République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi –Tébessa-
Faculté des Sciences Exactes et des Sciences de la Nature et de la Vie
Département : Informatique

MEMOIRE DE MASTER
Domaine: Informatique
Filière: Informatique
Option: Réseaux & sécurité informatique

Thème:

Ontologies dans les réseaux pairs à pairs

Présenté par:
Basli Med El Mahdi
Fetni Aya

Devant le jury:

Menassel Rafik	MAA	Université de Larbi Tébessi	Président
Azzeddine Abd El Ghafour	MAA	Université de Larbi Tébessi	Rapporteur
Mekhazenia Tahar	MAA	Université de Larbi Tébessi	Examinateur

Date de soutenance: 29/05/2016

Note :…… Mention :……..……

Résumé

Malgré le succès des systèmes P2P dans le domaine du partage de fichiers, ils ont étaient capables d'évaluer uniquement des requêtes simples. Nouvellement, plusieurs travaux de recherche sont effectués afin de montrer ces systèmes pour qu'ils permettent le partage de données avec une granularité fine et l'évaluation de tous types de requêtes.

Et à cause des propriétés du système P2P (distribution des données, tolérance aux pannes, passage à l'échelle, dynamicité et autonomie de nœuds) et surtout le domaine de partage de données nous voulons améliorer les réponses obtenues par les pairs du réseau concerné. Pour ces raisons nous avons proposés d'utiliser les ontologies dans les systèmes P2P pour améliorer les résultats obtenus et, au même temps, minimiser le temps de réponse.

Dans notre mémoire nous proposons une approche pour l'amélioration et donc l'optimisation des résultats obtenus par les utilisateurs du système P2P. Notre approche est fondée sur l'intégration des ontologies dans les systèmes P2P en se basant sur le regroupement des ontologies des utilisateurs qui ont les mêmes profils dans des Cluster (groupes).

Abstract

Despite the success of P2P systems in the area of file sharing, they were able to assess only simple queries. Recently, more research is conducted to show these systems to allow data sharing with fine granularity and evaluation of all types of queries.

Because of the P2P system properties (data distribution, fault tolerance, scalability, dynamicity and knots autonomy) and especially the data sharing area, we want to improve the responses that are obtained by the peers of the very network. For these reasons we have proposed to use ontologies in P2P systems to improve performance and, at the same time, minimize the response time.

In our brief project, we propose an approach for the improvement and thus the optimization of results by the users of P2P system. Our approach is based on the integration of ontologies in P2P systems relying on the grouping of the ontologies of the users who have the same profiles in the clusters (groups).

ملخص

على الرغم من نجاح أنظمة الـ P2P في مجال مشاركة الملفات، إلا أنهم كانوا قادرين فقط على تقييم الاستعلامات أو الاستفسارات البسيطة. حديثا، بعد اجراء المزيد من البحوث لإظهار أن هذه الأنظمة تسمح بتبادل البيانات بتفاصيل دقيقة و تقييم جميع أنواع الاستعلامات، ونظرا لخصائص نظام الـ P2P (توزيع البيانات، خطا التسامح، التدرج، الحيوية و الاستقلالية) و خاصة مجال مشاركة البيانات.

نحن نرغب في تحسين الاجابات المحصلة من طرف مستخدمي الشبكة المعنية، لهذه الأسباب اقترحنا استخدام تجميعات في أنظمة الـ P2P لتحسين النتائج و في نفس الوقت التقليل في زمن الاستجابة.

في مذكرتنا نقترح فكرة لتحسين النتائج المحصلة من طرف مستخدمي نظام الـ P2P، و تستند فكرتنا على ادماج التجميعات في نظام الـ P2P، وذلك بالاعتماد على ادماج التجميعات الخاصة بالمستخدمين الذين يملكون نفس اللمحات أو الميولات في مجموعات أو كتل.

Dédicaces

A nos grand pères, A nos grand-mères qui nous ont tous quitté mais qui de leur vivant ont toujours souhaité nous voir terminer nos études. Que Dieu –Tout Puissant- accueille nos grands-parents dans son Paradis.

A ceux qui n'ont jamais cessé de nous encourager de nous apporter leur aide et leur soutien moral et matériel à nos très chers papas ; Mohammed Naceur et Moussa. Qu'ils trouvent ici l'expression de nos reconnaissances notre amour et notre gratitude.

A celles que nous aimons et que nous adorons, celles qui nous ont mis au monde et qui ont tout fait et continue de le faire pour que nous réussissions. Celles qui nous ont donné leur amour, leur patience, leur temps en dépit de leur santé et de leur bonheur à elles. A nos mères Fatma et Zhaira Qu'elles trouvent ici toutes nos reconnaissances et nos gratitudes.

A ceux qui nous ont encouragés et soutenus et qui, sans leur sacrifice, ce travail n'aurait pas vu le jour. A notre encadreur et nos professeurs ainsi que toutes nos familles.

A nos Frères et sœurs ; Abdelmadjid, Mouna, Mohamed Amine, Amani, Radhia.

A nos chères tantes, nos chers oncles, nos cousines et cousins

A tous nos amis et surtout Oubaid qui a passé des nuits blanches avec Mahdi pour nous aider et avancer rapidement.

A toutes nos camarades.

A tous ceux qui, de près ou de loin, nous ont apporté leur aide et soutien pour l'accomplissement de ce modeste travail.

Aya et Mahdi

Remerciements

Nos remerciements les plus sincères vont à notre encadreur Mr Azzedine Abdelghafour pour toute son aide, son dévouement et son encouragement tout le long de la période de réalisation de ce projet.

Nos remerciements s'adressent aussi à tous les enseignants d'informatique qui nous ont apporté leur aide par leurs précieuses orientations.

Nous remercions également nos amis et collègues de nous avoir supportés et encouragés durant la période de notre formation.

Mahdi et Aya

Liste des Tableau Et Figures

LISTE DES TABLEAUX

LISTE DES FIGURES

Table de matières

TABLE DES MATIERES

Chapitre II :Contribution

Introduction

Contexte général :

L'informatique a connu deux grandes évolutions majeures ces quarante dernières années. D'une part, la démocratisation de l'ordinateur personnel, a permis au grand public de consulter, de manipuler et de créer du contenu numérique. D'autre part, l'apparition du Web dans les années 1990, a permis de créer du contenu et de le partager avec les autres utilisateurs du Web. Grâce à ce mode de partage une grande quantité de données est disponible sur le web, de nombreux outils ont été proposés pour y accéder à ces données en particulier des moteurs de recherche tels que Google [1], Yahoo ! [2], etc.. Qui permettent aux internautes de formuler des requêtes, généralement à l'aide de mots-clés, et de récupérer un ensemble de documents dont il souffre de problèmes de linguistique : synonymie, polysémie, etc. Pour pallier ces problèmes, une solution consiste à considérer que les documents et les requêtes peuvent être représentés à l'aide de concepts issus d'une ontologie. Une ontologie est une structure permettant de représenter des connaissances en définissant des concepts et en liant les concepts par des relations [3, 4].

Les ontologies permettent de raisonner sur les concepts qui les composent [5, 6].

Le Web fonctionne sur le modèle client/serveur : les clients accèdent aux serveurs pour consulter les données qui sont stockées. Ce paradigme possède un certain nombre de limites. En premier temps il n'est pas tolérant aux pannes car la défaillance d'un serveur rend indisponible l'ensemble des données qui s'y trouvaient. Deuxièmement l'utilisateur perd le contrôle Lorsqu'il publie une donnée sur un serveur : il est désormais tributaire du serveur et le fait que de nombreux clients se connectent au même serveur peut conduire au phénomène de goulot d'étranglement.

Les systèmes pair-à-pair (P2P) représentent une alternative au modèle client/serveur [7, 8, 9,10]. Dans ce modèle chaque participant, appelé pair ou nœud, chaque pair joue à la fois le rôle de client et le rôle de serveur. Les systèmes P2P présentent des propriétés intéressantes tel que:le passage à l'échelle, la tolérance aux pannes, l'autonomie des pairs, la gestion de la dynamicité.

Dans des systèmes à grande échelle comme le Web, les utilisateurs ont souvent des objectifs, des points de vue, des niveaux d'expertise différents [11, 12,13]. Même s'ils sont intéressés pour partager des documents sur un domaine précis, il est peu probable qu'ils arrivent à s'entendre sur l'usage d'une ontologie universelle pour représenter leurs données. Dans le cas où plusieurs ontologies sont utilisées dans un système, nous parlons d'hétérogénéité sémantique. L'hétérogénéité sémantique est un frein à l'interopérabilité car les requêtes émises par les pairs peuvent être incomprises par les autres pairs.

Il existe des méthodes qui permettent d'identifier des correspondances entre les concepts de deux ontologies [14, 15, 16] nous citions par exemple l'alignement d'ontologies. Bien que ces participants peuvent se communiquer en utilisant des ontologies différentes ils peuvent utiliser ces correspondances.

Toutefois, dans les systèmes P2P, il se peut qu'un pair ne reçoive aucune réponse à une requête soit parce qu'il n'existe pas suffisamment de correspondances entre son ontologie et celles des pairs qui la reçoivent, ou soit parce que les pairs capables de la comprendre ne sont pas contactés pour la traiter.

L'objectif de ce travail est de concevoir un système permettant à des participants de partager des documents bien qu'ils puissent utiliser des ontologies qui sont regroupées selon leur hétérogénéité dans le systèmeP2P pour obtenir des résultats pertinentes. Nous nous intéressons plus précisément à la recherche d'information car elle représente une fonctionnalité centrale dans le partage de données. Nous voulons que le système proposé soit tolérant aux pannes (cela exclut donc toute centralisation de données dans le système), qu'il passe à l'échelle, qu'il garantisse l'autonomie des participants, et qu'il permette aux participants de rejoindre ou quitter le système à tout moment.

Ce travail est organisé de trois chapitres après l'introduction général nous présentons un état de l'art concernant les différents domaines sur lesquels portent ce travail :

Le premier chapitre est un état de l'art concernant les systèmes P2P, leurs caractéristiques, leurs classes et le type de requêtes en environnement P2P

Le deuxième chapitre concerne les ontologies et la recherche d'informations, leurs composants, leur hétérogénéité, l'alignement d'ontologies et enfin la recherche d'information et ces modèles.

Le troisième chapitre détaille notre contribution et concerne l'évaluation des performances des méthodes proposées au sein de notre mémoire. Nous expliquons les hypothèses prises en compte, les jeux de données utilisées et les méthodologies d'évaluations. Nous montrons également les résultats obtenus et nous commentons ces résultats.

Enfin, une conclusion va résumer l'essentiel de notre travail et présentation de quelques perspectives et questions de recherche ouvertes dans ce cadre.

Etat de l’art

Chapitre I

Systèmes P2P et Partage de données

1. Introduction

Le terme « pair-à-pair » devient très populaire ces dernières années. Grace à ses avantages le système p2p représente une nouvelle façon de partager de ressources via internet comme le partage de fichiers, partage de capacité de calcul et l'échange de messages instantanés. Aujourd'hui, on parle sérieusement du p2p comme un modèle de communication capable de changer radicalement certaines approches de l'informatique en réseau. Le domaine le plus important est le partage de données à une granularité fine et des requêtes de haut niveau. A cette étape les deux problèmes les plus intéressants sont : trouver les ressources de données pertinentes et la génération d'un plan d'exécution proche de l'optimal.

Dans ce chapitre nous présentons une synthèse des systèmes p2p, de leurs caractéristiques et leurs classes.

2. Approches de partage de données en environnements distribués

Il existe trois approches principales utilisées pour le partage de données en environnements distribués [17].

2.1 L'approche de SGBD réparti

Afin de donner une idée sur comment une requête est évaluée selon cette approche, nous prenons le cas des bases de données relationnelles qui sont largement utilisées ces jours-ci. Un SGBD réparti évalue une requête, écrite selon le schéma global du système, en quatre phases :

a) Décomposition de requêtes en requêtes algébriques;
b) Localisation de fragments (chaque relation est fragmentée sur plusieurs sites);
c) Optimisation globale
d) Optimisation locale et exécution.

Les trois premières phases sont effectuées sur un site centralisé qui utilise des informations stockées dans le catalogue global du système. Cependant, la quatrième phase est effectuée sur des nœuds désignés durant la phase de l'optimisation globale. Dans les SGBD répartis, de plus du schéma global, le catalogue global stocke des informations concernant les données partagées. Il contient aussi des informations sur les placements des fragments et sur leurs caractères physiques. Par conséquent, ce catalogue joue un rôle essentiel dans le traitement de requêtes surtout lors de la localisation de fragments et de l'optimisation globale.

2.2 L'approche de l'entrepôt de données

L'approche de l'entrepôt de données consiste à remplacer les sources de données par

une source unique (appelée dépôt de données). Toutes les requêtes doivent être exécutées sur le site du dépôt de données. Cette approche offre une bonne performance en termes de temps de réponse car l'exécution de requêtes ne prend pas en compte les communications inter-sites. Cependant, la mise à jour de données stockées dans le dépôt de données est typiquement complexe et coûteuse surtout lorsque les sources de données sont instables et nombreuses. De plus, l'exécution de requêtes sur un seul site peut créer un goulet d'étranglement particulièrement lorsque le nombre de sources de données est élevé. On peut ajouter aussi que dans certains cas, on ne peut pas dupliquer des données stockées sur certaines sources à cause des raisons de sécurité, ou des spécifications particulières de ces sources.

2.3 L'approche de médiation de données

Au lieu de déplacer les données, cette approche permet d'accéder directement aux données sur leurs sources en offrant une intégration virtuelle de données. Un schéma global représentant toutes les données partagées dans le système. L'architecture classique de l'intégration virtuelle de données est l'architecture médiateur/adaptateur. Des règles de correspondances entre le schéma global et les schémas locaux des adaptateurs sont créées et stockées d'une façon centralisée. Le médiateur offre un point d'accès unique et uniforme aux sources de données. Il offre aussi une interface permettant d'interroger les sources de données et il contient des informations concernant l'intégration de données. L'adaptateur est un composant reliant une source de données au médiateur. Il rend l'implémentation interne de cette source transparente au médiateur. Il offre des informations concernant la source de données au médiateur en termes de schéma du médiateur. Il peut accepter d'exécuter de sous- requêtes venant du médiateur.
L'évaluation d'une requête dans les systèmes de médiation de données est effectuée en quatre phases:

a) reformulation de requête en termes des schémas locaux des adaptateurs.
b) décomposition de la requête reformulée en sous-requêtes afin de les envoyer aux adaptateurs (étant donné qu'il y a plusieurs méthodes pour décomposer une requête, cette phase inclut une étape d'optimisation distribuée).
c) exécution des sous-requêtes sur les adaptateurs
d) envoi des résultats des sous-requêtes au médiateur qui continue à exécuter les étapes restées et qui délivre enfin un résultat complet.

Parmi les inconvénients de ces approches de partage de données en environnement distribués on trouvent : la limitation du point de vue de passage à la grande-échelle (gestion des centaines de milliers de sources de données) et du point de vue de la centralisation de contrôle et des informations. Pour cette raison, est apparue une nouvelle approche utilisant les systèmes P2P pour partager les données dans un environnement distribué à grande-échelle.

3. Systèmes Pair à Pair (P2P)

Les systèmes P2P offrent une opportunité importante pour qu'un grand nombre

(centaines de milliers) de nœuds se coopèrent l'un avec les autres afin de partager leurs ressources via un réseau largement distribué (i.e. l'Internet). Plus le nombre de ressources dans un système P2P est élevé, plus la puissance de calcul et la capacité de stockage ont des valeurs importantes. Cet avantage permet aux systèmes P2P d'exécuter de tâches complexes avec un coût relativement faible sans avoir besoin d'installer des gigantesques serveurs. Au sein d'un système P2P, les nœuds jouent des rôles équivalents. Chaque nœud joue le rôle d'un : client lorsqu'il consomme de ressources disponibles, serveur lorsqu'il offert des ressources, routeur lorsqu'il propage de requêtes reçues aux autres nœuds dans le système et hôte de source de données lorsqu'il partage ses propres données avec d'autres nœuds.

En cours de la recherche bibliographique de ce mémoire, nous avons trouvées plusieurs définitions au réseau P2P qui sont proposées par différents acteurs de la communauté P2P, nous citons trois parmi eux :

« Peer-to-peer is a class of applications that take advantage of resources (storage, cycles,content, human presence) available at the edges of the Internet. Because accessing these decentralized resources means operating in an environment of unstable connectivity and unpredictable IP addresses, peer-to-peer nodes must operate outside the DNS and have significant or total autonomy of central servers » [18] .

« A distributed network architecture may be called a Peer-to-Peer network, if the participants share a part of their own hardware resources (processing power, storage capacity, network link capacity, printers, . . .). These shared resources are necessery to provide the Service and content offered by the network. They are accessible by other peers directly, without passing intermediary entities. The participants of such a network are thus resource providers as well as resource requestors » [18].

« Peer-to-peer systems are distributed systems consisting of interconnected nodes able to self-organize into network topologies with the purpose of sharing resources such as content, CPU cycles, storage and bandwidth, capable of adapting to failures and accommodating transient populations of nodes while maintaining acceptable connectivity and performance, without requiring the intermediation or support of a global centralized server or authority » [18].

Ces définitions différentes introduisent plusieurs concepts généraux qui sont :

- Le partage des ressources (définitions 1, 2 et 3) .
- Le caractère dynamique des participants (définitions 1 et 3)
- La capacité à s'auto-organiser (définitions 1 et 3) .
- L'absence d'élément central (définitions 1, 2 et 3).

De notre côté, la définition du modèle P2P que nous adoptons fait l'abstraction de ces concepts:

« Le Système P2P est une architecture d'applications distribuées qui partitionne les tâches ou les charges de travail entre les nœuds. Ces derniers sont équi-privilégiés et participent de façon équivalente dans l'application. »

3.1 Les modèles d'architecture des réseaux Pair-à-Pair

Plusieurs modèles d'architecture P2P existent. D'ordinaire, c'est le degré de décentralisation qui est retenu pour leur classification, puisque c'est là l'idée de base du P2P.

Ce degré de décentralisation de l'architecture caractérise aussi l'index de référence des ressources.

La première génération des applications P2P était à architecture centralisée. Une deuxième génération à architecture décentralisée non structurée a vite vu le jour, suivie par une troisième génération d'applications à architecture hybride.

Au niveau de l'infrastructure P2P, des protocoles et des architectures décentralisées structurées se sont développées en parallèle aux applications P2P, à partir de la deuxième génération.

Nous distinguons donc trois familles d'architectures P2P (figure 1) :

- L'architecture centralisée.
- L'architecture décentralisée, qui peut être structurée ou non structurée.
- L'architecture hybride.

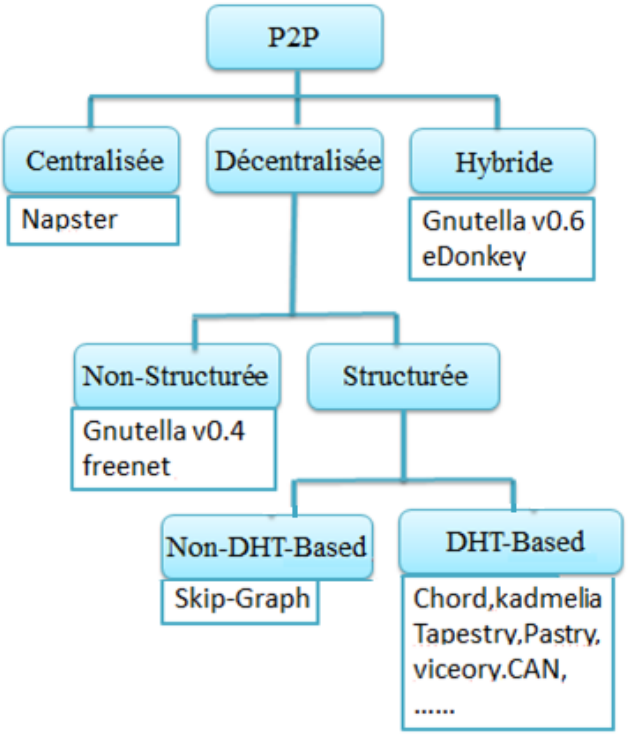

Figure 1 : *Les modèles d'architecture P2P.*

3.1.1 L'architecture centralisée

L'architecture centralisée est caractérisée par un index central pour repérer les ressources. Cet index reçoit tous les messages de contrôle et toutes les requêtes, mais par la suite, l'échange se fait directement entre les pairs concernés. C'est ce qui distingue principalement cette architecture d'une architecture traditionnelle de type client-serveur.

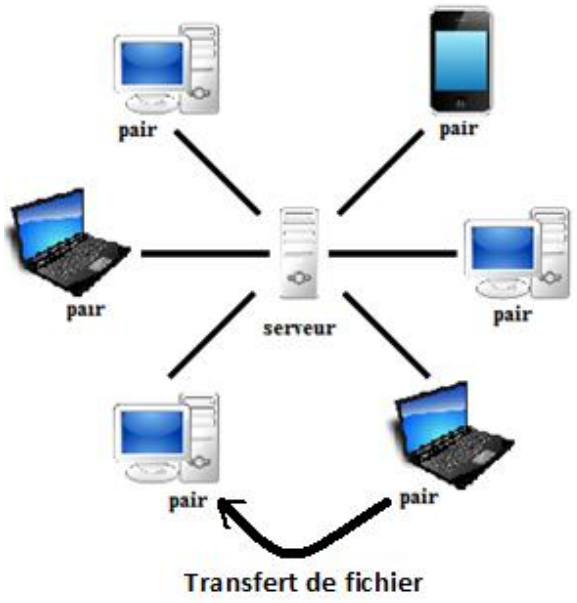

Figure 2 : *Architecture P2P centralisée*

Ce serveur centralisé ne dispose pas des fichiers. Il contient principalement deux types d'informations : celles sur le fichier (nom, taille, ...), et celles sur l'utilisateur (nom utilisé, IP, nombre de fichiers, type de connexion, ...).

Comme toute architecture centralisée, cette architecture facilite la gestion des différentes ressources de l'index et offre une bonne performance de découverte des ressources (le coût de localisation des ressources est faible), ainsi que Le confort d'utilisation peut être amélioré, puisqu'il n'y a pas de soucis de connexion au bon serveur. Cependant la centralisation représente un goulot d'étranglement au niveau de l'index, tant par la charge de son utilisation que de sa mise à jour. De plus, au niveau de la sécurité, l'index est le *talon d'Achille* de ce type d'architecture et aussi aucun anonymat n'est possible, puisque chaque utilisateur est identifié sur le serveur. Alors ce type est sensible au partitionnement du réseau et aux attaques.

L'exemple type d'une telle architecture est Napster (1999-2002) qui a déclenché le phénomène P2P. Il a aussi été le modèle le plus spectaculaire de la réussite technologique du P2P.

3.1.2 L'architecture décentralisée

Dans l'architecture P2P décentralisée, tous les pairs sont fonctionnellement parfaitement équivalents. C'est un modèle P2P pur (Figure 3) Cette architecture est caractérisée par une décentralisation de l'index, qui devient local à chaque pair, faisant effet de table de routage. La

décentralisation rend le système autonome et répartit la charge équitablement. L'architecture décentralisée est donc plus robuste que l'architecture centralisée, mais le temps de découverte des ressources est évidemment plus long.

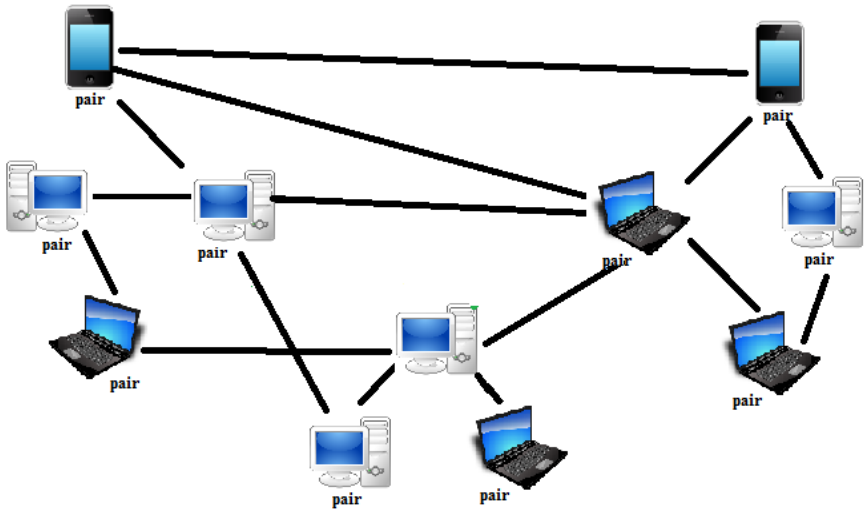

Figure 3 : *Architecture P2P décentralisée*

On distingue dans ce type d'architecture deux sous catégories :

- L'architecture décentralisée non structurée
- L'architecture décentralisée structurée.

3.1.2.1 Architecture décentralisée non structurée

Une architecture P2P décentralisée [17] est dite non structurée lorsqu'aucune contrainte topologique n'existe.

Un système P2P à architecture décentralisée est aussi dit : système P2P pur. Aucune connaissance de la topologie n'est donc disponible au préalable et chaque pair dans le réseau est autonome. La maintenance et la mise à jour des connexions se font par sondage périodique du voisinage, et la découverte des ressources se fait par une technique d'inondation associant un champ TTL (Time To Live) à chaque message de recherche envoyé, afin de comptabiliser le nombre de retransmissions restantes.

L'architecture décentralisée et non structurée permet de pallier les points faibles de la centralisation. Elle est simple et flexible, et ne requiert pas des requêtes exactes ou des demandes de recherches très précises. Néanmoins, cette architecture ne peut pas être gérée (par un opérateur de réseau ou fournisseur de services). Elle présente aussi certains inconvénients résultant de l'utilisation de la technique d'inondation et du TTL :

- La génération d'un nombre important de messages, dont découlent :
  - une consommation importante de la bande passante et donc une consommation des ressources du réseau.
  - la visite des pairs par des messages (requêtes ou réponses) non sollicités.
- l'absence de garantie de résultat et de connectivité.

- un passage à l'échelle assez limité.
-

L'exemple type d'une architecture P2P décentralisée non structurée est Gnutella dans sa version initiale, Le TTL y était typiquement fixé à 7. De ce fait, une mise hors réseau (panne ou déconnexion) d'un nombre aléatoire de pairs scinderait le réseau en deux.

Gnutella est le premier réseau pur pair-à-pair, il succède Napster et son architecture centralisée jugée vulnérable. Gnutella est un protocole ouvert décentralisé pour des recherches distribuées sur un ensemble de pairs non hiérarchisés. Tous les utilisateurs sont des « servent » c'est-à-dire qu'ils jouent le rôle de clients et de serveurs en même temps. A n'importe quel moment les pairs peuvent se connecter et intégrer le réseau selon certaines règles, ils n'ont aucune connaissance de la topologie ou de la localisation des données, c'est pour cette raison qu'un index est créé en local au fur et à mesure. Lorsqu'un pair effectue une requête, elle est propagée à ses voisins qui la propagent à leur tour aux leurs. Ce mécanisme qui se base sur la technique de broadcast est appelée « *flood* », cette approche passe mal à l'échelle (risque de saturation du réseau). Pour limiter cette surcharge, la requête se voit attribuer une durée de vie déterminée TTL décrémentée à chaque passage par un pair [17].

3.1.2.2 Architecture décentralisée structurée

On parle de réseau décentralisé structuré lorsque la topologie du réseau est connue et que l'emplacement des données est choisi dans le but d'optimiser les recherches. Les fichiers sont donc stockés à des emplacements spécifiques. Les pairs sont reliés entre eux selon des règles bien définies suivant un algorithme de recherche déterministe de telle sorte que pour une source donnée, correspond un endroit fixé au préalable. L'emplacement est choisi de telle sorte qu'il soit le plus simple d'accès aux pairs. Ceci permet à n'importe quel pair de se joindre au réseau ou de le quitter (auto-organisé), Un autre avantage de cette architecture est la garantie qu'elle offre sur les résultats des requêtes. Cette section sera détaillée dans le chapitre suivant.

3.1.3 Architecture hybride

L'architecture hybride est une architecture décentralisée dont chaque nœud est l'élément central d'une architecture centralisée. Ces éléments centraux sont des super-pairs par rapport à l'architecture globale.

Pour une homogénéité entre les super-pairs, c'est souvent le critère de la bande passante – identifiée suivant le type de connexion (e.g. par modem ou haut débit) qui est pris en compte. Mais d'autres critères peuvent aussi entrer en jeu, comme la puissance de calcul, l'espace de stockage, etc. Un nœud peut, avec le temps, à plusieurs reprises gagner et perdre le statut de super-pair.

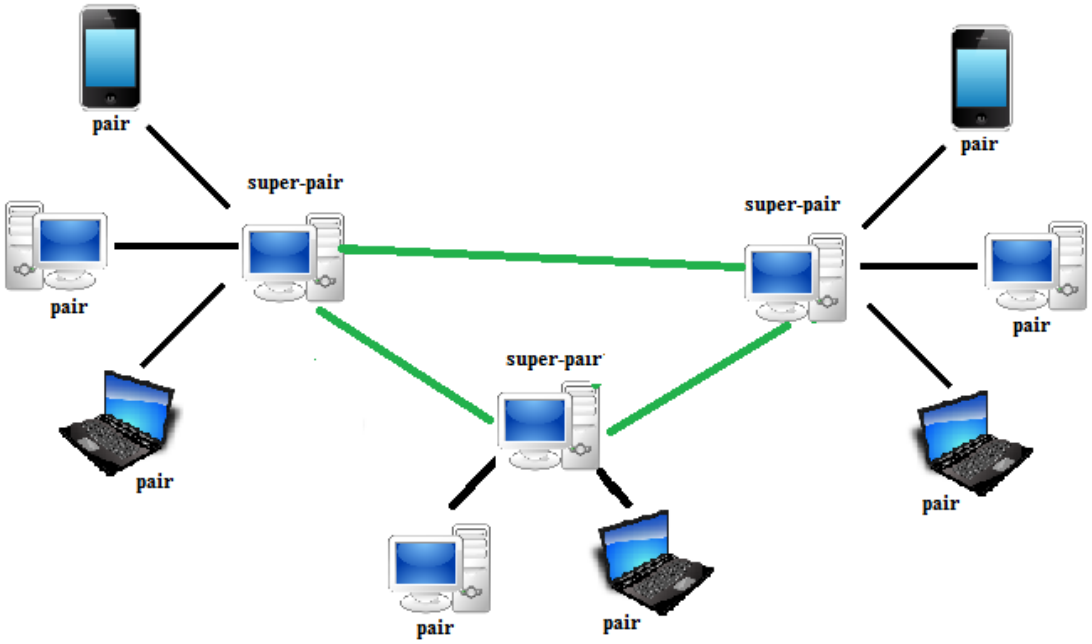

Figure 4 : *Architecture P2P hybride*

L'architecture hybride tire avantage simultanément de l'architecture centralisée et de l'architecture décentralisée. Le nombre de pairs mis en jeu dans la découverte des ressources est aussi réduit, et avec lui, le trafic global entre les pairs. Ceci permet une économie de bande passante et facilite le passage à l'échelle.

Un super-pair a cependant les mêmes faiblesses que l'index d'une architecture P2P centralisée. Une amélioration possible est d'assurer une redondance en choisissant, pour un même sous-groupe, un ou plusieurs partenaires au super-pair. L'architecture sera dite à super-pairs redondants.

Des supers-pairs partenaires ont les mêmes connexions aux autres pairs et possèdent des index identiques de toutes leurs ressources. Pour une répartition de charge équitable entre les super-pairs partenaires, Chaque Super-Pair réfère une recherche aux autres Super-Pairs. La recherche est donc plus rapide car elle ne se fait que dans les fichiers indexés par le Super-Pair auquel on est connecté, les pairs d'un sous-groupe envoient leurs requêtes selon le principe d'ordonnancement *round Robin.*

Voici quelques exemples d'applications basées sur un réseau P2P hybride : Gnutella (à partir de la version 0.6) , FastTrack[24] , eDonkey/eMule , Skype .

Il est à noter que les architectures hybrides et les architectures centralisées sont parfois regroupées au sein d'une même classe dite architecture à serveurs.

3.2 Caractéristiques des réseaux pair-a-pair

Le modèle P2P apporte une nouvelle manière de concevoir et mettre en œuvre les applications réseaux. De par ses caractéristiques intrinsèques, il permet de résoudre certains

problèmes posés par le modèle client-serveur, comme par exemple, la limite du passage à l'échelle ou le coût important pour l'achat et la maintenance des équipements. Toutefois, il en induit d'autres, liés par exemple à la sécurité ou la pérennité des ressources. Le tableau 1.1 met en évidence plusieurs différences fondamentales entre les modèles client-serveur et P2P. D'une manière générale, on constate que le modèle client-serveur est associé à un fonctionnement, un environnement et des ressources statiques et connues, alors que le modèle P2P est lié au concept de dynamicité. Dans cette section, nous passons en revue l'ensemble des caractéristiques que présente le modèle P2P ainsi que les différentes propriétés qu'elles lui confèrent.

Critère	***Modèle Client-Serveur***	***Modèle P2P***
Gestion	Supervisé	Auto-organisé
Présence	Permanente	Ad Hoc
Accès aux ressources	Recherche	Découverte
Organisation	Hiérarchique	Distribuée, Equitable
Mobilité	Statique	Mobile
Disponibilité	Dépendante du serveur	Indépendante des pairs
Nommage	Reposant sur le DNS	Indépendant
Modèle de programmation	RPC	Asynchrone

Tableau 1.1- *Extrait de [19]. Comparaison des infrastructures client-serveur et P2P*

3.2.1 Décentralisation

La décentralisation est la caractéristique du modèle P2P. Elle s'applique à différentes fonctions et aux trois sous-modèles. Dans le cas du modèle P2P centralisé, seules les ressources sont décentralisées mais les mécanismes de recherche et de localisation restent centralisés. Par contre, dans le cas du modèle pur, tout est décentralisé, des ressources aux mécanismes de découverte, localisation, sécurité, routage, . . . Quelle que soit sa nature, cette décentralisation confère au modèle P2P un ensemble de propriétés que nous détaillons maintenant.

3.2.1.1 L'´équilibre de la charge et du trafic

La première conséquence induite par la décentralisation sur les applications P2P, concerne l'équilibre de la charge et du trafic. Dans le cas du modèle client-serveur, le serveur de l'application concentre tout : les données, l'ensemble des mécanismes qui assurent l'accès à celles-ci et leur manipulation, mais aussi le trafic généré sur le réseau qui héberge le serveur. Au contraire, le modèle P2P permet de repartir et équilibrer au mieux la charge induite par l'exécution du service ainsi que le trafic généré sur le réseau.

3.2.1.2 Le passage à l'échelle

Le modèle P2P centralisé possède lui aussi un très bon passage à l'échelle car il ne centralise pas les ressources mais un index de références. Ainsi la charge et le trafic qui lui sont attribués sont acceptables et permettent un bon passage à l'échelle des applications qui reposent

sur ce modèle. Les deux exemples-phares qui ont révèle cette bonne propriété sont Napster , qui permit à plusieurs dizaines de milliers d'usagers d'utiliser simultanément son service, et Seti@Home , une application de calcul distribué qui compte plusieurs milliers d'utilisateurs simultanés.

3.2.2 La répartition des coûts

Le modèle P2P permet de réduire fortement le coût par l'utilisation de machines situées en bordure de l'Internet. Ces machines appartiennent toutes à des propriétaires différents qui peuvent être par exemple des particuliers, des universités, des entreprises ou des administrations et qui sont toutes achetées, mises en œuvre et maintenues par ces différentes organisations. L'utilisation du modèle P2P permet donc à un fournisseur de service de réduire fortement les coûts d'équipements et, au-delà de l'aspect financier, de concentrer son action sur l'aspect logiciel de l'infrastructure qu'il propose.

3.2.3 La tolérance aux fautes

Dans l'architecture client-serveur la disponibilité d'un service repose intégralement sur celle du serveur. Si celui-ci s'écroule, le service qu'il fournit devient indisponible. Dans un contexte P2P, il n'existe potentiellement aucun point central de faute. Si un pair disparait, le service continuera d'être fourni par ceux qui restent. En d'autres termes, la disponibilité d'un service n'est plus liée aux pairs mais à la communauté de pairs qui le fournissent.

3.2.4 Auto-organisation

L'absence d'élément central dans les applications P2P nécessite la mise en place de mécanisme d'auto-organisation qui permettent à une communauté de délivrer son service quels que soient les allers et venues des pairs et la disponibilité des ressources. Cette auto-organisation couvre plusieurs aspects qui peuvent être fonctionnels, communautaires et topologiques.

3.2.5 Connectivité Ad Hoc

Le modèle P2P se caractérise par une connectivité intermittente des pairs qui le composent. Cette nature Ad Hoc est principalement due à deux phénomènes. Le premier concerne le comportement des usagers qui utilisent les services P2P. Les pairs sont en effet exécutés dans un cadre de travail ou personnel sur des machines d'utilisateurs qui peuvent se connecter et se déconnecter de manière spontanée et donc imprévisible. Le second phénomène concerne la mobilité. Les ordinateurs portables munis d'interfaces de communication sans fil sont de plus en plus courants. Les usagers disposant de cette technologie ont souvent un comportement nomade, se trouvant certaines fois sur des sites qui permettent de se connecter à l'Internet, et d'autres fois pas. En outre, la manière dont ils atteignent une passerelle de connexion varie. Elle peut être directe ou n´nécessiter le routage des données à travers différents terminaux mobiles qui forment un réseau Ad Hoc.

3.2.6 Un réseau virtuel

Les pairs participant à un service P2P forment souvent un réseau virtuel, appelé overlay [20] construit au-dessus de la couche transport. Un exemple d'overlay est représenté sur la figure 3 . Généralement, un tel réseau virtuel présente une propriété de transparence qui s'applique à plusieurs points. Tout d'abord, il permet de faire abstraction des différences de nature des pairs.

Une communauté peut être composée de pairs dont les caractéristiques différentes sur les plans :

- matériel : avec par exemple des stations de travail, des téléphones mobiles, des assistants personnels, ou un cluster de machines.
- logiciel : au niveau des systèmes d'exploitation et langages de programmation.
- communication : avec l'utilisation de technologies et de piles de protocoles différentes.

La transparence s'applique aussi sur le routage effectué au niveau sous-jacent ; deux voisins de la topologie virtuelle ne le sont pas forcément physiquement ; ils peuvent être situés dans des espaces physiques et sur des réseaux différents. L'overlay rend ainsi transparent le routage effectué au niveau physique. Enfin, concernant les ressources, un overlay rend transparent leur accès, leur localisation et leur duplication. Lorsqu'un pair accède à une ressource, il ne sait pas si cette ressource est locale ou distante. Dans le cas où la ressource est distante, il n'a pas de connaissance de l'hôte qui l'héberge, et si elle est dupliquée, il ne sait pas à quel réplica il accède.

L'utilisation d'un overlay n´nécessite toutefois la mise en œuvre des mécanismes de nommage et routage dédiés. Les pairs ne sont donc plus représentés par leur adresse de niveau transport ou adresse physique mais par un identifiant défini dans le cadre de l'overlay. En outre, pour pouvoir découvrir et accéder à des ressources, des services de routage, calcul de routes, découverte et accès sont mis en place.

3.2.7 Anonymat

L'anonymat est une fonction qui permet de cacher son identité. Dans le cadre des systèmes distribués relatifs au partage de documents. L'anonymat est utilisé dans plusieurs applications qui voient dans le modèle P2P une infrastructure qui se prête particulièrement à cette fonction. Tout d'abord, Freenet utilise une forme de routage qui garantit l'anonymat du serveur, de l'auteur et du lecteur en ne permettant à aucun nœud de savoir quelle est la source et la destination d'une requête. Ensuite, FreeHaven et Publius mettent explicitement en œuvre des mécanismes d'anonymat qui ont pour rôle principal d'assurer la persistance et la disponibilité de documents dans un environnement soumis à la censure.

3.3 Avantages et limites des réseaux pair-à-pair

3.3.1 Avantages

Les architectures P2P présentent beaucoup d'avantages parmi eux :

- **Répartition de la charge** : Contrairement aux réseaux client/serveur où le serveur principal est chargé de gérer l'ensemble des activités du réseau, d'où un risque de saturation, dans un réseau pair-à-pair, ce sont tous les pairs qui contribuent à la gestion des différentes requêtes et des ressources disponibles.

- **Capacité de stockage décuplée** : Il est à savoir que dans un réseau à serveur central, toutes les données sont stockées au niveau de ce serveur, ce qui vient en opposition aux réseaux pair où bien que chaque pair n'offre qu'un petit espace disque (relativement au serveur) la capacité de stockage au sein du réseau est décuplée grâce à la contribution de tous les pairs.

- **Puissance de calcul** : Des études avancent que la puissance de calcul utilisée en moyenne par un utilisateur est de 20%, le processeur est donc sous exploité. Certaines applications pair-à-pair développées dans le domaine de la recherche (Seti@home) visent à se servir de cette puissance inutilisée pour réaliser des tâches réparties qu'un simple ordinateur ne pourrait accomplir en un temps raisonnable.

- **Réseaux très extensibles** : Une particularité des réseaux pair-à-pair est qu'ils sont de nature « Ad-hoc » c'est-à-dire que des nœuds peuvent apparaître et disparaître à tout moment, cette gestion dynamique des pairs facilite l'intégration de nouveaux pairs au sein du réseau.

- **Résistance aux pannes** : Dans un réseau pair-à-pair, les ressources sont réparties sur l'ensemble des utilisateurs connectés, la panne d'un pair ne peut donc pas altérer le fonctionnement du réseau.

- **Disponibilité accrue des ressources** : Comme les réseaux pair-à-pair sont extensibles, et que les pairs sont fournisseurs de ressources au sein du réseau, plus le nombre d'utilisateurs augmente, plus la disponibilité des ressources présentes augmente.

- **Diversité des chemins dans le réseau** : La topologie logique du réseau pair-à-pair étant maillée, plusieurs chemins de communication sont possibles entre chaque couple de pairs. De plus, les échanges se font plus rapidement étant plus directs.

- **Maintenance et coûts réduits** : Le serveur d'une architecture client/serveur reste très gourmand en matière de ressources, sa maintenance est très fastidieuse et son coût peut s'avérer très élevé. De ce fait, l'absence de celui-ci dans les architectures pair-à-pair rend ces réseaux peu coûteux.

- **Anonymat** : Certains algorithmes de routage ne permettent pas le pistage d'une requête, garantissant ainsi l'anonymat aux utilisateurs.

- **Meilleure exploitation de la bande passante** : Dans les réseaux à serveurs centraux, les goulots d'étranglements sont relativement fréquents au niveau de ces derniers, ce qui paralyse le réseau, ainsi l'absence d'organisme central dans les réseaux pair-à-pair facilite la circulation des flux, et augmente par conséquent l'utilisation de la bande passante.

3.3.2 Limites

Les réseaux pair-à-pair rencontrent de nombreux problèmes et présentent quelques inconvénients, on peut citer essentiellement :

- **Topologie instable** : Les pairs d'un réseau pair-à-pair étant dynamiques, ils peuvent apparaître et disparaître à tout moment, par conséquent les ressources aussi.

- **Sécurité** : Nous pouvons citer quelques exemples : Crackers, Virus, Distributed Deny of Service (ddoS), Confidentialité, Authentification.

- **Contenu trompeur** : Un des inconvénients majeurs des applications pair-à-pair est que les données circulent librement dans le système, sans vérification. Certains fichiers peuvent être corrompus : mauvaise qualité, défectueux, contenu non correspondant à l'intitulé.

- **QoS (Quality of Service)** : Bien que très utilisées dans d'autres domaines, beaucoup d'applications pair-à-pair sont employées de nos jours pour le téléchargement souvent illégal de fichiers, ce qui pousse certains fournisseurs à vouloir limiter au maximum les flux pair-à-pair au sein du réseau. La bande passante allouée aux applications pair-à-pair est donc considérablement réduite, rendant ainsi les échanges pair-à-pair très lents.
- **Loi du Wild Wild Web** : Les applications pair-à-pair sont devenues la cible de plusieurs organismes de protection des droits d'auteurs et lutte contre les contenus immoraux.

- **Régulation/Répression** : Certains modèles de réseaux pair-à-pair garantissent l'anonymat aux utilisateurs, il est donc plus difficile aux autorités d'appliquer les lois.

3.2.6 Comparaison

Dans cette section nous présentons une brève comparaison de différentes architectures du système p2p.

Les critères sur lesquels nous nous focalisons sont le passage à l'échelle, la dynamicité, la tolérance aux pannes ou aux attaques, l'autonomie des pairs (particulièrement le choix de la

représentation des données), l'expressivité des requêtes et l'efficacité des méthodes de routage de requêtes.

Dans le tableau 1, nous notons les systèmes p2p à index centralisé ne permettent pas le passage à l'échelle car le fait qu'un seul pair (ou serveur) Stocke l'index de toutes les données stockées dans le système peut emmener à l'effet de goulot d'étranglement .Surtout, le système est devenu sensible aux pannes concernant l'aspect centralisé et préviens les pairs comment ils peuvent représenter leurs données (ils sont contraints par le schéma de l'index central).L'architecture centralisée procure un service de recherche efficace, qui est pas du nombre de pairs du système.

Les systèmes p2p non structurés ont l'avantage de fournir un outil de recherche efficace .L'inconvénient est qu'il faut garder une topologie particulière et aussi les requêtes correspondant à des clés (ne sont pas des requêtes riches/complexes) et cela peut limiter l'autonomie des pairs.

Les systèmes hiérarchiques présentent certains avantages et inconvénients des systèmes non structurés et structurés, son utilisation convient de l'usage que l'on souhaite en faire.

	Central	Non-struct	Struct	Hiérarch
Passage à l'échelle	-	+	+	+
Dynamicité	+	+	-/+	-/+
Tolérance aux pannes	-	+	+	+
Autonomie	-	+	-/+	-/+
Expr. des requêtes	+	+	-/+	-/+
Rout. de requêtes	+	-	-/+	-/+

Table 1.2 – *Tableau comparatif des différentes architectures de systèmes P2P.*

4. Recherche d'informations des systèmes P2P :

La majorité des systèmes P2P existants dans la vie quotidienne sont des systèmes de partage de fichiers. Dans un tel environnement on trouve un nombre important (des milliers) de nœuds et un très grand nombre de fichiers. Etant donné le nom d'un fichier recherché par l'utilisateur, le système cherche les nœuds stockant ce fichier et envoie les adresses IPs de ces nœuds à l'utilisateur afin de choisir un nœud et de télécharger ensuite le fichier. Habituellement, le nom du fichier est connu par la majorité des utilisateurs dans le système. Ainsi, cette connaissance permet d'avoir une sémantique commune concernant les noms de fichiers partagés ce qui mène à un échange compréhensif de fichiers. C'est-à-dire, que lorsqu'un nœud répond à une requête, il y a une grande probabilité que ce nœud envoie le fichier souhaité par le nœud initialisant la requête. Par conséquent, les requêtes dans ce type de partage de données sont des requêtes "simples".

Elles consistent à chercher les adresses IPs des nœuds stockant le fichier demandé en se basant sur le nom du fichier. Si le fichier obtenu par ce type de traitement n'est pas celui demandé, dans ce cas-là le nœud initialisant la requête choisit un autre nœud afin de télécharger

le fichier demandé. Par conséquent, il faut exécuter une autre fois la requête ce qui nécessite de consommer plus des ressources disponibles dans le système. La phase la plus délicate dans le partage de fichiers est le routage de requêtes vers les nœuds stockant les fichiers demandés.

Les systèmes P2P offrent de "bonnes" techniques permettant d'effectuer un routage de requêtes relativement efficace en termes de consommation de la bande passante réseau et de temps nécessaire pour le routage.

Les approches utilisées pour le partage des données réparties sur l'Internet sont capables d'évaluer des requêtes plus complexes que celles soumises dans les systèmes P2P de partage de fichiers. Cependant, elles sont limitées du point de vue du nombre de sources de données partagées tout en garantissant une évaluation efficace de requêtes. Récemment, une nouvelle tendance est apparue pour dépasser ces limites. Les chercheurs dans le domaine de gestion de données essaient de combiner les techniques P2P et celles utilisées dans les bases de données. Cela permet aux systèmes P2P de partager de données ayant une granularité plus fine (i.e. un attribut atomique) et d'offrir la capacité de traiter des requêtes complexes (i.e. SQL). La difficulté de trouver une solution pour évaluer des requêtes complexes mène aux travaux de recherche abordant des types spécifiques de requêtes.

4.1 Types des requêtes évaluées en environnement P2P

Pour effectuer des recherches dans les systèmes P2P on peut utiliser plusieurs types de requêtes, parmi ces derniers on peut citer :

a) Requêtes d'intervalle: ce sont les requêtes par lesquelles l'utilisateur demande des données ayant des valeurs appartenant à un intervalle précis. Les auteurs de [20, 21] ont étudié ce type de requêtes en environnement P2P.
b) Requêtes meilleur- k : dans ce type de requête, l'utilisateur peut être satisfait des meilleur-k réponses trouvées dans le système. Ce type de requêtes attire beaucoup d'attention dans plusieurs domaines de l'informatique comme la recherche d'informations [22, 23], Monitoring de systèmes et de réseaux, bases de données multimédias [24, 25], analyse de données spatiales [26]. Récemment, ce type de requêtes est utilisé dans le partage de données dans les systèmes P2P [27].
c) Requêtes d'agrégation : ce sont des requêtes contenant un opérateur d'agrégation comme la requête suivante : Select Op_Agr(Col) From R Where conditon-selection ; Op-Agr est un opérateur d'agrégation comme : Sum, Count, AVG etc. Col est un attribut numérique de la relation R ou une expression contenant plusieurs attributs. La condition-sélection décide quels sont les tuples à être présentés dans l'agrégation.

d) Requêtes de jointure : ce sont de requêtes multi-relations multi-attributs dans lesquelles il y a un ou plusieurs opérateurs de jointure.

e) Requêtes complexes : ce sont des requêtes constituées des combinaisons des requêtes précédemment citées ci-dessus. e.g. requêtes SQL générales (sans aucune spécification).

Les données interrogées par ces types de requêtes sont souvent des données représentées par des schémas hétérogènes. Cette hétérogénéité est due à l'autonomie de nœuds qui caractérise les environnements P2P. Donc, au moment de l'évaluation de ces requêtes, nous avons besoin de faire un matching[1] de schémas permettant de créer des règles de correspondance et servant à traduire une requête donnée entre les différents schémas.

Par conséquent, le routage de requêtes et le matching de schémas sont des problèmes abordés par les chercheurs en bases de données pair-à-pair (P2P Databases) afin de localiser les sources des données désirées par une requête. Dans la section suivante, nous présentons une partie des principaux travaux de recherche effectués pour résoudre ces deux problèmes.

4.2 Routage de requêtes en environnement P2P

Nous remarquons que les méthodes de routage de requêtes dépendent beaucoup des topologies des systèmes P2P; autrement dit des types de systèmes P2P. Dans la suite, nous discutons une partie importante de ces méthodes au sein de chaque type des systèmes P2P.

4.2.1 Routage de requêtes dans les systèmes P2P non-structurés

Pour accéder aux données, les pairs émettent des requêtes qu'il envoie à leur voisinage, alors plusieurs méthodes de routage ont été proposées :

L'algorithme Breadth-First Search (BFS, algorithme de parcours en largeur) considère une valeur de TTL (Time-To-Live) afin d'inonder le système dans un certain rayon autour du pair initiateur de la requête

Les algorithmes Modified BFS et Intelligent BFS [28] ont été proposés pour limiter le nombre de messages échangés. La réduction du nombre de messages échangés impacte la qualité des résultats (certaines données pertinentes peuvent ne pas être retournées), et le coût de maintenance de statistiques sur les requêtes.

Dans l'algorithme Random Walks [29], le pair initiateur transmet sa requête à seulement k de ses voisins.

D'autres méthodes de routage ont été proposées, comme par exemple :

Iterative Deepening [30], Adaptative Probabilistic Search [31], Local Indices [32], Bloom Filter-based Indices [33].

L'algorithme Fully Distributed (FD) [34] est un algorithme top-k qui vise à obtenir les k données les plus pertinentes (selon une fonction de score) accessibles dans un certain rayon autour du pair initiateur. L'objectif de FD est de réduire le temps de réponse et le nombre de messages échangés.
L'algorithme est composé de quatre phases :
- Transmission de la requête
- Exécution locale de la requête
- Fusion et remontée des résultats
- Récupération des données

4.2.2 Routage de requêtes dans les systèmes P2P structurés

Dans les systèmes structurés, on utilise le système de clé pour accéder aux données, si un pair souhaite accéder à une donnée il doit générer sa clé (à partir du nom du fichier par exemple) et utiliser la primitive get pour récupérer la donnée chez le pair qui la stocke. Le routage de la requête utilise le fait que les pairs sont organisés.

4.2.3 Routage de requêtes dans les systèmes P2P super-pairs

Dans ces systèmes, les super-pairs accèdent aux données de la même manière que dans les systèmes non structurés ou structurés (selon la leur organisation) les autres pairs sont en contacte avec le super-pair auquel ils sont liés et dont le rôle est de leur fournir les données recherchées.

Conclusion

Les systèmes pair-à-pair sont très utiles et ont prouvé leur efficacité durant les dernières années. Ils sont structurés suivant différents modèles. Ils peuvent être centralisés, décentralisés ou hybrides. Parmi ces différences entre ces architectures on s'intéresse à la recherche des ressources c.-à-d. le routage dans les réseaux pair-à-pair.

Les domaines d'applications des pairs à pairs sont nombreux comme le partage de connaissances, travail collaboratif, etc. Même si le plus utilisé à nos jours est le partage de fichiers sans oublier que l'utilisation principale est le téléchargement qui est considérée comme du piratage

Dans le prochain chapitre on va étudier les ontologies et la recherché d'informations.

Chapitre II

Ontologies et recherche d'informations

1. Introduction

La notion ontologie a été considérée comme une voie prometteuse depuis son apparition à la fin des années 90 dans plusieurs axes de recherche. Le développement des ontologies ces dernières années a permis l'amélioration du processus d'ingénierie des connaissances et cela a aidé à résoudre plusieurs problèmes surtout la liaison entre la connaissance du domaine et le raisonnement.

Définition d'ontologie

D'après T. GRUBER, « une **ontologie** est une spécification explicite d'une conceptualisation ».

On peut brièvement dire qu'une ontologie est un produit d'ingénierie composé d'un ensemble de vocabulaire destiné à éclairer d'une façon précise et consensuelle en harmonie de voie à suivre (conceptualisation),à laquelle s'ajoute l'ensemble des hypothèses définissants le sens des termes utilisés qui a l'occasion démontre la relation décrite dans ce langage formel appelé spécification.

Les ontologies doivent permettre le partage et la réutilisation des connaissances qui servira comme outil de communication :
– entre les humains,
– entre les humains et les machines,
– entre les machines.

2. Composants des ontologies :

Une ontologie est composée de parties suivantes :

2.1 Les Concepts

Un objet matériel, une notion ou une idée peut se représenter comme un concept, On les appelle termes ou classe de l'ontologie. Les ontologies sont la base qui manipule la forme des objets. Ils conviennent aux concepts pertinents du domaine du problème occupé en fonction des objectifs qu'on se donne et de l'application aperçue pour l'ontologie.

2.2 Les Relations

Le domaine analysé se présente par des concepts qui devinent des interactions existantes. Ces relations se présentent comme une multitude de données qui se rattachent a un produit cartésien de plusieurs ensembles, la relation de combinaison, la relation d'instanciation, etc. Ces relations nous permettent de capturer, l'organisation ainsi que l'interaction entre les concepts, ce

qui nous permet de représenter une masse importante de structuration de la sémantique de l'ontologie.

2.3 Les Fonctions

Sont des cas spécifiques de relations aux termes desquels le nième élément de la relation est définit de façon particulière à partir des n-1 éléments précédents.

2.4 Les Axiomes

Sont des assertions véridiques qui permettent l'accès aux modèles choisis, à propos des Idées du domaine traduites par l'ontologie. Ils permettent de combiner des concepts, des relations et des fonctions pour instaurer des règles d'inférences et qui peuvent concourir, par exemple, dans la déduction, la définition des concepts et des relations, ou alors pour restreindre les valeurs des propriétés ou les arguments d'une relation.

2.5 Les Instances

C'est la façon pour élargir la définition extensionnelle de l'ontologie. Elles indiquent les éléments propres conduisant les idées concernant le domaine du problème.

3. Hétérogénéité : niveaux et sources

De manière générale, la conception d'une ontologie est une tâche complexe et critique car l'objectif est de « représenter le monde réel ». Il est raisonnable de penser que deux personnes peuvent avoir des points de vue différents sur le monde et sur la manière de le représenter.

3.1 Niveaux d'hétérogénéité

L'hétérogénéité entre deux ontologies peut apparaître à différents niveaux. Dans [25, 36], Euzenat et al. En identifient quatre :

– Le niveau syntaxique : concerne essentiellement le fait qu'une ontologie peut être modélisée dans différents langages comme OWL, KIF, DAML+OIL, etc. Les langages peuvent adopter des paradigmes différents (paradigme logique, paradigme objet, etc.) ou simplement utiliser des syntaxes différentes. Pour adresser ce problème, le W3C a proposé un certain nombre de standards.

– Le niveau terminologique : concerne les différences de nommage des entités. Ces différences sont principalement liées à la complexité des langues naturelles. Par exemple, il peut y avoir des cas de polysémie (deux entités identiques nommées de manières différentes), de synonymie (deux entités différentes nommées de la même manière), de multilinguisme (deux entités identiques nommées de la même manière, mais dans des langues différentes), etc.

– Le niveau conceptuel/sémantique : regroupe trois aspects : le point de vue duquel s'est placé le concepteur de l'ontologie (la perspective), le domaine du monde qu'il a cherché à représenter (la couverture), et le niveau de détail qu'il a voulu atteindre (la granularité). La perspective est issue de la difficulté à créer une ontologie avec un regard objectif.

– Le niveau sémiotique : intervient lors de l'utilisation des ontologies : un même concept peut être utilisé de plusieurs manières.

3.2 Sources de l'hétérogénéité sémantique

Une ontologie étant une représentation subjective de la réalité, il semble difficile (voire impossible) d'imaginer que les concepteurs d'ontologies puissent se mettre d'accord sur une représentation commune de la réalité, et que celle-ci soit adaptée aux besoins de tous les utilisateurs.

Smart et Engelbrecht soulignent dans [36] que les ontologies sont un élément central du web sémantique et que sa nature ouverte, distribuée et décentralisée tend à encourager la conception d'ontologies par des développeurs dont les objectifs sont variés. Hameed, Preece et Sleeman mettent en avant plusieurs raisons pour lesquelles il n'est pas envisageable de concevoir une ontologie universelle [38]. Premièrement, un certain nombre d'ontologies ont déjà été développées, donc même s'il était possible de concevoir une ontologie universelle, il faudrait réconcilier cette ontologie avec celles déjà utilisées.

Deuxièmement, la conception d'une ontologie est subjective et est guidée par l'utilisation qui en sera faite. Par exemple, une entreprise qui développe une ontologie n'a pas forcément pour but d'avoir une représentation fidèle du monde ; elle souhaite simplement représenter son domaine avec son propre point de vue. Enfin, dans l'hypothèse où un accord serait trouvé pour adopter une unique ontologie, il semble probable que celle-ci évolue au cours du temps car les connaissances dans certains domaines sont en perpétuelle évolution. Klein et Fenseil traitent ce problème spécifique dans [39].

Ces sources de disparité conduisent à la création d'ontologies pouvant différer à différents niveaux. Deux ontologies peuvent différer car elles modélisent différents domaines d'applications (couverture), à des niveaux de détail différents (granularité) et avec des points de vue différents (perspective)
[40].

4. Alignement d'ontologies

L'alignement d'ontologies est considère comme le processus qui permet d'amener deux ou plusieurs ontologies hétérogènes a un "accord mutuel" en les rendant ainsi consistantes et mutuellement cohérentes. L'alignement d'ontologies nécessite la transformation des ontologies

impliquées en procédant à l'élimination des entités non pertinentes et au rajout des entités manquantes.

En règle générale, un processus d'alignement fait intervenir plusieurs méthodes qui ont chacune pour objectif de trouver des correspondances entre deux ontologies. Chaque processus prend en entrée deux ontologies o et o′, et un alignement ai (éventuellement vide) et produit un alignement aj en sortie . Un ensemble de paramètres et de ressources sont également utilisées. Les méthodes d'alignement peuvent être locales ou globales [40].

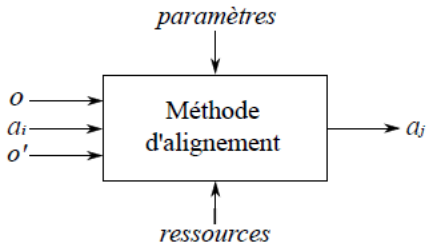

Figure 5 – Processus d'alignement d'ontologies.

4.1 Applications

Les alignements permettent à des agents utilisant différentes ontologies d'inter-opérer. Par exemple lorsque deux participants utilisent des ontologies différentes et qu'ils souhaitent s'échanger des requêtes, il est crucial qu'ils puissent traduire les requêtes qu'ils reçoivent ou qu'ils envoient.

Les alignements d'ontologies aident également à la conception de nouvelles ontologies. Les alignements sont également centraux pour l'intégration de données lorsque les sources de données sont représentées suivant différentes ontologies. Dans ce contexte, les requêtes sont transmises à un broker (un courtier) qui se charge de transmettre les requêtes à des médiateurs. Ces derniers utilisent des alignements pour traduire les requêtes exprimées par rapport une ontologie globale, afin qu'elles soient exprimées par rapport aux ontologies locales (qui sont utilisées pour représenter les données). Un autre exemple d'application pour laquelle les alignements d'ontologies sont utiles concerne la composition de services. En effet lorsque ceux-ci sont décrits par des ontologies différentes, il est intéressant de pouvoir faire correspondre les descriptions des services afin de savoir si deux services sont composables même s'ils ne sont pas décrits suivant la même ontologie.

Les alignements sont également la base de processus complexes appliqués sur les ontologies :

– fusion d'ontologies : création d'une ontologie à partir de deux ontologies [41],
– intégration d'ontologies : intégration d'une ontologie dans une autre,
– la traduction/transformation d'ontologies : expression d'une ontologie dans le "vocabulaire" d'une autre ontologie,

– réconciliation d'ontologies : transformation de deux ontologies dans le but de les uniformiser [42]. Ils sont également utilisés pour travailler sur les données :
– traduction de données : intégration des instances d'une ontologie dans une autre ontologie,
– intégration sémantique de données : possibilité d'interroger les données indépendamment de la manière dont elles sont représentées [43].

5. Mesures sémantiques

Dans cette section nous présentons un état de l'art concernant les mesures de similarité sémantique entre entités d'une même ontologie (similarité intra-ontologie), et sur les mesures de (dis) similarité entre deux ontologies.

5.1 Similarités intra-ontologie

Mesurer la similarité entre deux concepts d'une même ontologie peut servir dans plusieurs contextes. Par exemple, cela permet d'étendre des requêtes sémantiques en recherche d'information [44, 45] ou de désambiguïser des termes [46]. De nombreuses mesures ont été proposées dans la littérature. Elles permettent d'estimer la proximité sémantique entre concepts d'une même ontologie. Un certain nombre d'entre elles considèrent la structure hiérarchique des ontologies. C'est le cas de la mesure proposée par Rada [47]. Elle est définie par :

$$\mathrm{simRa}(\mathrm{c1}, \mathrm{c2}) = \frac{1}{1 \; + \; \mathrm{dist}(\mathrm{c1}, \mathrm{c2})}$$

où dist(c1, c2) correspond au nombre d'arcs qu'il faut traverser dans l'ontologie pour relier les concepts c1 et c2. Quelles que soient leurs positions dans l'ontologie, plus les concepts sont éloignés dans la hiérarchie, moins la similarité est importante.

Wu et Palmer ont proposé une mesure prenant en compte la position des concepts, c'est-à-dire la profondeur à laquelle ils se trouvent dans l'ontologie [48] :

$$\mathrm{simWP}(\mathrm{c1}, \mathrm{c2}) = \frac{2 * pro\ f(c)}{\mathrm{pro\ f\ (c1)} \; + \; \mathrm{pro\ f\ (c2)}}$$

Dans cette mesure, c désigne le plus petit ancêtre commun à c1 et c2 dans l'ontologie, et pro f (c) est une fonction donnant la profondeur du concept c dans la hiérarchie (la profondeur du concept racine vaut 0). La mesure de Wu et Palmer ne prend pas en compte la profondeur de l'ontologie. Pourtant celle-ci peut avoir une importance. La profondeur d'une ontologie o est définie par :

Prof (o) = maxc∈o prof (c).

Resnik a défini une mesure de similarité utilisant cette notion [49] :

$$\text{simRe}(c1, c2) = 2 \cdot \text{Prof}(o) - \text{dist}(c1, c2)$$

Des approches considèrent le contenu informationnel des concepts pour mesurer la similarité entre deux concepts. C'est le cas de la mesure proposée par Richardson, Smeaton et Murphy [50] :

$$\text{simRi}(c1, c2) = \max_{c \in C} \text{ci}(c)$$

Où C correspond à l'ensemble des ancêtres communs aux concepts c1 et c2, et ci(c) correspond au contenu informationnel associé au concept c. Plusieurs approches ont été proposées pour estimer le contenu informationnel d'un concept. Resnik propose d'utiliser un corpus [49],

Seco, Veale et Hayes proposent de se limiter à l'ontologie en considérant que le contenu informationnel d'un concept décroît avec le nombre de concepts qui le spécialisent [51].

Jiang et Conrath [52] et Lin [53] ont proposé des mesures combinant les deux approches (celle basée sur la structure de l'ontologie et celle basée sur le contenu informationnel des concepts). La mesure de Lin est définie par :

$$\text{simLin}(c1, c2) = \frac{2 \cdot \log(\text{P}(c))}{\log\big(p(c1)\big) + log(p(c2))}$$

Où c'est le plus petit ancêtre commun à c1 et c2, et P(c) correspond à la probabilité que le concept c appartienne à un corpus donné.

5.2 Similarités entre ontologies

Il est souvent utile de pouvoir évaluer la distance entre deux ontologies. Pour cela il faut disposer de mesures de (dis) similarité. Certaines de ces mesures permettent de décider s'il est raisonnable d'aligner deux ontologies. Cela permet de ne pas lancer un processus d'alignement coûteux et produisant peu de correspondances dans le cas où les ontologies sont trop différentes. Le calcul de la distance doit être rapide (en tout cas plus rapide que l'alignement lui-même).

Maëdche et Staab définissent plusieurs mesures de similarité entre ontologies [54]. L'une d'entre elles se concentre sur le niveau lexical. Elle considère les labels attribués aux concepts et est basée sur la distance d'édition (mesure de Levenshtein [55]). Ils proposent également une mesure de similarité à partir des hiérarchies de deux ontologies. Cette mesure permet de mesurer à quel point deux ontologies sont similaires au niveau de leurs structures. Enfin ils proposent une mesure prenant en considération les relations liant les concepts Dans [56], David et Euzenat présentent différentes mesures de (dis) similarité entre ontologies. Étant donnée une mesure de dissimilarité d entre entités de deux ontologies o et o′, les auteurs définissent la mesure de dissimilarité Average Linkage :

$$\Delta alo(o, o') = \frac{1}{|o| \times |o'|} \sum_{(e,e') \in o \times o'} б(e, e')$$

Où |o| correspond au nombre d'entités contenues dans l'ontologie o. La mesure *d* est une mesure de dissimilarité quelconque entre deux entités. Dans le même contexte, les auteurs présentent une mesure de Hausdorff qui mesure une distance entre deux ontologies :

$$\Delta_{Hauss}(o, o') = \max\left(\max_{e \in o} \min_{e' \in o'} \delta(e, e'), \max_{e' \in o'} \min_{e \in o} \delta(e, e')\right)$$

Enfin, les auteurs introduisent la notion de graphe d'alignement maximal de poids minimal (minimum weight maximum graph matching). Un tel graphe G ⊆ o × o′ vérifie que quel que soit un autre graphe

$$\sum_{\langle p,q' \rangle \in G} \delta(p, q) \leqslant \sum_{\langle p,q' \rangle \in G'} \delta(p, q')$$

La distance MWMGM (Minimum Weight Maximum Graph Matching) est définie par :

$$\Delta_{mwmgm}(o, o') = \frac{\sum_{\langle p,q' \rangle \in G} \delta(p, q') + \max(|o|, |o'|) - |G|}{\max(|o|, |o'|)}$$

Euzenat, Alloca, David et al. Présentent un ensemble de mesures de distance entre ontologies dans [57]. Certaines considèrent uniquement les labels assignés aux entités : la distance d'Hamming sur les noms *scn*, la similarité Common Name *scn*, etc. Une mesure similaire à la similarité
Common Name est définie sur les axiomes des ontologies : *scax*. En considérant la signification des ontologies, les auteurs présentent une mesure sémantique : la distance Ideal Semantic. Elle prend en compte l'ensemble des conséquences d'une ontologie. Étant donné que cet ensemble peut être infini, les auteurs proposent de considérer l'ensemble des conséquences d'une ontologie par rapport à une autre. Ils définissent ainsi la similarité Common Conséquence :

$$\sigma_{ccsq}(o, o') = \frac{|LCn(o, o') \cap LCn(o', o)|}{\max(|o|, |o'|)}$$

Où LCn(o, o′) = o ∩ Cn(o′). Les auteurs proposent également une mesure qui permet de se focaliser sur les concepts les plus importants des ontologies. Pour cela les travaux de Peroni, Motta et d'Aquin peuvent être considérés [64]. La fonction KC permet de retourner l'ensemble des n concepts clés d'une ontologie. La mesure de similarité est définie par :

$$\sigma_{ukc}(o,o') = \frac{|KC(o,n) \cap KC(o',n)|}{n}$$

Dans cette définition, KC (o, n) désigne l'ensemble des n concepts les plus importants. L'expression A ∩ B désigne ici l'ensemble des concepts de A et de B ayant le même label. Cette mesure est un raffinement de la mesure scn, et elle ne tient pas compte de l'ordre des concepts importants.

Les auteurs proposent une mesure permettant de prendre en compte cet aspect :

$$\sigma_{rkc}(o,o') = \frac{1}{n} \sum_{c \in KC(o,n)} \sigma_{kc}(KC(o,n), KC(o',n), c)$$

Où, étant donnés deux ensembles ordonnés de concepts S1 et S2 de cardinalité n, la mesure skc est définie par

$$\sigma_{kc}(S_1,S_2,c) = \begin{cases} 1 - \frac{1}{n}|rang(S_1,c) - rang(S_2,c)| & \text{si } c \in S_1 \cap S_2 \\ 0 & \text{sinon} \end{cases}$$

Contrairement à toutes les mesures présentées précédemment (qui sont définies dans l'espace des ontologies), certaines mesures ont été définies dans l'espace des alignements, c'est-à-dire qu'elles mesurent la distance entre deux ontologies en prenant en compte des alignements existants entre elles. Euzenat et al. Proposent un ensemble de mesures [63, 57]. Ils définissent par exemple une mesure de similarité de manière très simple :

$$\sigma_{ap}(o,o') = \begin{cases} 1 & \text{si } o = o' \\ 2/3 & \text{si } o \neq o' \wedge \mathcal{A}(o,o') \neq \emptyset \\ 1/3 & \text{si } o \neq o' \wedge \mathcal{A}(o,o') = \emptyset \wedge \mathcal{A}^*(o,o') \neq \emptyset \\ 0 & \text{sinon} \end{cases}$$

Où A (o, o′) désigne l'ensemble des alignements existants entre o et o′ dans l'espace d'alignement A, et A∗ (o, o′) désigne l'ensemble des chemins d'alignements entre o et o′ dans A. Une mesure considérant le plus court chemin d'alignements a également été proposée (ssap). La mesure Alignement Convergé (cov) se focalise sur le nombre d'entités d'une ontologie o préservées par un alignement a. Elle est définie par :

$$cov(o,o',a) = \frac{|\{e \in o : \exists \langle id,e,e',r,n\rangle \in a\}|}{|o|}$$

Une mesure de distinguabilité est également définie :

$$sep(o,o',a) = \frac{|\{e' \in o' : \exists \langle id,e,e',r,n\rangle \in a\}|}{|\{e \in o : \exists \langle id,e,e',r,n\rangle \in a\}|}$$

La mesure cov détermine à quel point les concepts d'une ontologie sont préservés par un alignement, alors que la mesure sep détermine à quel point les concepts sont distincts lorsqu'un alignement est appliqué. Ces deux mesures peuvent être combinées pour définir la mesure covdis. Étant donné un ensemble de chemins d'alignement A∗(o, o′) entre deux ontologies o et o′, la mesure Largest Covering Préservation (slcp) permet de déterminer à quel point les concepts d'une ontologie sont préservés sur ce chemin :

$$\sigma_{lcp}(o,o') = \max_{a \in \mathcal{A}^*(o,o')} covdis(o,o',a)$$

Enfin, la mesure de similarité Union Path Coverage (supc) considère l'union de différents chemins d'alignements entre deux ontologies, plutôt qu'un seul chemin. L'idée sous-jacente est qu'une requête propagée entre deux individus au travers d'un chemin d'alignement peut être décomposée et propagée parallèlement à travers différents chemins. La mesure de similarité est donc définie par :

$$\sigma_{upc}(o,o') = \frac{|\{e \in o : \exists a^* \in \mathcal{A}^*(o,o') \text{ tel que } e \text{ est préservé sur } a^*\}|}{|o|}$$

Dans [58], d'Aquin propose des mesures d'accord et de désaccord (agreement et désagrément) entre deux ontologies. Pour cela il considère le degré d'accord entre toutes les déclarations des ontologies. Par exemple la déclaration hc1, ra, c2i contenue dans l'ontologie o déclare que le concept c1 est lié au concept c2 via la relation ra. Si les concepts c1 et c2 sont alignés avec les concepts c′1 et c′2 de l'ontologie o′ et que o′ contient la déclaration hc′1, rb, c′2i alors la mesure d'accord (ou de désaccord) dépend de la proximité entre ra et rb :

$$agreement(o,o') = \frac{1}{|ST| + |ST'|} \sum_{st \in ST} agr(st,o') + \sum_{st' \in ST'} agr(st',o)$$

où ST est l'ensemble des déclarations contenues dans l'ontologie o, et agr(st, o′) mesure l'accord de la déclaration st dans l'ontologie o′

6. Recherche d'information dans les systèmes P2P

Généralement, les méthodes de recherche dans les systèmes P2P non-structurés sont classées en trois catégories : aveugle, informée ou sémantique.

6.1 Recherche aveugle

Lorsque les noeuds ne disposent pas d'information concernant la localisation des ressources dans le réseau, type de recherche aveugle se fait par un parcours partiel de l'espace de recherche. En effet, lorsqu'une requête est soumise à un noeud, elle est résolue sur celui-ci et se propage sur l'ensemble des noeuds voisins. Le processus de propagation s'arrête lorsqu'un certain seuil du nombre fixé de rebonds est atteint. Dans cette catégorie, nous trouvons les algorithmes suivants:

Gnutella [Gnutella, 2000]: l'algorithme original de Gnutella utilise la propagation de recherche en largeur d'abord (BFS : Breadth-first search) pour la découverte d'objets ; tous les noeuds accessibles sont contactés dans les limites de la valeur de durée de vie de la requête (TTL : Time-To-Live). Cette approche bien qu'elle parait simple, elle ne passe pas à l'échelle puisqu'elle produit une charge élevée quand il s'agit d'un nombre important de pairs ;

Recherche en largeur modifiée (BFS modifié) [Kalogeraki et al., 2002]: c'est une variation du schéma de propagation où les pairs choisissent uniquement un sous ensemble de leurs voisins pour leur envoyer la requête. L'avantage de cet algorithme est la réduction du nombre de messages échangés par comparaison à la méthode précédente, mais elle contacte toujours un nombre élevé de pairs. Marche aléatoire (Random Walks) [Lv et al., 2002]: dans cet algorithme, le pair initiateur envoie la requête à un nombre k de voisins choisis au hasard. La requête effectue une marche aléatoire dans le réseau des noeuds. L'avantage majeur de cet algorithme est la réduction significative des messages qu'il réalise. Cependant, son principal inconvénient essentiel est sa grande variation de performance.

D'autres protocoles de recherche agissant dans les réseaux P2P hiérarchiques (pairs /super-pairs) sont apparus. Par exemple, Guess [Daswani, et al., 2002] et Gnutella2 [Stokes, 2002] qui sont des améliorations du protocole de Gnutella. Ils se basent sur la notion de pairs et super-pairs. La solution proposée est de laisser à l'initiateur de la recherche le soin d'interroger successivement un ensemble de super-pairs jusqu'à ce qu'il obtienne un nombre de résultats satisfaisants.

Tous ces protocoles tendent à réduire la charge du réseau en diminuant le nombre de messages échangés. Ce flux peut être réduit avec l'utilisation d'une structure hiérarchique de type super-pair. Cependant, cette catégorie d'algorithmes de recherche reste très pauvre dans le sens où aucune sémantique n'est prise en compte lors du routage des requêtes.

6.2 Recherche informée

La recherche informée se base sur l'utilisation d'informations concernant la localisation des ressources qui peut être centralisée (c'est-à-dire qu'il existe un répertoire connu par tous les nœuds à l'instar de Napster) ou décentralisée (chaque noeud détient une partie de l'information). Le principe consiste à ajouter de la connaissance au contenu des noeuds pour permettre une propagation de requêtes non plus à l'aveugle, mais guidée par cette connaissance. Dans cette catégorie, nous trouvons les algorithmes suivants :

La recherche en largeur intelligente (Intelligent-BFS) [Kalogeraki et al., 2002] qui permet aux nœuds de stocker les couples des requêtes récemment traitées par leurs voisins.

la recherche probabiliste adaptative (APS : Adaptive Probabilistic Search) [Tsoumakos et al., 2003] permet à chaque nœud de maintenir un index local composé d'une entrée et d'une valeur. Les Indices locaux (Local indices) [Yang et al., 2002] permettent le stockage des fichiers dans tous les nœuds dans un rayon R fixé. Cette approche est très précise puisque chaque noeud contacté indexe plusieurs pairs. Avec cette technique le temps de traitement des messages est réduit puisque la requête n'est pas traitée par tous les nœuds.

Une autre approche utilise des connaissances locales pour l'amélioration des performances des systèmes P2P. Cette approche est basée sur la constitution des groupes de pairs ayant les mêmes intérêts [Nakauchi et al., 2004] ce qui est traduit par le fait que chaque pair doit savoir les 'bons pairs'. Le terme « bons pairs » désigne les pairs qui renvoient des réponses correctes à la demande d'un pair spécifique. Cette approche réduit le trafic puisqu' un grand nombre de fichiers correspondant à la cible de la requête peut être obtenu avec une faible valeur de TTL (Time-To-Live).

le Protocole de localisation de ressource distribuée (DRLP: Distributed Resource Location Protocol) [Menasce et al., 2002] permet aux noeuds n'ayant pas d'informations sur la localisation d'un fichier d'envoyer une requête à un sousensemble de leurs voisins. De même les noeuds ayant des informations concernant la localisation indexée contactent directement le noeud spécifié. Dans les réseaux qui évoluent rapidement, cette approche échoue et beaucoup de noeuds doivent faire une recherche aveugle. Dans un routage aveugle, le choix d'un pair se fait complètement au hasard tandis que dans un routage informé, le choix tombe uniquement sur les pairs pouvant bien répondre à la requête.

REMINDIN (Routing Enabled by Memorizing Information about Distributed Information) [Tempich et al., 2004] est fortement inspiré des métaphores sociales.

REMINDIN montre que l'exploitation de cette métaphore sociale permet d'améliorer la recherche d'information et ainsi les réponses aux requêtes. Pour cela, il propose une méthode pour router les requêtes. Cette méthode consiste à observer les pairs qui répondent avec succès aux requêtes, à mémoriser cette information et enfin à utiliser cette information pour sélectionner les pairs pertinents pour des demandes futures.

REMINDIN maintient une table de routage au niveau de chaque pair. Il est implanté au-dessus de SWAP (Semantic Web And Peer-to-Peer). Une requête dans SWAP est formée du triplet RDF composé du sujet, de l'objet et du prédicat.

Les deux approches citées précédemment se focalisent essentiellement sur une recherche aveugle ou basée sur des connaissances des réponses obtenues par des pairs pour router les requêtes. Dans ces deux approches la sémantique est absente. Elles ne tiennent pas en compte du contenu des données ou des schémas des pairs. Nous explorons dans le paragraphe suivant d'autres techniques de recherches d'information basées sur la sémantique. Nous montrons que l'utilisation de la sémantique dans ce contexte permet d'améliorer la recherche des pairs pertinents pour une requête donnée.

6.3 Recherche sémantique

Nous avons vu plus haut que dans les systèmes P2P non-structurés le routage de requêtes se fait essentiellement selon une des approches suivantes : à l'aide d'un serveur central (cas de Napster) ; par diffusion à tous les pairs voisins (cas de Gnutella) ou en sélectionnant des pairs voisins de façon aléatoire (cas de Gnutella2) ou informée. L'intégration de la sémantique dans

les systèmes P2P a pour objectif d'améliorer le routage de requêtes. L'idée de base est de remplacer le routage aléatoire ou informé par un routage guidé par la sémantique. Pour ce faire, plusieurs dimensions du problème ont été proposées dans [Defude, 2007]: quelle sémantique : il s'agit de savoir quelle sémantique utilisée pour orienter le processus de routage des requêtes. Ceci peut refléter le contenu ou l'intérêt des pairs ou l'intérêt des pairs à travers leurs requêtes déjà émises; quelle représentation de la sémantique : ceci concerne la façon de représenter la sémantique de façon structurée (par exemple à l'aide des ontologies) ou non structurée (connaissances sur les réponses des pairs ou sous forme d'une liste simple de concepts etc.) comment construire la sémantique : cette sémantique peut être construite manuellement (par exemple une ontologie existante) ou bien elle est obtenue de façon automatique (divers algorithmes); qu'est ce qui est partagé entre les pairs : les pairs peuvent partager la même ontologie ou des groupes de pairs peuvent partager des ontologies différentes; comment utiliser la sémantique : généralement la sémantique va être utilisée pour sélectionner le sous ensemble des pairs les plus « pertinents » pour une requête donnée. Cela peut aussi servir à organiser le réseau des pairs (classification des pairs selon leur contenu par exemple) ou bien à modifier les requêtes ; comment diffuser la sémantique : la connaissance obtenue au niveau d'un pair doit être diffusée aux autres pairs pour qu'ils puissent améliorer leur connaissance du réseau.

Cette diffusion peut être globale (à tous les pairs) ou partielle (à quelques-uns) et la fraîcheur est également importante.

De nombreux travaux ces dernières années ont visé à améliorer la fonction de recherche dans les systèmes non structurés. Les techniques de recherche d'informations dans les P2P sémantiques sont nécessaires pour améliorer l'efficacité et l'évolutivité des techniques de recherche d'informations et de partage de ressources dans les systèmes P2P. En ce sens, la notion de P2P sémantique est introduite afin de mettre l'accent sur la nécessité des liens sémantiques entre les pairs afin de maintenir des informations sur les noeuds ayant des contenus similaires.

6.4 Profils et filtrage

Le profil représente l'ensemble des centres d'intérêt, des préférences, des connaissances ou des habitudes de l'utilisateur [Bouzeghoub et al., 2005]. L'ensemble des connaissances, nécessaires à une évaluation des requêtes, représente un profil nécessaire à la production d'une information pertinente adaptée à chaque utilisateur. Il ne faut pas mélanger entre la notion de profil et la notion de requête. Un profil est un modèle personnalisé d'accès à l'information alors qu'une requête est une demande pour satisfaire l'utilisateur tout en tenant compte de son profil.

Il existe plusieurs types de filtrage d'informations afin d'obtenir l'information appropriée :

6.4.1 Filtrage basé sur le contenu

[Lang et al., 1995] [Lieberman et al., 1995] [Pazzani et al., 1997] ***:*** Le choix des documents proposés est basé sur une comparaison des thèmes abordés dans les documents par rapport aux thèmes qui intéressent l'utilisateur. C'est un système de recherche d'information qui a le rôle d'un filtre permanent entre le profil (sorte de requête à long terme et évolutive) et le flot de documents entrants (sorte de corpus évolutif). Deux fonctionnalités centrales résultent: tout

d'abord, la sélection des documents pertinents vis-à-vis du profil et ensuite la mise à jour du profil fourni par l'utilisateur sur les documents qu'il a reçu. Cette mise à jour se fait par l'intégration des thèmes abordés dans les documents pertinents.

6.4.2 Filtrage collaboratif

Il signifie que tout utilisateur recherchant l'information peut utiliser ce que d'autres personnes ont déjà trouvé et évalué [Resnick et al., 1997] [Berrut et al., 2003]. Donc, tout va vers l'évaluation des documents par les utilisateurs.

Chaque utilisateur d'un système de filtrage collaboratif aura un ensemble de proches voisins, qui suivant leurs appréciations, l'utilisateur recevra un document. Les utilisateurs peuvent fournir des jugements sur des documents de leur choix, dans les 32 systèmes de filtrage collaboratif, exprimés sous forme d'une note. L'objectif d'un système est alors de comparer les différents goûts des utilisateurs afin de prédire les notes qui n'ont pas été fournies au système.

7. Modèles de recherche d'information

Un modèle de recherche d'information définit de quelle manière les documents et les requêtes doivent être représentées ainsi que la manière d'estimer la pertinence entre un document et une requête. La représentation des données (documents et requêtes) sert à :

(i) stocker les informations de manière synthétique,
(ii) faciliter la comparaison entre deux données (par ex. un document et une requête),
(iii) unifier les données (l'information contenue dans une page web ou une vidéo peut être représentée de la même façon). Cette représentation est obtenue par indexation. L'indexation est le processus qui consiste à extraire les informations importantes d'un document ou d'une requête. L'objectif est d'obtenir, pour chaque document et chaque requête, une description reflétant leur contenu. L'indexation peut être réalisée manuellement ou bien de manière automatique.

Dans le cas de l'indexation manuelle, il faut que celle-ci soit réalisée par des spécialistes du domaine traité dans les documents. En effet, pour obtenir une indexation de qualité, il faut que les termes de l'index soient choisis avec précision. Le coût d'une telle indexation est important, d'autant plus lorsque la collection de documents à indexer est grande.

L'indexation automatique est bien moins coûteuse, et fournit des résultats similaires à ceux obtenus manuellement. Baeza-Yates et Ribeiro-Neto assimilent l'indexation à un pré-traitement des documents [59]. Les phases clés du pré-traitement sont :

– L'élimination des mots vides de sens (stop words) : le, la, du, pour, etc. Ces derniers sont tellement communs qu'il ne sert à rien de les considérer. Ils sont généralement spécifiques à une langue et/ou à une collection de documents.
– La lemmatisation (ou racinisation) consiste à ramener les termes à une racine commune. Par exemple, les termes précieux, précieuses, ou préciosité, sont transformés en une racine simple : précieux.

– La sélection des termes de l'index consiste à choisir les termes porteurs de sens et significatifs. En général ils sont choisis parmi les noms car ceux-ci sont plus significatifs que les verbes ou les adjectifs.
– La pondération des termes se base souvent sur les facteurs tf (la fréquence d'apparition d'un terme dans un document) et idf (la fréquence d'apparition de ce même terme dans toute la collection considérée) qui permettent de considérer les pondérations locale et globale d'un terme.

Le calcul de pertinence (ou calcul de similarité) entre un document et une requête sert à déterminer si un document est pertinent pour une requête donnée. Le résultat de ce calcul peut être booléen (le document est pertinent ou il ne l'est pas), ou il peut être numérique (par ex. une valeur comprise entre 0 et 1). Dans ce dernier cas, la mesure de pertinence permet de classer les documents en fonction de leur pertinence.

7.1 Modèle booléen

7.1.1 Représentation des documents et des requêtes

Dans le modèle booléen, les documents sont représentés par des ensembles de termes issus d'un vocabulaire V = t1, ..., tn. La représentation d'un document d est l'ensemble des termes de V qui apparaissent dans d. Par exemple un document peut être représenté par l'ensemble {t1, t3, t8}.

Les requêtes sont exprimées suivant le formalisme de l'algèbre de Boole. Une requête est une expression booléenne dans laquelle les variables logiques sont les termes du vocabulaire V, et les opérateurs usuels sont considérés : conjonction (∧), disjonction (∨) et négation (¬). La requête t2 permet d'obtenir les documents dont la représentation contient le terme t2. La requête t2 ∨ (t3 ∧ t8) permet d'obtenir les documents dont la représentation contient soit le terme t2, soit à la fois les termes t3 et t8.

7.1.2 Calcul de pertinence

La mesure de pertinence d'un document d par rapport à une requête q est donnée par :

$$pert(d,q) = \begin{cases} 1 & \text{si } d \text{ satisfait } q \\ 0 & \text{sinon} \end{cases}$$

On dit alors simplement que d est pertinent pour q si pert(d, q) = 1.

Par exemple le document d = {t1, t3, t8} n'est pas pertinent pour la requête t2 (car t2 /∈ d). Par contre ce même document est pertinent pour la requête t2 ∨ (t3 ∧ t8) (car t3 ∈ d ∧ t8 ∈ d). Dans ce modèle, le résultat du calcul de pertinence est une valeur booléenne. Il ne permet donc pas de classer les documents en fonction de leur pertinence par rapport à une requête.

7.1.3 Avantages et inconvénients

Le modèle booléen a l'avantage d'être simple à mettre en oeuvre. En effet, un document est seulement représenté par l'ensemble des termes qui le composent. L'indexation est donc relativement simple à effectuer. Ce modèle est également très simple du point de vue des utilisateurs car les requêtes sont simples à écrire. Néanmoins il a l'inconvénient de ne pas permettre le tri des documents pertinents. Dans le cas d'une requête renvoyant de nombreux documents cela est handicapant pour les utilisateurs qui doivent parcourir la liste en intégralité.

7.2 Modèle vectoriel

7.2.1 Représentation des documents et des requêtes

Dans le modèle vectoriel, les documents et les requêtes sont représentés par des vecteurs à n dimensions. Chaque dimension correspond à un terme d'un vocabulaire V = t1, ..., tn. Chaque dimension du vecteur d'un document (ou d'une requête) est pondérée selon l'importance du terme dans le document. La valeur de pondération peut être normalisée dans l'intervalle [0, 1]. Le vecteur d'un document dont toutes les dimensions sont pondérées à 0, sauf les dimensions relatives aux termes t1, t3 et t8. Cela signifie que seuls ces concepts sont représentatifs du contenu du document.

La pondération des dimensions reflète l'importance des termes dans le document. Elle peut être calculée en considérant la fréquence d'apparition de chaque terme dans le document. La fréquence du terme ti dans un document d est notée tf (pour term frequency) et est donnée par :

t_1	t_2	t_3	…	t_8	…	t_n
0,3	0	1	…	0,5	…	0

Figure 6 **–** Exemple d'un vecteur représentant un document.

$$tf(t_i, d) = \frac{\text{nombre d'apparitions de } t_i \text{ dans } d}{\text{nombre de termes dans } d}$$

Plus un terme apparaît dans un document, plus sa pondération dans le vecteur est élevée. La fréquence des termes d'un document doit être recalculée à chaque fois que le document est modifié. La fréquence d'un terme dans un document peut être élevée pour deux raisons : soit parce que le terme est très représentatif du document, soit parce que le terme est très général. Pour capturer ce phénomène, il faut considérer la fréquence du terme dans l'ensemble des documents de la base B (ou du corpus). La fréquence inverse du document (notée idf pour inverse document frequency) est donnée par :

$$idf(t_i, \mathcal{B}) = \log\left(\frac{\text{nombre de documents dans la base } \mathcal{B}}{\text{nombre de documents contenant le terme } t_i}\right)$$

Cette mesure détermine à quel point un terme est général. Elle doit être recalculée à chaque fois que la base B est modifiée (ajout ou suppression de documents). La pondération du terme ti dans document d (de la base B) est généralement donnée par le tf · idf définit par le produit tf(ti, d) × idf(ti, B).

7.2.2 Calcul de pertinence

Dans ce modèle, la pertinence entre un document et une requête se calcule en mesurant la similarité entre vecteurs. La mesure la plus utilisée est le cosinus :

$$pert(d,q) = \cos(\vec{d}, \vec{q}) = \frac{\vec{d} \cdot \vec{q}}{\| \vec{d} \| \times \| \vec{q} \|}$$

Dans cette mesure, −→d · −→q représente le produit scalaire entre le vecteur du document et le vecteur de la requête ; et k−→v k représente la norme du vecteur −→v.

7.2.3 Avantages et inconvénients

Le premier avantage de ce modèle est que les documents et les requêtes sont représentés de manière simple. Le deuxième avantage est qu'il permet de pondérer les termes dans les index. Au-delà de la recherche d'information, cela peut être utile pour caractériser un document (en présentant ses termes les plus représentatifs), ou pour catégoriser les documents. Enfin le dernier avantage est qu'il permet de trier les documents en fonction de leur pertinence. Cela représente évidemment un confort supplémentaire pour l'utilisateur final. Le principal inconvénient de ce modèle est que les dimensions qui forment l'espace vectoriel sont indépendantes. Ainsi les dimensions correspondant aux termes "voiture" et "véhicule" sont aussi indépendantes que les dimensions "voiture" et "arbre". Cela est un problème car une requête ne contenant que le terme "véhicule" n'est pas satisfaite par un document ne contenant que le terme "voiture" (alors qu'une voiture est un véhicule).

Notons que ce problème est également présent dans le modèle booléen.

7.3 Modèle booléen étendu

7.3.1 Représentation des documents et des requêtes

Dans le modèle booléen étendu proposé par Salton, Fox etWu [SFW83] les documents sont représentés par des vecteurs à n dimensions (comme dans le modèle vectoriel) et les requêtes sont des expressions booléennes (comme dans le modèle booléen). L'idée est de profiter des avantages du modèle vectoriel (le fait que les termes sont pondérés en fonction de leur importance dans le document) et des avantages du modèle booléen (la simplicité du point de vue de l'utilisateur).

7.3.2 Calcul de pertinence

Pour expliquer le calcul de la pertinence d'un document par rapport à une requête, nous nous limitons à un espace à deux dimensions : t1 et t2. Dans un souci de simplicité, nous

présentons la mesure de pertinence sur deux requêtes particulières. Soient deux requêtes q∧ et q∨ définies par t1 ∧ t2 et t1 ∨ t2. Pour qu'un document soit totalement pertinent pour q∧, il faut que les termes t1 et t2 soient pondérés à 1 : il doit se trouver à la coordonnée (1, 1). De manière générale, plus le document est proche de la position (1, 1), plus il est pertinent pour la requête q1.

Dans ce cas, le degré de pertinence d'un document par rapport à une requête est donné par :

$$pert(\vec{d}, q_{\wedge}) = 1 - \sqrt{\frac{(1 - \vec{d}[t_1])^2 + (1 - \vec{d}[t_2])^2}{2}}$$

Où −→d [ti] représente la pondération du terme ti dans le vecteur −→d .

Pour qu'un document soit totalement pertinent pour q∨, il faut que le terme t1 ou le terme t2 soit pondéré à 1 : il doit se trouver à la coordonnée (1, 0), à la coordonnée (0, 1) ou à la coordonnée (1, 1). Dans ce cas, le degré de pertinence d'un document par rapport à une requête est donné par :

$$pert(\vec{d}, q_{\vee}) = \sqrt{\frac{\vec{d}[t_1]^2 + \vec{d}[t_2]^2}{2}}$$

Où −→d [ti] représente la pondération du terme ti dans le vecteur −→d .

Les mesures de pertinence peuvent être généralisées au cas des requêtes composées d'un nombre quelconque de termes. Une mesure permet également de mesurer la pertinence d'un document par rapport à une requête combinant disjonctions et conjonctions. Les détails sont donnés dans [SFW83].

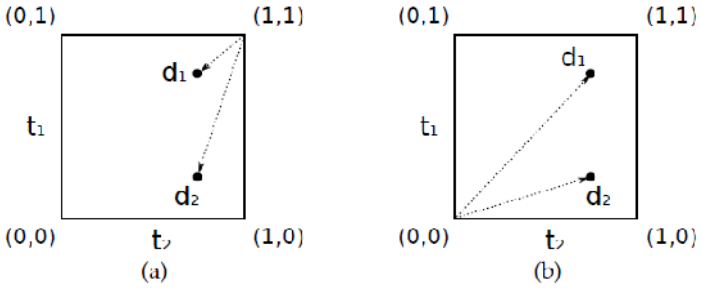

Figure 7 – Exemple de deux documents d1 et d2 représentés dans un espace à deux dimensions. Dans le modèle booléen étendu, la pertinence est mesurée par la distance entre les documents et les coordonnées (0, 0) pour les requêtes disjonctives (a), et (1, 1) pour les requêtes conjonctives (b).

7.3.4 Avantages et inconvénients

Le premier avantage de ce modèle est que les requêtes sont simples à formuler pour les utilisateurs. Le deuxième concerne la représentation des documents et la mesure de pertinence. En effet, ceux-ci permettent de trier les documents en fonction de leur pertinence (contrairement au modèle booléen classique). Les désavantages sont les mêmes que pour les modèles vectoriels.

7.4 Modèle probabiliste de pertinence

7.4.1 Représentation des documents et des requêtes

Le modèle probabiliste a été proposé par Robertson et Spärck Jones en 1976 [60]. Dans ce modèle les documents et les requêtes sont représentés par des ensembles de termes d'un vocabulaire V. On note d [ti] la pondération du terme ti dans le document d. Elle vaut 0 si le terme ti est absent de d, et 1 s'il est présent.

7.4.2 Calcul de pertinence

La mesure de pertinence considère la probabilité P(R|d) que le document d soit pertinent pour une requête q, et la probabilité P(R|d) que le document d ne soit pas pertinent pour la requête q. La mesure de pertinence est définie par :

$$pert(d,q) = \frac{P(R|d)}{P(\overline{R}|d)}$$

En utilisant le théorème de Bayes [61], on obtient :

$$pert(d,q) = \sum_{i=1}^{|V|} d[t_i] \times w_i$$

Où Wi dépend de la probabilité de présence du terme ti dans l'ensemble des documents pertinents et des documents non pertinents. Au départ ces probabilités sont estimées de manière triviale puis sont raffinées au cours des itérations durant lesquelles les utilisateurs donnent un avis (un feedback).

7.4.3 Avantages et inconvénients

Les systèmes utilisant le modèle probabilistique obtiennent de bons résultats, à l'image du système OKAPI [62]. Néanmoins ce modèle est assez complexe à mettre en œuvre car il nécessite d'avoir une estimation des probabilités initiales.

Conclusion :

Dans la première partie de ce chapitre on a présenté les ontologies, et leurs rôles dans l'échange de données dans les systèmes pair à pair, l'état de l'art concernant les mesures de similarité sémantique entre entités d'une même ontologie (similarité intra-ontologie), et sur les mesures de (dis) similarité entre deux ontologies, l'accent est mis sur l'hétérogénéité et l'alignement des ontologies, ainsi qu'un panorama des mesures sémantiques et le calcul de la similarité.

La deuxième partie est consacrée pour expliquer les différents méthodes et modèles de recherches d'information,

Dans le prochain chapitre on va présenter notre contribution dans ce domaine.

Chapitre III

Contribution

1. Introduction :

Dans ce chapitre nous représentons nos contributions. Nous commençons par la description logique des ensembles de connaissances du système P2P proposé en utilisant les langages RDF(s) et OWL. Ensuite nous créons des ontologies des utilisateurs qui ont les même profils en utilisant l'outil protégé 2000 qui nous permet la visualisation de notre ontologie.

2. Solution proposé pour les réseaux pairs à pairs :

Notre idée repose, au début, sur la création des profils pour les utilisateurs du web pour simplifier la recherche des données partagées et améliorer la qualité de réponse. L'utilisateur est identifié par son adresse physique parce qu'elle est unique. Chaque utilisateur a trois classes dont la première est son profil qui est construit sur la base de ses intérêts, la deuxième est l'index au moyen duquel on fait la recherche. Enfin la troisième classe concerne les données partagées.

3. principe général :

Quand l'utilisateur envoie une requête pour chercher des données, il cherche tout d'abord dans son index et ainsi il l'envoie à ces voisins qui à leurs tours sont à la recherche dans leurs index. Dans le cas où ils trouvent les données souhaitées ils répondent à la requête, dans le cas où les données sont trouvées, le pair qui a ces données transmet son adresse à l'utilisateur pour créer un canal entre eux ; au même temps il ajoute ces données à son index pour être à jour.

A partir de ce concept nous regroupons les utilisateurs dans des clusters (groupe d'utilisateurs qui ont des profils similaires).

4. Démonstration :

4.1 Premier scénario :

Pour qu'on puisse expliquer nos idées nous présentons le Pair sur ce schéma :

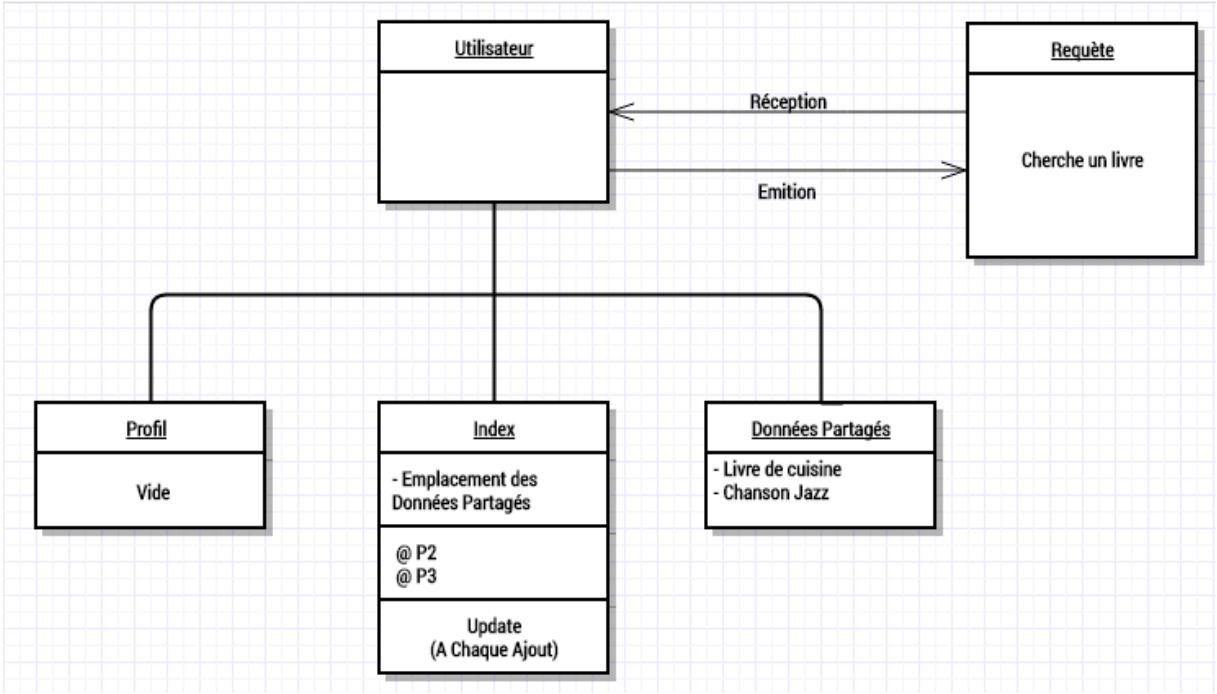

Figure 8 : *La composition d'utilisateur.*

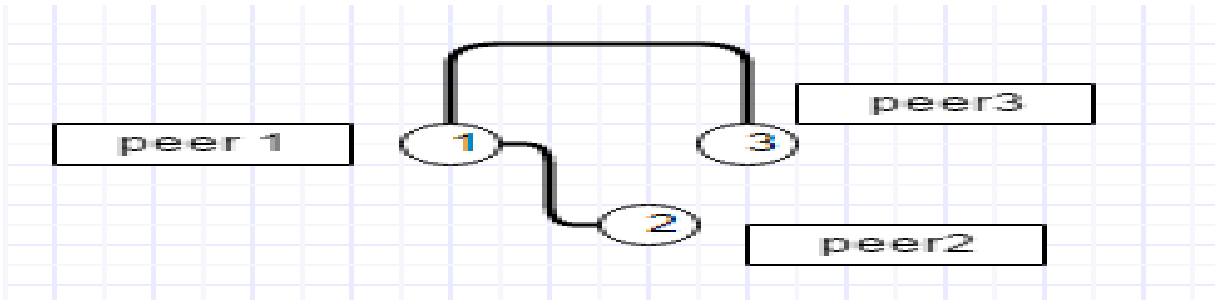

Figure 9 : *Les pairs connectés.*

Le Pair 1 (ex : il cherche un livre) il cherche localement dans son index et envoie une requête à ces voisins : pair2, pair3 qui, à leurs tours cherchent dans leurs index.

Cas 1 : s'il trouve les données cherchées dans p2 et p3 (requête satisfaite).

Cas 2 :s'il ne trouve pas ce qu'il cherche alors p2 et p3 passent la requête à leurs voisins.

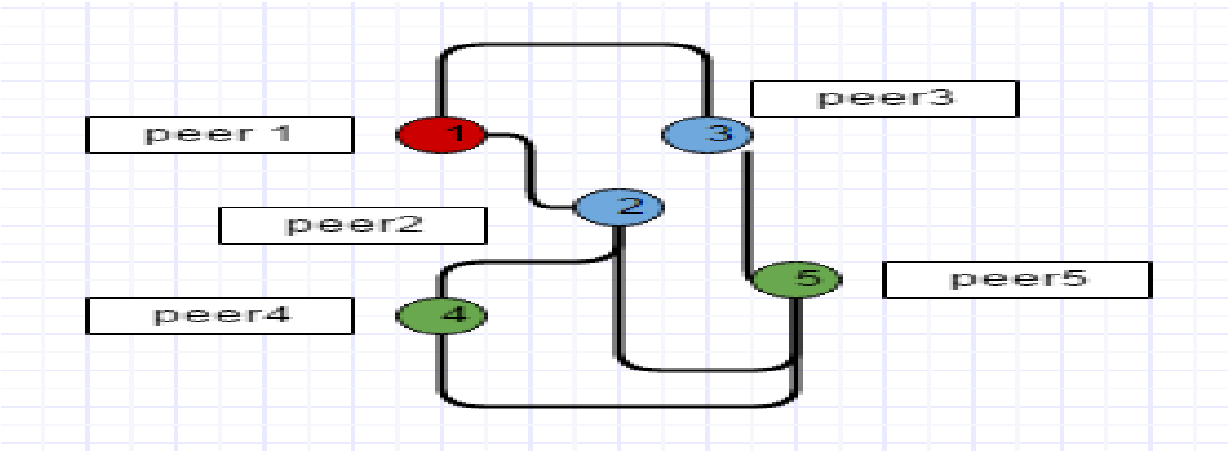

Figure 10 : *La notion de voisinage.*

La requête est au niveau de p4 et p5, supposant que p1 a trouvé ce qu'il cherche, les pairs répondent en envoyant leurs adresses (p4 et p5), alors p1 crée un lien avec ces deux nouveaux Pairs.

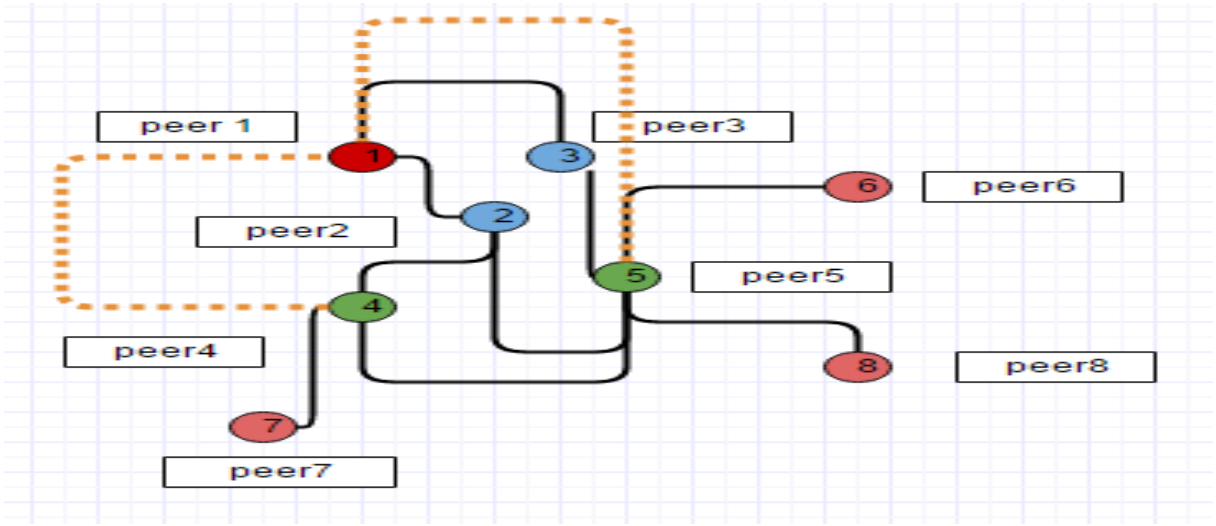

Figure 11 : *La création des liens.*

Le Pair 1 ajoute à la classe profil (livre) et à la classe index : @p4 et @p5 (mise-à-jour en cas des modifications)

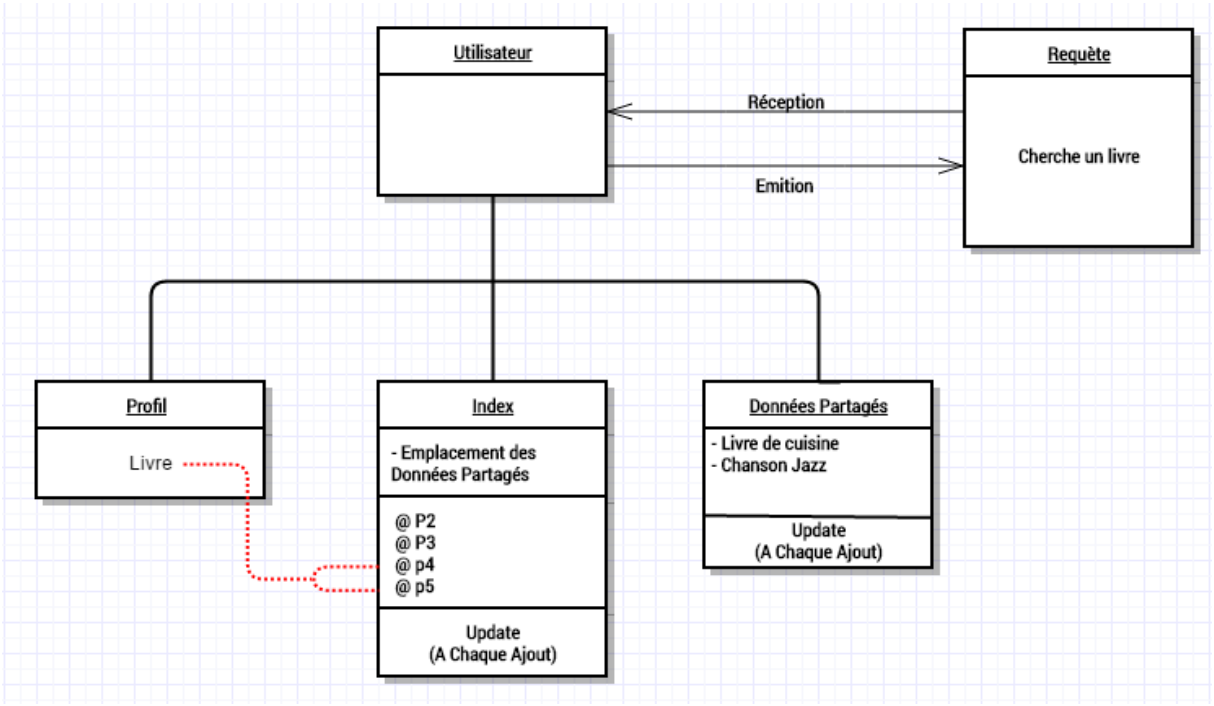

Figure 12 : *L'ajout des connaissances.*

4.2 Deuxième scénario :

Le Pair 1 cherche un film, alors la requête s'envoie à p2, p3, p4 et p5.

Le film se trouve au niveau de p6 et p7 alors le schéma se construit comme suit :

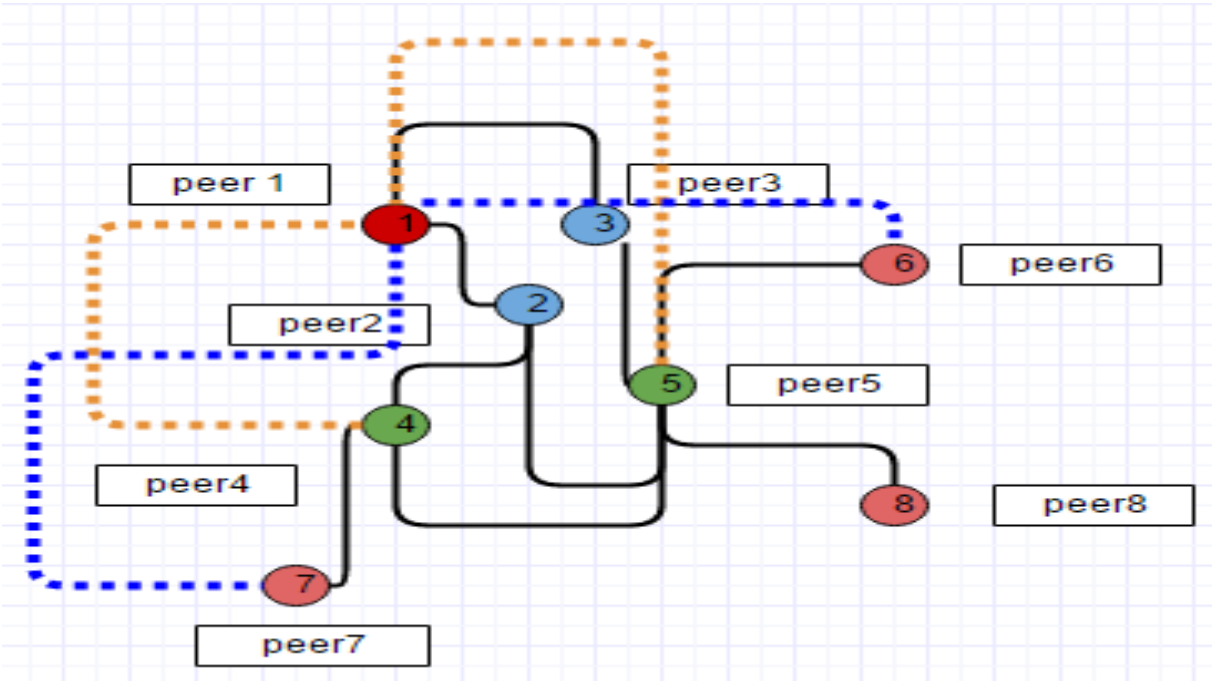

Figure 13 : *La connexion avec des pairs pertinents.*

Le Pair1 ajoute film à la classe profil et @p6, @p7 à la classe index :

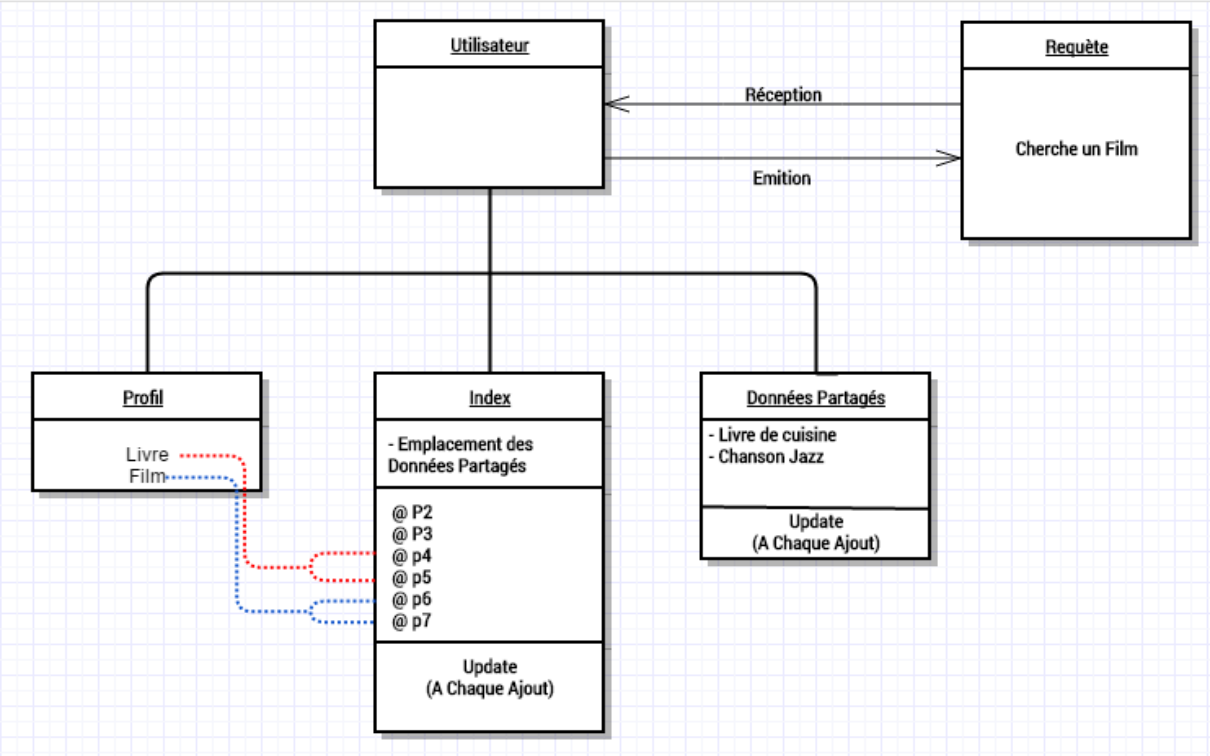

Figure 14 : *L'ajout des voisins pertinent avec remplissage du profil.*

L'utilisateur est le seul responsable de ce qu'il veut le partager.

La question qui se pose : où est le rôle d'ontologie ?

Lorsque le pair envoie une requête pour chercher des données, son profil sera rempli selon la recherche effectuée, ainsi que son index qui sera aussi rempli avec les adresses des pairs qui contiennent les données cherchées, et la même chose pour les voisins. Alors le pair aura acquis des connaissances concernant les intérêts des pairs de ce réseau, ce qui rend le pair intelligent, il deviendra donc plus rapide et précis dans ses prochaines recherches et ses réponses.

Les pairs ayant des profils similaires se regroupent dans un cluster (ensemble).

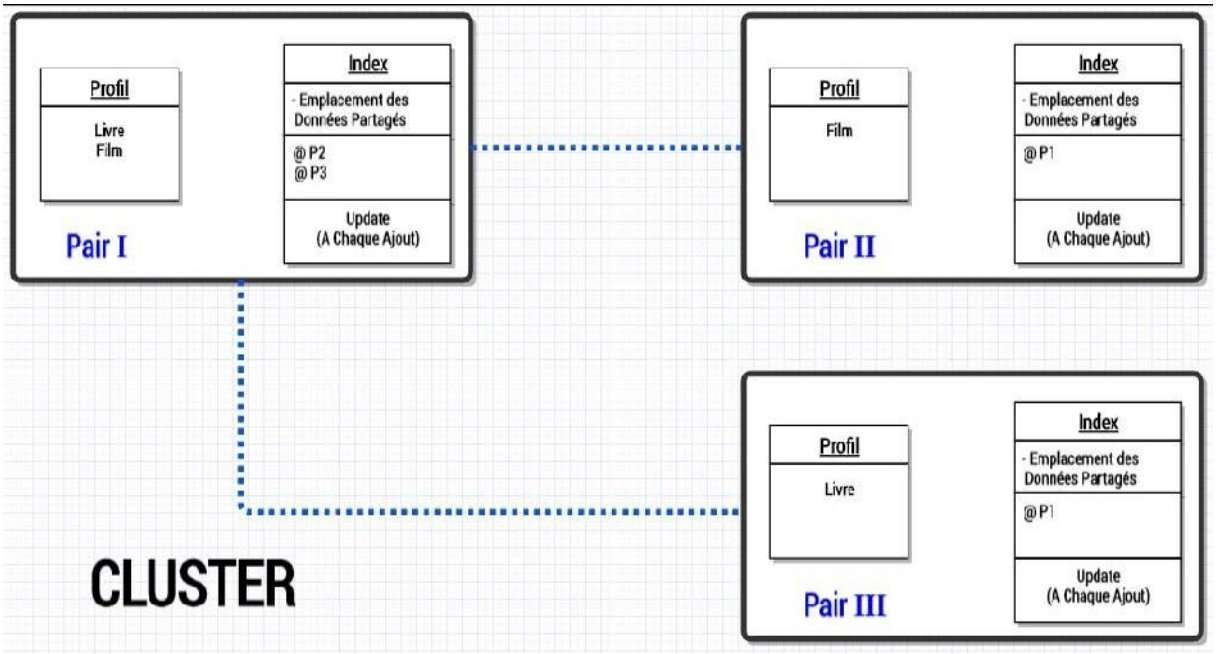

Figure 15 : *Cluster.*

5. L'éditeur d'ontologies PROTEGE :

Protégé est un outil de développement d'ontologies utilisé par les développeurs et les experts de domaine. Les applications développées avec Protégé sont utilisées pour la résolution de problèmes et la prise de décision dans un domaine précis.

Cet outil est une interface modulaire permettant la création, la visualisation, le contrôle d'ontologie, l'extraction d'ontologies à partir de sources textuelles, et la fusion semi-automatique d'ontologies. Le modèle de connaissances sous-jacent à protégé 2000 est venue du modèle des frames et comporte des classes, des slots (propriétés) et des facettes (valeurs des propriétés et contraintes), ainsi que des instances des classes et des propriétés. Il autorise la définition de méta-classes, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de construire une ontologie

6. Description :

- ✓ L'utilisateur est une classe ;
- ✓ Requête est une classe disjoint de utilisateur ;
- ✓ Profil est une partie de la classe utilisateur ;
- ✓ Index est une partie de la classe utilisateur ;
- ✓ Donnée est une partie de la classe utilisateur.
- ✓ Les classe profil, index, donnée sont disjoint.
- ✓ L'utilisateur envoi une requête ;
- ✓ Requête reçoit par l'utilisateur ;

7- les étapes de la création de l'ontologie :

Nous accédons à l'outil protégé 2000

La première étape est de choisir « New Project » après l'ouverture du l'éditeur protégé.

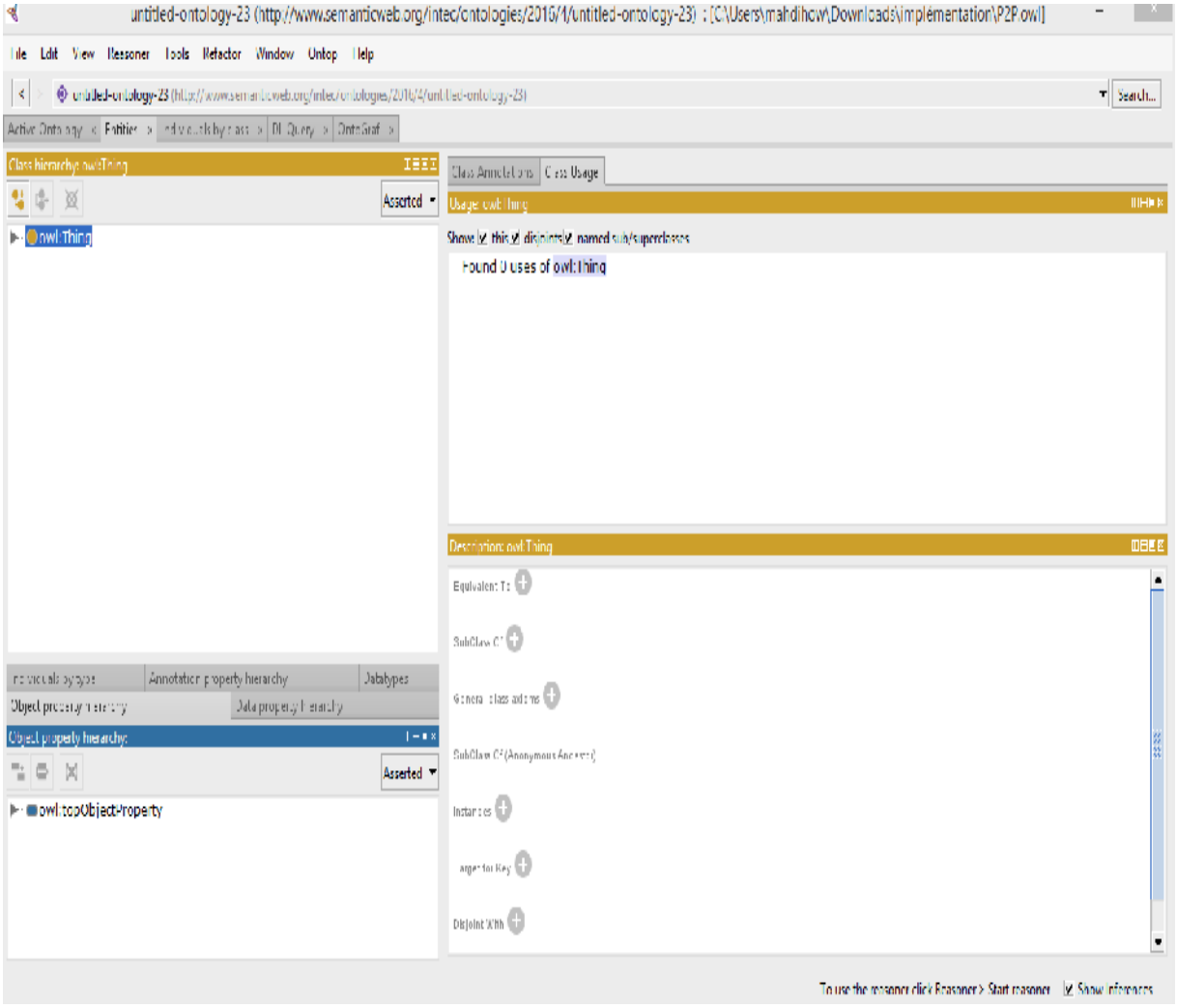

Figure 16 : *Interface de protégé.*

Nous passons à la création des classes, en commençant par la classe Requête :
C'est la classe responsable de l'envoi et de la réception des requêtes

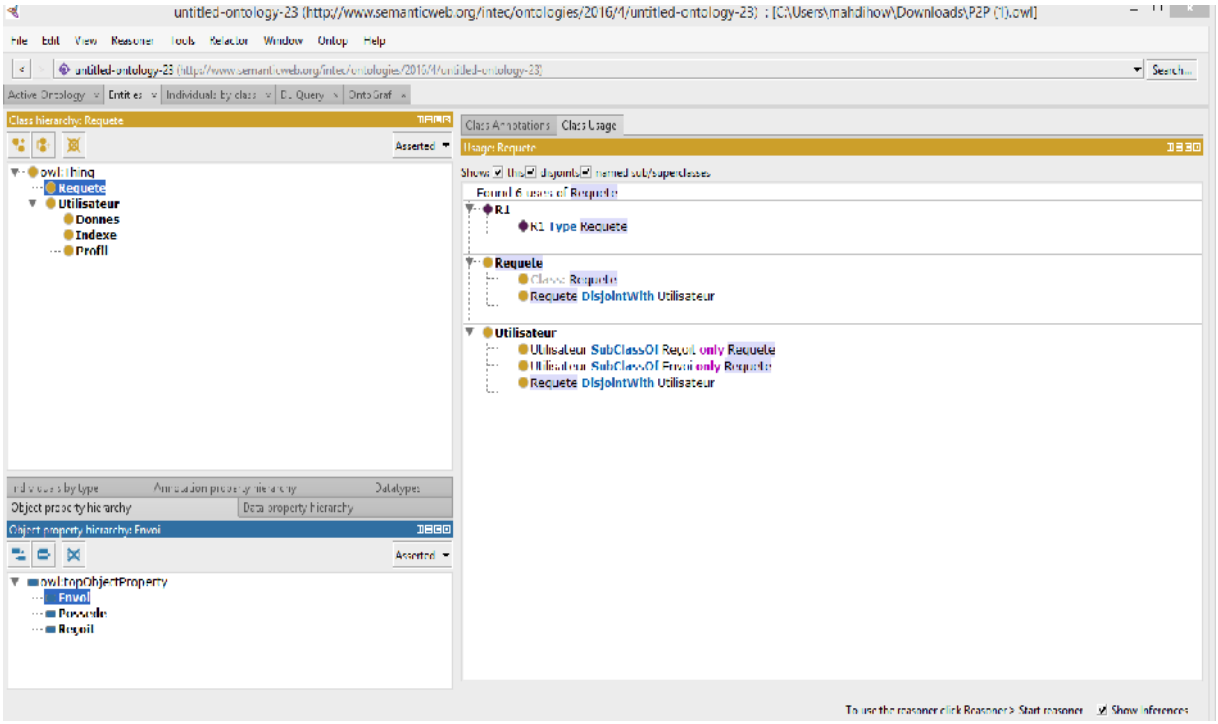

Figure 17 : *La création de la classe requête.*

7.1 La création de la classe utilisateur :

C'est la classe mère qui regroupe toutes les sous classes, et c'est elle qui représente le pair dans le système P2P

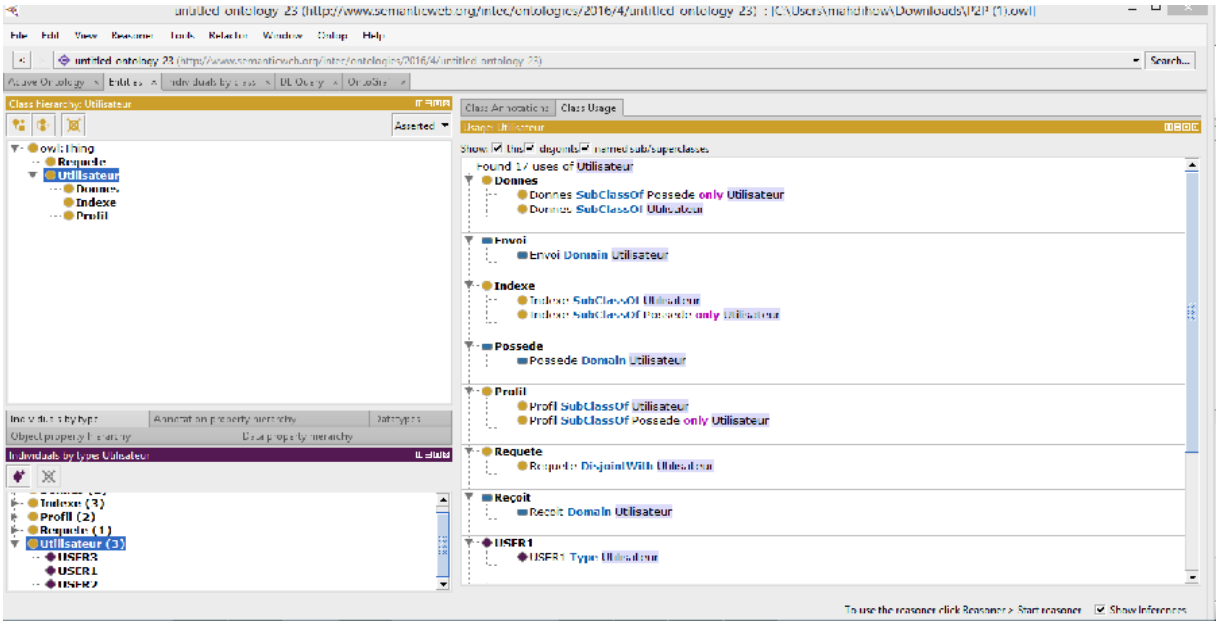

Figure 18 : *La création de la classe utilisateur.*

7.2 La création de sous-classe donnée :

C'est la classe qui contient les données partagées par le pair dans le système P2P proposé.

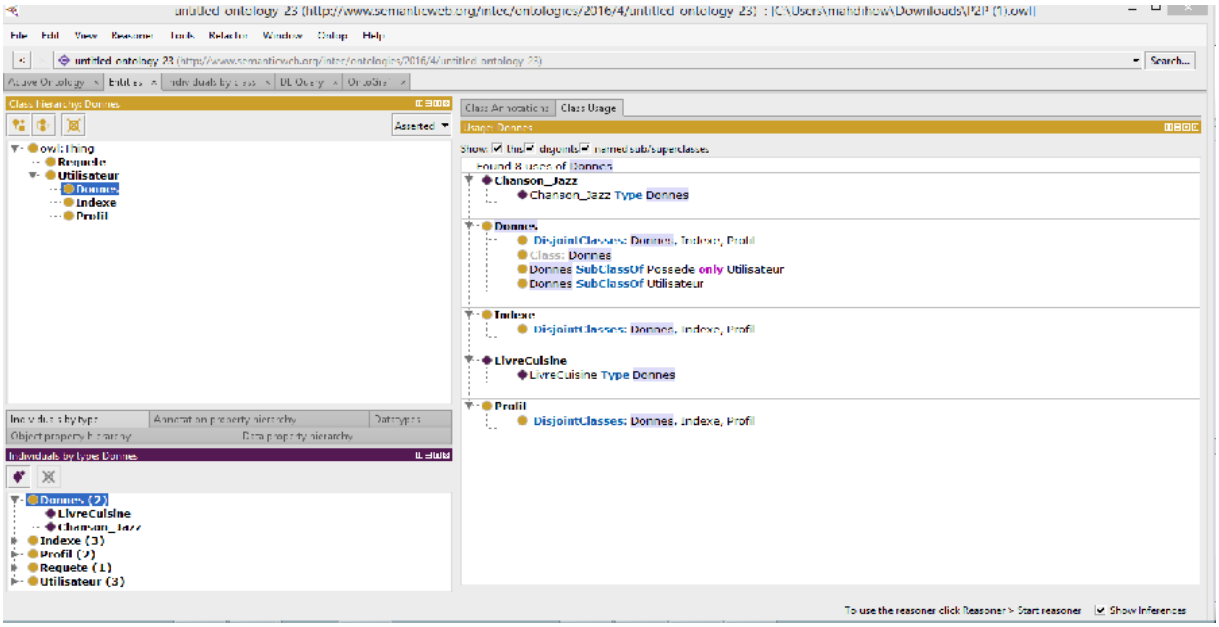

Figure 19 : *La création de sous-classe donnée.*

7.3 La création de sous-classe index :

C'est la classe qui contient l'emplacement de données partagées de l'utilisateur ainsi que les adresses des pairs détenteurs de ces données. Après Chaque modification dans le réseau, l'index se mettra à jour.

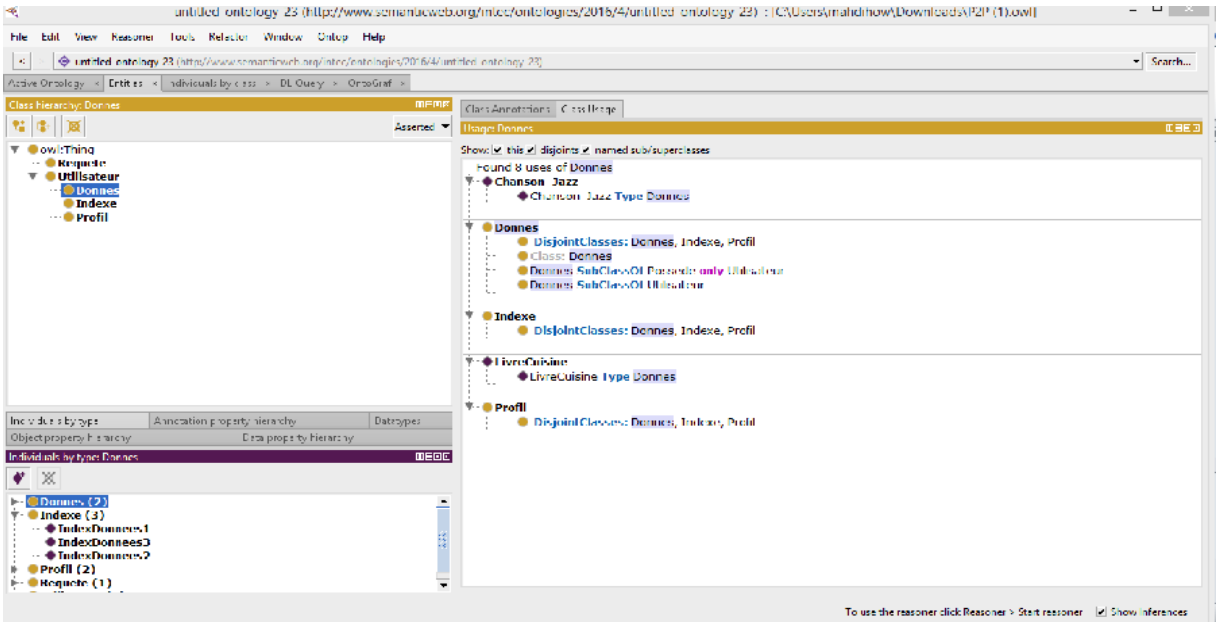

Figure 20 : *La création de sous-classe index.*

7.4 la création de sous-classe profil :

C'est la classe qui contient les types de données qui intéressent chaque utilisateur du système P2P

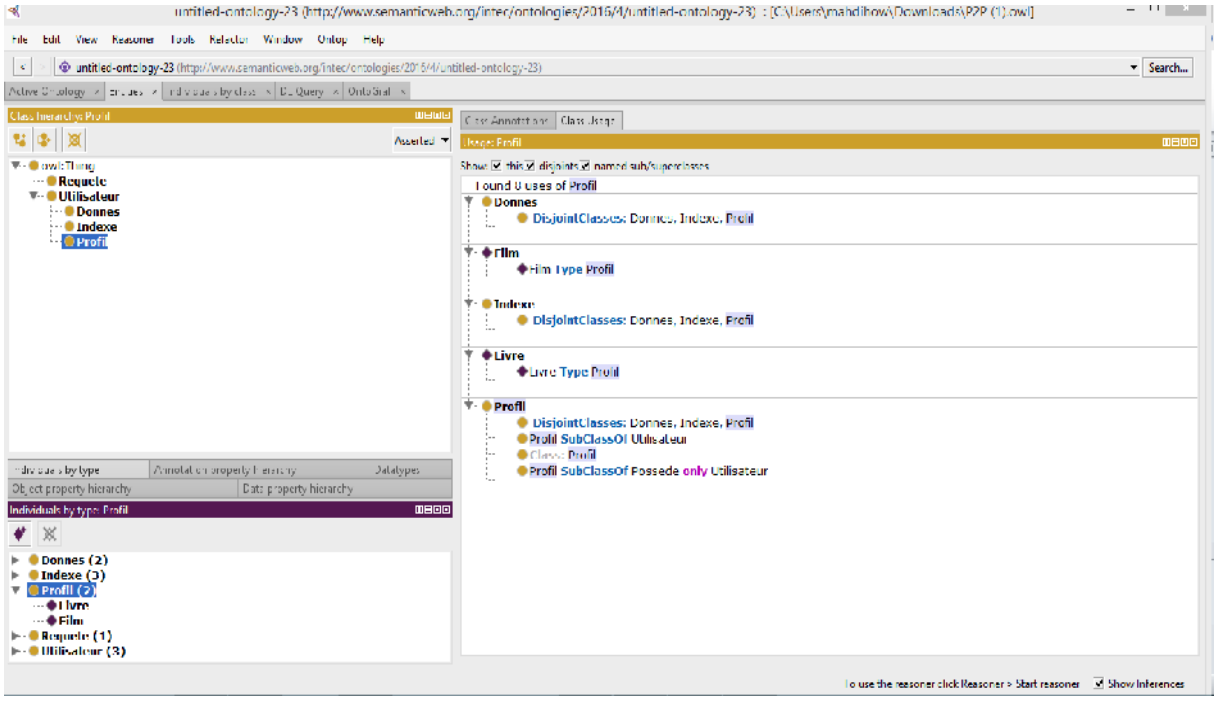

Figure 21 : *La création de sous-classe profil.*

8. Représentation d'ontologie dans l'outil protégé 2000 :

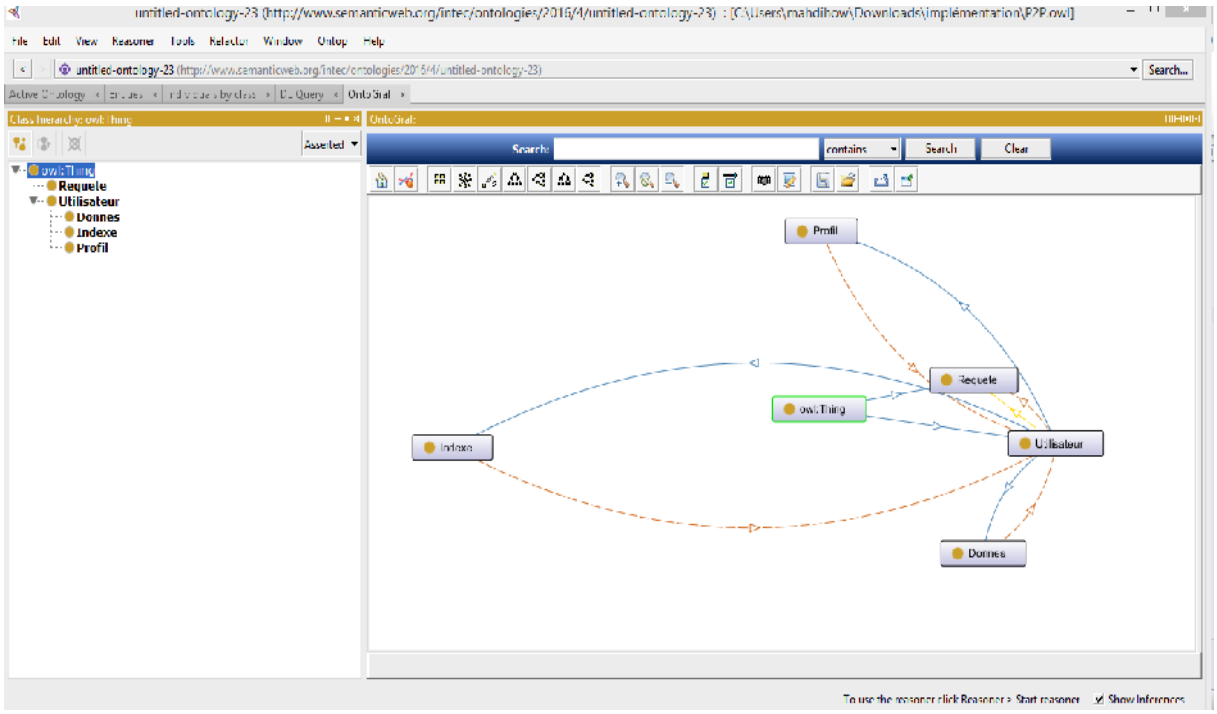

Figure 22 : *Ontologie graphe.*

9. Visualisation détaillé en ligne :

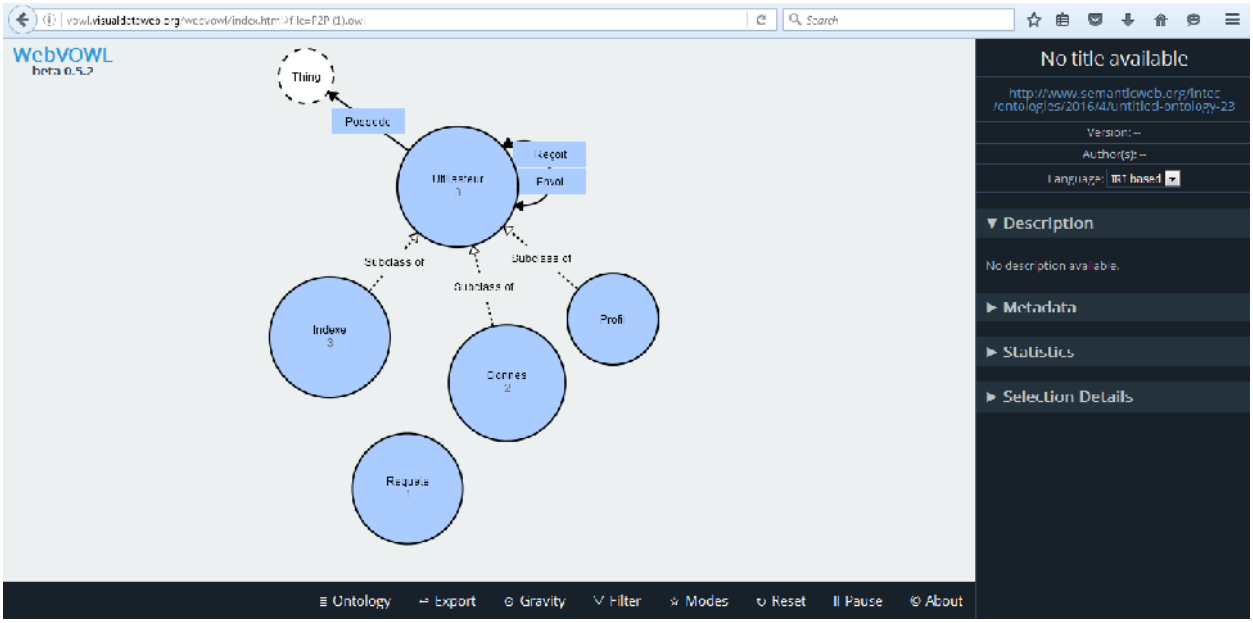

Figure 23 : *Ontologie graphe en ligne.*

C’est un schéma explicatif qui détaille notre ontologie et sa visualisation en ligne.

Conclusion

Dans ce chapitre nous avons présenté les étapes de la création d’une ontologie avec protégé 2000 et la solution proposée et nous exposerons des captures d’écran de notre travail en effectuant des scénarios au système P2P proposé pour obtenir des meilleurs résultats.

Conclusion Générale

Conclusion générale

Dans ce travail nous avons proposé un ensemble de solutions permettant d'adresser le problème des ontologies dans les systèmes P2P non-structurés.

Dans un premier temps, nous avons défini un système P2P dont lequel nous pourrons travailler et réaliser notre contributions alors nous avons commencé par la description d'un système P2P en utilisant les langages RDF(s) et OWL qui sont basés sur la logique de description après nous présentions les connaissances du système proposé : concepts, relations, fonctions, axiomes et instances et toutes ces connaissances ont été présenté sur protégé 2000 qui est une interface modulaire dont nous parlé dans le chapitre2.

Dans un deuxième temps nous avons regroupé les ontologies des utilisateurs qui ont les mêmes profils dans des groupes ou Cluster dans le but de réduire le temps de réponse et alors améliorer notre réseau P2P et donc l'optimiser.

Enfin nous avons testé ce système en envoyant des requêtes dans le but de déterminer les résultats obtenus et discuter ces résultats pour fixer l'amélioration de notre réseau en utilisant l'approche proposée.

Pour conclure, nous pouvons dire que nous avons défini un système P2P dont nous utilisons le regroupement d'ontologies des utilisateurs pour l'amélioration de ce système et leur fonctionnement. Cette proposition répond à la problématique que nous avons proposé d'aborder en introduction. Les perspectives de notre travail sont montrées dans la section suivante.

Perspectives et travaux futures

Il s'agit dans un premier temps de passer à une grande échelle c à d agrandir notre réseau pour arriver ensuite à la phase de simulation à l'aise d'un simulateur adéquat.

Puis nous atteindrons l'étape de l'expérimentation dans la réalité et ce dans le but d'améliorer la recherche dans les systèmes P2P en intégrant les ontologies.

BIBLIOGRAPHIE

Bibliographie

[1] Lawrence Edward Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking : Bringingorder to the web, 1999.

[2] UdiManber, Ash Patel, and John Robison.The business of personalization : experience with personalization of yahoo !Communications of the ACM, 43(8) :35–39, 2000.

[3] Steffen Staab and StudiStuder.Handbook on Ontologies.Springer, 2004.

[4] Shelley Powers. Practical RDF.O'Reilly & Associates, Inc., 2003.

[5] Ellen M. Voorhees. Query expansion using lexical-semanticrelations. In 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval, pages 61–69, 1994.

[6] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre,and Patrick Valduriez. Improving interoperability usingquery interpretation in semantic vector spaces. In 5th EuropeanSemantic Web Conference, pages 539–553, 2008.

[7] Stefan Saroiu, Krishna P. Gummadi, and Steven D. Gribble.Measuring and analyzing the characteristics of napster andgnutella hosts. Multimedia Systems, 9(2) :170–184, 2003.

[8] Ion Stoica, Robert Morris, David Liben-Nowell, David R.Karger, M. FransKaashoek, Frank Dabek, and HariBalakrishnan.Chord : a scalable peer-to-peer lookup protocolfor internet applications. IEEE/ACM Transactions on Networking,11(1) :17–32, 2003.

[9] Bram Cohen. Incentives build robustness in BitTorrent. In1stWorkshop on Economics of Peer-to-Peer Systems, 2003.

[10] Ben Y. Zhao, Ling Huang, Jeremy Stribling, Sean C. Rhea,Anthony D. Joseph, and John Kubiatowicz.Tapestry : a resilientglobal-scale overlay for service deployment. IEEEJournal on Selected Areas in Communications, 22(1) :41–53,2004.

[11] JérômeEuzenat, Thanh Le Bach, JesúsBarrasa, Paolo Bouquet,Jan De Bo, Rose Dieng-Kuntz, Marc Ehrig, Manfred Hauswirth, Mustafa Jarrar, Ruben Lara, Diana Maynard,Amedeo Napoli, GiorgosStamou, HeinerStuckenschmidt,PavelShvaiko, Sergio Tessaris, Sven Van Acker, and IlyaZaihrayeu. State of the art on ontology alignment.KnowledgeWeb Deliverable 2.2.3, 2004.

[12] AdilHameed, AlunPreece, and Derek Sleeman.Ontologyreconciliation. Handbook on ontologies, pages 231–250, 2004.

[13] Paul R. Smart and Paula C. Engelbrecht. An analysis ofthe origin of ontology mismatches on the semantic web. InEKAW, pages 120–135, 2008.

[14] JérômeEuzenat and PavelShvaiko.Ontology matching.Springer-Verlag, 2007.

[15] David Aumueller, Hong Hai Do, Sabine Massmann, andErhard Rahm.Schema and ontology matching withCOMA++. In SIGMOD, pages 906–908, 2005.

[16] Kelly Moran, Kajal T. Claypool, and Benjamin J. Hescott.Compositematch : Detecting n-ary matches in ontology alignment.In OM, 2009.

[17] RaddadAL KING. Thèse doctorat : Localisation de sources de données et optimisation de requêtes répartiesen environnement pair-à-pair. Ecole doctorale :Mathématiques, Informatique, élécommunications deToulouse .2010

[18] S. Androutsellis-Theotokis and D. Spinellis.A survey of peer-to-peer content distribution technologies. ACM Computing Surveys, 36(4) :335–371, December 2004.

[19] D. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing.Technical Report HPL-2002-57, HP Laboratories, 2002.

[20] Andersen D. G., Balakrishnan H., Kaashoek F., Morris R. (2001) Resilient Overlay Networks. In: 18th Symposium on Operating Systems Principles (SOSP), Oct 21-24, Banff, Canada. ACM Press, pp.131–145

[21] N. Harvey, M. Jones, S. Saroiu, M. Theimer and A. Wolman. “Skipnet: a scalableoverlay network with practical locality properties”. USENIX Symp.on InternetTechnologies and Systems, pp : 113-126, 2003.

[22] S. Michel, P. Triantafillou and G. Weikum. “KLEE: a framework for distributedtop-k query algorithms”. Int. Conf. on Very Large Databases (VLDB’05), pp. 637-648, 2005.

[23] W. T. Balke, W. Nejdl, W. Siberski and U. Thaden. “Progressive distributed top-kretrieval in peer-to-peer networks”. In Proc. of the Int. Conf. on Data Engineering(ICDE), pp : 174-185, 2005.

[24] S. Chaudhuri, L. Gravano and A. Marian. “Optimizing top-k selection queries overmultimedia repositories”.IEEE Transactions on Knowledge and Data Engineering,16(8), pp. 992- 1009, 2004.

[25] A.P. DeVries, N. Mamoulis, N. Nes and M.L. Kersten. “Efficient k-++ search onvertically decomposed data”. ACM Int. Conf. on Management of Data(SIGMOD’02), pp. 322-333, 2002

[26] C. Bohm, S. Berchtold and D.A. Keim. “Searching in high-dimensional spaces:index structures for improving the performance of multimedia databases”.ACMComputing Surveys, 33(3), pp. 322-373, 2001.

[27] R. Blanco, N. Ahmed, D. Hadaller, L.G.A. Sung, H. Li and M.A. Soliman. “Asurvey of data management in peer-to-peer systems”.Technical Report CS-2006-18, University of Waterloo, 2006.

[28] VanaKalogeraki, DimitriosGunopulos, andD. Zeinalipour-Yazti. A local search mechanism for
peer-to-peer networks, 2002. (Cité page 36.)

[29] Qin Lv, Pei Cao, Edith Cohen, Kai Li, and Scott Shenker.Search and replication in unstructured peer-to-peer networks.In 16th ACM International Conference on Supercomputing,pages 84–95, 2002.

[30] Beverly Yang and Hector Garcia-Molina.Improving searchin peer-to-peer networks. In 22nd International Conferenceon Distributed Computing Systems, pages 5–14, 2002.

[31] DimitriosTsoumakos and Nick Roussopoulos. Adaptiveprobabilistic search for peer-to-peer networkks. In 3rd InternationalConference on Peer-to-Peer Computing, pages 102–
109, 2003.

[32] Arturo Crespo and Hector Garcia-Molina.Routing indicesfor peer-to-peer systems. In 22nd International Conferenceon Distributed Computing Systems, pages 23–33, 2002. (Citépage 36.)

[33] Sean C. Rhea and John Kubiatowicz.Probabilistic locationand routing.In 21st Annual Joint Conference of the IEEE Computerand Communications Societies, pages 1248–1257, 2002.

[34] Reza Akbarinia, Esther Pacitti, and Patrick Valduriez.Reducingnetwork traffic in unstructured P2P systems usingtop-k queries. Distributed and Parallel Databases, 19(2-3) :67–86, 2006.

[35] Paolo Bouquet, Marc Ehrig, JérômeEuzenat, Enrico Franconi,Pascal Hitzler, Markus Krötzsch, Luciano Serafini,GiorgosStamou, York Sure, and Sergio Tessaris. Specificationof a common framework for characterizing alignment.Knowledge Web Deliverable 2.2.1, 2004.

[36] JérômeEuzenat and PavelShvaiko.Ontology matching.Springer-Verlag, 2007.

[37] Paul R. Smart and Paula C. Engelbrecht. An analysis ofthe origin of ontology mismatches on the semantic web. InEKAW, pages 120–135, 2008.

[38] AdilHameed, AlunPreece, and Derek Sleeman.Ontologyreconciliation. Handbook on ontologies, pages 231–250, 2004.

[39] Michel C. A. Klein and Dieter Fensel.Ontology versioningon the semantic web. In SWWS, pages 75–91, 2001.

[40] JérômeEuzenat, Thanh Le Bach, JesúsBarrasa, Paolo Bouquet,Jan De Bo, Rose Dieng-Kuntz, Marc Ehrig, ManfredHauswirth, Mustafa Jarrar, Ruben Lara, Diana Maynard,

Amedeo Napoli, GiorgosStamou, HeinerStuckenschmidt,PavelShvaiko, Sergio Tessaris, Sven Van Acker, and IlyaZaihrayeu. State of the art on ontology alignment.KnowledgeWeb Deliverable 2.2.3, 2004.

[41] Jos de Bruijn, Marc Ehrig, Cristina Feier, Francisco Martíns-Recuerda, François Scharffe, and Moritz Weiten. OntologyMediation, Merging, and Aligning, pages 95–113.John Wiley& Sons, Ltd, 2006.

[42] AdilHameed, AlunPreece, and Derek Sleeman.Ontologyreconciliation. Handbook on ontologies, pages 231–250, 2004.

[43] Fatiha Saïs. Intégration Sémantique de Données guidée par uneOntologie. PhDthesis, Université Paris-Sud, 2007.

[44] YonggangQiu and Hans-Peter Frei. Concept based queryexpansion. In 16th Annual International ACM SIGIR Conferenceon Research and Development in Information Retrieval,pages 160–169, 1993.

[45] Anthony Ventresque, Sylvie Cazalens, Philippe Lamarre,and Patrick Valduriez. Improving interoperability usingquery interpretation in semantic vector spaces. In 5th EuropeanSemantic Web Conference, pages 539–553, 2008.

[46] David S. Batista, João D. Ferreira, Francisco M. Couto, andMário J. Silva.Toponym disambiguation using ontologybasedsemantic similarity. In 10th International Conferenceon Computational Processing of the Portuguese Language, pages179–185, 2012.

[47] R. Rada, H. Mili, E. Bicknell, and M. Blettner.Developmentand application of a metric on semantic nets. IEEE Transactionson Systems, Man, and Cybernetics, 19(1) :17–30, 1989.

[48] Zhibiao Wu and Martha Palmer.Verb semantics and lexicalselection. In 32nd Annual Meeting of the Association forComputational Linguistics (ACL), pages 133–138, 1994.

[49] Philip Resnik. Using information content to evaluate semanticsimilarity in a taxonomy. In In Proceedings of the 14th International Joint Conference on Artificial Intelligence, pages448–453, 1995.

[50] R. Richardson, A. F. Smeaton, and J. Murphy.Using wordnetas a knowledge base for measuring semantic similaritybetween words.Technical Report CA-1294, Dublin CityUniversity, 1994.

[51] NunoSeco, Tony Veale, and Jer Hayes.An intrinsic informationcontent metric for semantic similarity in wordnet.In 16th Eureopean Conference on Artificial Intelligence, pages1089–1090, 2004.

[52] J. J. Jiang and D. W. Conrath. Semantic similarity based oncorpus statistics and lexical taxonomy. In 10th InternationalConference Research on Computational Linguistics, pages 19–33, 1997.

[53] Dekang Lin. An information-theoretic definition of similarity.In 15th International Conference on Machine Learning,pages 296–304, 1998.

[54] Alexander Maedche and Steffen Staab.Measuring similaritybetween ontologies. In 13th International Conferenceon Knowledge Engineering and Knowledge Management, pages
251–263, 2002.

[55] Vladimir I. Levenshtein. Binary codes capable of correctingdeletions, insertions, and reversals. Soviet Physics Doklady,10(8) :707–710, 1966.

[56] Jérôme David and JérômeEuzenat. Comparison betweenontology distances (preliminary results). In 7th InternationalConference on The Semantic Web (ISWC), pages 245–260,Berlin, Heidelberg, 2008. Springer-Verlag

[57] Jérôme Euzenat, Carlo Allocca, Jérôme David, Mathieu d'Aquin, Chan Le Duc, and OndrejSvab-Zamazal. Ontologydistances for contextualisation. Deliverable, NeOn,2009.

[58] Mathieu d'Aquin. Formally measuring agreement and disagreementin ontologies. In 5th International Conference onKnowledge Capture, pages 145–152, 2009.

[59] Ricardo Baeza-Yates and BerthierRibeiro-Neto. ModernInformation Retrieval. Addison Wesley, May 1999.

[60] Stephen E. Robertson and Karen Spärck Jones.Relevanceweighting of search terms. Journal of the American Society forInformation Science, 27(3) :129–146, 1976.

[61] Thomas Bayes. An essay towards solving a problem in thedoctrine of chances. Philosophical Transactions of the RoyalSociety of London, 53 :370–418, 1763.

[62] Stephen E. Robertson, Steve Walker, and MichelineHancock-Beaulieu. Experimentation as a way of life : Okapiat trec. Information Processing and Management: 95–108,2000.

[63] Jérôme David, JérômeEuzenat, and Ondˇrej̆ŠvábZamazal.Ontology similarity in the alignment space.In 9th InternationalSemantic Web Conference, 2010.

[64] Silvio Peroni, Enrico Motta, and Mathieu d'Aquin.Identifyingkey concepts in an ontology, through the integrationof cognitive principles with statistical and topological measures.In 3rd Asian Semantic Web Conference, pages 242–256,2008.