gérer les opérations de recherche. Une entrée dans la table contient la paire (*clé, valeur*), où la clé est l'identifiant associé à un objet partagé et la valeur correspond à l'information de localisation d'objet recherché. P4L **[10]**, Cycloid **[11]**, Chord **[12]**, CAN **[13]** et EZSearch **[14]** sont des exemples des réseaux structurés. Ils sont basés sur le concept de table de hachage distribuée (*DHT*). Avec l'approche DHT, chaque nom d'entité dans le système peut être met en cohérence sur le même espace de recherche (*identificateur*), en utilisant une fonction de hachage telle que SHA-1 **[15]** ou SHA-2. Cependant, toutes les entités dans le système ont une vue consistante de cette correspondance **[16]**. Étant donné cette vue consistante, plusieurs structures d'espace de recherche sont définies pour localiser ces entités. A titre d'exemple, dans Chord, l'espace de recherche est basé sur une topologie en anneau. Dans P4L, il est structuré sous forme d'anneaux hiérarchiques.

7.1.2. Architecture non structurée

Dans les réseaux pair à pair non structurés, les nœuds sont strictement autonomes et ne sont responsables que de leurs propres ressources. Il n'y a à priori aucune relation entre le nom d'un objet et le pair qui en est responsable. Gnutella **[17]**, Napster **[18]**, ainsi que Bittorrent **[19]**, sont des exemples de réseaux non structurés. Ils sont généralement basés sur un index global (*partiellement centralisé*), ou bien ils utilisent des algorithmes de diffusion pour localiser et découvrir d'autres peers ou ressources. La maintenance de l'architecture de ce type de réseau est globalement simplifiée, tandis que la scalabilité représente un grand obstacle.

7.2. Selon le degré de centralisation

Dans cette catégorie, on distingue trois types d'architectures :

7.2.1. Architecture centralisée

Le représentant le plus connu de ce mode de P2P est sans doute Napster. Cette architecture se compose d'un unique serveur, dont le but est de recenser les fichiers proposés par les différents clients. Contrairement au mode (Client/Serveur) , ce serveur centralisé ne dispose pas de fichiers. Il dispose principalement de deux types d'informations : celles sur les fichiers (*nom, taille, ...*), et celles sur l'utilisateur (*nom utilisé, @IP, nombre de fichiers, type de connexion, ...*). De côté du client : une fois connecté, il peut faire des recherches comme avec un moteur classique. On obtient alors une liste d'utilisateurs disposants de la ressource désirée. Le reste est complètement indépendant du serveur et représente la seule partie P2P du logiciel.

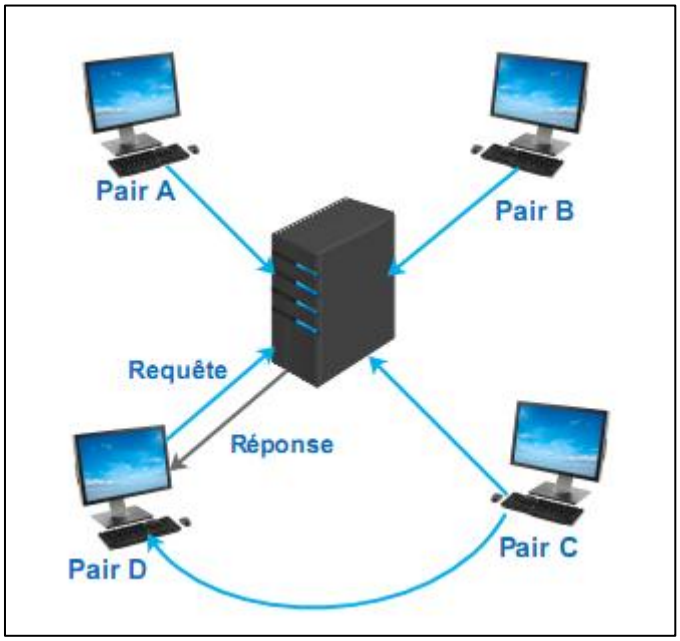


Figure 1. 5: P2P Centralisé

7.2.2. Architecture décentralisée

Un réseau décentralisé (*distribué*) n'utilise aucun serveur. Chaque utilisateur est à la fois un client et un serveur (*Servent* = ***Serv****eur* + *Cli****ent***)). Tous se découvrent et se connectent dynamiquement entre eux en relayant les requêtes d'ordinateur à ordinateur. Cette architecture présente l'avantage d'être robuste, mais elle génère en revanche un trafic important et la recherche de fichier est coûteuse. Chaque requête est adressée à chaque utilisateur connecté, chacun d'entre eux faisant de même, ce qui peut prendre un temps

considérable si des milliers d'utilisateurs sont présents **[16]**. L'architecture décentralisée tend également à être insensible aux défaillances, car l'arrivée ou le départ d'un peer donné n'affecte en rien le reste du système.

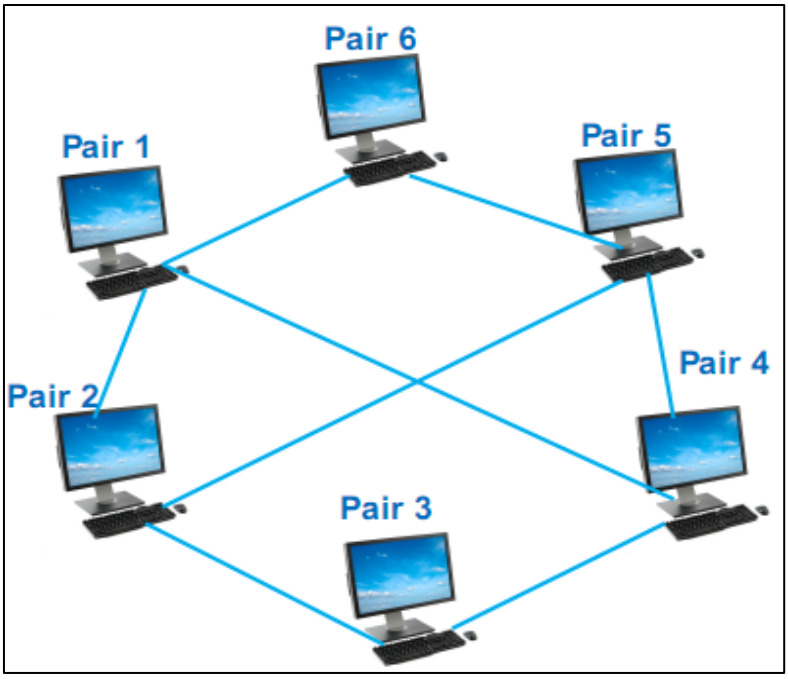


Figure 1. 6: P2P décentralisé

7.2.3. Architecture hybride

Le réseau hybride (*ex. Gnutellav0.6*) est plus complexe à mettre en œuvre, il combine à la fois le réseau centralisé et distribué. Un réseau de ce type s'appuie sur un ensemble de serveurs (*l'ordinateur d'un utilisateur peut devenir un serveur*) gérant un groupe d'utilisateurs suivant l'architecture centralisée **[16]**. Chaque serveur est ensuite connecté à d'autres serveurs suivant l'architecture distribuée. De cette façon, si un fichier recherché par un utilisateur n'est pas indexé par le serveur auquel il est rattaché, celui-ci transmet alors la requête à un autre serveur. Cette architecture permet de bénéficier d'une meilleure bande passante tout en réduisant le trafic des requêtes.

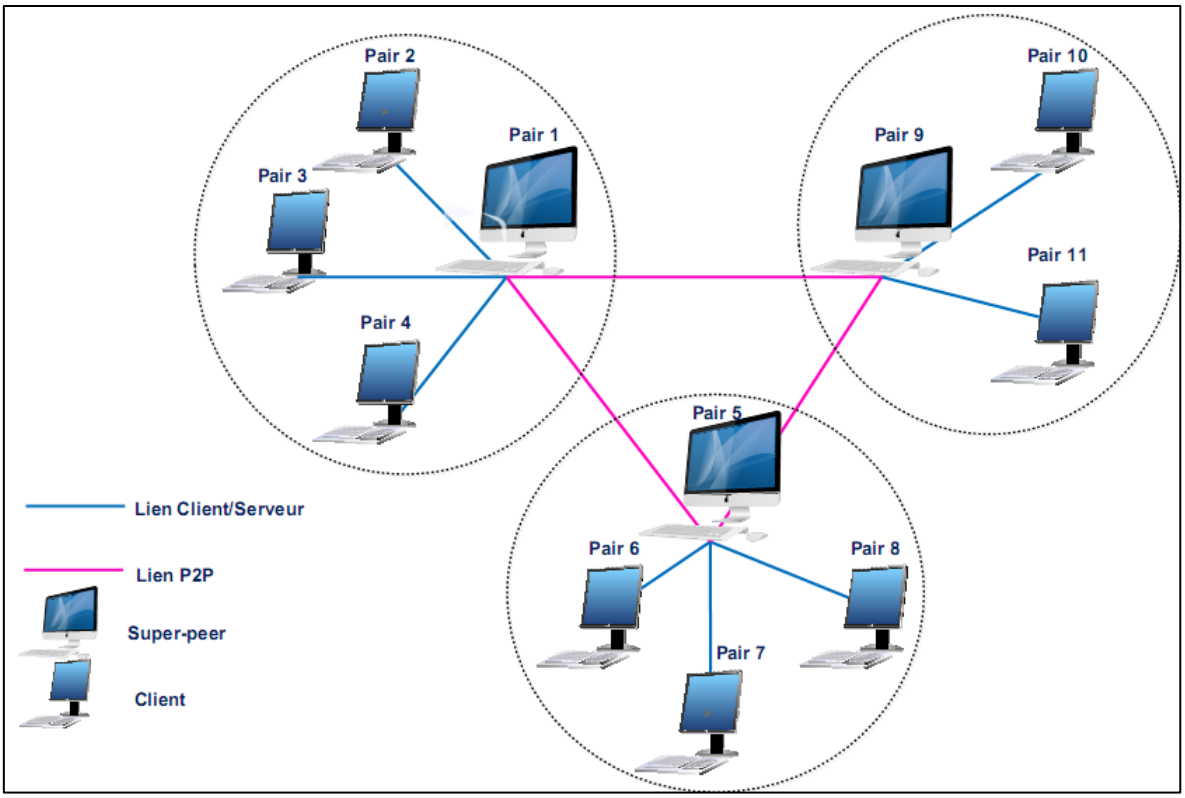


Figure 1. 7: P2P Hybride ou Super Peer

8. Quelques exemples de réseaux P2P : architectures et protocoles

Voici trois exemples de technologie présentés de manière plus détaillée.

8.1 Napster

Napster est l'un des plus vieux et plus célèbre système P2P, il est destiné au partage de fichiers musicaux . C'est un système P2P hybride dans lequel la recherche d'entités est centralisée . Dans Napster, les utilisateurs ont recours à un serveur central possédant l'index de tous les utilisateurs connectés, ainsi que l'index de leurs données partagées **[5,7]**. Le serveur se charge de mettre en relation un peer demandeur avec un peer possédant les fichiers désirés. Le stockage des fichiers demeure sur les machines utilisateurs. Le processus de transfert s'effectue directement entre les peers sans passer par le serveur central.

Napster utilise le protocole Napster_entre les pairs et le Serveur Napster central: pour s'enregister , pour partager des fichiers (1) et pour trouver des fichiers (2),et le protocole HTTP entre les pairs pour le transfert de fichiers (*téléchargement*) (3).

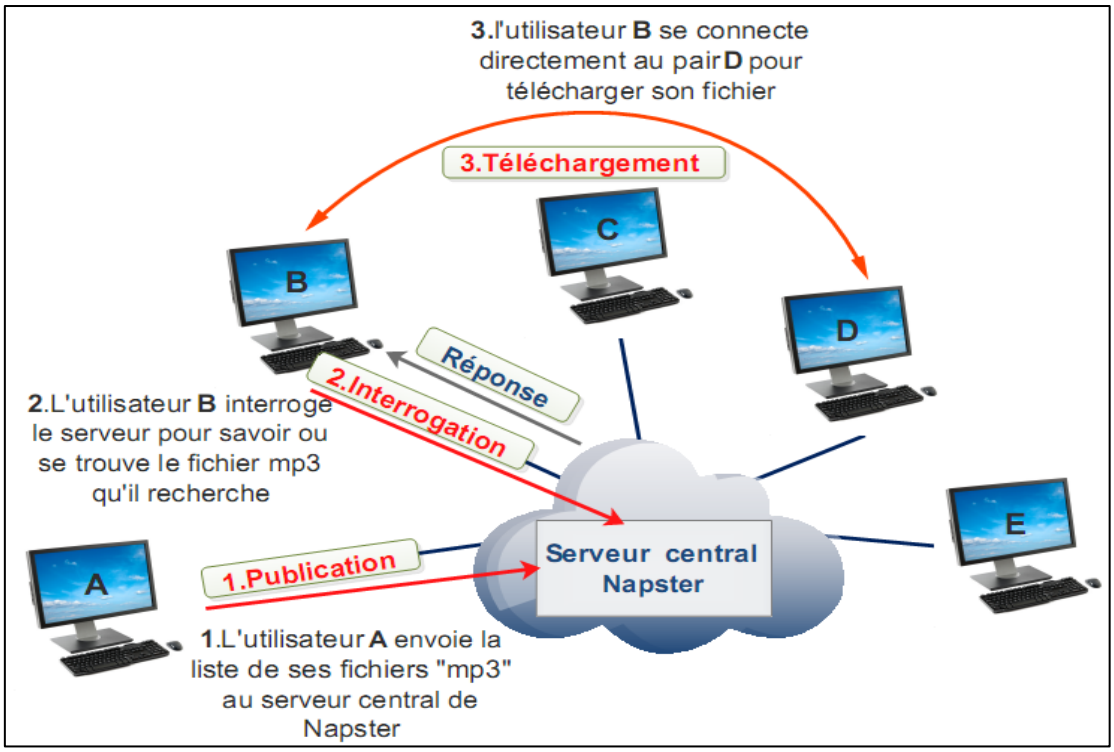


Figure 1. 8: Architecture de Napster

Principe de fonctionnement

Le principe de fonctionnement de Napster est le suivant : à chaque connexion de l'utilisateur de l'application Napster, l'application envoie au serveur l'adresse IP assignée à la machine de l'utilisateur, ainsi que la liste des titres mp3 qu'il désire partager. Ceux-ci sont alors stockés dans un répertoire (centralisé sur un serveur). Si un autre utilisateur Napster cherche un titre particulier, le serveur Napster fournit la liste des IP des utilisateurs en ligne qui le possèdent (*comme un serveur Google*) et l'utilisateur demandeur peut ainsi directement télécharger le titre sur un autre utilisateur en ligne (*Si un fichier est disponible sur plusieurs pairs, le client mesure la qualité de la connexion (ping) pour ces pairs et choisit celui qui a les meilleures performances*).

L'avantage principal de Napster est la localisation rapide et efficace *O(1)* ; (*vue globale de tous les pairs*). Par contre, il présente plusieurs inconvénients :

- Si le serveur tombe en panne, on ne plus découvrir des fichiers.
- Congestion du serveur puisqu'il reçoit un grand nombre de requêtes.
- Taille du répertoire central : O(*n*) avec *n*: nombre de pairs (*La taille du répertoire augmente suivant le nombre de pairs problème de passage à l'échelle (scalabilité)*).
- Vulnérabilité aux attaques (*un serveur central peut être facilement attaqué*).

8.2. Gnutella

Contrairement à Napster, Gnutella est un protocole de recherche P2P purement décentralisé. Il a pris la suite en résolvant le problème de centralisation de Napster. Lors d'une recherche, le peer demandeur envoie la requête à ses voisins, qui font de même et ainsi de suite, jusqu'à atteindre les nœuds à distance 7 du demandeur, cette distance est l'équivalent du nombre maximum de sauts pour une requête HTTP **[16]**.

Principe de fonctionnement

Chaque poste d'un réseau Gnutella est à la fois client et serveur, on les appelle les servents, car le protocole Gnutella à la particularité de supporter le transfert d'informations bidirectionnel. Pour devenir membre du réseau, il suffit d'installer une application Gnutella sur son poste, et de se connecter à un nœud Gnutella, généralement ce nœud est connu par avance (*adresse IP fixe*). Bien que ce protocole supporte une architecture client-serveur, son principal attrait est d'être parfaitement décentralisé, et de ne pas solliciter de serveur particulier. Ce type de structure présente un avantage marquant : là où la panne d'un serveur perturberait sérieusement le fonctionnement d'un système basé sur l'architecture client-serveur, la panne (*ou la mise hors ligne*) d'un ou de plusieurs servents Gnutella n'aura qu'un faible impact, grâce à la redondance des liaisons entre servents.

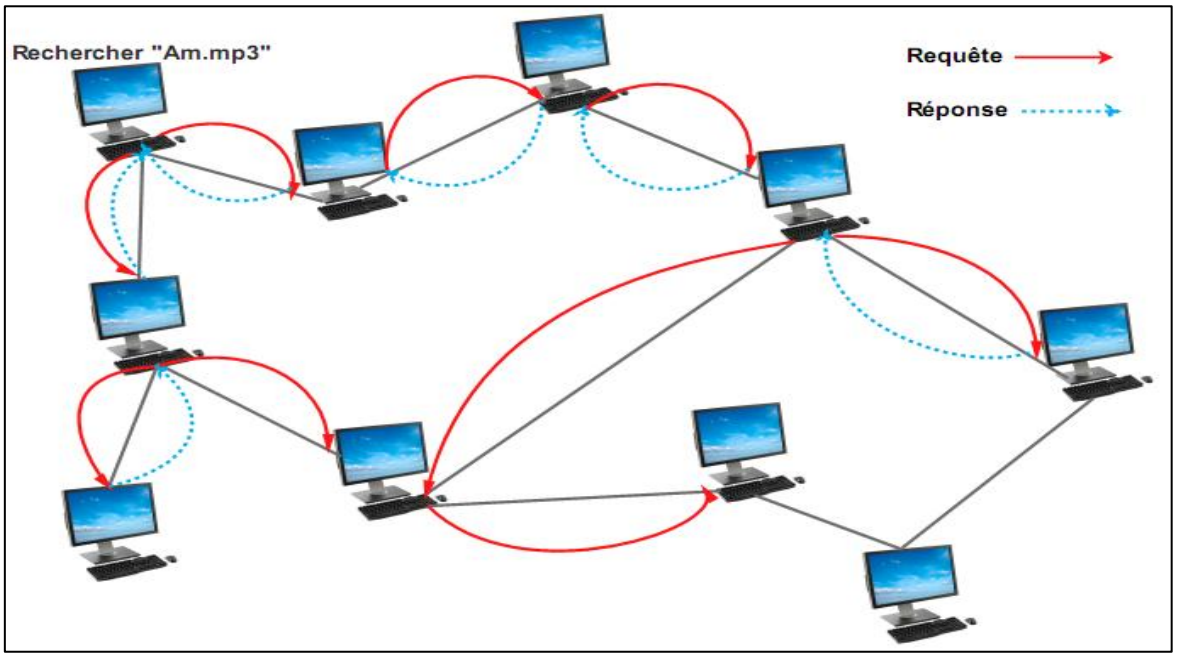


Figure 1. 9: Architecture de Gnutella

Types de requêtes (Descripteurs)

Le protocole Gnutella fonctionne au moyen de cinq descripteurs principaux, qui permettent la transmission des informations entre les servents (*ou nœuds*) du réseau, et d'un ensemble de règles qui régissent l'échange de ces descripteurs. Les voici :

Types	Description	Information
Ping	Annonce la disponibilité, et lance une recherche de pair	Vide
Pong	Réponse à un ping	IP et N° de port. Nombre et taille des fichiers partagés
Query	Requête	Bande passante minimum demandée. Critères de recherche
QueryHit	Réponse à query si on possède la ressource	IP + N° de port + Bande Passante. Nombre de réponses + descripteur
Push	Demande de téléchargement pour les pairs derrière un firewall	IP du pair ; index du fichier demandé. Adresse IP + N° de port où envoyer le fichier

Tableau 1. 1:Cinq descripteurs utilisés dans le protocole Gnutella

Ajout d'un nouveau pair

Un pair devient membre d'un réseau Gnutella en établissant en moyenne trois connexions TCP à d'autres pairs actifs du réseau, dont les adresses IP peuvent être obtenues d'un serveur d'initialisation (*bootstrap server*) . Si un pair était déjà membre du réseau, il peut rejoindre le réseau de nouveau en se connectant aux pairs auxquels il était connecté lors de la dernière session. Les pairs auxquels le nouveau pair est connecté sont appelés les *pairs voisins* de ce pair.

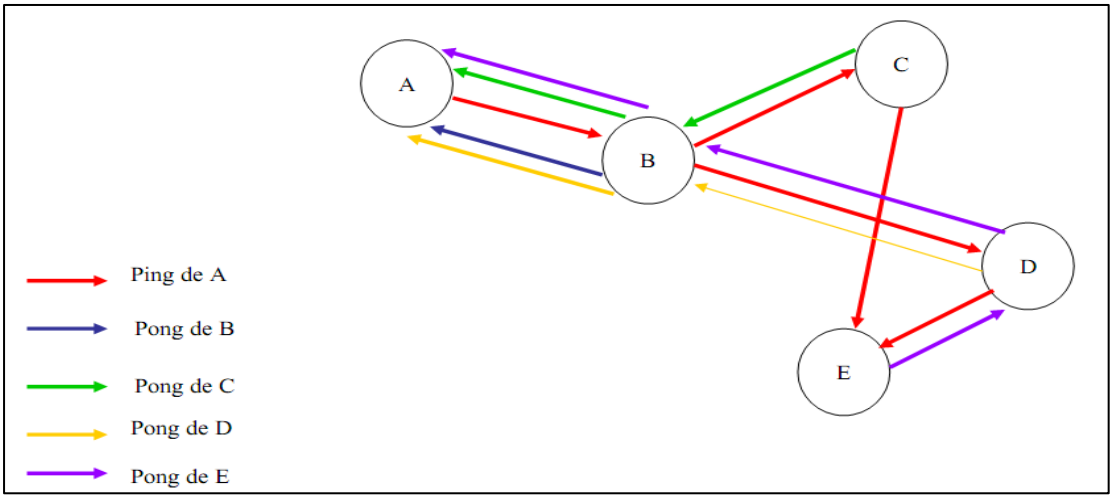


Figure 1. 10:Ajout d'un nœud dans le protocole Gnutella

Découverte des pairs

Étant connecté, un pair envoie périodiquement des requêtes (*Ping*) à ses voisins. Cela permet de sonder le réseau à la découverte d'autres pairs. Tous les pairs qui reçoivent un Ping répondent avec un Pong qui contient l'adresse IP et la quantité de données partagée.
La propagation des Ping est semblable à la propagation des messages Query *(voir partie suivante*).

Recherche de fichiers par Gnutella

Lorsqu'un pair cherche un fichier, il lance à tous ses voisins, une requête (*Query*) comportant des mots clés spécifiant le fichier désiré. Chaque pair voisin transfère la requête à ses pairs voisins, qui eux-mêmes transfèrent la requête à leurs voisins, et ainsi de suite… Si l'un des pairs possède le fichier en question, il répond à la requête en renvoyant une réponse (*QueryHit*) au voisin qui lui a transmis la requête, spécifiant dans la réponse où le fichier peut être téléchargé (*son adresse IP et son port TCP*). La réponse remonte de proche en proche jusqu'au pair qui a initialisé la requête. Le pair initiateur de la requête choisit ensuite les fichiers à télécharger en envoyant directement une requête HTTP de téléchargement au pair qui possède le fichier.

Gnutella inonde le réseau pour trouver le fichier désiré. Néanmoins, pour ne pas inonder le réseau durant un temps trop long, un temporisateur **TTL** (*Time-To-Live*) est utilisé (*comme dans le protocole IP*). La valeur attribuée au TTL est généralement égale à 7. Lorsqu'elle arrive à zéro, la requête n'est plus renvoyée. Le coût de la recherche dans Gnutella est *O(n).*

Si le fichier se trouve sur un pair qui est derrière un Firewall, il est impossible d'établir une connexion TCP avec lui. Afin de résoudre ce problème, le pair demandeur envoie un message **Push** à ce pair protégé par le Firewall, pour lui demander de « pousser » le fichier. Le message Push emprunte le chemin inverse du message QueryHit correspondant.

Dans l'exemple décrit sur la figure 1.11, le nœud X recherche la donnée D1, il envoie une requête Query à tous ses voisins, après l'expiration de TTL (*TTL = 0*), la donnée est trouvée dans les nœuds K et G, ces derniers répondent par des Query Hit, le nœud X envoie un Push aux deux nœuds K et G en utilisant leurs adresses IP retournées dans Query Hit.

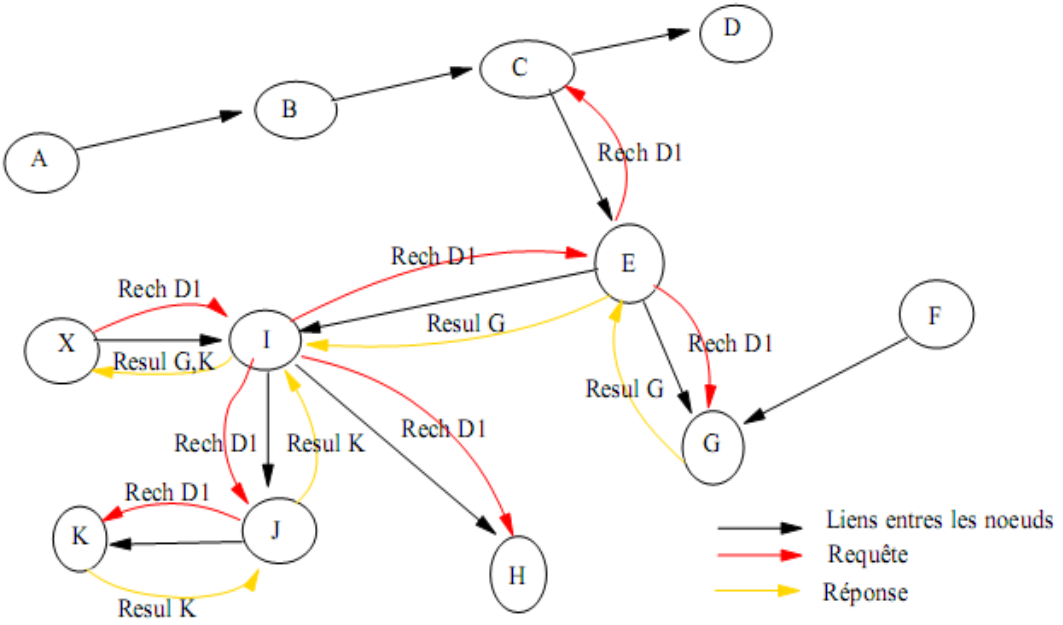


Figure1. 11: Mécanisme de recherche dans Gnutella [5]

L'avantage principal du protocole Gnutella est la simplicité du mécanisme de routage utilisé, qui est basé sur l'inondation (*diffusion*) : Envoi de la requête à tous les voisins, puis à tous les puis à tous les voisins de tous les voisins, etc, à distance bornée afin de ne pas saturer le réseau. L'inconvénient majeur de ce mécanisme, provient de l'expiration du TTL avant le parcours de l'intégralité du réseau, ce qui peut aboutir à l'échec d'une recherche bien que l'objet désiré soit disponible sur le réseau P2P, il passe mal à l'échelle et génère une surcharge du réseau par les trames de broadcast diffusées.

8.2 Chord

Chord **[12]** est un protocole de recherche distribué, qui repose sur une structure en anneau, représentant 2^m valeurs (*m est la taille d'un identifiant*). Typiquement, m vaut 160, l'identifiant d'un pair est un condensat SHA-1 réalisé à partir de son adresse IP et l'identifiant d'une ressource est le condensat SHA-1 de la donnée stockée. Les ressources sont réparties sur les différents nœuds de l'anneau. Une clé K est attribuée au premier nœud immédiatement supérieur (*ou égale*) à K, ce nœud est dit successeur de K et est noté successeur(K).

Pour la recherche d'une ressource dans Chord, le nœud demandeur doit obtenir la clé K de la ressource en hachant son nom, et en contactant successeur(K), pour cela une méthode simple mais peu efficace existe, chaque nœud conserve l'adresse IP de son successeur réel dans le cercle (*successeur de 8 est 14, successeur de 14 est 21,….etc*). Le nœud demandeur

peut alors envoyer un paquet de requête contenant son adresse IP et la clé de la donnée qu'il recherche. Le paquet circule dans le cercle jusqu'à arriver au successeur de la clé recherchée. Ce nœud regarde alors s'il possède des informations correspondant à la clé, et les renvoie directement au nœud demandeur vu qu'il a son adresse IP, le résultat suit le chemin dans le sens inverse.

Dans le but d'accélérer les recherches, une table de routage est utilisée appelée "*finger table*" ou table de raccourcis. Chaque nœud p dispose d'une table de repérage à m entrées, l*a* i^{iemme} entrée ($1 <= i <= m$) contient l'identifiant du nœud succédant p par au moins 2^{i-1}. i.e successeur ($p+ 2^{i-1}$). Chaque nœud possède donc une table de taille O(log N) avec $N \leq 2^m$. (*N représente le nombre de nœuds contenus dans l' environnement pair-à-pair à un moment donné*).

Pour rendre le schéma plus lisible, dans la figure 1.12 , la $1^{ére}$ entrée de la table finger du nœud 8 est le successeur de ($8+2^{1-1}$) qui est le nœud 14, la 2éme entrée correspond au successeur de ($8+2^{2-1}$) à savoir 14, et ainsi de suite pour les autres nœuds.

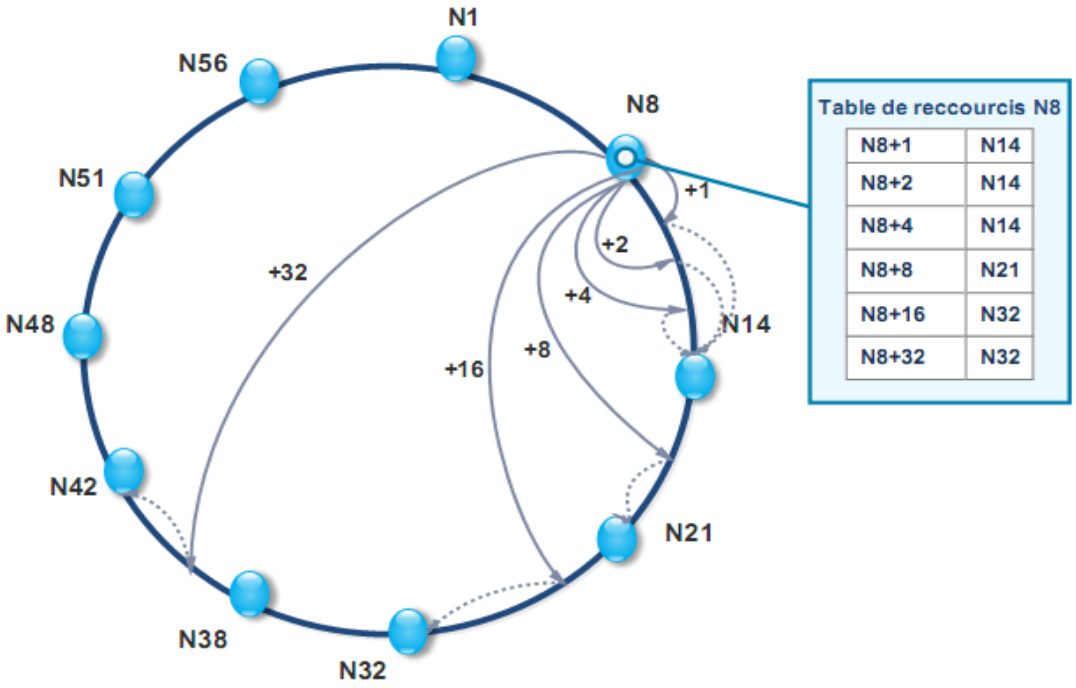


Figure 1. 12:La table de raccourcis (Fingers) dans Chord

Quand un nœud p reçoit une requête de recherche d'une clé K, il vérifie d'abord si elle existe localement. Si oui il renvoie la valeur associée sinon, il recherche dans la table de routage un nœud avec la plus grande valeur inférieure ou égale à la clé cherchée, puis il transmet la requête au nœud sélectionné et appliquer récursivement.

La figure 1.13 illustre comment les clés sont affectées aux nœuds (*l'espace d'adressage*).

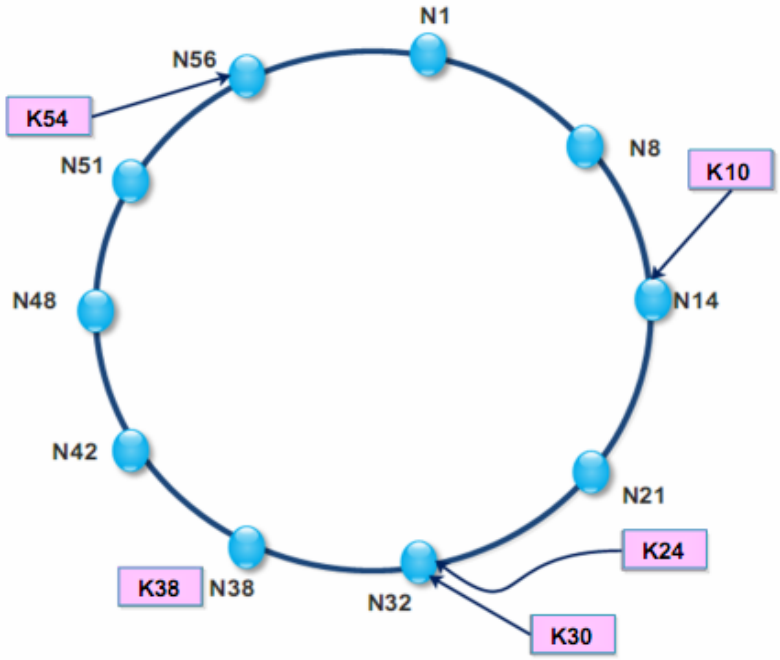


Figure 1. 13: Espace d'adressage de Chord

La figure 1.14 montre la recherche de la clé 54 depuis le nœud 8, tout d'abord le nœud 8 consulte sa table de raccourcis pour déterminer le plus grand prédécesseur de 54, il s'agit du nœud 42, par conséquent la requête est transmise directement vers le nœud 42 qui en prend en charge. Donc la requête de clé K54 est acheminée en trois sauts, en utilisant les tables des raccourcis. Le nombre maximum de nœuds parcourus pour acheminer une requête est alors exprimé en O(ln(N)). La recherche est représentée dans l'algorithme suivant :

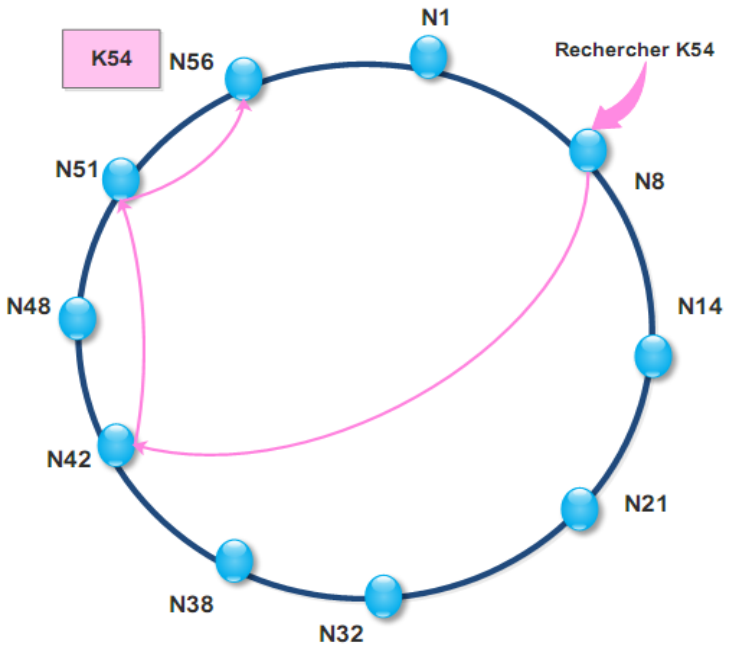


Figure1. 14:Routage d'un objet dans Chord

Algorithme de recherche avec des tables de fingers (Chord)

1 : **Begin**

// Demander le noeud n de trouver le successeur de id

2*:* $n.$ **find successor**(id)

3: if ($id \in (n, successor])$

4: **return** $ccessor$;

5: **else**

6: $n' = closest\_preceding\_node(id)$;

7: return $n'.find\_successor(id)$;

// rechercher la table locale pour le plus grand prédécesseur de "id"

8: $n.$ **closest_preceding_node**(id)

9: for $i = m$ **downto** 1

10: if $(finger[i] \in (n, id))$

11: **return** $finger[i]$;

12: **return** n;

13 : **End**

9.Mécanisme de Bootstrapping

Le bootstrapping est le processus qu'un nouveau pair (*qui désire adhérer à un réseau P2P*) utilise pour découvrir des informations de contact d'un autre pair dans le réseau P2P existant, afin de joindre le réseau. Quatre types de bootstrapping sont cités et résumés dans **[20]** comme suit :

- **Serveurs overlay statiques (*Static Overlay Nodes-Bootstrapping servers*)** : dans un réseau P2P comme Gnutella, le problème de bootstrapping est résolu en plaçant des nœuds statiques dans l'overlay. Ces nœuds de bootstrapping, appelés «caches-pong » dans Gnutella, collectent les informations topologiques et les adresses IP des autres nœuds participants dans l'overlay. Lorsque un nouveau nœud veut joindre l'overlay, les serveurs « caches-pong » renvoient une liste d'adresses des nœuds vu récemment dans l'overlay. Le nouveau nœud va ensuite tenté d'établir des connexions à l'overlay à travers les nœuds de cette liste.
- **Groupe d'adresses dans une cache *(Out-of-bande Address caches)*** : dans un overlay P2P, les nœuds déclarent activement les nœuds appropriés (par exemple, avec de longues

durées de fonctionnement) à des caches basés sur HTTP. Les nouveaux nœuds contactent les caches qui sont accessibles via URLs , afin d'obtenir une liste d'adresses IP.

- **Génération des adresses aléatoires** (***Random Address Probing***) : pour un réseau overlay à grande échelle, des adresses IP aléatoires peuvent être mieux appropriées pour le bootstrapping. Un nœud souhaitant entrer dans le réseau overlay, extrait aléatoirement une adresse IP à partir de l'espace des adresses IP valides, ensuite il tente d'établir une connexion à cette adresse IP sélectionnée en utilisant un port bien connu. Dans le cas où la connexion échoue (*ex. le nœud n'existe pas ou il ne participe pas au réseau*), une autre adresse sera sélectionnée pour un autre essai.
- **Utilisation des mécanismes de la couche réseau** (***Employing Network Layer Mechanism*)** : la découverte des nœuds durant le processus de bootstrapping dépend de la structure logique de réseau utilisé. Si des nœuds existent sur le même segment, ça convient efficacement et parfaitement aux mécanismes implémentés au niveau de la couche réseau de connecter ces nœuds . Dans un réseau supportant le multicast, un groupe multicast peut être utilisé comme bootstrapping.

10. Problèmes courants du P2P

Des nombreux travaux existent dans le domaine des réseaux P2P, mais malgré ça un nombre important de problèmes reste à résoudre d'une manière efficace. Dans cette section nous présentons quelques problèmes courants **[16]**.

- **Freeloading :** les réseaux peer to peer ne peuvent fonctionner correctement que s'il y a participation active de leurs membres. Ainsi, une des pratiques les plus néfastes au fonctionnement de ces réseaux est le freeloading **[21]**. Les freeloaders prennent sans donner; ils bénéficient des ressources partagées sans pour autant partager les leurs **[16]**.
- **Optimisation de routage et proximité physique :** les protocoles P2P sont conçus au niveau de la couche application du modèle TCP/IP. Par conséquent, quelque que soit l'optimisation effectuée au niveau de routage, il reste moins performant par rapport au routage effectué au niveau de la couche réseau. Pour ces raisons, l'optimisation de routage pour les réseaux P2P reste toujours un domaine très actif. La recherche s'oriente vers la prise en compte de la proximité physique dans les architectures P2P **[22]** .
- **Sémantique de recherche :** de nombreuses approches se penchent aujourd'hui sur la sémantique dans les réseaux P2P pour capter non plus la localité géographique, mais la

localité d'intérêt (*sémantique*). Ces techniques utilisent des similitudes d'intérêts entre les utilisateurs d'une application donnée **[23]**.

- **Média streaming**[15] **en temps réel :** les données multimédia (*audio ou vidéo*) sont très volumineuses et consomment beaucoup de bande passante de les acheminer vers leurs destinations. La solution consiste généralement à construire un arbre de diffusion optimisé pour lequel les algorithmes classiques tels que Djikstra, Belman Ford, ...ne sont pas bien adaptés au contexte P2P **[24]**. Par conséquent, des algorithmes spécifiques pour chaque architecture à part sont considérés **[25]**.
- **Équilibrage de charge :** théoriquement, tous les nœuds d'un réseau P2P jouent le même rôle, tantôt serveur, tantôt client, mais ils assurent aussi avec une responsabilité partagée toutes les fonctionnalités de réseau au sein duquel il appartient (*ex. auto-organisation, routage,...).* Néanmoins, les nœuds de faible capacités tel que les PDAs ne peuvent pas assurer toutes ces tâches. C'est la raison pour laquelle les réseaux P2P hybrides qui sont basés principalement sur des super nœuds (*bonnes capacité et disponibilité*) sont les plus implémentés **[26]**.
- **Organisation d'architecture :** afin de faciliter et d'accélérer certaines tâches fréquemment utilisées tel que le routage, plusieurs architectures spécifiques ont été proposées, nous citons attitre d'exemples : les Skip graphes **[27]**, les graphes de De Bruijn **[28]**,les graphes de Pancake **[29]** et les graphes de Knödel **[30]**...
- **Sécurité et mobilité :** à l'inverse des architectures client/serveur, les mécanismes de sécurité matérielle ne peuvent pas s'appliquer sur des architectures P2P, mais aussi les mécanismes de sécurité logiciels sont difficiles à implémenter à cause de la nature ouverte de ce type de réseau (*pas de contrôle d'accès*). Il est à noter qu'actuellement, un nombre important d'attaques sur Internet est dû aux applications P2P **[31]**. La mobilité est aussi un facteur très important pour les applications actuelles qui utilisent des technologies récentes et avancées *(ex.téléphone portable*), mais elle constitue aussi un obstacle, car les arrivées et les départs fréquents des nœuds dans le réseau sont difficiles à gérer.
- **Pare-feu (*Firewalls*) et Nats :** les réseaux P2P sont confrontés aux pare-feux et NATs[16] conçus principalement pour filtrer le trafic entrant dans un réseau local afin de le protéger **[32]** .Un nœud faisant partie d'un réseau P2P qui est derrière un pare-feu ou un NAT est

[15]Streaming : Mode de transmission de données audio et vidéo. Ces dernières sont transmises en flux continu dès que l'internaute sollicite le fichier plutôt qu'après le téléchargement complet de la vidéo et de l'extrait sonore. http://www.futura-sciences.com/magazines/high-tech/infos/dico/d/internet-streaming-1958/

[16] NAT : Network Address Translation

généralement injoignable pour un téléchargement à partir de l'extérieur, à moins qu'il initie lui-même la connexion pour un téléchargement. Pour cela, plusieurs travaux ont été menés dans ce sens, afin de contourner les pare-feux et les NATs (*ex. STUN, TURN*, ..), mais le problème est toujours posé, car il n'y a pas de solution universelle.

- **Calcul parallèle** : un très grand nombre de problèmes nécessitent un calcul énorme et puissant dans lequel les machines les plus puissantes dans le monde ne peuvent pas l'accomplir toutes seules **[33]**. Des architectures spécialisées (*parfois matérielles*) sont très bien adaptées à ce type d'applications, mais malheureusement, ces architectures ne sont pas à la portée de tout le monde, car elles sont coûteuses et réservées aux grands centres de recherche et d'entreprises commerciales. Pour ces raisons, quelques solutions essayent de faire participer les machines oisives de réseaux Internet afin de réaliser des calculs complexes.

11. Conclusion

Les systèmes P2P représentent un domaine de recherche très actif grâce à leurs actuelles utilisations et leurs applications, ils sont classés selon de critères ;selon leur degré de structuration et selon leur degré de centralisation, chacun de ces réseaux possède des caractéristiques bien différentes, ils offrent beaucoup d'avantages indéniables tels que: la répartition de la charge et la résistance aux pannes, mais aussi quelques inconvénients liés surtout à l'aspect de sécurité des données circulant dans le système.

Le chapitre suivant sera consacré à l'étude des réseaux sociaux qui sont basés sur une architecture centralisée (*Client/Serveur*).

Chapitre 2

État de l'art sur les Réseaux Sociaux

1. Introduction

Les réseaux sociaux sont omniprésents depuis l'avènement d'Internet. Ils permettent aux différents utilisateurs d'interagir en communauté et de se regrouper selon des critères qui leur sont importants. Ces réseaux sociaux sont de différents types. Certains sont connus de tous (*ex. Facebook*[17], *Twitter*[18], *LinkedIn*[19]) et comptent des millions de membres. D'autres exploitent des niches moins connus et peuvent passer relativement inaperçus ou rester confidentiels, tels les réseaux d'entreprise.

Dans ce chapitre, nous allons présenter les OSNs[20], leur définition, leur intérêt, leurs enjeux et leurs perspectives.

[17] http ://www.facebook.com
[18] https ://www.twitter.com
[19] https ://www.linkedin.com
[20] OSN :Online Social Network,

2. Bref historique

Les réseaux sociaux en ligne sont apparus en 2002 avec le site américain Friendster **[34]**, premier à utiliser le principe du cercle d'amis en ligne. Puis vint Myspace **[35]**, créé en 2003, qui dépassa Friendster. À l'origine, l'objectif de Myspace était de permettre à des musiciens de proposer certains de leurs morceaux de musique à l'écoute et de construire leur réseau en devenant amis avec d'autres membres. Ce site est rapidement devenu populaire auprès des jeunes qui l'adoptèrent pour rester en contact avec leurs amis et s'en faire de nouveaux, l'argument musical étant relégué au second plan.

Facebook est né en 2004. Cette plateforme créée par Mark Zuckerberg était à l'origine destinée aux étudiants d'Harvard[21] souhaitant communiquer entre eux. Face à son succès fulgurant, le site est devenu grand public en 2006. Le cas de Facebook marque un tournant dans la démocratisation des réseaux sociaux sur Internet. Pour beaucoup, ce fut une porte d'entrée vers l'univers du Web 2.0[22] et des réseaux sociaux. Premier réseau social au monde en 2008, il comptait 350 millions de membres en 2009 et en compte 500 millions en juillet 2010.

Twitter est apparu en 2006. Cette plateforme repose sur le principe du microblogging[23] : les messages postés par les utilisateurs sont limités à 140 caractères. Très populaire aux Etats-Unis où de nombreuses personnalités y ont un compte.. Le site connaît néanmoins une très forte croissance, supérieure à celle de Facebook. D'après comScore[24] (*société américaine spécialisée dans l'analyse web*), le réseau aurait enregistré une croissance annuelle de 109 % en termes de visiteurs uniques entre juin 2009 et juin 2010 . De nombreux internautes s'en servent comme une véritable source d'information, notamment pour faire de la veille.

Au départ c'était un outil de communication, les réseaux sociaux sont devenus des outils de diffusion de l'information qui, comme nous allons le voir, ont changé notre rapport à l'information et la façon dont nous la découvrons et la partageons. Tous les utilisateurs des réseaux sociaux sont désormais de potentiels producteurs et diffuseurs de l'information.

[21] Harvard univesity :http://www.harvard.edu

[22] Le **Web 2.0** est l'évolution du Web vers plus de simplicité (*ne nécessitant pas de connaissances techniques ni informatiques pour les utilisateurs*) et d'interactivité (*permettant à chacun, de façon individuelle ou collective, de contribuer, d'échanger et de collaborer sous différentes formes*). L'expression **« Web 2.0 »** désigne l'ensemble des techniques, des fonctionnalités et des usages du *World Wide Web* qui ont suivi la forme originelle du web, en particulier les interfaces permettant aux internautes ayant peu de connaissances techniques de s'approprier les nouvelles fonctionnalités du web. Ainsi, les internautes contribuent à l'échange d'informations et peuvent interagir (*partager, échanger, etc*.) de façon simple, à la fois avec le contenu et la structure des pages, mais aussi entre eux, créant ainsi notamment le Web social (*Wikipédia*).

[23] Le microblogging désigne l'activité de création de contenus courts sur des réseaux sociaux de type Twitter.

[24] https://www.comscore.com

3. Définition

Dans la littérature, il existe plusieurs définitions de réseaux sociaux, nous présentons quelques-unes:

Un réseau social est constitué à la fois par un ensemble de personnes liées entre elles et par la force de ces liens. On peut aussi dire qu'un réseau social est un ensemble d'individus liés entre eux par des liens caractérisés par un degré de familiarité variable qui va de la simple connaissance aux liens familiaux les plus étroits **[36]**.

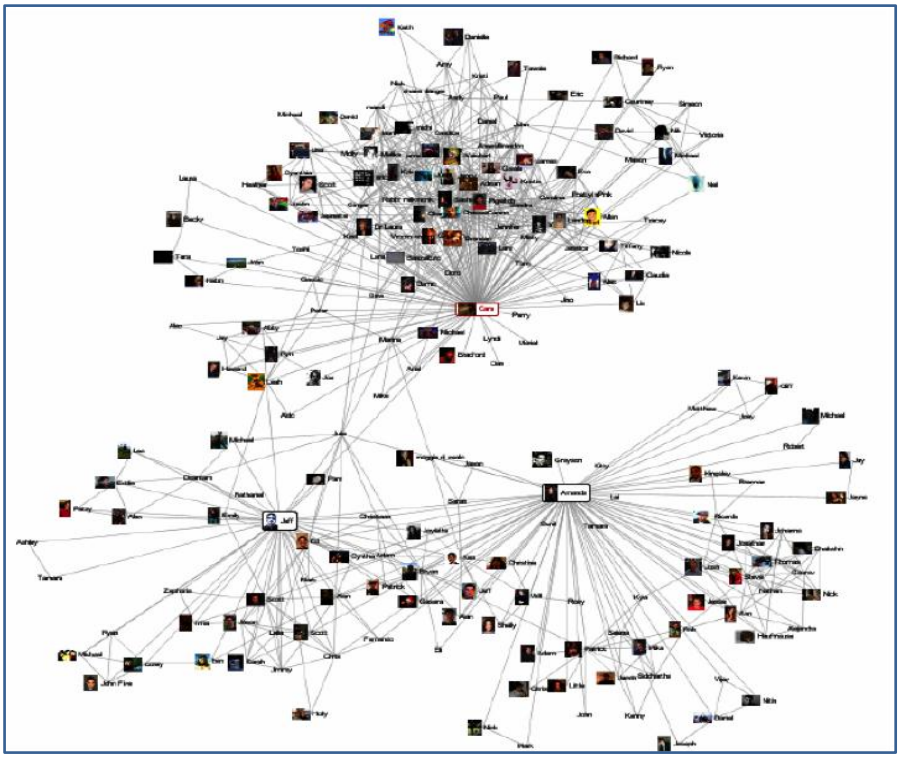


Figure 2. 1 : Représentation des réseaux sociaux [37]

Pour Yahoo (*acteur dans le domaine des réseaux sociaux avec son Yahoo !360*^{0}) , un réseau social est « un terme assez large qui désigne des sites Internet qui aident leurs utilisateurs à créer leur propre profil Internet et à partager une partie de leurs contenus préférés, y compris des photos et de la musique » **[38]**.

Par réseau social, nous entendons donc toute plateforme en ligne dont la finalité est de mettre en relation des membres, et sur laquelle un individu peut s'inscrire librement, construire son propre réseau, produire du contenu, le partager et interagir avec les membres de son réseau. Un membre peut y créer un profil public visible par tous ou privé, visible par ses contacts uniquement. C'est ce profil qui servira de carte de visite à l'internaute sur le réseau

social dont il est membre. L'intérêt de telles plateformes est notamment de pouvoir suivre l'actualité des membres de son réseau et d'éventuellement la commenter **[39].**

Une autre définition dans **[40]**: « Le réseautage social (*distinct du concept de réseau social en sociologie*) se rapporte à une catégorie des applications d'Internet pour aider à relier des amis, des associés, ou d'autres individus employant ensemble une variété d'outils. Ces applications, connues sous le nom de « service de réseautage social en ligne » (*en anglais social networking*) deviennent de plus en plus populaires. Elles peuvent aussi permettre une meilleure distribution artistique, en favorisant la formation de contacts, et en invitant des artistes à assurer une visibilité de leur travail (*ex. musique, vidéo, photographie*) .

Selon la définition proposée par Boyd et Ellison (*2007*) **[41]**, les réseaux sociaux sont des espaces d'échange sur Internet qui permettent aux individus de construire des profils publics ou semi publics, associés à une liste de contacts inscrits sur le même site **[42]** . Actuellement, Les OSNs intéressent différents domaines de recherche tels que : sociologie, psychologie, communication, informatique, marketing, ...

4 . Types de réseaux sociaux

Dans le monde des réseaux sociaux, on trouve plusieurs classifications suivant plusieurs critères, parmi ces typologies:

4.1. Typologisation des OSNs selon l'évolution et l'apparition

Nous retrouvons également une tentative de *typologisation* des réseaux sociaux numérique, selon laquelle les auteurs dressent l'évolution et l'apparition des réseaux. Pour eux, il existe huit types de réseaux : les réseaux généralistes, les réseaux politiques, les réseaux hyper locaux , les réseaux d'universités, d'entreprises, les réseaux associatifs et ceux des communautés d'intérêts. Voici leurs typologies :

- **Réseaux généralistes** : sont plutôt centrés autour de l'individu, et non plus de l'activité professionnelle. Les questions sont plus orientées sur les goûts culturels des membres. Ces sites permettent indirectement de nouer des affinités personnelles, sans pour autant avoir vocation unique d'être des sites de rencontre **[43]**. Ces sites permettent de créer et d'agrandir son cercle d'amis, le plus connu étant Facebook, le plus intime étant Meetic [25] . ils sont organisés sous la forme de groupes spécifiques à des thématiques **[44]**.

[25] http://www.meetic.fr/

- **Réseaux politiques :** du fait que le monde politique s'interroge beaucoup sur les réseaux sociaux et notamment après avoir mis pas mal de temps à apprivoiser les blogs, ils entrent de plein pied dans la réflexion des politiques avec deux expériences : la campagne d'Obama[26] et celle de Ségolène Royale[27]. Ainsi, nous trouvons en France « réseaux sociaux numériques politiques » par exemple : Créateurs du possible [28] de l'UMP et le Coopol [29] (**Coo**pérative **pol**itique) rattaché au PS (*Parti socialiste*).
- **Réseaux hyperlocal :** dans le but de renforcer les liens au niveau local, ce type de réseau a vu le jour. L'idée sous-jacente est de mieux connaitre ses voisins, de promouvoir et d'encourager les solidarités **[36]**. Par exemple : Voisinéo[30] et Peuplade[31] , la Ruche à Rennes[32].De plus, l'hyperlocal peut aussi s'exprimer par la géolocalisation lié à la mobilité permise par les téléphones portables comme les IPhones (*ex : Foursquare*[33]).
- **Réseaux d'universités:** comme Zeeya[34], Réseau campus[35], etnoka[36].
- **Réseaux d'entreprises :** via les OSNs, ces entreprises peuvent évaluer leur présence en termes d'opportunité et de risques.
- **Réseaux associatifs :** les auteurs donnent deux exemples : l'Association Française de Sociologie et le Réseau des Créatifs Culturels.
- **Réseaux sociaux de communautés d'intérêts :** ce sont des réseaux sociaux qui vont proposer des types de relations beaucoup plus spécifiques et qui pour certains s'apparentent à des communautés d'intérêt. D'un côté, il existe des réseaux liés aux âges de la vie comme Beboomer[37] qui s'adresse aux séniors actifs de la génération baby-boom[38]. De l'autre côté, il y a des réseaux sociaux réservés aux enfants comme Globe2child[39].Des

[26] http://bababillgates.free.fr/index.php/comment-obama-a-utilise-le-webmarketing-pour-remporter-lelection-americaine/
[27] Voir la Ségosphère, une cartographie réalisée par Linkfluence sur le site : http://www.observatoire-presidentielle.fr/?pageid=12
[28] http://www.lescreateursdepossibles.com/
[29]https://www.lacoopol.fr/
[30] http://www.voisineo.com/
[31] http://www.peuplade.fr/home/nHome.php
[32] http://beta.ruche.org
[33] https://foursquare.com
[34] http://zeeya.net/
[35] http://reseaucampus.com/
[36] http://etnoka.fr/
[36] http://fr.beboomer.com/
[37] http://fr.beboomer.com/
[38] Le baby-boom ou « pic de la natalité » est une augmentation importante du taux de natalité dans certains pays, juste après la fin de la seconde guerre mondiale.
[39] http://www.globe2child.org/xwiki/bin/view/main/webhome.
[35]http://www.memoree.fr/

réseaux encore plus spécifiques existent comme Memoree[40] dont l'objectif est de conserver les mémoires des siens, vivants ou décédés en partageant des souvenirs.

- **Réseaux de passionnées:** comme les passionnés de chats.

Les auteurs terminent leur typologie en notant que des réseaux comme Culture visuelle[41] qui traite de l'image sous toutes ses formes est déjà plus qu'une communauté d'intérêt et que Knowtex[42] est le réseau des passionnés de la science.

4.2. Typologisation des réseaux sociaux numérique selon la fonctionnalité

D'autres chercheurs ont tenté de classer et de catégoriser les réseaux sociaux. Pascal Faucompré a essayé dans un billet intitulé : *Ras le bol des réseaux sociaux ?*, d'élaborer un classement suivant la fonctionnalité :

- **Networkings :** qui permettent les échanges entre les professionnels.
- **Bloglikes :** ils ressemblent vaguement aux blogs et sont souvent le refuge d'ados en mal de reconnaissance.
- **Spécialisés :** ils regroupent des communautés autour d'un thème bien précis.
- **Micro-blogging :** chat public instantané.
- **Fourres-tout :** ce sont les inclassables qui se servent du collaboratif ou du participatif pour alimenter leur service.
- O**pen-sources :** plateformes qui permettent aux utilisateurs de créer leurs propres réseaux.

4.3. Typologisation selon Wikipédia

Sur Wikipédia, nous trouvons un classement de réseaux sociaux selon trois catégories : Réseaux ouverts, Réseaux sur invitation et Services en ligne de réseautage professionnels qui favorisent les rencontres professionnelles, les offres de poste et la recherche de profils.

4.4. Typologisation selon le point de vue des chercheurs

Thelwall **[45]** catégorise les OSNs selon leurs trois objectifs qu'il nomme respectivement : socialisation, réseautage et navigation (*sociale*) :

- **Réseaux sociaux de socialisation :** cette catégorie se caractérise par son aspect récréatif et conçue pour les loisirs de communication sociale entre les membres. Les connexions

[41] http://culturevisuelle.org/
[42] http:// www.knowtex.com/

sont souvent utilisées pour trouver et afficher des listes d' « amis » existants d'ores et déjà. De plus, les connexions sont souvent utilisées pour trouver d' « amis » existants hors ligne, comme par exemple : MySpace et Facebook et Cyworld[43] (*un monde virtuel coréen lancé en 2001*).

- **Réseaux sociaux de réseautage :** utilisés davantage pour trouver de nouveaux contacts et peuvent servir davantage pour trouver de nouveaux contacts et entrer en connexions avec des personnes inconnues auparavant comme c'est le cas de LinkedIn ou Viadeo, site de réseautage à caractère professionnel
- **Réseaux sociaux de navigation :** comme Digg[44] ou Del.icio.us[45], qui sont des sites de partage de liens Internet (*connu sous le social bookmarking*). Ce type de réseaux est un moyen d'aider les utilisateurs à trouver une information ou des ressources. Autrement dit, nous trouvons des listes de contacts, listes permettant l'accès à l'information et aux ressources associés à ceux-ci. Les membres peuvent soit lire les propositions mises en avant en page d'accueil, soit utiliser la navigation sociale en lisant les informations postées ou recommandées par leurs amis, ou bien pour certains, recouvrir plusieurs objectifs **[36]**.

5. Intérêts des réseaux sociaux

L'écosystème des médias sociaux illustré dans la figure 2.2 s'organise autour de quatre grands usages : la publication, le partage, la discussion et le réseautage. À chacun de ces usages correspondent des services, certains étant dédiés à une fonction bien particulière *(ex : Instagram qui ne sert qu'à publier des photos depuis son Smartphone)*, tandis que d'autres sont plus versatiles *(Tumblr est une plateforme de blogs créée en 2007* à *New York qui est difficile à caser)*.

les quatre grands usages sont complémentaires : les utilisateurs publient des contenus, en partagent d'autres, cela génère des conversations qui leur permet de développer leur réseau de contacts. Nous sommes donc dans une expérience continue et non fragmentée. Ceci étant dit, les services sont classés selon les quatre grands usages suivants :

1. **La publication** avec les plateformes d'hébergement de blog (*WordPress, Blogger, Live Journal, TypePad, Over-Blog, SquareSpace, Medium, …*), la nouvelle génération de services de publication minimalistes (*Svbtle, Ghost, Sett*), les wikis (*Wikipedia,*

[43] http://us.cyworld.com/
[44] http://digg.com/
[45] https://delicious.com/

Wikia, Mahalo…) et les services intermédiaires de publication / partage comme Tumblr.

2. **Les services de partage** de photos (*Flickr, Imgur, 500px Pinterest,…*), de vidéos (*YouTube, Vimeo, Dailymotion…*), de musique (*Spotify, Deezer, SoundCloud, MySpace…*), de liens (*Delicious, Scoop.it*), de lieux (*Foursquare, Swarm*), les applications mobiles (*Instagram, Slingshot, Riff, Vine…*), les communautés d'acheteurs (*TheFancy, Polyvore, Shopstyle, Bezar, Lyst, Yeay…*) ainsi que des communautés verticales comme ces trois-là dédiées aux designeurs (*Behance, Dribbble, DeviantArt*).
3. **La discussion** avec les plateformes conversationnelles (*Quora, Reddit, Github, Tieba Baidu, Disqus, Muut*), les outils de communication grand public (Skype, Sina Weibo, Tencent Weibo), les applications mobiles de communication (*Facebook Groups, BlackBerry Messenger, MessageMe, Telegram, Pheed, Hike, Wire, Bleep…*), et les outils de communication professionnels (*Slack, Yammer, Chatter, Jive Chime, Caliber…*).
4. **Le réseautage** avec les réseaux sociaux grand public (*Tagged, Nextdoor, Notabli, Ello…*) et leurs équivalents asiatiques et russes (*Qzone, VKontakte, RenRen, Mixi, StudiVZ…*), les services de rencontre (*Badoo, OKcupid…*), les applications mobiles de rencontre (*Tinder, Skout*), et les réseaux sociaux BtoB (*LinkedIn, Viadeo, Xing*).

Figure 2. 2: : Panorama des médias sociaux 2015 [46]

Selon le panorama dressé ci-dessus, il est opportun de constater quatre grands domaines d'application qui représente l'intérêt des réseaux sociaux **[36]** :

a - Outils d'expression : qui permettent aux utilisateurs de prendre la parole, de discuter et d'agréger leurs productions, ils comportent :

- **Outils de publication :** dans lesquels nous trouvons les blogs (*Blogger , Typepad , WordPress*), les plateformes de wiki (*Wikipedia, Wetpaint , Wikia*), les plateformes de microbloging tels que Twitter , les portails de news et de « journalisme citoyen » (*Digg, Wikio, Le Post*) ainsi que les outils de livecast (*JustinTV, Ustream, BlogTV*).
- **Outils de discussion :** il s'agit des plateformes de forum (*Phorum*) et de forum vidéo (*Seesmic*), des logiciels et services de messagerie instantanée (*LiveMessenger, Meebo, eBuddy et Y!Messenger*) des services de gestion de commentaires comme Cocomment, Backtype, IntenseDebate et Disqus.
- **Services d'agrégation**, par exemple : FriendFeed, Profilactic, LifeSteam et autres.

b - Services de partage : qui permettent la publication et le partage de contenu. Ils comportent :

- **Outils de Partage de contenu** : de vidéos (*YouTube, Dailymotion, Vimeo*), de photos (*Flickr, SmugMug, Picasa et Fotolog*), de musique (*Last.fm, Deezer, …*), de liens (*Delicious, Reddit, …*), de documents (*Slideshare, Scrib, Slideo*).
- **Partage de produits** : offerts par les services de recommandations (*Crowdstorm, ThisNext, StyleHive*), de suggestions d'évolution (*User Voice, GetSatisafaction*) et d'échange (*LibraryThing, Shelfari, SwapTree*).
- **Partage de lieux :** en fonction des adresses (*BrightKite, Loopt, Whrrl, Moximity*), des évènements (*Upcoming, Zvents, EventFul, Socializr*), des voyages (*TripWolf, TripSay, Driftr, Dopplr*).

c -Services de réseautage : leur but est la mise en relation des individus.

- Les réseaux de recherche d'anciens camarades (*CopainsDavant, Trombi,MyYearBook*), de personnes (MyLife) ou de « conjoints » (*Badoo*).
- Les réseaux de niche : PatientsLikeMe , Dogster , etc.
- Les réseaux B to B : LinkedIn, Viadeo, Xing, etc.
- Les réseaux mobiles : Groovr, MocoSpace, etc.
- Les outils de création/gestion de réseaux : Ning, KickApps, CrowdVine, …

d- Services de jeux en ligne » : très variés :

- Les portails de « casual games »: Pogo , Cafe , Doof , Kongregate , PlayFirst, PopCap, BigFish, Prizee.
- Les portails de « social games »: Zynga, SGN480, ThreeRings, PlayFish, CasualCafe, ChallengeGames.
- Les « MMORPG »: jeux de rôle massivement multijoueurs comme world of Warcraft, EverQuest, Lord of the Rings Online, EVE Online, Lineage,Dofus, Runescape.
- Les « MOG » : jeux massivement multijoueurs, par exemple : Drift City, Maple Story, Combat Arms, Quake Live.
- Les « casual MMOG » : qui selon l'auteur, se positionnent à mi-chemin entre les
- deux catégories précédentes (*MMORPG ET MOG*) : Puzzle Pirates , Club Penguin, Neopets, Gaia Online, SmallWorlds, OurWorld.

La figure 2.3 illustre bien la richesse et la diversité des médias sociaux.

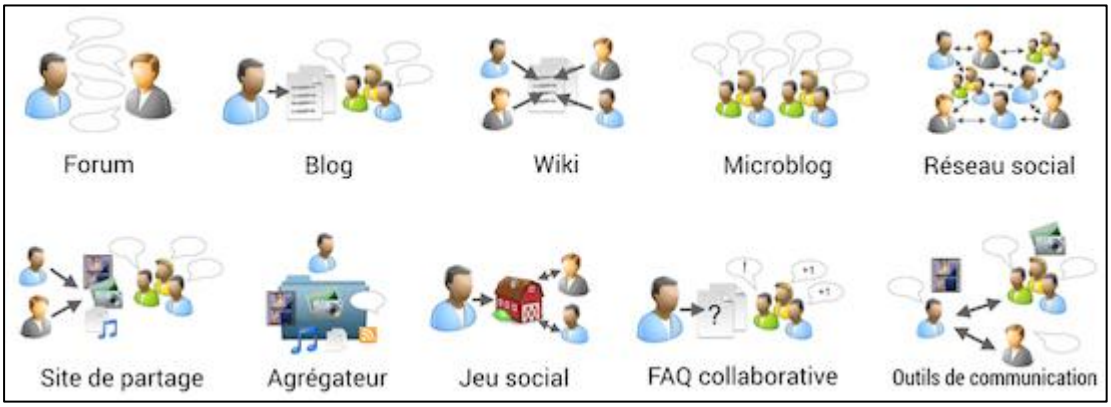


Figure 2. 3: Les différents types de médias sociaux [46]

6. Principaux réseaux sociaux

6.1. Exemple de réseaux sociaux grands publics

- **Facebook** : c'est le réseau social le plus connu : proche du milliard de membres d'inscrits *(à l'heure du post)*. Le principe est d'échanger avec sa communauté d'amis sur tout et n'importe quoi. L'inscription est obligatoire pour l'utiliser. Pour être amis sur Facebook avec une personne, il faut lui envoyer une demande et que cette dernière l'accepte. Facebook permet également de réagir sur les commentaires et news postés par ses amis via le « Like » ou J'aime. C'est un moyen pour dire que l'on a trouvé un

commentaire ou un post à son goût. Il est devenu fréquent d'entendre le verbe « liker » dans une conversation. Facebook permet beaucoup d'autre chose : discussion instantanée, envoi de message direct, identifier des amis sur une photo…

- **Twitter** : c'est le « petit » qui monte en flèche. Il s'agit d'une plateforme de microblogging. Comme Facebook, Twitter permet de partager avec d'autres. Le fonctionnement est toutefois différent de Facebook : une limitation à 140 caractères par message, la possibilité de suivre d'autres comptes, pas de demande d'invitation, le partage de photo, de vidéo ou d'article se fait par l'utilisation de lien. En gros, on peut publier des SMS avec des liens. Sur Twitter, contrairement à Facebook, nous avons pas un mur mais une timeline ou fil. Alors que Facebook est destiné à échanger avec ses amis, Twitter, lui, possède un côté plus pro et relationnel. La notion d'influence y est très présente de par sa logique de suivi d'informations données par d'autres. Avec Twitter, nous lisons les tweets de personnes que nous suivons et pouvons les Retweeter pour les partager avec vos followers.

- **Youtube:** Il peut aussi être classé dans les réseaux sociaux puisqu'il permet de partager ses vidéos et de commenter les vidéos postées. Youtube appartient à Google. Il n'est pas nécessaire d'être inscrit pour regarder les vidéos postées. Mais, pour y déposer vidéos et commentaires, une inscription à Youtube est obligatoire. La logique d'amis » avec les autres internautes inscrits existe et Youtube permet également de suivre des thèmes pour être informé de leurs actualités. La grande majorité du contenu présent provient de particuliers **[36]**.
- **Flickr:** il appartient également à Google. Il est distribué par Yahoo et se prononce Flickère. Flickr met à votre disposition un espace pour poster photos et vidéos. Il existe également une logique d'amis et groupe d'amis comme Facebook ce qui nous permet de choisir le niveau de partage souhaité : tout le monde, quelque amis, votre famille… nous pouvons également géolocaliser le lieu où a été prise la photo. Nos amis peuvent commenter nos photos.

6.2 Exemple de réseaux sociaux professionnels

- **Linkedin** : un réseau professionnel international permet la mise en relation entre des professionnels. Il offre un espace de présentation de ses compétences et expériences qui peut être consultable par le public. Très utile pour le recrutement. En mars 2011, le

site revendique plus de 130 millions de membres issus de 170 secteurs d'activités dans plus de 200 pays et territoires.

- **Viadeo :** Il est le pendant français du réseau social Linkedin. Il permet lui aussi de construire et de gérer son réseau professionnel. Viadeo est plus populaire et connu en France que Linkedin. Il offre à peu près les mêmes possibilités que Linkedin.
- **Xing :** C'est une plate-forme allemande qui permet de construire et d'agréger son réseau professionnel .Il possède 3,5 millions d'utilisateurs répartis sur plus de 190 pays **[47]**.

Le tableau suivant décrit le Top 3 des réseaux sociaux en Décembre 2014[46] **:**

December 2014 ALEXA			
Countries	SNS #1	SNS #2	SNS #3
Argentina	Facebook	Twitter	Linkedin
Australia	Facebook	Twitter	Linkedin
Belgium	Facebook	Twitter	Linkedin
Brazil	Facebook	Twitter	Linkedin
Canada	Facebook	Twitter	Linkedin
Denmark	Facebook	LinkedIn	Twitter
Finland	Facebook	Twitter	Linkedin
France	Facebook	Twitter	Linkedin
Germany	Facebook	Twitter	Ask.fm
India	Facebook	LinkedIn	Twitter
Italy	Facebook	Twitter	Linkedin
Netherlands	Facebook	Twitter	Linkedin
Norway	Facebook	Twitter	Reddit
Portugal	Facebook	Twitter	Linkedin
Sweden	Facebook	Twitter	Linkedin
Spain	Facebook	Twitter	Linkedin
United Kingdom	Facebook	Twitter	Linkedin
United States	Facebook	Twitter	Reddit

Tableau 2. 1:Top 3 des réseaux sociaux (*Décembre 2014*)

[46] Source : http://vincos.it/world-map-of-social-networks/

7. Cartographie des réseaux sociaux numériques

Une cartographie des réseaux sociaux éditer en Aout 2015, montrant les sites de réseautage social les plus populaires par pays, selon Alexa[47].

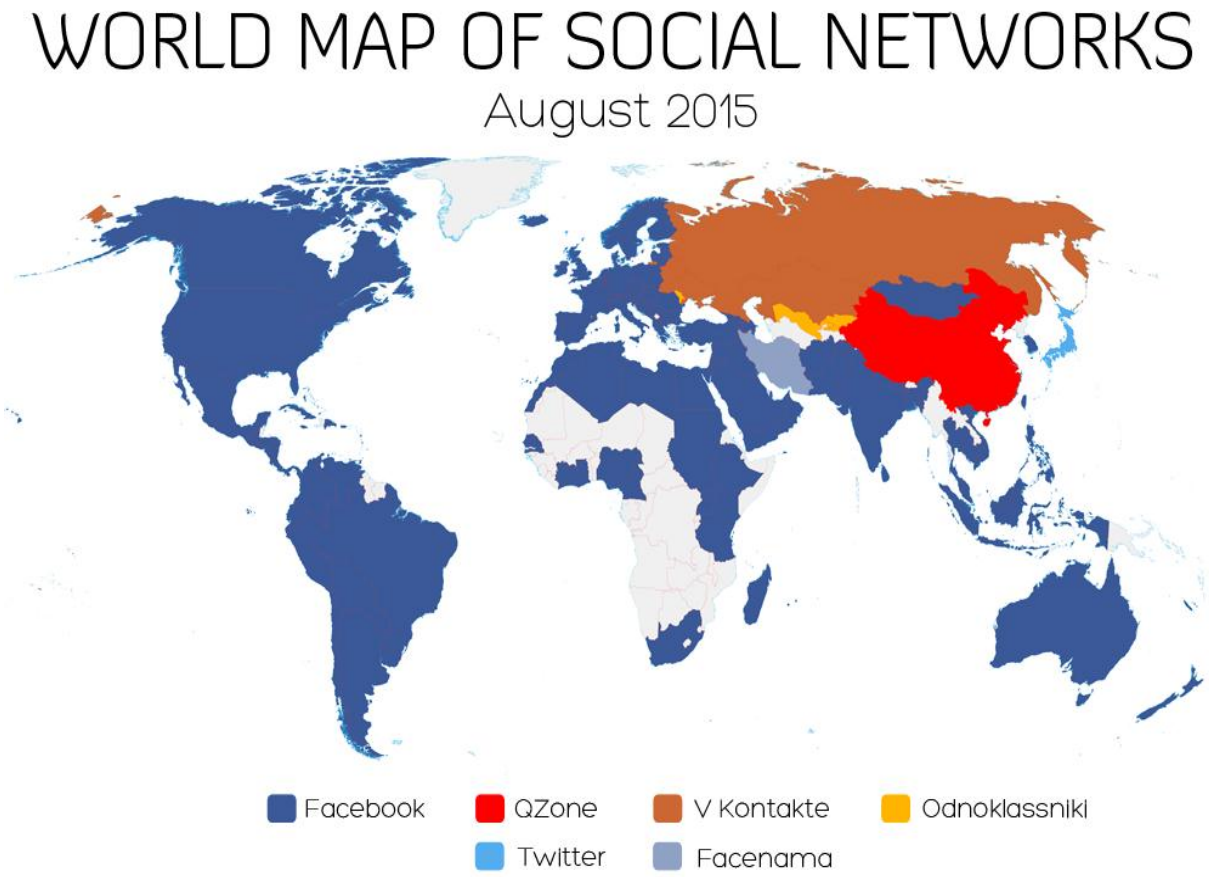


Figure 2. 4: Carte mondiale des réseaux sociaux [48]

Nous pouvons noter la très forte implantation de Facebook en Europe Occidentale, sur le continent américain, en Afrique du Nord ,ainsi qu'en Australie .

8. Fonctions et fonctionnalités

Le fonctionnement de ces sites est toujours le même : pour créer sa page d'accueil, on demande au nouvel arrivant de définir un profil qui constituera la représentation qu'il a de lui-même sous la forme de textes, de photos, de vidéos, de musiques et de liens. Des questionnaires lui imposent parfois de préciser ses goûts, ses affinités, ses opinions, etc. Facebook permet également aux utilisateurs d'enrichir leur profil avec toutes sortes d'applications. Les listes d'«amis» font le lien entre les profils. Chaque participant au site peut

[47] http//:www.alexa.com/

demander à d'autres participants au même site de faire partie de ses «amis»; si sa proposition est acceptée, le nom et une photo (*ou un avatar*) de l' «ami» sont affichés sur le site de l'autre. Par «contagion» les participants du site étendent ainsi continuellement leur liste d'«amis». Mais qui dit «ami» sur Facebook ne dit pas forcément camarade dans la vraie vie **[49]**.

La liste d'«amis» ne se résume pas à l'exposition des liens forts d'amitié qui ont une réalité dans la «vraie vie». Elle permet aussi de mesurer l'audience que chaque participant a auprès d'autres participants, souvent inconnus dans un premier temps, qui partagent les mêmes activités, goûts, opinions, etc. Ces liens sont dits faibles. Deux pratiques différentes tournent autour de cette distinction : l'accès d'un profil peut être restreint aux «amis», liens forts ou faibles, ou au contraire ouvert à tous, pour faciliter la croissance de la liste.

La recherche à tout prix de nouveaux «amis» peut conduire à accepter des inconnus dans la liste, sans vérifier leur identité, ni s'ils sont véritablement des «amis» d'«amis». C'est un des moyens couramment utilisés pour atteindre les données protégées d'un compte.

La troisième par ordre d'importance est la fonction de revue publique ; **« Témoignages » , « Commentaires », « Panneau d'affichage [50].** Cette fonctionnalité essentielle des réseaux sociaux permet à tout visiteur de déposer des **commentaires** sur une page d'accueil (*la sienne ou celle de quelqu'un d'autre*). Ces commentaires sont lisibles par toutes les personnes autorisées à accéder au profil (*Mur sur Facebook*). Toutes ces possibilités permettent une médiatisation de soi via un profil détaillé et des contenus diversifiés (*statuts, photos, liens*), ainsi qu'une socialisation active avec les membres de son entourage au moyen des commentaires.

La plupart des sites sociaux propose l'échange de messages privés entre participants. Ils peuvent aussi offrir des services de partage de photos, de vidéos, et offrir des interactions avec les téléphones mobiles. Ces trois fonctionnalités – profils, listes d'amis, commentaires – constituent la structure de base des réseaux sociaux numériques, même si certains d'entre eux proposent des fonctions complémentaires. Les réseaux sociaux numériques permettent à leurs visiteurs de naviguer d'un « Ami » à l'autre et de correspondre avec quiconque possède un profil visible **[50].**

Les réseaux sociaux numériques permettent la navigation d'un profil à l'autre et rendent accessible la communication avec tout participant. Le mode d'accès le plus répandu est cependant basé sur l'existence de groupes d'«amis» préexistants, ayant rejoint le site ensemble, pour conserver ensuite le contact entre eux (*élèves d'une classe, d'une école, etc.*) et pouvoir communiquer deux à deux tout en faisant partie d'un groupe **[49]**.

9. Enjeux des réseaux sociaux numériques

Les enjeux dans le domaine des réseaux sociaux sont multiples tant pour les utilisateurs que pour les acteurs. Du côté des utilisateurs, les réseaux sociaux regroupent toutes sortes d'individus provenant de tous secteurs. Ainsi, sur le réseau LinkedIn, les utilisateurs du service viennent de tous les horizons : plus de 120 secteurs d'activités sont représentés et aucun secteur ne représente plus de 11% de la base des inscrits . D'après Scott Allen[48] que 11,8% des utilisateurs de LinkedIn sont PDG, 10,2% sont vice-présidents ou directeurs généraux et 1,3% sont membres d'un conseil d'administration. Cela ne vaut pas que pour les réseaux sociaux professionnels. Sur MySpace, par exemple, il est possible de trouver la plupart des grands groupes de musique qui ont créé leur page MySpace officielle .

De ce fait, il est possible de retrouver quasiment n'importe quelle personne quel que soit sa situation géographique ou son poste au sein de la société **[42]**. En plus de pouvoir visualiser des informations sur les personnes, les réseaux sociaux offrent l'opportunité d'entrer en contact avec toutes ces personnes. Les liens entre tous les membres d'un réseau sont les profils personnalisés, ce que l'on peut appeler la carte d'identité numérique. Ainsi, les réseaux sociaux permettent de gérer son identité numérique ainsi que sa réputation en ligne.

Lorsque l'utilisateur remplit sa fiche, il a le choix d'y intégrer les informations qu'il souhaite et de cacher celles qu'il estime privées. Les réseaux sociaux lui offrent donc de la visibilité et lui permettent de contrôler son « extimité ». Plus l'internaute arrive à se mettre en avant et se rendre visible sur la toile moins les problèmes de vie privée apparaissent car les informations visibles seront les informations choisies. De même selon les informations entrées sur la fiche d'identité, l'utilisateur va pouvoir se mettre en avant devant telles ou telles personne et dans un cadre bien précis (*recherche d'emploi, contact pour développer une entreprise…*).

Et en raison de la popularité croissante des réseaux sociaux en ligne (*OSN*) et énorme quantité de données partagées sensibles, la préservation de la vie privée devient un enjeu majeur pour les utilisateurs OSN. Les Services de réseaux sociaux existants sont centralisées et les entreprises qui fournissent les services ont l'autorité exclusive de contrôler toutes les données des utilisateurs. Les utilisateurs ont généralement peu de contrôle sur comment et ce que les informations les concernant sont présentés à leurs amis en ligne.

La présentation de leur information dépend en grande partie sur la conception du service de réseau social que les utilisateurs utilisent. plusieurs architectures décentralisées ont

[48] Editeur de onlineBusinessnetworks.com qui fournit une étude sur l'utilisation de LinkedIn

récemment été proposées pour des OSN décentralisée basée sur les réseaux P2P comme une alternative des architectures OSN centralisées.

Du côté des grands acteurs, il existe également de nombreux enjeux :

Il s'agit essentiellement là d'enjeux économiques mais aussi de visibilité. Ainsi les milliers d'utilisateurs inscrits offrent, indirectement, une source de revenu importante. MySpace compte entre 80 et 100 millions de profils créés dont un million rempli en détail. Il est ainsi possible de les cibler très précisément pour leur proposer du contenu publicitaire en adéquation avec leurs passions et leurs centres d'intérêts. Outre la publicité directe, les réseaux sociaux, grâce à leurs nombres importants d'utilisateurs, offrent aux grands groupes (*audiovisuels, musicaux, informationnels…*) un beau support de diffusion avec un large public qu'ils peuvent toucher de manières très pertinentes. Ainsi, des chaînes comme celles du groupe Fox ou encore des maisons de disques profitent des réseaux sociaux pour diffuser des contenus adaptés au profil des membres de communautés. De plus, il est très facile d'infiltrer la communauté (*création de page personnelle pour un utilisateur fictif, mise en ligne de vidéo marketing…*) afin de mettre en place une opération de marketing viral.

Ainsi, il est essentiel pour les acteurs du monde des réseaux sociaux d'accroître le nombre d'utilisateurs qui est au final leur vrai fonds de commerce.

10. Réseaux sociaux : développement, perspectives et évolutions

10.1. Développement des réseaux sociaux

Développement de la technologie : un point qui peut sembler être un détail mais qui, au contraire, s'il n'est pas bien analysé peut faire beaucoup de dégâts. Il faut tout d'abord savoir que le système d'un réseau social est bien plus complexe que n'importe quels autres sites, forums ou blogs. L'algorithme de mise en relation et de navigation au sein de ces liens demande beaucoup de ressources. De même les bases de données doivent gérer une énorme quantité d'informations et supporter une charge importante d'enregistrement. Dès lors, il est essentiel de bien dimensionner la puissance de son parc de serveurs et d'estimer correctement la bande passante nécessaire **[42]**. Effectivement, la technologie n'est pas critique pour débuter mais elle est indispensable pour atteindre le succès. Si 10 000 ou 100 000 utilisateurs sont gérables, quelques grands noms comme Friendster ou Orkut [49]se sont effondrés une fois le million d'utilisateurs inscrits. Par ailleurs, l'architecture doit être bien réfléchie dès le début de l'aventure car tous changements de structure (*langage, serveurs etc.*) seraient difficilement

[49] Site communautaire de Google, http//:www.orkut.com/

envisageables. Les systèmes P2P est une alternative intéressante de la technologie actuelle utilisée (*les OSNs sont basés sur une architecture Client/Serveur*) qu'il faut l'exploiter dans les applications des réseaux sociaux (l'énorme volume d'informations sera distribué)

Développement de fonctionnalités : d'autres fonctionnalités peuvent être imaginées comme la création de widgets [50]. Ces petits modules ne proposant qu'une fonctionnalité (*recherche, gestion photo, lecteur audio ou de flux d'informations…*) peuvent être ajoutés ou supprimés selon les besoins. Il est également envisageable de coupler certaines fonctionnalités comme celle de partage de fichiers via les **réseaux de P2P**. Ainsi les artistes ou même les utilisateurs peuvent facilement se partager et s'envoyer des fichiers en les mémorisant sur leur propre ordinateur.

Enfin, il y a des fonctionnalités plus utiles pour les machines que pour les utilisateurs mais qui, au final, rendent bien service à ces derniers. Il s'agit des micro-formats qui permettent de normaliser du contenu afin d'être plus facilement identifiable par les robots, agents intelligents ou moteurs de recherche. Ces micro-formats font entrer une nouvelle notion : la sémantique. Ils vont permettre de décrire et de donner du sens à un curriculum vitae, à une fiche d'identité ou à un espace personnel de manière totalement compréhensible par les machines pour qu'elles puissent les ressortir de façon pertinente lors d'une requête précise effectuée par les utilisateurs. Plusieurs micro-formats sont ainsi apparus comme FOAF (*Friend Of A Friend permet de décrire un individu et ses connaissances*), Hcard (*format dédié à la description des individus, des entreprises et organisations*) ou encore OpenID (*format et service permettant la gestion décentralisée de l'identité*). Désormais, ces micro-formats sont utilisés dans différents services comme Technorati (*annuaire universel de blogs*), Naimz ou encore ClaimID qui sont des services de gestion d'identité numérique. Ces quelques points clés permettront, s'ils sont mis en pratique, d'améliorer les services de réseaux sociaux en ligne en facilitant la vie des utilisateurs.

10.2. Perspectives et évolutions

Nombreuses sont les perspectives et les évolutions à venir dans le monde des réseaux sociaux. Certains spécialistes pensent que le nombre grandissant de réseaux sociaux est un réel problème pour les utilisateurs. Chaque internaute, s'il veut être visible sur la majorité des réseaux doit créer un profil différent à chaque fois. Ils se retrouvent vite à devoir gérer plus d'une dizaine de réseaux et donc de contacts et autant de profils. Ainsi, l'une des évolutions

[50] Un widget peut également désigner un petit utilitaire permettant à son utilisateur d'afficher des informations diverses, comme un calendrier, la météo, un traducteur...

possible serait d'avoir un service unique, un agrégateur de profils de réseaux sociaux : une identification, un profil, une liste de contact unique…Il s'agit là d'une utopie bien évidemment mais il existe des services, dans la même optique, permettant d'effectuer des recherches au sein de plusieurs réseaux sociaux **[42]**. *Fly over PSN*, proposé par *6nergies*[51] permet d'effectuer une recherche de profil sur plusieurs réseaux sociaux professionnels[52]. C'est un bon début mais il reste encore du chemin avant d'obtenir l'outil universel.

D'autres perspectives sont ouvertes notamment au niveau des réseaux sociaux de niche. Même si MySpace regroupe plus de 100 millions de profils et Skyblog plus de cinq millions de blog, il n'est pas certain que tous les utilisateurs y trouvent leur bonheur. Ainsi, des réseaux sociaux spécialisés autour d'un thème précis seraient tout à fait légitimes. Il existe déjà quelques initiatives comme Boompa pour les fans de voitures et autres engins à moteur (*l'un des réseaux sociaux les plus avancés en terme de fonctionnalités relationnelles*), BakeSpace pour les amateurs de cuisine, Eons pour les plus de 50 ans ou encore FindaGoth pour les gothiques célibataires. On trouve ainsi des réseaux sociaux pour tous les goûts et il reste certainement des niches ouvertes comme des réseaux sociaux autour de l'immobilier ou du sport par exemple.

Si ces réseaux sociaux sont certainement viables et pertinents, le même problème de gestion de multiples profils est de nouveau présent. D'où l'importance de créer **un outil universel** afin de standardiser tous les réseaux sociaux et leurs contenus.

D'autres évolutions, notamment du support, sont amenées à apparaître d'ici quelques années. Si l'on en croit Bradley Horowitz, responsable des développements technologiques de Yahoo! : « *L'avenir est aux médias sociaux et dans cinq ans, tout sera mobile* » tout en ajoutant « *Nous nous en tenons à la formule originale de Yahoo! qui consiste à faire travailler ensemble machines et humains* ». Yahoo! dont le réseau social se nomme Yahoo 360 devrait prochainement proposer CheckMate, un service qui permet de savoir où sont nos amis et de les retrouver quand ils sont à proximité. Le principe de ce service repose sur la **géolocalisation du téléphone**. Des étiquettes géographiques ou « ZoneTags » sont générées automatiquement par l'identification de la latitude et la longitude des antennes relais utilisées par les téléphones portables. Ce tag est ensuite placé sur une carte qui sera envoyée sur FlickR (*le service de photos on-line racheté par Yahoo!*) et qui pourra être accessible par les parents, amis ou relations.

[51] 6nergies.net est une entreprise de réseautage d'affaires créée par Alain Lefebvre en 2004. http//:www.6nergies.net

[52] Appelés également les PSN pour Professional Social Networks

Enfin, on note une dernière évolution majeure dans le domaine du réseau social : le **Social Media Optimization**. Il s'agit d'un standard, non officiel, autour d'un concept où, selon son créateur Rohit Bhargava, Vice-Président marketing interactif de la division relations publiques d'Ogilvy, les réseaux sociaux gagneraient en effectivité si certains critères étaient respectés. Ces optimisations sont au nombre de 16. Elles tournent autour de différentes thématiques comme le contenu, la visibilité ou encore la gestion de sa communauté. Même si ce dernier point n'est pas vraiment une évolution en soit, il montre néanmoins que des groupes de réflexion travaillent sur une amélioration des services pour les réseaux sociaux.

11. Conclusion

De nos jours, les réseaux sociaux numériques sont devenus des outils de communication incontournables. Dans ce chapitre, nous avons présenté les réseaux sociaux numériques, nous avons commencé par donner une définition, ensuite nous avons cité quelques typologies des OSNs, leurs intérêts, les enjeux, et nous avons terminé par donner les perspectives et l'évolution dans le domaine des OSNs.

Parmi les enjeux des réseaux sociaux est que la plupart reposent sur une architecture centralisée qui présentent deux problèmes. Tout d'abord, les informations sur un seul site OSN est inutilisable dans les autres. Deuxièmement tels sites ne permet pas aux utilisateurs plus de contrôle sur la façon dont leurs informations personnels sont diffusés, ce qui entraîne des problèmes de confidentialité potentiels.

L'adoption d'une architecture décentralisée est une meilleure alternative pour pallier ces enjeux.

Plusieurs architectures décentralisées ont récemment été proposées pour OSN décentralisé.

Dans le chapitre suivant, nous allons présenter une étude des propositions existantes des architectures distribuées pour les réseaux sociaux.

Chapitre 3

Modélisation des réseaux sociaux distribués

1. Introduction

Malgré leur immense succès, les Réseaux sociaux en ligne (*OSN*) ont des problèmes inhérents à la vie privée et la protection des données de l'utilisateur. Ces enjeux ont motivé les chercheurs à faire un changement de paradigme dans l'architecture OSN en proposant de remplacer l'OSN de contrôle centralisé avec OSN décentralisé (*DOSNs* [53]) dans un environnement peer-to-peer. Les DOSNs donnent aux utilisateurs une plus grande autonomie avec la possibilité de participer à des réseaux sociaux sans perdre le contrôle de leurs données. Les diverses propositions de DOSN ont des différences significatives dans leur services proposés, l'architecture et l'étendue de la décentralisation. Dans ce chapitre, nous étudions un certain nombre de propositions des architectures DOSNs existantes, nous donnons une comparaison et une taxonomie.

[53] Decentralized Online Social Network

2. Architecture de système d'OSN

Il y a deux paradigmes de mettre en application un OSN dans la littérature **[51]**, l'architecture client server et l'architecture (*P2P*) peer-to-peer, qui rapporte les systèmes centralisés et distribués, respectivement.

2.1 Architecture Client/Server

Les OSNs d'aujourd'hui sont centralisés et basés sur les serveurs web. Toutes les fonctionnalités, comme le stockage, l'entretien, et l'accès aux services d'OSN sont offertes par les fournisseurs commerciaux d'OSN tels que Facebook Inc.[54], LinkedIn Corp.[55], et XING AG [56]. Cette architecture a l'avantage d'être simple et facile à mettre en application, alors qu'elle souffre de tous les inconvénients des systèmes centralisés. Par exemple, n'importe quelle entité centrale peut facilement être un point unique de panne, un mono cible pour des attaques de déni-de-service (*DoS*) (*pour plus de détails, voir la référence* **[52]**), et également un goulot d'étranglement pour les performances du réseau.

2.2 Architecture P2P

Il y a une tendance forte de concevoir une architectureP2P pour les OSNs de la prochaine génération. Il adopte une architecture décentralisée se fondant sur la coopération d'un certain nombre de parties indépendantes, qui sont également des utilisateurs de l'OSNs.

Les espaces personnels d'utilisateurs[57] sont stockés et maintenus d'une manière distribuée. En soutenant l'échange d'informations direct entre les dispositifs, que ce soit entre les utilisateurs avant qu'ils se sont réunis ou entre les nœuds adjacents d'un réseau maillé de ville. Une architecture P2P peut tirer profit de vrais réseaux sociaux et de proximité géographique pour supporter des services locaux quand l'accès à Internet est indisponible.
Mais pour l'architecture P2P, offrir quelques fonctionnalités d'OSN (*par exemple, la recherche globale*) de manière distribuée est un problème difficile.

[54] www.facebook.com
[55] www.linkedin.com
[56] www.xing.com
[57] La collection d'un profil d'utilisateur particulier, et des liens sociaux reliés s'appelle l'espace personnel numérique.

3. Architecture de Référence

L'architecture de référence se compose de six couches et fournit une abstraction architecturale de la variété des approches actuelles liées à la décentralisation des réseaux sociaux dans la littérature de recherche. Ces couches sont représentées dans la figure 3.1 **[53]** :

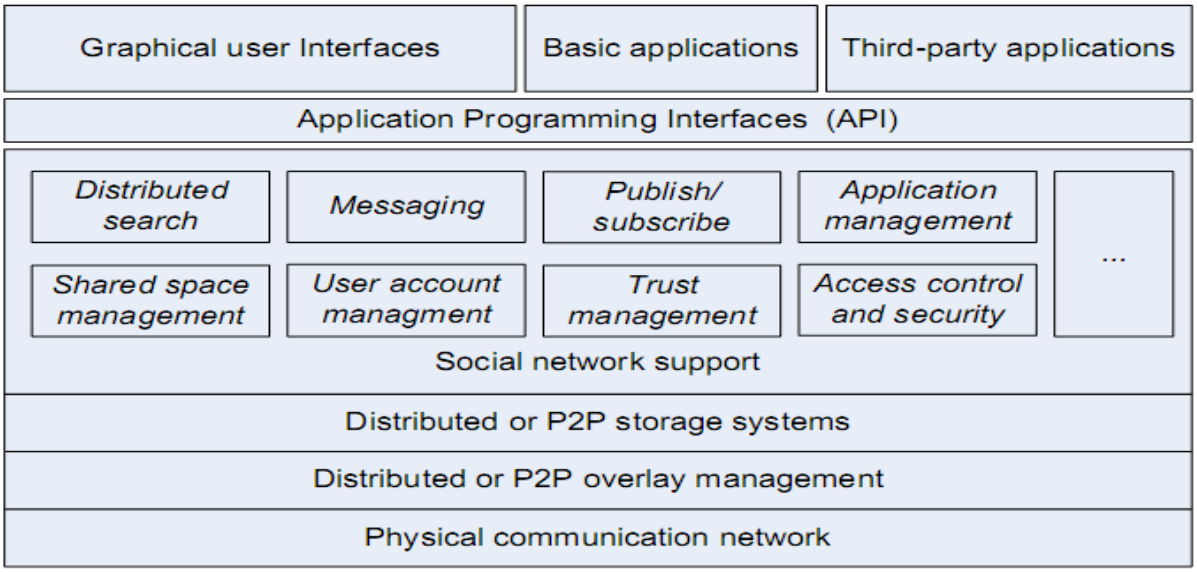


Figure 3. 1: Architecture générale d'un réseau social en ligne distribué

La couche inférieure de cette architecture est le réseau de communication physique, qui peut être l'Internet ou un réseau ad hoc (*dans le cas où l'on considère un réseau social en ligne mobile*).

La gestion distribuée ou P2P overlay (*The distributed ou P2P overlay management*) offre des fonctionnalités de base pour gérer les ressources dans l'infrastructure de support du système, qui peut être un réseau distribué de serveurs de confiance ou un overlay P2P. Plus précisément, cette couche fournit aux couches supérieures les possibilités de rechercher des ressources, le routage des messages, et la récupération des informations de manière fiable et efficace entre les nœuds dans l'overlay.

Au-dessus, il y a la couche de gestion de données décentralisée (*The decentralized data management layer*), qui met en œuvre les fonctionnalités d'un système d'information distribué ou peer-to- peer tel que: interroger, insérer, mettre à jour de divers objets persistants aux systèmes.

La couche de réseau social implémente toutes les fonctionnalités et fonctions de base qui sont fournis par des services modernes de réseaux sociaux centralisés, à savoir la capacité

de recherche dans le système : recherche distribuée de l'information pertinente, la gestion des utilisateurs et de l'espace partagé (*ex. compte d'utilisateur et gestion de l'espace de l'action*), la gestion de la sécurité et les questions d'accès de contrôle (*ex. gestion de confiance, contrôle d'accès et sécurité*) , la coordination et la gestion des applications sociales.

La couche de réseau social expose et met en œuvre une interface de programmation d'application (*API* [58]) pour soutenir le développement de nouvelles applications par les développeurs indépendants et d'autres tiers (*third-party*)[59], ainsi que pour permettre la personnalisation du service de réseau social pour répondre à diverses préférences de l'utilisateur.

La couche supérieure de l'architecture comprend l'interface utilisateur pour le système et diverses applications construit sur le dessus de la plate-forme de développement fournie par le DOSN. L'utilisateur DOSN devrait fournir à l'utilisateur la transparence nécessaire pour utiliser le DOSN comme n'importe quel autre OSN centralisé. Les applications peuvent être mises en œuvre par le fournisseur DOSN ou développés des tiers (*third-party*), et peuvent être installés ou retirés du système selon les préférences de l'utilisateur.

4. architecture des réseaux sociaux distribués

Les soucis de confidentialité et de la nature fermée d'OSN existants ont motivé les chercheurs à venir avec différentes propositions de DOSN. Dans cette section, nous donnons une brève présentation d'un certain nombre de propositions dans la littérature. Ces projets présentent une variation suffisante dans leur architecture, les services proposés et les choix de conception.

4.1 PeerSon

PeerSon **[54]** est une approche Peer-to-Peer distribuée et couplée avec un système de chiffrement. Son infrastructure est faite justement pour supporter les caractéristiques les plus

[58] API : Application Programming Interface

[59] Third -party (tiers) : Dans le monde de l'informatique, un tiers peut se référer à fabricant du matériel ou un développeur logiciel. Il est une étiquette donnée à des entreprises qui produisent des matériels ou logiciels pour un produit d'une autre société. Matériel tiers se réfère à des composants qui sont développés par des entreprises outre le fabricant de l'ordinateur d'origine.
Logiciel tiers se réfère à des programmes qui sont élaborés par des sociétés autres que la société qui a développé le système d'exploitation. Par conséquent, toutes les applications Macintosh qui ne sont pas développés par Apple sont considérées comme des applications tierces. De même, tous les programmes Windows développés par des sociétés autres que Microsoft sont appelés programmes tiers. Comme la plupart des programmes sont développés par des sociétés autres que Apple et Microsoft, les applications tierces constituent la majorité des programmes de logiciels. Source : http://techterms.com/definition/thirdparty

importantes des réseaux sociaux en ligne (*telles que : la messagerie instantanée, le mur, le fil d'actualité,…*) dans un environnement distribué.

Il vise à maintenir les caractéristiques des réseaux sociaux en ligne en surmontant deux limitations : La question de confidentialité et l'exigence d'une connectivité Internet pour toutes les transactions **[55]**.

Pour résoudre le problème de confidentialité, un système de cryptage et de contrôle d'accès est utilisé en couplage avec une approche P2P pour remplacer l'autorité centralisée des réseaux sociaux en ligne classiques. Ces mesures empêchent les violations de la vie privée des utilisateurs par les fournisseurs de ces systèmes, les annonceurs et même les utilisateurs eux même. Sa conception répond principalement à trois exigences : le chiffrement des données, la décentralisation de l'architecture et l'échange direct entre les utilisateurs. En un mot, le chiffrement assure la confidentialité pour les utilisateurs, la décentralisation basée sur l'utilisation d'une infrastructure P2P permet quant à elle l'indépendance des fournisseurs des réseaux sociaux en ligne, ce qui rend plus facile l'intégration de l'échange direct des données entre les appareils des utilisateurs dans le système.

a- Chiffrement des données :le moyen de protection de la vie privée dans ce contexte est de permettre aux utilisateurs de crypter et contrôler l'accès à leurs données. L'accès à ces données se fait à travers le partage de la clé appropriée, cependant, les considérations de la sécurité incluent l'amorçage (*bootstraping*), la distribution et la révocation des clés.

b- Décentralisation de l'architecture : pour mieux protéger la vie privée, les données des utilisateurs sont cryptées et accessibles uniquement à ceux qui ont les clés de déchiffrement. C'est l'utilisateur lui-même qui détermine avec qui partager les clés de décryptages des données qu'il publie.
Par ailleurs, il n'est pas nécessaire de faire valoir le fait que même si les données sont cryptées et centralisées, le prestataire du service central pourrait être en mesure de décrypter ces données et de les utiliser.

c- Echange directe entre les utilisateurs : puisque le service du réseau social est décentralisé et n'est pas basé sur un serveur web, les utilisateurs n'ont pas besoin d'être constamment en ligne pour l'utiliser. Ils peuvent donc échanger des informations directement entre eux quand ils se connectent.

Ainsi, les utilisateurs peuvent stocker les données des autres utilisateurs et diffuser les informations à travers le réseau social physique ou retarder le téléchargement des données jusqu'à ce que quelqu'un dispose d'une connectivité en ligne.

4.2 Safebook

Safebook **[56]** est une nouvelle approche de réseaux sociaux en ligne qui repose sur deux principes:

- **Architecture peer-to-peer** afin d'éviter le contrôle des données de l'utilisateur et le comportement d'une entité unique, en tant que fournisseur de services;
- **Confidentialité et gestion de la confiance** pour les données utilisateur et des communications dans le système de OSN exploitant les relations de confiance du réseau social.

Safebook a pour but la protection de la vie privée en se basant sur les relations de confiance de la vie réelle. Ces relations de confiance **[57]** directes sont exploitées à des fins de stockage de données et la disponibilité des données. Puisque les nœuds d'amis sont considérés comme honnêtes mais curieux, les données sont stockées sous forme cryptée.

La disponibilité des services de base tels que le stockage de données, la consultation de données et la communication est garantie par la coopération des nœuds pairs. Une telle coopération est imposée par confiance réelle parmi les membres d'OSN.

Selon Refik Molva et .al **[58]** cette approche est basée sur deux principales raisons :

- La décentralisation via une architecture peer-to-peer, afin d'éviter le contrôle de données des utilisateurs et leur comportement sur le réseau par une seule entité tel que le fournisseur du service.
- L'exploitation de la confiance de la vie réelle , via la gestion de confiance et la protection de la vie privée des données utilisateurs, et les communications dans le réseau social en ligne, en exploitant les relations de confiance à partir de ce même réseau social.

Sa conception répond à un large éventail d'exigences de sécurités dont la pertinence est recueillie à partir d'une série d'études, on distingue : la confidentialité de bout en bout, l'authentification, le contrôle d'accès, la vie privée, l'intégrité des données et la disponibilité des données.

a. Confidentialité de bout en bout

Elle a pour but de garantir qu'aucune autre partie en dehors des deux peers communicants ne pourra avoir accès aux données échangées en rendant également impossible l'écoute.

Parce que dans les systèmes peer to peer les messages échangés par les peers peuvent avoir un contenu malveillant posté par une attaque de man in the middle **[59]**. Il est important d'édifier un centre spécial contre ces attaques qui peuvent facilement être monté dans un tel environnement.

b. Authentification

Une authentification particulière des membres est requise afin d'achever le contrôle d'approche. La politique « Fine-Grained Policy **[60]** » du contrôle d'accès basée sur les attributs du profile et les données privées peut être utilisée pour garantir la divulgation de données en fonctions de l'intégrité du requérant.

c. Vie privée

La vie privée vise l'anonymat, la non traçabilité et l'incapacité à suivre des communications d'utilisateurs aussi bien que le respect de la confidentialité des informations personnelles vis-à-vis des intrus et du fournisseur du système.

d. Intégrité des données

L'intégrité des données vise à empêcher la falsification des données des profils car la propriété de disponibilité de ces données représente une exigence permettant une facilité principale d'utilisation. Cela garantit l'accès aux profils à tout moment ainsi que la délivrance des messages à tout utilisateur dans les mêmes conditions en empêchant des attaques de déni de service et les attaques Sybil **[61]**.

Bien que Safebook respecte ce qui a été mentionné ci-dessus, la faisabilité d'une approche décentralisée en termes de disponibilité des données, ainsi que la réactivité du système restent une question ouverte.

4.3 DECENT

DECENT **[62]** est un réseau social en ligne décentralisé, qui emploie une DHT pour stocker et récupérer des objets de données créés par leurs propriétaires. Chaque objet est crypté pour fournir la confidentialité. L'avantage principal de cette architecture est sa

modularité, c'est-à-dire, les politiques d'accès, les mécanismes cryptographiques et la DHT sont trois composants séparés, interagissant l'un avec l'autre par des interfaces bien définies.

Pour la mise en œuvre du prototype, la conception modulaire fournit la capacité d'utiliser n'importe quel type de DHT et n'importe quel type de plan cryptographique.

C'est une conception concernant les réseaux sociaux décentralisés mettant l'accent sur la sécurité et la vie privée.

Les utilisateurs de DECENT utilisent un mécanisme cryptographique efficace pour la confidentialité, combinant des plans cryptographiques traditionnels et avancés pour l'intégrité et la disponibilité. La simulation et les expériences avec le prototype préliminaire de DECENT montrent que le respect de la vie privée a été amélioré **[63]**.

L'architecture DECENT fournit la flexibilité pour la direction de données dans une conception orientée objet. Elle utilise un plan cryptographique approprié et avancé qui soutient une révocation d'accès efficace et des politiques « Fine-Grained Policy » sur chaque pièce de données.

Les autres architectures se concentrent seulement sur un ou deux des trois contraintes des réseaux sociaux distribués (*La confidentialité, la disponibilité et l'intégrité*) mais pas l'ensemble de ces aspects. Cependant DECENT combine ces trois contraintes en utilisant les fonctionnalités d'une DHT. La nouveauté de cette architecture réside dans l'intégration des primitives existantes qui sont adaptés pour améliorer la sécurité et la vie privée des réseaux sociaux en ligne. Ces primitives sont entre autre la politique d'accès, le mécanisme de chiffrement et les algorithmes recherche utilisés dans les DHTs.

4.4 CACHET

Cachet [64] est une approche d'un réseau social distribué, basée sur DECENT, il fournit des garanties de sécurité et de confidentialité forte tout en préservant la fonctionnalité principale des réseaux sociaux en ligne. C'est un OSN purement décentralisée où les utilisateurs collaborent entre eux pour stocker leur contenu sans service centralisé. En particulier, Cachet protège la confidentialité, l'intégrité et la disponibilité des données des utilisateurs, aussi bien que l'intimité de leurs relations via le réseau.

Cachet utilise un réseau distribué de nœuds pour le stockage des données et assure la disponibilité de ces derniers. Mais ces nœuds ne sont pas dignes de confiance, donc, le niveau de cryptographie a été augmenté par rapport à Decent afin de protéger les données.

Le contenu des données des utilisateurs est stocké dans un groupe de nœuds distribués basé sur la DHT FreePastry[60].

Une cryptographie adaptée est utilisée pour le stockage des données dans les nœuds pour une authentique mise à jour des requêtes. Un cryptage basé sur les attributs des objets échangés dans le réseau est utilisé pour diminuer le temps de cryptage/décryptage qui était important dans l'architecture de Decent.

Cachet est en réalité une évolution de DECENT. Le résultat obtenu montre l'importance d'utiliser le réseau de Cachet, qui réduit la latence de la visualisation d'une page d'actualité de 100s en moins de 10s. Cette architecture démontre que la combinaison de plusieurs systèmes distribués et les techniques cryptographiques peuvent être utilisées pour avoir une alternative de protection de vie privée convaincante par rapport aux réseaux sociaux centralisés.

4.5 SuperNova

SuperNova **[65]** propose une architecture de DOSN basée sur le concept de *super-peers.* SuperNova permet aux utilisateurs de faire partie de l'OSN et de partager leur contenu tout en conservant la propriété du contenu complet. En outre, les utilisateurs peuvent imposer l'accès de public (*accessible à tous*), privé (*accessible à aucun*) ou protégée (*accessible à un sous-ensemble d'amis*) à tout leur contenu. Dans SuperNova, des super-peers participent activement à la formation de l'infrastructure de contrôle de l'OSN. Les utilisateurs de SuperNova peuvent stocker leurs contenus sur leurs propres machines.

Cependant, le système de stockage d'un utilisateur pourrait ne pas être très disponible, par exemple, un téléphone portable qui pourrait ne pas avoir 24/7[61] connectivité Internet. Pour faire face à ce problème, les utilisateurs peuvent répliquer leurs contenus à d'autres utilisateurs connus comme *entreposeurs* (*storekeepers*) **[65]**.

[60] http://www.freepastry.org/

[61] 24/7 est couramment utilisé pour décrire une connexion Internet qui est disponible 24 heures par jour, sept jours par semaine. 24/7 connexions se réfèrent généralement à des connexions Internet à haut débit tels que DSL

Un super-peer aide les utilisateurs à choisir leurs entreposeurs (*storekeepers*) en leur fournissant le modèle de disponibilité des autres utilisateurs.

Un utilisateur choisit un autre utilisateur en tant que son entreposeur quand il peut obtenir un accord avec l'utilisateur invité. Pendant la disponibilité d'un utilisateur, des mises à jour sur son contenu (*par exemple, commentaire sur photos, vidéos etc*.) sont transmises à cet utilisateur seulement.

Les répliques sont mis à jour quand l'utilisateur se déconnecte, et les mises à jour subséquents sont gérées jusqu'à ce que l'utilisateur devient de nouveau en ligne. Pour assurer la confidentialité des données, elles sont cryptées dans les répliques. Quand un nouvel utilisateur rejoint le système, son super-peer fournit un service de storekeeping temporaire jusqu'à ce que le nouvel utilisateur forme des groupes de réplication.

Les Super-peers sont les blocs fonctionnels de base de l'architecture de SuperNova. Ils sont responsables de fournir le service de recherche (*par exemple, trouver réplique potentielle, trouver des amis*), service de stockage, services de comptabilité, service de recommandation pour une partie des utilisateurs du système, et de garder une trace des répliques de ces utilisateurs. La participation des super-peers au système leur coûte de la puissance de calcul, la bande passante et l'espace de stockage. Les auteurs **[66]** discutent brièvement une incitation à base de publicité pour les super-peers, de sorte que les super peers peuvent mettre des publicités sur les profils d'utilisateurs de tirer le profit monétaire. Cependant, les modèles d'incitation proposées ne sont pas très concrètes et un modèle d'incitation plus concret est nécessaire pour motiver fortement les utilisateurs à fournir un service de super-peers au coût de leurs ressources.

5. Prototype des réseaux sociaux distribués

Nous avons étudié soigneusement les architectures et les composants des principaux réseaux sociaux, et nous les avons simplifiés afin de les présenter comme suit :

5.1 PeerSon

PeerSon a une architecture à deux tiers. Logiquement, le premier tiers sert de service de recherche (*lookup service*) qui utilise actuellement une table de hachage distribuée (*DHT*) pour le service de consultation. Le deuxième niveau est constitué de pairs et contient les données de l'utilisateur, tels que les profils.

Les services de recherche stockent les métadonnées nécessaires pour trouver les utilisateurs et les données qu'ils stockent, par exemple, l'adresse IP, les informations sur les fichiers et les notifications pour les utilisateurs. Un pair qui veut se connecter à un autre pair demande le service de recherche directement pour obtenir toutes les informations nécessaires. Les pairs alors se connectent directement. Après avoir échangé un message ou un fichier, les pairs se déconnectent immédiatement, sauf pour la messagerie instantanée. La figure 3.2 montre l'architecture simplifiée du PeerSon.

La logique de cette architecture à deux tiers est de permettre aux clients légers tels que les téléphones mobiles ou les ordinateurs de poche à utiliser le système, en dépit de leurs contraintes de ressources qui les empêchent d'être des nœuds DHT fiables.

Sonja et .al **[55]** ont défini une approche P2P distribuée couplée avec un système de chiffrement et une extension de l'approche décentralisée par l'échange directe des données entre les utilisateurs.

a- 1er tiers (*le lookup service*)

Il s'agit d'un système de consultation ou de recherche (*lookup service*) des DHT, qui stocke les métadonnées telles que : Les adresses IP, les informations sur les fichiers, les informations sur les utilisateurs, …

Ces métadonnées sont nécessaires afin de pouvoir trouver les utilisateurs, leurs statuts et les données qu'ils stockent.

PeerSon utilise OpenDHT[62], donc si l'utilisateur est hors ligne, alors la DHT pourra stocker les données qu'elle reçoit pour une durée limitée.

b - 2eme tiers (*Les peers*)

Ce tiers est constitué des peers et contient les données des utilisateurs. Chaque utilisateur qui se connecte au réseau contactera d'abord le lookup service pour le prévenir de sa connexion. Un utilisateur qui souhaite communiquer avec un autre de manière synchrone ou asynchrone interroge d'abord le lookup service pour obtenir toutes les informations nécessaires. Les peers se connectent directement entre eux après cela et interrompent leur lien de conversation directement après l'échange des messages asynchrones.

Une représentation simplifiée de l'architecture PeerSon est montrée dans la figure 3.2 **[65]**.

[62] http://www.opendht.org/

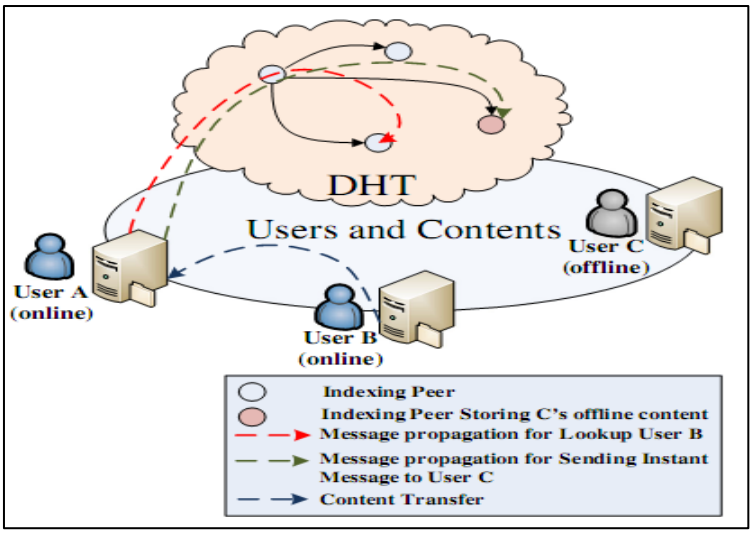


Figure 3. 2: Architecture PeerSon

5.2 Safebook

Leucio Antonio et .al **[58]** ont proposé une nouvelle approche pour s'attaquer aux problèmes de sécurité et de confidentialité liés aux réseaux sociaux centralisés. Le prototype de cette approche est écrit en python et peut être exécuté sur de multiples systèmes d'exploitation. Il est composé de quatre gestionnaires différents **[67]** :

- Le gestionnaire de communication qui s'occupe de l'envoi et de la réception des paquets sur le réseau.
- Le gestionnaire de S2S (*une DHT dérivée de Kademlia* **[68]** *est utilisé comme système de P2P , voir* **[69]**) qui construit principalement l'overlay DHT.
- Le gestionnaire de Matryoshka (*la notion de Matryoshka sera détaillé par la suite*) qui se base sur la couche du réseau social.
- Le gestionnaire de l'utilisateur qui implémente l'interface utilisateur.

La figure 3.3 représente l'architecture de Safebook **[65]**.

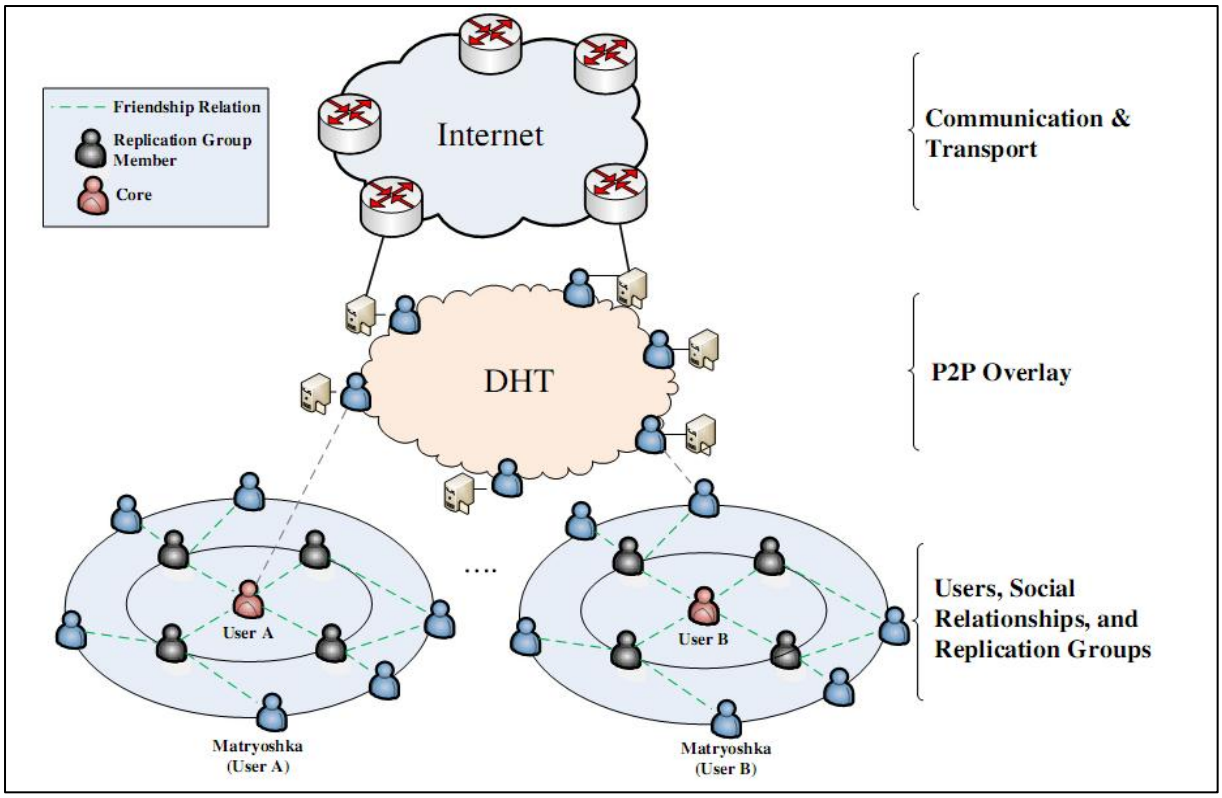


Figure 3. 3: : Architecture Safebook

Afin d'assurer la confidentialité des utilisateurs face à d'éventuelles violations de la vie privée par le fournisseur, l'approche proposée adopte une architecture 3 tiers. Cette approche est décentralisée avec un mapping directe entre les trois couches qui constituent le réseau social, en s'appuyant sur la coopération entre un certain nombre de parties indépendantes qui sont également des utilisateurs de l'application de ce même réseau.

Les couches et principaux composants de Safebook sont représentés dans la figure 3.4 **[63]** et détaillés comme suit :

- La couche du réseau social centrée sur l'utilisateur implémente le niveau du réseau social qui est la représentation numérique des membres et leurs relations dans les réseaux sociaux en ligne.
- Le service de management offert par le fournisseur du réseau social est implémenté dans la couche du support P2P afin de gérer l'infrastructure de l'application.
- Internet qui, elle, représente la couche de communication et de transport.

Dans Safebook, chaque partie est représentée par un nœud considéré comme un nœud hôte dans la couche Internet, un nœud peer dans la couche P2P et nœud membre dans la couche du réseau social. Ces nœuds forment trois types de couches :

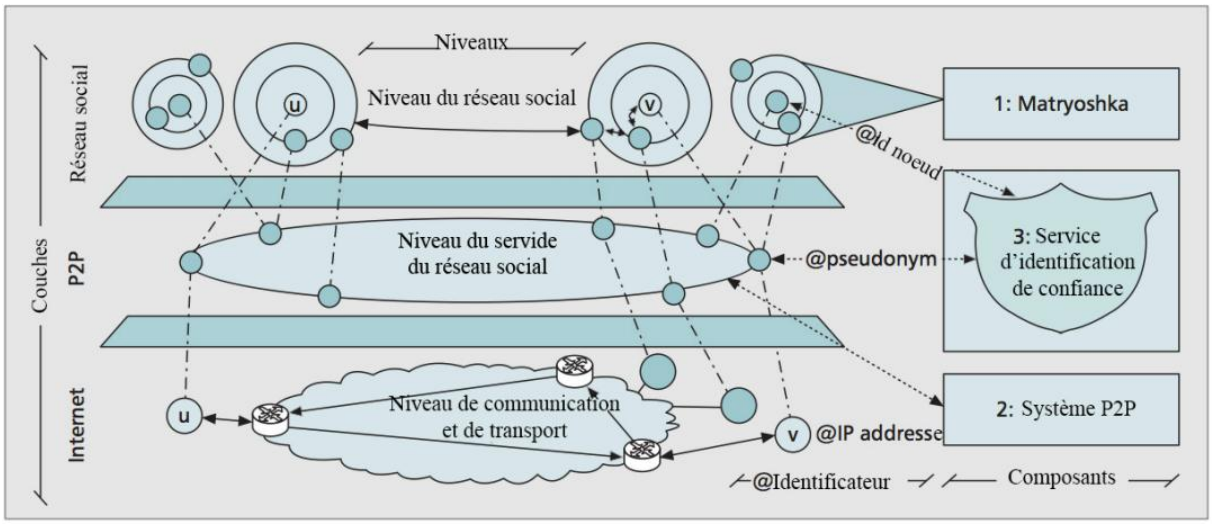


Figure 3. 4: :Couches de Safebook (*à droite*) et principaux composants (*à gauche*)

a- Matryoshka

Un ensemble de Matryoshka est une structure concentrique dans la couche du réseau social, elle fournit le stockage des données et la confidentialité des communications autour de chaque nœud.

Les Matryoshkas sont construites autour de chaque nœud membre afin de fournir un stockage de données fiable, une possibilité de récupération de données et un brouillage des communications.

Chaque Matryoshka, comme représentée dans la figure 3.5 **[63]**, protège le centre des nœuds qui est appelé noyau, ce qui, dans la couche du réseau social est adressé par son identificateur de nœud. Ce noyau est encerclé par des nœuds. Ces nœuds sont reliés par des chemins radiaux sur lesquels les messages peuvent être relayés de manière récursive à partir de la couche externe vers le noyau et vice versa. Tous les chemins sont basés sur des relations de confiance apparentées au réseau social. Chaque saut connecte une paire de nœud appartenant à des utilisateurs liés par une relation de confiance dans la vie réelle.

Les nœuds qui sont les plus internes et, ceux les plus externes ont un rôle particulier.
Le cercle **[58]** le plus interne se compose des nœuds appartenant aux contacts de confiance du noyau, chacun de ses contacts stocke chez lui les données de ce noyau sous forme cryptée.

Ces nœuds sont donc appelés miroir. Quant aux nœuds du cercle le plus externe, ils sont considérés comme une passerelle pour toutes les demandes adressées à ce noyau. Ils sont appelés des points d'entrées.

Les nœuds **[58]** sur le cercle intérieur mettent en cache les données pour le noyau et servent les requêtes si le noyau est déconnecté. Toute demande vers un noyau, lui est adressée en utilisant son identificateur, qui peut être retrouvé dans la DHT. Cependant, la communication en temps réel est transmise par le noyau lui-même, par contre les communications en mode hors ligne peuvent être desservies par un de ces miroirs.

Le nombre de miroirs et de points d'entrée dans chaque voie est fixé. Par contre, le nombre de nœud entre eux peut être variable, ce qui conduit à des chemins de longueurs variables dans la même Matryoshka.

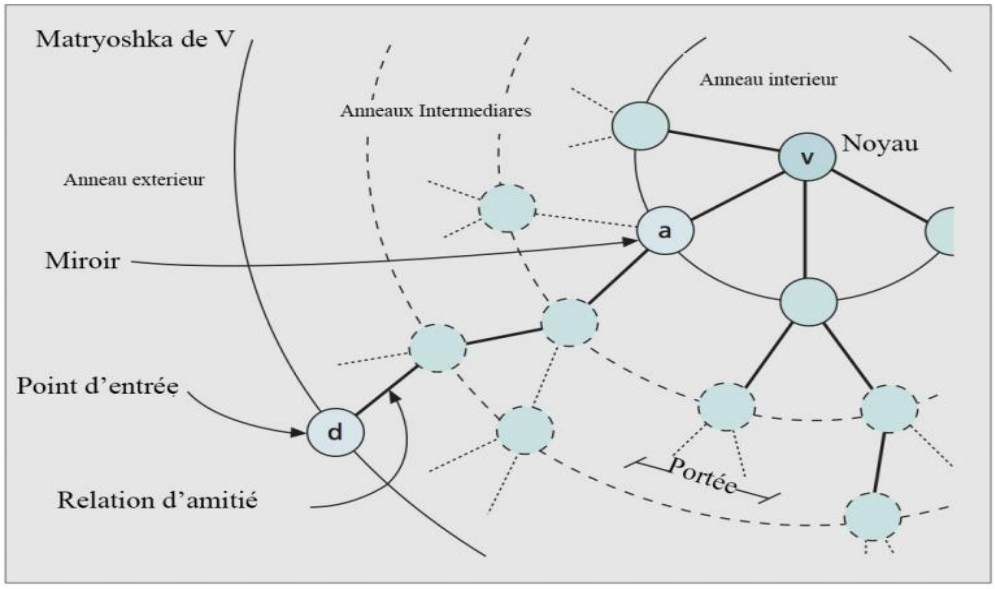


Figure 3. 5: Structure d'une Matryoshka

Il est important de noter que les nœuds sur le même cercle ne partagent pas nécessairement des rapports de confiance entre eux-mêmes, à l'exception du cercle intérieur, dans lequel tous partagent leur relation avec le noyau.

b- Système P2P

Afin de fournir un service de localisation pour trouver les points d'entrés vers une Matryoshka d'un utilisateur, les nœuds créent un support P2P. Actuellement ce support ressemble à Kademlia **[68]**; ainsi les pseudonymes sont utilisés comme identifiants pour la

DHT. Les clés de recherche qui sont enregistrées sont la propriété des membres participants, ainsi que leurs identifiants de nœuds.

Cependant, la communication en mode P2P ne repose pas sur des liens de confiance contrairement à la voie à travers une Matryoshka. Toutefois, l'utilisation des pseudonymes permet la protection des membres contre la violation des droits de la vie privée fondée sur l'identification et la recherche des nœuds via des liens P2P non fiables.

c - Service d'indentification de confiance

Ce service garantit que chaque utilisateur de Safebook obtient au plus un identifiant unique dans chaque catégorie d'identifiants. Il est basé sur une procédure d'authentification nommée out-of-band (*Hors bande*) (*voir la référence* **[70]**). Ce service d'identification de confiance accorde à chaque utilisateur une paire unique d'identifiants de nœud et de pseudonyme. Cette paire est une clé résultante du calcul d'une fonction de hachage sur l'ensemble des propriétés qui l'identifient de façon unique dans la vie réelle telles que : nom, prénom, date de naissance, lieu de naissance, …

Il est à noter que ce service d'identification de confiance est utilisé comme une troisième partie qui est centralisée et semble s'opposer au but de la décentralisation. Cependant, il ne constitue aucune menace à la vie privée, car il ne peut pas tracer les utilisateurs ou leurs messages échangés.

5.3 DECENT et Cachet

DECENT comme montré dans la figure 3.6 est basé sur une architecture 2 tiers où les données sont stockées comme des objets dans une DHT qui utilise un « object ID » comme clé en plus d'un standard « push and get operation » ; la DHT supporte aussi une opération additionnelle au stockage dans les nœuds avec la WritePolicy **[71]** sur les objets.

La disponibilité des données est assurée par leur réplication . Le lookup service utilise le protocole de la WritePolicy pour empêcher les utilisateurs malveillants de modifier ou supprimer les données disponibles dans la DHT parce qu'ils n'ont pas la bonne signature. Celle-ci est générée par le hachage de la clé publique de la WritePolicy et le cryptage des données **[64]**. Elle fera partie de l'objet métadonnée également crypté. Donc le nœud de stockage refusera de réécrire l'objet stocké sauf si la nouvelle donnée est signée proprement par la clé WritePolicy.

La clé publique de la WritePolicy est aléatoire pour chaque objet afin d'empêcher la liaison d'un objet à son propriétaire.

Cependant, vu que Cachet est une évolution de DECENT, alors il se base bien évidemment sur son architecture mais en rajoutant un tiers, là ou dans DECENT il n'y avait que les peers et les DHTs, Cachet a rajouté des DHTs pilotes comme montré dans la figure 3.6. Ces DHTs contiennent l'annuaire globale du réseau, parce qu'elles ont connaissance de contacts appartenant à quelle DHT. Cela facilitera la recherche et optimisera le temps.

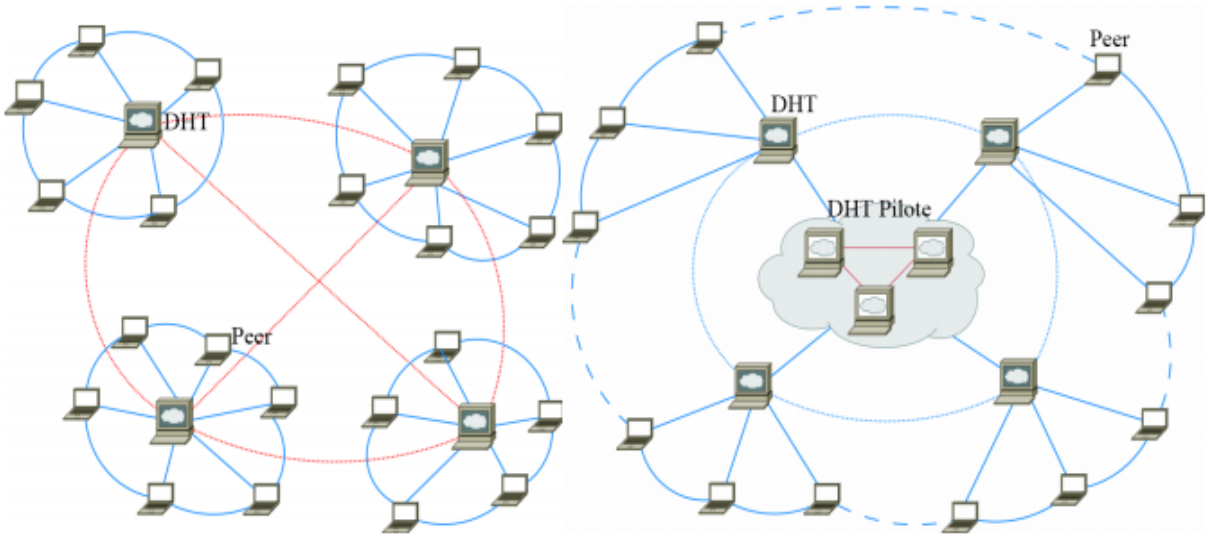


Figure 3. 6: : Architecture de DECENT à gauche et architecture de Cachet à droite [63]

5.4 SuperNova

Nous décrivons d'abord les différentes entités du système avant de décrire le processus de démarrage d'un nouveau nœud. Nous décrivons également les différentes stratégies qu'un nœud peut sélectionner pour augmenter la disponibilité de ses données.

SuperNova introduit des super nœuds pour l'amorçage et a contourné les inconvénients de l'utilisation des nœuds des amis pour le stockage **[72]**.

La figure 3.7 montre l'architecture du SuperNova

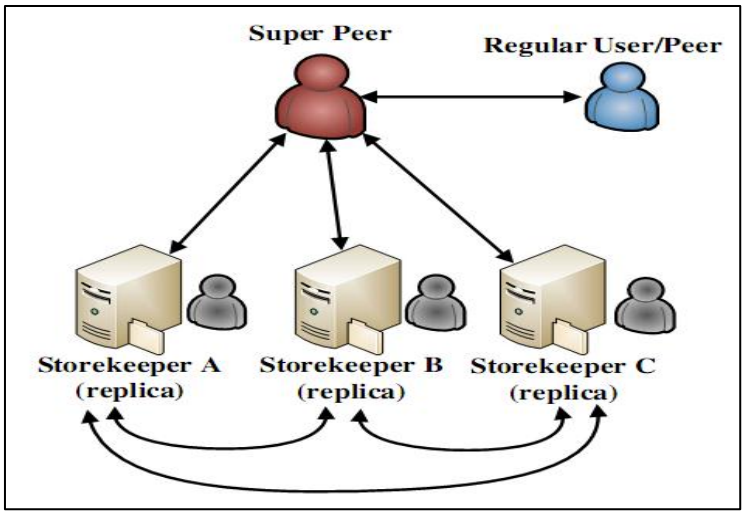


Figure 3. 7: Architecture de SuperNova [65]

Les différentes entités de l'architecture SuperNova sont :

a. Profil : chaque nœud (*ou utilisateur*) dans DOSN est représenté par un profil, ainsi que des données d'utilisateur qui peuvent être publiques, ou pour protégées (*accès limité à un sous-ensemble des amis*) ou privées et inaccessible à toute personne (*par exemple des données pour le backup*).

b. Friendlist : chaque profil utilisateur est lié avec son profil de son ami, l'ajout d'un l'utilisateur soit sur la base de connaissances ou d'intérêt similitude. La collection de tous les profils liés est appelé friendlist. L'amitié est supposée une relation symétrique.

c. Entreposeurs (*Storekeepers*) : les entreposeurs (*storekeepers*) sont la liste des utilisateurs qui ont accepté de garder des répliques de données d'un autre utilisateur de sorte que quand un nœud particulier ***n*** est en panne, alors les amis de ***n*** peuvent communiquer avec les entreposeurs pour accéder aux données de ***n***. En d'autres termes, les entreposeurs de ***n*** donnent une visibilité à ***n*** , quand il est en panne. Chaque ami n'est pas prêt à agir en tant que entreposeur en raison de la charge précédente ou pour des raisons personnelles.

Chaque entreposeur gère une liste de nœuds pour lesquels ils font un storekeeping pour la synchronisation de données. Contrairement à l'amitié, storekeeping n'est pas une relation symétrique. Pour un nœud particulier ***n***, chaque ami dans son friendlist connaît la liste de tous les entreposeurs pour le profil (*utilisateurs*).

Durant la période initiale de temps, quand un nœud n, n'a pas assez d'amis, il demande à l'un des super-peers de stocker ses données. Ainsi, pour les nouveaux nœuds, les super-peers agissent comme les entreposeurs.

Les données de profil de chaque nœud est divisé en trois types à savoir :

- Publique (*Pb*), protégé (*Pt*) et privé (*Pv*). Chaque nœud dispose de liste d'amis F_1, F_2, ..., F_f dont son profil est lié.

Chaque ami F_x ($1 \leq x \leq f$) dispose des informations sur la liste des entreposeurs (SK_1, SK_2, .., SK_{sk}) qui stockent des données protégées du nœud n.

Chaque entreposeur SK_y($1 \leq y \leq sk$) dispose des informations sur tous les autres entreposeurs qui stockent les données de nœud n. Les amis peuvent accéder à des données protégées et publiques. Les données publiques sont accessibles par n'importe quel nœud dans le réseau, alors que les données privées est uniquement pour le nœud n lui-même.

d. Super-peers : les Super-peers sont l'une des entités les plus importantes dans l'architecture SuperNova. Tout nœud peut devenir un super-peer en fournissant ses services au système en général, et surtout aux nouveaux nœuds. Les Super-peers fournissent du stockage pour les nouveaux nœuds qui n'ont pas assez d'amis dans le réseau pour une période initiale. Ils maintient également et gèrent les différents types de services (*par exemple, l'entretien de la liste d'utilisateurs*) pour DOSN en coopérant entre eux.
Dans le système envisagé, les différents super-peers peuvent fournir différents types de services. Certains d'entre eux pourraient être offrant un bon stockage et ressources en bande passante, d'autres sont peut-être effectuer une bonne qualité de comptabilité, des recommandations et la mise en relation (*par exemple, pour aider les nouveaux utilisateurs à trouver des amis avec des intérêts partagés, ou trouver les entreposeurs fiables*).

Des mécanismes d'incitation et de réputation appropriée sont nécessaires pour permettre aux super-peers à être équitablement récompensés pour leurs contributions.

e. Synchronisation : indépendamment de l'horloge physique, chaque nœud n maintient une horloge logique pour lui-même et la partage avec tous les amis et entreposeurs. Chaque ami et entreposeurs (*storekeeper*) maintient une horloge séparée pour chaque utilisateur avec lequel il est associé. L'horloge logique est utilisée pour maintenir une synchronisation entre un nœud n et ses amis et entreposeur. Elle est également utilisée pour résoudre de multiples écritures et mises à jour pour un nœud ***n***.
f. Liste des répertoires : ce qui suit est la liste des répertoires étant maintenu dans le système DOSN.

- **Liste des utilisateurs :** l'identification de chaque utilisateur et ses indicateurs des informations publiques (*les nœuds qui stockent les données publiques*). Cette liste est maintenue par des super-peers.
- **Liste des super-peers:** liste des super-peers et leurs services respectifs et les détails de l'accord (*agrément pour stocker les données des autres nœuds*) .Cette liste est maintenue par les super-peers.

6. Prototypes des réseaux sociaux distribués

6.1. Protocole de PeerSon

Cette section décrit comment l'architecture de PeerSon est implémentée et à quoi ressemblent les différents protocoles. Le détail de toute la syntaxe du protocole peut être retrouvé dans **[14]**.

a- Identifiant globale unique (*GUID*) : le système du réseau social distribué doit résister à des attaques importantes comme les attaques Sybil et les dénis de services tout en s'assurant que les identités des utilisateurs sont uniques.
Cette résistance peut être facilement déployée dans un environnement centralisé, hormis le fait que dans un environnement décentralisé le déploiement est très difficile. PeerSon assume qu'aujourd'hui que chaque utilisateur possède une adresse email qui est unique et donc utilisable comme identifiant. Mais pour empêcher les nœuds malicieux dans les DHTs de collecter les adresses emails, un hash de chaque émail est calculé et utilisé comme identifiant unique.

b - Création d'un nouveau profil : quand un nouvel utilisateur voudra s'inscrire sur le réseau PeerSon, il devra télécharger d'abord une application qui lui permettra l'utilisation de PeerSon et le stockage des fichiers.

Lors de la première utilisation, le nouvel utilisateur créera un profil, la DHT utilisera le hachage de son email pour générer un identifiant globale unique (*GUID*). L'utilisateur doit alors confirmer son GUID, et au final la DHT ajoutera alors les métadonnées nécessaires pour sa localisation et son statut. Il est à noter aussi que dans PeerSon l'utilisateur peut se connecter depuis plusieurs endroits, c'est la combinaison GUID/statut qui permet justement ce genre de connexion. La procédure d'inscription est récapitulée dans la figure 3.8.

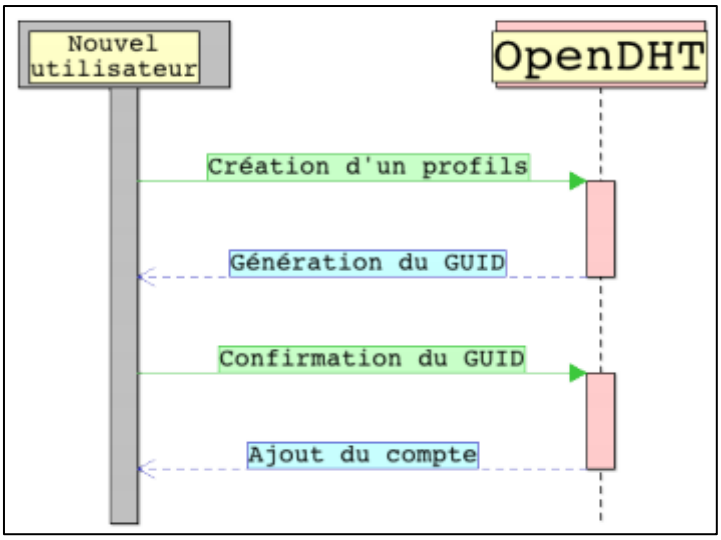


Figure 3. 8: PeerSoN - Inscription

c - Procédure de connexion : se connecter au réseau signifie qu'un certain utilisateur annonce qu'il est actuellement en ligne avec les métadonnées nécessaires pour le joindre, ainsi qu'une liste des fichiers que ce peer stocke.

Cette annonce est envoyée à l'OpenDHT qui à son tour prévient les autres utilisateurs. C'est ce qui permet aux utilisateurs de suivre le statut de leurs amis. L'OpenDHT lui renvoie alors le statut des autres appareils depuis lesquels il a l'habitude de se connecter et si l'un d'entre eux est en mode actif, alors le mode en ligne se mettra sur cet appareil et le mode actif apparaîtra sur l'appareil avec lequel il vient de se connecter. Si aucun appareil n'est en mode actif, alors ce dernier recevra ce mode par défaut.

Dans la figure 3.9, cette procédure est dénommée par le mot login, ces trois étapes ne changent pas pour les autres utilisateurs lorsqu'ils se connectent au réseau.

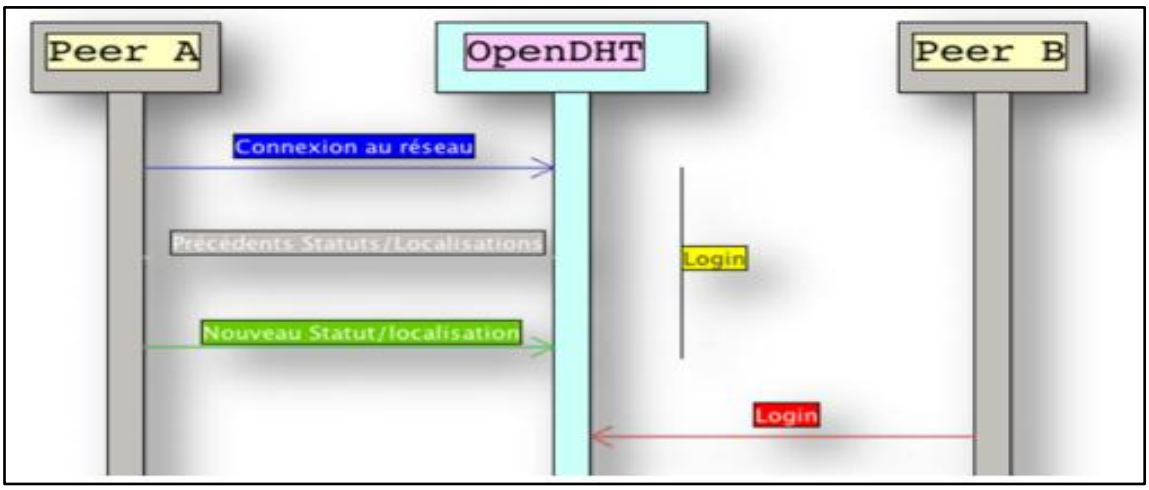


Figure 3. 9: PeerSoN - Connexion au réseau

d - Récupération d'un fichier : les messages sur PeerSon sont traités comme des fichiers. La découverte de ces fichiers est traitée par des requêtes envoyées à la DHT pour trouver quel peer contient le fichier recherché. Le résultat renvoi le nom du fichier si celui-ci existe, le numéro de la (*les*) version (*s*) existante (*s*) ainsi que le nom du peer contenant ce fichier. C'est l'utilisateur demandant qui décidera quelle version il souhaite obtenir et depuis quel peer.

6.2. Prototype de Safebook

Safebook a implémenté différentes opérations des réseaux sociaux en ligne telles que : la création des comptes, la publication des données, la récupération des données, les requêtes de demande de contact et d'acceptation ainsi que la gestion des messages.

Lors de l'ajout d'un nouveau contact, une confiance particulière lui est attribuée de la part de celui qui l'ajoute. Ce niveau de confiance ne peut être connu que par l'utilisateur lui-même et non pas par ses contacts. Donc ses contacts ne peuvent savoir dans quel cercle de confiance ils se trouvent. C'est ce niveau de confiance qui permet de déterminer qui de ces contacts va être utilisé comme miroir pour répliquer ses données.

Pour garantir la confidentialité dans Safebook, les données peuvent être privées, protégées ou publiques.

- Privées : les données ne sont pas publiées
- Protégées : les données sont publiées mais cryptées.
- Publiques : les données sont publiées.

Toutes les données publiées sont répliquées dans les miroirs de l'utilisateur ; cependant, dans Safebook si un utilisateur et tous ses miroirs sont hors ligne alors son contenu sera inaccessible.

a - création d'un nouveau profil : pour rejoindre le réseau Safebook , un nouveau membre doit être invité par un de ses amis de la vie réelle déjà membre enregistré dans le réseau. Le compte de ce nouveau membre est donc créé lors de deux étapes : création de l'identité et création de la Matryoshka **[63]** .

Nous avons simplifié ces étapes de création et nous vous les présentons comme suit:

- **Création de l'identité**

Nous avons supposé que l'utilisateur A est déjà enregistré sur le réseau Safebook. Cet utilisateur va alors envoyer une invitation par email à un de ses amis (*amis, famille, …*) de la

vie réelle. Si cette personne décide de rejoindre le réseau, elle se connecte sur ce dernier et crée un profil. Le profil est créé dans la machine de l'utilisateur et référencé dans la DHT, cette dernière ajoutera les métadonnées du nouveau profil et préviendra le service d'identification de confiance (*S.I.C*)[63]. Le S.I.C va donc demander une fourniture de preuve pour s'assurer qu'il s'agit bien de cette personne en fournissant par exemple des informations telles que le nom, le prénom, la date et le lieu de naissance.

Cette preuve est présente sous forme de processus out-of-band et consiste à rajouter une pièce d'identité, un passeport ou encore une réunion en face à face entre le nouvel utilisateur et la représentation du S.I.C. Cette étape est primordiale, car elle permet d'éviter les attaques d'usurpation d'identité et les attaques Sybil.

Après la vérification de la certitude des informations, le S.I.C calcule l'identifiant du nœud de ce membre qui sera unique dans le réseau. Une fois que l'utilisateur aura reçu son identifiant il pourra alors rejoindre le système P2P en utilisant le contact qu'il l'a invité comme un nœud d'amorçage et peut donc commencer la création de sa Matryoshka.

- **Création de la Matryoshka**

Le nouveau membre a seulement son invitant comme contact pour commencer. Il lui envoie une requête pour créer le chemin qui contient les clés de la recherche qu'il souhaite enregistrer dans la DHT, le temps de vie de cette requête (*TTL*) et le nombre des membres à qui son invitant devrait transmettre la requête. Ce nombre sera appelé **facteur de portée** par la suite.

L'invitant sélectionne un intervalle de portée des sauts suivants et leur transmet ce message d'enregistrement. Ce processus est répété de manière récursive jusqu'à ce que le TTL expire. A son expiration, le dernier nœud qui aura reçu le message d'enregistrement l'enregistre dans la DHT en utilisant son identifiant et son adresse à lui et commence à agir comme étant son premier point d'entrée.

La DHT renvoi alors à ce nouvel utilisateur la confirmation de la création de sa Matryoshka ainsi que l'adresse de son premier point d'entrée. Cependant, parce que le facteur de portée peut être différent d'un chemin à l'autre, plusieurs points d'entrées peuvent être définis et ne pas appartenir à la même couche.

[63] S.I.C : Service d'Identification de Confiance

b - Se connecter au réseau : quand un utilisateur se connecte au réseau, il envoie une requête de connexion à la DHT, cette dernière diffuse l'information que le peer A s'est connecté sur le réseau P2P, ses amis
qui se trouvent en ligne à ce moment vont être informés de sa connexion et si ses miroirs sont en ligne, alors ils synchroniseront avec lui les données asynchrones publiées en son absence comme montré dans la figure 3.10.

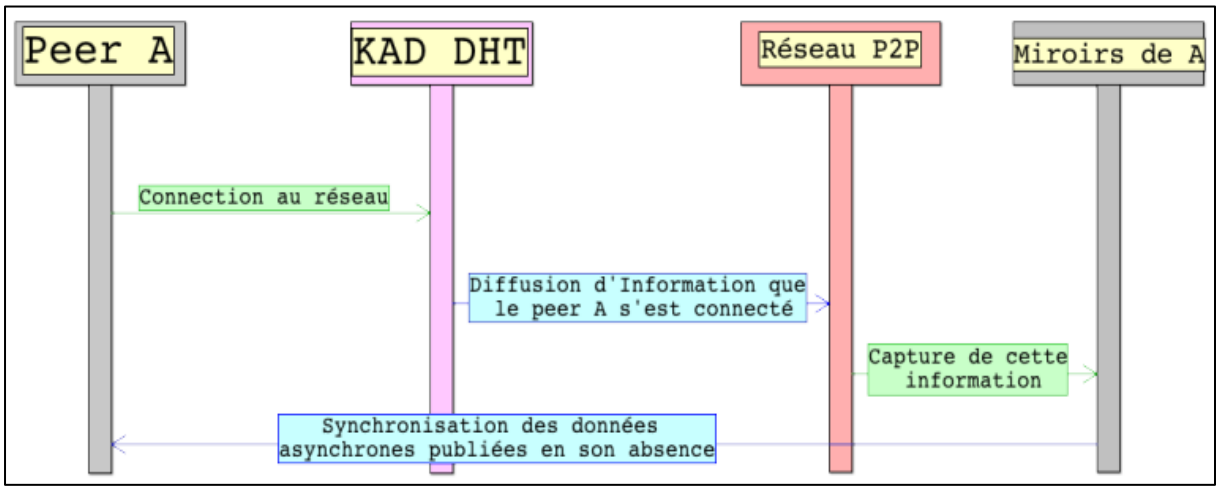


Figure 3. 10: Safebook - Se connecter

6.3. Prototype de Decent et Cachet

DECENT et Cachet ont implémenté différentes parties dans leurs prototypes propre à eux et qui font leurs points forts tels que l'« ABE Policy» **[73]** et la « Cryptography Policy » **[74]**, où ils sont passés de quelques centaines de secondes de cryptage et décryptage chez DECENT à quelques dizaine de secondes seulement chez Cachet. Pour leur prototype, DECENT et Cachet ont développé un mur et une page d'actualité qui pour un utilisateur est une collection des derniers statuts mise à jour par chacun de ses contacts (*amis*) pour avoir une page d'actualité pour chaque utilisateur ; Le format ABE contient la politique nécessaire pour le chiffrement des données, les utilisateurs peuvent savoir s'ils seront aptes à décrypter l'objets ou non et s'ils seront autorisés à lire les informations ou pas donc ils ne perdront pas de temps à décrypter des informations s'ils n'ont pas le droit de les lire.

Dans l'architecture de DECENT, le décryptage des données nécessite beaucoup de temps sans aucune performance de « caching » pour générer la page d'actualité. Par ailleurs, l'utilisateur n'avait pas besoin de lire toutes les actualités de ses contacts, une sélection devient donc nécessaire, elle est introduite dans l'architecture de Cachet.

De plus, Cachet adopte le « social caching », car étant donné l'utilisation de la décentralisation et la cryptographie dans l'architecture précédente qui est DECENT, télécharger et reconstruire le mur d'un contact social ou la page d'actualité est un long processus exigeant les étapes suivantes :

- Décrypter des objets de mise à jour, qui sont ABEncrypted suivant la ABE Policy;
- Rapporter des métadonnées comme une mise à jour ;
- Indexer la clé de décryptage symétrique dans la DHT;
- Accéder à des petits objets multiples situés dans différents nœuds de stockage utilisant la DHT fournie dans l'étape précédente ;
- Décrypter des objets de mise à jour avec leurs clés symétriques correspondantes.

a - Inscription dans le réseau : un nouvel utilisateur sur le site du réseau, créera un nouveau profil, la DHT lui demandera alors de définir son ABE Policy, quand c'est fait, la DHT ajoutera son profil sur le réseau et pourra donc commencer à l'utiliser.

b - Se connecter au réseau : quand un utilisateur A se connecte dans Decent comme montrée à la figure 3.11, la DHT ou il stocke ses données propage l'information qu'il s'est connecté aux DHT voisines, toutes les DHTs font pareil jusqu'à ce que tout le réseau soit au courant, les DHTs qui contiennent l'annuaire de ses amis les notifieront. De son côté, l'utilisateur sera notifié des éventuelles publications le concernant effectuées en son absence.

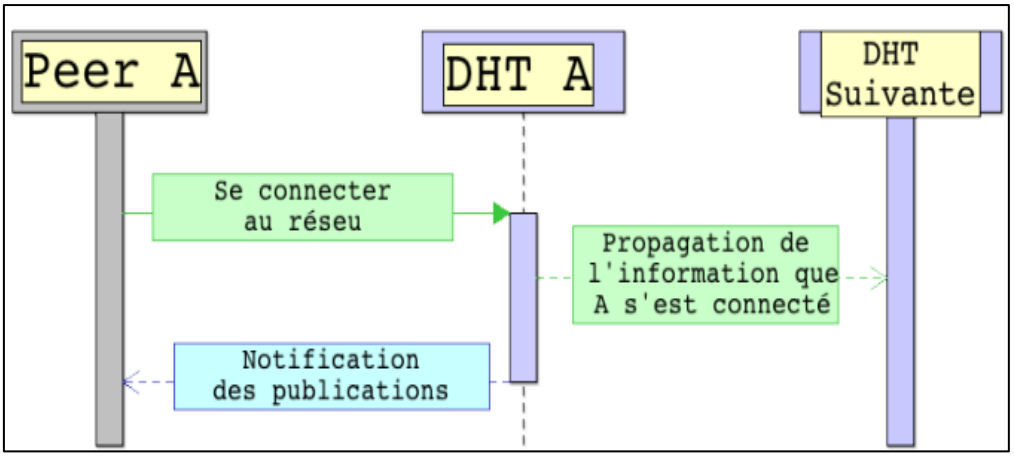


Figure 3. 11: DECENT- Se connecter

c - Recherche d'un contact social : pour permettre aux utilisateurs de rechercher leurs contacts, Cachet propose un service d'administration centralisé qui maintient le mapping entre les contacts et leurs racines (*profiles*). Pour permettre à ce service de connaître les relations des utilisateurs, cette architecture utilise :

- Un canal de communication anonyme ;
- Afin de cacher les noms des utilisateurs dans l'annuaire de la DHT, il est nécessaire d'augmenter le niveau du protocole de sécurité concernant la vie privée.

Il est d'usage de croire que cachet garantit le respect de la vie privée et surpasse tous les autres systèmes existant dans ce domaine mais, beaucoup de changements afin de l'améliorer sont encore à mettre en place tels que:

- l'algorithme de social caching transmet des informations sur les identités des utilisateurs.
- le mur d'actualité révèle si un utilisateur est en ligne ou hors ligne.

d- Suppression et annulation : quand un utilisateur efface un objet ou modifie la politique d'accès à celui-ci, ces changements sont affectés immédiatement aux données. Le protocole de modification n'est pas spécifié dans ce travail, l'utilisateur peut donc envoyer une requête de révocation pour effacer des objets sur Cachet.

e - Protocole de présence : en générale, dans un réseau centralisé le serveur garde la trace de la présence de ses utilisateurs. Cependant, dans Cachet une approche distribuée est appliquée quand chaque nœud enregistre sa présence dans la DHT afin que la présence des utilisateurs dans le réseau soit effective.

La présence de cet utilisateur en ligne a la même structure que les objets et est cryptée avec ABEncryption pour empêcher la DHT de lire le contenu de cet objet. Cette structure est définie par une adresse IP et un numéro de port. Ce qui permet à l'utilisateur d'effectuer des modifications à son profil en ajoutant ou supprimant des informations quand il le désire. La DHT met à jours ces informations qui sont similaires au remplissage de la page d'actualité.

6.4 Prototype de SuperNova

a- Amorçage de nouveau nœud

Un nouveau nœud n_{nn} rejoint DOSN, s'appuie d'abord sur les super-peers pour augmenter sa disponibilité des données. Plus tard, il peut trouver des amis (*ou étrangers*) dans le DOSN à l'aide de super-peers qui peuvent agir comme entreposeur pour lui.

Nous expliquons les différentes phases d'un nouveau nœud n_{nn} au point où il entre dans le réseau jusqu'au point où il s'installe dans le réseau.

1. Phase initiale: quand un nouvel utilisateur joint le DOSN, il crée un profil. Nous appelons cette période initiale comme la «phase initiale» (*IP*). Durée de temps de cette phase

peut être de quelques heures à quelques jours. Nous supposons (*mais pas nécessaire*) que l'utilisateur placera son nom d'utilisateur, l'emplacement et l'intérêt comme données publiques, puisque ce sont des attributs les plus utilisés pour trouver un utilisateur dans le réseau. Une donnée publique est un élément d'information à travers laquelle un utilisateur peut annoncer sur lui-même pour faire de nouvelles connaissances. Dès qu'un profil est créé par l'utilisateur, une entrée est ajoutée à la liste des utilisateurs, qui est maintenu par des super-peers (*nous notons que les systèmes comme Skype suivent une approche similaire pour le maintien d'un annuaire distribué des utilisateurs*) **[75]** .

Puisque un nouveau nœud ne pourrait pas obtenir assez d'amis ou de connaissances immédiatement, donc il pourrait communiquer avec des super-peers pour stocker ses données. Quand un nouveau nœud rejoint le réseau, il peut voir une liste des super-peers (*qui est une information publique*) et la liste des services qu'ils fournissent. Un super-peer annonce la liste suivante de services pour les nouveaux nœuds :

- **Stockage**: quelle quantité de données qu'ils peuvent stocker. Au lieu d'offrir un espace au nouveau nœud, dans le futur le super-peer pourrait prendre l'espace du nouveau nœud pour stocker ses données personnelles ou de stocker les données de certains autres nœuds.
- **Temps d'accord:** pour combien de temps ils peuvent stocker les données des nœuds (*par exemple, pendant un mois*).
- **Temps de disponibilité:** à quels moments de la journée, ils peuvent couvrir l'absence d'un nœud.
- **Type de contenu:** prêt à stocker ce type de contenu.
- **Publicité**: quel genre de publicités sera indiqué sur le profil de l'utilisateur par le super-peer.

Un nouveau nœud sélectionne un super-peer et envoie une demande de storekeeping. Une fois un accord est obtenu entre un super-peer et le nouveau nœud, les données sont répliquées depuis nouveau nœud au super-peer. Le nouveau nœud n_{nn} entre maintenant à **Take Care phase (*TCP*)** où un super-peer se charge des données de n_{nn}. Les données dans le super-peer (*et à l'entreposeur*) sont stockées sous forme cryptée.

2. Take Care Phase (*phase de prise de soin*): une fois un accord est obtenu entre n_{nn} et un super-peer, n_{nn} dans **Take Care phase (*TCP*)**, dans lequel le super-peer contribue à améliorer la disponibilité des données de n_{nn} et agit comme un entreposeur ainsi. Notez que, le

super-peer peut déléguer la tâche de storekeeping à d'autres nœuds qu'ils lui doivent parce que le super-peer avait déjà aidés quand ils étaient de nouveau dans le système.

La période Take Care est le temps où le nouveau nœud essaie de trouver et de nouer des amitiés dans le système.

3.Settled Phase (*phase arrangée*) : un nœud peut prolonger sa phase de prise de soin (*TCP*) avec un super-peer s'il ne peut pas obtenir suffisamment d'entreposeurs. Une fois un nœud dispose de suffisamment d'entreposeurs (**s**) (*autres que les super-peers*) pour gérer ses données, il peut renoncer les services de super-peers. Pour chaque nœud ***n***, chaque storekeeper accepte de stocker une quantité limitée de données qui nécessite de l'espace .

7. Critères de comparaison

Nous présentons l'ensemble de critères dans **[65]** pour comparer différentes propositions de DOSN. Pour chaque critère, nous discutons également les issues qui doivent être prises en compte pour les comparer.

7.1. Architecture

Les premiers critères pour la comparaison est l'architecture des différents DOSNs, c.-à-d., comment les différents composants de système sont organisés pour atteindre les buts de système. Dans le contexte de DOSN, nous nous concentrons en particulier sur l'organisation des composants de système suivants :

- **Contrôle:** le cadre de contrôle de l'architecture est composé de services de recherche (*utilisateurs et Lookup contenu*) et des services de gestion d'identité.
- **Stockage:** la partie de stockage de l'architecture se compose de stockage du contenu de l'utilisateur afin d'assurer sa grande disponibilité.

L'organisation de ces composants ressemble étroitement à celui des systèmes P2P, et nous avons donc le classement suivant:

- **Structuré:** les utilisateurs participent directement pour former un recouvrement structuré ou utiliser les services fournis par un recouvrement structuré tiers. Toute demande dans le système peut être résolue en utilisant un réseau de recouvrement structuré en un nombre borné d'étapes.

- **Semi-structuré:** un sous-ensemble de tous les utilisateurs du système (*des super-peers*) prendre la responsabilité pour stocker l'index et la gestion d'autres utilisateurs. Les super-peers sont chargés de fournir l'interface pour le reste des utilisateurs pour effectuer les différentes opérations du système. La participation des utilisateurs pour fournir un service de super-peer peut être volontaire ou à base d'incitation.
- **Non structuré:** aucun utilisateur dans le système ne maintient un index, et les opérations de propagation utilisées sont généralement effectuées à l'aide des inondations ou des chemins aléatoire.

7.2 Architecture de disponibilité

Dans les OSN_S centralisés, les fournisseurs assurent la disponibilité du système en fournissant des serveurs dédiés et différents mécanismes de tolérance aux pannes. Cependant, en cas de DOSNs, les utilisateurs ont la liberté de stocker leur contenu sur l'appareil de leur choix. Ces appareils ne doivent pas nécessairement avoir une disponibilité élevée (*high uptime*) et ils peuvent être très enclins à des échecs. Par exemple, un utilisateur peut choisir d'héberger une partie du contenu de leur téléphone cellulaire, qui ne pourrait pas avoir la connectivité de l'Internet 24/7 et peut se désactiver si la batterie meurt. Par conséquent, assurer la disponibilité 24/7 de contenu des utilisateurs stockées dans un cadre décentralisé est une question difficile. La réplication et la mise en cache sont des techniques éprouvées pour assurer la disponibilité de la source originale du contenu pourrait ne pas avoir une disponibilité élevée. Dans le cadre de DOSN, nous considérons les questions suivantes concernant la disponibilité:

– **Mécanisme de disponibilité**: les propositions de DOSN montrent deux classes importantes des techniques pour assurer la disponibilité :

- **La réplication et la mise en cache:** le contenu d'utilisateur est répliqué et / ou mis en cache à un ensemble d'utilisateurs afin qu'ils puissent agir comme un proxy[64] lorsque le propriétaire du contenu n'est pas en ligne.
- **Les nœuds stables:** l'architecture suppose que le système de stockage est hautement disponible. Cette hypothèse limite le choix de l'utilisateur d'un système de

[64] Un serveur proxy est un ordinateur ou un module qui sert d'intermédiaire entre un navigateur Web et Internet. Le proxy participe à la sécurité du réseau. Source : http://www.commentcamarche.net/faq/17453-qu-est-ce-qu-un-proxy

stockage. Par exemple, les passerelles domestiques et les décodeurs ont des capacités de stockage et ils ont une disponibilité plus élevée par rapport à un téléphone portable ou un ordinateur portable.

Malgré les mécanismes ci-dessus, certains des propositions de DOSN n'abordent pas la question d'assurer une haute disponibilité à tous.

– **Politique de sélection de réplique (*Replica Selection Policy*):** si le DOSN se charge de la réplication du contenu des usagers alors le choix de placement de la réplique est une autre question importante à considérer. Dans le cadre de DOSN, les politiques suivantes pour le placement de réplique sont possibles:

  - **Répliques sélectionnés par utilisateurs** basées sur la relation de confiance d'un utilisateur avec d'autres utilisateurs.
  - **Répliques sélectionnés par système** basées sur certains paramètres, par exemple, la disponibilité de la configuration de l'utilisateur.
  - **Réplication déterminée par utilisateur**, où la réplication est effectuée en fonction de la souscription par les utilisateurs.

– **Synchronisation de la réplique:** si le contenu de l'utilisateur est répliqué alors garder une seule image de toutes les copies est également un problème.

7.3. Scalabilité

La scalabilité est mesurée comme la capacité d'un système à fournir une bonne ou raisonnable performance selon certaines métriques sous de grandes charges. Dans le cadre de DOSN, les différents utilisateurs deviennent une partie de l'infrastructure et ils ont quelques stockages, bande passante et charges de traitement. Dans un OSN, les utilisateurs et leur contenu sont la charge principale pour l'ensemble du système. Dans un contexte de décentralisation, la fréquence d'un utilisateur allant en ligne et hors ligne, c'est à dire, *le taux de désabonnement churn*[65], met également une surcharge (*overhead*) sur les composants du système. Ainsi, les paramètres de charge suivants sont importants pour un DOSN:

[65]Le churn : le terme « churn » désigne la dynamique des environnements pair-à-pair, il caractérise l'arrivée et le départ continu d'une partie des membres du réseau au sein du système. On assimile généralement le départ d'un nœud à une panne et on considère qu'un nœud est en panne lorsqu'il cesse de communiquer avec le reste du système pendant une période prédéfinie, laquelle varie en fonction des besoins du système. Aussi, on ne sait pas quand un nœud décidera de quitter le réseau et on ne peut pas contraindre un pair à rester connecté au système contre sa volonté. En conséquence, un environnement pair-à-pair est un réseau composé d'un nombre de nœuds potentiellement grand, dont on ne maîtrise ni le comportement, ni la disponibilité.

- Nombre d'utilisateurs dans le système,
- Nombre d'amis d'un utilisateur,
- Le volume de contenu d'un utilisateur,
- Le volume du contenu dans le système,
- Churn rate

Pour analyser la scalabilité d'un système, nous avons mis l'accent sur le changement des paramètres mentionnés ci-dessus sur les mesures de performance suivantes pour chaque utilisateur:

- Bande passante du réseau,
- Stockage,
- Overhead de Mise à jour

7.4. Contrôle de confidentialité

Les utilisateurs peuvent utiliser les mécanismes de contrôle de la vie privée pour contrôler la mesure dans laquelle leurs données est visible aux autres utilisateurs de l'OSN. Un service d'OSN fournit des politiques de confidentialité pour le contenu stocké dans le réseau.

Le changement dynamique dans l'intimité de contenu est également une autre préoccupation qui donne aux utilisateurs la possibilité de modifier la visibilité de leur contenu à tout moment.

8 .Classification comparative

Les critères de comparaison que nous avons sélectionnés pour comparer les quatre architectures sont les types d'architectures, les DHTs et leurs services, la disponibilité, et la scalabilité de ces architectures.

- **Architecture**

La plupart des propositions pour DOSN (*PeerSon, Safebook, Cachet et DECENT*) utilisent une superposition de P2P structuré (*structured P2P overlay*) pour le contrôle.

SuperNova propose d'utiliser une architecture semi-structurée en fonction de super-peers, où chaque super-peer est responsable de la gestion d'un certain nombre d'utilisateurs

http://linuxfr.org/news/p2p-hacker-fr-premier-etat-de-l-art-sur-la-decentralisation

réguliers. Le super peer offre un service de lookup, suit le temps de disponibilité de l'utilisateur pour recommander les utilisateurs appropriés pour la réplication, gère les répliques de ses utilisateurs, et agit comme une sauvegarde finale en cas toutes les répliques d'un utilisateur échouent.

- **Scalabilité**

Les architectures structurées présentent généralement une relation logarithmique entre le nombre de messages et le nombre d'utilisateurs dans le système. Avec l'augmentation du nombre d'utilisateurs, il y a peu d'augmentation du nombre de messages générés dans le système. Par conséquent, avec une augmentation du nombre d'utilisateurs dans le système, la bande passante du réseau ne devienne pas un goulot d'étranglement de l'évolutivité pour la partie contrôle de PeerSon et Safebook. Cependant, la bande passante du réseau peut devenir un goulot d'étranglement de la scalabilité pour les diffuseurs, à savoir, les utilisateurs avec un grand nombre d'amis ou les suiveurs dans le système.

Dans une architecture semi-structurée basée sur les super-peers comme SuperNova, les super-peers ont des besoins de stockage plus élevés par rapport aux utilisateurs du système, car ils fournissent une sauvegarde d'un grand nombre d'utilisateurs en cas de panne de l'ensemble des répliques des utilisateurs. Par conséquent, les super-pairs peuvent constituer un goulot d'étranglement potentiel sur l'évolutivité si le réseau de super-peer n'est pas mise à l'échelle avec le nombre d'utilisateurs dans le système. Aussi, la SuperNova update-sur-échec politique encourt haute charge de mise à jour sous haut taux de churn. Ainsi, SuperNova n'évolue pas bien avec une augmentation du nombre d'utilisateurs et un taux de churn élevé.

Par conséquent, les super-pairs peuvent constituer un goulot d'étranglement potentiel de la scalabilité si le réseau de super-peer n'est pas redimensionné avec le nombre d'utilisateurs dans le système.

D'autre part, bien que Safebook utilise également la réplication, la plupart des utilisateurs Safebook encourent moins d'overhead de stockage que SuperNova. Ceci est dû à sa stratégie de réplication de contenu à des amis très proches, qui sont petits en nombre pour la plupart des utilisateurs. Toutefois, les diffuseurs Safebook peuvent exiger de reproduire le contenu d'un grand nombre d'utilisateurs, souffrent de l'évolutivité en termes de surcharge de stockage.

- **Disponibilité**

PeerSon reconnaît les problèmes de disponibilité dans DOSN, cependant, il ne prévoit pas de mécanisme pour assurer une haute disponibilité du contenu dans le système **[65]**.

Concernant les autres systèmes, le contenu d'un utilisateur est répliqué à un certain nombre d'autres utilisateurs qui peuvent servir comme un proxy lorsque l'utilisateur se déconnecte.

Cachet et DECENT répliquent les données de leurs utilisateurs dans une table de hachage distribuée (*utilise Neighbor replication*[66]) .Bien que cette approche assure la disponibilité, elle augmente également l'overhead [67] de communication. Comme OSN éprouvent habituellement un taux de churn élevé , les données ont souvent besoin d'être transféré au départ des nœuds vers d'autres membres de DHT .

Ceci est particulièrement le cas pour les nœuds mobiles. Aussi, Cachet ne minimise pas le nombre de répliques, ce qui augmente l'overhead de garder toutes les répliques des données d'un utilisateur mise à jour.

Concernant Safebook, il reflète les données de chaque utilisateur à un sous-ensemble de leurs amis directs. Malheureusement, un utilisateur dépend donc de ses contacts sociaux pour le stockage de données, comme il a besoin de suffisamment d'amis appropriés qui se qualifient comme nœud miroir. Cela est difficile pour de nombreux utilisateurs dans OSN qui maintiennent peu de liens sociaux. En conséquence, Safebook obtient généralement de faible taux de disponibilité de données.

SuperNova fourni un stockage des données de leur utilisateurs par altruisme. Cependant, l'approvisionnement altruiste, généralement à partir d'un ensemble limité de bénévoles, est peu probable pour satisfaire la demande d'un réseau social à grande échelle avec un grand nombre de plusieurs centaines de millions d'utilisateurs

Parmi tous les DOSNs que nous avons étudié, seulement Safebook permet à un utilisateur de créer un groupe de réplication basée sur la relation de confiance ou d'amitié avec d'autres utilisateurs. Le placement des répliques dans SuperNova et Cachet est réglé par le système, c.-à-d., par les super peers et les nœuds de DHT, respectivement.

[66] Neighbor replication : appelée aussi la réplication simple où les réplicas de données sont stockées dans le pair racine, et dans ses successeurs, ou dans ses prédécesseur pour la DHT Chord , ou dans ses leafset pour la DHT pastry.
[67] Overhead :les messages générés lors de migration des répliques

En comparant toutes les approches de stockage de données mentionnées ici, les systèmes SuperNova essaient de tirer parti des avantages de la combinaison de différentes approches, en permettant une multitude de choix de stockage et de stratégie. Les super-peers dans SuperNova aident à réaliser un nuage de stockage par les pairs provisionné

Les utilisateurs peuvent stocker leurs données à des amis ou des super-pairs ou à nœuds étranges. L'adaptation, la coordination et la comptabilité des nœuds étranges sont effectuées par des super-peers.

Le stockage basé uniquement sur les relations sociales et la confiance, tels que dans le système de Safebook, comporte certaines limites et peut conduire à faible niveau de

disponibilité, car les utilisateurs ne bénéficient que dans les limites de leurs amis les plus proches individuelles, plutôt qu'au niveau de la communauté au sens large.

Afin d'offrir les mêmes caractéristiques que les réseaux sociaux en ligne centralisés, des solutions décentralisées doivent surmonter certains défis tels que la disponibilité des données.

Le tableau comparatif suivant résume les différentes caractéristiques pour les DOSNs étudiés :

Système / **Caractéristiques**	**PeerSon**	**Safebook**	**Cachet**	**SuperNova**
Architecture	structurée	structurée	structurée	Semi-structurée
Indépendance des nœuds puissants	Oui	Oui	Oui	Non
Stockage des données	DHT	Les amis de confiance (*nœuds miroirs*)	DHT	Super peer
DHT	OpenDHT	Kademlia	FreePastry	-
Service DHT	- Routage - Stockage pendant une durée de 7 jours	- Routage	- Routage - Stockage des données	-

Mécanisme de disponibilité	Non abordé	Réplication	Réplication et Mise en cache (*Caching*)	Réplication
L'Overhead généré	Faible Overhead	Faible overhead	Overhead élevé	Faible Overhead
Visibilité de contenu	Non abordé	Groupe	Tous	Tous
Haute Disponibilité de données	Non	Non	Oui	Non
Mode hors ligne	Oui si Friendstore en ligne	Oui si miroir en ligne	Oui toujours	Oui toujours
Placement de répliques	Non abordé	Par l'utilisateur (*sur les nœuds amis*)	Par le système (*placé par la DHT*)	Par le système (*placé par le super peer*)
Scalabilité pour la Bande passante du réseau	Non scalable pour Diffuseurs (*Broadcasters*)	Non scalable pour Diffuseurs (*Broadcasters*)	Non scalable pour Diffuseurs (*Broadcasters*)	Non scalable pour Diffuseurs (*Broadcasters*)
Scalabilité pour le stockage	Non abordé	Moins scalable avec l'augmentation du contenu	Scalable	Non scalable pour les super-peers quand le nombre d'utilisateurs augmente
Mode connexion	Avec ou sans Internet	Connexion Internet requise		

Tableau 3. 1: Synthèse des architectures DOSN étudiées

9. Conclusion

Dans ce chapitre, nous avons étudié les différents aspects de quatre propositions DOSN. Tandis que cet ensemble n'est pas censé pour être approfondi, il fournit une vue globale de différents choix de conception.

Nous avons sélectionné un ensemble de critères pour comparer les propositions des différents DOSNs.

Enfin, nous avons fourni par une analyse comparative des propositions DOSN, la base de notre ensemble de critères proposés.

Chapitre 4

Proposition d'une architecture P2P pour les réseaux sociaux

1. Introduction

Le chapitre précédent a été consacré à l'étude de l'état de l'art où nous avons présenté, discuté, synthétisé et comparé les travaux récents les plus intéressants qui traitent les architectures p2p pour les réseaux sociaux (*DOSN*[68]). Lors de cette étude, nous avons constaté l'existence de plusieurs défis pour la conception des architectures DOSNs concernant la disponibilité des profils des utilisateurs, et l'overhead élevé engendré lors de la migration des données.

Dans ce chapitre, nous présentons notre proposition d'une nouvelle architecture p2p pour les réseaux sociaux. L'architecture proposée fournit une solution qui améliore les performances en utilisant une architecture structurée hiérarchique.

Nous détaillerons certains aspects et caractéristiques de cette architecture (*structure logique du réseau, stratégie de routage, l'arrivée et le départ des nœuds,…*).

Mais avant cela, nous illustrons la nécessité de proposer une nouvelle architecture p2p pour les réseaux sociaux en se basant sur les problèmes tirés des architectures précédemment étudiées.

[68] DOSN :Decentralized Online Social Network

2. Problématique et motivation

Les DOSNs à base de DHT sont tous des systèmes structurés purement plats et non hiérarchiques (*ex. PeerSon, Safebook, Cachet, Decent*). Ces systèmes classiques organisent les peers, ayant la même responsabilité, en un seul réseau overlay avec une performance de recherche de *O(log(N))*, pour un système à *N* peers. Cependant, l'utilisation d'un système DHT plat ne considèrent pas ni l'autonomie des organisations virtuelles et leurs intérêts conflictuels ni le principe de la localité, qui est une considération cruciale dans les réseaux sociaux en ligne.

Le principal avantage de l'utilisation de ces infrastructures dans un cadre à grande échelle est leur grande évolutivité et l'auto-organisation.

Mais, dans un état de churn élevé[69], le maintien de la structure de DHT génère un overhead très élevé et par conséquence il dégrade les performances du système. Pour la disponibilité des données, ces DOSNs à base de DHT plats répliquent les données des utilisateurs dans des nœuds de la DHT choisis aléatoirement. Bien que cette approche assure la disponibilité, elle augmente également l'overhead de communication, produit par le churn, où les données doivent souvent être transférées au départ des nœuds vers les autres nœuds de DHT.

Comme la taille du système augmente, les peers faibles du DHT peuvent gravement compromettre la scalabilité et la stabilité de l'ensemble du réseau.

Des recherches récentes **[76]** ont prouvé que les overlays hiérarchiques ont les avantages de réduire les délais de lookup, avec moins de messages échangés (*moins d'overhead*) entre les nœuds, et une meilleure scalabilité.

En se basant sur l'étude faite sur les architectures DOSNs existantes, nous proposons une nouvelle architecture hiérarchique pour la réalisation de réseaux sociaux en ligne décentralisés[70] (*DOSNs*) qui permet d'améliorer les performance du système tout en réduisant le coût de routage et l'overhead généré par la maintenance des tables de routages et la migration des répliques des données utilisateurs (*leurs profils*).

[69] Churn : la connexion/déconnexion fréquente des peers.
[70] nous considérons les termes décentralisé et peer-to-peer équivalent et référent à ces systèmes comme décentralisée réseau social en ligne (*DOSN*) dans le nôtre mémoire.

L'objectif de notre proposition est de minimiser le nombre de sauts dans le processus de localisation des nœuds qui stockent les données (*ou des ressources*) dans un réseau P2P purement décentralisé, et d'obtenir un overhead limité en répliquant les données dans les nœuds les plus disponibles.

Pour remédier à ces problèmes, notre solution tourne autour de l'organisation des nœuds dans une architecture hiérarchique à deux niveaux (*2 tiers*). En divisant le réseau plat en des régions plus petites, la complexité du système réduit ainsi. À savoir, un exemple typique est le Domain Name System (*DNS*), qui réalise l'évolutivité grâce à une conception hiérarchique.

Des études **[77]** ont identifié un grand degré d'hétérogénéité dans les capacités et les durées de fonctionnement de nœuds membres joignant un overlay p2p. Alors que la majorité des peers restent pour de courte durée et ont des capacités faibles, un petit pourcentage restent généralement pendant de longues périodes et ont relativement une meilleure capacité de stockage, puissance de traitement, et stabilité.

Cette hétérogénéité peut être exploitée dans la conception de grand système fiable, où les nœuds les plus stables peuvent former dynamiquement un overlay de niveau supérieur. Les autres nœuds avec une courte durée de vie peuvent se connectés en tant que sous-overlay des nœuds de niveaux supérieurs.

3. Architecture proposée: Concepts et Principe de fonctionnement

L'hétérogénéité entre les nœuds **[78]** est un sujet important qui doit être considéré. Dans les DOSNs étudiés à base du réseau P2P structuré, tous les nœuds sont traités de manière égale et il n'y a pas de différence entre eux. Cependant, dans un monde réel, les nœuds ont des capacités différentes, telle que la puissance de traitement différente, le stockage et la bande passante. En outre, la disponibilité de nœuds affecte aussi indirectement la disponibilité d'un système P2P. Par exemple, si un nœud a plus de disponibilité et une meilleure tolérance aux pannes dans le système, les autres nœuds peuvent utiliser son service beaucoup plus en réduisant le taux de churn. Ainsi, les nœuds avec des ressources plus importantes et caractéristiques désirables devraient être regroupés comme super-nœuds alors que le reste des nœuds sont classés comme des nœuds ordinaires.

Notre architecture est un overlay hiérarchique se compose de deux niveaux dans lequel les peers sont organisés en groupes[71] disjoints (*clusters*). L'architecture overlay hiérarchique offre plusieurs avantages importants par rapport à l'architecture overlay à base de DHT **[79]** :

1- Elle réduit le nombre moyen de sauts pour une requête de recherche. Moins de sauts par requête de recherche signifient moins d'overhead.
2- Moins d'overhead de la maintenance: maintenance de réseau recouvrant structuré implique la correction des états de routage des nœuds pour s'adapter à des événements dynamiques du nœud (*ex. connexion, déconnexion, la migration d'une clé, ou défaillance*). Etant donné que la structure hiérarchique divise les nœuds dans des overlays multiples, dont chacun est plus petit qu'un overlay plat, les messages de maintenance sont acheminés seulement dans un de ces petits overlays. Cela accélère la correction des états De routage tout en réduisant le nombre de messages de stabilisation traités par chaque nœud.
3- Isolement de churn: les changements de topologie dans un groupe en raison du churn, à savoir, des changements continus en raison des connexions (*joins*), déconnexions (l*eaves*), ou des échecs des nœuds , n'affectent pas l'overlay de niveau supérieur ou d'autres groupes. Ils seront traités en tant qu'événements locaux qui affectent un seul groupe, et les tables de routage en dehors du groupe ne sont pas affectées **[80]**.
4- Elle réduit la latence du réseau lorsque les peers dans le même groupe (*cluster*) sont proches topologiquement (*groupement par proximité physique*).

Parce que le nombre de groupes sera plus petit que le nombre total de peers, les requêtes nécessitent moins de sauts et la stabilité des super-peers de haut niveau contribuera aussi à réduire le délai de requête.

5- Elle facilite le déploiement à grande échelle en fournissant une autonomie administrative et une transparence tout en permettant à chaque groupe participant de choisir son propre protocole d'overlay. L'overlay peut utiliser divers algorithmes de lookup inter et intra-groupe simultanément.

Le nombre d'étapes dans une telle requête de recherche locale sera O(*log(M)*), où M est le nombre de peers dans le groupe.

Nous n'allons pas étudier de manière exhaustive tous les problèmes comme par exemple le problème de sécurité, mais nous allons plutôt porter notre attention sur le problème de

[71] Un groupe et un cluster réfère à la même chose

disponibilité des données qui est un défi majeur pour les DOSNs, tout en minimisant l'overhead produit par la migration fréquente des répliques des données.

Pour réduire la dégradation des performances constatée dans les DOSNs existants, nous proposons un overlay hiérarchique basé sur les tables de hachage distribuées à deux niveaux, nous avons choisis la DHT Chord pour chaque cluster de notre réseau social DOSN.

Dans cet overlay, le niveau supérieur se compose des nœuds stables avec plus de ressources tandis que le niveau inférieur contient des nœuds moins stables et des nœuds ordinaires avec des ressources limitées.

Les coûts de construction et de maintenance de l'overlay dans cette architecture sont également moins par rapport à des implémentations de DHT classique et donc, nous sommes tenus de mettre à jour les tables de raccourcis et les listes de successeurs de nœuds de niveau supérieur avec moins de fréquence.

3.1 Présentation de l'architecture

Notre architecture est composée de 2 niveaux hiérarchiques d'overlay structuré :

1- Les super-peers forment l'anneau de niveau supérieur (*l'anneau supérieur*) de la DHT hiérarchique, et se connectent les uns aux autres dans une structure en forme d'anneau (*Chord*), ils agissent comme des passerelles entre les niveaux supérieur et inférieur de cette architecture.

Dans notre architecture, l'anneau du niveau supérieur contient au maximum 256 nœuds (*super-peers*), qui sont des nœuds stables et puissants, et chaque super-peer gère un groupe de peers du niveau inférieur, est responsable de la propagation des requêtes pour les peers dans son groupe.

2- Le niveau inférieur est aussi un réseau de P2P structuré suivant Chord et comprenant des groupes de peers qui sont moins stables (*les participants de DOSN*).

Les peers dans chaque cluster sont physiquement et logiquement proches, ceci réduit considérablement et efficacement la latence de recherche.

Chaque peer [72] dans ce niveau appartient exactement à un cluster. Tous les clusters sont définis par un graphe orienté *(C, E)*, où *C = {C1, C2,. . . , CK}* représente l'ensemble de tous les clusters et *E* désigne l'ensemble des liens virtuels entre les clusters dans *C*. La figure 4.1 illustre l'architecture P2P hiérarchique proposée.

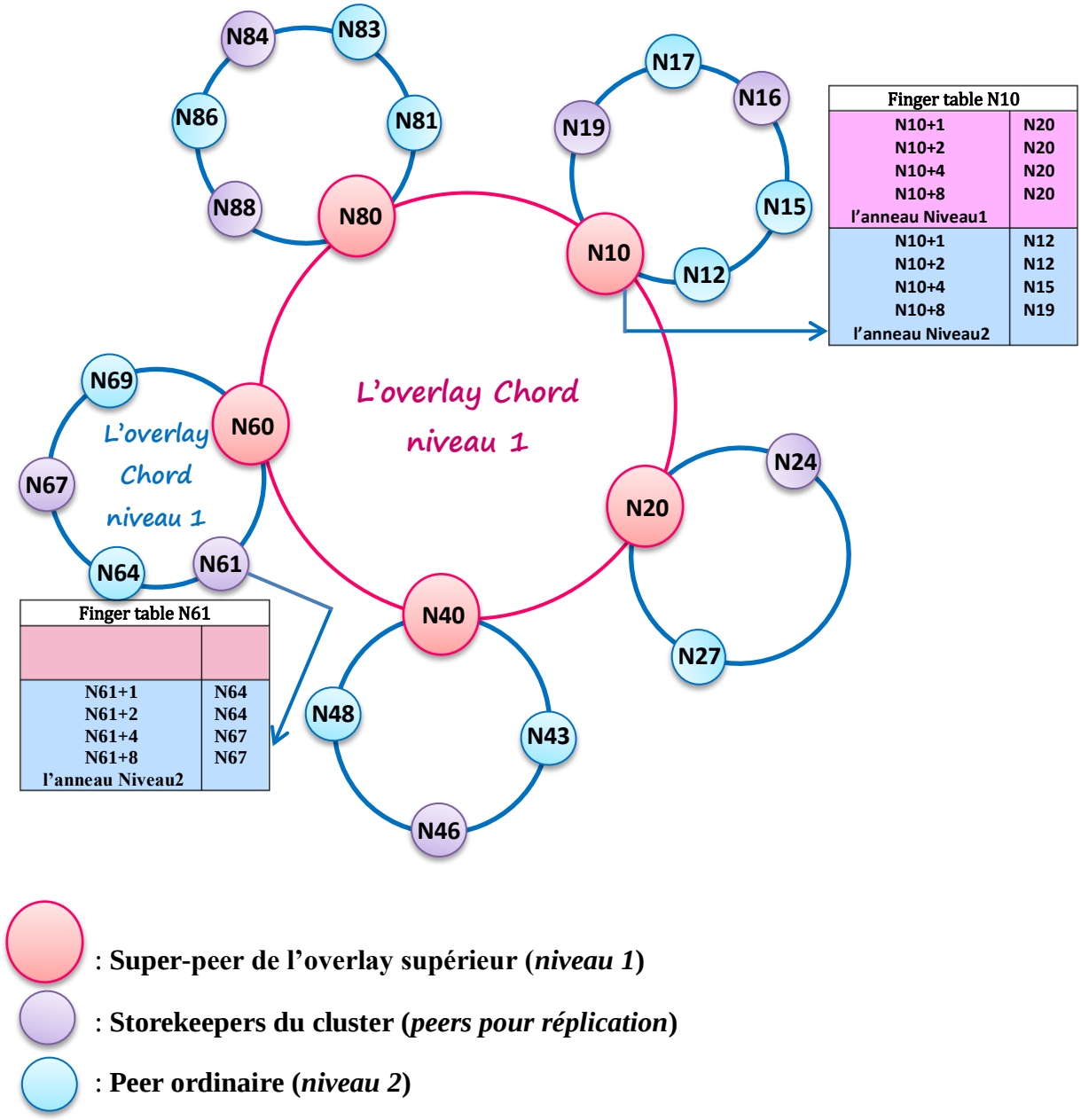


Figure 4. 1: L'architecture hiérarchique proposée (DOSN- 2 Tiers)

[72] Un nœud et un peer réfère à la même chose.

Création des identifiants des nœuds

Afin de soutenir l'architecture de DOSN hiérarchique, nous définissons un espace hiérarchique d'identificateurs qui permet d'identifier chaque peer d'une manière unique. L'identifiant d'un peer dans notre architecture s'obtient par le hachage de son adresse email. Pour pouvoir identifier les clusters (*les super-peers*) à partir de l'adresse email, nous procédons à paramétrer la fonction de hachage de telle sorte à obtenir des collisions qui nous permettent à définir les overlays de *niveau 1* et *niveau 2* respectivement en calculant la paire d'identifiants $(id_{cluster}, id_{peer})$. $Hash(email) = (id_{cluster}, id_{peer})$

L'identifiant d'un peer $p, \boldsymbol{ID} = \begin{cases} id_{cluster}\ ,\ si\ le\ peer\ p\ est\ dans\ le\ niveau\ 1 \\ id_{peer}\ ,\ si\ le\ peer\ p\ est\ dans\ le\ niveau\ 2 \end{cases}$

L'*id_cluster* obtenu après le hachage $\in [0, value]$, c'est avec cette condition que nous paramétrons la fonction de hachage pour avoir des collisions et par conséquent avoir des valeurs identiques de *id_cluster* , ce qui permet de construire les clusters de niveau 2.

Pour le cas de notre architecture, la valeur de $value = 255$, qui représente la taille de l'overlay du niveau 1, nous utilisons la fonction de hachage SHA-1 **[15]** pour créer les identifiants :

SHA-1 (*Secure Hash Algorithm*) est une fonction de hachage cryptographique qui produit une chaîne de 160 bits (*appelé condensé d'une chaîne*) à partir d'un message. Elle est suggérée pour être employé par l'équipe de Chord.

Le SHA-1 de l'email d'un peer produit l'*ID* d'un peer qui est une chaine binaire de *n-bits*.il est divisé en deux parties d'identifiants:

- L'identifiant *id_cluster* est une partie de *m-bits* les plus à droite du résultat de la fonction de hachage qui identifie le cluster et il est commun pour tous les peers dans un cluster, autrement dit, tous les nœuds dans le même cluster partagent les *m-bits* les plus à droite de leurs *ID.*
- L'identifiant *id_peer* est la partie de *$(n-m)$ bits*, elle identifie le peer à l'intérieur du cluster.
- L'identifiant *ID* d'un peer dans le réseau overlay est la concaténation de *m-bits* de l'*id_cluster* avec *(n-m) bits* de id_{peer} du résultat de la fonction de hachage de l'email

(Par exemple, si SHA-1 (*miama@univ-bejaia*) = 198104 et *id_cluster* = 198 et id_{peer} = 104 alors *ID*=198104).

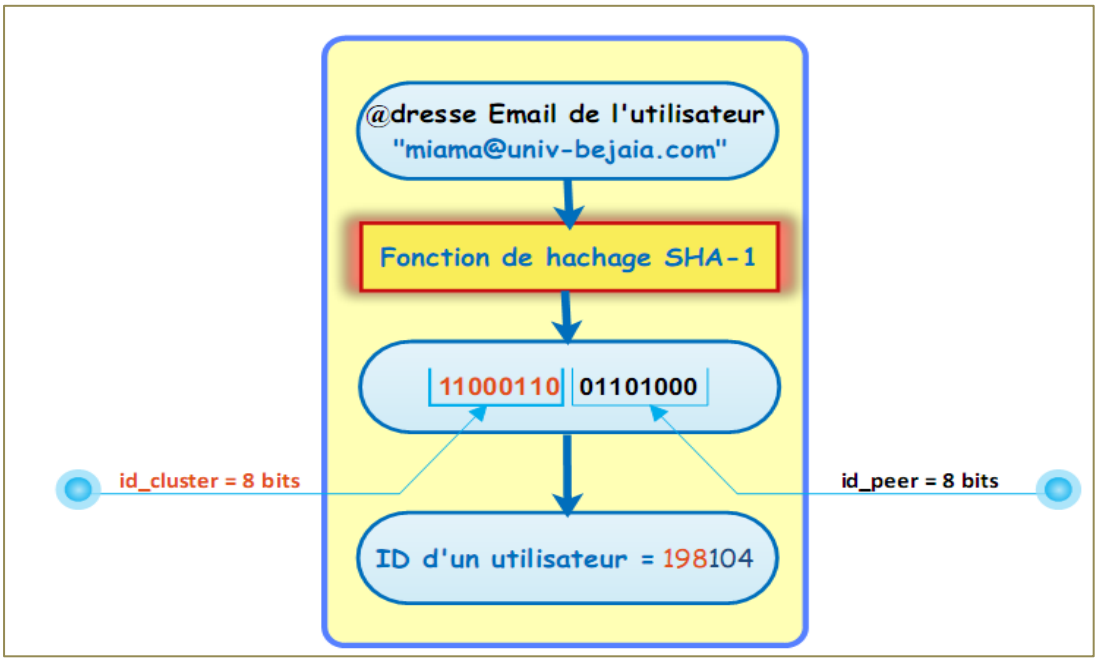


Figure 4. 2: : Exemple de création d'un Identifiant ID

Comme notre architecture est basée sur le protocole Chord, les structures de données et les variables maintenues dans chaque nœud sont les mêmes qui sont utilisées dans Chord:

- **Liste de successeur (*successor list*)**: une liste de quelques nœuds successeurs est maintenue à chaque nœud, et contient les ***s*** premiers nœuds successeurs d'un nœud.
- **Le nœud prédécesseur**: l'identifiant de nœud prédécesseur.
- **Table de raccourcis (*Finger table*)** : table de routage qui fournit des informations sur 2^i nœuds successeurs, qui lui sert pour la recherche de ressources.
- **Liste des amis :** une liste qui contient des informations concernant les amis de l'utilisateur. Elle est digitalement signée et encryptée.

A la différence des autres nœuds, les super-peers de l'overlay supérieur doivent maintenir deux tables de raccourcis (*pour la table du niveau supérieur, elle va contenir au maximum log(256) entrées*), deux listes de successeur, deux pointeurs pour les nœuds prédécesseurs pour les deux niveaux supérieur et inférieur respectivement **[77]**.

3.2 Connexion et Routage

3.2.1 Routage d'une requête de recherche

Le routage dans notre architecture est **un routage intra-cluster** et **un routage inter-cluster** :

- Dans **le routage intra-cluster** où la clé recherchée *k* se trouve localement dans le même cluster, le processus de recherche sera le même utilisé dans le protocole Chord.
- Dans **le routage inter-cluster** où la clé *k* se trouve dans un autre cluster, la requête de recherche est relayée par les super-peers sur la DHT. Afin de localiser le cluster responsable de la clé k, le peer qui cherche la clé k contacte le super-peer dans son cluster, le super-peer par la suite achemine la requête sur la DHT au super-peer du cluster de destination, et à la fin la requête est transmise au peer de destination.

Le processus de recherche s'effectue en trois étapes comme suit :

1- Tout d'abord, une requête de recherche pour la clé *k* est acheminée vers le super-peer du cluster initiateur (*les peers du cluster connaissent leur super-peer*), si la clé *k* est trouvée localement, la recherche est terminée, sinon poursuivre l'étape 2 et 3.

2- Deuxièmement, en utilisant l'algorithme de recherche Chord, la requête de recherche est acheminée vers le super-peer du cluster dont l'identifiant cluster est $id_{cluster}$ = successeur (*k*) dans l'overlay de niveau supérieur (*niveau 1*), afin de localiser le cluster responsable de *k*.

3- En troisième lieu, la requête de recherche peut en outre être transmise à l'un des nœuds de deuxième niveau dans le cluster responsable de k.

Si le nœud n'est pas trouvé dans le statut online (il a quitté le DOSN), chercher une de ses répliques de son profil stockées dans les nœuds de réplication de son cluster avec le **statut offline**.

Comme le montre la Figure 4.3, une requête de recherche de la clé (*198|104*), initié par le nœud de deuxième niveau **A**, est transmise à son super-peer 70|190, cette clé n'appartient pas au cluster 70 (*l'id_cluster* de la requête est $198 \neq 70$), donc le super-peer 70|190 dirige cette requête vers l'overlay de niveau 1 (*étape 1*). Dans l'overlay de niveau 1, la requête de recherche est acheminée vers le super-peer 198|60 du cluster 198 en utilisant Chord (*étape 2*).

Enfin, le super-peer 198|60 peut transmettre la requête au nœud 198|104 de son cluster en procédant par l'algorithme Chord (*étape 3*).

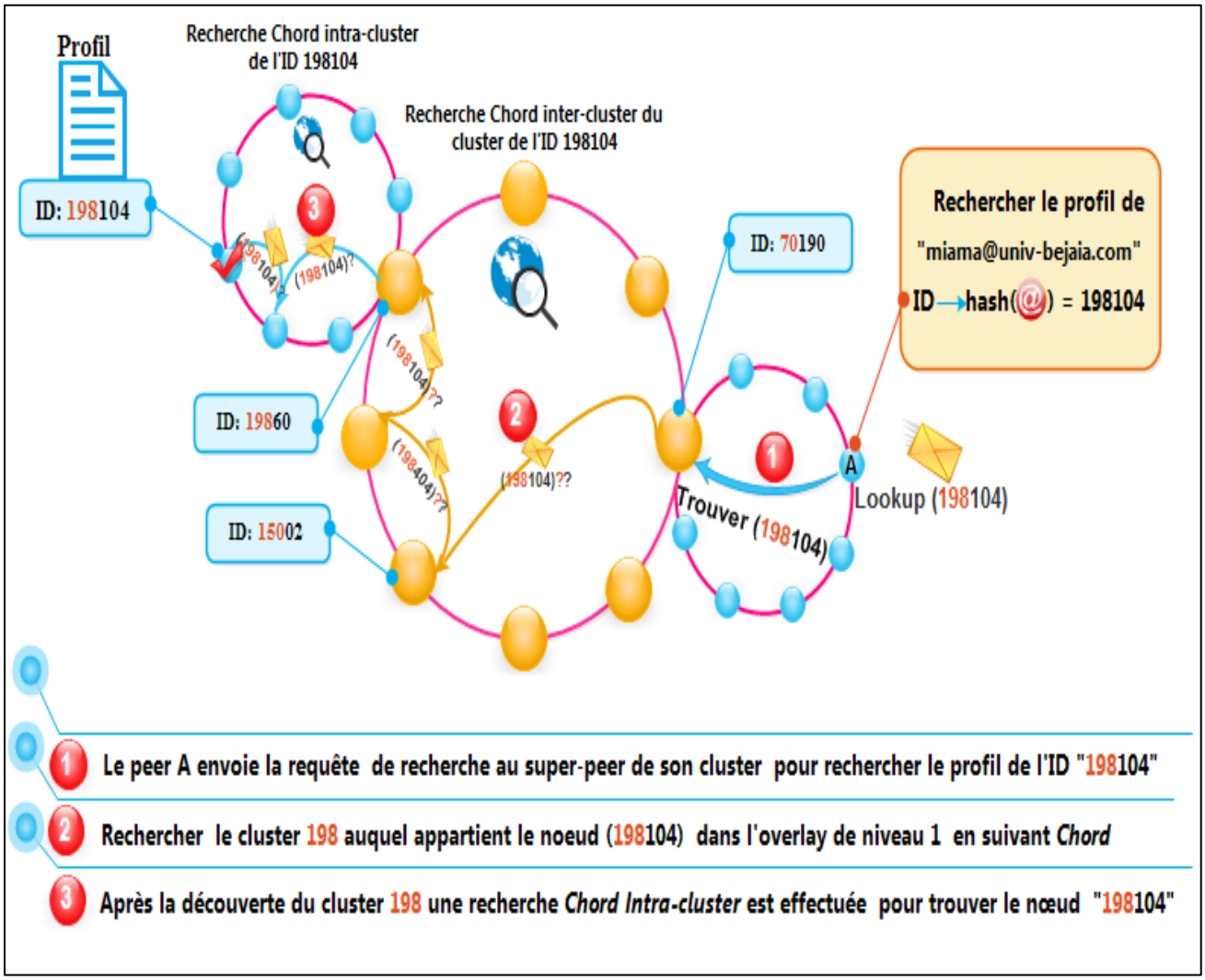


Figure 4. 3: : Exemple de processus de recherche

En se basant sur l'algorithme de recherche du protocole Chord (*voir chapitre 1*) , notre algorithme de recherche est détaillé dans le pseudo code suivant :

Algorithme 1 Pseudo code de processus de recherche *lookup* $k(c_{n1}c_{n2})$ dans 2-niveaux-DHT

Begin

1: rechercher *la clé* ***k*** $(c_{n1}c_{n2})$ auprès de super-peer$_{Query_k}$ dans le même cluster ;

// Super-peer$_{Query_k}$ est le super-peer du cluster d'où la requête de recherche de la clé *k* est sortie.

2: ***If*** **(** **k.**$c_{n1==}$ super-peer$_{Query_k}$. $id_{cluster}$) ***then*** // *la clé* ***k*** se trouve au même cluster

3: super-peer.***find_successor_niveau-2***(*k*) ; // rechercher le successeur de *la clé* ***k*** en utilisant ***find_successor ( )*** de l'algorithme Chord.

4: ***Else*** // *la clé* ***k*** ∉ au cluster de , la requête est dirigée par le super-peer au niveau 1.

5: super-peer$_{Query_k}$. ***find_successor_2niveaux_DHT*** (***k***) ; // localiser le cluster où ∈ la clé ***k*** dans l'overlay de niveau 1 et le successeur de la clé ***k*** dans son cluster

6: ***EndIf***

7: Super-peer$_{Query_k}$.***find_successor_2niveaux_DHT(k);***

8: ***If*** **(** ***k***.c_{n1} ∈ (super- peer$_{Query_k}$, successor-super-peer$_{Query_k}$]) ***then*** //

9: ***return*** successor-super- peer$_{Query_k}$; //le super-peer du cluster où ∈ la clé **k** est trouvé

10: Super-peer $_{(k)}$.***find_successor_niveau-2***($k.c_{n2}$) ; //en utilisant ***find_successor()*** du Chord ,rechercher le successeur de *la clé* ***k*** par son super-peer de son cluster de l'overlay niveau 2

11: ***Else*** //transmettre la requête de recherche autour de **l'overlay niveau-1**

12: super-peer$_{lookup}$ **=** ***closest_preceding_node*** **(**$k.c_{n1}$**);**

// la procédure ***closest_preceding_node*** de Chord choisit de la table de raccourcis de Super-peer$_{Query_k}$ le peer avec le plus grand id qui précède $k.c_{n1}$.

13: ***return*** super-peer$_{lookup}$.***find_successor_niveau_1***(*k*.c_{n1}**);** // poursuivre la recherche de super-peer$_{(k)}$ par le super-peer$_{lookup}$ de **l'overlay niveau_1**

14: Super-peer$_{(k)}$.***find_successor_niveau_2***($k.c_{n2}$); //en utilisant ***find_successor()*** du Chord ,rechercher le successeur de la clé ***k*** dans son cluster de **l'overlay niveau 2.**

15: ***EndIf***

16: ***If*** l'utilisateur recherché n'existe pas avec le ***statut_online*** ***then***

17: rechercher une de ses répliques de son profil avec le ***statut_offline*** ;

18: ***EndIf***

19: ***return*** **(***le résultat de recherche*) ;

20:*End*

Complexité et Coût de recherche

Pour une bonne d'illustration, nous considérons les notations suivantes :

Notation	Désignation
$\boldsymbol{N}$	nombre total des utilisateurs
$\boldsymbol{S}$	nombre de super-peers
$\frac{\boldsymbol{N}}{\boldsymbol{S}}$	nombre de peers dans un cluster comprenant le super-peer (*tous les clusters ont la même taille*).
$\boldsymbol{\gamma}$	la probabilité que le peer demandeur et le peer de destination pour une requête sont dans le même cluster ; c'est-à-dire la probabilité de trouver une ressource dans le même cluster. Plus la valeur de γ est petite, plus la proximité entre les peers dans le même cluster est moins probable.
(1 - γ)	la probabilité de la communication inter-cluster

Pour le cas de notre architecture, la probabilité $\boldsymbol{\gamma}$ sera choisi élevée ; les peers dans un cluster sont physiquement et logiquement proches, et la chance d'avoir des amis dans le même cluster est considérable[73].

La complexité de notre algorithme de recherche (lookup) est comme suit :

$$O((log(S)) + (log(\frac{N}{S}))$$

Le coût de recherche (*nombre de sauts*) pour notre architecture est défini par la formule suivante :

$$\textbf{Coût de recherche} = \boldsymbol{\gamma}.\ {}^{\mathbf{1}}/_{\mathbf{2}}\ \mathbf{log}\left(\frac{\boldsymbol{N}}{\boldsymbol{S}}\right) + (\mathbf{1}-\boldsymbol{\gamma})({}^{\mathbf{1}}/_{\mathbf{2}}\ \mathbf{log}\left(\frac{\boldsymbol{N}}{\boldsymbol{S}}\right) + {}^{\mathbf{1}}/_{\mathbf{2}}\ \mathbf{log}(\boldsymbol{S}))$$

[73] Une analyse des réseaux sociaux de plus de 500.000 blogueurs LiveJournal a révélé que la proximité géographique a augmenté de manière significative la probabilité de formation de l'amitié. Dans cette communauté, les deux tiers des utilisateurs un moyen amis étaient géographiquement proches. Une autre analyse des données à partir de sites populaires de réseau social tels que Facebook et MySpace ont constaté que les personnes listées comme les meilleurs amis dans les profils en ligne ont tendance à résider géographiquement proches. En outre, la fréquence et la durée de la communication en ligne augmentent lorsque la distance géographique entre les individus diminue. Collectivement, ces études fournissent des preuves solides et cohérentes que la proximité géographique joue un rôle important dans le développement des relations **[88]**.

3.2.2 Connexion à l'overlay (*joining*)

Les nœuds qui veulent participer dans un réseau overlay ; initialement doivent trouver au moins un nœud qui fait partie de ce réseau, et joindre le réseau à travers ce nœud (*l'amorçage ou le bootsrapping*) .

Afin de réduire la complexité de notre architecture, pour le bootstrapping des nœuds, nous choisissons le mécanisme de serveurs overlay statiques (*Static Overlay Nodes-Bootstrapping servers*).*(voir le Bootstrapping dans le chapitre 1).*

Tous les nœuds qui arrivent au système rejoignent le niveau inférieur d'abord(*niveau 2*), ils ne peuvent pas être promu à devenir super-peers directement.

Pour qu'un nouveau peer p_{new} joigne l'overlay, il doit d'abord s'authentifier chez un serveur de bootstrapping bien connu. Ce dernier garde une liste actualisée des peers présents dans le réseau.

Le peer p_{new}, joignant l'overlay, obtient de cette liste de quelques adresses IP de nœuds qui sont déjà dans le réseau, il contacte l'un de ces nœuds p_{exist} et demande à p_{exist} de trouver le point d'attachement pour p_{new} dans l'overlay de niveau 2 ; c'est-à-dire trouver son successeur. Le peer p_{exist} envoie l'email haché (*l'ID*) de p_{new} au super-peer de son cluster et ce dernier examine l'*ID* de p_{new} et découvre si p_{new} doit être attaché dans l'overlay de niveau 2 de p_{exist} ou dans un autre overlay de niveau 2.

Si p_{new} n'appartient pas à l'overlay de p_{exist} , le super-peer cherche dans l'overlay de niveau 1 pour localiser le cluster où appartient p_{new} en utilisant la procédure ***find_successor( )*** du Chord. après avoir localiser son cluster, p_{new} sera attaché devant son successeur de l'overlay de niveau 2 en utilisant la procédure ***find_successor( )*** du Chord (*pour plus de détails sur Chord, voir chapitre 1*).

Dans le cas où le cluster de p_{new} n'existe pas, p_{new} va être insérer dans l'overlay de niveau 1 comme un super-peer d'un nouveau cluster.

Algorithme 2 Pseudo code Join dans 2-niveaux-DHT

Begin

Initialement le peer p_{new} fait une connexion à un serveur de Bootstraping pour s'authentifier
Le serveur lui envoie une liste des **@ ip** des nœuds déjà connectés au réseau
// Le peer p_{new} va joindre l'overlay à partir de p_{exist} sélectionné de cette liste

1: p_{new}**.join (**p_{exist}**)**

2: p_{exist} envoie l'*ID* de p_{new} au super-peer de son cluster

3: **If (**$p_{new}.id_{cluster==}$ super-peer$_{(pexist)}.\ id_{cluster}$**) then** // p_{new} et p_{exist} ∈ au même cluster

4: Successor = p_{exist}***.find_successor_niveau-2***(p_{new}) ; // rechercher le successeur de p_{new}

5: insérer p_{new} avant son successeur dans l'overlay de niveau 2

6: **Else** // le peer $p_{new} \notin$ au cluster de p_{exist}

7: Successor = super-peer$_{(pexist)}$***.find_successor_2niveaux_DHT***(p_{new}); // localiser le cluster où $\in p_{new}$ dans l'overlay de niveau 1 par le super-peer de p_{exist} et le successeur de p_{new} dans son cluster par le super-peer de p_{new} dans l'overlay de niveau 2 .

8: super-peer$_{(pexist)}$**..*find_successor_niveau-1***($p_{new}.id_{cluster}$) ; //rechercher le bon cluster auquel ∈ p_{new} en utilisant Chord**.**

9: super-peer$_{(pnew)}$***.find_successor_niveau-2***($p_{new}.id_{peer}$) ; // en utilisant **find_successor()** du Chord ,rechercher le successeur de p_{new} dans son cluster de **l'overlay niveau 2.**

10 *:* *EndIf*

11:*End*

3.2.3 Départ d'un nœud (*Leave*)

L'opération de départ des peers de leur clusters est ordinaire, elle ne nécessite aucun traitement particulier, lorsque un peer veut quitter l'overlay, la procédure de stabilisation du Chord sera exécuter pour mettre à jour les tables des fingers et les pointeurs.
Concernant le départ d'un super-peer qui n'est pas une opération fréquente (*l'overlay niveau 1 est plus stable*), une ré-sélection d'un nouveau super-peer aura lieu ; le peer le plus stable dans le cluster en termes de performance et de disponibilité sera choisi.

3.3 Réplication des données

Chaque nœud (*ou utilisateur*) dans DOSN est représenté par un profil, qui est l'ensemble des données de l'utilisateur. Elles peuvent être publiques, protégées par un accès limité à un sous-ensemble d'amis, ou privée et inaccessible à quiconque **[66]**.

Pour assurer la disponibilité des profils de leurs utilisateurs (*contenu public*), les architectures DECENT et Cachet (*les autres architectures étudiées PeerSon et SuperNova ne garantissent pas une meilleure disponibilité*) utilisent la DHT (*pour le stockage des répliquas des profils utilisateurs*).

Cachet met les réplicas des profils dans des nœuds choisis aléatoirement, tandis que DECENT utilise l'ensemble de voisins du nœud responsable du profil (*utilise Neighbor Replication*).

Dans notre architecture, pour garantir une haute disponibilité des profils utilisateur, Nous utilisons la DHT comme Cachet et DECENT, mais nous proposons une stratégie de réplication différente à celle utilisée dans ces architectures.

Nous allons maintenir les réplicas de données utilisateur dans les *k* nœuds les plus disponibles de l'ensemble de successeurs du nœud avec prise en compte de l'aspect d'amitié en stockant les données dans les nœuds amis (*avoir un degré de sécurité*) ,ce qui permet de minimiser le cout de maintenance (*l'overhead de communication généré lors de départ des nœuds où les répliques seront migrées vers les autres nœuds de la DHT pour restaurer le degré de réplication*) par rapport aux autres architectures (*Cachet et DECENT*).

Notre stratégie de réplication vise donc à obtenir une meilleure disponibilité avec un minimum d'overhead sans négliger l'aspect de sécurité.

3.3.1 Stratégie de réplication

Notre stratégie sert à choisir les ***k*** nœuds les plus disponibles (***les nœuds storekeepers***) pour le placement des réplicas, ce qui permet d'améliorer la résistance au churn. La stratégie de réplication « Plus disponibles » (***Most-available replication strategy***) **[81]** est une extension de la stratégie standard dont le but est de réduire le nombre de migrations d'objets en choisissant les nœuds hautement disponibles pour composer l'ensemble de nœuds de réplicas (*replicas sets*).

La réplication fonctionne comme suit: a tout moment, un objet[74] est répliqué sur les ***k*** nœuds les plus disponibles choisis parmi l'ensemble des premiers nœuds ***S*** qui représente la liste de successeurs avec les plus grands identificateurs que l'identificateur de l'objet.

Toutes les configurations de Chord utilisent un facteur de réplication de 7 (*chaque ressource est répliquée dans le propriétaire et dans 6 de ses successeurs*) **[82]**.

Le nombre ***S*** des successeurs est log(M), avec M : le nombre de peers dans un cluster.

La valeur de **k nœuds de réplicas** $= \begin{cases} S, & si \quad S \leq 6 \\ 6, & sinon \end{cases}$

Donc, le choix de la valeur de **k** est corrélé avec le nombre de nœuds dans le cluster. La figure 4.4 représente la stratégie de réplication utilisée par la DHT de chaque cluster.

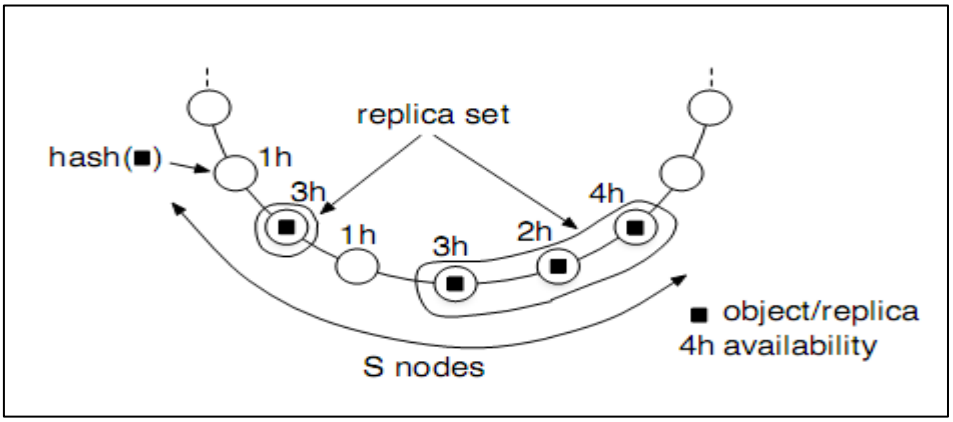


Figure 4. 4: : Most Available Replication Strategy (*k=4*) [81]

Cette stratégie requiert d'être en mesure de prédire la disponibilité des nœuds. Dans notre architecture, nous supposons qu'il y a un mécanisme de mise à jour qui assure la cohérence des réplicas. Comme les amis sont intéressés par le contenu d'un utilisateur, nous supposons qu'ils sont prêts à le stocker comme un effet de leur intérêt.

Nous supposons également que le temps de transfert des données d'un utilisateur est négligeable. Typiquement le contenu du profil utilisateur de DOSN est de petite taille. En outre, les objets avec une grande taille telle que les vidéos peuvent être téléchargées à partir du Cloud[75] (*par exemple, YouTube*) et seul le lien vers cet objet est partagé par l'utilisateur de

[74] Un objet dans la réplication représente les données des utilisateurs de DOSN (leurs profils).
[75] Le cloud : C'est un ensemble de services qui permet à un utilisateur, de transférer ses stockages et ses traitements informatiques, depuis un ou plusieurs de ses serveurs locaux, vers d'autres serveurs extérieurs.
Source : http://www.claranet.fr/cloud.htm

DOSN. Les applications des réseaux sociaux ne sont pas similaires à celles des partages de fichiers.

3.3.2 Prédiction de disponibilité

Plusieurs heuristiques efficaces pour prédire la disponibilité des nœuds ont été proposées dans **[83] [84]**. Un des types de prédiction utilise la prédiction linéaire, qui est une technique statistique commune pour prédire les séries chronologiques. Un autre type de prédiction où les auteurs émettent l'hypothèse que la disponibilité actuelle devrait être un bon prédicteur de la disponibilité future **[85]**. Dans **[86]** ,K. KIM et al. supposent que la disponibilité d'un nœud est la valeur de prédiction de combien de temps un nœud est vivant après qu'il joigne le système P2P.

Pour choisir les nœuds de réplication de notre architecture, nous choisissons d'utiliser le mécanisme de prédiction de disponibilité utilisé dans **[86]**. Nous utilisons le temps moyen de fonctionnement avant panne ***MTTF*** (***Mean Time To Failure***) et le temps moyen de reprise ***MTTR*** (***Mean Time To Recover***) pour estimer la disponibilité du nœud. ***MTTF*** est la valeur moyenne de combien de temps un nœud est vivant après qu'il se joigne et ***MTTR*** est la valeur moyenne de combien de temps un nœud est en sommeil après qu'il parte.

Le TTF et TTR sont obtenus en utilisant le dernier temps de connexion (*the last join time*), la dernière fois de départ (*the last leave time*) et le temps courant de connexion (*the current join time*). À la différence de TTR, nous mettons à jour périodiquement le TTF en employant le temps courant et le temps courant de connexion. Le MTTF et MTTR sont obtenus par la moyenne pesée de TTF et de TTR. Dans ce cas-là, le α et le β sont mis à 0.9.

La figure 4.5 illustre la prédiction de disponibilité des nœuds :

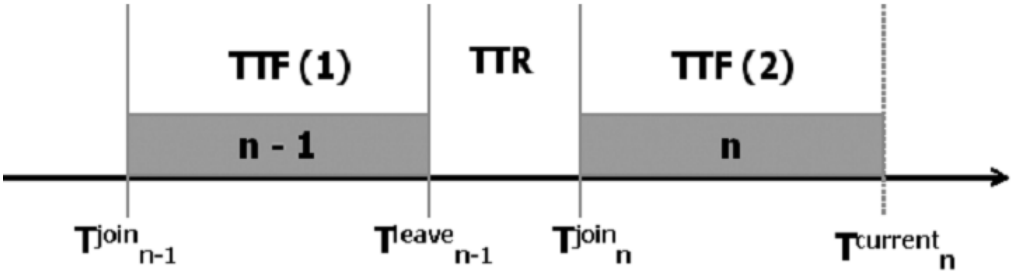


Figure 4. 5: Calcul de TTF et TTR [86]

Calcul de Mean Time To Failure (*MTTF*)

- **Time to failure (*TTF*)**

1) La mesure à l'arrivé du nœud (*At joining measurement*):

$$\mathbf{TTFn} = \boldsymbol{T}\text{leave}_{\boldsymbol{n-1}} - \boldsymbol{T}\text{join}_{\boldsymbol{n-1}}$$

2) La mesure périodique (*At periodic measurement*):

$$\mathbf{TTFn} = \boldsymbol{T}\text{current}_{\boldsymbol{n}} - \boldsymbol{T}\text{join}_{\boldsymbol{n}}$$

- $\boldsymbol{MTTF_n} = \alpha * \text{TTF}_\text{n} + (1-\alpha) * \text{MTTF}_{\text{n}-1}, (0 < \alpha < 1)$

Calcul de Mean Time To Recover (*MTTR*)

- **Time to Recover (*TTR*)**

$$\boldsymbol{TTR_n} = \boldsymbol{T}\text{join}_{\boldsymbol{n}} - \boldsymbol{T}\text{leave}_{\boldsymbol{n-1}}$$

- $\boldsymbol{MTTR_n} = \beta * \text{TTR}_\text{n} + (1-\beta) * \text{MTTR}_{\text{n}-1}, (0 < \beta < 1)$

La disponibilité (*Node Availibility*) du nœud est calculée avec l'équation (*1*) :

Dispo = MTTF / (MTTF + MTTR)	…… (*1*)

La disponibilité du nœud est calculée par chaque nœud , et il doit l'annoncé à tous les membres de l'ensemble cohérent (*liste des successeurs*).

Pour détecter une défaillance d'un nœud, un nœud envoie périodiquement un message **ping** à tous les membres de l'ensemble cohérent. La disponibilité du nœud est portée sur ce message. Chaque nœud maintient les disponibilités des nœuds de l'ensemble des nœuds de la liste de successeurs.

Comme la disponibilité est notre premier souci, nous choisissons le coefficient $\varphi = 1$ Soit A le nœud qui veut répliquer son profil, et sa liste de successeurs :

Succ-list = {B,C,D,E,F,G}

Après avoir calculé la disponibilité des nœuds B, C, D, E, F, G de la liste des successeurs, le nœud A va chercher dans sa liste des amis, s'il y a parmi ces nœuds des nœuds amis à lui. Deux cas seront apparus :

1- Nœud A trouve des amis : afin de choisir l'ensemble de réplication de k nœuds (*replicas_set*) en prenant compte l'aspect de sécurité (*amitié*), nous allons appliquer la formule (*2*) où le choix est fait selon deux (*2*) facteurs : la disponibilité et l'amitié, en utilisant la formule de choix suivante :

$$\boldsymbol{Choice\_Replicas\_Set\ CRS = \varphi * Dispo + \lambda * f} \quad \ldots\ldots\ldots\ldots\ldots\ldots(2)$$

Avec : ***Dispo :*** est calculée selon l'équation (1)

$\boldsymbol{\varphi}$: est une valeur empirique de combien on donne importance à la disponibilité

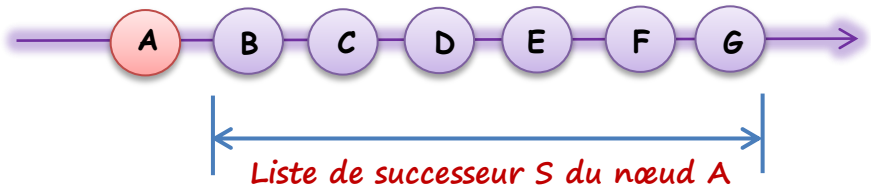
$\boldsymbol{\lambda}$: est une valeur empirique de combien on donne importance à l'amitié (*le poids d'amitié*).

$$\boldsymbol{f} \text{ est une variable } = \begin{cases} f = 1\,, \ si\ le\ noeud\ est\ un\ ami\ proche \\ f = 0.5, \ si\ le\ noeud\ est\ un\ ami\ ordinaire \\ f = 0, \ si\ le\ noeud\ n'estpas\ un\ ami \end{cases}$$

Les nœuds seront ordonnés dans un vecteur selon un ordre décroissant de leur valeur ***CRS*** calculée, les répliques sont alors placées dans les **k** premiers nœuds ayant la plus grande valeur de ***CRS***.

2- Nœud A ne trouve pas des amis : les nœuds seront ordonnés dans un vecteur selon un ordre décroissant de leur disponibilité ***Dispo***, les répliques sont alors placées dans les **k** premiers nœuds les plus disponibles.

Exemple :



Le nœud A veut répliquer son profil, le vecteur de disponibilité de A est défini comme suit

Nœud	**B**	**C**	**D**	**E**	**F**	**G**
Dispo	**0.2**	**0.6**	**0.7**	**0.4**	**0.9**	**0.2**

Nous supposons que les nœuds amis de A sont : B, D, et E, et ce sont des amis proches, c'est-à-dire $f = 1$.

Nous mettons *k*=3 (*replicas_set=3 nœuds*)

Nous calculons le CRS en utilisant la formule (2) de ces nœuds pour trois cas différents :

Cas 1 : $\lambda = 0.5$ (*l'amitié est pondérée*)

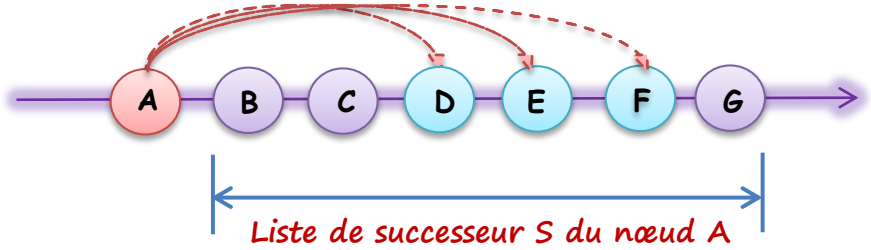
En appliquant la formule (2) sur les nœuds du vecteur, nous obtenons les valeurs suivantes :

Nœud	B	C	D	E	F	G
Dispo	0.2	0.6	0.7	0.4	0.9	0.2
CRS	0.7	0.6	1.2	0.9	0.9	0.2

Le vecteur ordonné sera :

Nœud	D	E	F	B	C	G
CRS	1.2	0.9	0.9	0.7	0.6	0.2

Les k nœuds choisis seront : D, E, F.



Cas 2 : $\lambda = 0.1$ (*l'amitié n'est pas favorisée*)

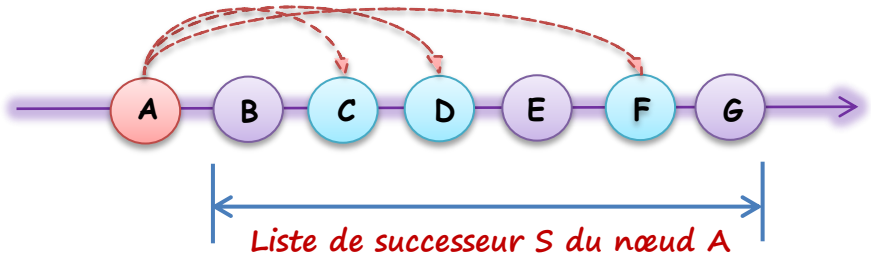
Dans ce cas-là , nous favorisons la disponibilité, les valeurs de CRS seront :

Nœud	B	C	D	E	F	G
Dispo	0.2	0.6	0.7	0.4	0.9	0.2
CRS	0.3	0.6	0.8	0.5	0.9	0.2

Le vecteur ordonné sera :

Nœud	F	D	C	E	B	G
CRS	0.9	0.8	0.6	0.5	0.3	0.2

Les k nœuds choisis seront : F, D, C, c.à.d. deux nœuds ayant une bonne disponibilité plus le nœud D qui est un nœud ami et disponible.



Cas 3 : $\lambda = 0.8$ (*l'amitié est favorisée*)

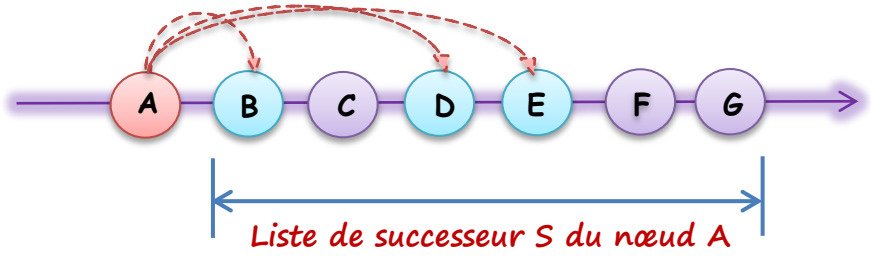
Dans ce cas-là, nous favorisons l'aspect d'amitié , les valeurs de CRS seront :

Nœud	B	C	D	E	F	G
Dispo	0.2	0.6	0.7	0.4	0.9	0.2
CRS	1	0.6	1.5	1.2	0.9	0.2

Le vecteur ordonné sera :

Nœud	D	E	B	F	C	G
CRS	1.5	1.2	1	0.9	0.6	0.2

Les k nœuds choisis seront : D, E, B, c.à.d. les nœuds amis.



Dans cet exemple, nous remarquons que l'ensemble de *k* nœuds de réplication se change selon la valeur de λ. Le choix de cette valeur est fortement couplé à la nature de l'application du réseau social, dont la mesure où la disponibilité du contenu peut être plus importante que la sécurité pour des types d'application.

Algorithme 2 : Réplication des profils

Begin

For (*chaque cluster*) **do**

For (*chaque nœud de DHT*) **do**

Calculer la disponibilité ***Dispo*** par la formule **MTTF / (MTTF + MTTR)**

Chercher dans Succ-list des nœuds amis

Calculer leur ***CRS*** en utilisant la formule *(2)*

Construire un vecteur de ***CRS*** de l'ensemble de nœuds de ***successor-list***

Ordonner le vecteur ***CRS*** selon l'ordre décroissant

Choisir du vecteur ***CRS*** les k-premiers nœuds avec la plus grande valeur de ***CRS***

End for ;

End for ;

For (i=1 to k) **do**

Peer i= R //placer les répliques dans les k peer disponible

End for ;

End.

4. Conclusion

Dans ce chapitre, nous avons commencé par motiver la nécessité de proposer une nouvelle architecture P2P pour un DOSN. Nous avons ensuite présenté en détail notre architecture proposée en se basant sur la description des principaux concepts de cette architecture.

Le prochain chapitre sera consacré à la simulation et l'étude du comportement de l'architecture proposée en vue de sa validation.

Chapitre 5

Simulation et Analyse des Performances

1. Introduction

Afin d'évaluer les performances de notre architecture hiérarchique 2 tiers (*DOSN-2 Tiers*), nous allons simuler analyser et son fonctionnement .

Dans ce chapitre, nous présentons quelques évaluations de performances de notre architecture hiérarchique à travers des résultats analytiques et des simulations à l'aide d'un simulateur développé par le langage JAVA, cela en le comparant avec quelques architectures DOSNs étudiées. Nous présentons en premier lieu, les métriques de performances mesurées, ensuite nous présentons les résultats obtenus.

2. Simulation et analyse des performances

Nous avons simulé la solution proposée par JAVA. Cette décision a été prise après l'étude des simulateurs de réseaux P2P existants (*Oversim ,OMNet++ , Peersim*…) , qui étaient pour la plupart beaucoup trop lourds et compliqués et peu adaptés à nos besoins.

La plupart de ces simulateurs sont développés hors de la nécessité de répondre à un besoin particulier. En général aucun des simulateurs donne une plate-forme commune pour concevoir et tester toute structure de p2p. En particulier, aucun du simulateur disponible fournit une classification des nœuds en considérant l'hétérogénéité qui est inévitable dans tout réseau informatique. Il est donc difficile à mettre en œuvre notre architecture hiérarchique en utilisant ces simulateurs,

Nous évaluons les performances de notre architecture Chord à 2 niveaux, puis nous comparons ses résultats avec d'autres solutions DOSNs existantes proposées (*DECENT, Cachet, Safebook*).

2.1 Mesures d'évaluation

Afin d'évaluer le comportement de notre architecture, plusieurs mesures ont été définies. Ces mesures ont pour but d'évaluer les différents aspects de fonctionnement de l'architecture.

Nous avons défini les métriques de performance suivantes :

- ✓ **Nombre de sauts** (*coût de recherche*) : qui représente la distance entre le nœud source qui a initié la recherche et le nœud destination (*l'emplacement de la donnée recherchée*), mesurée par le nombre de nœuds intermédiaires qui doit être parcouru entre la source et le nœud de destination.
- ✓ **Coût des structures de données :** la taille de la table de raccourcis maintenu par chaque nœud dans le réseau**.**
- ✓ **L'overhead de réplication (***les messages générés lors de migration des répliques***) :** qui représente le volume du trafic généré par les mises à jour des répliques (*pour maintenir le degré de réplication k*) **,** mesuré par le nombre de messages générés après l'arrivée (*join*) ou le départ (*leave*) des nœuds.

2.2 Paramètres de simulation

Afin d'effectuer les expérimentations, nous avons utilisé un certain nombre de paramètres dont les valeurs sont définies comme suit:

- Taille de la liste des Successeur : 8
- Délai de Join (*délai entre les tentatives de join*): 10 secondes
- Taille de *replica-Set* (*facteur ou degré de réplication*) : k = 4
- Nombre total des nœuds dans le système : $[1..2^{32}]$

- Ratio super-peer α :1 /10,1/50,1/100,1/1000 *(Ratio super –peer* ***α*** *: est le nombre total des super-peers divisé par le nombre total des peers dans le système)*

2.3 Résultats et analyses

Nous présentons dans cette section l'ensemble des résultats auxquels nous avons abouti tout en fournissant les interprétations nécessaires. Les performances de notre architecture DOSN hiérarchique à 2 tiers (*DOSN-2 Tiers*) étaient sujettes de comparaison avec les performances des DOSNs plates (*DECENT, Cachet, Safebook*) qui sont parmi les DOSNs les plus référencés dans le domaine. Ces comparaisons ont été menées sous des conditions différentes de nombre de nœuds et de nombre de clusters.

2.3.1 Coût de recherche (*Nombre de sauts*)

La figure 5.1 illustre le coût de recherche obtenu pour notre architecture à base de DHT à 2 niveaux (*DOSN -2Tiers*) comparé avec l'architecture DOSN DECENT à base de DHT plate FreePastry

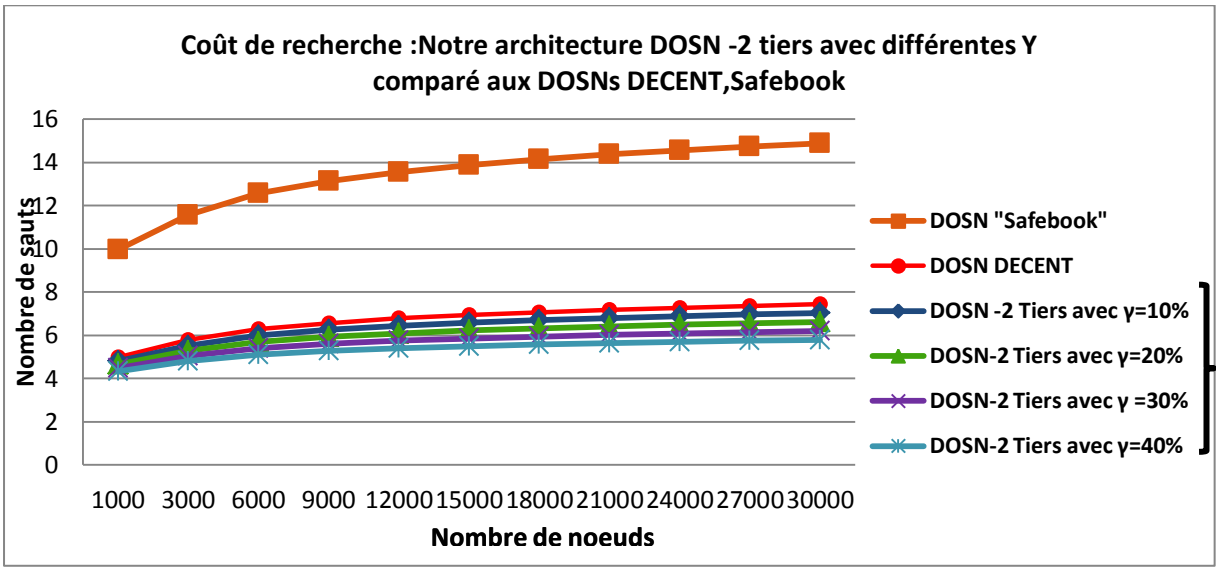


Figure 5. 1: Coût de recherche (*nombre de sauts*) : DOSN- 2 Tiers et DOSNs DECENT

Nous remarquons que notre architecture hiérarchique proposée donne de meilleurs coûts de recherche par rapport aux autres architectures (*DECENT, Cachet, Safebook*). La figure 5.1 montre que le coût de recherche en termes de nombre de sauts est réduit d'une manière significative comparativement au DOSN DECENT avec différentes valeurs de γ (*la probabilité que le nœud source et destination soient dans le même cluster*).

Le ratio de super-peer **α** est fixé à 1: 100. Les différentes courbes représentent le nombre de sauts nécessaires à notre architecture, pour différentes valeurs de probabilité.

On peut voir que le nombre de sauts pour notre architecture est inférieure à l'architecture DECENT lorsque la probabilité de trouver la donnée à l'intérieur du cluster est seulement de 10 %. En outre, la performance de notre système augmente avec l'augmentation des valeurs de probabilité.

La figure 5.2 représente le coût de recherche de notre architecture DOSN 2-Tiers en variant La valeur de γ de 0 à 1 avec un nombre de nœuds $N = 2^{32}$, et le nombre de clusters $C = 2^{10}$.

A titre d'exemple, pour un cluster de 1024 nœuds γ = 0.1, le coût de recherche dans DOSN 2-Tiers est 15 sauts, il est égale à 32 sauts pour les autres architectures DOSN. Par conséquent, quand la probabilité de trouver la donnée à l'intérieur de cluster augmente ; le coût de recherche décrémente (*les nœuds du même cluster sont géographiquement proches*).

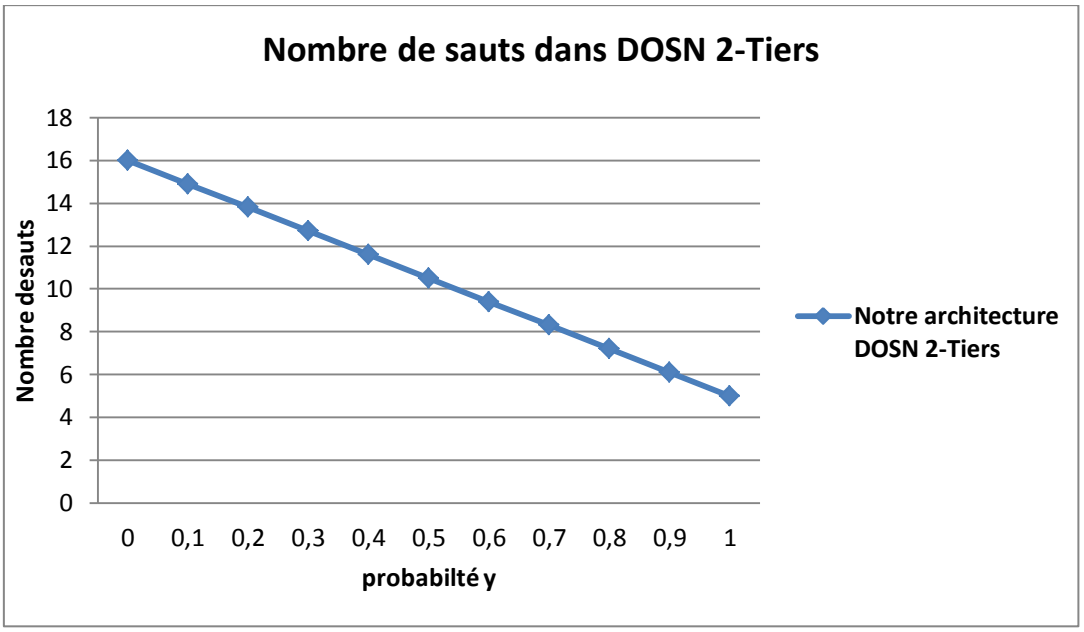


Figure 5. 2: Nombre de sauts dans l'architecture DOSN 2-Tiers avec différentes probabilité γ

La figure 5.3 représente le coût de recherche de notre architecture DOSN 2-Tiers en variant le ratio super-peer de $\frac{1}{10}, \frac{1}{50}, \frac{1}{100}, \frac{1}{1000}$, avec un nombre de nœuds [100,10000] et une probabilité γ = 0.5 , comparé avec les DOSNs DECENT, Safebook.

Nous remarquons que le nombre de sauts augmente avec l'augmentation du ratio super-peer.

Pour le ratio $\alpha = \frac{1}{10}$ le nombre de sauts pour N = 10000 est 3 sauts, et pour $\alpha = \frac{1}{1000}$ et un nombre de nœuds N = 10000, le nombre de sauts = 6 , 2 fois plus le nombre de sauts pour $\alpha = \frac{1}{10}$.

Nous pouvons constater l'impact de la taille du cluster sur les performances de notre architecture.

Avec les ratios $\frac{1}{10}, \frac{1}{50}, \frac{1}{100}$, et pour les courbes (3),(4),(5) respectivement , notre architecture est meilleure en terme de nombre de sauts comparée aux architectures Safebook (*courbe (1)*),DECENT (*courbe (2)*).Cependant ,pour le ratio $\frac{1}{1000}$, représenté par le courbe (6), et pour N< =1000 , l'architecture DECENT donne des performances équivalentes à notre architecture (*lorsque N=1000 ,le nombre de sauts de notre architecture et de DECENT est de 5 sauts*).

Nos résultats indiquent que n'est pas recommandé de créer de grands groupes. Avec quelques milliers de nœuds dans chaque groupe, le nombre de sauts augmente ce qui dégrade les performances de notre système.

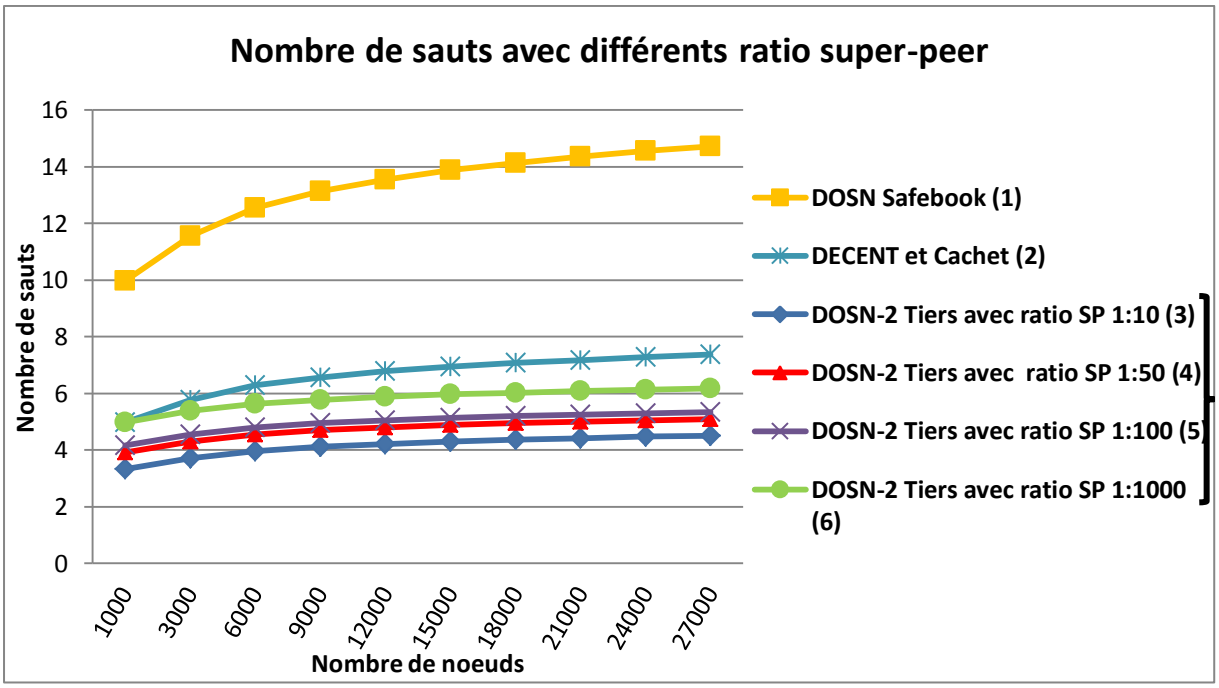


Figure 5. 3:Nombre de sauts avec différents Ratios super-peer comparé avec les DOSNs DECENT, Safebook

La figure 5.4 illustre bien l'impact de la taille du cluster sur les performances globales de notre architecture (*La plupart des trafics traverseront les liens inter-cluster déjà congestionnées par les clusters, va encourir des délais de réseau intolérables*), en fixant le

nombre de nœuds N à 2^{32} (4294967296), et la probabilité γ à 0.5 , en variant la taille du cluster N/S de 1 jusqu'à 4294967296.

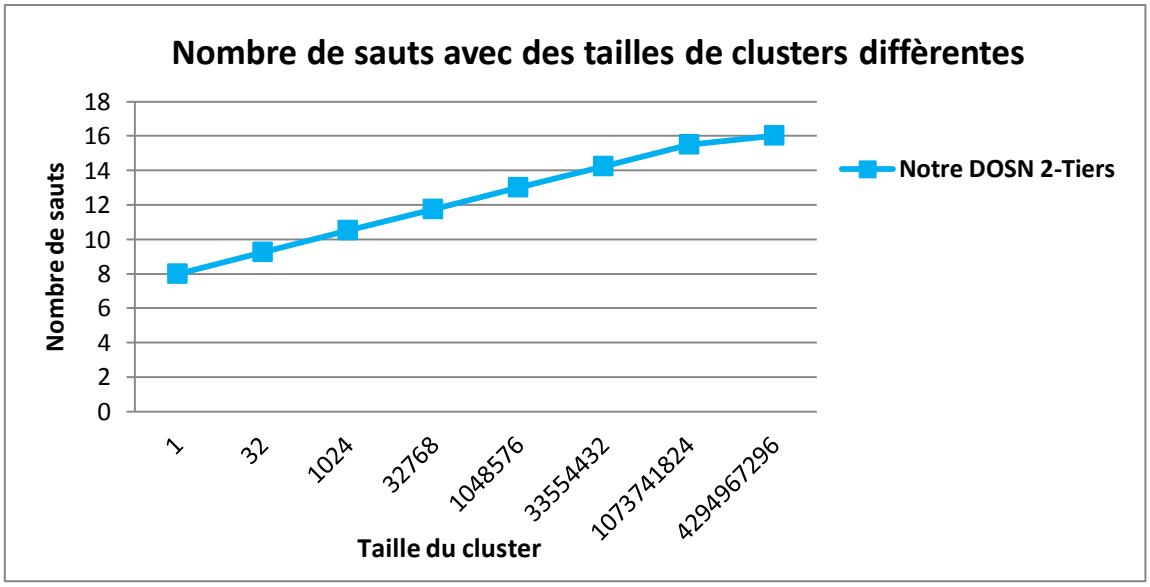


Figure 5. 4:Nombre de sauts avec des tailles différentes de clusters

2.3.2 Coût des structures de données

La figure 5.5 illustre la taille de la table de raccourcis (*Finger table*) pour notre architecture comparé aux autres architectures DOSNs Safebook, DECENT, et Cachet .

Avec l'augmentation du nombre de nœuds, la taille de cette table augmente, mais pour notre architecture cette augmentation n'est pas considérable (*elle est proche d'être stable*) par rapport aux autres architectures comme elle montre la figure, la table de raccourcis dans notre architecture est égale à $S*\log_2(S)$ pour le niveau supérieur , et $\log_2(N/S)$ dans chaque cluster, ce qui est donne comme taille total $S*\log_2(S) + \log_2(N/S)$. Par contre dans les autres architectures, la taille de la table de raccourcis augmente d'une manière logarithmique. La taille de cette table est un facteur critique pour le passage à l'échelle (*scalabilité*), surtout dans des environnements critiques (*ex. réseaux mobiles ou réseaux de capteurs*).

Nous constatons que notre architecture passe à l'échelle mieux que les autres architectures DOSNs.

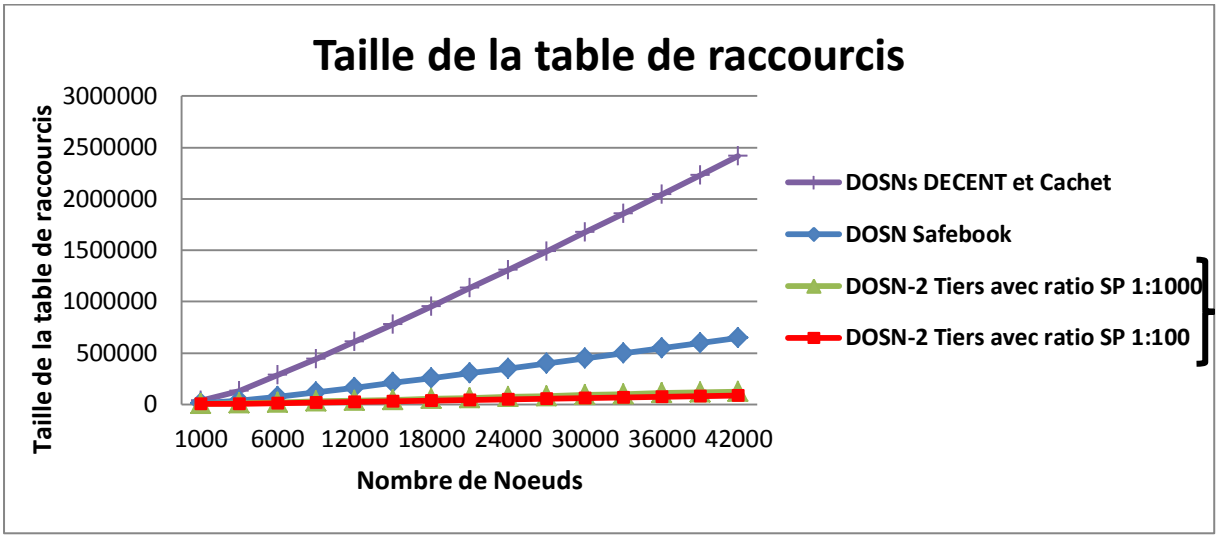


Figure 5. 5:Taille de la table de raccourcis

2.3.3. L'overhead de réplication

La figure 5.6 illustre le nombre de messages (*l'overhead généré lors de migration des répliques*) générés après l'évènement de join des nouveaux nœuds, pour notre architecture DOSN-2 Tiers comparé avec les deux architectures DECENT et Cachet. Pour assurer une meilleure disponibilité (*Availability*) des données avec un overhead limité, nous avons choisi une stratégie de réplication dans les nœuds les plus disponibles dans l'overlay (*Most-available replication strategy)*. Les architectures DOSNs qui assurent une haute disponibilité des données, souffrent d'un overhead important, car elles appliquent une stratégie de réplication simple (*la réplication est effectuée dans des nœuds choisi aléatoirement*).

Les résultats de simulation interprétés par la figure 5.6 montre que l'overhead pour l'architecture DOSN-2 Tiers est significativement réduit par rapport aux autres architectures DECENT et Cachet. A titre d'exemple, lorsque 100 nouveaux nœuds joignent l'overlay, et pour restaurer le degré de réplication k (*k=4*) , la stratégie de réplication adoptée par notre architecture DOSN-2 Tiers génère 200 messages, tandis que la stratégie de réplication pour les architectures DECENT et Cachet donne 400 messages. L'explication vient du choix des nœuds de réplication dans les deux stratégies (*replicas_set)*. Pour la stratégie adoptée par notre architecture ,si un nouveaux nœud n'a pas une meilleure disponibilité par rapport aux nœuds de *replicas_set (Dispo>=0.5)*, les données ne seront pas migrer vers ce nœud, donc pas de messages générés, par contre dans le cas de l'autre stratégie de réplication, les données seront migrer vers les nouveaux nœuds sans voir leur disponibilité.

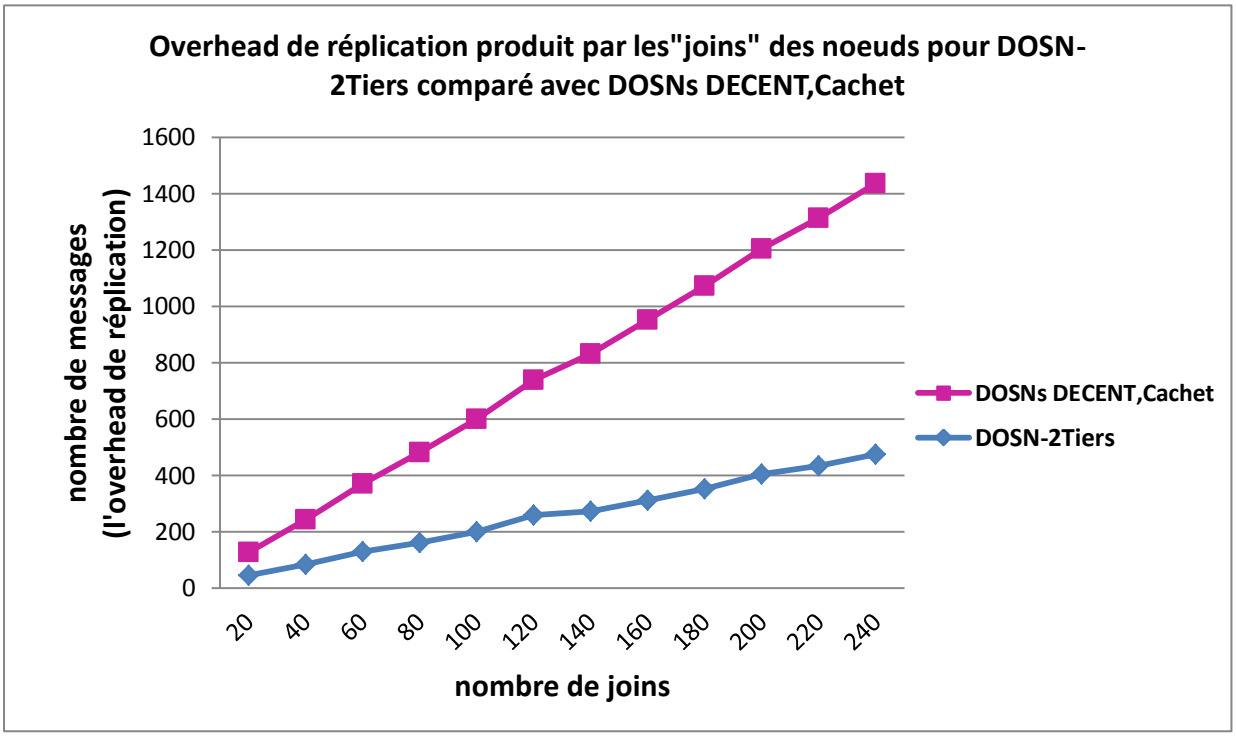


Figure 5. 6: Overhead de réplication produit par les"joins" des noeuds pour DOSN-2Tiers comparé avec DOSNs DECENT,Cachet

Les figures 5.7 et 5.8 montrent l'overhead généré par les événements de l'arrivée (*join*) et de départ (*leave*) respectivement pour la stratégie de réplication simple et la stratégie de Most available replication utilisée par notre architecture, avec des différentes tailles de replicas_set. Les figures illustrent bien la réduction de l'overhead obtenue par la stratégie de Most available replication par rapport à la stratégie de réplication simple.

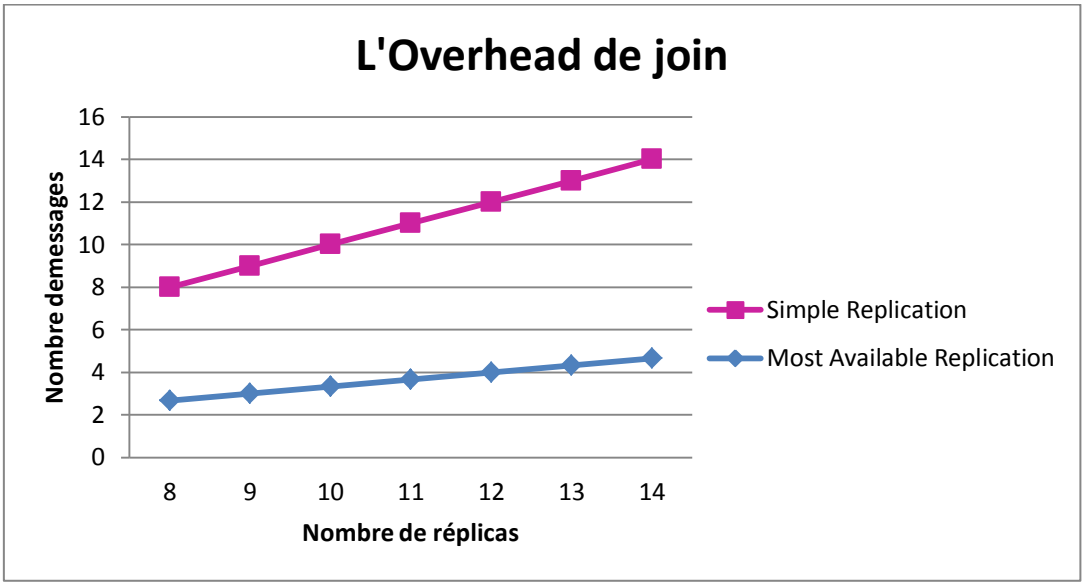


Figure 5. 7: Overhead de join

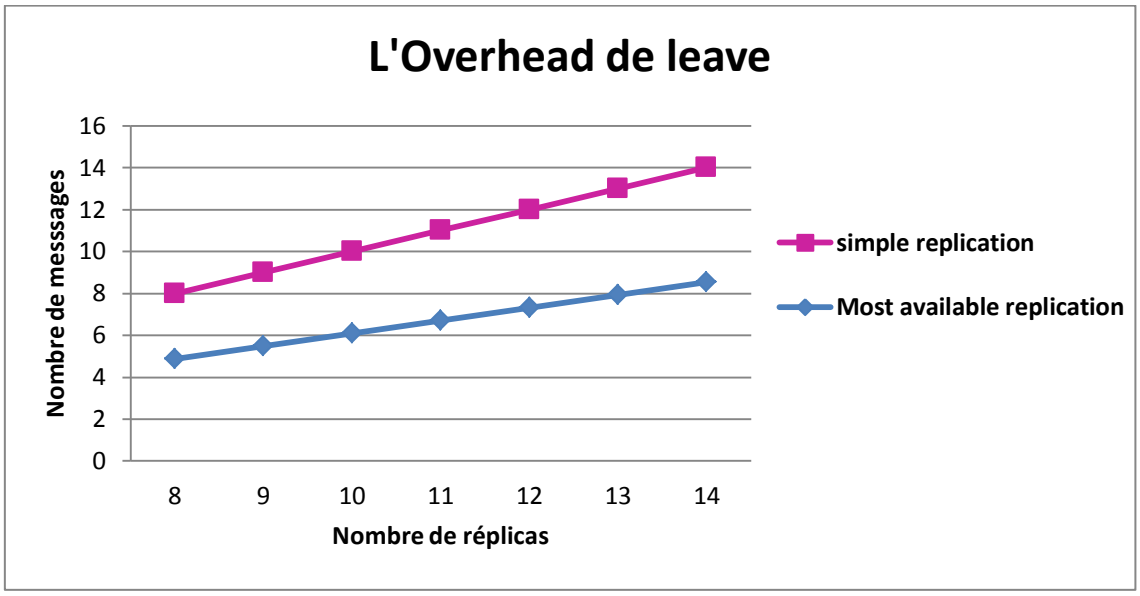


Figure 5. 8:Overhead de Leave

3.Conclusion

Dans ce chapitre, nous avons évalué les performances de notre architecture DOSN hiérarchique en la comparant avec trois architectures DOSNs concurrents. Nous avons constaté que notre architecture offre de meilleurs résultats en termes de performances du réseau (*cout de recherche*) , et qu'elle est mieux adaptée à aux réseaux sociaux en ligne.

Conclusion générale et Perspectives

L'explosion des réseaux sociaux a mis en lumière l'importance de leur usage dans les communications d'aujourd'hui. Cependant leur architecture centralisée qui présente de nombreux problèmes concernant la protection de la vie privée de l'utilisateur, a mis en évidence le besoin d'une architecture alternative; une architecture décentralisée, qui est devenu une nécessité.

Une infrastructure décentralisée peer-to-peer pour les réseaux sociaux offre des avantages et des opportunités pour plusieurs parties concernées par le phénomène des réseaux sociaux en ligne. Les utilisateurs peuvent conserver les avantages des réseaux sociaux en ligne traditionnels tout en profitant de contrôle sur leur vie privée et de la propriété intellectuelle, et de compléter l'accès Web par échange direct. Par conséquent, l'aspect d'optimisation sur plusieurs critères des réseaux sociaux distribués (*espace mémoire, temps de calcul, temps de réponse, durée de vie des données, disponibilité des données*,…) est devenu un challenge.

Dans ce mémoire, nous nous sommes intéressés à l'optimisation d'une architecture P2P pour réseaux sociaux. Cela nous a fait étudier un ensemble des architectures DOSNs les plus intéressantes dans ce domaine, et nous a permis de découvrir les problèmes courants des DOSNs. Nous avons étudié différents aspects de quatre propositions DOSNs. Bien que cet ensemble ne vise pas à être exhaustif, elle fournit une large couverture des différents choix de design.

Les architectures DOSNs existantes étudiées sont des architectures P2P basées sur des DHTs plates non hiérarchiques. Ces différents DOSNs offrent différents degrés de décentralisation et abordent différents sous-ensembles de problèmes liées la sécurité, mais souffrent de plusieurs inconvénients qui concernent la disponibilité des données des utilisateurs. Notre contribution a consisté de proposer une architecture P2P hiérarchique à deux niveaux qui apporte plusieurs avantages par rapport aux architectures DOSNs étudiées en termes de scalabilté, efficacité, disponibilité avec moins d'overhead. Les coûts de construction et maintenance de l'overlay dans notre système sont également moins par rapport à des implémentations plates.

Notre architecture réduit le nombre moyen de sauts pour une requête de recherche ce qui permet d'obtenir moins d'overhead. La topologie overlay du niveau supérieur constituée de super-peers stables aura plus de stabilité. Cette stabilité accrue permet aux requêtes de recherche d'atteindre des performances optimales concernant le coût de recherche (nombre de sauts). Elle permet aussi faciliter le déploiement à grande échelle en fournissant une autonomie administrative et une transparence tout en permettant à chaque groupe participant de choisir son propre protocole de l'overlay. Autrement dit, tous les événements de churn au sein d'un groupe sont locaux pour le groupe en termes de changements, et les tables de routage en dehors du groupe ne sont pas affectées.

Au début, nous avons étudié les réseaux P2P de manière générale, nous avons cité quelques domaines d'applications de ces réseaux, leurs caractéristiques, les différentes architectures et les problèmes courants du P2P.

Après, nous avons fait une étude sur les réseaux sociaux, leur définition, leur intérêt, leurs enjeux et leurs perspectives, ensuite, nous avons présenté les différents architectures DOSNs proposées dans la littérature.

Nous avons proposé une nouvelle architecture P2P hiérarchique pour les réseaux sociaux basée sur les tables de hachage distribuées de deux niveaux Cette architecture traite plusieurs problèmes des architectures existantes : la scalabilité, le cout de recherche optimal et la disponibilité avec un minimum d'overhead, en minimisant le cout de maintenance engendré par migration des données par l'adoption d'une stratégie de réplication.

Nous envisageons comme perspective du travail réalisé dans ce mémoire :

- Prise en charge explicite de la proximité géographique dans la construction des clusters (*les membres du même cluster sont proches géographiquement*).
- Prise en charge de l'aspect d'amitié dans la construction des clusters pour renforcer la sécurité.
- Prise en charge de l'aspect de sécurité, dans les deux niveaux de l'architecture.
- Développement d'un mécanisme de choix des super-peers dans les overlays du niveau 1

Références bibliographiques

[1] KEEPER SECURITY. [En ligne]. https://blog.keepersecurity.com/2012/06/07/linkedin-passwords-leaked-6-5-million-accounts-compromised/

[2] DWYER Catherine, "Privacy in the Age of Google and Facebook," *Technology and Society Magazine, IEEE*, vol. 30, no. 3, pp. 58-63, 2011.

[3] SITE WEB de l'universite de pau et des pays de l'adour. [En ligne]. http://web.univ-pau.fr/~cpham/M2SIR/BIBLIO/DOC02-03/p2p.doc

[4] CAPIT. [En ligne]. http://www.capit.net/technique/p2p.aspx

[5] AMAD Mourad, "Découverte et localisation de services en mode P2P," Université Abderrahmane Mira de Béjaia, mémoire du magister 2005.

[6] HAFI Houda, "Protocole pour la Sécurité des Réseaux Sans Fil Peer To Peer," Université Kasdi Merbah- Ouargla, mémoire du Magister 2012.

[7] MAZYAD Hanaa, "Une approche Multi-agents à Architecture P2P pour l'apprentissage collaboratif," Université du Littoral Côte d'Opale, Thèse de Doctorat 2013.

[8] [En ligne]. http://webilus.com/tableau/repartition-du-trafic-internet-web-ftp-email-p2p

[9] [En ligne].http://www.isima.rnu.tn/pages%20fr/telechargement/cours%20td%20tp/cours/troisi-eme%20annee/PaP-louati-isima-p1-p27.zip

[10] AMAD mourad , MEDDAHI Ahmed , "P4L : a four layers P2P model for optimizing resources discovery and localization," *9th Asia-Pacific Network Operations and Management Symposium (APNOMS),Springer-verlag,LNCS 4238*, pp. 342-351, Septembre 2006.

[11] SHEN Haiying, XU Cheng-Zhong, et CHEN Guihai Shena , "Cycloid : A constant-degree and lookup efficient P2P overlay network," *Journal of performance evaluation, Elsevier*, vol. 63, no. 3, pp. 195-216., 2006.

[12] STOICA Ion, MORRIS Robert, KARGER David, et al, "Chord: A Scalable Peer-to-peer Lookup Protocol for Internet Applications," *IEEE/ACM Transactions on networking*, vol. 11, no. 1, Novembre 2003.

[13] RATNASAMY Sylvia, FRANCIS Paul, HANDLEY Mark, et al, "A scalable centent adressable network," *ACM SIGCOMM*, pp. 161-175, Aout 2001.

[14] TRAN Duc A, NGUYEN T, "Hierarchical multidimensional search in peer-to-peer networks," *Journal of Computer Communications, Elsevier*, vol. 31, no. 2, pp. 346-357, 2008.

[15] SECURITE INFORMATIQUE. [En ligne]. https://www.securiteinfo.com/cryptographie/hash.shtml

[16] AMAD Mourad, "Performance et Optimisation des Architectures P2P Pour Les Applications à Grande Échelle," Université de Bejaia, thèse de Doctorat avril 2011.

[17] GNUTELLA. [En ligne]. http ://www.gnutella.com.

[18] NAPSTER. [En ligne]. http ://www.napster.com.

[19] BITTORRENT. [En ligne]. http://www.bittorrent.com/

[20] CRAMER Curt, KUTZNER Kendy, et FUHRMANN Thomas, "Bootstrapping locality-aware P2P networks," *Networks, 2004.(ICON 2004). Proceedings. 12th IEEE International Conference on. IEEE*, pp. 357-361, 2004.

[21] KEIDAR Idit, MELAMED Roie, et ORDA Ariel, "Equicast: Scalable multicast with selfish users," *Computer Networks*, vol. 13, no. 53, pp. 2373-2386, 2009.

[22] AMAD Mourad, MEDDAHI Ahmed, AISSANI Djamil, et al, "HPM: a novel hierarchical peer-to-peer model for lookup acceleration with provision of physical proximity," *Journal of Network and Computer Applications*, vol. 6, no. 35, pp. 1818-1830, 2012.

[23] SHEN Heng Tao, SHU Yanfeng, et YU Bei, "Efficient semantic-based content search in P2P network. Knowledge and Data Engineering," *IEEE Transactions on*, vol. 16, no. 7, pp. p. 813-826, 2004.

[24] AMAD Mourad, MEDDAHI Ahmed, "A Scalable P2P model for optimizing application layer multicast," *The 6th ACS/IEEE International Conference on computer systems and applications (AICCSA'08)*, 2008.

[25] RAINER Benjamin, TIMMERER Christian, KAPAHNKE Patrick, et al, "Real-time multimedia streaming in unstructured peer-to-peer networks," *Consumer Communications and Networking Conference (CCNC), 2014 IEEE 11th*, pp. 1136-1137, Janvier 2014.

[26] GODFREY Brighten, LAKSHMINARAYANAN Karthik, SURANA Sonesh, et al, "Load balancing in dynamic structured P2P systems," *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4, 2004.

[27] YU Qifeng, XU Tianyin, YE Baoliu, et al, "SkipStream: A clustered skip graph based on-demand streaming scheme over ubiquitous environments," *Parallel Processing, 2009. ICPP'09. International Conference on. IEEE*, pp. 269-276, 2009.

[28] FRAIGNIAUD Pierre, GAURON Philippe, "D2B: a de Bruijn based content-addressable network," *Theoretical Computer Science*, vol. 355, no. 1, pp. 65-79, 2006.

[29] SUZUKI Yasuto, KANEKO Keiichi, "An algorithm for node-disjoint paths in pancake graphs," *IEICE TRANSACTIONS on Information and Systems*, vol. 86, no. 3, pp. 610-615, 2003.

[30] FERTIN Guillaume, RASPAUD André, "A survey on Knödel graphs," *Discrete Applied Mathematics*, vol. 137, no. 2, pp. 173-195, 2004.

[31] WANG Yufeng, NAKAO Akihiro, VASILAKOS Athanasios V, et al, "P2P soft security: On evolutionary dynamics of P2P incentive mechanism," *Computer Communications*, vol. 34, no. 3, pp. 241-249, 2011.

[32] PAULSAMY Victor, CHATTERJEE Samir, "Network convergence and the NAT/Firewall problems," *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on. IEEE*, p. 10, 2003.

[33] DAI Zhihui, VIALE Fabien, CHI Xuebin, et al, "A task-based fault-tolerance mechanism to hierarchical master/worker with divisible tasks," *High Performance Computing and Communications, 2009. HPCC'09. 11th IEEE International Conference on. IEEE*, pp. 672-677, 2009.

[34] FRIENDSTER. [En ligne]. http://www.friendster.com/

[35] MYSPACE. [En ligne]. http://fr.myspace.com/

[36] ZAMMAR Nisrine, "Reseaux Sociaux numeriques: essai de categorisation et cartographie des controverses," Université Rennes 2, Thèse de doctorat 2012.

[37] HEER Jeffrey, BOYD Danah, "Vizster:Visualizing online social networks," *Information Visualization, 2005. INFOVIS 2005. IEEE Symposium on. IEEE*, pp. 32-39, 2005.

[38] LES Z'ED. [En ligne]. http://les-zed.com/qu-est-ce-que-les-reseaux-sociaux/

[39] CREFF Marie, "Réseaux sociaux : quelles opportunités pour les services?Le cas de l'assistance en ligne d'Orange," Institut national des techniques de la documentation , Mémoire de fin d'étude INTD 2010.

[40] Bibliothèque municipale de Lyon. [En ligne]. http://www.pointsdactu.org/article. php3?idarticle=1293

[41] ELLISON Nicole, et al, "Social network sites: Definition, history, and scholarship," *Journal of Computer-Mediated Communication*, vol. 13, no. 1, pp. 210-230, 2007.

[42] TORLOTING Philippe, "Enjeux et perspectives des réseaux sociaux," Institut Supérieur du Commerce,Paris, mémoire de fin d'étude 2006.

[43] LIEN OPTIONNEL. [En ligne]. http://www.lien-optionnel.com/reseaux-sociaux.html

[44] LES PETITS DEBROUILLARDS. [En ligne].http://lespetitsdebrouillardspc.org/IMG/pdf/reseaux sociaux.pdf

[45] THELWALL Mike, "Social network sites: Users and uses," *M. Zelkowitz (Ed.) Advances In computers Elsevier*, vol. 76, pp. 19-73, 2009.

[46] FREDERIC cavazza. [En ligne]. http://www.fredcavazza.net/2015/05/29/panorama-des-medias-sociaux-2015/

[47] CANALBLOG. [En ligne]. http://reseauxlapie.canalblog.com

[48] SOCIALTIMES. [En ligne]. http://www.adweek.com/socialtimes/vincos-blog-world-map-of-social-networks-august-2015/626833

[49] FILLIETTAZ François, GREGORI Marco, "Comprendre les réseaux sociaux numériques," *Direction des systèmes d'information et service écoles-médias*, 2011.

[50] BOYD danah, "Social Network Sites: Public, Private, or What?," *Knowledge Tree*, May 2007.

[51] ZHANG Chi, JINYUAN Sun, XIAOYAN Zhu, et YUGUANG Fang, "Privacy and Security for Online Socia lNetworks: Challenges and Opportunities," Juilllet/Aout 2010.

[52] VADIVU, G. Senthil, GOMATHI, C., et PRIYA A, "Protecting peer-to-peer networks from the denial of service attacks, 3rd International Conference on Electronics Computer Technology (ICECT)," 2011. [En ligne]. http://fr.wikipedia.org/wiki/Attaque_par_déni_de_service

[53] DATTA Anwitaman, BUCHEGGER Sonja, VU Le-Hung, THORSTEN Strufe, et KRZYSZTOF Rzadca, "Decentralized Online Social Networks," *Handbook of Social Network Technologies and Applications. Springer US*, pp. 349-378, 2010.

[54] PEERSON. [En ligne]. www.peerson.net

[55] BUCHEGGER Sonja, SCHIÖBERG Doris, VU Le-Hung, and DATTA Anwitaman, "PeerSoN: P2P social networking: early experiences and insights," , *et Proceedings of the Second ACM EuroSys Workshop on Social Network Systems. ACM*, pp. 46-52, 2009.

[56] SAFEBOOK. [En ligne]. www.safebook.us

[57] CUTILLO Leucio Antonio, "Protection des Données Privées dans les Réseaux Sociaux," ENST, Paris, Thèse de doctorat Avril 2012.

[58] CUTILLO Leucio Antonio, MOLVA Refik, et STRUFE Thorsten, "Safebook: Feasibility of Transitive Cooperation for Privacy on a decentralized social network," *World of Wireless, Mobile and Multimedia Networks & Workshops, 2009.WoWMoM 2009. IEEE International Symposium on a. IEEE, 2009*, pp. 1-6, December 2009.

[59] ORNAGHI Alberto, VALLERI Marco, "Man in the middle attacks Demos," *Blackhat Conference USA*, vol. 19, 2003.

[60] ZHANG Yanchao, FANG Yuguang, "A Fine-Grained Reputation System for Reliable Service Selection in Peer-to-Peer Networks," *Parallel and Distributed Systems, IEEE Transactions*, vol. 18, no. 8, pp. 1134-1145, Aout 2007.

[61] XIANG Xu, "Defeating against Sybil-attacks in Peer-to-peer Networks," *Parallel and Distributed Processing Symposium Workshops & PhD Forum (IPDPSW), 2012 IEEE 26th International. IEEE*, pp. 1218-1222, 2012.

[62] JAHID Sonia, NILIZADEH Shirin, MITTAL Prateek, BORISOV Nikita, et KAPADIA Apu, "DECENT: A decentralized Architecture for enforcing privacy in Online Social Networks," *In*

Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on. IEEE, pp. 326-332, Mars 2012.

[63] KHATIR Abderrezzaq, LABBAS Imane, "Conception et réalisation d'un réseau social distribué," université ABOU BAKR BELKAID, TLEMCEN, mémoire du magister 2013.

[64] NILIZADEH Shirin, JAHID Sonia, MITTAL Prateek, BORISOV Nikita, et KAPADIA Apu, "Cachet: A Decentralized Architecture for Privacy Preserving Social Networking with Caching," *CoNEXT '12 Proceedings of the 8th international onference on Emerging networking experiments and technologies*, 2012.

[65] CHOWDHURY Shihabur Rahman, ROY Arup Raton, SHAIKH Maheen, et DAUDJEE Khuzaima, "A taxonomy of decentralized online social networks," *Peer-to-Peer Networking and Applications*, vol. 8, no. 3, pp. 367-383, mai 2014.

[66] SHARMA Rajesh, DATTA Anwitaman, "Supernova: Super-peers based architecture for decentralized online social networks," *Communication Systems and Networks (COMSNETS), 2012 Fourth International Conference on. IEEE*, pp. 1-10, 2012.

[67] CUTILLO Leucio Antonio, MOLVA Refik, et ÖNEN Melek, "Safebook: A distributed privacy preserving online social network," *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2011 IEEE International Symposium on a. IEEE*, pp. 1-3, 2011.

[68] MAYMOUNKOV Petar, MAZIERES David, "Kademlia: A peer to peer Information System Based on the XOR Metric," *Peer-to-Peer Systems. Springer Berlin Heidelberg*, pp. 53-65, 2002.

[69] SALOMONI Paola, MURATORI Ludovico Antonio, PAU Giovanni, and al, "S2S: a peer to peer protocol for participative sensing," *Proceedings of the 27th Annual ACM Symposium on Applied Computing. ACM*, pp. 634-635, 2012.

[70] AUTHENTIFY. [En ligne]. http://www.authentify.com/solutions/out-of-band-authentication.html

[71] GOLDFARB Stephen M, "Writing policies and procedures manuals," *IEEE Transactions on Professional Communication*, 2013.

[72] PAUL Thomas, FAMULARI Antonino, et STRUFE Thorsten, "A Survey on decentralized online social networks," *Computer Networks*, vol. 75, pp. 437-452, octobre 2014.

[73] HAN Jinguang, SUSILO Willy, MU Yi, et al, "Privacy-Preserving Decentralized Key-Policy Attribute-Based Encryption," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 11, pp. 2150-2162, 2012.

[74] HOFFMAN Lance J., ALI Faraz A., HECKLER Steven L., et al, "Cryptography policy," *Communications of the ACM*, vol. 37, no. 9, pp. 109-117, 1994.

[75] BIAN Jiang, "An analysis of the skype peer-to-peer voip system," *College of Computing Georgia Institute of Technology*, 2006.

[76] GARCES-ERICE Luis, "Un Réseau P2P Hiérarchique:Design et Applications," ENST, Paris, thèse de doctorat 2004.

[77] PANDEY Mayank, AHMED Syed Mushtaq, et CHAUDHARY Banshi Dhar, "2T- DHT: A Two Tier DHT for Implementing Publish/Subscribe," *Computational Science and Engineering, 2009. CSE'09. International Conference on. IEEE*, pp. 158-165, 2009.

[78] LI Bo-Wei, WANG Kuochen, et HSIEH Yi-Ling, "A hierarchical social network-based P2P SIP system for mobile environments," *Personal Indoor and Mobile Radio Communications (PIMRC), 2010 IEEE 21st International Symposium on. IEEE*, pp. 2581-2585, 2010.

[79] BUFORD John, YU Heather, et LUA Eng Keong, "P2P networking and applications". USA: Kaufmann Series in Networking, Elsevier Inc, 2009.

[80] LI Kuan-Ching, HSU Ching-Hsien, YANG Laurence Tianruo, DONGARRA Jack, et Zima et Hans," Handbook of Research on Scalable Computing Technologies", *chapitre Hierarchical Structured Peer-to-Peer Networks.* USA, USA: IGI Global, 2010.

[81] PACE Alessio, QUÉMA Vivien, et SCHIAVONI Valerio, "Exploiting node connection regularity for DHT replication," *Reliable Distributed Systems (SRDS), 2011 30th IEEE Symposium on. IEEE*, pp. 111-120, 2011.

[82] PAIVA João, LEITÃO João, et RODRIGUES Luís, "Rollerchain: a DHT for high availability," *Proceedings of the Workshop on Posters and Demos Track. ACM*, p. 17, 2011.

[83] MICKENS James W, NOBLE Brian D, "Exploiting availability prediction in distributed systems," *Proceedings of the International Symposium on Networked Systems Design and Implementation (NSDI),Ann Arbor*, vol. 1001, p. 48103, Mai 2006.

[84] MICKENS James W, NOBLE Brian D, "Predicting node availability in peer-to-peer networks," *ACM SIGMETRICS Performance Evaluation Review. ACM*, vol. 33, no. 1, pp. 378-379, 2005.

[85] LEDLIE Jonathan, SHNEIDMAN Jeff, AMIS Matthew, et SELTZER Margo, "Reliability-and capacity-based selection in distributed hash tables," Harvard University Computer Science, Technical report Septembre 2003.

[86] KIM Kyungbaek , PARK Daeyeon, "Reducing Replication Overhead for Data Durability in DHT based P2P System," *IEICE Transactions on Information and Systems*, vol. E90-D , no. 9, pp. 1452-1455, Septembre 2007.

[87] SANCHEZ ARTIGAS Marc, "A Hierarchical Framework for Peer-to-Peer Systems: Design and Optimizations," Université Pompeu Fabra (UPF), Espagne, Thèse de doctorat 2009.

[88] CINDY Hazan, CAMPA Mary, "Human Bonding: The Science of Affectional Ties", Guilford Press, 2013.