N° d'ordre : 01/2009-D / IN

République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université des Sciences et de la Technologie Houari Boumediene
Faculté d'Electronique et d'Informatique

# Thèse

Présentée pour l'obtention du diplôme de Doctorat

EN : INFORMATIQUE

*Spécialité : Intelligence Artificielle et Bases de Données Avancées*

Par : **Samir KECHID**

Sujet :

# Intégration du modèle utilisateur dans un système de recherche d'information distribuée

Soutenue le 11/04/2009, devant le jury composé de :

| | | |
|---|---|---|
| **Mme. A. AISSANI-MOKHTARI** | Professeur, USTHB. | **Président** |
| **Mme. H. DRIAS** | Professeur, USTHB. | **Directeur de Thèse** |
| **Mr. M. BOUGHANEM** | Professeur, Univ Paul Sabatier de Toulouse. | **Examinateur** |
| **Mr. Y. AIT AMEUR** | Professeur, ENSMA-POITIERS. | **Examinateur** |
| **Mr. S. LARABI** | Professeur, USTHB. | **Examinateur** |
| **Mr. A. BELKHEIR** | Maître de Conférence, USTHB. | **Examinateur** |
| **Mme. T. TEBIBEL** | Maître de Conférence, INI. | **Examinateur** |

# Résumé

Notre but principal dans cette thèse était de concevoir une approche permettant d'intégrer le profil de l'utilisateur dans son accès à de multiples sources d'information. Pour assurer cet objectif, nous nous sommes investis à l'intégration du profil de l'utilisateur dans le processus de sélection de sources d'information et dans le processus de fusion des résultats retournés par les sources sélectionnées. Notre contribution consiste à répondre aux questions suivantes ; (1) comment définir, acquérir, et mettre à jour le profil de l'utilisateur ; (2) comment intégrer ce profil pour personnaliser le processus de sélection de sources ; (3) comment intégrer ce profil pour personnaliser le processus de fusion des résultats des sources sélectionnées.

Nous avons commencé notre approche par la définition d'un profil utilisateur constitué de plusieurs dimensions ; (1) son identité ; (2) son historique des recherches ; (3) son centre d'intérêts ; (4) son historique des sources sélectionnées. Puis nous avons défini les techniques nécessaires permettant d'exploiter ce profil dans la personnalisation du processus de sélection de sources et du processus de fusion des résultats des sources sélectionnées. L'approche a été évaluée en utilisant les 08 moteurs de recherches suivants : GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chaque moteur de recherche présente une source d'information.

Cependant, l'approche présente des insuffisances à savoir ; (1) un utilisateur peut avoir plusieurs centres d'intérêts différents, pas un seul seulement ; (2) l'utilisation du centre d'intérêts seulement pour un utilisateur n'est pas suffisante pour décrire ses besoins en information, un utilisateur peut avoir des préférences sur la nature (critères) des documents qu'il veut consulter. Nous avons pallié à ces insuffisances en proposant des techniques permettant de prendre en compte ; (1) les différents centres d'intérêts d'un utilisateur ; (2) les préférences de l'utilisateur relatives aux critères des documents des sources, à savoir ; le type, la langue, la fraîcheur et le coût d'un document. Dans cette approche la pertinence des résultats est relative à trois mesures correspondantes à la requête de l'utilisateur, son centre d'intérêts et ses préférences. L'approche a été évaluée en utilisant plusieurs sources d'informations réparties sur des différents secteurs.

L'approche présente d'autres insuffisances concernant le temps de réponse et l'extensibilité du système. Afin d'améliorer le temps de réponse et l'extensibilité du système, nous avons adapté une architecture à base d'agents à notre approche précédente.

**Mots Clés :** Recherche d'Information Distribuée, sélection de sources, fusion des résultats des sources, profil utilisateur, profil source, système multi-agents.

# *Remerciements*

*L'achèvement de tout travail mené sur plusieurs années procure une grande satisfaction. Il est l'occasion de se remémorer les étapes passées et les personnes qui y ont contribué.*

*Tout d'abord je remercie Dieu le tout puissant de m'avoir donné le courage de continuer ce travail.*

*Aussi, j'adresse mes sincères remerciements à ma Directrice de thèse Madame Drias Habiba, Professeur à l'Université des Sciences et de la Technologie Houari Boumediene (USTHB) pour avoir dirigé mes recherches. Ses conseils, ses critiques, ainsi que la confiance qu'elle m'a toujours témoignée m'ont été d'un grand apport tout au long de mes recherches. Qu'elle soit assurée de mon très grand respect.*

*Je tiens à exprimer ma profonde gratitude à Monsieur Mohand Boughanem, professeur à l'Université Paul Sabatier de Toulouse et à Madame Lynda Tamine Lechani, Maître de Conférences à l'Université Paul Sabatier de Toulouse qui m'ont accueilli plusieurs fois au sein de leur équipe et m'ont apporté une aide très précieuse lors de la réalisation de ce travail.*

*Je souhaite exprimer ma gratitude à Madame Zaia Alimazighi, Professeur à l'USTHB et Monsieur Abdelkader Belkheir, Maître de conférence à l'USTHB, qui ont accepté d'évaluer ce travail.*

*Je tiens également à remercier Madame Aicha Aissani-Mokhtari, Professeur à l'USTHB, Monsieur Mohand Boughanem, Professeur à l'Université Paul Sabatier de Toulouse, Monsieur Yamine Ait-Ameur, Professeur à l'Ecole Nationale Supérieure de Mécanique et d'Aérotechnique (ENSMA) de Poitiers, Madame Thouraya Tebibel, Maître de conférence à l' Institut National d'Informatique (INI), Monsieur Slimane Larabi, Professeur à l'USTHB et Monsieur Abdelkader Belkheir, Maître de conférence à l'USTHB, pour l'intérêt qu'ils ont porté à mes travaux en examinant cette thèse et pour l'honneur qu'ils me font en participant à ce jury.*

*Mes remerciements vont de même à tous les membres du laboratoire LRIA et particulièrement aux membres de l'équipe IACHM.*

*Je remercie du fond du coeur et avec un grand amour mes parents et ma femme qui n'ont jamais cessé de croire en moi pendant toutes mes années d'études. Je souhaite exprimer ma profonde gratitude à toute ma famille pour leur soutien indéfectible.*

*Enfin, j'offre mes sincères remerciements à toutes les personnes qui ont participé de près ou de loin à l'aboutissement de ce travail. A toutes et à tous je leur dis merci.*

*Je dédie ce modeste travail*
*à mon petit ange Houda*

# Sommaire

## Chapitre Introductif



## Première partie : Recherche d'information : Etat de l'art

### Chapitre 1 : La recherche d'information



### Chapitre 2 : La recherche d'information distribuée

## Chapitre 3 : Accès personnalisé à l'information

# Deuxième partie : Contributions aux systèmes de recherche d'information distribuée

## Chapitre 4 : Accès personnalisé à de multiples sources d'information



## Chapitre 5 : Intégration des préférences utilisateur dans un SRID



## Chapitre 6 : Un système multi-agents pour l'intégration du modèle utilisateur dans un SRID

## Troisième partie: Validation Expérimentale

## Chapitre 7 : Les prototypes mis en oeuvre



## Conclusion et perspectives

# Chapitre Introductif

## 1. Introduction

Le développement des technologies de l'information et des communications et le nombre croissant de secteurs de l'activité humaine a aujourd'hui pour conséquence la production d'un volume sans précédent d'informations. Outre, l'interconnexion progressive des sites par le biais de vastes réseaux informatiques comme l'Internet, ainsi que la normalisation des techniques d'accès et de production de ces informations (URL, HTML, XML…) mettent aujourd'hui une quantité très importante de documents à la portée de tout internaute.

Face à cette considérable masse d'information, il est devenu difficile voire impossible à l'utilisateur de trouver l'information pertinente dont il a besoin. Cette difficulté d'accès à l'information a donné naissance à plusieurs outils de recherches d'informations, dans le but d'aider l'utilisateur à trouver l'information pertinente qu'il cherche. On y retrouve les outils de recherche par mots clés (les moteurs de recherche), par thème (les outils de recherche thématiques), par région (les outils de recherche géographiques) ou par l'exploitation de plusieurs moteurs de recherche (méta-moteurs).

Dans cette perspective, l'utilisateur joue un rôle particulièrement actif du fait que la recherche d'information est réalisée sur la base de requêtes qu'il définit explicitement et soumet à un ou plusieurs outils de recherche. Ce type d'approche, que l'on qualifie quelquefois de collecte active d'information, est une approche qui, par essence, laisse une place importante à l'utilisateur mais qui, de ce fait, lui impose également des contraintes de compétence (l'utilisateur doit savoir aussi précisément que possible ce qu'il cherche pour qu'il soit capable de formuler des requêtes pertinentes) et de disponibilité (l'utilisateur pilote le processus de recherche et doit être de ce fait prêt à investir dans cette activité le temps nécessaire).

Dans cette thèse, nous nous intéressons aux systèmes de recherche d'information sur Internet (SRIs). Ces derniers se divisent en deux catégories, à savoir ; les systèmes de recherche d'information centralisée (SRICs) et les systèmes de recherche d'information distribuée (SRIDs). Dans cette thèse nous nous intéressons particulièrement aux SRIDs et précisément aux étapes de sélection de serveurs (sources) d'information et de fusion des résultats des serveurs.

Ce chapitre introductif est organisé comme suit ; dans la section 1.1 nous présentons la Recherche d'Information (RI) d'une manière générale, les notions de base d'un SRI, le processus de SRIC, le processus de SRID et l'apport des SRIDs aux SRICs ; nous détaillons dans la section 1.2 la problématique de cette thèse ; la section 1.3 présente les objectifs visés ; nous présentons nos contributions dans la section 1.4. Nous terminons ce premier chapitre par le plan de cette thèse dans la section 1.5.

## 1.1. La recherche d'information

La recherche d'information (RI) [Frakes 1992] [Grossman et Fieder 1998] [Salton 1971] [Yates et Neto 1999] est l'ensemble des techniques permettant de gérer des textes, ou des documents. Gérer des textes, ou des documents, implique stocker, rechercher et explorer des documents pertinents. Plusieurs concepts clés sont véhiculés autour de la RI :

- *Base documentaire :* la base documentaire (ou fonds documentaire) constitue l'ensemble des informations exploitables et accessibles. Elle est constituée d'un ensemble de documents. Dans le cas général et pour des raisons d'optimalité, la base constitue des représentations très simplifiées mais suffisantes de ces documents. Ces représentations sont étudiées de telle sorte que la gestion (ajout, suppression d'un document) et l'interrogation (recherche) de la base se font dans les meilleures conditions de coût. Dans la suite de cette thèse, nous utilisons indifféremment les termes, serveur, source ou collection pour designer une base documentaire.

- *Document :* un document constitue l'information élémentaire d'une base documentaire. Le document, d'un point de vue fonctionnel, est défini comme étant une entité atomique qui peut être recherchée, retrouvée, et consultée, sans pour autant être nécessairement physiquement sauvegardée comme une entité unique. D'un point de vue logique, un document est une entité véhiculant une ou plusieurs informations, présentées sous des formes variées (texte, son, image, vidéo, multimédia). Nous utilisons dans la suite de cette thèse indifféremment les termes document et information.

- *Requête :* une requête constitue l'expression du besoin en information de l'utilisateur. Elle contient en général un ensemble de mots clés. On la rencontre également sous forme d'une phrase ou d'un paragraphe. Elle représente la description des spécifications des documents souhaités. Mais le plus souvent cette description est courte et ambiguë et ne spécifie pas tous les détails du besoin d'information. La requête est exprimée dans un langage d'interrogation parmi lesquels nous citons :

    - des requêtes simples de mots clés ;

- des requêtes booléennes où les mots clés sont liés par des opérateurs booléens (AND, OR, NOT) ;
- des requêtes structurées basées sur des attributs tels que les noms d'auteurs la date de parution … etc. ;
- des requêtes sous forme d'expressions régulières ;
- des requêtes complexes qui englobent les types précédents.

- *Pertinence système :* c'est le degré de pertinence de l'information calculé par le système de RI.

- *Pertinence utilisateur :* c'est le degré de pertinence de l'information correspondant au jugement de l'utilisateur.

### *1.1.1. Un système de recherche d'information*

Un système de recherche d'information (SRI) intègre un ensemble de modèles et de processus permettant de sélectionner des informations pertinentes en réponse au besoin d'un utilisateur exprimé à l'aide d'une requête [Belkin et Croft 1992]. Ces processus permettent :

- la représentation des documents et des besoins, ce processus est appelé indexation ;
- l'interrogation, la recherche et la sélection des informations répondant aux besoins d'un utilisateur, ce processus est appelé appariement requête-document ;
- la reformulation de requête.

La figure 1 illustre l'architecture d'un système de recherche d'information.

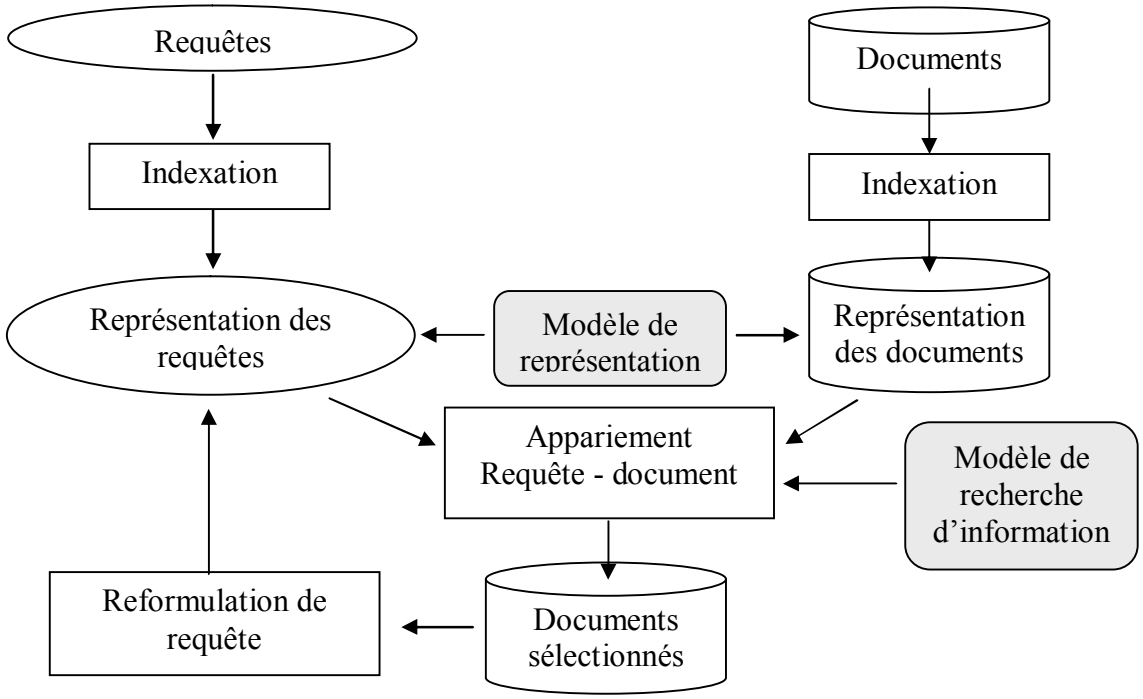


**Figure 1.** Architecture générale d'un système de recherche d'information

Les composants de ce système sont décrits comme suit :

- *L'indexation :* afin d'assurer la recherche dans des conditions acceptables en coût, une étape primordiale précédant l'étape de recherche effective d'information doit s'effectuer. Celle-ci consiste à analyser le document lors de l'organisation du fonds documentaire afin de produire un ensemble de mots clés que le système pourra gérer aisément, pour pouvoir les utiliser dans le processus de recherche ultérieur. Cette opération est appelée *indexation* [Salton 1971]. Un *index* est une structure qui permet d'associer, à chaque terme (mot clés) d'indexation, la liste des documents qui contiennent ce terme. En plus de la simple relation d'appartenance d'un terme à un document, un index peut fournir d'autres informations, comme ; le poids du terme dans le document, la co-occurrence des termes dans un document et la position du terme dans le document. L'ensemble des termes extraits de tous les documents est stocké dans une structure spécifique appelée ***Fichier Inverse.***

- *L'appariement requête-document :* une fois l'indexation des requêtes et des documents est effectuée, l'appariement entre la requête et le document peut désormais s'effectuer. L'appariement consiste à détecter les documents pertinents pour la requête posée, et éventuellement calculer le degré de cette pertinence appelé aussi score ou *RSV (Retrieval Status Value)*. Ce score sert à trier la liste des documents retournés.

- *La pondération des termes :* l'élément fondamental dans un système de recherche d'information est la technique utilisée pour pondérer les termes [Sparck Jones 1971] [Sparck Jones 1979]. La pondération des termes est utilisée dans l'opération d'indexation, permettant d'associer à chaque terme son poids dans le document. La plupart des techniques de pondération des termes sont basées sur les facteurs *Tf* et *Idf* :
    - *Tf (term frequency) :* cette mesure est proportionnelle à la fréquence du terme dans le document. Elle est souvent exprimée selon l'une des déclinaisons suivantes :
        - *Tf :* utilisation brute ;
        - *a + log(Tf) ;*
        - *[0/1]* : Présence, absence ;
        - *0.5 + 0.5 * (tf / Max(tf) )* ;

    - *Idf (Inverse of Document Frequency)* : mesure l'importance d'un terme dans toute la collection. Un terme trop fréquent dans la collection ne doit pas avoir le même impact sur la collection qu'un terme moins fréquent. Elle est exprimée selon l'une des déclinaisons suivantes :

- *Log (N /df) ;*
- *Log ( (N-df) / df).*

Où *df* est le nombre de documents contenant le terme et *N* est le nombre de documents dans toute la collection.

- *Le modèle de recherche d'information (MRI) :* lorsque l'on parle de MRI, on englobe les notions d'index et de fonction d'appariement. La fonction d'appariement utilise des informations contenues dans l'index. Ce dernier doit donc être construit en rapport avec cette fonction. Le MRI prédit les documents qui sont pertinents (et calcule leur degré de pertinence) et ceux qui ne le sont pas. Il existe essentiellement quatre modèles de RI classiques : le modèle booléen, le modèle vectoriel, le modèle probabiliste et les modèles logiques. Ces modèles sont détaillés dans [Ricardo 1999].

- *La reformulation de requête :* Compte tenu des volumes croissants des bases d'information, retrouver les informations pertinentes en utilisant seulement la requête initiale de l'utilisateur est une tâche quasi-impossible. Afin de rapprocher au mieux la pertinence système de la pertinence utilisateur, une étape de reformulation de la requête est souvent utilisée dans les systèmes de recherche d'information. La reformulation de la requête consiste à modifier la requête de l'utilisateur par ajout de termes significatifs et/ou ré-estimation de leurs poids.

- *Un moteur de recherche :* est une machine spécifique (matérielle et/ou logicielle) capable de construire, sur la base d'une collection de documents, un index approprié, et ensuite offrir à l'utilisateur le moyen de rechercher dans cette collection. Après la saisie d'une requête, le moteur applique la fonction d'appariement entre la requête et les documents indexés, et retourne à l'utilisateur une liste de résultats.

- *La performance et l'efficience :* la performance d'un SRI est sa capacité à satisfaire l'utilisateur en terme de pertinence des documents retournés. Nous parlerons de l'efficience d'un SRI lorsqu'il s'agit d'évaluer tous les critères autres que la performance, à savoir, le temps de réponse, le coût et le temps d'indexation, la facilité d'utilisation, la simplicité de l'interface utilisateur … etc.

- *Le bruit et le silence :* Le bruit, dans une réponse représente tous les documents non pertinents à la requête. Le silence, dans une réponse, représente tous les documents pertinents de la collection du système n'ayant pas été retrouvés pour cette requête.

### *1.1.2. Un système de recherche d'information centralisée*

Un système de recherche d'information centralisée (SRIC) fonctionne d'une manière totalement centralisée. Sa particularité est l'unicité de son index. Tous les documents de sa

collection sont indexés dans le même fichier ou la même structure qui sert de base pour la recherche. La figure 2 illustre l'architecture générale d'un SRIC :

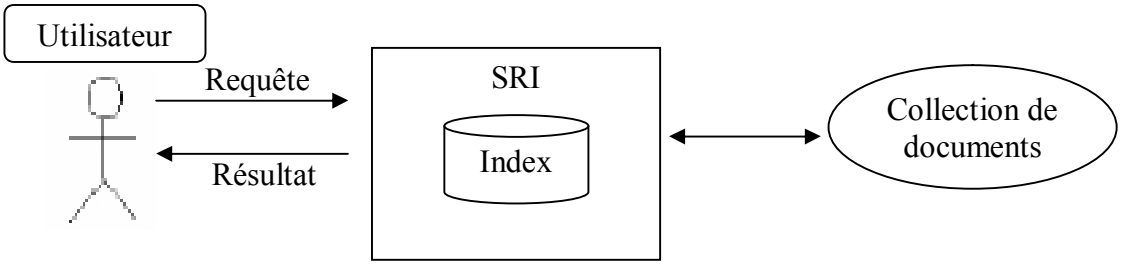


**Figure 2.** Architecture générale d'un système de recherche d'information centralisée

L'unicité de l'index dans le SRIC possède des inconvénients majeurs, face au rythme de croissance des données sur Internet :

1. la puissance et la capacité de sauvegarde n'évoluent pas avec la même vitesse que les données sur Internet ;

2. il est impossible de se procurer tous les documents existants. En effet, il existe sur Internet des documents inaccessibles ;

3. il est très coûteux de construire un index pour la totalité des documents collectés, car il est nécessaire de les charger localement afin de les indexer. Or, ce chargement ralentit le fonctionnement du réseau et les serveurs à partir desquels le chargement s'effectue ;

4. la mise à jour de l'index unique n'est pas facile, et cette tâche peut durer longtemps. Or, certains sites, comme par exemple les sites d'information, mettent leurs documents à jour quotidiennement ;

5. l'unicité de la méthode d'indexation malgré l'hétérogénéité des sources des documents est inadéquate. En effet, des documents rédigés dans des langues différentes sont indexés de la même façon.

### *1.1.3. Un système de recherche d'information distribuée*

Un système de recherche d'information distribuée (SRID) a le même but qu'un SRIC, à savoir, satisfaire le besoin de l'utilisateur en information, sauf qu'un SRID exploite plusieurs serveurs d'information, dont chaque serveur indexe ses documents séparément des autres serveurs, ce qui n'est pas le cas d'un SRIC. Les SRIDs ont été conçus pour assurer une grande couverture de l'information et pour pallier les problèmes rencontrés par les SRICs. Un SRID est constitué d'un courtier qui communique avec un ensemble de serveurs. Chaque serveur correspond à un SRI, qui contient un index et un moteur de recherche qui exploite cet index. La figure 3 présente l'architecture générale d'un SRID.

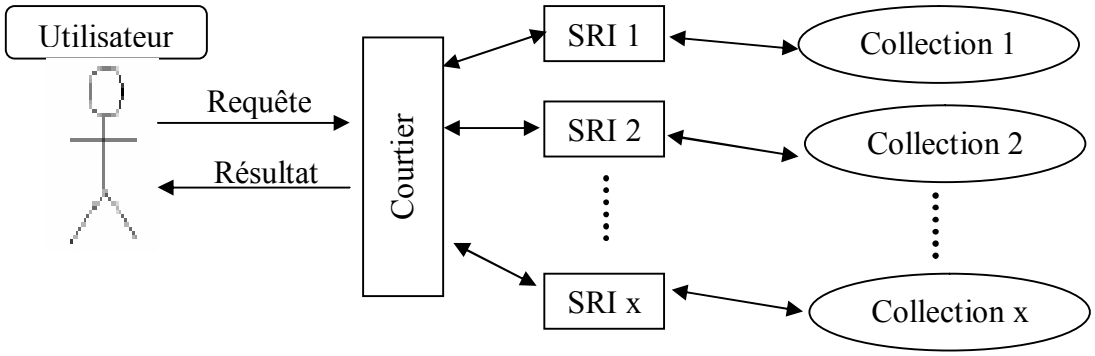


**Figure 3.** Architecture générale d'un système de recherche d'information distribuée

En général, un SRID est composé d'un courtier qui représente le cœur du système. L'utilisateur formule son besoin d'information sous forme de requête et la soumet au courtier. Le courtier choisit un certain nombre de serveurs qu'il juge aptes à répondre d'une façon satisfaisante à la requête. Cette opération est appelée sélection de serveurs. Puis, le courtier transmet la requête aux serveurs ainsi sélectionnés. Les serveurs interrogés transmettent leurs listes de documents trouvés. Le courtier se charge alors de fusionner ces listes afin de constituer une liste unique qui sera présentée à l'utilisateur. Cette opération est appelée fusion des résultats des serveurs.

Les SRIDs possèdent l'avantage d'éviter à l'utilisateur de consulter indépendamment chaque serveur. Imaginons un utilisateur qui cherche des articles scientifiques sur un domaine particulier. Il est fort probable que les articles recherchés soient disponibles sur plusieurs sites d'universités ou d'éditeurs de revues scientifiques. Dans ce cas, l'utilisateur peut envoyer sa requête à chacun de ces sites (en supposant qu'il les connaisse) et examiner les documents retournés par chacun d'eux, ceci est une tâche fastidieuse. Premièrement, l'utilisateur devra se familiariser avec les diverses interfaces d'interrogation de chaque site. Deuxièmement, il devra consulter chaque liste retournée. Enfin, l'utilisateur sera submergé par le nombre de documents retournés si le nombre de sites interrogés s'avère important.

### *1.1.4. L'apport des SRIDs par rapport aux SRICs*

Les SRIDs, en distribuant le processus d'indexation et de recherche, apportent des solutions aux problèmes posés par les SRICs :

1. La couverture maximale de l'information disponible n'est plus une tâche qui incombe à un seul serveur. La couverture obtenue est l'union des couvertures des serveurs du SRID. Un SRID permet également d'exploiter efficacement les ressources matérielles puisque la tâche de stockage est distribuée sur un ensemble de serveurs.

2. Une source de documents ne souhaitant pas être indexée par d'autres serveurs, peut installer son propre SRI et s'insérer dans le SRID. De cette façon, ses documents sont visibles sans qu'ils ne soient pour autant indexés par des serveurs étrangers. C'est le cas par exemple des serveurs payants, qui souhaitent être référencés sans que le contenu de leurs documents ne soit consultable.
3. Les serveurs adhérant au SRID, sont responsables de la construction et de la mise à jour de leur collection de documents. La décentralisation de ces deux tâches les rend moins coûteuses.
4. Chaque serveur étant responsable de l'indexation de sa collection, il est libre de choisir la méthode d'indexation appropriée à sa collection de documents, par exemple en fonction de la langue.

## 1.2. Problématique

Nous détaillons dans cette section les problèmes posés par la recherche d'information distribuée (RID). Deux principaux problèmes sont posés ; le premier problème est la sélection des sources d'information à interroger pour répondre au besoin de l'utilisateur en information dans le but de réduire l'espace de recherche sans diminuer l'efficacité des résultats ; le second problème est celui de la fusion des résultats retournés par les différentes sources interrogées afin de produire une seule liste de résultats qui sera présenter à l'utilisateur du système de recherche d'information distribuée (SRID).

### *1.2.1. Sélection des sources*

La sélection des sources, consiste à sélectionner parmi les sources d'information accessibles au SRID, les sources capables de fournir des informations pertinentes aux besoins de l'utilisateur. La sélection des sources a plusieurs buts qui peuvent être regroupés en deux catégories. Premièrement, le but d'efficience, en d'autres termes, réduire le coût de la recherche en diminuant le nombre de sources à interroger. Il est important de réduire la quantité d'information nécessaire au processus de sélection, pour que le système soit extensible. Deuxièmement, le but de performance en interrogeant le plus petit nombre de sources, sans détériorer la performance du système voire en augmentant sa performance si l'on écarte uniquement les sources ne possédant aucun document pertinent.

La plupart des méthodes de sélection existantes adoptent la méthode de classement des sources selon leur degré de pertinence. Dans ce cas, un score est associé à chaque source. Celui-ci est calculé différemment d'une méthode à une autre selon les données et les techniques adoptées. Les sources étant classées selon leur score, quelques possibilités de sélection peuvent être envisagées. Par exemple, interroger les $n$ premières sources les mieux classées, ou interroger les sources dont le score dépasse un certain seuil fixé, noté $s$.

Mais alors comment choisir l'une ou l'autre de ces constantes *n* ou *s* afin de bien dissocier les sources pertinentes de celles qui ne le sont pas ?

Plusieurs approches de sélection de sources on été développées. L'idée générale de ces approches est de calculer un score entre chaque source et la requête de l'utilisateur puis classer et sélectionner les sources selon les meilleurs scores. Ces méthodes diffèrent par la technique utilisée pour le calcul de ce score. Nous pouvons citer : la méthode *CORI (Collection Retrieval Inference network)* [Callan *et al* 1995], la méthode CVV *(Cue Validity Variance)* [Yumono et Lee 1997], la méthode *GLOSS (Clossary of Servers Server)* [Gravano et al 1999], la méthode LWP (Light Weight Probes) [Hawking et Thistlewaite 1999], la méthode DTF (*Decision-Theoretic Framework*) [Fuhr 1999], la méthode UUM (*Unified Utility Maximization strategy*) [Si et Callan, 2004], *et* d'autres ….

### *1.2.2. Fusion des résultats provenant des différentes sources interrogées*

Une fois l'étape de la sélection de sources franchie, l'ensemble des sources sélectionnées sont interrogées. Il reste alors à sélectionner et fusionner dans une seule liste les documents à retenir parmi ceux retournés par chaque source interrogée. Dans ce cas, l'objectif est de retrouver le plus grand nombre de documents pertinents de chaque source tout en minimisant le nombre de documents non pertinents.

Plusieurs approches de fusion ont été avancées. Ces approches se basent sur le score de chaque document pour les trier et les fusionner, cependant elles se distinguent par la technique et les données utilisées pour le calcul de ce score. Nous pouvons citer, les méthodes basées sur les fonctions de combinaison CombMAX, CombMIN, CombMED, CombSUM, CombANZ et CombMNZ [Fox et Shaw 1994]. La stratégie, *round robin* nommée aussi *à chacun son tour* [Voorhees et al 1995b], la stratégie *fusion par le score* appelée aussi dans la littérature *Raw Score Merging* [Kwok et al 1995], l'approche *CORI* [Callan et al 995], la stratégie *SSL (The Semi-Supervised Learning)* [Si et Callan. 2003] [Si et Callan 2005]

## 1.3. Objectifs

Nous avons cité dans la section précédente les deux problèmes auxquels on est confronté lorsque l'on veut concevoir un SRID (sélection de sources et fusion des résultats des sources). Les méthodes de sélection de sources et fusion des résultats des sources développées dans la littérature sont effectuées sur la base des informations contenues dans les sources et la requête. L'utilisateur, ses préférences, ses centres d'intérêts récurrents ne sont pas pris en compte. Notre but est de proposer une approche permettant d'intégrer l'utilisateur dans le processus de recherche d'information distribuée précisément dans les phases de sélection de sources et de fusion des résultats des sources

Il s'est avéré avec le temps que la requête de l'utilisateur seule, ne peut pas exprimer toujours son besoin en information. Par exemple, une requête formulée par un utilisateur contenant le mot *virus*, ne peut pas être compréhensible, puisqu'elle a deux significations. Il peut s'agir de virus dans les domaines informatique ou médical. Pour qu'elle soit plus compréhensible, il est nécessaire de connaître le domaine de l'utilisateur. Nous pouvons voir aussi le cas d'un utilisateur qui formule une requête pour chercher des documents sur un certain domaine. Il est évident que le résultat retourné ne sera pas le même suivant que l'utilisateur est un spécialiste ou débutant du domaine.
A partir de ces simples exemples, il est clair que dans certains cas, il faut avoir des connaissances sur l'utilisateur pour bien comprendre sa requête. La complexité qui se pose, est de cerner les informations qu'il faut savoir sur l'utilisateur et comment les utiliser pour bien comprendre sa requête.

L'intégration de l'utilisateur dans la RI a été déjà abordée dans la littérature, ce qui a donné naissance à un nouvel axe, connu par la personnalisation de l'information. La personnalisation de l'information se définit, entre autres, par un ensemble de préférences individuelles représentées par des couples (attribut, valeur), par des ordonnancements de critères ou par des règles sémantiques spécifiques à chaque utilisateur ou communauté d'utilisateurs [Bouzeghoub et Kostadinov 2004]. Ces modes de spécification servent à décrire le centre d'intérêts de l'utilisateur, le niveau de qualité des données qu'il désire ou des modalités de présentation de ces données. L'ensemble de ces informations est représenté dans un modèle d'utilisateur appelé souvent *profil*.

Un profil regroupe l'ensemble des connaissances nécessaires à une évaluation efficace des requêtes et à une production d'une information pertinente adaptée à chaque utilisateur. Il convient donc de distinguer la notion de profil de la notion de requête. Un profil peut être défini comme un modèle personnalisé d'accès à l'information alors qu'une requête est l'expression d'un besoin circonstancié que l'utilisateur souhaite voir satisfait en tenant compte de son profil. Un profil a un caractère plus invariant que les requêtes même si le centre d'intérêts et les préférences de l'utilisateur peuvent légitimement évoluer.

Les approches de personnalisation de l'information développées dans la littérature, ont montré l'intérêt d'intégrer le profil dans les requêtes de l'utilisateur. Il existe plusieurs exemples de systèmes de personnalisation. Nous présentons dans ce qui suit quelques exemples des systèmes de personnalisation d'une manière générale. La personnalisation de l'information sera bien détaillée dans le chapitre 2. Nous pouvons donner à titre d'exemple :

***Letizia*** [Lieberman 1995] : un système qui assiste l'utilisateur en lui recommandant des liens à explorer en fonction de la page en cours d'être visitée par l'utilisateur. La représentation du profil dans cette approche n'est pas détaillée. Cependant, comme les

documents sont représentés selon le modèle vectoriel (vecteurs de mots pondérés), nous supposons que même le profil est représenté ainsi.

***WebACE*** [Boley et al 1998] : est un assistant à la navigation. Il construit le profil de l'utilisateur à partir des documents qui l'intéressent, en incluant des paramètres comme : le nombre de fois qu'un document est visité et le temps dépensé dans la lecture du document. WebACE sépare les intérêts de l'utilisateur en utilisant des clusters. Ainsi les clusters sont utilisés pour générer des requêtes et chercher des documents similaires.

***WebWatcher*** [Armstrong et al 1995] : un autre assistant à la navigation, qui recommande des pages dans des sites particuliers, il agit comme un guide, dirigeant l'utilisateur aux documents intéressants sur ce site. Initialement l'utilisateur saisit les mots clés qui représentent son centre d'intérêts. Ensuite, le système observe les chemins explorés par l'utilisateur dans le site, et lui permet d'indiquer les pages qui l'intéressent, et d'indiquer si son besoin en information est satisfait. Ces informations seront utilisées pour des futures recommandations.

***SiS (Stuff I've Seen)*** [Dumais et al 2003] : un système qui permet de personnaliser des informations déjà consultées par l'utilisateur, selon différentes formes (Email, page Web, page intranet ….). Le système se déroule en deux phases ; la première consiste à créer un index unifié pour toutes les informations consultées indépendamment de la source et de la forme, qui constitue le contexte de l'utilisateur ; la deuxième phase exploite ce contexte, en effectuant un accès personnalisé à des sources d'information déjà utilisées. Le système SIS permet à l'utilisateur de définir des critères tels que la date, auteur et score, sur lesquels il se base pour présenter les résultats.

## 1.4. Contributions

Nous nous sommes intéressés dans cette thèse à l'accès personnalisé à de multiples sources d'information. Nos contributions majeures s'articulent autour des travaux suivants :

1- Proposition d'une approche permettant : (1) la définition d'un profil utilisateur constitué de plusieurs dimensions à savoir ; son identité, son historique des recherches, son centre d'intérêt, son historique des sources sélectionnées ; (2) la définition des techniques nécessaires permettant d'exploiter ce profil dans la personnalisation du processus de sélection de sources et du processus de fusion des résultats des sources sélectionnées.

2- Proposition d'une approche d'intégration des préférences de l'utilisateur sur des critères des documents des sources, à savoir ; le type, la langue, la fraîcheur et le coût d'un document.

3- Proposition d'une architecture à base d'agents pour l'intégration du modèle utilisateur dans un SRID.

4- Implémentation des prototypes relatifs aux approches proposées.

## 1.5. Plan de la thèse

La suite de la présentation de notre travail se divise en trois parties : la première constituée des chapitres 1, 2 et 3 présente un état de l'art des concepts et des techniques de la RI, la RID et la RIP que nous avons utilisées pour élaborer une solution aux problèmes posés. La deuxième est constituée des chapitres 4, 5 et 6, où nous présentons nos majeures contributions. La troisième est consacrée à la validation et la mise en œuvre des approches proposées.

- Le chapitre 1 présente un état de l'art sur le domaine de la recherche d'information
- Le chapitre 2 décrit les étapes d'un processus de recherche d'information distribuée (RID). Dans ce chapitre nous présentons les problèmes rencontrés, en mettant l'accent sur le problème de sélection de sources d'information et le problème de fusion des résultats des sources. Nous présentons par la suite une synthèse des différentes approches existantes dans la littérature concernant les étapes de sélection de source, la fusion des résultats ainsi que les approches d'évaluation des SRIDs.
- Le chapitre 3 est consacré à la recherche d'information personnalisée. Nous présentons l'utilité d'intégrer l'utilisateur via son profil dans le processus de recherche d'information. Nous présentons par la suite une synthèse des modèles et techniques qui existent dans ce domaine.
- Les chapitres 4 et 5 présentent nos contributions pour l'accès personnalisé à de multiples sources d'information. Les différentes étapes abordées sont ; la modélisation de l'utilisateur, la modélisation de la source d'information, le processus d'évaluation de la requête utilisateur, le processus de sélection des sources d'information, ainsi que le processus de fusion des résultats des sources. Afin de valider nos contributions, nous présentons nos expérimentations et évaluations, en commentant les résultats obtenus.
- Le chapitre 6 présente une approche d'adaptation d'un système multi-agents à notre approche précédente (séquentielle) présentée dans le chapitre 5.
- La troisième partie présente les prototypes que nous avons implantés.

# Première partie

# La recherche d'information
# Etat de l'art

# Chapitre 1

# La recherche d'information

## 1. Introduction

La recherche d'information (RI) est un domaine très utile en informatique. Il s'intéresse à l'acquisition, l'organisation, le stockage, la recherche et la distribution de l'information. L'objectif d'un système de recherche d'information est de fournir des informations pertinentes pour des requêtes écrites dans un langage libre. Une requête est une représentation structurée du besoin en information (c'est l'expression mentale d'un utilisateur). Une information est dite *pertinente* si elle satisfait les besoins de l'utilisateur. Cette notion est subjective car elle dépend de l'utilisateur (la requête est elle une bonne représentation du besoin ?) et du système, donc c'est difficile à automatiser *remplacer l'utilisateur par un système*.

Avoir une information pertinente revient à minimiser le bruit (l'ensemble des documents retrouvés) et le silence (l'ensemble des documents pertinents). [Lamrous 1999].

Un système de recherche d'information nécessite la conjugaison de modèles et algorithmes permettant la représentation, le stockage, la recherche et la visualisation d'information. L'objectif fondamental de la recherche d'information consiste à mettre en œuvre un mécanisme d'appariement entre une requête utilisateur et des documents d'une base documentaire afin de restituer l'information pertinente. L'élaboration d'un processus de recherche d'information pose alors des problèmes liés tant à la modélisation qu'à la localisation de l'information pertinente. En effet, la recherche d'information induit un processus d'inférence de la sémantique véhiculée par l'objet de la requête, en se basant sur une description structurelle des unités d'informations.

Nous présentons dans ce chapitre les modèles et les techniques du domaine de la recherche d'information.

### 1.1. Les Notions de base

Plusieurs concepts clés sont véhiculés autour de la RI à savoir : une base documentaire, un document, une requête, un processus d'indexation, un processus d'appariement requête-

document, un processus de pondération des termes et un modèle de recherche d'information.

Pour bien illustrer l'utilité de ces concepts dans un système de recherche d'information voir la section 1.1 du chapitre introductif où nous avons présenté une étude des concepts de la RI.

# 2. Les modèles de recherche d'information

Un modèle de recherche d'information est formellement décrit par un quadruple **[D, Q, F, $R(q_i , d_j)$]** [Yates et Neto, 1999] [Tamine 2000 ] où :

**D** : Ensemble des représentants de documents de la collection
**Q** : Ensemble de représentants des besoins en informations
**F** : Schéma du support théorique de représentation des documents, requêtes et relations associées
**$R(q_i , d_j)$** : Fonction d'ordre associée à la pertinence

La définition d'un modèle de recherche d'information induit ainsi la détermination d'un support théorique comme base de représentation des unités d'informations et de formalisation de la fonction pertinence du système. De très nombreux modèles sont proposés dans la littérature. Le présent paragraphe a pour objectif de présenter les principaux modèles de base et modèles dérivés construits sur chacun d'eux.

Nous adoptons dans la suite, les principales notations suivantes :

$Q_k$ : kième requête
$D_j$ : jème document de la collection
$RSV(Q_k,D_j)$ : Valeur de pertinence associée au document $D_j$ relativement à la requête $Q_k$
$q_{ki}$ : Poids d'indexation du terme $t_i$ dan la requête $Q_k$
$d_{ji}$ : Poids d'indexation du terme $t_i$ dans le document $D_j$
T : Nombre total de termes d'indexation dans la collection
N : Nombre total de documents dans la collection
$n_i$ : Nombre de documents de la collection contenant le terme $t_i$

## 2.1. Le modèle booléen

### *2.1.1. Le modèle de base*

Le modèle booléen propose la représentation d'une requête sous forme d'une équation logique. Les termes d'indexation sont reliés par des connecteurs logiques *ET*, *OU* et *NON*. Le processus de recherche mis en œuvre par le système consiste à effectuer des opérations sur ensembles de documents définis par l'occurrence ou absence de termes d'indexation

afin de réaliser un appariement exact avec l'équation de la requête. De manière formelle, le modèle de recherche booléen est défini par un quadruplet *(T, Q, D, F)*

Où :

T: Ensemble des termes d'indexation

Q : Ensemble de requêtes booléennes

D : Ensemble des documents de la collection

F : Fonction présence définie par : $D \times Q \longrightarrow \{0, 1\}$

$F(d,t) = 1$ si t se trouve dans D

$= 0$ sinon

Sur la base de cette fonction, on calcule la ressemblance relativement à la forme de la requête comme suit :

| ***Formulation booléenne*** | ***Formule d'évaluation*** |
|---|---|
| $F(d_k , t_i$ et $t_j )$ | $\text{Min}(F(d_k , t_i ) , F(d_k , t_j )) = F(d_k , t_i ) * F(d_k , t_j )$ |
| $F(d_k , t_i$ ou $t_j )$ | $\text{Max}(F(d_k , t_i ) , F(d_k , t_j )) = F(d_k , t_i ) + F(d_k , t_j ) - F(d_k , t_i ) * F(d_k , t_j )$ |
| $F(d_k$ , Non $t_i)$ | $1 - F(d, t_i )$ |

Le modèle booléen présente le principal avantage de simplicité de mise en œuvre. Toutefois, il présente les principaux inconvénients suivants :

- les formules de requêtes sont complexes, non accessibles à un large public,
- la réponse du système dépend de l'ordre de traitement des opérateurs de la requête,
- la fonction d'appariement n'est pas une fonction d'ordre,
- les modèles de représentation des requêtes et documents ne sont pas uniformes. Ceci rend le modèle inadapté à une recherche progressive.

### *2.1.2. Le modèle booléen étendu*

Le modèle booléen étendu [Fox 1983] [Salton 1989] complète le modèle de base en intégrant des poids d'indexation dans l'expression de la requête et documents. Ceci a pour conséquence la sélection de documents sur la base d'un appariement rapproché (fonction d'ordre) et non exact. A cet effet, l'opérateur $L_p$*-Norm* est défini pour la mesure de pertinence requête-document. Cette mesure est évaluée pour des requêtes décrites sous la forme conjonctive ou disjonctive, comme suit :

***Opérateur OR :*** $$RSV(Q_k,D_j)=\left[\frac{\sum_{i=1}^{T}\sum_{i=1}^{T} q_{ki}^{P} d_{ji}^{P}}{\sum_{i=1}^{T} q_{ki}^{P}}\right]^{1/P}$$

***Opérateur AND :*** $$RSV(Q_k,D_j)=\left[\frac{\sum_{i=1}^{T}\sum_{i=1}^{T} q_{ki}^{P}(1-d_{ji}^{P})}{\sum_{i=1}^{T} q_{ki}^{P}}\right]^{1/P}$$

Où :

P est une constante.

La littérature rapporte qu'aucune méthode formelle n'est proposée pour la détermination de la valeur du paramètre *P* [Ponte 1998].

### *2.1.3. Le modèle des ensembles flous*

La théorie des ensembles flous est due à Zadeh [Zadeh 1965]. Elle est basée sur l'appartenance probable, et non certaine, d'un élément à un ensemble. Un ensemble flou est formellement décrit comme suit :

$$EF=\{(e_1, f_{EF}(e_1)) , .... (e_n, f_{EF}(e_n)) \}$$

Où :
$e_i$ : Elément probable de E
$f_{EF}$ : E ⟶ [0, 1]
$e_i$ ⟶ $f_{EF}(e_i)$ = degré d'appartenance de $e_i$ à E

Les opérations de base sur les ensembles flous sont alors définies comme suit :

**Intersection** : $f_{A\cap B}(e_i)=Min(f_A(e_i),f_B(e_i)) \quad \forall e_i \in E$

**Union** : $f_{A\cup B}(e_i)=Max(f_A(e_i),f_B(e_i)) \quad \forall e_i \in E$

**Complément** : $f_{A'}(e_i)=1-f_B(e_i) \quad \forall e_i \in E \ \ avec\ A'=\{x \in A \wedge x \notin B\}$

Une extension du modèle booléen basée sur les ensembles flous est proposée par Salton [Salton 1989]. L'idée de base est de traiter les descripteurs de documents et requêtes comme étant des ensembles flous. L'ensemble flou des documents supposés pertinents à une requête est obtenu en suivant les étapes suivantes :

1. Pour chaque terme $t_i$ de $Q_k$, construire l'ensemble flou $D_t$ des documents contenant ce terme.
2. Effectuer sur les ensembles $D_t$, les opérations d'intersection et union selon l'ordre décrit dans l'expression de $Q_k$ relativement aux opérateurs *ET* et *OU* respectivement.
3. Ordonner l'ensemble résultat de la précédente opération selon le degré d'appartenance de chaque document à l'ensemble associé à chaque terme.

Exemple
Soit la requête $Q_k = t_1 \wedge (t_2 \vee t_3)$
En posant :
$Q_{k1} = t_1 \wedge t_2 \wedge t_3$ , $Q_{k2} = t_1 \wedge t_2 \wedge \neg t_3$ , $Q_{k3} = t_1 \wedge \neg t_2 \wedge t_3$ ,
On obtient l'expression disjonctive suivante de la requête $Q_k$ : $Q_k = Q_{k1} \vee Q_{k2} \vee Q_{k3}$
1. On construit les ensembles flous d'occurrence des termes $t_1$ , $t_2$ et $t_3$ dans les documents $D_1$, $D_2$, $D_3$, $D_4$ et $D_5$, soient :
$D_{t1} =\{ 0.2, 0.4, 0.2, 0.6, 0.8\}$
$D_{t2} = \{0.1, 0.8, 0.4, 0.3, 0 \}$

$D_{t3}$ = *{ 0.4, 1, 0.1, 0.1, 0.2}*

On construit les ensembles compléments :
$D_{t2}$ = *{0.9, 0.2, 0.6, 0.7, 1 }*
$D_{t3}$ = *{ 0.6, 0, 0.9, 0.9, 0.8 }*

2. Les ensembles pertinents associés à la requête $Q_k$ sont obtenus par application des opérations sur les ensembles flous, comme décrit dans sa forme disjonctive
   $D_{Qk1}$ = *{0.1, 0.4, 0.1, 0.2, 0}*
   $D_{Qk2}$ = *{0.1, 0, 0.2, 0.3, 0}*
   $D_{Qk3}$ = *{0.4, 0.2, 0.1, 0.1, 0.2}*
   On a alors : $D_{QK}$ = *{0.1, 0.2, 0.1, 0.1, 0.2}*

3. On obtient ainsi la liste ordonnée des documents pertinents à la requête $Q_k$ :
   $DP(Q_k)$ = *{* $D_2$ , $D_5$ , $D_1$, $D_3$, $D_4$ *}*

Le principal intérêt de ce modèle est l'application d'opérations algébriques sur les ensembles de documents plutôt qu'une simple maximisation ou minimisation de valeurs d'ensembles [Yates et Neto, 1999].

Lucarella et Morara [Lucarella et Morara 1991] ont exploité le modèle des ensembles flous pour mettre en œuvre le système FIRST. Les auteurs ont proposé l'utilisation d'un réseau où chaque nœud représente un terme de document ou requête et un lien représente une relation sémantique entre termes. Chaque document $D_j$ est décrit par un ensemble flou comme suit :

$$D_j = \{(t_1, d_{j1}), \ldots, (t_T, d_{jT})\}$$

Une liaison entre concepts est valorisée de manière directe, ou dérivée par transitivité floue :

$$F(t_i , t_k ) = Min (F(t_i, t_j) , F(t_j , t_k))$$

Où F est la fonction de valorisation des liens

L'ensemble flou des documents pertinents à une requête $Q_k$ est obtenu comme suit :

1. Pour chaque terme $t$ de $Q_k$ , construire l'ensemble des documents $D_t$ reliés par lien direct ou transitif.
2. Pour chaque couple $(t , D_t)$, associer un degré d'appartenance égal à la valeur minimale de tous les liens qui figurent sur le chemin $t - D_t$
3. Effectuer sur les ensembles $D_t$, les opérations d'intersection et union selon l'ordre décrit dans l'expression de $Q_k$ relativement aux opérateurs *ET* et *OU* respectivement.

4. Ordonner l'ensemble résultat de la précédente opération selon le degré d'appartenance de chaque document à l'ensemble associé à chaque terme.

Des expérimentations réalisées sur une collection de tests italienne comprenant *300* documents, *175* concepts et *15* requêtes, ont montré que le modèle offre de meilleures valeurs de rappel relativement au modèle vectoriel.

Chen et Wang [Chen et Wang 1995] ont étendu ce modèle à l'utilisation d'intervalles de poids admissibles aux concepts, par opposition à l'utilisation de valeurs uniques, ainsi qu'à l'utilisation d'une matrice de concepts. La clôture transitive de cette matrice *T* est obtenue par multiplications successives de cette même matrice. La valeur de pertinence requête-document est obtenue selon la formule suivante :

$$RSV(Q_k,D_j)=\sum_{ti\in Qk} T(t_{ji},q_{ki})$$

Où :
T(x,y)=1-|x-y|
$t_{ji}$ : Minimum des poids des liens du document $D_j$ au terme $t_i$

## 2.2. Le modèle vectoriel

### *2.2.1. Le modèle de base*

Ce modèle préconise la représentation des requêtes utilisateurs et documents sous forme de vecteurs, dans l'espace engendré par les *N* termes d'indexation [Salton, 1968] [Salton, 1989]. De manière formelle, les documents et requêtes sont des vecteurs dans un espace vectoriel de dimension *N* et représenté comme suit :

$$D_j=\begin{bmatrix} d_{j1} \\ d_{j2} \\ \\ d_{jT} \end{bmatrix} \qquad Q_k=\begin{bmatrix} q_{k1} \\ q_{k2} \\ \\ q_{kT} \end{bmatrix}$$

Sous l'angle de ce modèle, le degré de pertinence d'un document relativement à une requête est perçu comme le degré de corrélation entre les vecteurs associés. Ceci nécessite alors la spécification d'une fonction de calcul de similarité entre vecteurs mais également du principe de construction qui se traduit par la fonction de pondération.

**1- Fonction de pondération**

La fonction de pondération la plus répandue est $d_{ji}=tf_{ji}*idf_i$ [Sparck Jones et Needham 1972]

Où :

$tf_{ji}$ : Décrit le pouvoir descriptif du terme $t_i$ dans le document $D_j$

$idf_i$ : Décrit le degré de généralité du terme $t_i$ dans la collection

De nombreuses autres fonctions d'indexation sont basées sur une variante du schéma balancé tf.Idf, on cite notamment :

Formule [Salton et Buckley 1988]

$$d_{ji}=\left(0.5+\frac{0.5*freq_{ij}}{Max_l\ freq_{jl}}\right)*\log\frac{N}{n_i}$$

Formule [Salton et Allan, 1994]

$$d_{ji}=\frac{freq_{ij}}{\sqrt{\sum_{j=1}^{N} freq_{ji}*\log_2^2\left(\frac{N}{n_i}\right)}}*\log\frac{N}{n_i}$$

Où :

$freq_{ij}$ : Fréquence d'apparition du terme $t_i$ dans le document $D_j$

Ces mesures supposent que la longueur d'un document n'a pas d'impact sur la mesure de pertinence ; or des expérimentations réalisées par Singhal [Singhal et al 1995] ont montré que les documents longs ont une plus grande probabilité de pertinence parce que contenant plus de termes d'appariement avec la requête.

L'analyse de la corrélation entre probabilité de sélection et probabilité de pertinence a permis la détermination d'une valeur pivot permettant d'ajuster la fonction de pondération par un facteur de normalisation lié à la longueur d'un document. Les auteurs proposent la fonction suivante :

$$d_{ji}=\frac{tf_{ji*}\log\left(\frac{N-n_i+0.5}{n_i+0.5}\right)}{2*\left(0.25+0.75*\frac{|D_j|}{\overline{|D_j|}}\right)+tf_{ji}}$$

Où :

$|D_j|$ : Longueur du document $D_j$

$\overline{|D_j|}$ : Longueur moyenne des documents dans la collection

**2- Fonction de similarité**

La fonction de similarité permet de mesurer la ressemblance des documents et de la requête. Les types de mesures les plus répandus sont :

***Mesure du cosinus*** [Salton, 1971] [Risjbergen 1979]

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki}d_{ji}}{\left(\sum_{i=1}^{T} q_{ki}^2\right)^{1/2}\left(\sum_{i=1}^{T} d_{ji}^2\right)^{1/2}}$$

***Mesure de Jaccard***

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki}d_{ji}}{\sum_{i=1}^{T}(d_{ji})^2+\sum_{i=1}^{T}(q_{ki})^2+\sum_{i=1}^{T} q_{ki}d_{ji}}$$

[Singhal et al 1995] proposent une fonction de pertinence normalisée par la longueur de document, définie comme suit :

$$RSV(Q_k,D_j)=\frac{\sum_{i=1}^{T} q_{ki}d_{ji}}{(1-s)+s*\frac{\sqrt{\sum_{k=1}^{N} d_{kj}^2}}{|D_j|}}$$

Où :

$|D_j|$ : Longueur du document $D_j$

s : Constante

L'utilisation répandue du modèle vectoriel en recherche d'information est principalement due à l'uniformité de son modèle de représentation requête-document, à l'ordre induit par la fonction de similitude ainsi qu'aux possibilités aisées offertes pour ajuster les fonctions de pondération afin d'améliorer les résultats de la recherche.

Toutefois, le modèle présente un inconvénient majeur lié au traitement des termes de documents de manière indépendante. Ceci ne permet pas en effet de reconstituer à travers le processus de recherche, la sémantique associative de termes et ainsi, de la comparer à celle véhiculée par la requête.

### *2.2.2. Le modèle vectoriel généralisé*

Dans le but de pallier au problème d'indépendance des termes, posé par le modèle vectoriel classique, Wong [Wong et al, 1985] a proposé une nouvelle base de référence pour la représentation des documents et requêtes. A cet effet, il définit sur une collection de termes d'indexation $\{t_1, ..., t_T\}$ :

1. Une base de vecteur binaires, non orthogonaux $\{m_i\}_{i=1..2^T}$

2. Un ensemble de min-termes associé à la base ; chaque min-terme correspond à l'ensemble de documents comprenant les termes d'indexation positionnés à *1* dans le vecteur de base correspondant

3. Une fonction de pondération $g_i(m_j)$ qui donne le poids du terme $t_i$ dans le min-terme $m_j$, soit $w_{ij}$

La base ainsi décrite supporte la représentation de la cooccurrence entre termes. Chaque document et requête est décrit dans la nouvelle base comme suit :

$$D_j=\sum_{i=1}^{T} d_{ji}K_i \qquad Q_k=\sum_{i=1}^{T} q_{ki}K_i$$

Où :

$$K_i = \frac{\sum_{\forall r, gi(mr)=1} C_{ir} m_r}{\sqrt{\sum_{\forall r, gi(mr)=1} C_{ir}^{\,2}}}$$

Avec :

$$C_{ir}= \sum_{dj / gi(dj)=gi(mr) \forall l} w_{ij}$$

Le calcul de pertinence *RSV(Q,D)* combine alors le poids des documents $w_{ij}$ et facteur de corrélation entre termes $C_{ir}$ . Malgré un accroissement du coût de calcul pour la mesure de similarité, relativement au modèle vectoriel classique, le modèle vectoriel généralisé a l'intérêt d'introduire l'idée de considérer la relation entre termes de manière inhérente au modèle de la fonction de pertinence.

### *2.2.3. Le modèle LSI*

L'objectif fondamental du modèle LSI [Dumais 1994] est d'aboutir à une représentation conceptuelle des documents où les effets dus à la variation d'usage des termes dans la collection sont nettement atténués. Ainsi, des documents qui partagent des termes co-occurrents ont des représentations proches dans l'espace défini par le modèle. La base mathématique du modèle LSI est la décomposition par valeur singulière SVD de la matrice Terme-Document. La SVD identifie un ensemble utile de colonnes de base de cette matrice. Ces vecteurs de base couvrent le même espace de vecteurs associé à la représentation des documents, car ils sont obtenus par rotation (multiplication par une matrice orthogonale) des vecteurs d'origine. On pose :

*X : Matrice Terme-Document*
*$T_0$ : Matrice avec colonnes orthonormées qui couvre l'espace des colonnes de X*
*$D_0$ : Matrice avec colonnes orthonormées qui couvre l'espace des lignes de X*

*$S_0$ : Matrice diagonale formée des valeurs singulières qui résultent de la normalisation de $T_0$ et $D_0$*

On a : $$X = T_0 S_0 D_0^T$$

La propriété de la SVD pour le modèle LSI est qu'en raison du tri des valeurs singulières dans l'ordre décroissant, la meilleure approximation de $X$ peut être calculée comme suit :

$$X = \sum_{i=1}^{k} T_{0x1i} S_{0x1i} D_{0xi}^T$$

La matrice $T_0$ a deux principales propriétés [Oard, 1996] :

1. Chaque paire de représentations de documents obtenue par combinaison linéaire des lignes de $T_0$ , a la même valeur de degré de similitude que les représentations associées classiques Document-Termes.
2. La suppression des composants de faible poids dans une ligne améliore l'ordre lors du calcul de similitude requête-document.

On passe ainsi d'une représentation de documents à base de termes, vers une représentation à base de concepts, dans un espace de dimension plus réduite.

La méthode LSI a été appliquée dans la collection TREC pour la tâche de croisement de langues [Deerwester et al 1990]. L'application de la méthode dans un corpus parallèle a permis d'identifier les principaux composants de l'espace vectoriel, associés à chaque langue, produisant ainsi une représentation unifiée de documents écrits dans différentes langues. En utilisant des requêtes en anglais, la méthode sélectionne en début de liste les versions traduites en français dans *92%* des cas.

Outre, l'accroissement de performances dû à son utilisation [Deerwester et al, 1990] [Dumais, 1994], le modèle LSI présente l'intérêt majeur d'introduire la notion de concept en recherche d'information à travers l'utilisation de la théorie relative à la décomposition par valeurs singulières.

Le modèle LSI probabiliste a été proposé par Hofman [Hofman 1999]. La particularité de ce modèle relativement au modèle LSI classique, est l'intégration de techniques statistiques pour le traitement des mots polysémiques. A cet effet, le modèle utilise des critères d'optimalité de la décomposition/approximation basée sur une distribution de probabilités.
Les expérimentations réalisées sur les collections MED, CACM, CRANFIELD et CISI prouvent la consistance du modèle et son impact positif sur le rappel du système relativement au modèle LSI classique [Hofman, 1999]. Le rappel est une mesure communément utilisée dans l'évaluation des modèles de recherche d'information. Elle sera définie ultérieurement dans la section 3 avec la précision qui est une autre mesure aussi importante

## 2.3. Le modèle probabiliste

### *2.3.1. Le modèle de base*

Le modèle de recherche probabiliste utilise un modèle mathématique fondé sur la théorie de la probabilité [Robertson & Sparck Jones, 1976]. Le processus de recherche se traduit par calcul de proche en proche, du degré ou probabilité de pertinence d'un document relativement à une requête. Pour ce faire, le processus de décision complète le procédé d'indexation probabiliste en utilisant deux probabilités conditionnelles :

$P(w_{ji} / Pert)$ : Probabilité que le terme $t_i$ occure dans le document $D_j$ sachant que ce dernier est pertinent pour la requête

$P(w_{ji} / NonPert)$ : Probabilité que le terme $t_i$ de poids $d_{ji}$ occure dans le document $D_j$ sachant que ce dernier n'est pas pertinent pour la requête

Le calcul des occurrences des termes d'indexation dans les documents est basée sur l'application d'une loi de distribution (type loi de poisson) sur un échantillon représentatif de documents d'apprentissage.

En posant les hypothèses suivantes :

1. La distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents
2. Les variables *document pertinent* et *document non pertinent* sont indépendantes.

la fonction de recherche est obtenue en calculant la probabilité de pertinence d'un document *D*, notée *P(Pert/D)* [Risjbergen 1979] :

$$P(Pert/Dj)=\sum_{i=1}^{T}\log\frac{P(w_{ji}/Pert)}{P(w_{ji}/NonPert)}$$

L'ordre des documents est basé sur l'une des deux méthodes suivantes :

1. Considérer seulement les termes présents dans les documents et requêtes

2. Considérer les termes présents et termes absents dans les documents et requêtes

Croft et Harper [Croft et Harper, 1979] intègrent au modèle les mesures de fréquence plutôt, que de considérer seulement la présence ou l'absence des termes. La similitude requête document est calculée comme suit :

$$RSV(Q_k,D_j)=C\sum_{i=1}^{T}q_{ki}d_{ji}+\sum_{i=1}^{T}f_{ji}q_{ki}d_{ji}\log\frac{N-n_i}{n_i}$$

Où :

$$f_{ji}=\frac{tf_{ji}}{\max\ tf_j}$$

; C : Constante

Robertson et Walker [Robertson et Walker 1994] intègrent la fréquence d'apparition des termes dans la formule de calcul de poids et ce, en se basant sur le modèle de Poisson

$$w_i = \log \frac{\left(p' + (1-p')\left(\frac{\mu}{\lambda}\right)^{tf} e^{j}\right)\left(q' e^{-j} + 1 - q'\right)}{q' + (1-q')\left(\frac{\mu}{\lambda}\right)^{tf} e^{j}\left((p' e^{-j}) + (1-p')\right)}$$

Où :

$\lambda$ : Paramètre de la loi de poisson pour les documents contenant t

$\mu$ : Paramètre de la loi de poisson pour les documents ne contenant pas t

j : Différence $\lambda$ - $\mu$

p' : Probabilité qu'un document contenant t soit pertinent

q' : Probabilité qu'un document contenant t ne soit pas pertinent

La difficulté majeure du modèle réside dans la détermination des valeurs p, p', $\lambda$ et $\mu$

Dans [Robertson et Walker, 1997], les auteurs montrent que le schéma de pondération de Croft et Harper, peut sous certaines conditions, produire des valeurs négatives. Ils proposent alors une fonction de pertinence d'un document relativement à une requête, basé sur le calcul de chacun des poids des termes d'indexation comme suit :

$$d_{ji} = \frac{k_5}{k_5+R}(k_4 + \log\frac{N}{N-n_i}) + \frac{R}{k_5+R}\log(\frac{r_i+0.5}{R-r_i+0.5}) - \frac{k_6}{k_6+S}\log(\frac{n_i}{N-n_i}) - \frac{S}{k_6+S}\log(\frac{s_i+0.5}{S-s_i+0.5})$$

Où :

R : Nombre de documents pertinents

$r_i$ : Nombre de documents pertinents contenant le terme $t_i$

S : Nombre de documents non pertinents

$s_i$ : Nombre de documents non pertinents contenant le terme $t_i$

$k_4$ , $k_5$ , $k_6$ : Constantes

De manière générale, le modèle probabiliste présente l'intérêt d'unifier les représentations des documents et concepts. Cependant, le modèle repose sur des hypothèses d'indépendance des variables pertinence non toujours vérifiées, ce qui entache les mesures de similitude d'imprécision.

En outre, le modèle ne prend pas en compte les liens de dépendance entre termes, et engendre des calculs de probabilité conditionnels complexes.

### *2.3.2. Le modèle de réseau infèrentiel bayésien*

Un réseau bayésien est un graphe direct acyclique où les nœuds représentent des variables aléatoires et les arcs des relations causales entre nœuds. Ces derniers sont pondérés par des valeurs de probabilités conditionnelles.
Le travail original en recherche d'information, et basé sur le modèle des réseaux bayésiens, est développé par Turtle [Turtle et Croft, 1991].
Dans l'espace défini par les termes d'indexation, on définit :
- $T$ variables aléatoires binaires $t_1, \ldots, t_T$ associés aux termes d'indexation
- $D_j$ : Variable aléatoire associée à un document
- $Q_k$ : Variable aléatoire associée à une requête

On calcule alors la mesure de pertinence de $Q_k$ relativement à $D_j$ en traitant les probabilités conditionnelles de Bayes selon la formule :

$$RSV(Q_k,D_j)=1-P(Q_k \wedge D_j)$$

Où :

$$P(Q_k \wedge D_j)=\sum_{i=1}^{T} P(Q_k/t_i)*\left(\prod_{ti \in Dj} P(t_i/D_j)*\prod_{ti \notin Dj} P(\bar{t}_i/D_j)\right)*P(D_j)$$

Avec :
$P(Q_k/t_i)$ : Probabilité que le terme $t_i$ appartienne à un document pertinent de $Q_k$
$P(t_i/ D_j)$ : Probabilité que le terme $t_i$ appartienne au document $D_j$ sachant qu'il est pertinent
$P(\overline{t_i} /D_j ) = 1- P(t_i/D_j)$
$P(D_j)$ : Probabilité d'observer $D_j$

Les probabilités conditionnelles de chaque nœud sont calculées par propagation des liens de corrélation entre eux.
Le modèle présente l'intérêt de considérer la dépendance entre termes mais engendre une complexité de calcul importante.
D'autres modèles sont décrits dans [Savoy et Desbois 1991] et [Haines et Croft 1993].

## 2.4. Le modèle connexionniste

Les SRI basés sur l'approche connexionniste utilisent les fondements des réseaux de neurones tant pour la modélisation des unités textuelles que pour la mise en œuvre du processus de recherche d'information.
Après un bref aperçu des concepts clés du modèle, nous décrivons avec plus de détails les modèles connexionnistes pour la recherche d'information.

### *2.4.1. Le modèle de base*

Le fonctionnement d'un neurone formel est inspiré du fonctionnement connu d'un neurone biologique. Un neurone biologique est le processeur élémentaire de traitement de l'information, il est composé d'un [Bourret et Samuelides 1991] :

- Réseau convergent d'entrée : **dendrites**
- Elément de traitement de l'information : **corps cellulaire**
- Réseau divergent : **axone**

La connexion de l'axone d'un neurone aux dendrites d'un autre neurone est appelé **synapse**. Les neurones sont connectés pour former un réseau. La transmission des signaux d'activation est effectuée par propagation depuis les entrées jusqu'aux sorties.
Par analogie, un neurone formel reçoit des entrées des neurones auxquels il est connecté en tant que successeur; le neurone calcule une somme pondérée des potentiels d'action de ses entrées puis calcule une valeur de sortie correspondant à son niveau d'activation.
Si le niveau dépasse un seuil, le neurone est activé et transmet une réponse; si cela n'est pas le cas, le neurone est dit inhibé et ne transmet aucun signal.
Un neurone est caractérisé par:

**1- La fonction d'entrée totale** $E= f(e_1 , e_2 , ..., e_n )$ qui peut être :

- Linéaire $E = \sum_{i=1}^{n} w_{ij} e_j$

- Affine $E = \sum_{j=1}^{n} w_{ij} e_j - a$

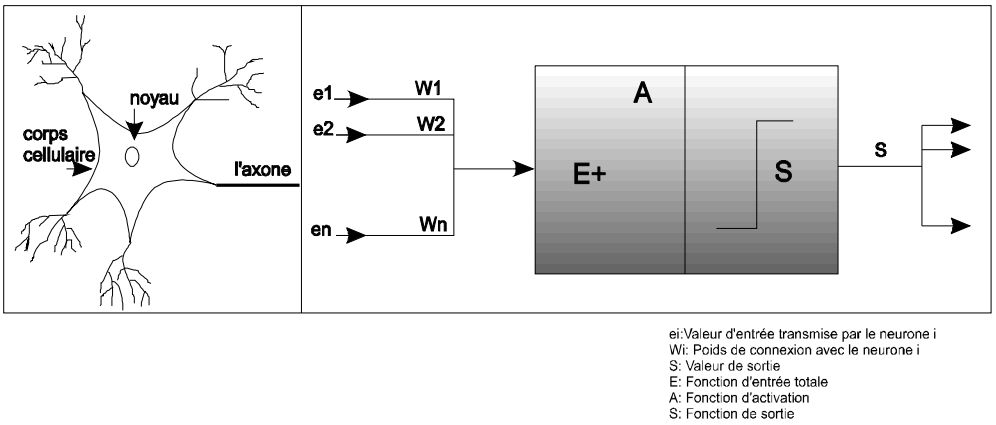


**Figure 1.1 :** Modèle de neurone formel et modèle de neurone biologique

**2- La fonction d'activation** $A= A(E)$ qui peut être

- La fonction binaire Heaviside $A = 0$ *Si* $E \leq 0$, $A= 1$ *Si* $E > 0$
- La fonction signe $A = -1$ *Si* $E \leq 0$, $A=1$ *Si* $E > 0$
- La fonction sigmoïde $A = a * \frac{ekx - 1}{ekx + 1}$ $A \in [-a \ a]$

**3- La fonction de sortie** $S= S(A)$ qui est généralement la fonction identité

Un réseau de neurones est caractérisé par deux propriétés fondamentales : dynamique des états et dynamique des connexions

- ***Dynamique des états :*** La dynamique des états correspond à l'évolution des états des différents neurones qui composent le réseau. Cette évolution est modulée d'une part par l'architecture du réseau et d'autre part, par la structure des poids des connexions et nature de la fonction d'activation

- ***Dynamique des connexions :*** La dynamique des connexions correspond à l'évolution des poids des connexions en cours du temps. Ceci traduit **l'apprentissage** du réseau par changement de son comportement d'après les résultats de son expérience passée. Comme pour l'activation des noeuds, les poids sont modifiés en parallèle mais varient généralement plus lentement que les niveaux d'activation.

De nombreuses approches de l'apprentissage ont été proposées; on y présente généralement des règles de modification de poids telles que la règle de Hebb [Hebb 1949], Windrow-Hoff et rétropagation du gradient [Bourret et Samuelides, 1991].

### *2.4.2. Le modèle à couches*

Les modèles connexionistes à couches sont d'utilisation répandue en recherche d'information [Wilkinson et Hingston, 1991] [Boughanem, 1992] [Kwok, 1995]. Le réseau est construit à partir des représentations initiales de documents et informations descriptives associées (termes, auteurs, mots clés ...). Le mécanisme de recherche d'information est fondé sur le principe d'activation de signaux depuis les neurones descriptifs de la requête, et propagation de l'activation à travers les connexions du réseau. La pertinence des documents est alors mesurée grâce à leur niveau d'activation.

Le réseau construit dans Kwok [Kwok, 1995] utilise trois couches interconnectées dans le sens requête - termes - documents. Les connexions sont bidirectionnelles et de poids asymétriques.

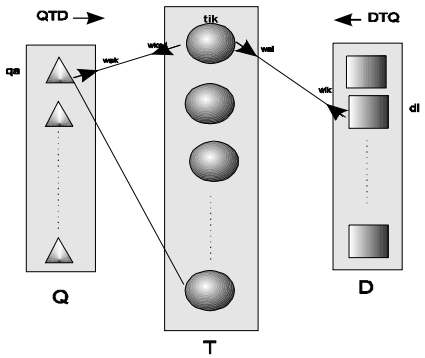


**Figure 1.2.** Le modèle de réseau Kwok [Kwok, 1995]

L'approche de Kwok est fondée sur l'idée que les requêtes et documents sont similaires, en ce sens qu'ils sont tous deux représentants de concepts. Sur cette base, il reprend des

éléments du modèle probabiliste pour classer les neurones documents selon la probabilité $W_i=W_{j/Q}+W_{j/D}$

Où :

$W_{j/Q}$ : Probabilité pour que la requête Q soit pertinente pour le document $D_j$

$W_{j/D}$ : Probabilité pour que le document $D_j$ soit pertinent pour la requête Q

La valeur de $W_{j/Q}$ est obtenue dans le sens **DTQ**, par simulation de la pertinence du document $D_j$ ; on injecte à l'entrée du neurone $d_i$ un signal de *1*, puis on évalue la valeur d'activation du neurone $q_a$ :

$$W_{qa}=\sum_{k=1}^{m} W_{ka}S_{ik} \qquad W_{jQ}=\sum_{k=1}^{T} W_{j/Q}d_{jk}$$

Ce processus est itéré pour chaque neurone document $D_i$.

D'autre part, on effectue la propagation dans le sens **QTD** par injection d'un signal de *1* à l'entrée du neurone $q_a$ . On évalue alors les valeurs de sortie aux neurones documents $W_{j/D}$

$$W_{j/D}=\sum_{k=1}^{T} W_{kj}S_{jk} \qquad S_{jk}=\sum_{k=1}^{T} W_{ak}q_{jk}$$

Où :

$W_{jk} = d_{jk}/L_j$

$W_{ak} = q_{ak}/L_a$

$q_{ak}= Log(r_{ak}/(1-r_{ak}))+Log((1-S_{ak})/S_{ak})$

$d_{jk} = Log(r_{jk}/(1-r_{jk}))+Log((1-S_{jk})/S_{jk})$

Avec

$L_j$ : Nombre de termes du document $d_j$

$q_{ak}$ : Fréquence du terme du terme $t_k$ dans la requête $Q_a$

$L_a$ : Nombre de termes de la requête $Q_a$

$r_{ik} = r_{ak} = 1/40$

$S_{jk} = (F_k - d_{ik})/(T- L_i)$

$S_{ak} = F_k / T$

$F_k$ : Fréquence du terme $t_k$ dans la collection

En outre, le modèle est doté de la capacité d'apprentissage par modification des poids de connexions suite à la perception des jugements de pertinence de l'utilisateur.

# 3. Les mesures d'évaluation d'un modèle de RI

Les mesures les plus utilisées pour l'évaluation d'un modèle de recherche d'information sont le *rappel* et la *précision*. De manière classique, ils sont obtenus en partitionnant l'ensemble des documents restitués par le SRI en deux catégories : documents pertinents et documents non pertinents.

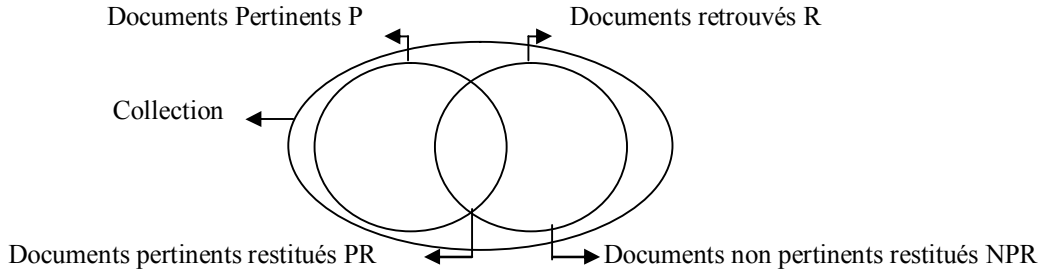


**Figure 1. 3.** Partition de la collection pour une requête

On définit :

**Rappel :** Proportion de documents pertinents restitués par le système relativement à l'ensemble des documents pertinents contenus dans la base. Le rappel est calculé selon la formule suivante :

$$Rappel = \frac{|PR|}{P}$$

**Précision :** Proportion de documents pertinents relativement à l'ensemble des documents restitués par le système. La précision est calculée selon la formule suivante :

$$\text{Pr}écision = \frac{|PR|}{|R|}$$

# 4. Conclusion

Ce premier chapitre a porté essentiellement sur une présentation des SRI de manière générale et des modèles de recherche et de représentation d'information de manière particulière. Il en ressort que chacun de ces modèles ou stratégies contribue en partie à la résolution des problèmes inhérents à la recherche d'information : perception du besoin en information, représentation du sens véhiculé par les documents, formalisation de la pertinence etc… Pour ce faire, les auteurs puisent dans une large mesure d'un support théorique permettant d'associer les différentes fonctions de calcul de poids des termes, liens terme-terme, terme-document, appariement requête-document etc…

Les SRIs se divisent en deux catégories ; à savoir les systèmes de recherche d'information centralisée (SRICs) et les systèmes de recherche d'information distribuée (SRIDs). Dans cette thèse nous nous sommes intéressés particulièrement aux SRIDs. Dans le chapitre suivant, nous présentons une étude détaillée des SRIDs, en se focalisant sur les deux problèmes liés aux SRIDs ; à savoir le problème de sélection de sources d'information et le problème de fusion des résultats des sources sélectionnées.

# Chapitre 2

# La recherche d'information distribuée

## 1. Introduction

La problématique de tout système de recherche d'information est comment trouver pour un utilisateur l'information pertinente qu'il cherche dans une masse d'information considérable. La forme la plus simple d'un SRI est celle qui permet à l'utilisateur de formuler sa requête, de traiter cette dernière et de lui fournir une liste de documents ou de références aux documents jugés pertinents. Nous pouvons trouver d'autres formes comme par exemple les systèmes d'aides à la navigation ou les annuaires.

Nous avons évoqué dans le chapitre introductif les deux catégories des systèmes de recherche d'information (SRIs), à savoir les systèmes de recherche d'information centralisée (SRICs) et les systèmes de recherche d'information distribuée (SRIDs). L'inconvénient des SRICs réside dans l'unicité de l'index, qui génère des problèmes avec la croissance de l'information exploitée (voir chapitre productif).

Le problème de la recherche d'information distribuée (RID) est soulevé lorsque l'utilisateur a accès à plusieurs SRIs, chacun oeuvrant souvent sur une source différente, et a besoin d'être aidé pour retrouver l'information pertinente, sans avoir à interroger séparément chaque SRI. L'utilisateur d'un SRID envoie sa requête à un seul système qui se charge d'interroger les différents SRIs.

Ce chapitre a pour but d'introduire le domaine de la recherche d'information distribuée dans lequel se situe cette thèse. Nous présentons d'abord la terminologie qui sera adoptée dans ce travail. Nous présentons ensuite les différents composants d'un SRID, ainsi que les problèmes subordonnés à la RID. Nous détaillons ensuite les points qui ont un rapport avec notre travail, à savoir l'évaluation des SRID, la sélection des sources d'information et la fusion des résultats.

## 2. Terminologie

Nous définissons dans cette section les termes que nous allons utiliser pour présenter le domaine de la RID :

1. ***Source ou collection d'information :*** est un ensemble de documents mis à disposition à des utilisateurs par un particulier ou une organisation.

2. ***Serveur d'information :*** est vu comme une machine qui offre à un utilisateur le service de recherche de l'information dans une source ou collection qu'il héberge. Un serveur contient un SRI local qui effectue cette tâche.

3. ***Liste de résultats :*** est une liste triée de documents (lien vers les documents) sélectionnés par le système pour une requête donnée.

4. ***Courtier (Broker) :*** est un module qui agit comme un pseudo moteur de recherche (méta-moteur). Le courtier joue le rôle d'intermédiaire entre l'utilisateur et les différentes sources d'information. Il reçoit en entrée une requête et fournit en sortie une liste de résultats. Le courtier assure les différents processus de RID, à savoir le processus de sélection de sources, le processus d'évaluation des requêtes et le processus de fusion des résultats des sources.

5. ***Représentant de serveur :*** est un ensemble d'informations détenues par le courtier qui décrivent le contenu et les caractéristiques du serveur, en guise d'exemple les termes apparaissant dans le contenu du serveur (des informations textuelles), leurs fréquences d'apparition (des informations statistiques), ainsi que des informations utiles à son interrogation (sa localisation, son coût, ses droits d'accès, …etc.).

6. ***Utilité d'un serveur pour une requête :*** se traduit par sa capacité à contenir et retourner des documents pertinents à cette requête.

7. ***Fréquence document d'un terme $t_j$ :*** est le nombre de documents, dans la source contenant $t_j$.

8. ***Fréquence source d'un terme $t_j$ :*** est le nombre de sources, dans un SRID contenant $t_j$.

# 3. Les systèmes de recherche d'information distribuée

Un système de recherche d'information distribuée (SRID) exploite plusieurs serveurs d'informations. Il sélectionne un sous-ensemble $S'$ en réponse à une requête utilisateur $q$ à partir d'un ensemble de serveurs d'informations $S$. Chacun de ces serveurs sélectionnés répond par une liste $L_i = (d_i, o_i)$. La liste $L_i$ est un ensemble de document $d_i$ rangés dans un certain ordre $o_i$. Le système par, la suite, fusionne et trie ces listes dans une seule liste finale $L$.

## 3.1. Architecture et fonctionnement d'un SRID

Un SRID comprend plusieurs serveurs et un courtier. Le courtier est le cœur du système, il contient 5 modules :

1. un module de description de serveurs ;
2. un module d'interface utilisateur ;
3. un module de sélection de serveurs ;
4. un module de communication ;
5. un module de fusion de résultats ;

D'une manière globale, le module de description de serveurs construit les représentants des serveurs connus par le système. Le module de sélection de serveurs reçoit la requête utilisateur via le module d'interface utilisateur, sélectionne les serveurs pertinents pour la requête. Le module de communication adapte la requête au langage d'interrogation de chaque serveur sélectionné, leur achemine la requête et réceptionne leur réponse. Finalement le module de fusion des résultats trie et groupe les réponses reçues par le module de communication et rend via l'interface utilisateur une seule liste de résultats à l'utilisateur. La figure 2.1 montre une architecture générale d'un SRID.

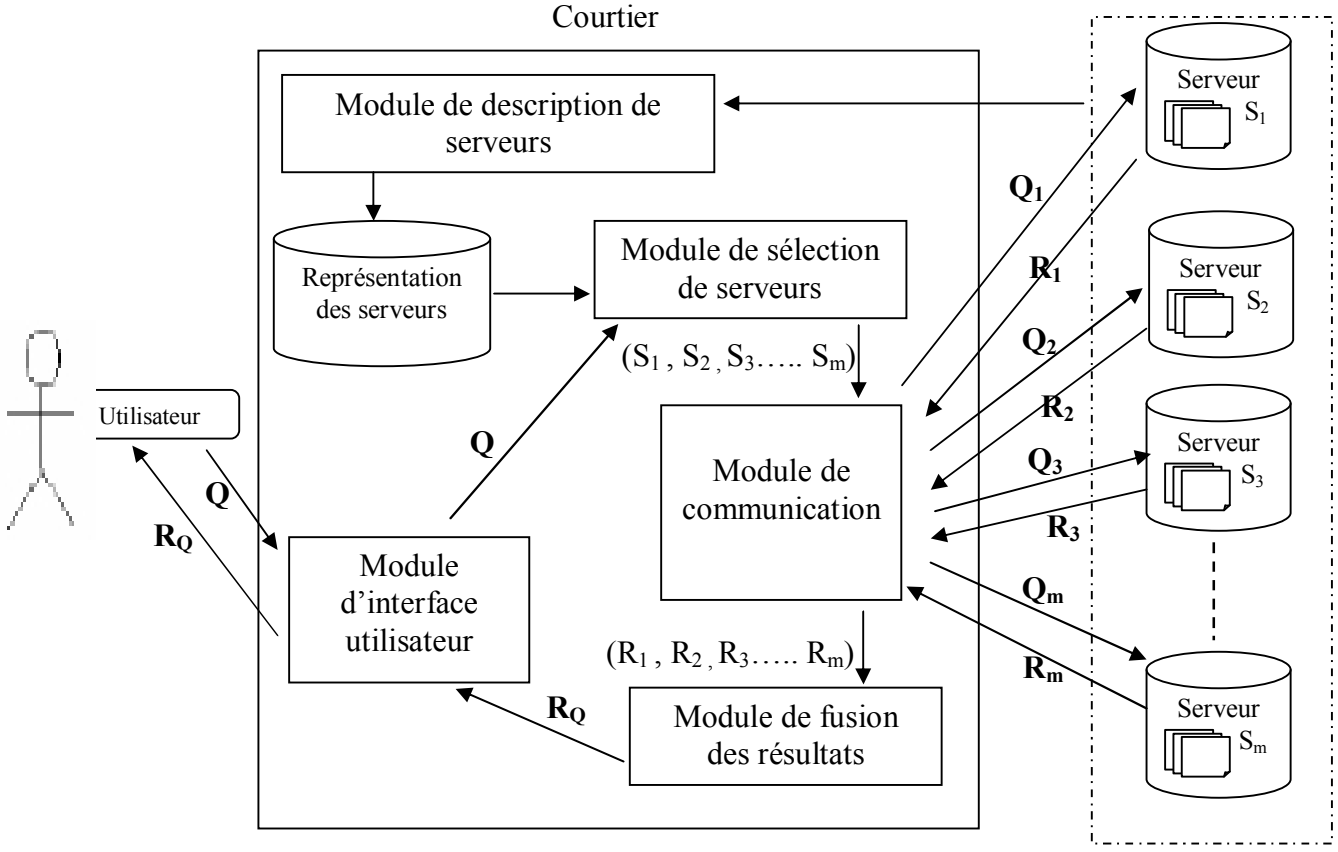


**Figure 2.1.** Architecture générale d'un système de recherche d'information distribuée (SRID)

Nous avons adopté dans le schéma les notations suivantes :

- $Q$ une requête d'un utilisateur.
- $R_Q$ la liste finale de résultats fusionnés.
- $(S_1, S_2, S_3 \ldots\ldots S_m)$ l'ensemble de serveurs sélectionnés pour une requête Q.
- $R_i$ la liste des résultats provenant du serveur $S_i$.
- $Q_i$ est la requête traduite dans le langage d'interrogation du serveur $S_i$ (requête modifiée suivant la syntaxe acceptée par le serveur).

### *3.1.1. Le module de description de serveur*

Ce module se charge de la construction, la représentation et la mise à jour du contenu de chaque serveur utilisé par le système (*les représentants des serveurs*). L'automatisation de cette tâche est très importante vu la rapidité avec laquelle les documents paraissent, sont modifiés et disparaissent.

Nous distinguons deux types de constructions ;

- construction manuelle [Chakravarthy et Haase 1995] : dans laquelle la représentation de chaque document (par exemple le choix des mots clés les plus représentatifs du document) est faite manuellement par un spécialiste ;

- construction automatique [Voorhess 1995] : dans laquelle la représentation de chaque document (par exemple le choix des mots clés les plus représentatifs du document) est faite automatiquement par des algorithmes d'indexation.

### *3.1.2. Le module d'interface utilisateur*

Ce module assure la communication entre l'utilisateur et le système. L'utilité principale de ce module est de faciliter à l'utilisateur la formulation de sa requête en lui offrant des fonctionnalités comme : le choix de sources à rechercher, le choix du nombre de documents qu'il voudrait retrouver, la possibilité d'affiner sa requête d'une manière manuelle ou automatique, etc…..

### *3.1.3. Le module de sélection de serveurs*

Ce module permet de sélectionner à partir des serveurs connus par le système l'ensemble des serveurs $(S_1, S_2, S_3 \ldots\ldots S_m)$ les plus susceptibles de contenir des documents pertinents à la requête de l'utilisateur. La sélection des serveurs pertinents pour une requête est une problématique abordée par plusieurs approches, chacune présente une méthode plus au moins efficace par rapport à l'autre.

Certains systèmes ne procèdent pas à une sélection et interrogent tous les serveurs disponibles. Cette approche est la plus répandue dans les méta-moteurs de recherche d'information sur Internet.

La plupart des méthodes de sélection existantes procèdent au classement des serveurs selon leur similarité (score) à la requête. Après le classement, la sélection proprement dite, peut s'effectuer selon plusieurs scénarios :

- *la sélection fixe :* consiste à sélectionner un nombre prédéfini *n_serv* de serveurs.
- *la sélection par seuil :* consiste à sélectionner tous les serveurs dont leurs scores dépassent un certain seuil *seuil ;*
- *la sélection économique :* consiste à choisir les serveurs qui satisfont certaines métriques de coût de façon à ce que l'on obtienne le maximum de documents pertinents au moindre coût. Par exemple, l'approche proposée par Furh [Furh 1999] se base sur des métriques comme le coût de traitement, la qualité de recherche et le temps de réponse de ce serveur.

Concernant la sélection *fixe* et *la sélection par seuil*, la définition du nombre de serveurs à sélectionner soulève des difficultés. En effet, si le nombre de serveurs est trop petit alors des serveurs contenant des documents pertinents peuvent être écartés, ce qui favorise l'augmentation du silence dans la réponse ; et s'il est trop grand, des serveurs ne contenant aucun document pertinent peuvent être aussi interrogés, ce qui augmente le bruit dans la réponse.

Le but principal de la sélection est de réduire le coût de la recherche, en diminuant le nombre de serveurs à interroger sans détériorer la performance du système, voire augmenter sa performance si l'on écarte uniquement les serveurs ne possédant aucun document pertinent. Ce qui regroupe en même temps *l'efficience* et *la performance*. (Voir la définition de la performance et l'efficience dans le *chapitre introductif*).

D'autres types de sélection existent, comme ; la sélection d'une façon aléatoire [Katz et al 1994] ; la sélection selon la topologie du réseau, en choisissant le serveur le plus proche [Guyton et Schwarz 1995] ; la sélection en choisissant le serveur le moins encombrant dynamiquement [Carter et Crovella 1996] ; la sélection en se basant sur des critères concernant les serveurs, notamment leur temps de réponse, la vitesse de transfert avec ses serveurs ….. etc [Bhattacharjee et al 1997].

### *3.1.4. Le module de communication*

Le module de communication assure deux fonctions principales. Premièrement avant d'interroger les serveurs sélectionnés, il convertit la requête dans un format compréhensive

par chaque serveur. Afin de résoudre le problème de traduction de requête, des protocoles de standardisation ont été suggérés comme, *Z39.50* [Org 1993], *Common Command Language (CCL)* [Negus 1979] et *ISO 8777* [ISO 1993].

Deuxièmement il interroge les serveurs par les requêtes traduites, récupère les listes des résultats des serveurs et les dirige vers le module de fusion des résultats.

### *3.1.5. Le module de fusion des résultats*

Ce module a pour rôle de regrouper les listes de résultats provenant des différents serveurs interrogés pour constituer une seule liste de résultats qui sera présentée à l'utilisateur en réponse à sa requête.

Sachant que les éléments (documents) de la liste finale de résultats doivent être triés par ordre décroissant de pertinence, un problème se pose ; sur quelle base seront-ils classés ?. En effet, les seules informations qui peuvent être disponibles dans les différentes réponses pour chaque document c'est son rang et son score. Si l'on considère que le score de chaque document retourné est disponible, nous sommes confrontés aux problèmes d'hétérogénéité et d'autonomie des serveurs, qui engendrent l'incompatibilité des scores. En réalité, chaque serveur a sa propre stratégie de calcul du score, son propre modèle de pondération et ses propres caractéristiques, qui de ce fait rendent difficile de comparer les scores des documents d'un serveur avec les scores des documents d'un autre serveur.

Pour résoudre ce problème de fusion et d'incompatibilité des scores des documents provenant des différents serveurs, plusieurs approches ont été développées. Nous présentons dans la section 3.4 une étude des différentes méthodes de fusion.

## 3.2. Hétérogénéité des serveurs

Les serveurs sont construits et gérés de façon indépendante, utilisent leurs propres stratégies d'indexation et fonction de correspondance. Ils décident eux-même quand et comment leurs index sont mis-à-jour, évaluent la requête, délivrent la réponse à leur propre rythme. En un mot les serveurs sont autonomes.

L'autonomie des serveurs est la cause de plusieurs hétérogénéités, notamment :

1. la méthode d'indexation : un serveur peut choisir d'indexer le document en totalité ou en partie, de lemmatiser ou non les termes du document, de supprimer ou non des mots vides (tels que les adverbes, liaisons, etc.).

2. la pondération des termes : l'importance d'un terme dans la représentation d'un document est représentée par une valeur numérique appelée poids. Il existe une multitude de fonctions de pondération qu'un serveur peut adopter :

   - la pondération binaire qui indique la présence (poids égale à 1) ou l'absence (poids égale à 0) d'un terme dans un document ;

   - la pondération par la fréquence. Le poids d'un terme dans un document dépend du nombre de fois où ce terme apparaît dans ce document,

   - la pondération tf*idf, pour calculer le poids d'un terme, deux mesures sont utilisées : la fréquence du terme dans chaque document, et le nombre de documents dans la collection qui contiennent le terme

3. la fonction de correspondance : plusieurs fonctions de correspondance peuvent être utilisées pour calculer la similarité (score) entre un document et une requête.

4. les collections : les collections des différents serveurs sont, sans doute, dissemblables ne serait-ce que par leur contenu (collections spécialisées ou généralistes), leur langue et leur taille.

## 3.3. La sélection de serveurs

Le but de la sélection des serveurs est de réduire le nombre de serveurs à interroger pour une requête donnée, sans réduire l'efficacité du processus de recherche. Nous présentons dans cette section les méthodes proposées dans la littérature pour la sélection de serveurs.

### *3.3.1. Les méthodes de sélection de serveurs*

La sélection de serveurs peut se faire manuellement ou automatiquement.

#### *3.3.1.1. La sélection manuelle*

Dans cette méthode, le système confie la tâche de sélection de serveurs à l'utilisateur. L'utilisateur dans ce cas, se base sur les descriptions des serveurs ou sur des connaissances à priori acquises qui sont en général très restreintes. Par exemple West-Law[1] présente à l'utilisateur les sources de droit qu'il désire consulter.

[1] Service en ligne pour la recherche dans le domaine juridique : http://www.westlaw.com

### *3.3.1.2. La sélection automatique*

Dans la sélection automatique, c'est le système qui sélectionne les serveurs jugés capables de fournir les informations pertinentes à l'utilisateur. Plusieurs approches de sélection automatique ont été définies dans la littérature.

- ***CORI (COllection Retrieval Inference network) :*** Callan et al [callan et al 1995] considèrent dans leur système CORI une collection comme un immense document virtuel. Ainsi le classement des collections peut se faire de manière similaire au classement des documents dans la RI. Pour calculer le score d'une collection pour une requête donnée, ils utilisent la *fréquence collection* et la *fréquence document* pour chaque terme de la requête.

- ***Sélection par hiérarchie d'information :*** Cette méthode est basée sur le concept de hiérarchie d'informations conçue par Dolin et al [Dolin et al 1996]. Une hiérarchie d'informations est un arbre qui classifie l'information par certains critères, par exemple date, thème, région géographique, …etc. Une taxonomie d'informations est associée à chaque hiérarchie d'informations. Une taxonomie d'informations est représentée par un arbre dont les documents peuvent être classifiés selon leur contenu. Le nœud de chaque arbre est formé de deux champs : un label (terme ou phrase) et une valeur numérique représentant la proportion de documents dont le contenu correspond au label. Dans cette méthode deux types de courtiers existent ; les courtiers de haut niveau et les courtiers de niveau intermédiaire. Les courtiers de haut niveau gèrent des données sur les sources qui se limitent aux plus hauts niveaux des taxonomies d'information. Chaque courtier de niveau intermédiaire contient des informations associées à une hiérarchie d'informations. Ces informations représentent la partie de la taxonomie d'information associée à une source qui n'est pas mise dans le courtier de haut niveau. Le processus de sélection se fait en deux phases, les courtiers de haut niveau sélectionnent les courtiers du niveau intermédiaire estimés capabes de conduire à des serveurs pertinents. Ensuite, les courtiers sélectionnés vont à leur tour sélectionner les serveurs jugés pertinents.

- ***CVV (Cue Validity Variance) :*** La méthode *CVV* développée par Yuwono et al [Yuwono et al 1997] est basée sur la CVV des termes de la requête. La CV (Cue Validity) d'un terme pour une source d'information donnée, indique sa capacité à distinguer les documents entre eux. La CV d'un terme est similaire à l'Idf classique sauf que Idf d'un terme permet de distinguer les documents où il apparaît des autres documents d'une même source. La CVV est la variance de CV à travers toutes les sources du système. Un serveur est représenté par le nombre de documents que sa source contient et la fréquence-document de chaque terme qu'il contient. Le score d'un serveur est obtenu en combinant les CVV des termes de la requête et leurs fréquence-

documents. Le score d'un serveur fournit des indications sur la concentration des termes de la requête dans la source du serveur.

- ***GLOSS (GLOssary –of Servers Server):*** *GLOSS* introduit par Gravano et al [Gravano et al 1999] est un projet de l'université de Stanford qui a étudié le problème de la sélection de serveurs pour le modèle booléen. Cette méthode fonctionne dans un environnement booléen où chaque serveur utilise un SRI booléen. Gloss utilise la fréquence-document et la taille des sources pour calculer le score d'un serveur. Ce score estime le nombre de documents pertinents que le serveur contient. GLOSS, proposé initialement pour le modèle booléen *bGLOSS*, a ensuite été généralisé en *vGLOSS* pour prendre en compte le modèle vectoriel. Dans cette nouvelle version vGLOSS, ils supposent que le même SRI est utilisé par tous les serveurs. Pour une requête donnée, le score d'un serveur est la somme des scores de ses documents supérieurs à un seuil donné. Le score d'un document est estimé sur la base des informations que le système (courtier) détient sur le contenu des serveurs. Les informations concernant les serveurs sont représentées par deux vecteurs, celui des valeurs de fréquence documents pour chaque terme de la source et celui de la somme des poids de chaque terme dans les documents de la source.

- ***Sélection probabiliste :*** Cette approche introduite par *Baumgarten* [Baumgarten 1999a] [Baumgarten 1999b] est basée sur un modèle probabiliste. Le but de la sélection dans cette approche n'est pas d'estimer la distribution des documents pertinents, mais de détecter les sources qui permettent d'obtenir les *n* documents les mieux classés si toutes les sources sont sélectionnées.

- ***LWP (Leightweight probes) :*** L'approche LWP proposée par Hawking [Hawking et Thislewaite 1999] permet d'effectuer la sélection des serveurs en utilisant des requêtes de sonde (*probe queries*). Le principe de l'approche est d'envoyer au moment de l'interrogation dans un premier temps une requête de sonde à tous les serveurs. Une requête de sonde est une requête de taille réduite construite en gardant *p* termes de la requête initiale. Les serveurs doivent répondre à la requête de sonde par des informations nécessaires telles que le nombre total des documents que contient le serveur, le nombre de documents contenant les termes de la requête de sonde, des informations sur la proximité et la co-occurrence des termes de la requête de sonde. Ces informations servent à calculer le score de chaque serveur.

- ***CORI enrichi par la performance des SRI locaux :*** Craswell et al [Craswell et al 2000] partent de l'hypothèse suivante : *il n'est pas intéressant de sélectionner un serveur qui contient des documents pertinents, qui n'est pas capable de les retrouver*. A partir de là, ils proposent de prendre en compte dans la sélection des serveurs, non seulement la probabilité qu'un serveur contient des documents pertinents, mais aussi sa

capacité à les retourner. Pour vérifier cette proposition, ils ajoutent un paramètre de mesure de performance à la formule de calcul du score dans CORI. Pour calculer le paramètre de performance d'un serveur, des requêtes de sonde sont envoyées à chaque serveur, et les dix premiers documents retournés par chaque serveur sont considérés pour chaque requête de sonde. Ainsi, la valeur du paramètre de performance d'un serveur est la moyenne des proportions de ses documents, pour toutes les requêtes de sondes.

- **Autres méthodes de sélection :**

*Wais* [Kahle et Medlar 1991] est un système qui classifie et résume les serveurs manuellement. La sélection des serveurs se fait par rapport à la présence des termes de la requête dans les résumés des serveurs.

*Moffat et Zobel* [Moffat et Zobel 1995] proposent de subdiviser les sources en plusieurs blocs de B documents. Ainsi, les représentants des blocs qui sont sauvegardés au lieu des représentants des sources. L'intérêt de cette approche est de restreindre la recherche à des portions de sources les plus susceptibles de contenir des documents pertinents.

*NetSerf* [Chakravarthy et Haase 1995] introduit la sémantique des termes dans la sélection. Il utilise un thésaurus sémantique (WordNet) et un dictionnaire en ligne (Webster's Dictionnary) pour représenter la requête, l'étendre et de la désambiguïser. Ensuite il compare les représentations sémantiques de la requête avec les descriptions des sources.

*Xu et Croft* améliore les performances de CORI par deux manières. La première consiste à l'expansion de requête avant d'effectuer la sélection par CORI. Cette méthode d'expansion des requêtes est appelée *Local context analysis* [Xu et Croft 1996]. La seconde consiste à enrichir les représentants des serveurs avec des informations sur les phrases qu'ils contiennent [Xu et Callan 1998].

*SavySearch* [Dreilinger et Howe 1997] est une approche qui se base sur l'historique des interrogations des serveurs pour calculer leurs scores. Les scores des serveurs qui répondent à une requête contenant un terme $t_k$, seront augmentés pour les nouvelles requêtes contenant $t_k$. A l'opposé, les scores des serveurs qui ne répondent pas à une requête contenant un terme $t_k$, seront diminués pour les nouvelles requêtes contenant $t_k$.

*Fuhr* [Fuhr 1999] dans sa méthode DTF (decision-theoretic framework), prend en compte la qualité et le coût des SRIs des serveurs. Il procède aux étapes suivantes :

1. estimer le nombre de documents pertinents dans chaque serveur,

2. estimer la qualité de recherche de chaque serveur,

3. calculer le coût d'interrogation de chaque serveur en fonction du nombre de documents à retourner,

4. combiner les coûts d'interrogation des serveurs en une fonction de coût global. Cette fonction est ensuite minimisée pour déterminer le nombre optimal $N_i$ de documents à demander au serveur $S_i$. Ainsi, chaque $N_i$ non nul indique que le serveur $S_i$ est sélectionné.

*Xu et Croft* [Xu et Croft 1999] proposent le regroupement des documents des serveurs selon leur thème, un groupe est formé des *k* premiers documents d'une source. Chaque groupe est alors indexé. Pour chaque groupe, ils attribuent un vecteur des fréquences des termes dans ses documents. A la réception d'une requête, la pertinence d'un groupe à la requête est calculée. Ensuite, la requête est dirigée vers les sources qui contiennent les documents du groupe.

*Savoy et Rasolofo* [Savoy et Rasolofo 2000] utilisent une méthode d'apprentissage pour la sélection des serveurs, qui est basée sur un arbre de décision. Le processus d'apprentissage fait correspondre pour chaque serveur et chaque requête un ensemble de couples (attribut, valeur) et la décision prise sur la sélection ou non du serveur. Plusieurs attributs ont été pris en compte, à savoir, le nombre de termes de la requête, la fréquence document des premiers termes de la requête, la moyenne de la fréquence document des termes de la requête, … etc. Le traitement d'une nouvelle requête q consiste à trouver les trois requêtes d'apprentissage les plus proches de *q*. Ensuite, la décision prise pour la majorité de ces trois requêtes est adoptée pour *q*.

Si et Callan [Si et Callan 2004], propose une approche appelée UUM *(Unified Utility Maximization Framework for Resource Selection),* dans laquelle ils se basent sur l'estimation de la taille de la source, pour estimer le nombre de documents pertinents à la requête que la source peut avoir en utilisant des fonctions de probabilités. Ce nombre de documents pertinents estimé est utilisé pour la sélection de sources.

## 3.4. La fusion des résultats des serveurs

Après que les serveurs ont été sélectionnés en un sous-ensemble **S'** = $\{S_1, S_2, \ldots, S_k\}$ et interrogés, les listes de résultats $\boldsymbol{L_i}$ des serveurs de **S'** doivent être fusionnées en un seul classement $\boldsymbol{L}$.

Fusionner les résultats d'une requête venant des serveurs qui utilisent des algorithmes de classement différents et non connus est un problème difficile. Tout d'abord, les serveurs peuvent utiliser des échelles différentes pour classer les documents, de plus, ces échelles peuvent être normalisées ou non.

Fusionner les résultats d'une requête reste difficile même si les serveurs utilisent le même algorithme de classement. La raison en est qu'un même algorithme peut classer différemment des documents selon la source dans laquelle ils appartiennent. Supposons, par exemple, que les deux serveurs $S_1$ et $S_2$ utilisent la formule *tf-idf* pour calculer les poids. Si le serveur $S_1$ est spécialisé en informatique, le mot *ordinateur* apparaîtra dans un grand nombre de ses documents. Ce mot tendra donc à avoir un faible poids associé dans $S_1$. D'autre part, ce mot tendra à avoir un fort poids associé dans une source $S_2$ sans aucun rapport avec l'informatique et comprenant très peu de documents contenant ce mot. En conséquence, $S_1$ peut assigner à ses documents un faible score pour une requête contenant le mot *ordinateur*, tandis que $S_2$ assignera un score élevé à un petit nombre de documents pour cette même requête. Il est donc possible, pour deux documents similaires $d_1$ et $d_2$ de recevoir deux scores très différents pour une requête donnée, si $d_1$ apparaît dans $S_1$ et $d_2$ apparaît dans $S_2$.

### *3.4.1. Les méthodes de fusion*

Plusieurs méthodes de fusion ont été définies dans la littérature. Le principe général de ces méthodes est d'attribuer un score pour chaque document de chaque liste, sur la base de ces scores, les documents seront fusionnés et triés. Cependant, la difficulté se pose lors du calcul de ces scores. Nous présentons dans cette section un état de l'art sur les méthodes de fusion. Ces méthodes sont classées selon les informations disponibles sur les documents retournés par les serveurs.

#### *3.4.1.1. Les serveurs ne retournent aucune information sur les documents*

Dans le cas où les serveurs ne retournent que des listes non triées de documents, il existe deux façons pour les fusionner :

1. les documents sont regroupés au hasard, par ordre d'arrivée des listes, par longueur des listes …etc.

2. analyser le contenu des documents et leur attribuer des scores, utilisant une technique de pondération de la RI. Ensuite les classer selon ce score. [Lawrence et Lee Giles 1998].

#### *3.4.1.2. Les serveurs retournent les rangs des documents*

Lorsque les listes des résultats retournent les rangs des documents, les méthodes suivantes sont envisageables :

1. La méthode développée par Voorhess [Voorhess 1995], consiste à ; (1) Attribuer un score temporaire à chaque liste de réponses. Le score temporaire d'une liste est égal au nombre de documents qu'elle contient ; (2) Sélectionner la liste dont son score temporaire est le plus élevé ; (3) Extraire le document de tête de cette liste sélectionnée ; (4) placer ce document dans la liste finale à la position courante ; (4) diminuer de 1 le score temporaire de cette liste sélectionnée. Remprendre le processus à partir de (2) jusqu'à ce que tous les documents des listes seront inclus dans la liste finale.

2. Yumono et al [Yumono et Lee 1997] attribuent un score pour chaque serveur dans la phase de sélection des *serveurs*. Pendant la phase de fusion, le rang de chaque document est combiné avec le score de son serveur pour calculer le score de ce document. Les documents sont ainsi fusionnés et triés selon ce score.

   Le score d'un serveur par rapport à une requête $q$ noté $score\_serv_{i,q}$ est calculé comme suit :

$$score\_serv_{i,q} = \sum_{j=1}^{M} CVV_j * DF_{i,j}$$

   Avec $DF_{i,j}$ est la fréquence du terme $j$ dans le $i^{ème}$ serveur. $M$ est le nombre de termes dans la requête $q$. $CVV_j$ est une estimation de l'utilité du terme $j$ pour distinguer un serveur des autres, qui est calculé comme suit :

$$CVV_j = \frac{\sum_{i=1}^{|S|} (CV_{i,j} - CV_j)^2}{|S|}$$

   Avec $S$ est l'ensemble de serveurs. $CV_{i,j}$ mesure le degré pour lequel le terme $j$ distingue les documents du $i^{ème}$ serveur de ceux des autres serveurs. $CV_j$ est la moyenne de la population de $CV_{i,j}$. Les deux mesures sont calculées comme suit :

$$CV_{i,j} = \frac{\frac{DF_{i,j}}{N_i}}{\frac{DF_{i,j}}{N_i} + \frac{\sum_{k \neq i}^{|S|} DF_{k,j}}{\sum_{k \neq i}^{|S|} N_k}}$$

   Avec $N_k$ est le nombre de documents dans le $i^{ème}$ serveur.

$$CV_j = \frac{\sum_{i=1}^{|S|} CV_{i,j}}{|S|}$$

Une fois le score s*core_serv*$_{i,q}$ du $i^{ème}$ serveur pour une requête $q$ est calculé, ce dernier sera combiné avec le score du document $d$ du $i^{ème}$ serveur noté $r_{i,d}$ pour calculer le score final de ce document noté $s_{i,d}$ comme suit :

$$s_{i,d} = 1 - (r_{i,d} - 1) D_i$$

Avec $D_i$ est l'estimation de la distance de pertinence entre deux documents consécutifs dans le résultat du $i^{ème}$ serveur. Il est calculé comme suit :

$$D_i = \frac{score\_serv_{\min,q}}{H * score\_serv_{i,q}}$$

Avec $H$ est le nombre de documents dans la liste finale.

3. L'approche *round robin* appelée aussi *à chacun son tour* [Kwok et al 1995] considère que chaque serveur retournera un nombre approximativement égal de documents pertinents et que ceux-ci se retrouvent distribués de manière identique dans les réponses obtenues. Ainsi la liste finale sera construite de l'ensemble de documents en tête des listes à fusionner, suivi de l'ensemble des documents du deuxième rang et ainsi de suite.

#### *3.4.1.3. Les serveurs retournent les scores des documents*

Dans le cas où le score d'un document est retourné par le serveur, comment l'utiliser pendant la phase de fusion ? Dans ce contexte plusieurs méthodes ont été définies :

Fox et Shaw [Fox et Shaw 1994] ont défini six méthodes basées sur des fonctions de combinaisons. Le principe de ces méthodes est le même, il s'agit de fusionner les documents sur la base des scores (poids) donnés par les serveurs. Cependant, ces scores peuvent être différents pour un même document qui se trouve dans plusieurs listes. Les solutions apportées à ce problème sont les suivantes :

1. *CombMAX*: considère le poids maximum du document dans toutes les listes.

2. *CombMIN* : contrairement à *CombMAX*, celle-ci considère le poids minimum du document dans toutes les listes.

3. *CombMED* : cette approche prend en compte les deux raisonnements des fonctions *CombMAX* et *CombMIN*. Elle calcule la moyenne des scores du document dans toutes les listes. Si le document n'existe pas dans une liste son score sera considéré nul. CombMED est la moyenne des scores des documents dans toutes les listes.

4. *CombSUM* : elle calcule la somme des scores du document de toutes les listes. Si le document n'existe pas dans une liste son score sera considéré nul. CombSUM est la somme des scores du document dans toutes les listes.

5. *CombANZ* : ignore les résultats qui ne contiennent pas le document. Elle calcule la moyenne des scores des documents non nuls. CombANZ est la moyenne des scores non nuls des documents dans toutes les listes.

6. *CombMNZ* : elle favorise les documents retrouvés dans plusieurs listes de résultats en leur donnant des poids élevés ainsi CombMNZ=CombSUM*nombre des scores non nuls.

[Kwok et al 1995] se basent sur l'hypothèse que pour chaque document retourné, les serveurs fournissent le degré de similarité (score) entre la requête et le document (ou un degré estimé de la pertinence du document retourné). Ils admettent que le tri des documents provenant des différentes sources peut donc être effectué sur la base de ce score. Cette méthode est nommée *fusion par le score*, appelée aussi dans la littérature anglophone *Raw Score Merging* (*RSM*).

Cependant, les scores locaux sont très souvent peu comparables, même si la même méthode de recherche est utilisée par tous les serveurs car le score d'un document est souvent calculé à partir des statistiques propres au contenu de la source. Lu et al [Lu et al 1996] proposent, afin de diminuer l'impact de la non-comparabilité des scores sur le résultat final, de pondérer le score local d'un document par le score du serveur qui le retourne.

Une autre approche a été définie pour tenir compte du problème de la non-comparabilité des scores sur le résultat final, qui procède à une normalisation des scores, par exemple, en divisant tous les degré de similarité d'une liste par le score maximal contenu dans cette liste (soit celui du premier article extrait). Une telle stratégie de fusion se notera fusion *par le score normalisé* [Voorhees et al 1995a] [Powell et al 2000]

Callan et al [Callan et al 1995] dans leur méthode CORI, calculent un facteur (coefficient) de pondération pour chaque source, indiquant l'importance relative à attribuer pour chaque

source. Durant la phase de fusion des listes retournées par les différentes sources, le degré de similarité de chaque document est multiplié par ce coefficient, sur la base des scores obtenus, les documents seront fusionnés et triés.

Dans la stratégie proposée par Le Calvé et Savoy [Le Calvé et Savoy 2000], la fusion peut s'effectuer selon la probabilité que le document soit pertinent, probabilité estimée selon la méthode de la régression logistique [Bookstein et al. 1992] sur la base du rang et du score obtenus par ce document.

Si et callan [Si et Callan 2003] [Si et Callan, 2005], proposent une autre approche de fusion appelée SSL (Semi-Supervised Learning). Cette approche attribue à chaque document retourné par les serveurs deux scores. Le premier score est attribué par le serveur dans sa liste de réponse. Le second score est calculé par l'approche en appliquant une technique de pondération jugée efficace. La collection de documents considérée dans le calcul du second score est l'ensemble des documents de toutes les listes des résultats. Les deux scores d'un document seront ensuite combinés pour lui attribuer un seul score final. Sur la base de ces scores finaux, les documents seront fusionnés et triés.

## 3.5. Evaluation des SRID

Le but d'un SRID est identique au but d'un SRI, c'est de retourner des résultats satisfaisant la requête utilisateur. Pour pouvoir évaluer les différents SRI, des collections de tests on été créées. De telles collections contiennent un ensemble de documents, un ensemble de requêtes et une liste de jugements de pertinence associée à chaque requête. Les collections de tests les plus connues sont celles de TREC (Text REtrieval Conference), de CLEF (Cross-Language Evaluation Forum) et de NTCIR (National institute of informatics Test Collection for Information Retrieval).

Il n'existe actuellement pas de collection de test spécifique pour l'évaluation des SRIDs. Une solution est de subdiviser l'ensemble des documents d'une collection de test en plusieurs sous-ensembles [Abbaci 2003] [Bailey et al 2003].

L'évaluation d'un SRID, peut porter sur plusieurs critères, à savoir : la qualité de recherche, le temps de réponse, l'extensibilité du système, le coût des mises à jour, la quantité d'information à transférer et la couverture de l'information.

Nous pouvons catégoriser les méthodes d'évaluation des SRID en deux catégories. (1) Les méthodes d'évaluation du processus de sélections de serveurs. (2) Les méthodes d'évaluation du processus de fusion.

### *3.5.1. Evaluation du processus de sélection des serveurs*

L'évaluation du processus de sélection de serveurs revient à déterminer sa capacité et son efficacité à sélectionner tous les serveurs pertinents et rien que des serveurs pertinents. Plusieurs méthodes d'évaluation ont été proposées dans la littérature. Ces méthodes se basent sur le classement des serveurs. L'évaluation consiste à comparer le classement automatique obtenu par le courtier au classement idéal. Le classement idéal d'un serveur est le classement par le score idéal des serveurs, qui dépend de la méthode utilisée pour calculer ce score. Par exemple, dans le cas où le score d'un serveur représente le nombre présumé de documents qu'il contient, son score idéal est le nombre exact de documents pertinents qu'il contient. Ce nombre est donné par les jugements de pertinence qui font partie des collections de tests. Parmi les méthodes d'évaluation nous citons :

1. Callan et al [Callan et al 1995] calculent un taux d'erreur du classement des serveurs, appelé *Mean-Squared error (MSE).*

$$MSE = \frac{1}{|S|}\sum_{i \in S}(Id_i - O_i)^2$$

   Où : S est la liste des serveurs triée par la méthode de classement à évaluer ; $Id_i$ est le rang du $i^{ème}$ serveur dans le classement idéal ; $O_i$ est le rang du $i^{ème}$ serveur attribué par la méthode de sélection à évaluer.

L'inconvénient de cette méthode est qu'elle ne rend pas compte de l'importance de l'erreur. Par exemple, si nous avons deux cas de classements ; le $1^{er}$ inverse le rang entre deux serveurs dont la différence de documents pertinents qu'ils contiennent est grande. Le $2^{éme}$ inverse le rang entre deux serveurs dont la différence de documents pertinents qu'ils contiennent est minime. L'erreur MSE calculée pour les deux cas de classements sera la même. Or, en réalité le nombre de documents pertinents qui peuvent être inversés dans les deux classements ne sera pas le même.

2. Gravano et al [Gravano et al 1999] utilisent des mesures similaires aux mesures conventionnelles du classement des documents dans la RI à savoir la *précision* et le *rappel,* noté respectivement *P* et *R*. Rappelons que, pour une requête donnée :

   - la précision détermine la capacité du système à ne retourner que les documents pertinents.

   - le rappel détermine la capacité du système à retourner tous les documents pertinents.

   *P et R sont* calculés comme suit :

$$P = \frac{\left| \left\{ S_i \in S / \quad score(S_i) > 0 \right\} \right|}{|S|}$$

$$R = \begin{cases} \dfrac{\sum_{i=1}^{i=|S|} Score\ (S_i)}{\sum_{i=1}^{i=|S|} ScoreI\ (S_i)} & Si \quad \sum_{i=1}^{i=|S|} ScoreI\ (S_i) > 0 \\ 1 & Sinon \end{cases}$$

où $Score(S_i)$ est le score du serveur $S_i$, $Scorel(S_i)$ son score idéal pour la requête de l'utilisateur. L'inconvénient de cette approche est qu'elle ne tient pas compte du nombre de documents pertinents dans un serveur. Par exemple, l'inversion de rang entre les serveurs ne contenant pas le même nombre de documents pertinent n'est pas pénalisée.

3. Lu et al [Lu et al 1996] calculent les mesures de *Rappel* et *Précision* en prenant en compte le nombre de documents pertinents contenus dans les serveurs, comme suit :

$$P = \frac{\sum_{i=1}^{|S|} n\_pert_i}{|S|} \qquad R = \frac{\sum_{i=1}^{|S|} n\_pert_i}{n\_pert}$$

où : $n\_pert_i$ est le nombre des documents pertinents contenus dans le serveur $S_i$
$n\_pert$ est le nombre total des documents pertinents contenus dans l'ensemble des éléments de $S$.

### *3.5.2. Evaluation du processus de fusion*

L'évaluation du processus de fusion permet d'évaluer le résultat final du système. Cette évaluation utilise les métriques de la RI classique (la précision et le rappel) :

$$P = \frac{|pert \cap ret|}{|ret|} \quad avec \quad |ret| \neq 0$$

$$R = \frac{|pert \cap ret|}{|pert|} \quad avec \quad |pert| \neq 0$$

Où : *pert* est l'ensemble des documents pertinents contenus dans le corpus (collection) pour une requête donnée.

*Ret* est l'ensemble des documents que le système a retournés ;

Généralement, lors d'un test d'évaluation d'un SRI, plusieurs requêtes sont utilisées. Dans ce cas, c'est la moyenne des valeurs de précision et de rappel pour l'ensemble des requêtes qui est calculée.

## 3.6. Etude comparative des méthodes de sélection de serveurs

Les différents environnements d'évaluation utilisés dans la littérature sont : TREC (Text REtrieval Conference) ; TREC-x (corpus de la x-ième conférence de TREC) ; CACM, CISI, MED (corpus constitués et distribués par Gérald Salton et ses étudiants) ; CRAN (Cranfiel collection).

Les méthodes de sélection retrouvées dans la littérature sont évaluées dans des environnements différents et avec des mesures d'évaluation différentes. Il est alors difficile de comparer leurs efficacités. Afin de faire une étude comparative de ces méthodes, nous décrirons quelques études expérimentales publiées.

La méthode *CORI* a été évaluée par Callan, Lu et Croft [Callan et al 1995], sur le corpus TREC-1 divisé en 19 sources équivalentes en nombre de documents. Cependant, aucune comparaison de CORI à d'autres approches n'a été effectuée.

La méthode vGloss est évaluée par Gravano et Garcia-molina [Gravano et Garcia-Molina 1995] en utilisant les deux métriques *rappel* et *précision*. Ils ont montré qu'elle donne de bons résultats, comparée au classement idéal. Le classement idéal consiste à classer les serveurs selon leurs scores, le score d'un serveur est obtenu par la somme des scores de ses documents qui sont supérieurs à un certain seuil.

L'approche proposée par Moffat et al [Moffat et Zobel 1995] a été évaluée en utilisant différentes valeurs de la taille des blocs, et comparée à l'approche centralisée. Les résultats ont montré que leur approche n'est bénéfique que dans le cas de subdivision en blocs de dix documents. Ce cas a l'inconvénient, qu'au niveau du courtier, l'index des blocs est de taille trop importante.

Yumono et lee [Yumono et Lee 1997] ont évalué plusieurs approches de sélection : *CVV, vGloss, bGloss* et *CORI*, en utilisant les métriques *rappel* et *précision*. Les résultats obtenus montrent que l'approche CVV est la plus efficace, vient ensuite vGloss, et en dernier CORI et bGloss. Cependant, ces résultats ne sont pas considérés convaincants. Cela

est dû aux corpus CACM, CISI, CRAN et MED, utilisés, qui sont considérés trop petits (7 MO en total, et 7097 documents).

Hawking et Thislewaite [Hawking et Thislewaite 1999] comparent la méthode *LWP* par rapport à celle de *Voorhees,* utilisant l'ensemble des documents de TREC5, divisé en 6 sources, ensuite en 98 sources. Les sources ont approximativement les mêmes tailles en octets qui varient de 1548 à 8527 documents. L'étude a montré que LWP est la plus performante.

Xu et Croft [Xu et Croft 1999] ont évalué leur méthode de sélection basée sur le regroupement par thèmes, sur un corpus de TREC subdivisé en 100 sources. Ils ont montré la performance de leur approche par rapport à *CORI.*

Craswell, Bailey et Hawking [Craswell et al 2000] comparent plusieurs méthodes de sélection de serveurs à savoir *CORI, vGloss, CVV et CORI-enrichi.* Utilisant 956 sources obtenues à partir des documents de TREC8 [Hawking et al 1999]. Ils ont montré que la méthode CORI dépasse vGloss qui surpasse CVV dans le cas de la sélection de dix serveurs sur 956. Cependant, *CORI-enrichi* ne présente aucune amélioration significative lorsque les performances des serveurs sont calculées automatiquement. Mais dans le cas où elles sont calculées manuellement l'amélioration des résultats est importante.

Savoy et Rasolofo [Savoy et Rasolofo 2000] évaluent leur approche de sélection sur 4 sources, en variant les SRI locaux. Leur approche s'avère moins performante que *CORI.*

Si et Callan [Si et Callan 2004], ont montré que leur méthode UUM est similaire à la méthode DTF, car les deux se basent sur d'autres paramètres et pas seulement sur la pertinence de document pour la sélection des serveurs, ce qui n'est pas le cas des autres approches. Cependant, leur méthode est meilleure que DTF, surtout dans le cas d'un nombre important de sources, car elle nécessite moins d'informations dans ses estimations.

Les études expérimentales précédentes, ont montré qu'il est difficile de confirmer définitivement quelle est la méthode la plus performante par rapport à toutes les autres méthodes.

## 3.7. Etude comparative des méthodes de fusion

Dans le cas où les serveurs ne retournent aucune information sur les documents, la méthode de regroupement par hasard, ne se base sur aucune information de pertinence des documents pour les classer. Cela court un risque très élevé de tomber dans le cas où des documents non pertinents sont bien classés. Cependant, la méthode proposée par [Lawrence et Lee Giles 1998], permet de classer les documents selon un ordre de pertinence des documents calculé sur l'ensemble de documents après réception des listes

des résultats. Cette méthode diminue le risque d'avoir des documents non pertinents en premier.

Dans le cas où les serveurs ne retournent que les rangs des documents, la stratégie proposée par Voorhees [Voorhess 1995] utilise le nombre de documents dans la liste de réponse comme paramètre pour calculer la pertinence d'un document. Ainsi cette stratégie favorise les documents provenant des listes de grande longueur, qui n'est pas toujours le cas, car un serveur peut retourner un nombre important de documents non pertinents, cependant, qu'un autre serveur peut retourner un petit nombre de documents mais pertinents. La stratégie *Round Robin*, n'aura une signification que dans le cas où tous les serveurs utilisent la même technique de pondération. Même dans ce cas, les scores des documents de différents serveurs peuvent être non comparables, puisque les informations utilisées par les serveurs ne sont pas les mêmes [Dumais 1994]. Cette stratégie permet d'obtenir une précision moyenne d'environ 40 % inférieure à l'interrogation d'une source unique regroupant tous les documents [Voorhees et al. 1995a][Callan et al 1995]. Ainsi, la stratégie proposée par Yuwono et al [Yumono et Lee 1997] donne des résultats meilleurs, car elle permet d'équilibrer les rangs des différentes listes par le score des serveurs qui les retournent.

Dans le cas où les serveurs retournent les scores des documents, l'expérimentation faite par Fox et Shaw [Fox et Saw1994] a montré que les combinaisons *CombANZ* et *CombMNZ* donne quelquefois de meilleures performances que *CombMAX* et *CombMIN*, cependant, *CombMNZ* fournit parfois une performance légèrement meilleure que la combinaison *CombANZ*. La combinaison *CombSUM* fournit une efficacité de recherche meilleure que *CombMAX*, *CombMIN* et *CombANZ*. La combinaison *CombMNZ* est meilleure que *CombSUM* parce qu'elle favorise les documents qui se retrouvent dans plusieurs listes de réponse.

Jacques Savoy et Yves Rasolofo [Savoy et Rasolofo 2000], ont expérimenté quatre approches : fusion par le score, score normalisé, CORI et fusion par probabilité. Les résultats obtenus sont comparés au cas où toutes les sources sont considérées comme un seul corpus, c'est-à-dire l'interrogation d'un corpus unique (pas de fusion). L'expérimentation a montré que la stratégie *a chacun son tour* présente une perte de précision moyenne de 21,72%, la stratégie *fusion par le score* une perte de 4,61%, la stratégie *fusion par probabilité* une perte de 3,93% et la stratégie *CORI* sans sa procédure de sélection a donné une perte de 12,42%. D'après cette expérimentation, ils constatent que la fusion par le score présente une performance intéressante et similaire à la meilleure approche soit la fusion par probabilité. Le modèle CORI calculé sans sa procédure de sélection occupe la troisième place et les stratégies *à chacun son tour* ou du score normalisé doivent être abandonnées.

Si et callan [Si et Callan. 2003] dans leurs expérimentation, ont comparé leur stratégie de fusion *SSL* par rapport à l'approche *CORI*. Les résultats obtenus ont montré que la stratégie *SSL* est plus performante.

# 4. Conclusion

Dans ce chapitre, nous avons présenté différents thèmes de recherches soulevés par la recherche d'information distribuée (RID). Deux problèmes centraux de la RID, à savoir, la sélection de serveurs et la fusion des résultats sont considérés.

Bien que l'approche de la RI distribuée tente de surmonter bon nombre des limitations de la solution de recherche centralisée, elle présente certains désavantages, notamment dans la performance et le temps de réponse. Entre la réception d'une requête et la livraison des résultats, un serveur de recherche centralisé n'a besoin de réaliser que des calculs et des opérations sur disque. Au lieu de cela, un system de RID est sûrement plus lent, réalisant d'abord une sélection, puis une recherche qui ne se termine qu'avec l'arrivée sur le réseau de la liste des derniers résultats et enfin une fusion des résultats.

La solution centralisée peut fournir des résultats rapides et efficaces, au prix de la construction, du maintien et de la recherche dans un gros index, et du manque de certains documents. Un système distribué découvrira un plus grand nombre de documents, au prix de la rapidité du traitement de la requête.

Dans ce contexte, plusieurs méthodes ont été développées dans le but d'améliorer la performance et le temps de réponse d'un système de RID. Nous avons présenté une étude des méthodes de sélection et de fusion recensées dans la littérature. Nous avons également rapporté les résultats des différentes études expérimentales publiées dont le but est de comparer les performances de certaines méthodes de sélection et de fusion. A travers cette étude, il s'avère qu'il est difficile d'avoir une idée très claire et précise des performances de ces méthodes à cause de la diversité des environnements de tests comme TREC-x, CACM, CISI, MED, CRAN, VLC2, WT2g, WT10g….. etc.

Nous avons finalement abouti aux constations suivantes :

- la majorité des méthodes, soit de sélection de serveur ou de fusion des résultats se basent sur la pertinence des résultats par rapport à la requête seulement pour mesurer la performance de la recherche.

- Certaines méthodes prennent en compte d'autres critères comme la qualité, le coût et le temps de réponse des systèmes de recherche locaux.

A travers cela, nous avons constaté que la pertinence des résultats par rapport à l'utilisateur n'est pas prise en compte par ces méthodes.

Pour mieux répondre aux besoins de l'utilisateur en informations, il faut inclure d'autres critères que l'utilisateur peut préférer, un par rapport à l'autre, comme le type d'information, la langue de l'information, le temps de réponse, la qualité d'information (la fiabilité, la fraîcheur, l'exactitude, la sécurité, le prix…). L'objectif de cette thèse est d'inclure ces différents critères propres à l'utilisateur, pour développer une approche de sélection et de fusion personnalisée et adaptée au contexte de l'utilisateur. L'utilisation de ces critères, nous permet de mieux cibler les préférences de l'utilisateur.

Avant d'aborder notre travail, nous présentons dans le chapitre suivant l'utilité d'intégrer l'utilisateur dans le processus de recherche d'information, à travers une synthèse des modèles et techniques dans le domaine de la recherche d'information personnalisée.

# Chapitre 3

# Accès personnalisé à l'information

## 1. Introduction

La personnalisation de l'information constitue un enjeu majeur pour l'industrie informatique. Que ce soit dans le contexte des systèmes d'information d'entreprise, du commerce électronique, de l'accès au savoir et aux connaissances ou même des loisirs, la pertinence de l'information délivrée, son intelligibilité et son adaptation aux usages et préférences des clients constituent des facteurs clés du succès ou du rejet de ces systèmes. La production massive de nouvelles ressources, conjuguée à la nature fortement hétérogène, multiforme et multilingue de l'information constituent des verrous autant technologiques que sémantiques que les systèmes d'accès et de filtrage actuels doivent lever pour produire une information pertinente et de qualité. La personnalisation a pour objectif, d'une part, de faciliter l'expression du besoin utilisateur et de rechercher des informations sur un sujet en écartant l'information non-pertinente et donc réduire considérablement l'espace de recherche et, d'autre part, de rendre cette information sélectionnée intelligible à l'usager et exploitable. La pertinence de l'information n'est cependant pas une mesure objective, généralisable à tous les utilisateurs. Elle se définit par un ensemble de critères et de préférences personnalisables spécifiques à chaque utilisateur ou communauté d'utilisateurs.

En effet, deux raisons fondamentales plaident pour la personnalisation :

- les utilisateurs ont des objectifs différents, des contextes différents et perceptions différentes de la notion de pertinence,
- un même utilisateur peut avoir différents besoins à différents instants.

Plusieurs approches de personnalisation ont été définies dans la littérature. Quelque soit l'approche de personnalisation, on a toujours besoin de collecter et sauvegarder des données décrivant les utilisateurs sous forme de ***profils***. Il convient de distinguer la notion de *profil* de la notion de *requête*. Un profil peut être défini comme une mise en équation du centre d'intérêts et des préférences de l'utilisateur, alors qu'une requête est l'expression

d'un besoin circonstancié que l'utilisateur souhaite voir satisfait en tenant compte de son profil. Un profil a un caractère plus invariant que les requêtes même si le centre d'intérêts et les préférences de l'utilisateur peuvent légitimement évoluer. Un profil peut être modélisé directement à partir des besoins et des préférences des utilisateurs ou découvert par des méthodes de *data mining* à partir de leur comportement passé.

## 1.1. Emergence de la personnalisation

De nombreux modèles théoriques sont à la base de la conception des systèmes de recherche d'information. Un modèle de recherche d'information est généralement décrit comme un quadruplet [D, Q, F, R(qi, dj)] [Yates et Neto 1999 ; Lechani et Boughanem 2005].où :

D : ensemble de documents,
Q : ensemble de requêtes,
F : schéma de représentation des documents et requêtes,
R(qi,dj) : fonction de pertinence.

La mise en œuvre naïve d'un tel modèle suppose que l'utilisateur est complètement représenté par sa requête et que les résultats retournés pour une même requête sont identiques même si elle est exprimée par des utilisateurs différents. Dans le but de montrer l'incidence d'une telle représentation, considérons à titre illustratif la requête : ***information about cats.*** On peut supposer différents scénarios d'utilisation :

- scénario 1 : étudiants vétérinaires qui écrivent un papier sur les cancers des chats,
- scénario 2 : architectes qui travaillent sur un projet de construction,
- scénario 3 : étudiants qui écrivent un papier sur l'Egypte ;

Ces différents scénarios illustrent sans doute la différence d'interprétation d'une requête en fonction du contexte de l'utilisateur qui l'exprime. Les problèmes immédiats posés par la non considération du contexte en cours du processus de recherche d'information sont notamment :

- l'ambiguïté du sens des mots,
- l'impossibilité de sélectionner des sources opportunes,
- l'inintelligibilité des résultats.

En outre, ces problèmes sont d'autant plus accentués que les requêtes sont courtes et que les sources d'informations sont volumineuses et hétérogènes. Ceci a pour corollaire la non pertinence des résultats de recherche et de fait, l'insatisfaction de l'utilisateur. Les premières solutions apportées à ce type de problèmes et pouvant s'apparenter à la personnalisation sont les techniques de reformulation de requête par injection de pertinence [Rocchio 1971 [Salton et Buckley 1990] et expansion de requêtes en utilisant des techniques de désambiguïsation [Chen et Wilensky 1997] [Dean et Henzinger 1999].

- *Reformulation de requête par injection de pertinence (relevance feedback)*
  C'est un processus évolutif et interactif, dirigé par l'utilisateur ayant pour objectif la génération d'une nouvelle requête plus adaptée que celle initialement exprimée par l'utilisateur. Son principe fondamental est d'utiliser la requête initiale pour amorcer la recherche puis modifier celle-ci à partir des jugements de pertinence et/ou de non pertinence de l'utilisateur. La nouvelle requête obtenue à chaque itération feedback, permet de corriger la direction de recherche dans le sens des documents pertinents au sens exprimé explicitement par l'utilisateur.

- *Désambiguïsation du sens des mots*
  Les techniques de désambiguïsation ont pour objectif d'identifier précisément le sens évoqué par les termes de la requête et cibler ainsi les documents contenant les mots cités dans le contexte défini par le sens correspondant. Ces techniques sont généralement basées sur une intervention explicite de l'utilisateur ou exploitation de ressources telles que les thesaurus et ontologies.

Cependant, vu le contexte actuel lié au volume d'informations, ces techniques sont peu viables [Jansen et al 1998] [Spink et al 1998]. En effet, la recherche devient performante après un nombre relativement élevé d'itérations feedback alors que celle-ci engendre une surcharge cognitive de l'utilisateur et par conséquent, une démotivation de ce dernier et donc un manque de fiabilité voire absence de son intervention (jugement de pertinence de documents, choix des termes, etc….). De plus, la stratégie de recherche n'est pas adaptée à un contexte d'utilisation dans différents domaines d'intérêts ; le contexte de l'utilisateur demeure peu connu et par conséquent de moindre impact sur le processus de recherche. C'est pourquoi les travaux de recherche se sont orientés vers *la recherche d'information contextuelle (modélisation de l'utilisateur)* [Budzik et Hammond 2000] [Su et Lee 2003].

### *1.1.1. Notion de contexte*

Le contexte de l'utilisateur peut être assimilé à l'ensemble des facteurs qui permettent de décrire ses intentions et perceptions de ce qui l'entoure. Ces facteurs peuvent couvrir divers aspects : psychologiques, sociaux, culturels, professionnels etc…

Vu sous l'angle de la recherche d'information, le contexte possède, d'après N. Fuhr [Fuhr 2000], trois principales dimensions : *social*, *application* et *temps*. La dimension sociale définit la composante d'appartenance de l'utilisateur : individuel, groupe ou communauté. La dimension application définit le contexte applicatif du besoin exprimé : recherche ad-hoc, résolution de problème ou *workflow*. La dimension temps permet de décrire la circonscription temporelle du besoin exprimé : temps passé (*Batch*), instant courant ou à court terme (*interactive*), intention ou long terme (*personnalisation*).

### *1.1.2. Recherche d'information contextuelle*

La recherche d'information contextuelle combine un ensemble de technologies et de connaissances portant sur la requête et le contexte utilisateur, dans une même infrastructure et ce, dans le but de délivrer les requêtes les mieux appropriées à son besoin en information [Fuhr 2000]. La recherche d'information contextuelle est une activité qui fait intervenir en grande partie l'utilisateur, assimilé en pratique, non plus à la requête, mais à un ensemble de facettes qui le décrivent. Ces facettes décrivent une structure qui est appelée communément profil et qui couvre, de manière non exhaustive, ses caractéristiques socio – professionnelles, ses centres d'intérêts et ses préférences.

Ce courant de recherches est assez récent (début des années 1990) et a suscité d'emblée des travaux émanant de plusieurs communautés : apprentissage automatique, modélisation utilisateur, recherche d'information, Web learning, analyse du langage naturel etc… les travaux visent principalement trois objectifs : accès personnalisé à l'information, assistance personnalisée à la navigation et présentation personnalisée des résultats de recherche [Brusilovsky et Marbury 2002].

Indépendamment de l'objectif applicatif visé, on identifie trois aspects à promouvoir dans les systèmes de recherche d'information contextuelle :

- capacité à identifier l'intention conceptuelle de l'utilisateur,
- possibilité d'exprimer des informations liées au contexte d'utilisation courant,
- convivialité des interactions utilisateurs – système.

### *1.1.3. La personnalisation de l'information*

La personnalisation de l'information est projetée sur deux niveaux. Le premier niveau est lié à l'introduction de la dimension utilisateur lors de la mise en œuvre d'un processus d'accès à l'information. Le second niveau se rapporte à l'exploitation des systèmes d'accès personnalisé.

## 1.2. Le premier niveau

L'introduction de la dimension utilisateur dans un processus d'accès à l'information, mérite voire nécessite des réflexions sur la modélisation de l'entité utilisateur de manière intrinsèque puis sous l'angle de son rapport avec une activité de recherche d'information circonscrite dans un court et/ou long terme [Pretschner et Gauch 1999][Li 2000][Gowan 2003]. Dans ce sens, les questions fondamentales posées par la conception de systèmes d'accès personnalisé sont de type Quoi, Comment, et Quand :

- Quoi ?
    - Quelles propriétés dérivent un utilisateur ?
    - Quelle représentation ou quel modèle de l'utilisateur adopter ?
    - Quel contexte d'utilisation considérer ?

- Comment ?
    - Comment construire le modèle de l'utilisateur ?
    - Comment découvrir son intention courante ?
    - Comment exploiter le modèle utilisateur lors du processus de recherche ?
    - Comment évaluer l'impact de la personnalisation sur le processus de recherche d'information ?

- Quant ?
    - Quand faut-il mettre à jour le modèle de l'utilisateur ?

L'ensemble des approches de personnalisation de l'information proposées dans la littérature tentent de répondre peu ou prou à ces questions.

### 1.3. Le Deuxième niveau

L'exploitation des systèmes d'accès personnalisé pose un problème fondamental, d'ordre technologique, qui porte sur la protection de la vie privée. En effet, la définition, utilisation et dissémination des profils constituent à la fois un tout et une contrainte pour assurer la portée des systèmes qui les supportent. C'est tout d'abord un atout dans le sens où les différents profils sont maintenus et sont accessibles, pouvant donc contribuer à mettre en œuvre une recherche d'information collective et dynamique par l'introduction de techniques d'apprentissage. Cependant, c'est ensuite une contrainte dans le sens où elle doit être impérativement soutenue par une réflexion sur les droits des personnes pour assurer la sécurité globale des profils.

## 2. Architecture générale

L'architecture générale d'un système d'accès personnalisé à l'information peut être présentée par la figure suivante (figure 3.1). Elle met en évidence l'ensemble des fonctionnalités de tels systèmes même si elles ne sont pas toujours présentes collectivement dans tout système. Cette architecture est centrée autour de l'utilisateur. En ce sens l'ensemble des modules du système fait intervenir en partie les informations descriptives du profil utilisateur. Sur la base de cette architecture, deux fonctions fondamentales sont dégagées : la gestion des profils et la sélection de l'information [Lechani et Boughanem 2005].

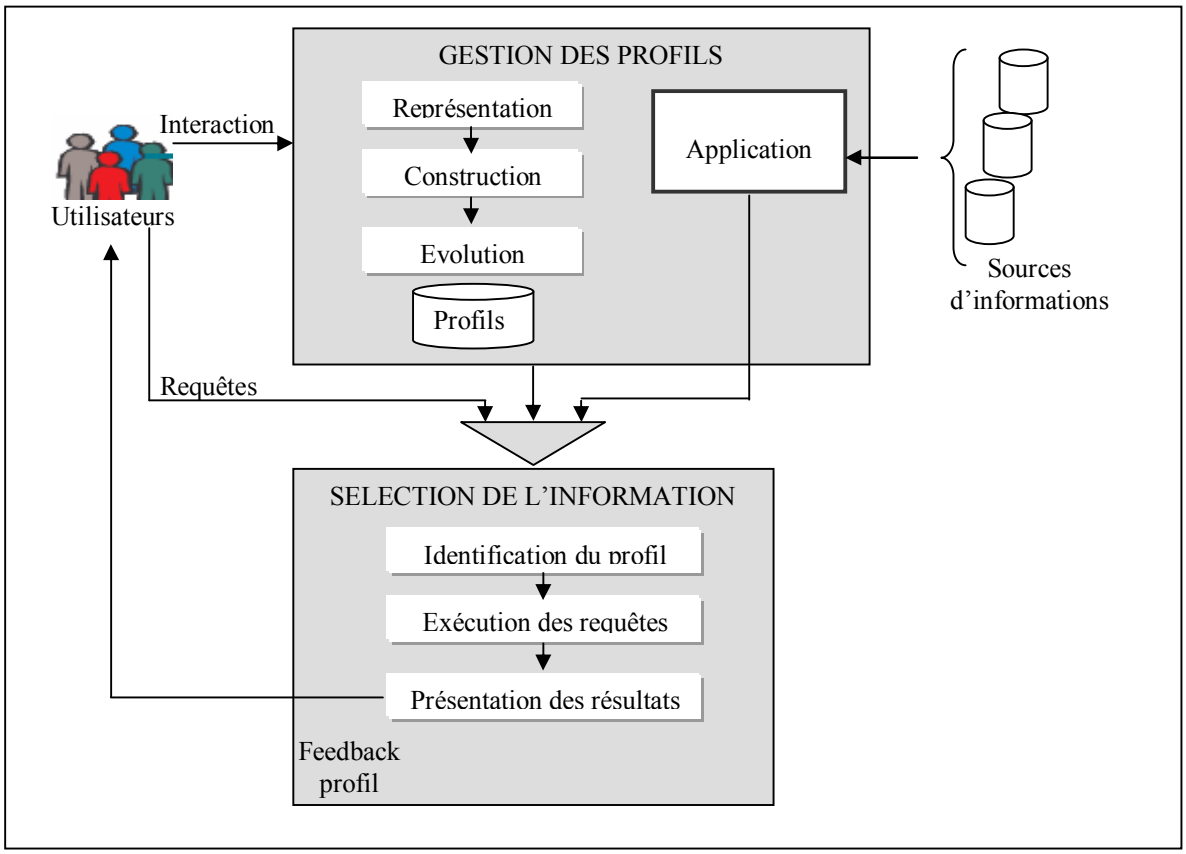


**Figure 3.1.** Architecture fonctionnelle d'un système d'accès personnalisé à l'information

## 2.1. Gestion des profils

La représentation de l'utilisateur à travers la notion de profil permet de mieux comprendre ses mécanismes cognitifs, notamment ceux permettant de percevoir le concept subjectif de la pertinence et au-delà, cibler ses besoins spécifiques dans le but d'améliorer les performances de recherche. [Daniels 1986] définit deux classes de modèles de profils utilisateurs :

- les modèles quantitatifs et empiriques : leur but est de modéliser le comportement externe de l'utilisateur,
- les modèles analytiques et cognitifs : leur but est de comprendre le comportement interne de l'utilisateur : connaissance, raisonnement etc.

Ces deux aspects sont généralement combinés pour représenter, construire et faire évoluer les profils

### *2.1.1. Représentation*

Le profil de l'utilisateur n'a pas forcément de structure explicite qui le représente. Il peut être constitué de paquets divers d'information qui traduisent une connaissance éparse sur l'utilisateur. Dans ce sens, la représentation des profils rejoint en grande partie la représentation de l'information dans le contexte de la recherche d'information. Il n'y a pas à notre connaissance de modèle spécifique, dédié à la représentation du profil de l'utilisateur. Les modèles proposés puisent largement de ceux proposés en recherche d'information. On en cite principalement quatre types de représentation

- ***Représentation vectorielle :*** Ce type de représentation s'appuie généralement sur le modèle vectoriel [Salton 1971]. Dans ce type de représentation, un profil utilisateur se présente sous la forme d'un vecteur ou ensemble de vecteurs de mots-clés (termes) et éventuellement des poids associés. Un poids traduit l'importance de chaque terme. Chaque ensemble de termes est associé à un utilisateur, qui définit son intérêt et son besoin en information. Ces termes sont généralement tirés des documents et pages web qui constituent le modèle utilisateur. Le choix de ces termes est basé généralement sur la technique TF*IDF, qui permet de trouver les termes les plus représentatifs d'un document. [Baudisch 1997][Callan 1998]. L'utilisation de plusieurs vecteurs permet de prendre en compte la diversité des domaines d'intérêts ou évolution dans le temps. Ce type de représentation est simple à mettre en œuvre. Cependant, il ne met pas en évidence ni la dimension liée au temps marquant l'évolution des profils, ni à l'organisation des informations pour hiérarchiser les centres d'intérêts.

- ***Représentation sémantique :*** La représentation sémantique met d'avantage en relief les relations de sens entre unités d'information représentant le profil en apportant des solutions aux problèmes de dissémination et synonymie. La direction proposée dans ce contexte, est la construction hiérarchique de concepts plutôt qu'une liste de structures indépendantes, à partir d'informations issus des fichiers *logs*. La hiérarchie peut rendre compte des niveaux de préférences de l'utilisateur et des associations latentes entre concepts et donc un raisonnement sémantique pour la dérivation du profil s'y prête aisément. Ce type de représentation utilise généralement des ontologies [Pretschner et Gauch 1999][Scime et al 2000].

- ***Représentation*** **connexionniste :** Ce type de représentation est basé sur l'interconnexion de nœuds représentant les termes, préférences ou document [Jenning et al 1993]. Il offre le double avantage de la structuration et de la représentation associative permettant de considérer l'ensemble des aspects représentatifs du profil.

- ***Représentation*** **multidimensionnelle :** Cette représentation se base sur le principe que le modèle utilisateur doit contenir toutes les dimensions qui représentent l'utilisateur. L'idée de cette représentation est de diviser le profil utilisateur en plusieurs catégories

chaque catégorie contient des informations spécifiques. [Bouzeghoub et Kostadinov 2004][kostadinov 2003] proposent une représentation du profil par un ensemble de dimensions, distinguant principalement huit dimensions : les données personnelles, le domaine, le centre d'intérêts, la qualité des résultats, la customisation, la confidentialité, le feedback, et les informations générales.

### *2.1.2. Construction*

La construction du profil traduit un processus qui permet d'instancier sa représentation. Ce processus peut être explicite ou implicite. La construction explicite est basée sur une collecte d'informations directement fournies par l'utilisateur via l'interface du système. La construction implicite, largement motivée par les travaux actuels dans le domaine, repose sur un procédé d'interface du contexte et préférence de l'utilisateur via son comportement lors de l'utilisation du système ou d'autres applications quotidiennes. Les informations exploitées pour la construction sont généralement issues :

- directement de l'utilisateur :
  - jugement explicite sur la pertinence des termes, documents,
  - définition de différents attributs : domaines d'intérêts, niveaux, langues, etc…
  - sélection de thèmes, sites favoris.

- indirectement de l'application
  - contenu des documents crées, consultés,
  - liens explorés,
  - durée de lecture des documents,
  - dernières pages visitées,
  - type d'application.

### *2.1.3. Evolution*

L'évolution désigne l'adaptation des structures représentatives des profils utilisateurs à la variation des besoins en information de ces derniers. Les principaux travaux mettant en exergue le processus d'évolution des profils utilisateurs ont porté notamment sur les systèmes de filtrage d'information [Allen 1990 ; Croft 1993] où la dimension temps est dominante pour un routage permanent des informations aux profils correspondants.

A notre connaissance, peu de travaux ont abordé le problème de l'évolution du modèle de l'utilisateur dans les systèmes personnalisés d'accès à l'information. Dans le cas de ces systèmes, l'évolution est d'avantage abordée comme un problème de représentation de la diversité des domaines d'intérêts de l'utilisateur [Pazzani et al 1996][Gowan 2003]. Cette

représentation est généralement basée sur un processus de classification dynamique qui tient compte des contextes courants issus de chaque session d'utilisation du système ou des applications courantes de l'utilisateur.

## 2.2. Sélection de l'information

Cette phase consiste à intégrer le profil ou le modèle utilisateur préalablement construit dans le processus de recherche d'information proprement dit. En ce sens, les informations contenues dans le profil sont exploitées pour identifier éventuellement le profil parmi ceux qui sont en cours de construction, réécrire puis exécuter la requête, enfin présenter les résultats de la recherche.

### *2.2.1. Identification du profil*

C'est une opération existante dans le cas des systèmes qui maintiennent un panel de profils canoniques ou dynamiques. Elle consiste à apparier la structure instanciée du profil avec ceux définis ou construits préalablement dans le système. L'appariement est généralement basé sur le calcul d'un score de probabilité de prédiction de profil [Horvitz et al 1998] ou similarité avec une classe de profils [Pazzani et al 1996][Gowan 2003].

### *2.2.2. Exécution des requêtes*

L'exécution d'une requête traduit la succession éventuelle des opérations de sélection d'information, reformulation et calcul d'un score de pertinence. La sélection personnalisée d'information est une pratique courante dans les méta-moteurs de recherche [Chau et al 2001][Glover et al 1999]. Leur principe est d'identifier, à travers le profil utilisateur, le type de requête (besoin général, actualité, scientifique …) puis l'adresser à un moteur de recherche approprié afin d'augmenter la précision de résultats. La reformulation de requête qui est une des premières techniques qui s'apparentent à la personnalisation, consiste à augmenter la requête avec des informations issues du profil avant de lancer le processus d'appariement [Liu et al 2002][Scime et al 2000]. La personnalisation peut également porter sur la définition de la fonction de calcul de la pertinence. Dans ce sens, [Fan et al 2004] ont proposé l'adaptation des paramètres de la fonction de pertinence au contexte de l'utilisateur, en utilisant les techniques de programmation génétique. [Jeh et Widom 2003] ont proposé une variante personnalisée de l'algorithme *PageRank* en l'occurrence PPV (*Personalized PageRank Vector*). Son principe fondamental est de privilégier les pages reliées aux pages préférées de l'utilisateur ou pages citées par ces dernières durant le processus de calcul des scores de sélection.

### *2.2.3. Présentation des résultats*

La présentation des résultats est la phase ultime du processus d'accès à l'information. Cette phase peut également considérer le profil de l'utilisateur en réordonnant les résultats fournis par le processus de sélection. En ce sens que l'ordre final des documents à présenter à l'utilisateur est une combinaison de l'ordre produit par le processus de sélection et celui donné par le contexte de l'utilisateur via un calcul de similarité [Vishnu 2004] ou jugements explicites de la pertinence [Gowan 2003].

# 3. Les prototypes des systèmes d'accès personnalisé

Les systèmes de recherche d'information personnalisés, sont classifiés en trois catégories [Shahabi et Chen 2003] [Lechani et boughanem 2005] ; les systèmes de recommandation, les systèmes d'accès contextuel et les méta-moteurs de recherche personnalisée. Dans chaque catégorie, plusieurs systèmes d'accès personnalisés ont été développés.

## 3.1. Les systèmes de recommandation

Les systèmes de recommandation se basent sur des informations liées à la recherche de l'utilisateur en cours, dans le but de l'aider, soit en lui sélectionnant les liens pertinents dans une page, soit en lui recommandant des documents similaires au document en cours de lecture. Dans cette catégorie de systèmes, le profil de l'utilisateur reflète son centre d'intérêts à court terme. Parmi les systèmes de recommandation : Letizia [Lieberman 1995], WebACE [Boley et al 1998], WebMat [Chen et Sycara 1998], WebWATCHER [Armstrong et al 1995], Syskill et Webert [Pazzani et al 1996], WAIR [Seo et Zhang 2000].

***Letizia :*** est un agent de recherche, qui assiste l'utilisateur en lui recommandant des liens à explorer en fonction de la page en cours d'être visitée par l'utilisateur. La représentation du profil dans cette approche n'est pas détaillée. Cependant, comme les documents sont représentés selon le modèle vectoriel (vecteurs de mots pondérés), nous supposons que même le profil est représenté ainsi. L'acquisition du profil se fait d'une manière implicite, à partir des pages explorées par l'utilisateur.

***WebACE :*** est un assistant à la navigation. Il construit le profil utilisateur à partir des documents qui intéressent l'utilisateur, en incluant des paramètres comme : le nombre de fois qu'un document est visité et le temps dépensé dans la lecture du document. WebACE sépare les intérêts de l'utilisateur en utilisant des clusters. Ainsi les clusters sont utilisés pour générer des requêtes et chercher des documents similaires.

***WebMate :*** est un autre agent qui assiste l'utilisateur dans sa recherche et sa navigation. Le système est composé d'un proxy qui observe les actions de l'utilisateur et son comportement. Initialement, le système exige à l'utilisateur de lui donner des exemples des documents pertinents, qu'il utilise pour construire les sous-catégories de l'intérêt de l'utilisateur. Chaque exemple, sera représenté par un vecteur TF-IDF de mots pondérés. Chaque vecteur, représentera un domaine d'intérêts particulier. Le nombre de domaines ne dépasse pas 10, dans le cas où le nombre dépasse 10, les deux vecteurs les plus similaires seront fusionnés.

***WebWatcher :*** est un assistant à la navigation, qui recommande des pages dans des sites particuliers, il agit comme un guide, dirigeant l'utilisateur aux documents intéressants sur ce site. Initialement l'utilisateur saisit les mots clés qui représentent son centre d'intérêts. Ensuite, le système observe les chemins explorés par l'utilisateur dans le site, et lui permet d'indiquer les pages qui l'intéressent, et d'indiquer si son besoin en information est satisfait. Ces informations seront utilisées pour des futures recommandations.

***Syskill et Webert :*** est un agent logiciel qui apprend des profils à plusieurs centres d'intérêts, dans le but de recommander des pages similaires à ces centres d'intérêts. Dans ce système, le profil est construit d'une manière explicite à partir du jugement de pertinence d'un index des pages recommandées. Le profil est représenté sous forme de classes séparées. Chaque classe représentant un centre d'intérêts, qui est modélisée à l'aide d'un vecteur booléen de mots-clés.

## 3.2. Les systèmes d'accès contextuel

Ce type de systèmes construit le profil utilisateur à partir des informations liées aux comportements de l'utilisateur, qui peuvent être : des tâches effectuées par l'utilisateur (ex traitement de texte), historique des interactions de l'utilisateur avec des applications Web. Le profil de l'utilisateur dans ce cas, reflète son centre d'intérêts à long terme. Parmi eux, on citera les systèmes Watson [Budzik et Hammond 2000], Web Personae [Gow 2003], et SIS [Dumais et al 2003].

***Watson :*** est un système sophistiqué qui surveille les documents édités et vus par l'utilisateur, afin d'impliquer automatiquement le contexte de l'utilisateur lors de ses recherches. Le profil de l'utilisateur n'a pas de structure bien spécifique, il est assimilé à une requête construite implicitement à partir du contenu des documents manipulés par l'utilisateur lors d'applications courantes. Les documents sont représentés par des mots pondérés, qui sont transformés sous une requête. La pondération des termes est basée sur l'emplacements des mots : les mots fréquents sont importants, les mots apparaissant avec des petites fontes sont moins importants, les mots apparaissant en marge des documents sont à ignorer, les mots figurant dans l'en-têtes et titres sont mieux pondérés que les autres. La requête est construite par les 20 mots les plus pondérés et dans l'ordre de leurs poids.

***Web Personae :*** le système web personae, combine le centre d'intérêts de l'utilisateur à long terme et celui de court terme pour personnaliser sa recherche. Il est composé de deux parties ; le constructeur et l'identificateur.

Le constructeur permet de collecter implicitement l'historique du comportement de l'utilisateur (fichier log, pages visités, liens explorés, …) et le contenu des documents associés à une liste d'URL fournie initialement par l'utilisateur *(bookmark, favoris...)*. A partir de ces documents, le profil est représenté sous forme d'une hiérarchie de classes, selon le modèle vectoriel, chaque classe décrit un domaine d'intérêts de l'utilisateur.

L'identificateur permet de découvrir le profil courant de l'utilisateur qui est exploité ultérieurement pour effectuer l'accès personnalisé proprement dit. Le principe est simple ; à partir des n dernières pages visités par l'utilisateur, un vecteur de termes pondérés selon la formulation TF*IDF est construit. L'identificateur calcule ensuite un score de similarité entre ce vecteur et chacun des vecteurs des classes construites préalablement pour la représentation des différents profils. Le profil qui a le score le plus élevé constitue le profil courant utilisé pour effectuer la personnalisation.

***SiS (Stuff I've Seen) :*** ce système permet de personnaliser des informations déjà consultées par l'utilisateur, selon différentes formes (Email, page Web, page intranet ….). Le système se déroule en deux phases ; la première consiste à créer un index unifié pour toutes les informations consultées indépendamment de la source et de la forme qui constituent le contexte de l'utilisateur ; la deuxième phase exploite ce contexte, pour effectuer un accès personnalisé à des sources d'information déjà utilisées. Le système SIS permet à l'utilisateur de définir des critères tels que la date, auteur et score, sur lesquels il se base pour présenter les résultats.

## 3.3. Les méta-moteurs de recherche personnalisée

Selon [Lawrence et Giles 1999], la quantité d'information sur le web couverte par un moteur, diminue avec l'accroissement de la taille du Web. Cela signifie que l'utilisation d'un seul moteur de recherche peut ne pas satisfaire la demande de l'utilisateur. A partir de là, les méta-moteurs de recherche sont apparus, qui accroissent la couverture de recherche en combinant les résultats issus de différents moteurs de recherche, comme MetaCrawler et DogPile. L'inconvénient de ces méta-moteurs de recherche est qu'ils retournent à l'utilisateur une quantité considérable de résultats. C'est pourquoi, les méta-moteurs personnalisés sont apparus. Leur principe est d'intégrer le contexte et le centre d'intérêts de l'utilisateur, pour lui permettre de filtrer et ne retourner que les informations pertinentes, selon ses besoins. Nous pouvons citer dans ce cas le méta-moteur *Inquirus*.

***Inquirus :*** est un système de recherche, qui interroge plusieurs moteurs de recherche pour une requête utilisateur. Dans *Inquirus*, l'utilisateur peut exprimer ces préférences, qui seront utilisées dans le système pour réécrire sa requête et identifier le moteur de recherche à exploiter. Ainsi, le profil est constitué de la requête et d'une catégorie de préférences et intérêt explicitement choisies par l'utilisateur. Cette catégorie permet au système d'associer un moteur adéquat, et de retourner des résultats adéquats

# 4. Evaluation des systèmes d'accès personnalisé à l'information

Les méthodes d'évaluation largement adoptées en recherche d'information, sont souvent basées sur une évaluation d'avantage quantitative que qualitative. En effet, les résultats obtenus sont issus de la comparaison de mesures et de métriques en termes de rappel et précision, sur les réponses fournies par le système relativement à celles issues des réponses attendues qui constituent le référentiel. Ce type d'évaluation, est orienté vers une approche comparative de plusieurs systèmes reposant sur le principe d'évaluation des collections de tests. Bien qu'adopté par des campagnes d'évaluation de référence en recherche d'information, tel que TREC, il est cependant contesté [Balpe et l996] notamment en raison de la non considération ni du contexte dans lequel se fait la recherche, ni de la perception de la pertinence des utilisateurs dans ce même contexte. L'introduction de la dimension utilisateur dans le processus d'accès à l'information, accentue d'avantage la difficulté d'évaluation. En effet, en raison du caractère subjectif des utilisateurs d'une part et du caractère dynamique ou adaptatif du système, d'autre part, il est difficile de fournir des valeurs absolues aux métriques. On cite dans ce qui suit les principaux problèmes liés à l'évaluation des systèmes personnalisés puis présentons quelques recommandations qui augmentent la fiabilité des résultats qui en sont issus [Lechani et Boughanem 2005].

## 4.1. Problèmes de l'évaluation

La rationalité des résultats issus d'un scénario d'évaluation d'un système personnalisé est compromise pour les principales raisons suivantes [Chin 2001]:

- Si l'évaluation des variables indépendantes est effectuée par différents utilisateurs, alors des différences personnelles exogènes à l'expérimentation, telles que l'intelligence, la capacité de raisonnement, l'expérience … etc, ont un impact sur les valeurs des autres variables.

- Si le même utilisateur est impliqué dans différents scénarios d'évaluation, alors son expérience passée avec le système peut influer sur sa perception de la pertinence.

- Les conditions de déroulement des expérimentations ont un impact sur les résultats comme : la vitesse des machines non suffisantes, les interfaces peu ergonomes, lieu inapproprié,…etc.

- La validation interne est difficile à mettre en œuvre. En effet, en raison des problèmes évoqués ci-dessus, il n'est pas aisé de séparer l'effet intrinsèque des variables pertinentes du système indépendamment des facteurs exogènes. La validation externe est dès lors difficile également. Elle l'est d'autant plus que la généralisation des résultats obtenus lors de la validation interne doit considérer une combinaison des divers types d'utilisateurs, des variables d'expérimentation et des situations d'utilisation.

### 4.2. Recommandation

Afin d'éviter des erreurs de mesures lors d'évaluation d'un système de recherche personnalisé, plusieurs recommandations ont été élaborées [Chin 2001 ; Lechani et Boughanem 2005] :

- définir un nombre suffisant de groupes d'utilisateurs avec des effectifs adéquats,
- isoler au mieux les utilisateurs,
- s'assurer de l'ergonomie des applications,
- préparer un canevas unique qui décrit le protocole d'expérimentation et l'adresser à l'ensemble des utilisateurs,
- effectuer des expérimentations pilotes avant de passer aux expérimentations effectives,
- les utilisateurs ne doivent pas être informés des facteurs à évaluer dans le système,
- les variables exogènes (âge, expérience, aptitudes, ….) doivent être identifiées explicitement et leur influence mesurée pour être prise en compte dans le processus global d'évaluation.

## 5. Discussion

Notre synthèse sur l'accès personnalisé à l'information fait ressortir plusieurs questions fondamentales, qui sont :

- ***Quelles informations faut-il pour représenter l'utilisateur ? :*** La prise en compte des dimensions liées au contexte de l'utilisateur intégrant l'application en cours, fait ressortir de nombreuses sources d'information : l'utilisateur lui-même, les documents qu'il consulte, les liens qu'il explore, les pages favorites qu'il sélectionne, les applications qu'il utilise, les pages qu'il visite etc.

- ***Quel modèle faut-il pour l'utilisateur ? :*** Une question préliminaire à celle-ci est sans doute : est ce que l'utilisateur est explicitement modélisé ou assimilé aux informations qui le décrivent ? c'est plutôt la seconde alternative qui est largement adoptée actuellement. Il y a peu voire pas de modèles globaux spécifiques à l'utilisateur. La modélisation se résume à la structuration des paquets d'information qui décrivent l'utilisateur : classes de termes, réseau connexionniste, ou bayésien de préférences, ontologies, vecteur de documents préférés ou jugés pertinents. La description d'un modèle unificateur de l'utilisateur n'est pas le souci fondamental tant les informations descriptives sont de source, natures et degrés de fiabilité diverse.

- ***Comment adapter le cycle de vie du processus d'accès ? :*** L'adaptation est effectuée principalement à l'un des différents niveaux : sélection des sources d'information, reformulation de requête, sélection de l'information et filtrage des résultats. La sélection personnalisée des sources d'information est opérée par les méta-moteurs de recherche. La reformulation de requête a pour objectif d'introduire dans la structure de la requête les termes issus du profil de l'utilisateur ; c'est la technique la plus largement répandue. La sélection adaptée de l'information, promue par de récents travaux, évoque la contextualisation de la fonction de pertinence en définissant des paramètres issus du profil de l'utilisateur. Enfin, le filtrage des résultats traduit la prise en compte des préférences de l'utilisateur à l'étape précédant la présentation des résultats et suivant leur sélection. L'adaptation consiste généralement à réordonnancer les résultats en tenant compte de critères descriptifs de l'utilisateur.

# 6. Conclusion

Nous avons essayé de présenter dans ce chapitre une étude des approches et techniques de l'accès personnalisé à l'information. Après avoir dégagé la problématique qui y était attachée, nous avons décrit des travaux s'attachant à comprendre les raisons qui font que la personnalisation permet ou non d'améliorer les performances d'une recherche d'information. Plusieurs systèmes d'accès personnalisé ont été développés et publiés dans la littérature. Tous ces systèmes s'appuient sur l'idée que la modélisation de l'utilisateur et son intégration dans le processus de recherche augmente sa performance, malgré une difficulté qui y était recensée lors du processus d'évaluation.

Dans le chapitre précédent (chapitre 1) nous avons montré les problématiques de la recherche d'information lors de l'accès à de multiples sources d'information. Nous nous sommes intéressés aux deux problématiques fondamentales de la RID, qui sont : (1) la sélection de sources pertinentes à l'utilisateur, (2) la fusion des résultats retournés par les sources interrogées.

Notre objectif dans cette de thèse est de personnaliser ces deux processus de la RID (sélection de sources et fusion des résultats), en d'autres termes d'intégrer le modèle

(profil) utilisateur dans un accès distribué à l'information et plus précisément dans les processus de sélection de sources et de fusion des résultats retournés par ces sources.

La partie suivante sera consacrée à la présentation de nos contributions permettant d'intégrer l'utilisateur via son profil dans les processus de sélection de sources et de fusions des résultats retournés par ces sources. Les questions fondamentales qui seront abordées et sur lesquelles se focalise thèse sont alors les suivantes :

- Quelles sont les informations qui représentent au mieux le profil utilisateur et le profil des sources?

- Quelle est la manière la plus fiable pour acquérir et mettre à jour ces informations ?

- Comment modéliser ces profils ?

- Comment intégrer le profil utilisateur dans les processus de sélection et de fusion ?

- Comment évaluer la performance de notre approche ?

# Deuxième partie

# Contributions aux systèmes de recherche d'information distribuée

# Chapitre 4

# Accès personnalisé à de multiples sources d'information

## 1. Introduction

Nous avons présenté dans les chapitres précédents, les problématiques rencontrées dans la recherche d'information en général (RI) et particulièrement dans la recherche d'information distribuée (RID). Nous nous sommes intéressés principalement aux problèmes de sélection de sources et de fusion des résultats retournés par les sources, plus précisément à l'intégration du profil de l'utilisateur dans les processus de sélection des sources et de fusion des résultats des sources sélectionnées. Dans ce chapitre, nous proposons notre approche d'acquisition des données sur l'utilisateur et sur les sources qui serviront à personnaliser le processus de sélection de sources et le processus de fusion des résultats.

Ce chapitre est organisé de la manière suivante ; Dans la section 2, nous présentons un rappel des deux problèmes rencontrés dans la RID. Nous présentons dans la section 3 et 4 les objectifs visés et les hypothèses posées. Dans la section 5, nous présentons notre contribution pour personnaliser le processus de sélection de sources et le processus de fusion des résultats des sources sélectionnées. La section 6 présente l'expérimentation et l'évaluation de l'approche. La section 7 conclut le chapitre.

Dans la suite de notre travail, nous utilisons le terme *source* d'information, pour désigner une collection d'information ou un serveur d'information.

## 2. Problématique de la RID

Les deux problèmes de la RID dont nous nous intéressons dans cette thèse sont ; (1) comment sélectionner les sources d'information les plus pertinentes pour l'utilisateur ? ; (2) comment fusionner les listes des résultats retournées par les sources interrogées ; pour construire une liste finale des résultats plus pertinente pour l'utilisateur ?.

Nous distinguons deux types concernant la signification du mot *pertinence* d'un document :

1. ***La pertinence système :*** la pertinence système est la pertinence relative au degré de similarité (score) entre la requête de l'utilisateur et le document calculé par le système de RI.

2. ***La pertinence utilisateur :*** la pertinence utilisateur est la pertinence d'un document relative au jugement de l'utilisateur lui-même.

# 3. Objectifs

Dès le début de nos travaux de recherche, nous nous sommes fixés le but principal suivant : *concevoir une méthode permettant de mieux satisfaire la requête utilisateur en information, tout en réduisant le coût de la recherche dans un système de recherche d'information distribuée (SIRD).*

L'étude menée dans le domaine de la RI, en particulier dans la RI personnalisée (RIP), a montré que généralement la requête de l'utilisateur n'est pas suffisante pour exprimer le besoin de l'utilisateur en information. Cela induit alors que la pertinence système calculée entre le document et la requête de l'utilisateur ne satisfait pas toujours le besoin de l'utilisateur. Pour mieux satisfaire le besoin de l'utilisateur en information, il faut doter les systèmes de RI par des techniques (stratégies) permettant d'acquérir des informations sur l'utilisateur, qui seront utilisées avec sa requête dans son processus de recherche d'information.

L'étude des travaux antérieurs dans le domaine de la RID nous a montré l'absence de stratégies permettant d'acquérir des informations sur l'utilisateur, pour mieux comprendre sa requête dans un SRID. Nous nous sommes alors fixés comme objectif de concevoir un SRID personnalisé, qui se résume par les points suivants :

1. Définir une stratégie d'acquisition des informations sur l'utilisateur, qui reflètent mieux son besoin en information.

2. Intégrer ces informations acquises de l'utilisateur dans le processus de sélection de sources d'information.

3. Intégrer ces informations acquises de l'utilisateur dans le processus de fusion des résultats retournés par les sources interrogées ;

A partir de là, nous proposons les hypothèses suivantes :

# 4. Hypothèses

**Hypothèse (1)** Une bonne sélection consiste à sélectionner les sources qui retournent en premier lieu des documents pertinents.

**Hypothèse (2)** Les premiers documents retournés par un SRI sont les plus représentatifs (importants) des résultats d'une source d'information.

**Hypothèse (3)** La requête de l'utilisateur seule n'est pas suffisante pour décrire son besoin en information.

**Hypothèse (4)** La pertinence système ne reflète pas toujours la pertinence utilisateur.

Pour appuyer notre première hypothèse, disons que sélectionner une source contenant des documents pertinents, mais classant ces derniers relativement bas, n'est pas très intéressant. En effet, une étude menée par [Wolfram et al 2001] [Spink et al 2001], a montré que les utilisateurs ne consultent qu'un nombre très restreint de pages de résultats.

Pour appuyer notre seconde hypothèse, disons qu'un moteur de recherche peut donner des résultats très satisfaisants ou complètement inadaptés, selon la requête posée. Ainsi, le meilleur moyen de vérifier la performance d'un système pour une requête donnée est d'analyser les premiers documents qu'il retourne [Abbaci 2003].

Notre troisième hypothèse, s'appuie sur l'étude menée par [Jansen et al 1998], qui montrent que les utilisateurs n'emploient habituellement que quelques mots (moins de 5) pour décrire le document recherché ce qui peut donner lieu à des ambiguïtés.

Pour appuyer notre quatrième hypothèse, disons que la pertinence d'un document par rapport à une requête est évaluée par ces services (*services d'un SRI*) pour un utilisateur moyen, sans prendre en compte les besoins de l'utilisateur effectif, ni son niveau d'expertise réel.

# 5. Approche de personnalisation des processus de sélection et de fusion

L'objectif de notre de thèse est d'intégrer les informations qui décrivent les besoins de l'utilisateur en information dans les processus de sélection de sources et de fusion des résultats des sources pour une requête de l'utilisateur. Pour ce faire nous avons suivi les étapes suivantes :

1. Définir les informations qui décrivent les besoins de l'utilisateur en information. Ces informations seront appelées *profil* de l'utilisateur.

2. Définir la manière d'acquérir ce profil de l'utilisateur.

3. Définir une stratégie pour intégrer le profil de l'utilisateur dans le processus de sélection de sources d'information.

4. Définir une stratégie pour intégrer le profil de l'utilisateur dans le processus de fusion des résultats des sources d'information.

Pour bien illustrer l'approche proposée, nous l'avons organisé en deux phases.

- La première phase appelée phase de construction et évolution du profil de l'utilisateur. Cette phase présente les deux premières étapes décrites ci-dessus (1 et 2).

- La deuxième phase appelée phase d'accès personnalisé à des sources d'informations distribuées. Cette phase présente les deux dernières étapes décrites ci-dessus (3 et 4).

L'architecture générale de notre démarche peut être représentée schématiquement par la figure suivante (figure 4.1):

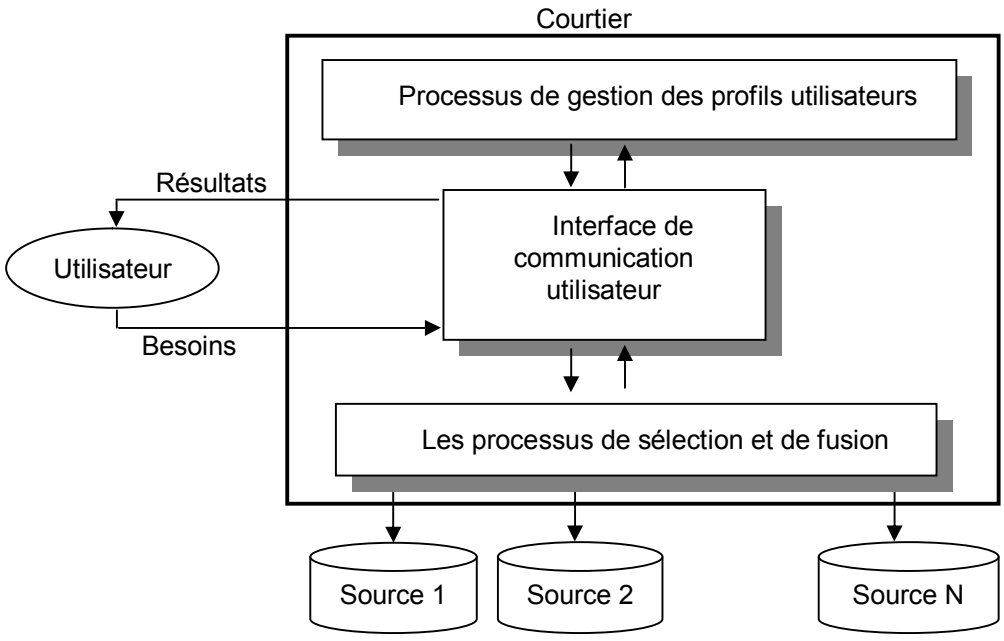


**Figure 4.1.** Architecture générale de l'approche proposée.

Dans la figure 4.1, le processus de gestion des profils utilisateurs assure la phase de construction et évolution du profil de l'utilisateur. Les processus de sélection de sources et de fusion des résultats des sources assurent la phase d'accès personnalisés à des sources d'information distribuées.

Avant de détailler les techniques qui sont derrière les deux phases de cette démarche, nous présentons d'abord une description fonctionnelle qui illustre le processus général d'accès personnalisé à des sources d'information distribuées que nous proposons.

Les grandes lignes de ce processus [Kechid et Drias 2006a] [Kechid et al 2006] qui représente un processus itératif sont les suivantes :

- un nouvel utilisateur doit s'inscrire d'abord et initialiser son profil auprès de l'interface de communication utilisateur,

- un utilisateur déjà inscrit doit ouvrir son compte, pour avoir la possibilité d'introduire sa requête auprès de l'interface de communication utilisateur,

- l'interface de communication utilisateur soumet la requête de l'utilisateur qu'il a introduit, au processus de gestion des profils utilisateurs,

- ce dernier, extrait le profil de l'utilisateur adéquat et le transmet à l'interface de communication utilisateur,

- l'interface de communication utilisateur soumet la requête accompagnée avec le profil de l'utilisateur adéquat aux processus de sélection et de fusion,

- le processus de sélection reformule la requête de l'utilisateur en utilisant son profil et interroge les sources d'information par la requête reformulée,

- à la réception des listes des résultats des sources, le processus de sélection récupère les $k$ premiers documents de chaque liste, dont il extrait une liste $L_s$ des termes pondérés les plus pertinents pour chaque source $s$,

- le processus de sélection calcule un score pour chaque source en utilisant sa liste $L_s$, la requête de l'utilisateur et d'autres informations sur l'historique des sources récupéré du profil de l'utilisateur,

- à partir des scores des sources calculés, le processus de sélection sélectionne les sources dont leurs scores dépassent ou égale la moyenne des scores,

- le processus de fusion calcule un score pour chaque document de chaque liste des résultats des sources sélectionnées, en utilisant la requête de l'utilisateur et son profil,

- le processus de fusion, fusionne en une seule liste les documents des listes et les trie par ordre croissant de leurs scores, qui sera renvoyée à l'utilisateur via l'interface de communication utilisateur,

- l'interface de communication utilisateur observe et récupère l'historique de recherche de l'utilisateur dans sa consultation aux résultats (les documents pertinents pour l'utilisateur), et les transmet au processus de gestion des profils utilisateurs,

- le processus de gestion des profils utilisateur met à jour le profil de l'utilisateur à partir de cet historique de recherche,

## Remarque

Pour les prochaines requêtes, l'utilisateur aura la possibilité de choisir à chaque fois si la sélection de sources sera exécutée ou non. Dans le cas où il choisit d'exécuter la sélection de sources, tout le processus décrit ci-dessus sera ré-exécuté. Dans le cas où il ne choisit pas d'exécuter la sélection de sources, les sources sélectionnées précédemment seront interrogées, et leurs résultats seront fusionnés.

Ce choix a pour but de :

1- Ne pas refaire la sélection de sources pour gagner du temps de calcul si la requête suivante est du même domaine (similaire) que la précédente.

2- Refaire la sélection de sources si la requête suivante n'est pas du même domaine (n'est pas similaire) que la précédente, pour vérifier la possibilité d'existence d'autres sources qui peuvent répondre aussi à la nouvelle requête.

Nous détaillons dans ce qui suit, les techniques qui sont derrière ces processus.

## 5.1. Phase de construction et évolution du profil de l'utilisateur

Dans cette section nous présentons notre façon à définir, instancier, et mettre à jour le profil de l'utilisateur.

### *5.1.1. Définition du profil de l'utilisateur*

Le profil utilisateur dans notre approche est utilisé pour ; (1) Reformuler sa requête pour mieux comprendre ses besoins en informations. (2) Sélectionner les sources pertinentes à interroger. (3) Fusionner les résultats des sources. Pour cela le profil utilisateur doit contenir les données suivantes :

- les données personnelles de l'utilisateur, pour l'identifier,
- l'historique des recherches de l'utilisateur, qui sera utilisé pour extraire les termes décrivant le centre d'intérêts de l'utilisateur,
- l'historique sur la sélection des sources, qui nous donne le nombre de fois qu'une source est sélectionnée pour l'utilisateur.

Nous pouvons schématiser le profil utilisateur par la figure suivante :

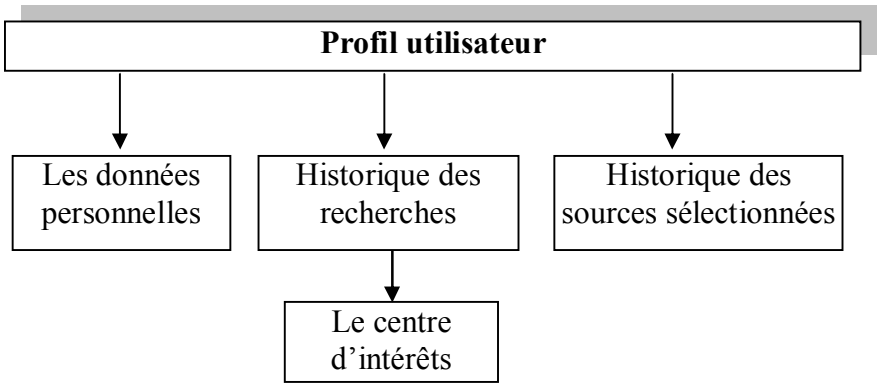


**Figure 4.2.** Profil utilisateur

#### *5.1.1.1. Les données personnelles*

Les données personnelles de l'utilisateur sont utilisées pour définir son identité. Nous pouvons utiliser, le code, le mot de passe, le nom, l'age, l'adresse et le domaine.

#### *5.1.1.2. Historique des sources sélectionnées*

L'historique des listes des sources sélectionnées nous permet de savoir le nombre de fois qu'une source est sélectionnée pour l'utilisateur. Ce nombre, peut être utilisé comme indice de pertinence pour la source. Nous notons ce nombre $NbrSelSouce_s^{(u)}$, qui signifie le nombre de fois qu'une source $s$ est sélectionnée pour l'utilisateur $u$.

#### *5.1.1.3. Historique des recherches*

L'historique des recherches de l'utilisateur contient les documents consultés par l'utilisateur et jugés pertinents pour l'utilisateur. Cet ensemble de documents est utilisé pour en extraire le centre d'intérêts de l'utilisateur, et le nombre de documents consultés par l'utilisateur de chaque source.

#### *5.1.1.4. Le centre d'intérêts de l'utilisateur*

Le centre d'intérêts de l'utilisateur est un ensemble de mot clés (termes) pondérés selon la méthode d'indexation *tf*idf* du modèle vectoriel [Salton et Allan 1994]. L'ensemble des documents de l'historique (ensemble des $k$ premiers documents retournés par chaque source) forment la collection du calcul. Chaque terme $t_i$ indexé (pondéré) suivant la formule (les mots vides ne sont pas pris en considération) :

$$Poids_{id} = \frac{freq_{id}}{\sqrt{\sum_{j=1}^{N} freq_{ij} * \log_2^2 (\frac{N}{n_i})}} * \log_2 \frac{N}{n_i}$$

Avec, $freq_{id}$ est la fréquence d'apparition du terme $t_i$ dans le document $d$ ; $freq_{ij}$ est la fréquence d'apparition du terme $t_i$ dans le document $j$ ; $N$ est le nombre de documents de l'historique; $n_i$ est le nombre de documents de l'historique contenant le terme $t_i$.

Le nombre de documents de chaque source que l'utilisateur a consulté, peut être utilisé comme indice de pertinence pour la source. Nous notons ce nombre $NbrDocPertSource_s^{(u)}$, qui représente le nombre de documents pertinents de la source $s$ pour l'utilisateur $u$ dans son historique des recherches.

### *5.1.2. Système d'apprentissage et d'acquisition du profil utilisateur*

#### *5.1.2.1. Instanciation du profil utilisateur*

Les données personnelles de l'utilisateur sont introduites par l'utilisateur. Cependant, l'historique des sources sélectionnées et l'historique des recherches de l'utilisateur sont initialement vides.

#### *5.1.2.2. Acquisition du profil utilisateur*

- ***Les données personnelles de l'utilisateur*** : sont fixées et ne peuvent pas être modifiées, sauf le mot de passe qui peut être changé par l'utilisateur quand il veut.

- ***L'historique des sources sélectionnées :*** est mis à jour par le courtier. Après chaque phase de sélection de sources, le courtier incrémente le nombre $NbrSelSources_s^{(u)}$ associé à chaque source *s* sélectionnée si elle existe dans la liste, sinon il l'ajoute à la liste, et initialise son $NbrSelSource_s^{(u)}$ à 1.

- ***L'historique des recherches de l'utilisateur :*** est mis à jour par le courtier. Durant la phase de recherche, le courtier observe l'utilisateur dans ses consultations aux résultats de la recherche, et sauvegarde les documents pertinents pour l'utilisateur, en associant pour chaque document la source d'où il provient.

A partir de cet historique des recherches, il déduit la valeur de $NbrDocPertSource_s^{(u)}$ (nombre de documents pertinents de la source *s* pour l'utilisateur *u* ) pour chaque source.

- ***Le centre d'intérêts de l'utilisateur :*** à partir de cet historique des recherches, il extrait les termes pertinents pondérés selon la méthode d'indexation *tf*idf* du modèle vectoriel décrite précédemment. Ces termes seront ajoutés au centre d'intérêts de l'utilisateur (les mots vides ne sont pas pris en considération).

## 5.2. Phase d'accès personnalisé à des sources d'informations distribuées

Dans cette phase nous décrivons les étapes (processus) d'évaluation de la requête de l'utilisateur, permettant de personnaliser son processus de recherche d'information dans un environnement distribué. Cette phase comprend trois processus :

(1) Processus de reformulation de requête.
(2) Processus de sélection de sources d'information.
(3) Processus de fusion des résultats des sources sélectionnées.

### *5.2.1. Processus de reformulation de la requête*

Pour la reformulation de la requête, nous utilisons le centre d'intérêts de l'utilisateur, qui est un ensemble (vecteur) de termes pondérés calculé dans le profil de l'utilisateur. Initialement cet ensemble est vide, mais il est alimenté durant le processus itératif de recherche.

Pour ce faire nous extrayons les X termes ayant les poids les plus élevés dans l'ensemble de termes du centre d'intérêts de l'utilisateur. Nous notons cet ensemble (vecteur) des X termes $P_u$. La requête utilisateur est reformulée en lui ajoutant les termes du vecteur $P_u$. La nouvelle requête est pondérée par la formule suivante inspirée de [Rocchio 1971] :

$$q^{nouvelle} = a.q^{ancienne} + \frac{b}{|P_u|}\sum_{t_i \in P_u} t_i$$

Avec $q^{nouvelle}$ est la requête reformulée, $q^{ancienne}$ est l'ancienne requête formulée par l'utilisateur, $a$ et $b$ sont des constantes, $a, b \in [0, 1]$.

### *5.2.2. Processus de sélection de sources*

La requête une fois reformulée, est envoyée à toutes les sources utilisées. Chaque source renvoie au courtier une liste de documents pertinents pour cette requête. Le courtier analyse les $k$ premiers documents de la liste de chaque source.

Le courtier associe à chaque source $s$ un score noté $ScoreSource_s^{(u)}$ pour l'utilisateur $u$. Sur la base de ce score, le courtier trie et sélectionne les sources pertinentes pour cet utilisateur.

Pour le calcul du score de chaque source $s$ ($ScoreSource_s^{(u)}$), le courtier combine deux mesures :

(1) le degré de similarité de la source à la requête reformulée de l'utilisateur noté $SimSource_s^{(u)}$,

(2) un poids mesurant l'historique de sélection des sources, noté $PoidsHistSource_s^{(u)}$.

Ces deux mesures sont calculées comme suit :

- ***Calcul de la similarité (score) entre la source et la requête utilisateur :*** le degré de similarité de la source à la requête utilisateur $SimSource_s^{(u)}$ est calculé en utilisant la formule de *cosinus* du modèle vectoriel [Salton 1971]. Une source est considérée comme un document gigantesque, représenté par les $k$ premiers documents qu'elle a retournés. L'ensemble de ces $k$ premiers documents retournés par les sources représente la collection ou le corpus dans les calculs.

$$SimSource_s^{(u)} = \frac{\sum_{i=1}^{T} t_i * q_i}{\left(\sum_{i=1}^{T} t_i^2\right)^{1/2}\left(\sum_{i=1}^{T} q_i^2\right)^{1/2}}$$

Avec, *s* est la source ; *u* est l'utilisateur représenté par sa requête *q* ; $t_i$ est le poids du $i^{ème}$ terme dans la source (ensemble des *k* premiers documents retournés par la source); $q_i$ : est le poids du $i^{ème}$ terme dans la requête ; *T* est le nombre de termes utilisés de la source. Le contenu d'une source est représenté par l'ensemble de ses *k* premiers documents retournés.

- ***Calcul du poids de l'historique de sélection de la source :*** le poids de l'historique de sélection de la source $PoidsHistSource_s^{(u)}$ indique une estimation d'un taux de sélection d'une source s par rapport aux autres sources. Ce taux est calculé par la combinaison des deux valeurs $NbrSelSource_s^{(u)}$, $NbrDocPertSource_s^{(u)}$, définies ci-dessus.

$$PoidsHistSource_s^{(u)} = \frac{NbrDocPertSource_s^{(u)} * NbrSelSource_s^{(u)}}{NbrDocPert * \sum_{s=1}^{N} NbrSelSource_s^{(u)}}$$

Avec *N* est le nombre de sources utilisées, et *NbrDocPert* est le nombre de documents pertinents de toutes les sources dans l'historique des recherches.

Le score final de la source *s* $ScoreSource_s^{(u)}$ est calculé par combinaison des deux mesures $SimSource_s^{(u)}$ et $poidsHistSource_s^{(u)}$ suivant notre formule :

$$ScoreSource_s^{(u)} = \alpha . SimSource_s^{(u)} + (1 - \alpha) . PoidsHistSource_s^{(u)}$$

$$\alpha \in [0, 1].$$

Sur la base de ce score final $ScoreSource_s^{(u)}$ de chaque source *s*, les sources sont triées par ordre décroissant. Les sources ayant le score qui dépasse ou égale la moyenne des scores sont sélectionnées.

### *5.2.3. Processus de fusion des résultats des sources*

Ce processus consiste à ordonner et fusionner l'ensemble des documents retournés par les sources sélectionnées dans une seule liste. Cette liste sera retournée à l'utilisateur.

L'ordonnancement des documents se fait sur la base du score final de chaque document *d* de chaque source *s* sélectionnée. Nous notons $ScoreDocFinal_{ds}^{(u)}$, le score final d'un document *d* de la source *s* de l'utilisateur *u.* Ce score final est calculé par combinaison des deux mesures :

(1) ***Le score de pertinence de la source $ScoreSource_s^{(u)}$ :*** ce score est calculé dans l'étape de sélection de sources.

(2) ***Le degré de pertinence du document d à la requête reformulée, $Score_{d,s}^{(u)}$ :*** ce score est calculé en utilisant la formule de cosinus du modèle vectoriel décrite précédemment. L'ensemble des documents retournés par les sources sélectionnées forme la collection de calcul.

Le score final d'un document *d* est calculé suivant notre formule suivante :

$$ScoreDocFinal_{d,s}^{(u)} = \alpha.Score_{d,s}^{(u)} + (1-\alpha).ScoreSource_s^{(u)}$$

$$\alpha \in [0, 1].$$

Sur la base de ce score $ScoreDocFinal_{ds}^{(u)}$ les documents sont fusionnés et triés par ordre décroissant dans une seule liste, qui sera retournée à l'utilisateur.

# 6. Expérimentation

Afin d'évaluer notre approche, nous avons développé un prototype (voir partie 3). Dans ce prototype nous avons considéré les 08 moteurs de recherche suivants : GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chaque moteur de recherche représente une source d'information.

Pour ne pas surcharger ce chapitre, le prototype est présenté dans la partie 3.

Pour expérimenter l'intérêt de notre approche, nous avons testé le prototype développé sur 100 requêtes différentes. Nous avons ensuite évalué notre approche en considérant les quatre cas (scénarios) suivants :

- **Cas 1 :** Pas de sélection de sources et pas d'utilisation de profil. Utilisation de tous les moteurs (sources) sans prendre en compte le profil utilisateur.
- **Cas 2 :** Sélection des sources sans l'utilisation du profil utilisateur, la sélection ici est faite sur la base de la requête utilisateur seulement.
- **Cas 3 :** Prendre en compte le profil utilisateur en utilisant toutes les sources.
- **Cas 4 :** Sélection des sources en utilisant le profil de l'utilisateur. Ce cas représente la contribution proposée dans cette approche.

Dans nos calculs, nous avons utilisé les paramètres suivants :

- Nombre de termes qui définissent le centre d'intérêts de l'utilisateur = 10.
- Nombre de termes extraits du profil, utilisés dans la reformulation de requête, $P_u$ = 5.
- Nombre de documents consultés et traités dans chaque résultat de recherche = 20.

Pour évaluer la pertinence des résultats, chaque utilisateur juge les 20 premiers documents retournés pour 5 sessions de recherche. Une valeur de précision est calculée pour chaque requête selon la formule habituelle suivante :

$$précision = \frac{nombre\ \ de\ \ documents\ \ pertinents\ \ sélectionés}{nombre\ \ de\ \ documents\ \ sélectionnés}$$

Une précision moyenne est calculée pour l'ensemble des requêtes pour chaque cas. Le tableau suivant présente les précisions moyennes trouvées pour chaque cas.

| | **Cas 1** | **Cas 2** | **Cas 3** | **Cas 4** |
|---|---|---|---|---|
| **Précision (%)** | 21.78 | 24.53 | 25.34 | 26.29 |

**Tableau 4.1.** Précisions moyennes de nos cas.

Ces différents cas ont montré que le 4ème cas donne des résultats plus efficaces sur le plan pertinence.

Ces résultats préliminaires nous donnent quelques indicateurs sur l'intérêt de mettre l'utilisateur au cœur du processus de la recherche d'information distribuée. D'autres expériences, plus poussées, sont nécessaires pour tirer de meilleures conclusions sur cette approche.

## 6.1. Discussion

Notre but principal est d'intégrer le centre d'intérêts de l'utilisateur dans les processus de sélection de sources et de fusion des résultats des sources dans un SRID.

L'expérimentation et l'évaluation de l'approche proposée, ont montré que l'intégration du centre d'intérêts de l'utilisateur dans son accès à des sources d'information distribuées améliore la pertinence des résultats. Cependant, nous avons constaté des insuffisances

importantes de l'approche, plus précisément dans la description du profil de l'utilisateur. Nous pouvons résumer ces insuffisances par les deux points suivants :

(1) Un utilisateur peut avoir plusieurs centres d'intérêts différents. Pour cela, il faut :
- définir une stratégie d'acquisition des différents centres d'intérêts d'un utilisateur,

- définir une méthode (technique) permettant d'extraire (associer) à chaque requête de l'utilisateur, le centre d'intérêts correspondant, qui sera utilisé dans son processus d'accès personnalisé à des sources d'information distribuées.

(2) L'utilisation du centre d'intérêts seul n'est pas suffisante pour décrire tout le besoin de l'utilisateur en information. Car, chaque utilisateur peut avoir des préférences différentes sur des critères (nature). des documents à collecter (autres que leurs contenus).

Pour cela, la suite de notre travail aura comme objectif de remédier à ces insuffisances rencontrées. Nous devons alors répondre aux questions suivantes :

- comment définir une stratégie d'acquisition des différents centres d'intérêts d'un utilisateur ?

- comment définir une méthode permettant d'extraire (associer) de chaque requête de l'utilisateur, le centre d'intérêts correspondant, qui sera utilisé dans son processus d'accès personnalisé à des sources d'information distribuées ?

- comment définir les critères des documents qui ont de l'importance pour l'utilisateur ?

- comment définir les préférences de l'utilisateur sur ces critères des documents ?

- comment intégrer les préférences de l'utilisateur sur les critères des documents dans le processus de sélection de sources d'information ?

- comment intégrer les préférences de l'utilisateur sur les critères des documents dans le processus de fusion des résultats des sources d'information sélectionnées ?

# 7. Conclusion

Dans ce chapitre nous avons présenté une approche concernant l'intégration du profil utilisateur dans les processus de sélection de sources et de fusion des résultats.

L'approche a été évaluée en utilisant les 08 moteurs de recherches suivants : GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chaque moteur de recherche représente une source d'information. L'approche est expérimentée sur 100 requêtes, par différentes façons :

- utilisation de toutes les sources (pas de sélection de sources) sans prendre en compte le profil utilisateur ;
- la sélection des sources sans l'utilisation du profil utilisateur, la sélection ici est faite sur la base de la requête utilisateur seulement ;
- prendre en compte le profil utilisateur en utilisant toutes les sources ;
- la sélection des sources sur la base du profil utilisateur, qui est la contribution proposée par notre approche.

Pour chaque cas nous avons calculé la précision moyenne des résultats. Les résultats obtenus ont montré que l'intégration du profil utilisateur dans la recherche d'information distribuée, améliore la pertinence des résultats de recherche.

Cependant, l'approche présente des insuffisances à savoir ; (1) un utilisateur peut avoir plusieurs centres d'intérêts différents, pas un seul seulement. (2) l'utilisation du centre d'intérêts seul pour un utilisateur n'est pas suffisante pour décrire ses besoins en information, un utilisateur peut avoir des préférences sur la nature (critères) des documents qu'il veut consulter.

Dans le chapitre suivant, nous présentons une approche d'intégration des préférences de l'utilisateur sur les critères des documents dans son accès à de multiples sources d'information. L'approche présente des solutions pour les insuffisances rencontrées dans l'approche précédente

# Chapitre 5

# Intégration des préférences utilisateur dans un SRID

## 1. Introduction

Dans ce chapitre nous présentons une approche [Kechid et Drias 2006b] [Kechid et Drias 2009] pour remédier aux insuffisances rencontrées dans l'approche précédente. Les solutions que nous avons envisagées s'articulent autour des points suivants :

1- Définir un profil utilisateur permettant de ; (1) décrire les préférences de l'utilisateur sur des critères des documents ; (2) décrire les différents centres d'intérêts d'un utilisateur.

2- Définir un profil pour chaque source d'information. Ce profil constituera un représentant de la source. Il définit le contenu de la source, et ses critères (caractéristiques) comme ; les types des documents, les langues des documents, la fraîcheur des documents et le coût des documents (gratuit ou payant).

3- Définir une fonction permettant d'apparier le profil utilisateur et sa requête avec le profil des sources. Cette fonction doit prendre en compte le centre d'intérêts de l'utilisateur et ses préférences.

4- Exploiter cette fonction dans les processus de sélection des sources et de fusion des résultats retournés par les sources sélectionnées.

Ce chapitre est organisé de la manière suivante ; Dans la section 2, nous présentons notre solution pour personnaliser le processus de sélection de sources et le processus de fusion des résultats des sources sélectionnées. La section 3 présente l'expérimentation et l'évaluation de l'approche. La section 4 conclut le chapitre.

# 2. Un système d'apprentissage des profils sources et utilisateurs prenant en compte les préférences des utilisateurs

Nous proposons la conception d'un système d'apprentissage par l'exemple pour l'acquisition du profil source et celui de l'utilisateur. Nous prenons en compte dans ce profil utilisateur ses préférences sur les critères des documents des sources.

Pour bien illustrer la solution proposée, nous l'organisons en deux phases (de la même manière que dans le chapitre précédent) :

- La première phase appelée phase de construction et évolution des profils. Cette phase consiste à développer un système d'apprentissage permettant d'acquérir le profil source ainsi que celui de l'utilisateur.

- La deuxième phase appelée phase d'accès personnalisé à des sources d'information distribuées. Cette phase décrit la manière d'intégrer le profil utilisateur et celui des sources dans le processus de sélection de sources et le processus de fusion des résultats des sources sélectionnées.

## 2.1. Phase de construction et évolution des profils

Dans cette section nous présentons notre façon à définir, instancier, et mettre à jour les profils des sources et ceux des utilisateurs.

### *2.1.1. Profil source*

#### *2.1.1.1. Définition du profil source*

Le contenu de ce profil constitue le représentant d'une source d'information. Il décrit les informations que la source contient et leur nature. La nature des informations représente les critères (caractéristiques) de la source. Nous définissons le profil d'une source par les attributs suivants :

- *Identité :* est un attribut qui permet d'identifier la source, nous pouvons utiliser l'*url* et le *nom*,

- *Contenu :* est un attribut qui donne une vue du contenu de l'ensemble des documents dans la source.

- *Critères-documents :* est un multi-attribut qui nous informe sur la nature des documents dans la source, à savoir, les types des documents, les langues des documents, la fraîcheur des documents et le coût des documents.

La figure suivante (figure 5.1) montre le schéma d'un profil source :

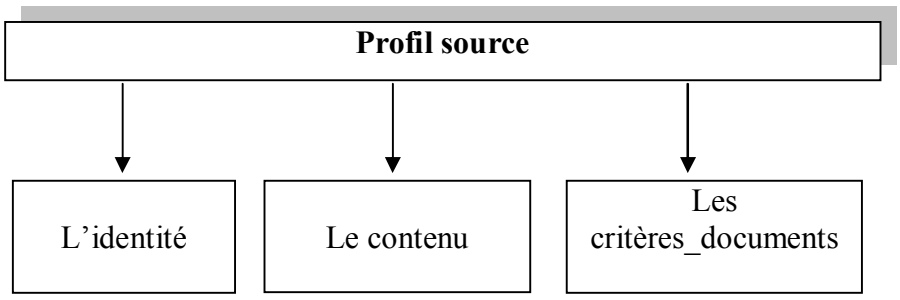


**Figure 5.1 :** Profil source

### *2.1.1.2. Instanciation du profil source*

L'étape de l'instanciation du profil source consiste à initialiser ces différents attributs :

- *Identité :* l'*url* et le *nom* d'une source sont fixés au moment de l'implémentation du prototype.

- *Contenu :* le contenu d'une source est représenté par un vecteur (ensemble) de termes pondérés, en utilisant le modèle vectoriel [Salton et al 1975], qui représente la source comme un gigantesque document. Nous utilisons le schéma *tf*idf* pour la pondération des termes. Ces termes sont extraits des *k* premiers documents retournés par la source pour la requête de l'utilisateur.

- *Critères-documents :* est un multi-attribut qui permet de mesurer l'importance de chaque critère dans la source. Chaque critère possède un ensemble de valeurs, pour chaque valeur nous associons un poids. Le poids d'une valeur d'un critère mesure son importance dans la source.

  Nous exploitons les critères suivants :

  - *Format :* ce critère indique les formats des documents dans la source, ses valeurs sont : *PDF, HTML, DOC, PS, XLS, PPT.*

  - *Langue :* ce critère indique les langues des documents dans la source, ses valeurs sont : *Français, Anglais.*

- *Fraîcheur :* ce critère indique la fréquence de mise à jour des documents de la source, ses valeurs sont : *Fraîche, Non_fraîche.* Le poids de la valeur *Fraîche* mesure le taux (une estimation) des documents d'actualités. Cependant, le poids de la valeur *Non_Fraîche* mesure le taux des documents qui ne sont pas d'actualités.

- *Coût :* ce critère mesure le taux des documents payants et celui des documents gratuits, défini respectivement par les deux valeurs : *Payant, Non_payant.*

Nous utilisons les notations suivantes pour exprimer les *critères-documents* d'une source *:*

$f$ est l'ensemble des critères-documents d'une source,

soit $f=\{Format, Langue, Fraîcheur, Coût\}$

$f(i)$ est le $i^{ème}$ critère d'une source, $f(i) \in f$

$vf(i)$ est l'ensemble des valeurs du $i^{ème}$ critère

$vf(1)$= {PDF, HTML, DOC, PS, XLS, PPT}
$vf(2)$= { Français, Anglais }
$vf(3)$= {Fraîche, Non_ Fraîche}
$vf(4)$= { Payant, Non_payant}

$vf(i,j)$ est la j$^{ème}$ valeur du i$^{ème}$ critère: $vf(i, j) \in vf(i)$

$p\_vf^{(s)}(i,j)$ est le poids associé à la j$^{ème}$ valeur du i$^{ème}$ critère d'une source *s.*

$p\_vf^{(s)}(i,j)$ est un poids qui représente l'importance de la j$^{ème}$ valeur du i$^{ème}$ critère d'une source *s.*

Ce poids est calculé comme suit :

Concernant les critères *Format, Langue, Coût,* le poids $p\_vf^{(s)}(i,j)$ représente un pourcentage des documents ayant la $j^{ème}$ valeur du $i^{ème}$ critère dans la source *s*. Ce poids est calculé par la formule suivante :

$$p\_vf^{(s)}(i,j) = \frac{\left|d^{(s)}(i,j)\right|}{\left|d^{(s)}\right|}$$

Avec $d^{(s)}(i,j)$ est l'ensemble des documents analysés ayant la $j^{ème}$ valeur du $i^{ème}$ critère dans la source *s*, $d^{(s)}$ est l'ensemble de tous les documents analysés de la source *s*.

Concernant le critère *Fraîcheur*, nous calculons les poids des valeurs *Fraîche* et *Non_Fraîche*, d'une manière qui nous permet de favoriser une source ayant des documents récents si l'utilisateur s'intéresse aux documents récents, et de favoriser les sources ayant les documents non récents dans le cas contraire.

La difficulté ou la contrainte concernant le critère *Fraîcheur*, est de déterminer sur quelle base nous pouvons dire qu'un document est récent ou non ? Cette contrainte ne nous permet pas d'utiliser la formule définie ci-dessus pour le critère fraîcheur.

Pour pallier à cette contrainte, nous définissons une autre formule pour calculer les poids des valeurs du critère *Fraîcheur*. Cette formule permet d'affecter un poids aux valeurs *Fraîche* et *Non_Fraîche* proportionnel aux dates d'éditions des documents d'une source comme suit :

$$p\_vf^{(s)}(Fraîcheur, j) = \begin{cases} \dfrac{\sum_{d=1}^{k} année_d^{(s)}}{k * annéeActue\ lle} & Si \quad j = Fraîche \\ 1 - \dfrac{\sum_{d=1}^{k} année_d^{(s)}}{k * annéeActue\ lle} & Si \quad j = Non\_fraîche \end{cases}$$

Avec *k* est le nombre de documents analysés de la source *s*, $année_d^{(s)}$ est l'année d'édition du document *d* d'une source *s*, *annéeActuelle* est l'année en cours.

### *2.1.1.3. Apprentissage du profil source*

Dans l'objectif de diminuer le temps de réponse du processus de recherche, nous avons décidé de ne pas mettre à jour le profil d'une source qu'à la demande de l'utilisateur ou lors d'une requête non similaire aux requêtes sur lesquelles le profil de la source est calculé. Le profil source est calculé pour une requête de l'utilisateur donnée, puis il sera utilisé pour les autres requêtes similaires à cette requête. Dans le cas où l'utilisateur demande la mise à jour du profil source, ou dans le cas d'une nouvelle requête exprimant un intérêt ou un domaine tout à fait différent de cette requête, le courtier refait les calculs

du contenu des attributs de la source tels qu'ils sont définis ci-dessus, en utilisant la requête en cours de l'utilisateur dans les calculs.

- **Notes importantes :**

(1) Le contenu et les poids des critères du profil d'une source sont associés à un utilisateur *u*, pour les autres utilisateurs le profil d'une source peut avoir d'autres contenus et d'autres poids des critères.

(2) Rappelons qu'un utilisateur peut avoir plusieurs centres d'intérêts. Pour cela, nous pouvons associer à l'utilisateur des profils sources différents, chaque profil source est associé à un centre d'intérêts de l'utilisateur.

Nous avons procédé de cette façon pour extraire le profil d'une source, car nous n'avons pas les moyens d'accéder à l'ensemble du contenu d'une source, qui peut être très vaste. La seule façon qui nous permet d'extraire le contenu d'une source est d'analyser la liste des documents qu'elle retourne pour une requête donnée. Cela montre bien qu'une source peut retourner des résultats différents pour des requêtes différentes, ce qui donne des profils différents à une source pour des requêtes différentes.

### *2.1.2. Profil utilisateur*

#### *2.1.2.1. Définition du profil de l'utilisateur*

Le profil de l'utilisateur dans notre approche, consiste à définir les différents centres d'intérêts de l'utilisateur, et les différentes préférences de l'utilisateur sur les critères des sources. Le profil de l'utilisateur est constitué alors des attributs suivants :

- *les données personnelles :* Cet attribut permet d'identifier l'utilisateur, nous utilisons : *code_utilisateur, mot de passe, nom, age,*

- *l'historique des recherches :* cet attribut permet d'identifier les centres d'intérêts et les préférences de l'utilisateur, il est constitué de l'ensemble de documents pertinents consultés par l'utilisateur durant ces recherches,

- *les centres d'intérêts de l'utilisateur :* les centres d'intérêts de l'utilisateur sont représentés par un ensemble de vecteurs de termes pondérés selon le schéma d'indexation *tf*idf* du modèle vectoriel. Chaque vecteur de termes définit un centre d'intérêts de l'utilisateur. L'ensemble des documents de l'historique des recherches relatif forme la collection de calcul.

- *les préférences :* c'est un multi-attribut permettant d'estimer les différentes préférences de l'utilisateur sur les critères d'une source, nous utilisons alors les préférences sur les types des documents, les langues des documents, la fraîcheur des documents, le coût des documents.

  La description des préférences de l'utilisateur doit être compatible avec celle des critères des sources, pour pouvoir les apparier. Nous utilisons alors les notations suivantes pour exprimer les *préférences* de l'utilisateur *:*

  *p* est l'ensemble des préférences de l'utilisateur sur les critères des sources,

  $p = \{Format, Langue, Fraîcheur, Coût\}$

  *p(i)* est la *préférence* de l'utilisateur relative au $i^{ème}$ critère des sources,

  $p(i) \in p$

  *vp(i)* est l'ensemble des valeurs de la $i^{ème}$ préférence

  *vp(1)*= {PDF, HTML, DOC, PS, XLS, PPT}
  *vp(2)*= {Français, Anglais }
  *vp(3)*= {Fraîche, Non_ Fraîche}
  *vp(4)*= {Payant, Non_payant}

  *vp(i,j)* est la $j^{ème}$ valeur de la $i^{ème}$ préférence : $vp(i, j) \in vp(i)$

  $p\_vp^{(u)}(i,j)$ est le poids associé au $j^{ème}$ valeur de la $i^{ème}$ préférence de l'utilisateur *u.*

  $p\_vp^{(u)}(i,j)$ est un poids qui représente l'importance de la $j^{ème}$ valeur de la $i^{ème}$ préférence pour l'utilisateur *u.*

La figure suivante (figure 5.2) montre le schéma d'un profil utilisateur :

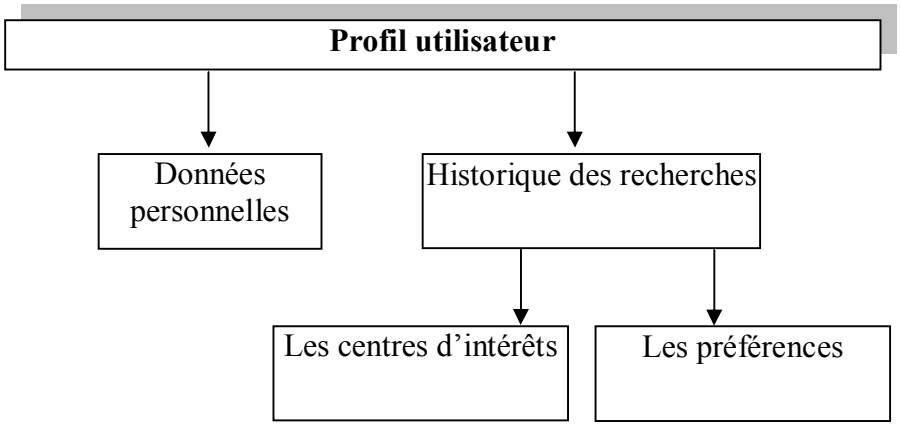


**Figure 5.2.** Intégration des préférences dans le profil utilisateur

#### *2.1.2.2. Instanciation du profil de l'utilisateur*

L'étape de l'instanciation du profil utilisateur consiste à initialiser les différents attributs qui le constituent :

- *les données personnelles :* le code de utilisateur, le mot de passe, le nom et l'age sont introduits par l'utilisateur,

- *l'historique des recherches :* initialement vide,

- *le centre d'intérêts :* initialement vide,

- *les préférences :* initialement les poids de toutes les valeurs des préférences de l'utilisateur sont équitablement initialisés à 1.

#### *2.1.2.3. Apprentissage du profil de l'utilisateur*

L'apprentissage du profil utilisateur est effectué après chaque session de recherche de l'utilisateur. Le courtier, sauvegarde l'ensemble des documents pertinents pour l'utilisateur parmi les documents qu'il a retournés à l'utilisateur. Cet ensemble est utilisé pour la mise à jour du profil utilisateur :

- ***Les données personnelles :*** ces données ne sont pas mises à jour, sauf le mot de passe, qui est modifié par l'utilisateur s'il-le désire.

- ***L'historique des recherches :*** c'est un ensemble de documents pertinents pour l'utilisateur constitué durant ces recherches précédentes. Cet ensemble de documents permet de spécifier les centres d'intérêts de l'utilisateur. A partir de cet ensemble de documents, le courtier peut déduire les centres d'intérêts de l'utilisateur et ses préférences. Cet ensemble contient les documents jugés pertinents par l'utilisateur durant sa dernière phase de recherche

- ***Les centres d'intérêts*** *:* un centre d'intérêts de l'utilisateur est un ensemble de termes pondérés extraits de l'historique des recherches, utilisant la technique d'indexation *tf*idf* du modèle vectoriel. Les études effectuées dans ce domaine, ont montré qu'un utilisateur peut avoir différents centres d'intérêts. Pour cela, nous utilisons plusieurs vecteurs de termes pondérés. Chaque vecteur de termes présente un centre d'intérêts particulier de l'utilisateur. Ces vecteurs sont calculés et mis à jour itérativement comme suit :

Soit $V^{(u)}$ l'ensemble des vecteurs des centres d'intérêts de l'utilisateur *u*. initialement $V^{(u)}$ est vide ($|V^{(u)}|=0$). Après chaque session de recherche de l'utilisateur, l'ensemble $V^{(u)}$ est mis à jour comme suit :

- Extraire de l'historique des recherches en cours le vecteur des termes pertinents, pondérés par le schéma *tf*idf*. Soit ce vecteur $v_i^{(u)}$

- Si $|V^{(u)}| = 0$ alors insérer le vecteur $v^{(u)}_i$ dans $V^{(u)}$.

- Sinon, calculer le degré de similarité entre chaque vecteur $v_j^{(u)}$ dans $V^{(u)}$ et le vecteur $v_i^{(u)}$. Nous utilisons la formule suivante dans le calcul de cette similarité :

$$Sim(V_j^{(u)}, V_i^{(u)}) = \frac{V_j^{(u)} \bullet V_i^{(u)}}{\left|V_j^{(u)}\right| \times \left|V_i^{(u)}\right|}$$

$$\forall\, j \in \left|V^{(u)}\right|$$

- S'il existe un vecteur $v_m^{(u)}$ ayant sa similarité avec le vecteur $v_i^{(u)}$ qui dépasse un seuil (défini lors de l'implémentation) alors :

  - fusionner les deux vecteurs $v_l^{(u)}$ et $v_m^{(u)}$ ayant la plus grande similarité. Nous notons le vecteur résultant $v_k^{(u)}$.

  - trier les termes du nouveau vecteur $v_k^{(u)}$ par ordre décroissant de leurs poids.

  - supprimer les termes du nouveau vecteur $v_k^{(u)}$ qui ont les poids les plus inférieurs, en gardant les *m* termes avec les poids les plus élevés.

- Sinon insérer le vecteur $v_i^{(u)}$ dans $V^{(u)}$.

- ***Les préférences :*** l'acquisition des préférences de l'utilisateur consiste à recalculer les poids des valeurs de chaque préférence de l'utilisateur sur les critères des sources. Ces poids sont recalculés périodiquement à chaque session de recherche de l'utilisateur. Nous utilisons l'historique des recherches de l'utilisateur dans ce calcul comme suit :

  Concernant les préférences sur les critères *Format, Langue, Coût,* le poids $p\_vp^{(u)}(i,j)$ représente le pourcentage des documents ayant la $j^{ème}$ valeur de la $i^{ème}$

préférence dans l'historique des recherches de l'utilisateur *u*. Ce poids est calculé par la formule suivante :

$$p\_vp^{(u)}(i,j) = \frac{\left|d^{(u)}(i,j)\right|}{\left|d^{(u)}\right|}$$

Avec $d^{(u)}(i,j)$ est l'ensemble des documents ayant la $j^{ème}$ valeur de la $i^{ème}$ préférence dans l'historique des recherches de l'utilisateur *u*, $d^{(u)}$ est l'ensemble de tous les documents dans l'historique des recherches de l'utilisateur *u*.

Concernant la préférence sur le critère *Fraîcheur*, nous calculons les poids des valeurs *Frarîche* et *Non_fraîche*, d'une manière qui nous permet d'affecter le poids élevé à la valeur *Fraîche* si l'utilisateur s'intéresse aux documents récents, et d'affecter le poids élevé à la valeur *Non_frarîche* dans le cas contraire.

La contrainte concernant la préférence *Fraîcheur*, ne nous permet de savoir sur quelle base un document est récent ou non. Par conséquent, comme dans le cas des sources, elle ne nous permet pas d'utiliser la formule définie ci-dessus pour calculer les poids des valeurs de la préférence *Fraîcheur*.

Pour pallier à cette contrainte, nous définissons une autre formule pour calculer les poids des valeurs de la préférence *Fraîcheur*. Cette formule permet d'affecter un poids à la valeur *Fraîche* proportionnel aux dates d'éditions des documents de l'historique des recherches de l'utilisateur *u*, comme suit :

$$p\_vp^{(u)}(Fraîcheur,j) = \begin{cases} \dfrac{\sum_{d=1}^{k} année_d^{(u)}}{k*annéeActuelle} & Si\ \ j=fraîche \\ 1-\dfrac{\sum_{d=1}^{k} année_d^{(u)}}{k*annéeActuelle} & Si\ \ j=non\_fraîche \end{cases}$$

Avec *k* est le nombre de documents dans l'historique des recherches de l'utilisateur *u*, $année^{(u)}_d$ est l'année d'édition du document *d* dans l'historique des recherches de l'utilisateur *u*, *annéeActuelle* est l'année en cours.

## 2.2. Phase d'accès personnalisé à des sources d'informations distribuées

Dans cette phase nous présentons notre processus d'évaluation de la requête de l'utilisateur, qui consiste à fournir à l'utilisateur une seule liste des documents retournés par plusieurs sources d'information. La liste de documents doit être la plus pertinente possible par rapport à sa requête, son centre d'intérêts et ses préférences. Pour cela, nous proposons les étapes suivantes de l'évaluation de la requête :

1. La recherche (sélection) du centre d'intérêts courant de l'utilisateur.
2. La sélection des sources pertinentes en exploitant le profil des sources (contenu et critères), le profil de l'utilisateur (centre d'intérêts et préférences) et sa requête.
3. La fusion des listes des documents retournées par les sources sélectionnées, en prenant en considération la requête, le profil de l'utilisateur et ses préférences

### *2.2.1. Sélection du centre d'intérêts courant de l'utilisateur*

Un utilisateur peut avoir différents centres d'intérêts. La première chose à faire après la réception de la requête de l'utilisateur, c'est de trouver parmi les centres d'intérêts, dans son profil, celui qui correspond à la requête en cours. Ce centre d'intérêts est appelé le centre d'intérêts courant.

Dans le but de trouver le centre d'intérêts courant, le courtier calcule la similarité entre la requête de l'utilisateur et les centres d'intérêts existants dans le profil de l'utilisateur. Le centre d'intérêts ayant la similarité la plus élevée avec la requête, sera considéré le centre d'intérêts courant. La similarité est calculée comme suit :

$$Sim(q\ ,V_i) = \frac{q \bullet V_i}{|q| \times |V_i|}$$

$$\forall\ i \in \left|V^{(u)}\right|$$

Avec $q$ est la requête en cours de l'utilisateur, $v^{(u)}{}_i$ est le i$^{ème}$ centre d'intérêts de l'utilisateur, $V^{(u)}$ est l'ensemble des centres d'intérêts de l'utilisateur.

Le centre d'intérêts ainsi trouvé noté $v^{(u)}{}_i$, sera utilisé dans la suite du processus d'évaluation de la requête.

### *2.2.2. Processus de sélection des sources*

Le but du processus de sélection de sources est de réduire l'espace de recherche, sans diminuer la pertinence des résultats. Notre approche comme mentionnée ci-dessus, consiste en l'intégration du centre d'intérêts et les préférences de l'utilisateur dans l'étape de

sélection de sources. C'est une manière d'élaguer certaines sources d'information pour réduire la complexité de la recherche. Pour cela nous calculons un score $score_u^{(s)}$ pour chaque source *s*, sur lequel les sources seront sélectionnées. Ce score combine trois mesures :

1. *La similarité de la source par rapport à la requête de l'utilisateur notée* $sim_q^{(s)}$ *:* cette similarité est calculée entre l'attribut *contenu* du profil de la source et la requête *q* de l'utilisateur. Nous utilisons la formule de cosinus du modèle vectoriel dans le calcul de cette similarité. En sachant que d'autres formules peuvent également être utilisées pour le calcul de la similarité. Nous avons choisi la formule de cosinus au lieu d'autres car l'objectif de la thèse n'est de comparer les formules de similarité pour prendre la meilleure.

$$sim_q^{(s)} = \frac{\sum_{i=1}^{T} t_i * q_i}{\left(\sum_{i=1}^{T} t_i^2\right)^{1/2} \left(\sum_{i=1}^{T} q_i^2\right)^{1/2}}$$

   Avec, *S* est la source ; *Q* est la requête ; $t_i$ est le poids du i$^{ème}$ terme dans la source (ensemble des *k* premiers documents retournés par la source); $q_i$ : est le poids du i$^{ème}$ terme dans la requête ; *T* est le nombre de termes de la source utilisés. Le contenu d'une source est représenté par l'ensemble de ses *k* premiers documents retournés.

2. *La similarité d'une source par rapport au centre d'intérêts de l'utilisateur notée* $sim_u^{(s)}$ *:* cette similarité est calculée entre l'attribut *contenu* du profil de la source et le centre d'intérêts courant $v^{(u)}{}_i$ de l'utilisateur. Nous utilisons la formule de cosinus du modèle vectoriel dans le calcul de cette similarité, décrite ci-dessus, sauf que nous remplaçons la requête par le centre d'intérêts courant $v^{(u)}{}_i$.

3. *Le degré d'exactitude (satisfaction) entre les critères de la source et les préférences de l'utilisateur noté* $acc_u^{(s)}$ *:* ce degré d'exactitude est calculé en utilisant les poids des valeurs des critères de la source *s* et les poids des valeurs des préférences de l'utilisateur *u* par la formule suivante :

$$acc_u^{(s)} = \frac{1}{|f|} \sum_{i=1}^{|f|} \left( \frac{2 * \sum_{j=1}^{|vf(i)|} p\_vf^{(s)}(i,j) * p\_vp^{(u)}(i,j)}{\sum_{j=1}^{|vf(i)|} p\_vf^{(s)}(i,j)^2 + \sum_{j=1}^{|vf(i)|} p\_vp^{(u)}(i,j)^2} \right)$$

- ***Explication de la formule :*** Cette formule calcule le degré de satisfaction des critères de la source aux préférences de l'utilisateur. Dans la formule nous multiplions chaque poids d'une valeur « j » d'une préférence « i » de l'utilisateur *p_vp(i, j)* par le poids de la valeur « j » d'un critère « i » correspondante de la source *p_vf(i, j)*. Par exemple : le poids de la valeur *PDF* de la préférence *Format* doit être multiplié par le poids de la valeur *PDF* du critère *Format*. Par la suite tous les résultats des multiplications seront additionnés. La somme trouvée reflète le degré de satisfaction des critères de la source aux préférences de l'utilisateur. Cependant, cette somme est une valeur non normalisée, car elle peut être supérieure à 1. Dans le but d'obtenir une valeur normalisée, qui appartient à l'intervalle [0, 1], premièrement, nous multiplions la somme trouvée par 2, ensuite nous divisons le tout sur la somme des carrés des poids de toutes les valeurs des critères de la source et des préférences de l'utilisateur. Deuxièmement, le résultat trouvé sera divisé par le nombre de critères utilisés "| f |". Ainsi, le résultat final ($acc^{(s)}{}_u$) est une valeur normalisée, qui appartient à l'intervalle [0, 1].

  En conclusion, nous pouvons dire que cette formule permet de faire une correspondance entre les poids des valeurs des préférences de l'utilisateur avec les poids des valeurs des critères d'une source. La formule assure que la source satisfait au mieux les préférences de l'utilisateur tant que sa valeur ($acc^{(s)}{}_u$) sera supérieure.

Le score final $score_u^{(s)}$ de la source s est calculé par la combinaison des trois mesures $sim_u^{(s)}$, $sim_q^{(s)}$ et $acc_u^{(s)}$ comme suit :

$$score_u^{(s)} = \alpha * sim_q^{(s)} + (1-\alpha) * (\beta * sim_u^{(s)} + (1-\beta) * acc_u^{(s)})$$

avec $\alpha, \beta \in [0, 1]$.

Les sources sélectionnées sont celles ayant leur score final $score_u^{(s)}$ supérieur ou égal à la moyenne des scores finaux des sources.

- ***Explication de la formule :*** Cette formule permet de calculer le score (similarité) de la source par rapport à la requête et le profil de l'utilisateur. Autrement dit, elle permet d'intégrer le profil de l'utilisateur dans le calcul du score final de la source. Sachant que le profil de l'utilisateur dans notre approche est composé de deux parties : le centre d'intérêts et les préférences. Pour cela, la formule est une addition, combinée avec deux

constantes $\alpha$ *et* $\beta$ des trois mesures : (1) la similarité entre la source et la requête de l'utilisateur notée $sim_q^{(s)}$ ; (2) la similarité entre la source et le centre d'intérêts de l'utilisateur notée $sim_u^{(s)}$ ; (3) le degré de satisfaction des critères de la source aux préférences de l'utilisateur noté $acc^{(s)}{}_u$. Sachant que $\alpha, \beta \in [0, 1]$. Les deux constantes $\alpha$ *et* $\beta$ sont utilisées dans la formule pour deux raisons :

1- Dans le but d'obtenir un score normalisé, qui appartient à l'intervalle [0, 1].

2- Ces deux constantes$\alpha$ *et* $\beta$ sont des paramètres empiriques qui nous permettent d'illustrer lesquelles de ces trois mesures (la requête $sim_q^{(s)}$, le centre d'intérêts $sim_u^{(s)}$, les préférences $acc_u^{(s)}$) sont importantes par rapport aux autres pour l'utilisateur. Autrement dit, ils nous permettent d'avoir l'ordre d'importance de ces trois mesures pour l'utilisateur. Pour calculer cet ordre d'importance de ces trois mesures, il suffit de changer les valeurs des constantes $\alpha$ *et* $\beta$ lors de l'implémentation et l'évaluation de l'approche et voir le jugement de pertinence de l'utilisateur pour chacun des cas des valeurs attribuées à $\alpha$ et $\beta$.

Ce deuxième point est très important dans l'évaluation de notre approche. Pour bien illustrer ce point voir la section d'évaluation (section 3.1).

### *2.2.3. Processus de fusion des résultats*

Le but de l'étape de fusion des résultats est de sélectionner et de trier les documents retournés par les différentes sources d'information sélectionnées dans une seule liste finale de résultats. Cette liste finale sera retournée à l'utilisateur. Dans le but d'améliorer la pertinence du processus de fusion des résultats, nous avons intégré le profil de l'utilisateur (centre d'intérêts et préférences) dans ce processus de fusion. Plus précisément, nous calculons trois mesures pour chaque document *d* de chaque liste de résultats retournée par les sources sélectionnées :

1. premièrement, nous calculons le degré de similarité entre le document *d* et la requête *q* de l'utilisateur *u,* noté $sim_q^{(d)}$,

2. deuxièmement, nous calculons le degré de similarité entre le document *d* et le centre d'intérêts courant $V^{(u)}i$ de l'utilisateur *u,* noté $sim_u^{(d)}$,

3. troisièmement, nous calculons le degré d'exactitude entre le document *d* et les préférences de l'utilisateur *u*, noté $acc_u^{(d)}$,

Dans le calcul des deux degrés de similarités $sim_q^{(d)}$, $sim_u^{(d)}$, nous utilisons la formule de cosinus du modèle vectoriel décrite ci-dessus pour la source. Sauf que nous remplaçons la source par le document.

Dans le but de calculer le degré d'exactitude $acc_u^{(d)}$, nous associons pour chaque document *d* les mêmes critères que nous avons présentés ci-dessus dans le profil des sources à savoir ; les *types, les langues, les coûts, et la fraîcheur* des documents.

La description de ces critères pour un document est la même description utilisée pour le profil des sources. La différence se focalise dans le calcul des poids des valeurs de ces critères.

Nous utilisons les notations suivantes pour exprimer ces critères de documents :

*fd* est l'ensemble des critères des documents ,

$fd = \{Format, Langue, Fraîcheur, Coût\}$

*fd(i)* est le $i^{ème}$ critère du document, $fd(i) \in fd$

*vfd(i)* est l'ensemble des valeurs du $i^{ème}$ critère

*vfd(1)*= {PDF, HTML, DOC, PS, XLS, PPT}
*vfd(2)*= {Français, Anglais}
*vfd(3)*= {Fraîche, Non_ Fraîche}
*vfd(4)*= {Payant, Non_payant}

*vfd(i,j)* est la j$^{ème}$ valeur du i$^{ème}$ critère: $vfd(i, j) \in vfd(i)$

$p\_vfd^{(d)}(i,j)$ est le poids associé à la j$^{ème}$ valeur du i$^{ème}$ critère du document *d.*

$p\_vfd^{(d)}(i,j)$ est un poids qui représente l'importance de la j$^{ème}$ valeur du i$^{ème}$ critère du document *d.*

Ce poids est calculé comme suit :

1- Pour les deux critères *Format*, *Langue*, le poids $p\_vfd^{(d)}(i,j)$ est calculé comme suit :

$$p\_vfd^{(d)}(i,j) = \begin{cases} 1 \ si \ la \ j^{ème} \ valeur \ du \ i^{ème} \ critère \ existe \ dans \ le \ document \ d \\ 0 \ sinon \end{cases}$$

Avec $fd(i) \in \{Format, Langue\}$

2- Pour le critère fraîcheur, le poids $p\_vfd^{(d)}(i,j)$ est calculé comme suit :

$$p\_vfp^{(d)}(Fraîcheur, j) = \begin{cases} \dfrac{année^{(d)}}{annéeActue lle} & Si \quad j = fraîche \\ 1 - \dfrac{année^{(d)}}{annéeActue lle} & Si \quad j = non\_fraîche \end{cases}$$

Avec $année^{(d)}$ est l'année d'édition du document *d*, *annéActuelle* est l'année en cours.

3- Pour le critère coût, le poids $p\_vfd^{(d)}(coût, j)$ est calculé comme suit :

$$p\_vfp^{(d)}(coût, j) = \begin{cases} \dfrac{prix^{(d)}}{prixMax} & Si \quad j = Payant \\ 1 - \dfrac{prix^{(d)}}{prixMax} & Si \quad j = Non\_payant \end{cases}$$

Avec $prix^{(d)}$ est le prix du document *d*, et *prixMax* est le maximum des prix de tous les documents dans les résultats.

Ainsi, le degré d'exactitude $acc_u^{(d)}$, est calculé en utilisant les poids des valeurs des critères des documents et les poids des préférences de l'utilisateur, suivant la formule suivante :

$$acc_u^{(d)} = \frac{1}{|f|} \sum_{i=1}^{|f|} \left( \frac{2 * \sum_{j=1}^{|vf(i)|} p\_vfd^{(d)}(i,j) * p\_vp^{(u)}(i,j)}{\sum_{j=1}^{|vf(i)|} p\_vfd^{(d)}(i,j)^2 + \sum_{j=1}^{|vf(i)|} p\_vp^{(u)}(i,j)^2} \right)$$

- ***Explication de la formule :*** Cette formule est analogue à la formule de calcul du degré de satisfaction des critères d'une source aux préférences d'un utilisateur, car une source est vue comme un document gigantesque. La formule peut être alors appliquée sur un document. Elle calcule le degré de

satisfaction des critères d'un document aux préférences de l'utilisateur. Dans la formule nous multiplions chaque poids d'une valeur « j » d'une préférence « i » de l'utilisateur *p_vp(i, j)* par le poids de la valeur « j » d'un critère « i » correspondante du document *p_vfd(i, j)*. Par exemple : le poids de la valeur *PDF* de la préférence *Format* doit être multiplié par le poids de la valeur *PDF* du critère *Format*. Par la suite tous les résultats des multiplications seront additionnés. La somme trouvée reflète le degré de satisfaction des critères du document aux préférences de l'utilisateur. Cependant, cette somme est une valeur non normalisée, car elle peut être supérieure à 1. Dans le but d'obtenir une valeur normalisée, qui appartient à l'intervalle [0, 1], premièrement, nous multiplions la somme trouvée par 2, ensuite nous divisons le tout sur la somme des carrés des poids de toutes les valeurs des critères du document et des préférences de l'utilisateur. Deuxièmement, le résultat trouvé sera divisé sur le nombre de critères utilisés "| f |". Ainsi, le résultat final ($acc^{(d)}{}_u$) est une valeur normalisée, qui appartient à l'intervalle [0, 1]. En conclusion, nous pouvons dire que cette formule permet de faire une correspondance entre les poids des valeurs des préférences de l'utilisateur avec les poids des valeurs des critères d'un document. La formule assure que le document satisfait au mieux les préférences de l'utilisateur tant que sa valeur ($acc^{(d)}{}_u$) sera supérieure.

Le score final de chaque document *d* noté $score_q^{(d)}$ est calculé par combinaison des ses trois mesures $sim_q^{(d)}$, $sim_u^{(d)}$ et $acc_u^{(d)}$ suivant la formule suivante :

$$score_u^{(d)} = \alpha * sim_q^{(d)} + (1-\alpha) * (\beta * sim_u^{(d)} + (1-\beta) * acc_u^{(d)})$$

Avec $\alpha$ , $\beta \in [0, 1]$.

Les documents sont finalement fusionnés et triés dans une seule liste par ordre décroissant par rapport à leur score final $score_u^{(d)}$. Cette liste sera retournée à l'utilisateur.

- ***Explication de la formule :*** Par analogie à la formule du calcul du score d'une source, cette formule permet aussi de calculer le score (similarité) du document par rapport à la requête et le profil de l'utilisateur. Autrement dit, elle permet d'intégrer le profil de l'utilisateur dans le calcul du score final du document, sachant que le profil de l'utilisateur dans notre approche est composé de deux parties : le centre d'intérêts et les préférences. Pour cela, la formule est une addition, combinée avec deux constantes $\alpha$ et $\beta$, des trois mesures : (1) la similarité entre le document et la requête de l'utilisateur notée $sim_q^{(d)}$ ; (2) la similarité entre le document et le centre d'intérêts de l'utilisateur notée $sim_u^{(d)}$ ; (3) le degré de satisfaction des critères du

document aux préférences de l'utilisateur noté $acc^{(d)}{}_u$. Sachant que $\alpha, \beta \in [0, 1]$. Les deux constantes $\alpha$ et$\beta$ sont utilisées dans la formule pour deux raisons :

1- Dans le but d'obtenir un score normalisé, qui appartient à l'intervalle [0, 1].

2- Comme dans le cas de la sélection des sources, ces deux constantes $\alpha$ *et* $\beta$ sont des paramètres empiriques, qui nous permettent d'illustrer lesquelles de ces trois mesures (la requête $sim_q^{(d)}$, le centre d'intérêts $sim_u^{(d)}$, les préférences $acc_u^{(d)}$) sont importantes par rapport aux autres pour l'utilisateur. Autrement dit, ils nous permettent d'avoir l'ordre d'importance de ces trois mesures pour l'utilisateur. Pour calculer cet ordre d'importance de ces trois mesures, il suffit de changer les valeurs des constantes $\alpha$ *et* $\beta$ lors de l'implémentation et l'évaluation de l'approche et voir le jugement de pertinence de l'utilisateur pour chaque cas des valeurs attribuées à $\alpha$ et $\beta$.

Ce deuxième point est très important dans l'évaluation de notre approche. Pour bien illustrer ce point voir la section d'évaluation (section 3.1).

Remarquons que dans ce travail, il y a une symétrie entre les sources à sélectionner et les listes des documents à fusionner. C'est la raison pour la quelle les formules proposées présentent la même ossature pour les deux cas.

# 3. Expérimentation

Afin de diversifier l'information traitée, nous avons exploité plusieurs sources d'information appartenant à des secteurs différents comme suit :

- ***Scientifique*** *:*
  (1) http://www.scirus.com/
  (2) http://www.in-extenso.org/
  (3) http://www.agrisalon.com/

- ***Informatique :***
  (1) http://www.lemondeinformatique.fr/
  (2) http://www.informatiques.eu
  (3) http://www.inria.fr/

- ***Economie :***

(1) http://ese.rfe.org/
(2) http://www.inomics.com/cgi/repec_search
(3) http://www.wu-wien.ac.at/inst/vw1/zagler/search/

- ***Médecine :***
  (1) http://www.galenicom.com/
  (2) http://www.medvet.umontreal.ca/biblio/rech.htm
  (3) http://www.ncbi.nlm.nih.gov/pubmed/
  (4) http://www.genethique.org/moteur_recherche/recherche.asp

- ***Presse (news) :***
  (1) http://c.asselin.free.fr/french/actua.htm
  (2) http://search.freefind.com/find.html?id=3225682
  (3) http://www.courrierinternational.com/gabarits/html/default_online.asp
  (4) http://www.excite.fr/search/news

L'approche est expérimentée sur 100 requêtes différentes.

Nous avons évalué notre approche de deux manières, d'abord par la variation des paramètres $\alpha$ *et* $\beta$ utilisés dans les processus de sélection et de fusion, en second lieu en comparant notre approche à un algorithme de base qui ne considère pas l'information personnalisée. L'algorithme de base utilisé est l'algorithme de CORI.

## 3.1. Première évaluation

Dans le but d'illustrer l'ordre d'importance entre la requête, le centre d'intérêts et les préférences pour l'utilisateur, nous avons évalué l'approche sur différents cas (la requête avec centre d'intérêts et préférences, la requête avec centre d'intérêts et sans préférences, la requête sans centre d'intérêts et avec préférences, …etc.). Pour cela, nous avons varié les valeurs des constantes $\alpha$, $\beta$ que nous avons utilisées dans le processus de sélection de sources et le processus de fusion des résultats, comme suit :

*1$^{er}$ cas $\alpha$=1, $\beta$ quelconque:* ce cas spécifie que la pertinence des résultats est relative à la requête de l'utilisateur seulement, les préférences et le centre d'intérêts de l'utilisateur ne sont pas pris en compte.

*2$^{ème}$ cas $\alpha$=0.5, $\beta$=1:* ce cas spécifie que les préférences de l'utilisateur ne sont pas prises en compte, la pertinence des résultats est relative à la requête et le centre d'intérêts de l'utilisateur.

*3ème cas α=0.5, β=0 :* ce cas spécifie que le centre d'intérêts de l'utilisateur n'est pas pris en compte, la pertinence des résultats est relative à la requête et les préférences de l'utilisateur.

*4ème cas α=0.5, β=0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur d'une façon équitable.

*5ème cas α=0.5, β< 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais les préférences de l'utilisateur sont plus importantes que son centre d'intérêts.

*6ème cas α=0.5, β > 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais le centre d'intérêts de l'utilisateur est plus important que ses préférences.

*7ème cas α<0.5, β < 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais les préférences de l'utilisateur sont plus importantes que son centre d'intérêts, et les deux sont plus importants que sa requête.

*8ème cas α>0.5, β < 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais le centre d'intérêts de l'utilisateur est plus important que ses préférences, et les deux sont plus importants que sa requête.

*9ème cas α<0.5, β > 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais les préférences de l'utilisateur sont plus importantes que son centre d'intérêts, et les deux sont moins importants que sa requête.

*10ème cas α<0.5, β < 0.5 :* ce cas spécifie que la pertinence des résultats est relative à la requête, le centre d'intérêts et les préférences de l'utilisateur, mais le centre d'intérêts de l'utilisateur est plus important que ses préférences, et les deux sont moins importants que sa requête.

A chaque session de recherche d'un utilisateur, une valeur de précision est calculée suivant la formule suivante :

$$précision = \frac{nombre\ \ de\ \ documents\ \ pertinents\ \ sélectionnés}{nombre\ \ de\ \ documents\ \ sélectionnés}$$

Une précision moyenne est calculée pour l'ensemble des requêtes des utilisateurs pour chaque cas décrit ci-dessus. Le tableau (Tableau 5.1) suivant montre la précision moyenne trouvée pour chaque cas :

| | Precision (%) |
|---|---|
| case 1 | 21.7 |
| case 2 | 25.8 |
| case 3 | 24.5 |
| case 4 | 26.2 |
| case 5 | 28.3 |
| case 6 | 28.8 |
| case 7 | 27.1 |
| case 8 | 27.9 |
| case 9 | 30.1 |
| case 10 | 31.6 |

**Tableau 5.1.** Précision moyenne de chaque cas.

## 3.2. Deuxième évaluation

Comme mentionné auparavant, dans cette expérimentation, nous avons évalué et comparé notre approche à l'approche de CORI comme algorithme de base qui ne considère pas l'information personnalisée, afin de montrer l'influence de l'intégration de la personnalisation dans les résultats de recherche.

Pour chaque requête, nous avons calculé deux valeurs de précision des résultats de recherche retournés respectivement par l'approche de CORI et notre approche. Chaque valeur de précision est calculée pour chaque premiers ; 5 documents, 10 documents, 15 documents, 20 documents, 25 documents et 30 documents. Nous avons calculé alors les deux précisions moyennes. La précision moyenne relative à l'approche de CORI et la précision moyenne relative à notre approche, sur la base de ; 5 documents, 10 documents, 15 documents, 20 documents, 25 documents et 30 documents.

Le tableau suivant montre ces moyennes de précisions obtenues :

| Nombre de documents exploités | CORI | Our approach |
|---|---|---|
| 5 docs | 0,35 | 0,39 |
| 10 docs | 0,32 | 0,37 |
| 15 docs | 0,31 | 0,36 |
| 20 docs | 0,29 | 0,34 |
| 25 docs | 0,28 | 0,33 |
| 30 docs | 0,26 | 0,31 |

**Tableau 5.2.** Nos précisions moyennes et celles de CORI.

## 3.3. Discussion des résultats

La première évaluation indique que les résultats obtenus par le $10^{ème}$ cas sont les plus pertinents. Ces résultats nous donnent comme indicateurs que l'intégration du profil d'utilisateur dans le processus de sélection de sources et le processus de fusion des résultats améliore le résultat du processus de recherche. Outre, la première évaluation nous montre que la requête de l'utilisateur est plus significative (importante) que son centre d'intérêts, et ce dernier est plus significatif (important) que ses préférences.

La deuxième évaluation indique aussi que l'intégration de la personnalisation dans le processus de sélection de source et dans le processus de fusion des résultats améliore la pertinence des résultats de recherche.

Pour plus de détails, l'évaluation que nous avons faite, nous permet de répondre à ces trois questions, qui reflètent la contribution de notre approche :

1- L'intégration du profil utilisateur dans le processus de sélection de sources et de fusion des résultats améliore-t-elle la pertinence des résultats de recherche ?

2- Quel paramètre est le plus important entre la requête et le profil de l'utilisateur dans la pertinence des résultats de recherche ?

3- Quel paramètre est le plus important entre le centre d'intérêts et les préférences de l'utilisateur dans la pertinence des résultats de recherche ?

A partir du tableau 5.1 obtenu par la première expérimentation nous pouvons répondre à ces questions :

1- L'intégration du profil de l'utilisateur dans le processus de sélection de sources et de fusion des résultats améliore la pertinence des résultats de recherche car, dans le tableau, les cas où le profil est utilisé (cas 2, 3 ….10) donnent des résultats meilleurs que le cas où le profil n'est pas utilisé (cas 1).

2- La requête de l'utilisateur est plus importante que son profil car, dans le tableau, les cas où nous avons donné de l'importance à la requête (cas 9, 10) donnent des résultats meilleurs que les cas où nous avons donné de l'importance au profil (cas 7, 8).

3- Le centre d'intérêts est plus important que les préférences dans le profil de l'utilisateur car, dans le tableau, le cas où nous avons donné de l'importance au centre d'intérêts (cas 6) donne des résultats meilleurs que le cas où nous avons donné de l'importance aux préférences (cas 5).

En outre, le tableau 5.2 de la deuxième expérimentation nous a confirmé de plus que l'intégration du profil de l'utilisateur dans le processus de sélection de sources et de fusion des résultats améliore la pertinence des résultats de recherche.

# 4. Conclusion

Dans ce chapitre nous avons présenté une approche permettant de personnaliser l'utilisateur dans son accès à de multiples sources d'information. L'approche propose des solutions pour pallier aux insuffisances rencontrées dans l'approche précédente. Nous avons proposé des techniques permettant de prendre en compte ; (1) les différents centres d'intérêts de l'utilisateur ; (2) les préférences de l'utilisateur relatives aux critères des documents des sources, à savoir ; le type, la langue, la fraîcheur et le coût d'un document. Dans cette approche la pertinence des résultats est relative à trois mesures correspondant à la requête de l'utilisateur, son centre d'intérêts et ses préférences. L'approche est évaluée sur 100 requêtes, en utilisant plusieurs sources d'information réparties sur différents secteurs, en deux expérimentations.

Le but de la première expérimentation est de connaître l'importance des ces mesures (requête, centre d'intérêts et préférences) dans la pertinence des résultats de recherche. Pour cela, nous avons évalué l'approche suivant 10 cas différents. Chaque cas présente une importance meilleure d'une mesure par rapport aux autres. Les résultats d'évaluation des différents cas montrent que l'intégration du profil utilisateur dans le processus de sélection et de fusion améliore les résultats de recherche. Ils montrent aussi que la requête de l'utilisateur est plus importante que son centre d'intérêts, et ce dernier est plus important que ses préférences.

Le but de la deuxième expérimentation est d'évaluer et comparer notre approche à une approche qui utilise un algorithme de base de CORI de sélection de source et de fusion des résultats. Les résultats obtenus nous confirment aussi que l'intégration de la personnalisation dans le processus de sélection de sources et dans le processus de fusion des résultats améliore la pertinence des résultats de recherche.

# Chapitre 6

# Un système multi-agents pour l'intégration du modèle utilisateur dans un SRID

## 1. Introduction

Dans le chapitre précédent, nous avons développé une approche d'accès personnalisé à des sources d'information distribuées. L'approche consiste en l'intégration du profil de l'utilisateur (ses centres d'intérêt et ses préférences) dans le processus de sélection des sources d'information à interroger et dans le processus de fusion des résultats retournés par ces sources. Les résultats d'évaluation que nous avons obtenus ont montré que l'intégration du profil de l'utilisateur dans le processus de sélection de sources et de fusion des résultats améliore la pertinence des résultats de recherche. Cependant, elle présente deux limites à savoir [Kechid et Drias 2008] :

1- *le temps de réponse :* le processus de recherche nécessite un temps de réponse important, qui est dû au considérable volume d'information à traiter par le processus de sélection de source et de fusion (plusieurs sources d'information). En effet, l'utilisateur est intéressé par les documents qui répondent à son profil, mais il préfère aussi les trouver plus rapidement.

2- *L'extensibilité :* l'ajout d'une nouvelle source d'information au système nécessite une importante modification du code source du système.

Dans ce chapitre, nous proposons une approche basée agents pour remédier à ces limites. Nous nous sommes attelés à concevoir un système multi-agents pour implémenter de manière efficace tous les concepts et techniques proposées dans les chapitres précédents. L'idée de base de cette réflexion est due aux points suivants ; (1) les agents s'adaptent bien aux problèmes distribués ; (2) nous avons déjà exploité les agents dans nos travaux de magister [Kechid et Drias 2003a][Kechid et Drias 2003b] [Kechid et Drias 2005]. Nous pouvons résumer les caractéristiques d'un agent intelligent comme suit [Ferber 1995] :

- ***L'autonomie*** : L'agent doit pouvoir prendre des initiatives et agir sans intervention de l'utilisateur final. Dans le contexte du Web il doit pouvoir ? même si l'utilisateur est déconnecté.

- ***La capacité à communiquer et à coopérer*** : L'agent doit pouvoir échanger des informations plus ou moins complexes avec d'autres agents, avec des sources d'informations ou avec des humains et coopérer entre eux.
- ***La capacité à raisonner*** et ***à réagir à l'environnement :*** L'agent doit être capable de s'adapter à son environnement (qui peut être composé d'autres agents, du Web en général ou d'utilisateurs humains) et aux évolutions de celui-ci. Cette adaptation doit s'appuyer sur l'analyse de l'environnement extérieur des agents.

Plusieurs approches ont été développées à base d'agents intelligents pour l'apprentissage de l'utilisateur [Baldwin et al 2000] [Chen et Sycara 1998] [Seo et Zhang 2000] dans le contexte de la recherche d'information distribuée. D'autres approches ont conçu des agents intelligents pour capturer les informations distribuées [Ogston 2003] [Yu et Singh 2003][Zhang et al 2004]. Notre contribution s'intéresse à la conception d'un système multi agents global pour l'intégration du modèle utilisateur pour la recherche d'information. Le concept d'agent était implicitement présent dans les approches séquentielles décrites précédemment. Le terme *courtier* par exemple est bel et bien une terminologie plutôt propre au domaine de la technologie des agents et plus précisément au commerce électronique. Ceci témoigne de l'adéquation et la pertinence de cette technologie pour aborder notre problématique.

Ce chapitre est organisé comme suit : la section 2 présente l'architecture du système proposé. La section 3 présente les éléments de base du système. La section 4 présente l'architecture interne des agents du système. Nous présentons dans la section 5 une étude comparative entre l'approche à base d'agents et l'approche séquentielle. Enfin la section 6 conclut le chapitre.

# 2. Architecture du système proposé

La figure 6.1 présente les différents composants du système que nous proposons. Avant d'aborder l'approche basée agents, nous présentons d'abord l'architecture générale de l'approche à base d'unités. L'architecture se divise en trois unités jouant chacune un rôle bien déterminé. Ces unités sont : (1) L'unité de communication avec l'utilisateur ; (2) L'unité de traitement de l'information ; (3) L'unité d'interrogation et d'extraction de l'information.

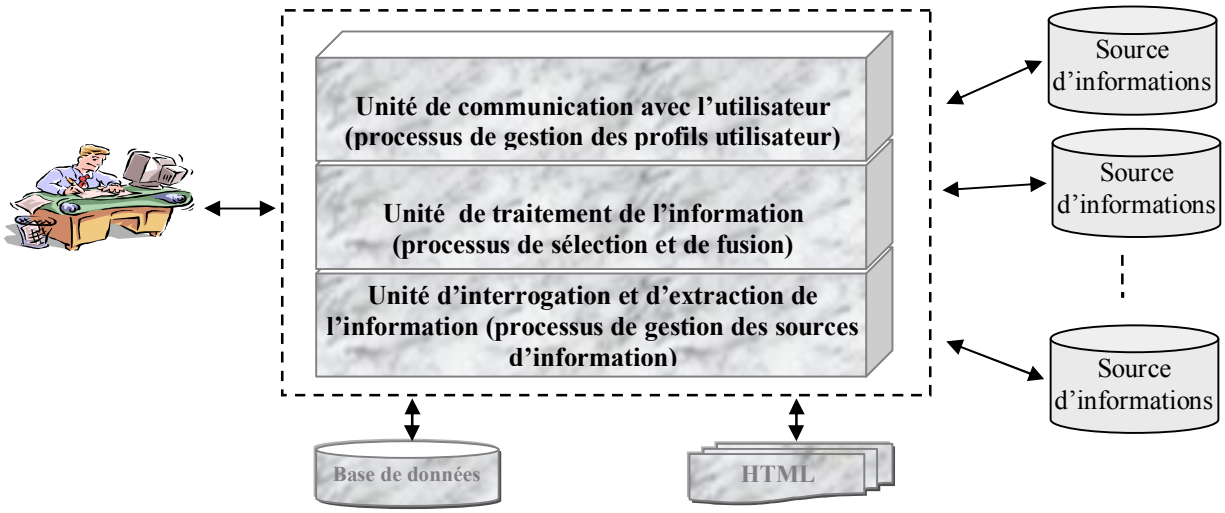


**Figure 6.1.** Composants du SRID proposé

- ***L'Unité de communication avec l'utilisateur*** *:* Cette unité est chargée de la communication entre l'utilisateur et les autres unités. Elle assure aussi la gestion (instanciation et mise à jour) des profils des utilisateurs. Nous pouvons résumer ces tâches par les points suivants; (1) la réception des besoins de l'utilisateur en information (requête et profil), et les transmet à l'unité de traitement de l'information ; (2) la réception de la liste des résultats à partir de l'unité de traitement de l'information et son transfert vers l'utilisateur ; (3) l'observation de l'utilisateur pendant ses consultations aux résultats de recherche, dans le but d'acquérir son profil.

- ***L'Unité de traitement de l'information*** *:* L'unité de traitement de l'information assure plusieurs tâches ; (1) la réception des besoins de l'utilisateur en information (requête et profil) à partir de l'unité de communication et les envoie à l'unité d'interrogation et d'extraction de l'information ; (2) la réception des résultats de recherche et les informations concernant chaque source d'information (tels que son score et les scores de ses documents) ; (3) à partir de ces informations, elle sélectionne les sources pertinences pour l'utilisateur, et fusionne les résultats des sources sélectionnées dans une seule liste, qui sera retournée à l'unité de communication avec l'utilisateur.

- ***L'Unité d'interrogation et d'extraction d'information :*** L'unité d'interrogation et d'extraction d'information communique avec les sources d'informations. Elle assure les tâches suivantes ; (1) la réception des besoins de l'utilisateur en information (requête et profil) à partir de l'unité de traitement de l'information ; (2) la transformation de ces informations en une requête compréhensible par les sources et les interroge par la requête résultante ; (3) à la réception des résultats des sources interrogées, elle extrait les informations que l'unité de traitement de l'information a besoin pour la sélection des sources et la fusion des résultats (le contenu représenté par un vecteur de mots pondérés et les critères de la source) et les lui envoie.

Rappelons que notre objectif est d'améliorer le temps de réponse et l'extensibilité du système. Nous avons envisagé l'intégration des agents dans notre approche, dans le but d'exécuter les tâches des unités en parallèle, pour accélérer le processus global de recherche. Pour cela, les tâches de chaque unité sont assurées par un ou plusieurs agents. L'architecture comporte quatre différents types d'agents :

- Un agent utilisateur, qui assure les tâches de l'unité de communication avec l'utilisateur.
- Un agent de sélection et de fusion qui assure les tâches de l'unité de traitement.
- Plusieurs agents sources, chaque agent source assure les tâches de l'unité d'interrogation et d'extraction de l'information pour une source.

L'architecture de notre approche à base d'agents est représentée par la figure suivante :

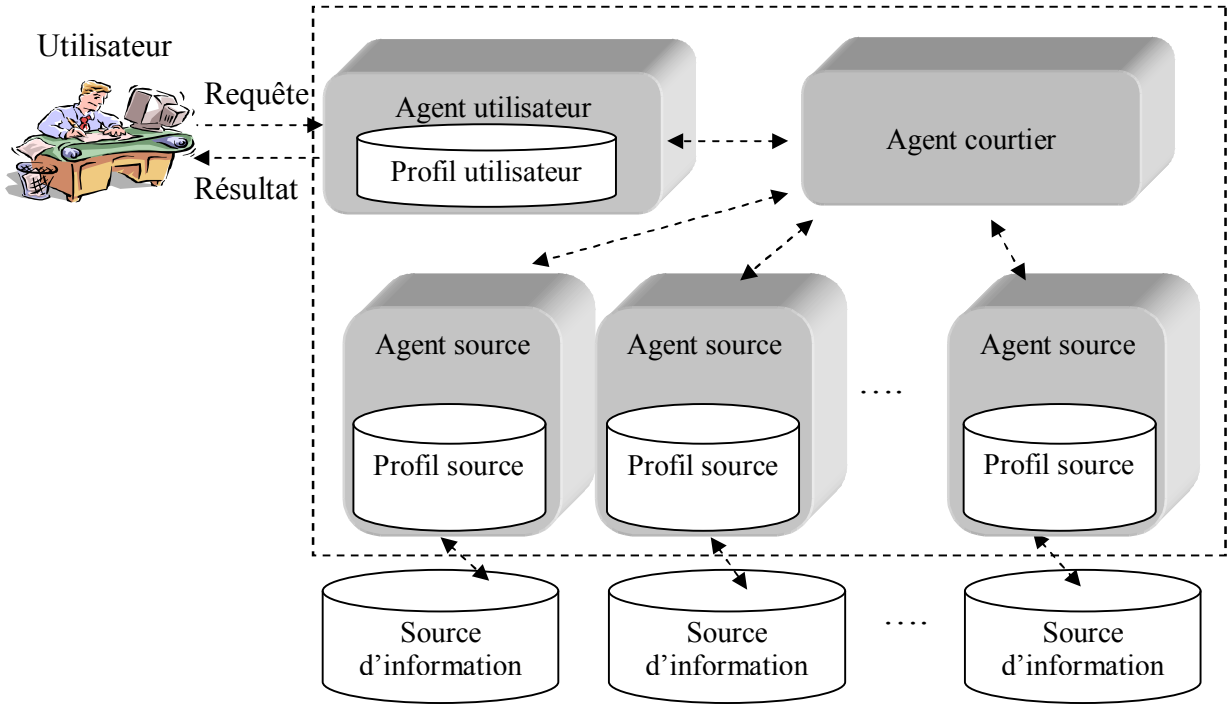


**Figure 6.2.** Architecture du système à base d'agents

## 3. Les éléments de base

Les différents éléments de base du système sont ; la requête, le document, le profil source, le profil utilisateur. Chaque élément est décrit comme suit :

- ***Une requête :*** est un ensemble de termes pondérés. Une requête est représentée par la figure suivante :

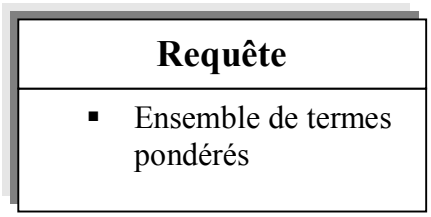


**Figure 6.3.** Description d'une requête

- ***Un document :*** décrit le contenu et les critères d'un document :
  - Titre : le titre du document,
  - Lien : le lien vers le document,
  - Contenu : le vecteur de termes pondérés,
  - Format : le format du document,
  - Langue : la langue du document,
  - Date publication : la date de création du document,
  - Coût : le prix du document.

Un document est représenté par la figure suivante :

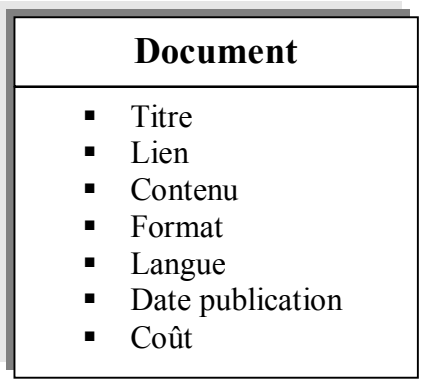


**Figure 6.4.** Description d'un document

- ***Un profil source :*** décrit les différents attributs d'une source :
  - Nom : nom de la source,
  - Url : l'url de la source,
  - Requête : la requête relative,
  - Score : le score de la source,
  - Fraîcheur : le poids du critère fraîcheur de la source
  - Coût : le poids du critère coût de la source,
  - Langue : les poids des attributs du critère langue,
  - Format : les poids des attributs du critère format,
  - Contenu : le vecteur de termes pondérés, décrivant le contenu de la source.

Un profil source est représenté par la figure suivante :

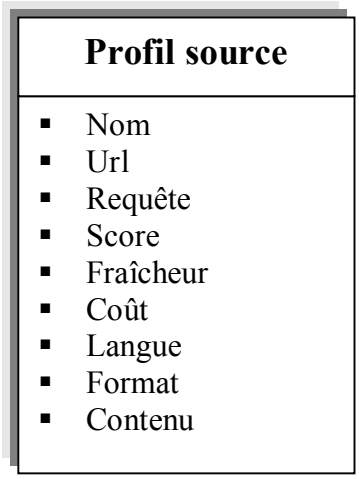


**Figure 6.5.** Description d'un profil source

- ***Un profil utilisateur :*** décrit les différents attributs d'une source :
  - IDF : l'identifiant de l'utilisateur,
  - Nom : le nom de l'utilisateur,
  - Thème : l'identifiant du thème du profil,
  - Historique : l'historique des recherches de l'utilisateur
  - Fraîcheur : le poids de la préférence fraîcheur de l'utilisateur,
  - Coût : le poids de la préférence coût de l'utilisateur,
  - Langue : le poids des attributs de la préférence langue,
  - Format : le poids des attributs de la préférence format,
  - Centre d'intérêt : le vecteur de termes pondérés décrivant le centre d'intérêts de l'utilisateur.

Un profil utilisateur est représenté par la figure suivante :

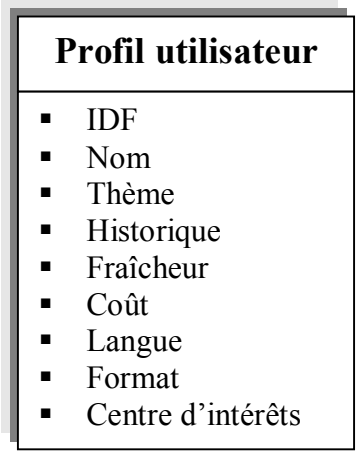


**Figure 6.6.** Description d'un profil utilisateur

# 4. Description interne des agents du système

Dans cette section nous présentons l'architecture interne des agents du système selon deux visions différentes. La première présente les agents dans un système à base de connaissances. La deuxième présente les agents dans un système à base de rôles.

Nous rappelons que cette approche est une modélisation à base d'agents de l'approche précédente (approche séquentielle). Par conséquent, nous utilisons les mêmes formules décrites dans l'approche séquentielle.

## 4.1. Un système à bases de connaissances

Un système à base de connaissances représente les agents comme un système expert (base de connaissances, système de raisonnement). Un agent intelligent ne s'implémente pas à l'aide d'un programme procédural. Il simule le comportement d'un humain, il possède donc des connaissances et un système de raisonnement ou d'apprentissage (moteur d'inférence).

### *4.1.1. Agent source*

L'agent source s'occupe d'une source d'information. A la réception de la requête de l'utilisateur et son profil via l'agent courtier, il lui extrait les informations nécessaires propres à sa source. Ces informations seront utilisées par l'agent courtier dans le processus de sélection de sources et le processus de fusion des résultats des sources sélectionnées. Ces informations forment le profil de la source. Le schéma descriptif suivant illustre l'agent source :

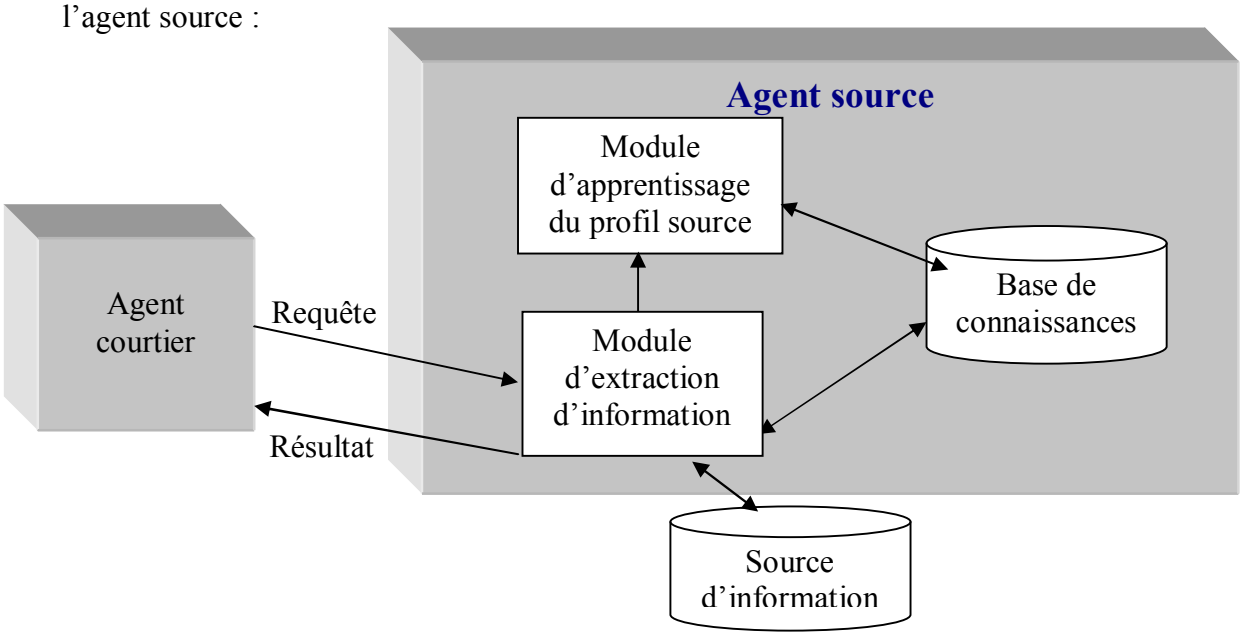


**Figure 6.7.** Schéma descriptif de l'agent source

*La base de connaissances* contient le *profil* de la source (contenu et critères documents) et les *règles d'inférences* (les différentes formules que nous avons développées afin d'accomplir le processus d'apprentissage du profil source).

Les deux modules, *module d'extraction d'information* et *module d'apprentissage,* assurent le fonctionnement (*rôle*) de l'agent source en exploitant sa *base de connaissances*. Les différentes tâches de chaque module sont décrites comme suit :

- ***Module d'extraction d'information*** : Il assure le processus d'extraction des informations nécessaires à une requête *q* donnée, selon les étapes suivantes ; (1) Il réceptionne la requête à partir de l'agent courtier ; (2) Il consulte sa base de connaissances (profil de la source) ; (3) Si une requête similaire à la requête *q* existe dans la base de connaissances, il extrait le profil de la source correspondant à la requête et l'envoie comme résultat à l'agent courtier ; (4) Sinon, il interroge la source pour en extraire l'information correspondante à la requête (liste de résultats) et l'envoyer au module d'apprentissage, afin de recevoir de ce dernier le profil de la source correspondant à la requête et l'envoyer à l'agent courtier.

- ***Module d'apprentissage*** : Il assure le processus d'apprentissage du profil source par l'exemple, selon les étapes suivantes ; (1) Il réceptionne la liste des résultats à partir du module d'extraction d'information ; (2) Il extrait à partir de cette liste le profil de la source correspondant à la requête en se basant sur les différentes formules associées (règles d'inférences) ; (3) Il ajoute ce profil à la base de connaissances et l'envoie au module d'extraction d'information.

### *4.1.2. Agent utilisateur*

Cet agent modélise le comportement de l'utilisateur vis-à-vis d'une requête de recherche de documents.

Le schéma descriptif (figure 6.8) illustre l'agent utilisateur. *La base de connaissances* contient le *profil* de l'utilisateur (centre d'intérêts et préférences sur les critères documents) et les *Règles d'inférences* (sont les différentes formules que nous avons développées afin d'accomplir le processus d'apprentissages du profil utilisateur).

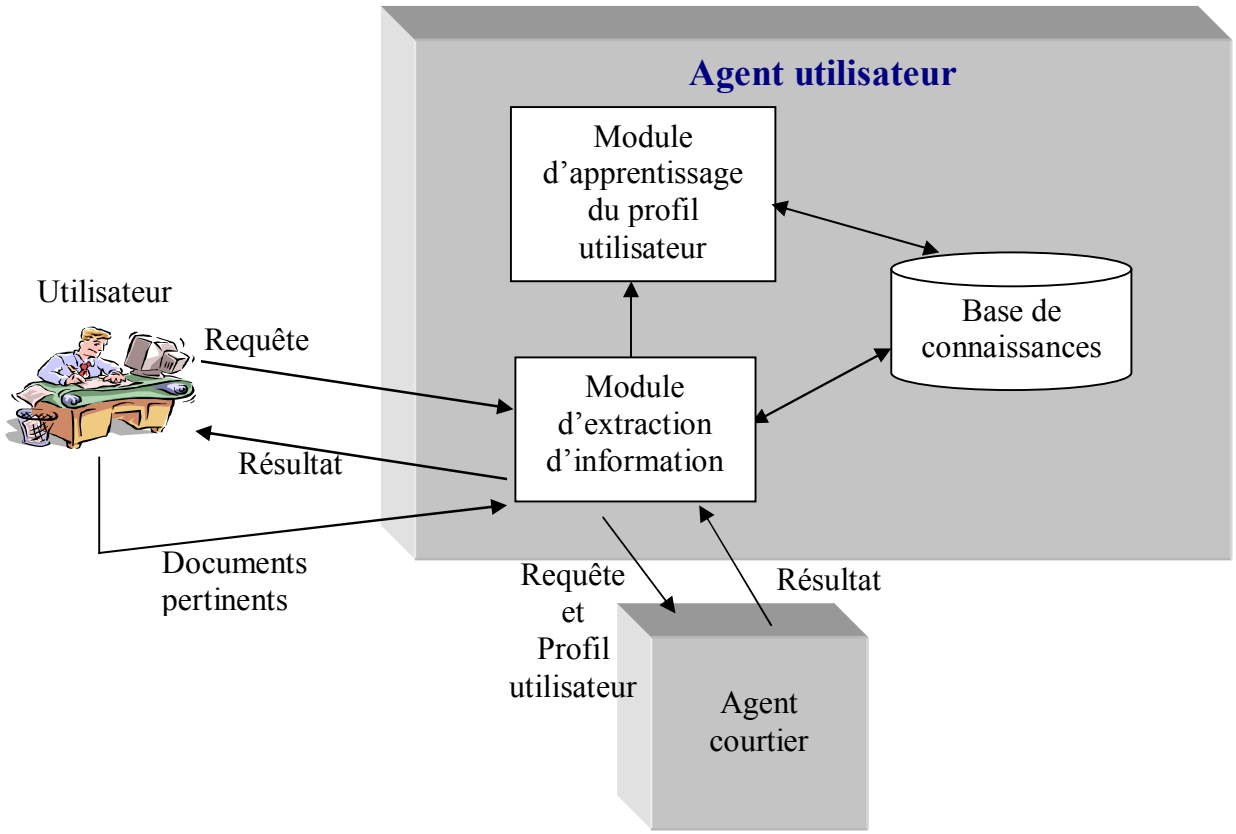


**Figure 6.8.** Schéma descriptif de l'agent utilisateur

Les deux modules, *module d'extraction d'information* et *module d'apprentissage,* assurent le fonctionnement (rôle) de l'agent utilisateur en exploitant sa *base de connaissances.* Les différentes tâches de chaque module sont décrites comme suit :

- ***Module d'extraction d'information*** : Il assure le processus d'extraction d'information nécessaires à une requête *q* donnée, selon les étapes suivantes ; (1) Il réceptionne la requête à partir de l'utilisateur ; (2) Il extrait à partir de la base des connaissances le profil courant de l'utilisateur correspondant à cette requête en exploitant les règles d'inférences correspondantes ; (3) Il envoie la requête et le profil courant de l'utilisateur à l'agent courtier, afin de recevoir de ce dernier la liste des résultats ; (4) Il affiche à l'utilisateur la liste des résultats reçue ; (5) Il récupère la liste des documents jugés pertinents par l'utilisateur et l'envoie au module d'apprentissage afin de mettre à jour le profil de l'utilisateur.

- ***Module d'apprentissage*** : Il assure le processus d'apprentissage du profil utilisateur par l'exemple, selon les étapes suivantes : (1) Il réceptionne la liste des documents pertinents pour l'utilisateur via le module d'extraction d'information ; (2) Il extrait de cette liste les informations nécessaires afin de mettre à jour le profil utilisateur, en se basant sur les règles d'inférences correspondantes.

### *4.1.3. Agent courtier*

Cet agent envoie la requête de l'utilisateur aux agents sources. Il reçoit le profil des sources (contenu et critères) et assure la sélection des sources pertinentes et la fusion des résultats des sources sélectionnées en une seule liste, qui sera retournée à l'utilisateur. L'agent courtier est présenté par le schéma suivant :

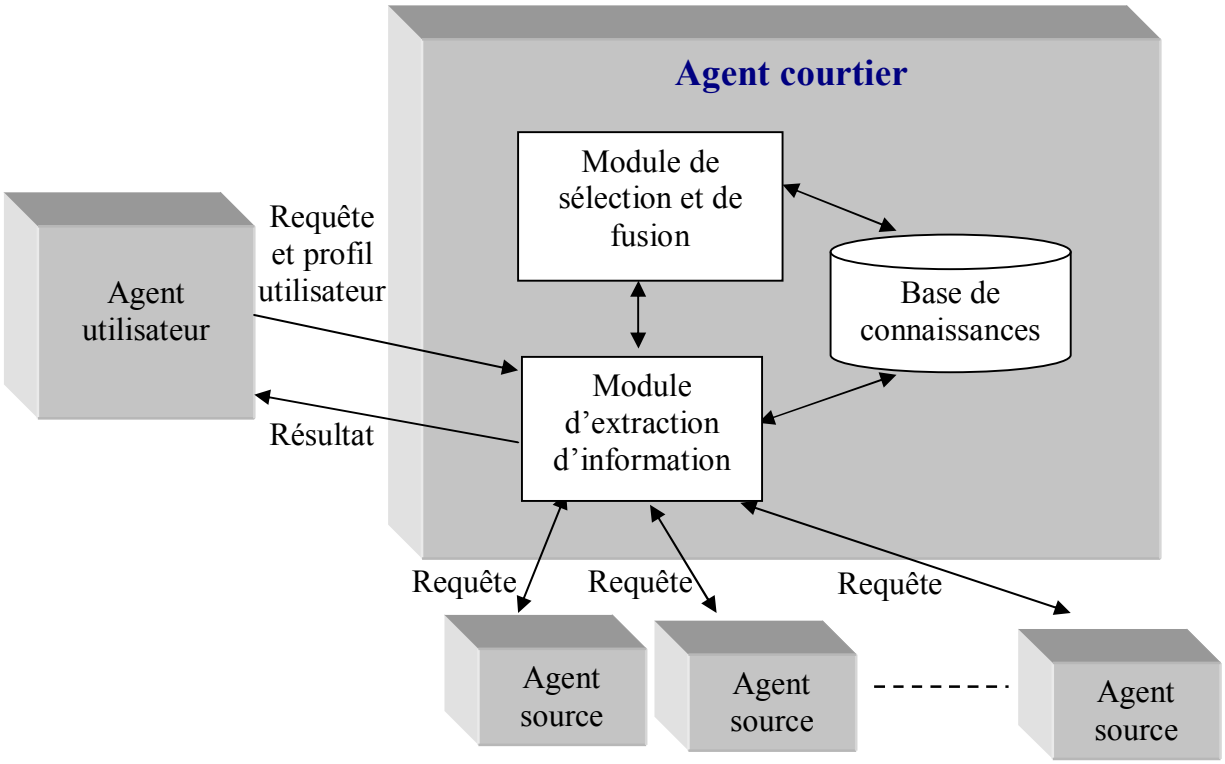


**Figure 6.9.** Schéma descriptif de l'agent courtier

*La base de connaissances* contient la liste des agents sources du système et les *Règles d'inférences* (les différentes formules que nous avons développées afin d'accomplir le processus de sélection de sources et celui de fusion des résultats des sources sélectionnées).

Les deux modules, *module d'extraction d'information* et *module de sélection et de fusion,* assurent le fonctionnement (rôle) de l'agent courtier en exploitant sa *base de connaissances*. Les différentes tâches de chaque modules sont décrites comme suit :

- ***Module d'extraction d'information*** : Il assure l'extraction d'information des sources via les agents sources pour une requête donnée ; (1) Il réceptionne la requête de l'utilisateur et son profil via l'agent utilisateur ; (2) Il envoie cette requête aux agents sources du système, afin de recevoir de ces derniers les listes des résultats; (3) Il envoie ces listes des résultats avec le profil de l'utilisateur au module de sélection et de fusion, afin de recevoir de ce dernier la liste finale des résultats; (4) Il envoie cette liste finale à l'agent utilisateur.

- ***Module de sélection et de fusion*** : Il assure la sélection des sources pertinentes et la fusion des listes des résultats retournées par ces sources ; (1) Il réceptionne les listes des résultats des sources et le profil de l'utilisateur via le module d'extraction d'information ; (2) Il sélectionne sur la base de ces listes les sources pertinentes en exploitant les règles d'inférences correspondantes ; (3) Il fusionne les listes des sources sélectionnées en exploitant les règles d'inférences correspondantes ; (4) Il envoie la liste finale des résultats au module d'extraction d'information.

### *4.1.4. Les interactions entre les agents*

Dans cette section nous décrivons le flux de données et de communication entre les différents agents.

**Utilisateur → Agent utilisateur :**

(1) Demander à l'agent utilisateur de s'inscrire (ou de se connecter).
(2) Introduire sa requête.
(3) Ajouter ses documents favoris.

**Agent utilisateur → Agent courtier :**

- Soumission de la requête et le profil de l'utilisateur.

**Agent courtier → Agent source :**

- Soumission de la requête et le profil de l'utilisateur.

**Agent source → Agent courtier :**

- Soumission du profil de la source.

**Agent courtier → Agent utilisateur :**

- Soumission de la liste finale des résultats.

**Agent utilisateur → utilisateur :**

- Soumission de la liste finale des résultats.

### *4.1.5. Communication entre agents*

Les agents communiquent entre eux par l'envoi de messages. Les messages utilisés diffèrent d'une plate-forme d'agents (langage de communication) à une autre. En général un message échangé entre deux agents doit contenir :

1- *L'émetteur :* désigne l'agent qui a soumis le message.
2- *Le récepteur :* désigne l'agent qui doit recevoir le message.
3- *Le contenu :* désigne le contenu du message.

Il existe plusieurs langages de communication d'agents. Sous la Plateforme JADE[2], un message est créé sous la syntaxe suivante:

```
ACLMessage msg = new ACLMessage();
msg.setSender("nom de l'agent émetteur");
msg.addReceiver("nom de l'agent récepteur"));
msg.setContent("le contenu du message");
send(msg);
```

## 4.2. Un système à base de rôles

Dans cette section nous présentons l'architecture des agents dans un système à base de rôles. Ce système est plus facile à implémenter, car on ne tient pas compte de l'aspect cognitif, entre autre, le raisonnement de l'agent. On s'intéresse beaucoup plus aux différentes tâches que l'agent doit assurer.

### *4.2.1. Agent source*

L'agent source contient le profil de la source. Il assure les tâches suivantes : (1) la réception de la requête provenant de l'agent courtier; (2) l'extraction du résultat de la source ; (3) l'apprentissage du profil de la source ; (4) l'envoi du profil de la source relatif à la requête vers l'agent courtier. Les attributs et le rôle d'un agent source sont résumés dans la figure suivante :

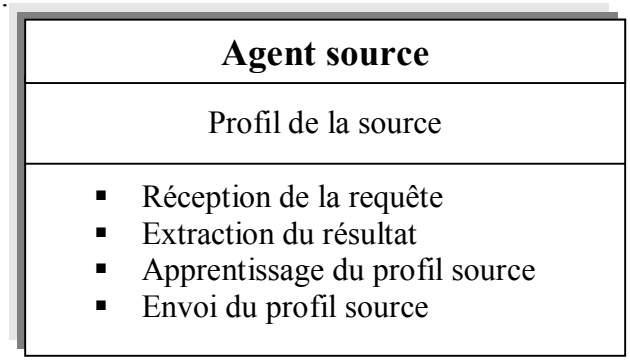


**Figure 6.10.** Attributs et rôle d'un agent source

#### *4.2.1.1. Attributs de l'agent source*

L'agent source contient le profil de la source qui lui est associée. Le profil de la source comme déjà présenté dans le chapitre précédent est composé des attributs suivants :

[2] Plate-forme JADE: Java Agent Development : http://jade.tilab.com

- *Identité :* est un attribut qui permet d'identifier la source, nous pouvons utiliser l'*url* et le *nom*,

- *Contenu :* est un attribut qui donne une vue du contenu de l'ensemble des documents dans la source.

- *Critères-documents :* est un multi-attribut qui nous informe sur la nature des documents de la source, à savoir, les types des documents, les langues des documents, la fraîcheur des documents et le coût des documents.

#### *4.2.1.2. Le rôle de l'agent source*

L'agent source s'occupe de l'instanciation et de la mise à jour du profil de la source après la réception de la requête de l'utilisateur. Ce profil sera envoyé à l'agent courtier.

- ***Instanciation du profil source :*** L'étape de l'instanciation du profil source consiste à initialiser ces différents attributs :

  - *Identité :* l'url et le nom d'une source sont fixés au moment de l'implémentation du prototype.

  - *Contenu :* le contenu d'une source est représenté par un vecteur (ensemble) de termes pondérés, en utilisant le modèle vectoriel [Salton et al 1975], qui représente la source comme un gigantesque document. Nous utilisons le schéma *tf*idf* pour la pondération des termes. Ces termes sont extraits des *k* premiers documents retournés par la source pour la requête de l'utilisateur.

  - *Critères-documents :* est un multi-attribut qui permet de mesurer l'importance de chaque critère dans la source. Chaque critère possède un ensemble de valeurs, pour chaque valeur, nous associons un poids. Le poids d'une valeur d'un critère, mesure son importance dans la source.

    L'agent exploite les critères suivants :

    - *Format :* ce critère indique les formats des documents dans la source, ses valeurs sont : *PDF, HTML, DOC, PS, XLS, PPT.*

    - *Langue :* ce critère indique les langues des documents dans la source, ses valeurs sont : *Français, Anglais.*

    - *Fraîcheur :* ce critère indique la fréquence de mise à jour des documents de la source, ses valeurs sont : *Fraîche, Non_fraîche.* Le poids de la valeur *Fraîche* mesure le taux (une estimation) des documents d'actualités. Cependant, le

poids de la valeur *Non_Fraîche* mesure le taux des documents qui ne sont pas d'actualités.

- *Coût :* ce critère mesure le taux des documents payants et celui des documents gratuits, défini respectivement par les deux valeurs : *Payant, Non_payant.*

Les notations suivantes sont définies pour exprimer les *critères-documents* d'une source *:*

*f* est l'ensemble des critères-documents d'une source,
soit *f={Format, Langue, Fraîcheur, Coût }*

*f(i)* est le $i^{ème}$ critère d'une source, $f(i) \in f$

*vf(i)* est l'ensemble des valeurs du $i^{ème}$ critère

*vf(1)*= {PDF, HTML, DOC, PS, XLS, PPT}
*vf(2)*= { Français, Anglais }
*vf(3)*= {Fraîche, Non_ Fraîche}
*vf(4)*= { Payant, Non_payant}

*vf(i,j)* est la $j^{ème}$ valeur du $i^{ème}$ critère: $vf(i, j) \in vf(i)$

$p\_vf^{(s)}(i,j)$ est le poids associé à la $j^{ème}$ valeur du $i^{ème}$ critère d'une source *s.*

$p\_vf^{(s)}(i,j)$ est un poids qui représente l'importance de la $j^{ème}$ valeur du $i^{ème}$ critère d'une source *s.*

Ce poids est calculé comme suit :

Concernant les critères *Format, Langue, Coût,* le poids $p\_vf^{(s)}(i,j)$ représente un pourcentage des documents ayant la $j^{ème}$ valeur du $i^{ème}$ critère dans la source *s*. Ce poids est calculé par la formule suivante :

$$p\_vf^{(s)}(i,j) = \frac{\left|d^{(s)}(i,j)\right|}{\left|d^{(s)}\right|}$$

Avec $d^{(s)}(i,j)$ est l'ensemble des documents analysés ayant la $j^{ème}$ valeur du $i^{ème}$ critère dans la source *s*, $d^{(s)}$ est l'ensemble de tous les documents analysés de la source *s.*

Concernant le critère *Fraîcheur*, l'agent calcule les poids des valeurs *Fraîche* et *Non_Fraîche*, d'une manière qui lui permet de favoriser une source ayant des documents récents si l'utilisateur s'intéresse aux documents récents, et de favoriser les sources ayant les documents non récents dans le cas contraire, via cette formule :

$$p\_vf^{(s)}(Fraîcheur, j) = \begin{cases} \dfrac{\sum_{d=1}^{k} année_d^{(s)}}{k * annéeActue\ lle} & Si \quad j = Fraîche \\ 1 - \dfrac{\sum_{d=1}^{k} année_d^{(s)}}{k * annéeActue\ lle} & Si \quad j = Non\_fraîche \end{cases}$$

où *k* est le nombre de documents analysés de la source *s*, $année_d^{(s)}$ est l'année d'édition du document *d* d'une source *s*, *annéeActuelle* est l'année en cours.

- ***Apprentissage du profil source :*** Dans l'objectif de diminuer le temps de réponse du processus de recherche, nous avons décidé de ne pas mettre à jour le profil d'une source qu'à la demande de l'utilisateur ou lors d'une requête non similaire aux requêtes sur lesquelles le profil de la source est calculé. Le profil source est calculé pour une requête de l'utilisateur donnée, puis il sera utilisé pour les autres requêtes similaires à cette requête. Dans le cas où l'utilisateur demande la mise à jour du profil source, ou dans le cas d'une nouvelle requête exprimant un intérêt ou un domaine tout à fait différent de cette requête, les agents source refont les calculs du contenu des attributs des sources tels qu'ils sont définis ci-dessus, en utilisant la requête en cours de l'utilisateur dans les calculs.

  La similarité entre deux requêtes $q_i$ et $q_j$ est calculée par la formule de cosinus suivante :

$$Sim(q_i, q_j) = \frac{q_i \bullet q_j}{|q_i| \times |q_j|}$$

**Notes importantes :** Le contenu et les poids des critères du profil d'une source sont associés à une requête q, pour les autres requêtes le profil d'une source peut avoir d'autres contenus et d'autres poids des critères. Nous avons procédé de cette façon pour extraire le profil d'une source, car nous n'avons pas les moyens d'accéder à l'ensemble du contenu d'une source, qui peut être très vaste. La seule façon qui nous permet d'extraire le contenu d'une source est d'analyser la liste des documents qu'elle retourne pour une requête donnée. Cela montre bien qu'une source peut retourner des résultats différents pour des requêtes différentes, ce qui donne des profils différents à une source pour des requêtes différentes.

### *4.2.2. Un agent utilisateur*

L'agent utilisateur contient les profils des utilisateurs. Il assure les tâches suivantes : (1) la réception de la requête provenant de l'utilisateur; (2) l'extraction du profil courant de l'utilisateur ; (3) l'envoi de la requête vers l'agent courtier ; (4) la réception du résultat provenant de l'agent courtier ; (5) l'affichage du résultat pour l'utilisateur ; (6) l'apprentissage du profil de l'utilisateur. Les attributs et le rôle d'un agent utilisateur sont représentés par la figure suivante :

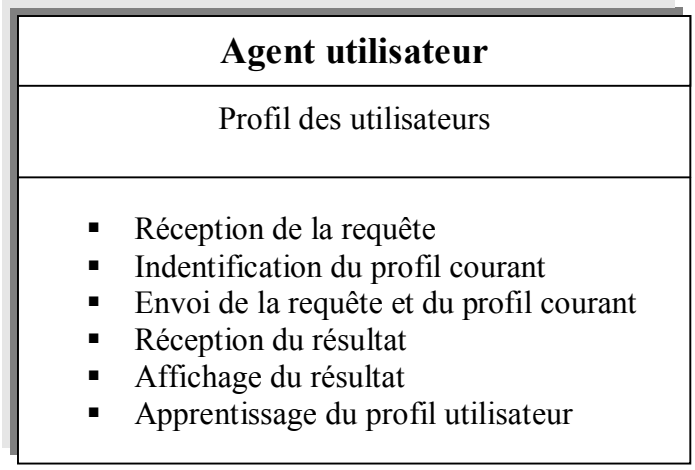


**Figure 6.11.** Attributs et rôle d'un agent utilisateur

#### *4.2.2.1. Attributs de l'agent utilisateur*

L'agent utilisateur contient les profils des utilisateurs. Le profil d'un utilisateur, comme déjà présenté dans le chapitre précédent, est composé des attributs suivants :

- *les données personnelles :* Cet attribut permet d'identifier l'utilisateur, nous utilisons : *code_utilisateur, mot de passe, nom, age,*

- *l'historique des recherches :* cet attribut permet d'identifier les centres d'intérêts et les préférences de l'utilisateur, il est constitué de l'ensemble de documents pertinents consultés par l'utilisateur durant ces recherches,

- *les centres d'intérêts de l'utilisateur :* les centres d'intérêts de l'utilisateur sont représentés par un ensemble de vecteurs de termes pondérés selon le schéma d'indexation *tf*idf* du modèle vectoriel. Chaque vecteur de termes définit un centre d'intérêts de l'utilisateur. L'ensemble des documents de l'historique des recherches relatif forme la collection de calcul.

- *les préférences :* c'est un multi-attribut permettant d'estimer les différentes préférences de l'utilisateur sur les critères d'une source, nous utilisons alors les

préférences sur les types des documents, les langues des documents, la fraîcheur des documents, le coût des documents.

La description des préférences de l'utilisateur doit être compatible avec celle des critères des sources, pour pouvoir les apparier. Nous utilisons alors les notations suivantes pour exprimer les *préférences* de l'utilisateur *:*

*p* est l'ensemble des préférences de l'utilisateur sur les critères des sources,

$p = \{Format, Langue, Fraîcheur, Coût\}$

*p(i)* est la *préférence* de l'utilisateur relative au $i^{ème}$ critère des sources,

$p(i) \in p$

*vp(i)* est l'ensemble des valeurs de la $i^{ème}$ préférence

*vp(1)*= {PDF, HTML, DOC, PS, XLS, PPT}
*vp(2)*= { Français, Anglais }
*vp(3)*= {Fraîche, Non_ Fraîche}
*vp(4)*= { Payant, Non_payant}

*vp(i,j)* est la $j^{ème}$ valeur de la $i^{ème}$ préférence : $vp(i, j) \in vp(i)$

$p\_vp^{(u)}(i,j)$ est le poids associé au $j^{ème}$ valeur de la $i^{ème}$ préférence de l'utilisateur *u.*

$p\_vp^{(u)}(i,j)$ est un poids qui représente l'importance de la $j^{ème}$ valeur de la $i^{ème}$ préférence pour l'utilisateur *u.*

#### *4.2.2.2. Le rôle de l'agent utilisateur*

L'agent utilisateur assure les tâches suivantes :

1- L'instanciation et l'apprentissage des profils des utilisateurs.

2- La sélection du profil de l'utilisateur, après réception de la requête de l'utilisateur.

3- L'envoi de la requête et le profil de l'utilisateur vers l'agent coutier et la réception des résultats de recherche via l'agent courtier, et leur affichage à l'utilisateur.

- ***Instanciation du profil de l'utilisateur :*** L'étape de l'instanciation du profil utilisateur consiste à initialiser les différents attributs qui le constituent :

- *les données personnelles :* le code de l'utilisateur, le mot de passe, le nom et l'âge sont introduits par l'utilisateur,

- *l'historique des recherches :* initialement vide,

- *le centre d'intérêts :* initialement vide,

- *les préférences :* initialement les poids de toutes les valeurs des préférences de l'utilisateur sont équitablement initialisés à 1.

- ***Apprentissage du profil de l'utilisateur :*** L'apprentissage du profil utilisateur est effectué après chaque session de recherche de l'utilisateur. L'agent utilisateur sauvegarde l'ensemble des documents pertinents pour l'utilisateur parmi les documents qu'il a retournés à l'utilisateur. Cet ensemble est utilisé pour la mise à jour du profil utilisateur :

  - *Les données personnelles :* ces données ne sont pas mises à jour, sauf le mot de passe, qui est modifié par l'utilisateur s'il-le désire.

  - *L'historique des recherches :* c'est un ensemble de documents pertinents pour l'utilisateur constitué durant ces recherches précédentes. Cet ensemble de documents permet de spécifier les centres d'intérêts de l'utilisateur. A partir de cet ensemble de documents, le courtier peut déduire les centres d'intérêts de l'utilisateur et ses préférences.

  - *Les centres d'intérêts :* un centre d'intérêts de l'utilisateur est un ensemble de vecteurs de termes pondérés extraits de l'historique des recherches, utilisant la technique d'indexation *tf*idf* du modèle vectoriel. Ces vecteurs sont calculés et mis à jour itérativement comme suit :

    Soit $V^{(u)}$ l'ensemble des vecteurs des centres d'intérêts de l'utilisateur $u$. Initialement $V^{(u)}$ est vide ($|V^{(u)}|=0$). Après chaque session de recherche de l'utilisateur, l'ensemble $V^{(u)}$ est mis à jour comme suit :

    - Extraire de l'historique des recherches en cours le vecteur des termes pertinents, pondérés par le schéma *tf*idf*. Soit ce vecteur $v_i^{(u)}$

    - Si $|V^{(u)}| = 0$ alors insérer le vecteur $v^{(u)}_i$ dans $V^{(u)}$.

    - Sinon, calculer le degré de similarité entre chaque vecteur $v_j^{(u)}$ dans $V^{(u)}$ et le vecteur $v_i^{(u)}$. Nous utilisons la formule suivante dans le calcul de cette similarité :

$$Sim(V_j^{(u)}, V_i^{(u)}) = \frac{V_j^{(u)} \bullet V_i^{(u)}}{\left|V_j^{(u)}\right| \times \left|V_i^{(u)}\right|} \qquad \forall\, j \in \left|V^{(u)}\right|$$

- S'il existe un vecteur $v_m^{(u)}$ ayant sa similarité avec le vecteur $v_i^{(u)}$ qui dépasse un seuil (défini lors de l'implémentation) alors :

  - fusionner les deux vecteurs $v_l^{(u)}$ et $v_m^{(u)}$ ayant la plus grande similarité. Nous notons le vecteur résultant $v_k^{(u)}$.

  - trier les termes du nouveau vecteur $v_k^{(u)}$ par ordre décroissant de leurs poids.

  - supprimer les termes du nouveau vecteur $v_k^{(u)}$ qui ont les poids les plus inférieurs, en gardant les *m* termes ayant les poids les plus élevés.

- Sinon insérer le vecteur $v_i^{(u)}$ dans $V^{(u)}$.

- *Les préférences :* l'acquisition des préférences de l'utilisateur consiste à recalculer les poids des valeurs de chaque préférence de l'utilisateur sur les critères des sources. Ces poids sont recalculés périodiquement à chaque session de recherche de l'utilisateur. Nous utilisons l'historique des recherches de l'utilisateur dans ce calcul comme suit :

  Concernant les préférences sur les critères *Format, Langue, Coût,* le poids $p\_vp^{(u)}(i,j)$ représente le pourcentage des documents ayant la $j^{ème}$ valeur de la $i^{ème}$ préférence dans l'historique des recherches de l'utilisateur *u.* Ce poids est calculé à l'aide de la formule suivante :

$$p\_vp^{(u)}(i,j) = \frac{\left|d^{(u)}(i,j)\right|}{\left|d^{(u)}\right|}$$

  où $d^{(u)}(i,j)$ est l'ensemble des documents ayant la $j^{ème}$ valeur de la $i^{ème}$ préférence dans l'historique des recherches de l'utilisateur *u*, $d^{(u)}$ est l'ensemble de tous les documents dans l'historique des recherches de l'utilisateur *u.*

  Concernant la préférence sur le critère *Fraîcheur*, l'agent calcule les poids des valeurs *Fraîche* et *Non_fraîche*, d'une manièe qui lui permet d'affecter le poids élevé à la valeur *Fraîche* si l'utilisateur s'intéresse aux documents récents, et d'affecter le poids élevé à la valeur *Non_frarîche* dans le cas contraire.

$$p\_vp^{(u)}(Fraîcheur, j) = \begin{cases} \dfrac{\sum_{d=1}^{k} année_d^{(u)}}{k * annéeActuelle} & Si \;\; j = fraîche \\ 1 - \dfrac{\sum_{d=1}^{k} année_d^{(u)}}{k * annéeActuelle} & Si \;\; j = non\_fraîche \end{cases}$$

où $k$ est le nombre de documents dans l'historique des recherches de l'utilisateur $u$, $année^{(u)}_d$ est l'année d'édition du document $d$ dans l'historique des recherches de l'utilisateur $u$, *annéeActuelle* est l'année en cours.

- ***Sélection du centre d'intérêts courant de l'utilisateur :*** Un utilisateur peut avoir différents centres d'intérêts. La première chose à faire après la réception de la requête de l'utilisateur, c'est de trouver parmi les centres d'intérêts, dans son profil, celui qui correspond à la requête en cours. Ce centre d'intérêts est appelé le centre d'intérêts courant. Dans le but de trouver le centre d'intérêts courant, le courtier calcule la similarité entre la requête de l'utilisateur et les centres d'intérêts existants dans le profil de l'utilisateur. Le centre d'intérêts ayant la similarité la plus élevée avec la requête, sera considéré comme centre d'intérêts courant. La similarité est calculée par la formule de cosinus comme suit :

$$Sim(q, V_i) = \frac{q \bullet V_i}{|q| \times |V_i|} \qquad \forall\, i \in \left|V^{(u)}\right|$$

Avec $q$ représentant la requête en cours de l'utilisateur, $v^{(u)}_i$ ~~est~~ le i$^{ème}$ centre d'intérêts de l'utilisateur, $V^{(u)}$ ~~est~~ l'ensemble des centres d'intérêts de l'utilisateur.

Le centre d'intérêts ainsi déterminé et noté $v^{(u)}_i$, sera utilisé dans la suite du processus d'évaluation de la requête.

### *4.2.3. Un agent courtier*

L'agent courtier contient la liste des agents sources. Il assure les tâches suivantes : (1) la réception de la requête et le profil de l'utilisateur provenant de l'agent utilisateur; (2) l'envoi de la requête aux différents agents source ; (3) la réception des profils des sources provenant des agents sources ; (4) la sélection des sources pertinentes ; (5) la fusion des résultats des sources sélectionnées ; (6) l'envoi du résultat final vers l'agent utilisateur. Les attributs et le rôle d'un agent utilisateur sont représentés par la figure suivante :

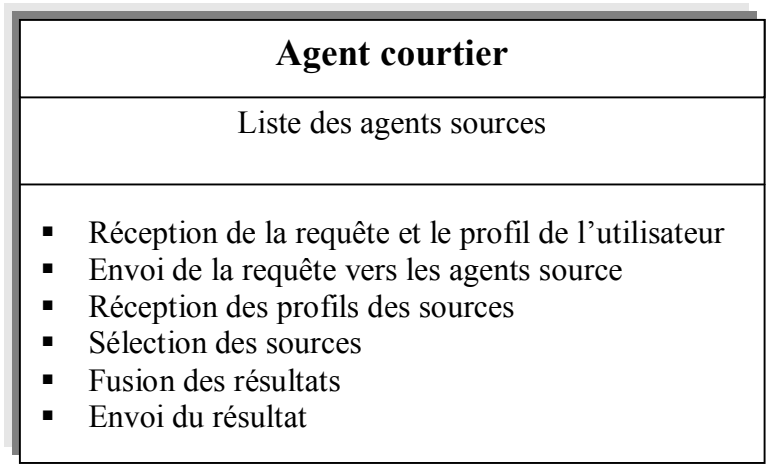


**Figure 6.12.** Attributs et rôle d'un agent courtier

Les deux processus de sélection et de fusion sont détaillés comme suit :

- ***Processus de sélection des sources :*** Le but du processus de sélection de sources est de réduire l'espace de recherche, sans diminuer la pertinence des résultats. L'approche comme mentionnée ci-dessus, consiste en l'intégration du centre d'intérêts et les préférences de l'utilisateur dans l'étape de sélection de sources. Pour cela l'agent calcule un score $score_u^{(s)}$ pour chaque source *s*, sur la base duquel les sources seront sélectionnées. Ce score combine trois mesures :

  - *La similarité de la source par rapport à la requête de l'utilisateur notée* $sim_q^{(s)}$ *:* cette similarité est calculée entre l'attribut *contenu* du profil de la source et la requête *q* de l'utilisateur. L'agent utilise la formule de cosinus du modèle vectoriel dans le calcul de cette similarité.

$$sim_q^{(s)} = \frac{\sum_{i=1}^{T} t_i * q_i}{\left(\sum_{i=1}^{T} t_i^2\right)^{1/2} \left(\sum_{i=1}^{T} q_i^2\right)^{1/2}}$$

    Avec : *S* est la source ; *q* est la requête ; $t_i$ est le poids du i$^{ème}$ terme dans la source (ensemble des *k* premiers documents retournés par la source); $q_i$ est le poids du i$^{ème}$ terme dans la requête ; *T* est le nombre de termes de la source utilisés. Le contenu d'une source est représenté par l'ensemble de ses *k* premiers documents retournés.

  - *La similarité d'une source par rapport au centre d'intérêts de l'utilisateur notée* $sim_u^{(s)}$ *:* cette similarité est calculée entre l'attribut *contenu* du profil de la source et le centre d'intérêts courant $v^{(u)}{}_i$ de l'utilisateur. L'agent utilise la formule de cosinus du modèle vectoriel dans le calcul de cette similarité, décrite ci-dessus, sauf qu'il remplace la requête par le centre d'intérêts courant $v^{(u)}{}_i$.

  - *Le degré d'exactitude (satisfaction) entre les critères de la source et les préférences de l'utilisateur noté* $acc_u^{(s)}$ *:* ce degré d'exactitude est calculé en utilisant les poids des valeurs des critères de la source *s* et les poids des valeurs des préférences de l'utilisateur *u* par la formule suivante :

$$acc_u^{(s)} = \frac{1}{|f|} \sum_{i=1}^{|f|} \left( \frac{2 * \sum_{j=1}^{|vf(i)|} p\_vf^{(s)}(i,j) * p\_vp^{(u)}(i,j)}{\sum_{j=1}^{|vf(i)|} p\_vf^{(s)}(i,j)^2 + \sum_{j=1}^{|vf(i)|} p\_vp^{(u)}(i,j)^2} \right)$$

Le score final $score_u^{(s)}$ de la source s est calculé par la combinaison des trois mesures $sim_u^{(s)}$, $sim_q^{(s)}$ et $acc_u^{(s)}$ comme suit :

$$score_u^{(s)} = \alpha * sim_q^{(s)} + (1-\alpha) * (\beta * sim_u^{(s)} + (1-\beta) * acc_u^{(s)})$$

avec $\alpha, \beta \in [0, 1]$.

Les sources sélectionnées sont celles ayant leur score final $score_u^{(s)}$ supérieur ou égal à la moyenne des scores finaux des sources.

- ***Processus de fusion des résultats :*** Le but de l'étape de fusion des résultats est de sélectionner et de trier les documents retournés par les différentes sources d'information sélectionnées dans une seule liste finale de résultats. Cette liste finale sera retournée à l'utilisateur. Dans le but d'améliorer la pertinence du processus de fusion des résultats, L'agent intègre le profil de l'utilisateur (centre d'intérêts et préférences) dans ce processus de fusion. Plus précisément, il calcule trois mesures pour chaque document *d* de chaque liste de résultats retournée par les sources sélectionnées :

  - premièrement, il calcule le degré de similarité entre le document *d* et la requête *q* de l'utilisateur *u,* noté $sim_q^{(d)}$,

  - deuxièmement, il calcule le degré de similarité entre le document *d* et le centre d'intérêts courant $V^{(u)}i$ de l'utilisateur *u,* noté $sim_u^{(d)}$,

  - troisièmement, il calcule le degré d'exactitude entre le document *d* et les préférences de l'utilisateur *u*, noté $acc_u^{(d)}$,

  Dans le calcul des deux degrés de similarités $sim_q^{(d)}$, $sim_u^{(d)}$, il utilise la formule de cosinus du modèle vectoriel décrite ci-dessus pour la source. Sauf qu'il remplace la source par le document.

  Dans le but de calculer le degré d'exactitude $acc_u^{(d)}$, il associe pour chaque document *d* les mêmes critères que nous avons présentés ci-dessus dans le profil des sources à savoir ; les *types, les langues, les coûts, et la fraîcheur* des documents.

  La description de ces critères pour un document est la même description utilisée pour le profil des sources. La différence se focalise dans le calcul des poids des valeurs de ces critères.

  Les notations suivantes sont utilisées pour exprimer ces critères de documents:

  *fd* est l'ensemble des critères des documents,
  *fd = {Format, Langue, Fraîcheur, Coût}*

*fd(i)* est le $i^{ème}$ critère du document, $fd(i) \in fd$

*vfd(i)* est l'ensemble des valeurs du $i^{ème}$ critère

*vfd(1)*= {PDF, HTML, DOC, PS, XLS, PPT}
*vfd(2)*= { Français, Anglais }
*vfd(3)*= {Fraîche, Non_ Fraîche}
*vfd(4)*= { Payant, Non_payant}

*vfd(i,j)* est la j$^{ème}$ valeur du i$^{ème}$ critère: $vfd(i, j) \in vfd(i)$

$p\_vfd^{(d)}(i,j)$ est le poids associé à la j$^{ème}$ valeur du i$^{ème}$ critère du document *d.*

$p\_vfd^{(d)}(i,j)$ est un poids qui représente l'importance de la j$^{ème}$ valeur du i$^{ème}$ critère du document *d.*

Ce poids est calculé comme suit :

- Pour les deux critères *Format*, *Langue*, le poids $p\_vfd^{(d)}(i,j)$ est calculé comme suit :

$$p\_vfd^{(d)}(i,j) = \begin{cases} 1 \text{ si la } j^{ème} \text{ valeur du } i^{ème} \text{ critère existe dans le document } d \\ 0 \text{ sinon} \end{cases}$$

Avec $f(i) \in \{Format, Langue\}$

- Pour le critère fraîcheur, le poids $p\_vfd^{(d)}(i,j)$ est calculé comme suit :

$$p\_vfp^{(d)}(Fraîcheur, j) = \begin{cases} \dfrac{année^{(d)}}{annéeActuelle} & Si \quad j = fraîche \\ 1 - \dfrac{année^{(d)}}{annéeActuelle} & Si \quad j = non\_fraîche \end{cases}$$

Avec $année^{(d)}$ est l'année d'édition du document *d*, *annéActuelle* est l'année en cours.

- Pour le critère coût, le poids $p\_vfd^{(d)}(coût, j)$ est calculé comme suit :

$$p\_vfp^{(d)}(coût, j) = \begin{cases} \dfrac{prix^{(d)}}{prixMax} & Si \quad j = Payant \\ 1 - \dfrac{prix^{(d)}}{prixMax} & Si \quad j = Non\_payant \end{cases}$$

Avec $prix^{(d)}$ est le prix du document *d*, et *prixMax* est le maximum des prix de tous les documents dans les résultats.

Ainsi, le degré d'exactitude $acc_u^{(d)}$, est calculé en utilisant les poids des valeurs des critères des documents et les poids des préférences de l'utilisateur, suivant la formule suivante :

$$acc_u^{(d)} = \frac{1}{|f|}\sum_{i=1}^{|f|}\left(\frac{2*\sum_{j=1}^{|vf(i)|} p\_vfd^{(d)}(i,j)*p\_vp^{(u)}(i,j)}{\sum_{j=1}^{|vf(i)|} p\_vfd^{(d)}(i,j)^2+\sum_{j=1}^{|vf(i)|} p\_vp^{(u)}(i,j)^2}\right)$$

Le score final de chaque document *d* noté $score_q^{(d)}$ est calculé par combinaison des ses trois mesures $sim_q^{(d)}$, $sim_u^{(d)}$ et $acc_u^{(d)}$ suivant la formule suivante :

$$score_u^{(d)} = \alpha * sim_q^{(d)} + (1-\alpha)*(\beta * sim_u^{(d)} + (1-\beta)*acc_u^{(d)})$$

Avec $\alpha$ , $\beta \in [0, 1]$.

Les documents sont finalement fusionnés et triés dans une seule liste par ordre décroissant par rapport à leur score final $score_u^{(d)}$. Cette liste sera retournée à l'utilisateur.

# 5. Discussion et comparaison

Dans cette section nous présentons une étude comparative entre l'approche à base d'agents et l'approche séquentielle. Nous nous intéressons dans notre comparaison à trois mesures à savoir ; la pertinence des résultats, le temps de réponse du processus de recherche et l'extensibilité du système.

1- *La pertinence des résultats :* Les deux approches ne montrent pas une grande différence concernant la pertinence des résultats, car la stratégie de recherche est la même.

2- *Le temps de réponse :* L'approche classique nécessite un temps de réponse important, car le processus de recherche s'occupe du traitement des informations provenant des différentes sources d'informations. Cependant, l'approche à base d'agents, nécessite un temps de réponse inférieur, car le traitement des informations provenant des différentes sources est réparti sur plusieurs agents qui s'exécutent en parallèle. Chaque agent assure le traitement d'une source seulement.

3- *L'extensibilité du système :* L'approche séquentielle nécessite une importante modification du code source lors de l'ajout d'une nouvelle source d'information au

système. Cependant, l'approche à base d'agents améliore l'extensibilité du système, car chaque agent est programmé pour une source donnée, donc l'ajout d'une nouvelle source peut être assuré sans la modification du code source des autres agents, il suffit de programmer un nouvel agent qui assure le traitement des informations de cette source, l'agent pourra par la suite interagir avec les autres agents dans le système.

Outre ces mesures, nous pouvons aussi citer l'avantage que possède l'approche à base d'agents par rapport à l'autre approche lors de l'implémentation du système. L'approche séquentielle doit être toute implémentée par le même programmeur, ce qui nécessite un temps de mise en œuvre important avec une importante charge du travail. Cependant, l'approche à base d'agents peut être implémentée par plusieurs programmeurs, dont chacun s'occupe de l'implémentation d'un agent, ils peuvent aussi travailler en parallèle, ce qui améliore le temps de mise en œuvre et diminue la surcharge du travail sur les programmeurs.

# 6. Conclusion

Dans ce chapitre nous avons présenté une approche d'adaptation d'une architecture multi-agents pour l'intégration du modèle utilisateur dans un accès personnalisé à de multiples sources d'information. L'approche représente une modélisation à base d'agents de notre approche précédente (l'approche séquentielle). Nous avons défini trois types d'agents à savoir ; un agent utilisateur ; un agent courtier ; et plusieurs agents source. Nous avons défini ensuite le contenu, le rôle et l'architecture de chaque agent et les interactions entre eux afin d'accomplir le but final, qui consiste à l'intégration du modèle utilisateur dans le processus de sélection de sources pertinentes et dans le processus de fusion des résultats de ces sources sélectionnées.

Le but de cette approche de modélisation à base d'agents est de remédier aux insuffisances rencontrées dans l'approche séquentielle concernant le temps de réponse et l'extensibilité du système. La modélisation de l'approche à base d'agents améliore le temps de réponse, car le traitement des informations provenant des différentes sources est réparti sur plusieurs agents qui s'exécutent en parallèle. Chaque agent assure le traitement d'une source seulement, ce qui permet d'accélérer le traitement. L'approche à base d'agents améliore aussi l'extensibilité du système, car chaque agent est programmé pour une source donnée, donc l'ajout d'une nouvelle source peut être assuré sans la modification du code source des autres agents, il suffit de programmer un nouvel agent qui assure le traitement des informations de cette source, l'agent pourra par la suite interagir avec les autres agents du système.

# Troisième partie

# Chapitre 7 : Validation expérimentale

# Les prototypes mis en œuvre

Nous présentons dans cette partie les trois prototypes développés dans notre travail de thèse

## 1. Premier prototype

### 1.1. Introduction

Afin d'évaluer l'approche proposée au chapitre 4, nous avons développé un prototype. Dans ce prototype nous avons considéré les 08 moteurs de recherche suivants : GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chaque moteur de recherche représente une source d'information.

### 1.2. Description du langage de développement utilisé

Nous avons choisi le langage PHP (abréviation de PHP Hypertext Preprocessor – Préprocesseur Hypertexte PHP) qui est un langage de script côté serveur.

La syntaxe de PHP s'inspire largement du langage C, tout en présentant certains traits de parenté avec le langage Perl et Java. PHP est un langage multi plate-forme. Il a été porté sur de nombreuses stations UNIX, telles que LINUX, où il fonctionne aussi bien que sur des machines Windows.

Les raisons qui nous ont amené à choisir le langage de script PHP sont :

- La richesse de ce langage: il couvre pratiquement tous les domaines en rapport avec les applications Web,

- C'est un langage o*pen source*: le programmeur peut accéder au source et apporter des modifications. Il possède un contrôle complet et une personnalisation instantanée,

- La connexion et l'interrogation d'une base de données sont des tâches simples pouvant être accomplies en deux ou trois lignes de code,

- Le moteur de script de PHP est parfaitement optimisé pour les temps de réponses nécessaires à des applications web, et peut être intégré au serveur web lui-même pour améliorer encore plus les sorties,

- La simplicité et la robustesse,

- La connectivité avec un nombre toujours croissant de serveurs web,

- La brièveté des cycles de développement et la facilité de création de composants modulaires réutilisables, grâce à sa syntaxe et à sa construction,

### *1.2.1. Le serveur web Apache*

Pour que les pages PHP puissent être publiées sur Internet, il est tout d'abord nécessaire que PHP soit installé sur le serveur web de votre fournisseur d'accès à Internet (FAI). La première solution consiste à utiliser PHP sous forme de programme CGI. Cependant, la meilleure consiste, pour le FAI, à utiliser le serveur web Apache et à installer PHP comme module Apache.

## 1.3. Le SGBD utilisé

Afin d'implémenter les tables pour stocker les informations nécessaires, nous avons choisi le système de gestion de base de données relationnelle MySQL.

MySQL est basé sur une bibliothèque de gestion de données prouvée depuis de nombreuses années et faisant appel à des indexes d'arbres binaires. Grâce à cela, le cœur du système peut afficher une performance remarquable, tout particulièrement dans les accès indexés.

MySQL utilise une architecture multi-utilisateur, multi-traitement. Cela permet d'établir des connexions rapides et d'utiliser la même mémoire cache pour plusieurs requêtes.

## 1.4. Les services Web

Un service Web est un module logiciel effectuant une tâche discrète ou un ensemble de tâches, qui peut être localisé et appelé au travers d'un réseau, en particulier le World Wide Web. Le développeur peut créer une application client qui appelle une série de services Web par l'intermédiaire d'appels de procédures distantes (RPC, remote procedure calls) ou d'un service de messagerie pour fournir une partie, petite ou grande, de la logique de l'application. Un service Web publié se décrit lui-même pour que les développeurs puissent le localiser et évaluer son adaptation à leurs besoins.

Les services Web utilisent SOAP (Simple Object Access Protocol) et un protocole de transport comme HTTP pour échanger les messages SOAP. Ces messages sont en réalité des documents XML qui sont échangés entre un service Web et l'application appelante.

### *1.4.1. Les normes des services web*

Les normes sur lesquelles repose le développement de services Web sont des technologies en cours d'évolution. Les principaux acteurs sont les suivants :

- SOAP (Simple Object Access Protocol)
- WSDL (Web Services Description Language)
- UDDI (Universal Description, Discovery and Integration)
- WSIL (Web Services Inspection Language)
- JAX-RPC (Java API for XML-based Remote Procedure Call)
- WS-I (Web Services Interoperability)
- SAAJ (SOAP with Attachments API for Java)

Nous décrivons ci-après, la norme SOAP qu'on a utilisée dans notre prototype :

#### *1.4.1.1. Simple Object Access Protocol (SOAP)*

SOAP est un protocole de messagerie indépendant du transport. Chaque message SOAP est un document XML. SOAP utilise des messages unidirectionnels, bien qu'il soit possible de combiner des messages en des séquences requête-réponse. La spécification SOAP définit le format du message XML, mais pas son contenu ni la manière dont il est réellement envoyé. SOAP spécifie toutefois la manière dont les messages SOAP sont routés sur HTTP. Le diagramme suivant (figure 7.1) indique comment SOAP est employé pour l'envoi de messages entre le fournisseur, le demandeur et le courtier.

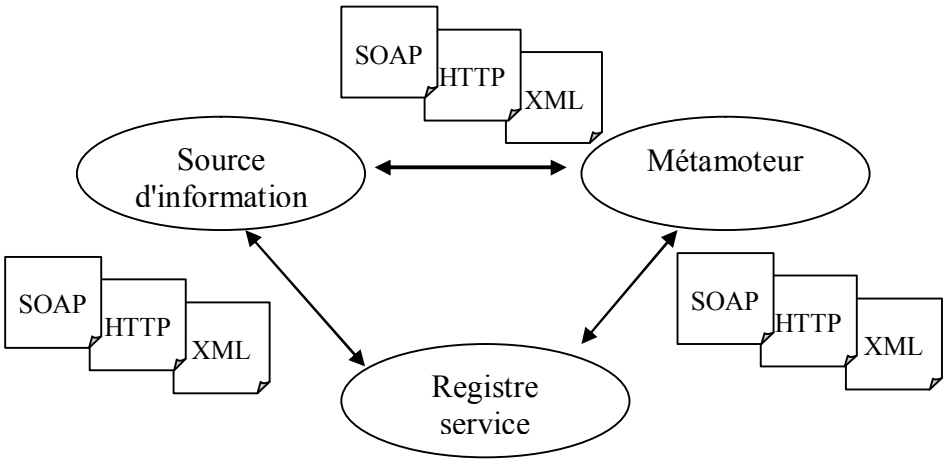


**Figure 7.1.** Architecture du protocole SOAP

Chaque document SOAP possède un *élément enveloppe racine*. L'*élément racine* est le premier élément d'un document XML, contient tous les autres éléments du document. Cette "enveloppe" se compose de deux parties ; un en-tête et un corps. L'en-tête contient les données de routage ou de contexte. Il peut être vide. Le corps contient le message réel. Il peut aussi être vide.

## 1.5. Les sources utilisées dans notre prototype

Nous avons exploité les moteurs de recherches suivants ; GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chaque moteur de recherche représente une source d'information. Notre prototype représente un méta-moteur de recherche personnalisée.

## 1.6. Quelques Interfaces du prototype

Nous présentons dans cette partie les principales fenêtres de notre prototype (méta moteur). Lorsque l'utilisateur saisit l'URL de notre prototype, la page qui lui est affichée est la suivante:

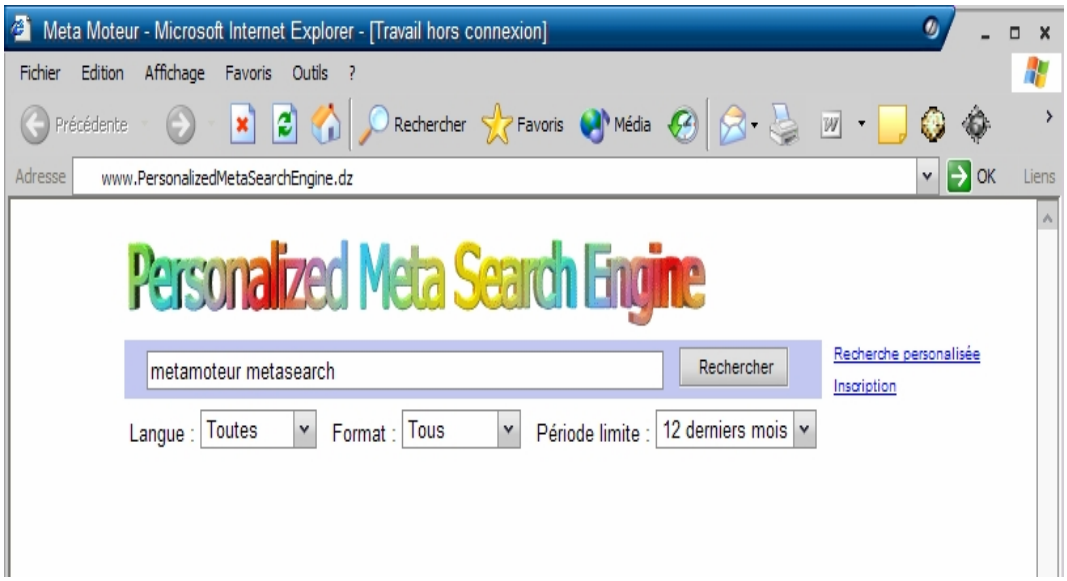


**Figure 7.2.** Fenêtre principale

Par défaut, la recherche est en mode non personnalisé. Si l'utilisateur veut effectuer une recherche personnalisée, il doit s'inscrire. En cliquant sur le lien *Inscription*, un formulaire d'inscription sera affiché à l'utilisateur. La figure suivante montre les détails de ce formulaire:

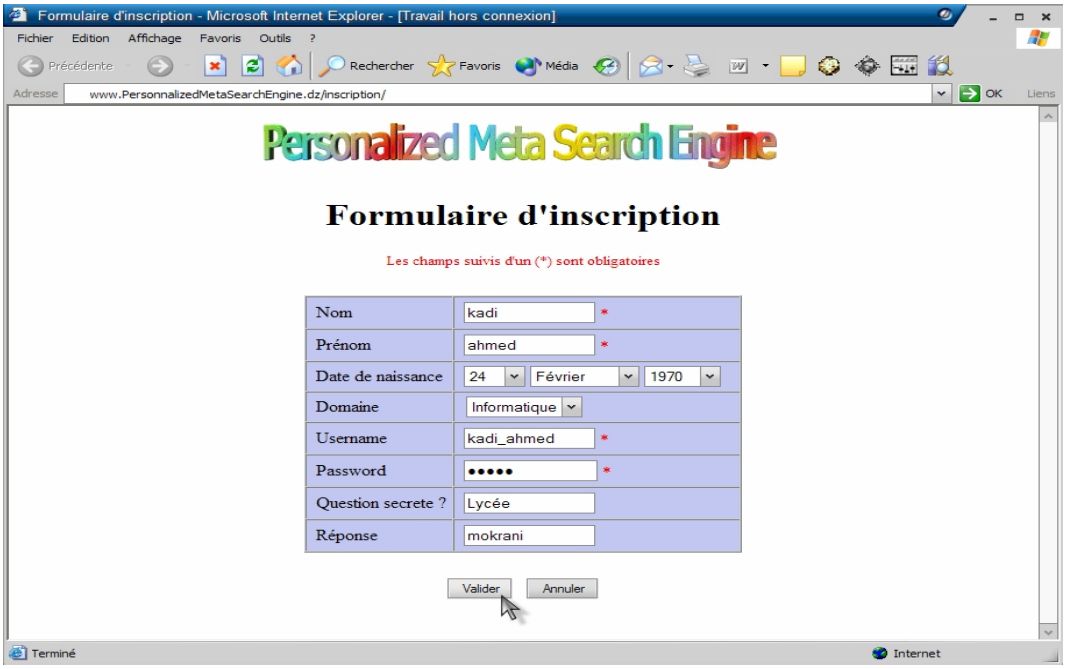


**Figure 7.3.** Fenêtre d'inscription

Une fois l'utilisateur est inscrit, il peut effectuer une recherche personnalisée après s'être identifié en cliquant sur le lien *Recherche personnalisée.* La figure ci-dessous illustre la fenêtre identification :

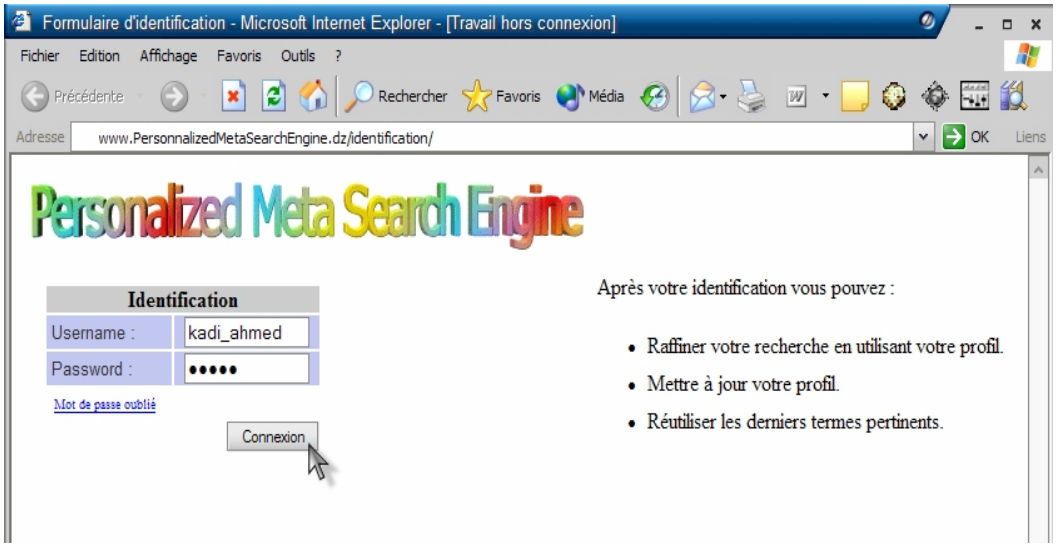


**Figure 7.4.** Fenêtre d'identification

L'utilisateur saisit sa requête et lance la recherche. Le méta-moteur lui retourne une liste triée de résultats comme le montre la figure suivante:

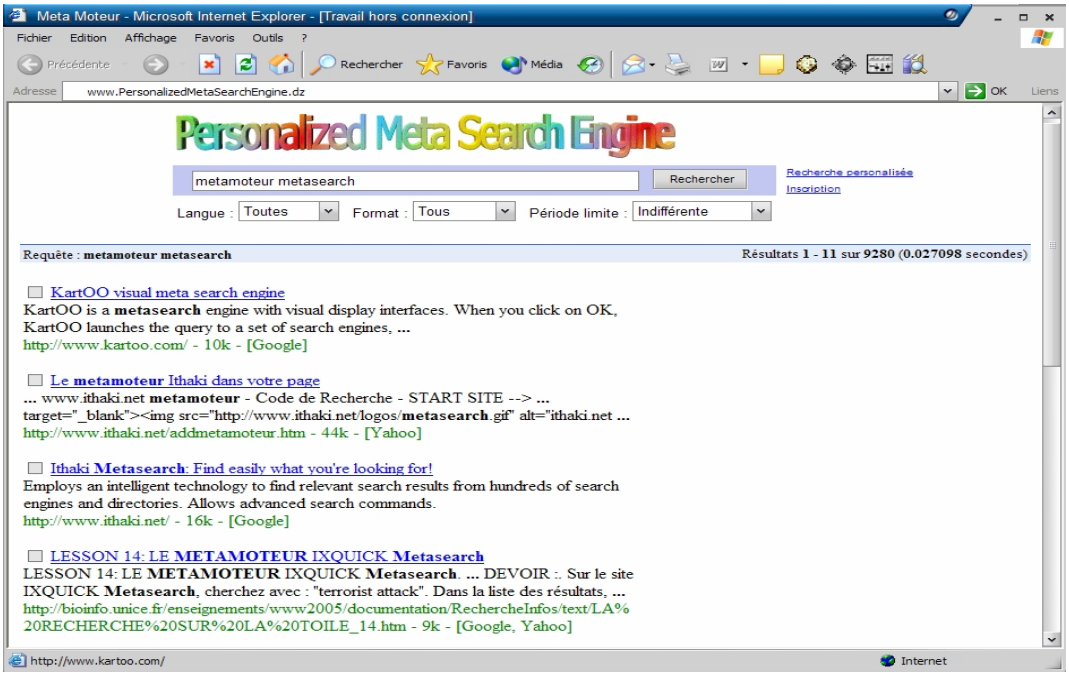


**Figure 7.5.** Fenêtre des résultats

Après consultation des résultats, l'utilisateur sélectionne les documents qu'il juge pertinents et relance la recherche en cliquant sur le bouton *Reformuler (figure 7.6)*. A partir des documents sélectionnés, le méta-moteur met à jour son profil, reformule la requête et l'envoie aux moteurs sélectionnés, fusionne leurs résultats et les affiche à l'utilisateur. L'utilisateur peut reformuler sa requête le nombre de fois qu'il veut.

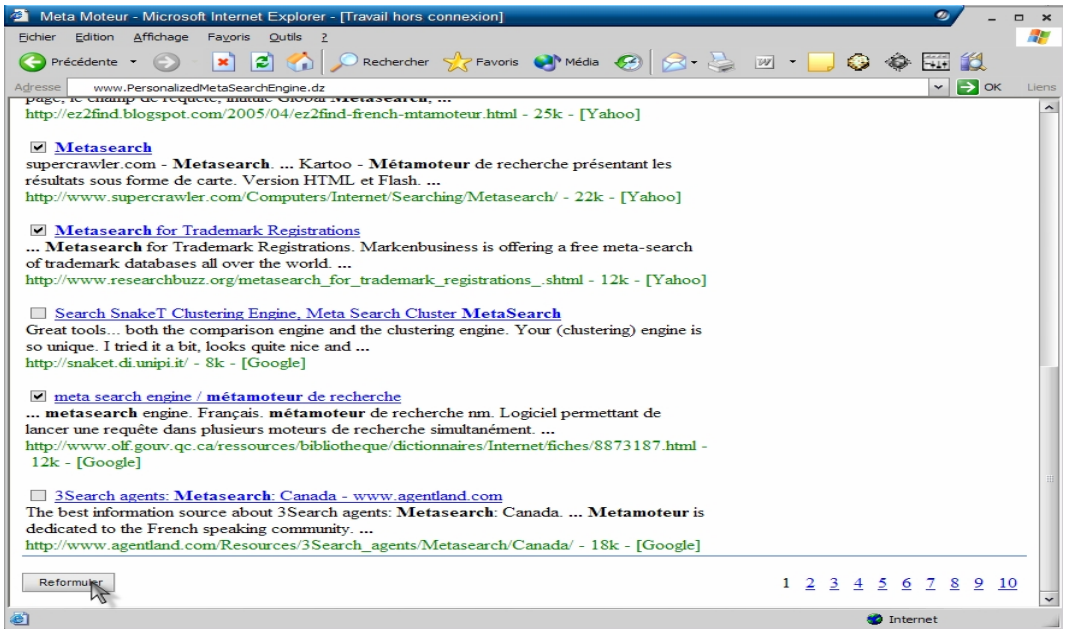


**Figure 7.6.** Fenêtre des résultats

## 1.7. Conclusion

L'approche a été évaluée en utilisant les 08 moteurs de recherches suivants : GOOGLE, YAHOO, ALTAVISTA, MSN, LYCOS, TEOMA, WISENUT, ALLTHEWEB. Chacun de ces moteurs de recherche présente une source d'information. L'approche est expérimentée sur 100 requêtes, par différentes façons ; (1) utilisation de toutes les sources (pas de sélection de sources) sans prendre en compte le profil utilisateur , (2) la sélection des sources sans l'utilisation du profil utilisateur, la sélection ici est faite sur la base de la requête utilisateur seulement , (3) prendre en compte le profil utilisateur en utilisant toutes les sources ; (4) la sélection des sources sur la base du profil utilisateur, qui est une des contributions proposée dans ce travail. Pour chaque cas nous avons calculé la précision moyenne des résultats. Les résultats obtenus sont probants et ont montré que l'intégration du profil utilisateur dans la recherche d'information distribuée, améliore la pertinence des résultats de recherche.

# 2. Deuxième prototype

Cette section présente l'implémentation du prototype relatif à l'approche proposée dans le chapitre 5 de notre contribution.

## 2.1. Introduction

L'approche est évaluée sur 100 requêtes, en utilisant plusieurs sources d'informations réparties sur différents secteurs en deux expérimentations. Le but de la première expérimentation est de connaître l'importance de ces mesures (requête, centre d'intérêts et préférences) dans la pertinence des résultats de recherche. Pour cela, nous avons évalué l'approche suivant 10 scénarios différents. Chaque scénario présente une importance meilleure d'une mesure par rapport aux autres. L'objectif de la deuxième expérimentation est d'évaluer et de comparer notre approche avec une approche qui utilise un algorithme de base de CORI de sélection de source et de fusion des résultats.

## 2.2. Environnement de développement

### *2.2.1. Le langage de programmation Java*

Pour la mise en œuvre de notre prototype, nous avons choisi le langage Java. En effet, Java est un langage de programmation orientée Objet : *(POO)* de très haut niveau capable de s'exécuter sur n'importe quelle machine. Java est axé sur la création d'objets (structures de données ou comportements) qui peuvent être répartis et manipulés par le programme. Ceci vient du fait que Java, à la différence de ***C++,*** est compilé en pseudo code et interprété par

une machine virtuelle ***(JVM )*** afin de faire le lien entre l'environnement de travail et ce dernier.

Parmi les caractéristiques fondamentales du langage qui nous ont amené à l'utiliser nous citons :

- Java fonctionne en mode interprété par opposition aux langages compilés et peut s'exécuter sur de nombreux systèmes d'exploitation.
- Java propose un mode de fonctionnement adapté aux applications réseaux et Internet.
- Java offre la possibilité de créer des applications multitâches de façon simple.

### *2.2.2. Le SGBD utilisé*

MySQL (My Structured Query Language) est un serveur de bases de données relationnelles SQL, très rapide, robuste et multi-utilisateurs. Cela permet d'établir des connexions rapides et d'utiliser la même mémoire cache pour plusieurs requêtes. MySQL est un logiciel libre développé sous licence GNU General Public License. MySQL peut fonctionner sur plusieurs plates-formes différentes.

### *2.2.3. Les sources utilisées dans le prototype*

Nous avons exploité plusieurs sources d'informations réparties sur des différents secteurs comme suit :

- ***Scientifique*** *:*
  (4) http://www.scirus.com/
  (5) http://www.in-extenso.org/
  (6) http://www.agrisalon.com/

- ***Informatique :***
  (4) http://www.lemondeinformatique.fr/
  (5) http://www.informatiques.eu
  (6) http://www.inria.fr/

- ***Economie :***
  (4) http://ese.rfe.org/
  (5) http://www.inomics.com/cgi/repec_search
  (6) http://www.wu-wien.ac.at/inst/vw1/zagler/search/

- ***Médecine :***
  (5) http://www.galenicom.com/
  (6) http://www.medvet.umontreal.ca/biblio/rech.htm
  (7) http://www.ncbi.nlm.nih.gov/pubmed/
  (8) http://www.genethique.org/moteur_recherche/recherche.asp

- ***Presse (news):***
  (5) http://c.asselin.free.fr/french/actua.htm
  (6) http://search.freefind.com/find.html?id=3225682
  (7) http://www.courrierinternational.com/gabarits/html/default_online.asp
  (8) http://www.excite.fr/search/news

## 2.3. Exemples d'interfaces du prototype

### *2.3.1. Interface du profil utilisateur*

Cette figure présente un exemple d'interface qui présente le contenu d'un profil utilisateur.

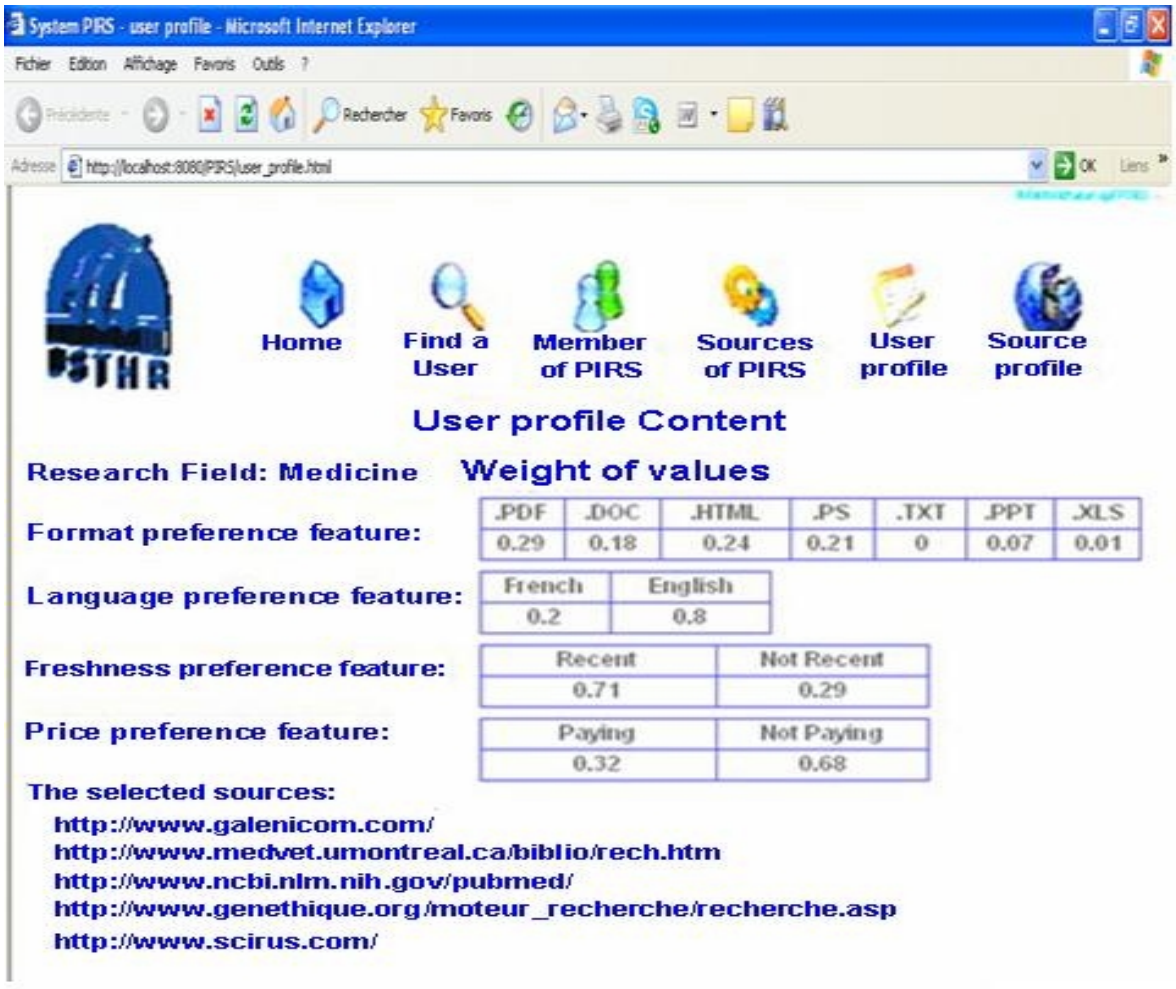


**Figure 7.7.** Profil utilisateur

### *2.3.2. Interface du profil source*

Cette figure présente un exemple d'interface qui présente le contenu d'un profil source.

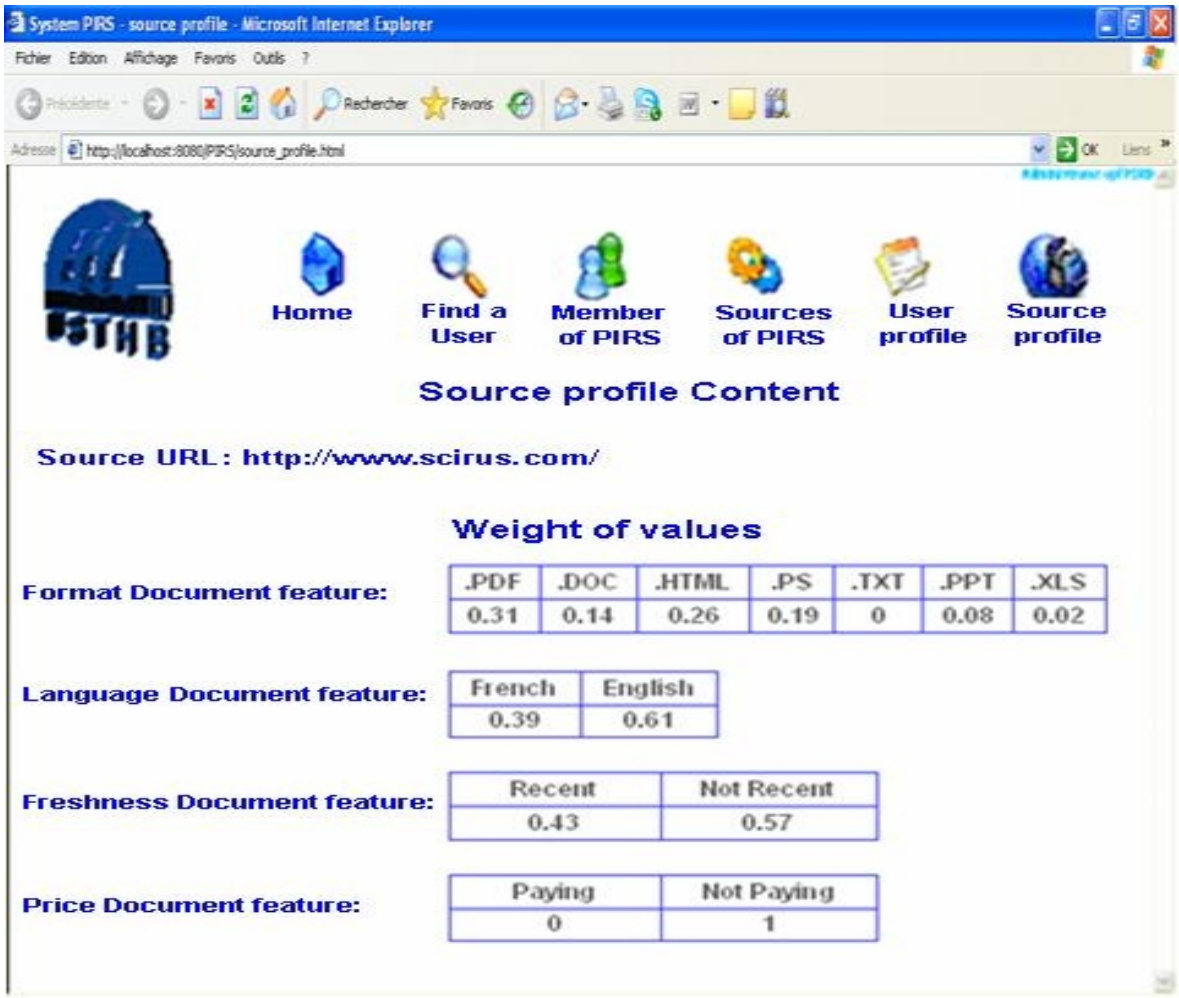


**Figure 7.8.** Profil source

## 2.4. Conclusion

Les résultats d'évaluation des différents cas montrent que l'intégration du profil utilisateur dans le processus de sélection et de fusion améliore les résultats de recherche. Ils montrent également que la requête de l'utilisateur est plus importante que son centre d'intérêts, et ce dernier est plus important que ses préférences.

Les résultats de la deuxième expérimentation obtenus nous confirment aussi que l'intégration de la personnalisation dans le processus de sélection de sources et dans le processus de fusion des résultats améliore la pertinence des résultats de recherche.

# 3. Troisième prototype

## 3.1. Introduction

Cette section présente l'implémentation du prototype relatif à l'approche proposée dans le chapitre 6 de notre contribution. L'intérêt d'utilisation d'un système multi-agents est d'améliorer le temps de réponse et l'extensibilité du système.

## 3.2. Environnement de développement

Il existe plusieurs plateformes pour la création d'agents par exemple ; ***JADE***, ***MACE***, ***ZEUS, MADKIT*** …etc. Il est possible aussi de créer les agents à l'aide de la notion de *Thread* en langage de programmation JAVA**.** Cette dernière permet la création des processus qui s'exécuteront en parallèle. Comme première implémentation de notre système d'agents, nous avons exploité la notion de thread en langage de programmation JAVA. Une autre implémentation à l'aide de la plateforme d'agents JADE est en cours de réalisation.

Un exemple de création des agents (processus) s'exécutant en parallèle à l'aide des threads en java est comme suit :

```
public static void AgentSource() {
  class connexion extends Thread {
        // Description des fonctions de l'agent source avec leurs paramètres d'entrées/sorties
    }
}
Thread T1 ;
```

Toutes les instances de la classe T1 à créer s'exécuteront en parallèle.

Dans la description des fonctions de l'agent, nous pouvons utiliser plusieurs paramètres. A la création d'une instance de la classe agent, on spécifie les valeurs de ces paramètres. Par exemple à la création d'une instance de l'agent source, on introduit le nom et l'url de la source vers laquelle l'agent doit se connecter et extraire les informations nécessaires.

## 3.3. Exemples d'interfaces du prototype

Nous présentons dans cette section quelques fenêtres de notre prototype.

### *3.3.1. La fenêtre principale*

C'est la page d'accueil. Dans cette fenêtre, même un utilisateur non inscrit peut effectuer des recherches.

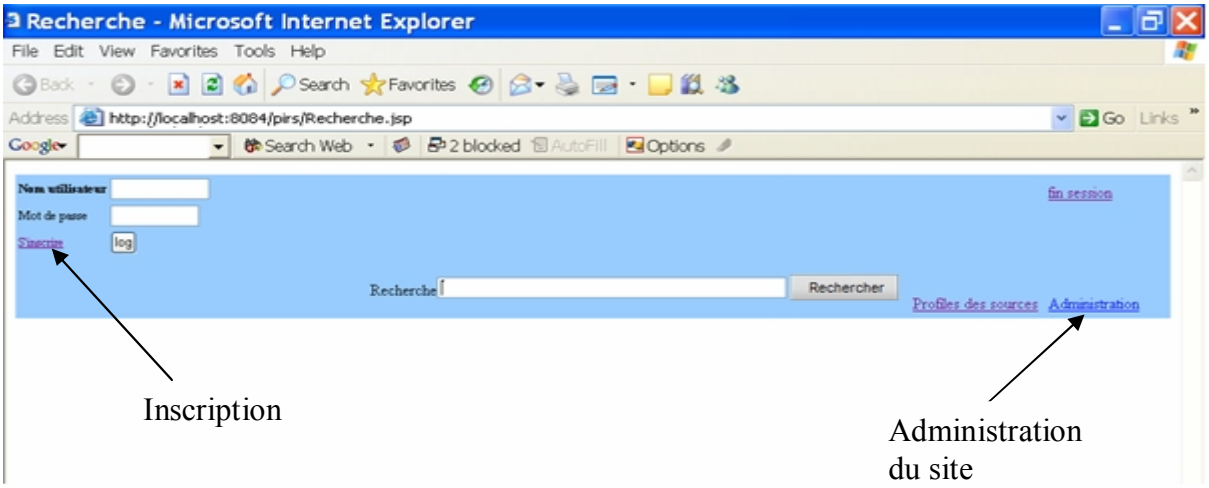


**Figure 7.9.** Interface principale

### *3.3.2. Fenêtre d'accueil d'un utilisateur inscrit*

Cette fenêtre permet à un utilisateur inscrit de voir toutes les dimensions de son profil. Elle lui permet également d'effectuer des recherches

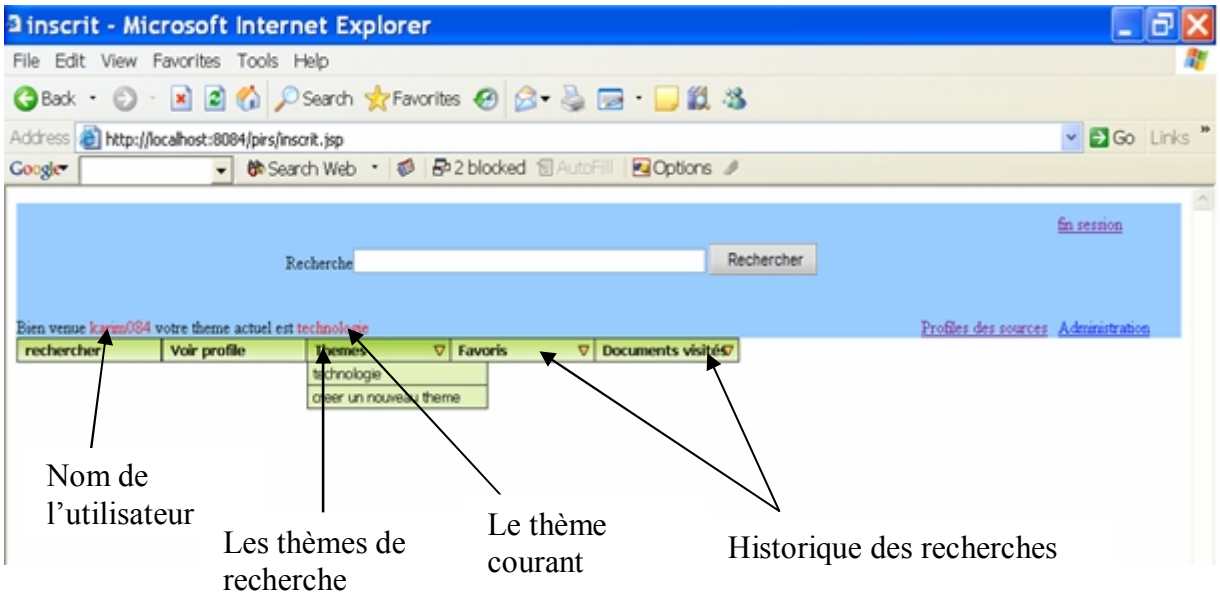


**Figure 7.10.** Fenêtre d'accueil d'un utilisateur inscrit

### *3.3.3. Fenêtre des résultats de recherche*

Cette fenêtre affiche les résultats de recherche suivant le centre d'intérêts et les préférences de l'utilisateur.

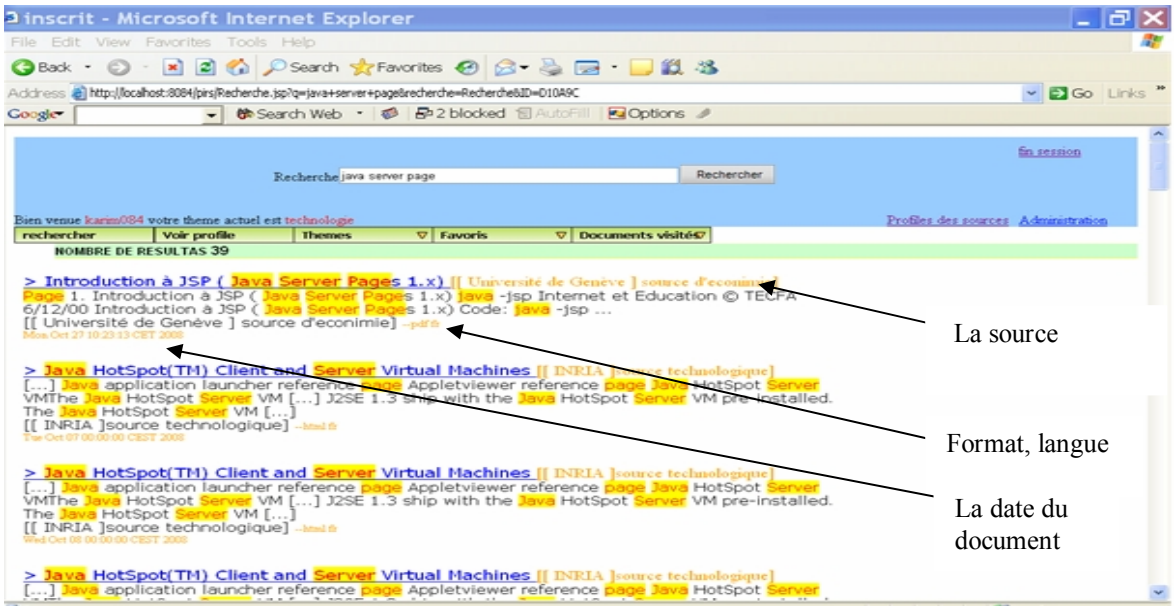


**Figure 7.11:** Résultats d'une recherche d'un utilisateur inscrit

### *3.3.4. Fenêtre d'un document visité par l'utilisateur*

Lorsque l'utilisateur consulte un document de la liste des résultats, la fenêtre d'affichage du document lui donne la possibilité de l'ajouter aux favoris si l'utilisateur le juge pertinent. Les documents ajoutés aux favoris forment son historique des recherches.

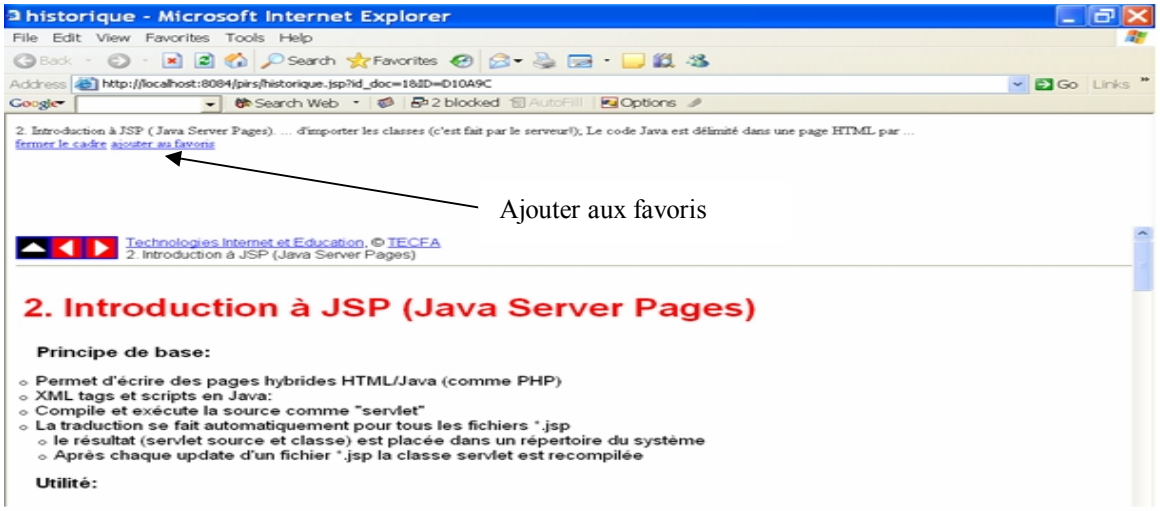


**Figure 7.12.** Fenêtre d'un document visité

### *3.3.5. Fenêtre de présentation des préférences d'un utilisateur inscrit*

Dans cette fenêtre l'utilisateur inscrit peut voir ses préférences comme la fraîcheur, coût, langues, types des documents.

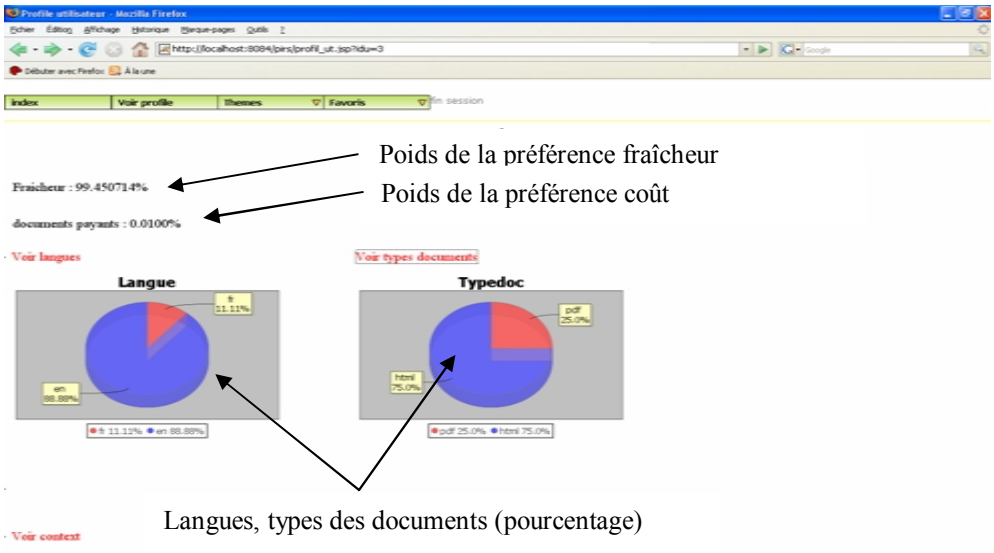


**Figure 7.13.** Profil utilisateur

### *3.3.6. Fenêtre de présentation des critères d'une source*

Cette fenêtre affiche un exemple des critères d'une source.

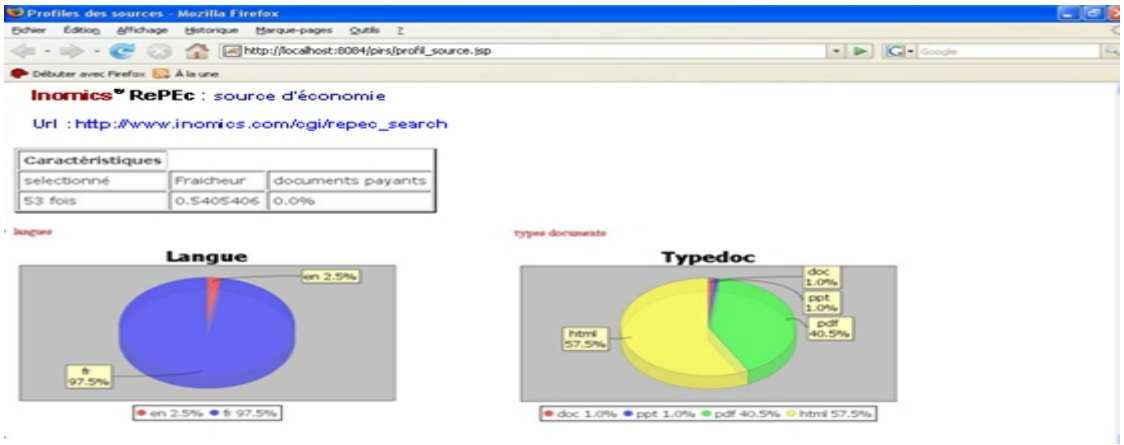


**Figure 7.14.** Fenêtre des critères d'une source

## 3.4. Conclusion

La modélisation de l'approche à base d'agents améliore le temps de réponse, car le traitement des informations provenant des différentes sources est réparti sur plusieurs agents qui s'exécutent en parallèle. Chaque agent assure le traitement d'une source seulement, ce qui permet d'accélérer le traitement. L'approche à base d'agents améliore aussi l'extensibilité du système, car chaque agent est programmé pour une source donnée, donc l'ajout d'une nouvelle source peut être assuré sans la modification du code source des autres agents, il suffit de programmer un nouvel agent qui assure le traitement des informations de cette source, l'agent pourra par la suite interagir avec les autres agents dans le système.

# Conclusion et perspectives

## 1. Conclusion

Nous avons abordé dans cette thèse le domaine de la recherche d'information distribuée (RID). Un système de recherche d'information distribuée (SRID) gère la recherche sur un ensemble de sources d'information distribuées. Un SRID dans sa configuration générale, se compose d'un courtier et d'un ensemble de sources d'information. A la réception d'une requête de l'utilisateur, le courtier choisit un sous-ensemble de sources pertinentes. Ce processus est appelé *sélection de sources*. Chaque source ainsi sélectionnée traite la requête et renvoie une liste de résultats au courtier. Ce dernier se charge alors de construire à partir des différentes listes reçues, une liste globale qui sera retournée à l'utilisateur. Ce processus est appelé *fusion des résultats.*

Ces deux processus *sélection de sources* et *fusion des résultats* sont les principaux problèmes auxquels on est confronté lorsque l'on veut concevoir un SRID. Les méthodes de sélection de sources et de fusion des résultats des sources développées dans la littérature sont effectuées sur la base des informations contenues dans les sources et la requête. L'utilisateur, ses préférences et ses centres d'intérêts récurrents (profil utilisateur) ne sont pas pris en compte.

Nous nous sommes intéressés dans cette thèse à l'intégration du profil de l'utilisateur dans le processus de sélection de sources d'information et le processus de fusion des résultats retournés par les sources sélectionnées. Bien entendu, ce processus permettra d'élaguer intelligemment des sources d'informations moins pertinentes pour l'utilisateur afin de réduire l'espace de recherche et donc la complexité de cette dernière. Nos contributions consistent à répondre aux questions suivantes : (1) Comment définir et acquérir le profil de l'utilisateur? (2) Comment intégrer ce profil pour personnaliser le processus de sélection de sources? (3) Comment intégrer ce profil pour personnaliser le processus de fusion des résultats des sources sélectionnées ?

Notre première contribution consiste d'abord à la définition d'un profil utilisateur constitué de plusieurs dimensions : (1) son identité ; (2) son historique des recherches ; (3) son centre d'intérêt ; (4) son historique des sources sélectionnées. Puis à la définition des techniques nécessaires permettant d'exploiter ce profil dans la personnalisation du processus de sélection de sources et du processus de fusion des résultats des sources sélectionnées.

Cependant, les résultats expérimentaux ont montré que l'approche présentait certaines insuffisances à savoir : (1) Un utilisateur peut avoir plusieurs centres d'intérêts différents,

pas seulement un seul; (2) L'utilisation du centre d'intérêt seul pour un utilisateur n'est pas suffisante pour décrire ses besoins en information, un utilisateur peut avoir des préférences sur la nature (critères) des documents qu'il veut consulter.

Dans la deuxième contribution, nous avons pallié à ces insuffisances en proposant une autre approche permettant de prendre en compte : (1) Les différents centres d'intérêts d'un utilisateur ; (2) Les préférences de l'utilisateur relatives aux critères des documents des sources, à savoir ; le format, la langue, la fraîcheur et le coût d'un document. Dans cette approche la pertinence des résultats est relative à trois mesures correspondantes à la requête de l'utilisateur, son centre d'intérêts et ses préférences. Cependant, l'approche présente des insuffisances concernant le temps de réponse et l'extensibilité du système.

Notre troisième contribution concerne le temps de réponse et l'extensibilité du système. Dans un souci d'amélioration du temps de réponse et d'extensibilité du système, nous avons également développé un système multi-agents pour l'intégration du modèle de l'utilisateur dans SRID.

Dans le contexte général de la RI, les solutions développées dans cette thèse permettent de réduire l'espace de recherche (réduire le nombre de sources) et donc la complexité de cette dernière, en augmentant la pertinence des résultats. En particulier, dans la RID, nos solutions consistent à personnaliser les processus de sélection de sources et de fusion des résultats, dans le but de rapprocher la pertinence système d'une source ou d'un document à leurs pertinences utilisateur.

## 2. Perspectives

Notre première perspective concerne les préférences de l'utilisateur sur les critères des documents. En effet, les préférences que nous avons exploitées ont donné des résultats positifs. Notre préoccupation future c'est d'exploiter d'autres critères des documents qui peuvent mieux décrire les natures des documents que l'utilisateur préfère.

Notre deuxième perspective concerne le temps de réponse pour trouver les résultats. En effet, l'utilisateur est intéressé par les documents qui répondent à son profil, mais il préfère aussi les trouver plus rapidement. Une opération d'optimisation du temps de réponse peut être entreprise.

Une troisième perspective concerne l'évolution des profils utilisateurs. En effet l'évolution des profils des utilisateurs est une tâche complexe. Nous avons abordé ce problème d'évolution des profils des utilisateurs, mais il nécessite encore d'autres réflexions afin de bien cerner ce problème. Nous proposons l'exploitation des réseaux bayésiens, les réseaux de neurones ou d'autres techniques d'apprentissage pour l'acquisition des profils.

Comme quatrième perspective concernant le profil d'une source, en se posant les questions ; Y a-t-il un moyen d'accès à tout le contenu d'une source ? Existe-il une possibilité d'avoir un profil source unique pour tous les utilisateurs ?

La cinquième perspective concerne l'implémentation de l'approche multi-agents comme un système à base de connaissances. L'objectif de cette perspective est d'exploiter au maximum les fonctionnalités d'un système multi-agents.

# Références

[Abbaci 2003] Méthode de sélection de collections dans un environnement de recherche d'information distribuée. Thèse de doctorat, université de neuchatel, 2003.

[Allen 1990] Allen R., User models, theory approach to speech act recognition. Technical report, 131/79, Dept. of computer science, University of Toronto, Canada, 1979.

[Armstrong et al 1995] Armstrong R., Freitag D., Joachims T., Mitchell T. WebWacher: A learning apprentie for the World Wide Web, AAAI Spring symposium on information Gathering from Heterogeneous distributed environments, Standford 1995.

[Bailey et al 2003] Bailey P., Craswell N., Hawking D., Engineering a multi-purpose test collection for Web retrieval experiments. Information Processing and Management 2003.

[Balpe et l1996] Balpe J.P., Lelu A., Papy F., Techniques avancées pour l'hypertexte. Hermes (Eds) 1996.

[Baudisch 1997] Baudisch P. The profile Editor: designing a direct manipulative tool for assembling profile. In proceeding of Fifth DELOS Workshop on Filtering and Collaborative Filtering, page 11-17, Budapest, November 1997.

[Baumgarten 1999a] Baumgarten C., Probabilistic Information Retrieval in a Distributed Heterogeneous environment. PhD thesis, XX, 1999.

[Baumgarten 1999b] Baumgarten C., A probabilistic solution to selection and fusion problem in distributed information retrieval. In the annual International ACM / SIGHIR Conference on Research and Development in information Retrieval Berkley, CA USA, 1999.

[Belkin et Croft 1992] Belkin W.B., Croft B., Information retrieval filtering : Two sides of the same coin ? CACM, Pages 29-38, 1992.

[Bhattacharjee et al 1997] Bhattacharjee S, Ammar M.H, Zegura EW. E.W, Shah. V and Fei. Z., Application-layer anycasting. In Infocom 1997.

[Baldwin et al 2000] Baldwin J.F., Martin T.P., Tzanavari A. User Modelling Using Conceptual Graphs for Intelligent Agents. In: B. Ganter and G.W. Mineau (eds). ConceptualStructures: Logical, Linguistic, and Computational Issues. LNAI 1867, Springer Verlag, 2000, pp. 193-206.

[Boley et al 1998] Boley D., Gini M., Gross R., Han E.H., Hastings K., Karypis G., Kumar V., Mobasher B., Moore J. Document categorization and query generation on the World Wide Web using WebAce 1998.

[Boughanem 1992] Boughanem M. Les Systèmes de Recherche d'informations : D'un Modèle Classique à un Modèle Connexionniste, Thèse de Doctorat de l'Université Paul Sabatier, Toulouse (France), Décembre1992

[Bourret et Samuelides 1991] Bourret P., Reggia Samuelides J. Réseaux de Neurones, une Approche Connexionniste de L'Intelligence Artificielle, Edition TEKNEA 1991

[Bouzeghoub et Kostadinov 2004] Bouzeghoub M., Kostadinov D. Une approche multidimensionnelle pour la personnalisation de l'information INRIA Rocquencourt et Laboratoire PRiSM, Université de Versailles 45, avenue des Etats-Unis, 78035 Versailles 2004.

[Brusilovsky et Marbury 2002] Brusilovsky P., Marbury M.T. From adaptive hypermedia to adaptive web. Communication of the ACM 45(5), Special issue on the adaptive Web, 2002.

[Budzik et Hammond 2000] Budzik J., Hammond K.J. User interactions with everyday applications as context for just-in-time information access. Proceeding of the 5th international conference on intelligent user interface page 44-51, 1998.

[Callan et al 1995] Callan J., Lu Z., Croft B. Searching distributed collection with inference networks. In the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, ACM-SIGIR'95, p. 21-28, july 1995.

[Callan 1998] Callan J. Learning while filtering documents. Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, pages 224-231, August 1998.

[Carter et Crovella 1996 ] Carter R.L., Crovella M.E., Dynamic server selection using bandwidth probing in wide-area networks. Technical Report BU-CS-96-2007, Boston University, march 1996.

[Chakravarthy et Haase 1995] Chakravarthy A.S., Haase K.B. Netserf ; Using semantic knowledge to find internet information archives. In the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Pages 4-11, Seattle, WA, July 1995.

[Chau et al 2001] Chau M., Zeng D., Chen H., Personalized spiders for Web search and analysis, In proceeding of ACM/IEEE Joint Conference on Digital Libraries, 2001.

[Chen et Wang 1995] Chen S., Yang J.Y. Document Retrieval Using Knowledge Based Fuzzy Information Retrival Technique. IEE Transactions on Systems, Man and Cybernetics, 793-803

[Chen et Wilensky 1997] Chen I., Wilensky R., An experiment in enhancing information access by natural language processing. Technical report CSD-97-963,Computer science division, University of California, Berkeley, 1997.

[Chen et Sycara 1998] Chen L, Sycara K. WebMate: a personal agent for searching and browsing. Proceedings of the 2nd International Conference on Autonomous Agents, 1998.

[Chin 2001]Chin David N. Empirical evaluation of user models and user adapted systems. User modelling and user-adapted interaction. Kluwer Academic Publishers, 2001.

[Craswell et al 2000] Craswell. N, Bailey. P, and Hawking. D., Server selection in the World Wide Web. In the Fifth ACM Conference on Digital Libraries, Pages 37-46, 2000.

[Croft 1993] Croft W.B. Knowledge_based and statistical approaches to text retrieval, IEEE Expert, 8(2), 1993.

[Croft et Harper 1979]. Croft W., Haper D. Using Probabilistic Models for Document Retrieval without Relevance Information, Journal of Documentation pp 185-202, 1979

[Daniels 1986] Daniels J.P., Cognitive models in information retrieval - An evaluation review, journal od documentation, 42(4), 272:304, 1986.

[Dean et Henzinger 1999] Dean J., Henzinger M.R., Finding related pages in the World Wide Web. In Proceeding of WWW-8, the Eight International World WideWeb Confference, Fortec Seminars 1999.

[Deerwester et al 1990] Deerwester S., Dumais S., Furnas S., Landauer G., Harshman R. Indexing by Latent Semantic Analysis : Journal of the American Society for Information Science, 391-407

[Dolin et al 1996] [DAEA96] Dolin R., Agrawal D., El-Abbadi A. Classifying network architectures for locating information sources. Technical Report TRCS96-23, Department of Computer Science, University of California, Santa Barbara, September 1996.

[Dreilinger et Howe 1997] Dreilinger D., Howe A.E. Experiences with selecting search engines using metasearch. ACM Transactions on Information Systems, 3(15):195-222, July 1997.

[Dumais 1994] Dumais S.T. Latent Semantic Indexing (LSI) and TREC-2. In D. Harmann, Ed., Proceedings of TREC'2, pp. 105-115. 2004.

[Dumais et al 2003] Dumais S., Cuttrell E., Cadiz J.J., Jancke G., Sarin R., Robbins Daniel C., Stuff I've seen : a system for a personal information retrieval and re-use. In Proceeding of WWW-8, the Eihgt International World Wide Web Conference, Fortec Seminars 1999.

[Fan et al 2004] Fan W., Gordon M.D., Pathak P., Discovery of context specific ranking functions for effective information retrieval using genetic programming, IEEE Transactions on knowledge and data engineering,Volume 16, issusee 4, pp 523-527, 2004.

[Ferber1995] Ferber J. Les systèmes Multi-Agents, vers une intelligence collective IterEditions. Paris, 1995.

[Fox 1983] Fox E.A. Extending the Boolean and Vector Space Models of Information Retrieval with P-Norm Queries and Multiple Concept Types. PhD thesis, Cornell University, Ithaca NewYork, 1983

[Fox et Shaw 1994] Fox, E. A., Shaw, J. A. Combination of Multiple Searches. Proceedings of the 2nd Text Retrieval Conference (TREC-2), National Institute of Standards and Technology (NIST) Special Publication 500-215, pp. 243-252, 1994.

[Frakes 1992] : Frakes W.B., Information retrieval, data structures and algorithms. PRENTICE HALL, Englewood Cliffs, NJ, 1992.

[Furh 1999] Fuhr N. A decision-theoretic approach to database selection in networked IRr. ACM Transactions on Information Systems, 17(03):229-249, July 1999.

[Fuhr 2000] Fuhr N., Information retrieval: introduction and survey, post-Graduate course on information retrieval, University of Duisburg-Essen, Germany 2000.

[Glover et al 1999] Glover E.J., Lawrence S., Gordon M.D., William P., Birmingham C., Lee Giles C., Recommending Web documents based on user preference. In proceedings of the Workshop SIGIR on recommender systems, August 1999.

[Gowan 2003] Gowan J.P Mc : A multiple model approach to personalized information access. Thesis of Mster in computer science. Faculty of science, university College Dublin, Fenrurary 2003.

[Gravano et Garcia-Molina 1995] Gravano L., Garcia-Molina H. Generalizing gloss to vector space databases and broker hierarchies. In 21st VLDB Conference, pages 78-89, 1995.

[Gravano et al 1999] Gravano L., Garcia-Molina H., Anthony T. Gloss : Text-source discovery over the internet. ACM transactions on Information Systems, 24(03) : 229-264, June 1999.

[Grossman et Fieder 1998] : Grossman D.A., Fieder O., Information retrieval : Algorithms and heuristics. Kluwer Academic Publishers, 1998.

[Guyton et Schwarz 1995] Guyton J., Schwarz M. Locating nearby copies of replicated internet servers. Technical Report CU-CS-762-95, University of Colorado at Boulder, 1995.

[Haines et Croft 1993] Haines D., Croft W.B. Relevance Feedback and Inference Networks, Conference on Research and Development in Information Retrieval (SIGIR), pp 2-11, 1993

[Hawking et Thislewaite 1999] Hawking D., Thislewaite P. Methods for information server selection. ACM Transactions on Information Systems, 17(01):40-76, January 1999.

[Hawking et al 1999] Hawking D., Voorhees E., Craswell N., Bailey P. Overview of the TREC8 Web trak. In Eight Text Retrieval Conference (TREC-8), Gainthersburg MD, November 1999.

[Hebb 1949] Hebb D.O. The Organization of Behaviour : J. Wiley et Sons, New York, 1949

[Hofman 1999] Hofman T. Probabilistic Latent Semantic Indexing : In the Proceedings of the 22nd Annual International ACM SIGIR, Conference on Research and Development in Information Retrieval, August, 1999, Buckley USA

[Horvitz et al 1998] Horvitz E., Breese J., Heckerman D., Hovel D., Rommelse K. The lumiere project : Bayesian user modeling for inferring the goals and needs of software users. In Proceeding of the Fourteenth conference on uncertainty in artificial intelligence, Madison, Wisconsin 256-265, 1998.

[ISO 1993] ISO 9777:1993 information and documentation-commands for interactive text searching. In International Organization for Standardization, Geneva, Switzeland, 1993.

[Jansen et al 1998] Jansen B., Spink A., Bateman J., Searchers the subjects they search and sufficiency: a study of a large sample of EXCITE searches. In Proceedings of Web-Net 1998, World conference of the WWW, Internet and Intranet. AACE Press 1999.

[Jeh et Widom 2003] Jeh G., Widom J., Scaling personalized Web search. In Proceedings of the 12th International World Wide Web Conference 2003.

[Jenning et al 1993] Jenning A., Higuchi H. A user model neural network for a personal news service. User modelling and user adapted interaction 1-25 1993.

[Kahle et Medlar 1991] Kahle B., Medlar A., An information system for corporate users : Wide area information servers. Technical Report TMC199, Thinking Machine Corporation, April 1991.

[Katz et al 1994] Katz E.D., Butler M., McGrath R. A scalable http server : the ncsa prototype. In First International World Wide Web Conference, Geneva, Switzerland, may 1994.

[Kechid et Drias 2003a] Kechid S., Drias H. Protocoles de négociation dans un marché électronique, Conférence Internationale sur la Productique, 2003, Alger, Algérie, pp14-16.

[Kechid et Drias 2003b] Kechid S., Drias H. Negotiation Protocol inter Agent in an Electronic Market. Arab Conference on Information Technology, 2003. Egypt, pp 432-437.

[Kechid et Drias 2005] Kechid S., Drias H. Negotiation Protocol Inter Agents in an Electronic Market. International Arab Journal on Information Technology, 2005, pp 112-117.

[Kechid et Drias 2006a] Kechid S., Drias H. Accès Personnalisé à de Multiples Serveurs d'Informations. In proceedings of COnférence sur la Recherche d'Informations Appliquée, Mars 2006, Lyon France, pp 249-254……

[Kechid et al 2006] Kechid S., Drias H. Tamine L., Personalized Access to multiple information sources, Information Sciences and Technology, 2006, Mérida, SPAIN, October, 25-28th, 2006, pp 84-88.

[Kechid et Drias 2006b] Kechid S., Drias H. Accès Personnalisé multicritères à de Multiples Sources d'Informations. In proceedings of Rencontre des Jeunes Chercheurs sur la Recherche d'Information, Mars 2006, Lyon France, pp 401-406.

[Kechid et Drias 2008] Kechid S., Drias H. A multi-agent approach for integrating the user model in a distributed information retrieval system, submitted.

[Kechid et Drias 2009] Kechid S., Drias H., Personalizing the source selection and the result merging process, In International Journal on Artificial Intelligence Tools. To appear in Vol. 18(2) April 2009 issue.

[Kostadinov 2003] Kostadinov D. Personnalisation de l'information et gestion des profils utilisateurs. Mémoire de DEA PRiSM , Versailles. 2003.

[Kwok 1995] Kwok K.L. A Network Approach to Probabilistic Information Retrieval, ACM Transactions on Information Systems, Vol 13 N3 pp 324 - 353, 1995

[Kwok et al 1995] Kwok K.L., Grunfeld L., Lewis DD, TREC-3 Ad-hoc, Routing Retrieval and Thresholding Experiments using PIRCS. Proceedings of TREC-3,1995, pp. 247-255.

[Lamrous 1999] Lamrous S. Modélisation et réalisation d'un système prototype interactif de recherche d'information multimédia à forte composante textuelle, Thèse de doctorat, UT Compiègne, Juin 1999.

[Lawrence et Lee Giles 1998] Lawrence S., Lee Giles C. Inquirus, the neci meta search engine. In The seventh International World Wide Web Conference, 1998.

[Lawrence et Giles 1999] Lawrence S., Giles C.L. Accessibility of Information on the Web. Nature Vol 400, 107-109, 1999.

[Le Calvé et Savoy 2000] Le Calvé A., Savoy J. (2000). Database Merging Strategy Based on Logistic Regression. Information Processing et Management, Vol. 36, pp. 341-359.

[Lechani et Boughanem 2005] Lechani L., Boughanem M. Accès personnalisé à l'information : Approches et Techniques. Equipe SIG/RFI , IRIT/Toulouse, Rapport Interne, Janvier 2005.

[Li 2000] Li X., Adaptive personalized information retrieval, a thought analysis and utililization of user preferences in IR 2000.

[Lieberman 1995] Lieberman H. Letizia, An Agent That Assists Web Browsing, 1995 International Joint Conference on Aritifical Intelligence, Montreal, CA, 1995.

[Liu et al 2002] Liu F., YU C.T., Meng W., Personalized Web search by mapping user queries to categories. In Proceeding of CIKM 2002.

[Lu et al 1996] Lu Z., Callan J., Croft B., Measures in collection ranking evaluation. Technical report, Computer Science Department, University of Massachusetts, 1996.

[Lucarella et Morara, 1991] Lucarella D., Morara R. FIRST Fuzzy Information Retrieval Systems. Journal of Information Science, 81-91, 1991

[Moffat A., Zobel 1995] Moffat A., Zobel J. Information retrieval systems for large document collections. In The Text REtrieval Conference (TREC-3), pages 85-94, National Institute of Standards and Technology, 1995.

[Negus 1979] Negus A. Development of the euronet-diane common command language. In Third International Online Information Meeting, pages 95-98, 1979.

[Oard 1996] Oards D W. Adaptive Vector Space Text Filtering for Monolingual et Cross-language applications, Dissertation for the requirement for the degree of Doctor of Philosophy, University of Maryland, 1996

[Ogston 2003] Ogston. E. Group formation among peer-to-peer agents: learning group. Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, 2003.

[Org 1993] National Information Standards Organization. Z39.50-1992 common command language for online interactive information retrieval. NISO Press, Bethesda, 1993.

[Pazzani et al 1996] Pazzani M., Muramatsu J., Billsus D. Syskill et Webert : Identifying interesting Web sites, In Proceedings of the Fourteenth national conference on Artificial intelligence, AAAI Press, 1996.

[Ponte 1998] Ponte J.M. A Language Modelling Approach to Information Retrieval, Doctor of philosophy thesis, University of Massachusetts, September 1998

[Powell et al 2000] Powell A.L., French J.C., Callan J, Connellet M., Viles C.L.. The Impact of Database Selection on Distributed Searching. Proceedings of the 23rd International Conference of the ACM-SIGIR'2000, pages 232-239, New York: The ACM Press, 2000.

[Pretschner et Gauch 1999] Pretschner A., Gauch S., Personalization on the Web. Technical report ITTC-FY2000-TR-13591-01, Information and telecommunication technology center, Department of electrical engineering and computer science, University of kanas, 1999.

[Ricardo 1999] Ricardo B.Y. Modern Information Retrieval, Addison Wesley 1999.

[Rijsbergen 1979] Van Rijsbergen., Information Retrieval, 2ème Edition, Butterworths, Londres(UK), 1979.

[Robertson et Sparck Jones 1976] Robertson S.E., Sparck Jones K. Relevance Weighting for Search Terms, Journal of The American Society for Information Science, Vol 27, N°3, pp 129-146, 1976.

[Robertson et Walker 1994] Robertson S., Walker S. Some Simple Effective Approximations to the 2-Poisson Model for Probabilistic Weighted Retrieval, In Proceedings of the seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp 232-241, 1994.

[Robertson et Walker 1997] Robertson S., Walker S. On Relevance Weights with Little Relevance Information, In Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development, pp16-24, 1997.

[Rocchio 1971] Rocchio, J., Relevance feedback information retrieval. In Gerald Salton (editor), The SMART retrieval system- experiments in automated document processing. Prentice-Hall, Englewood Cliffs, NJ, 1971.

[Salton 1968] Salton G. Automatic Information and Retrieval, Mcgrawhill Book Company, N. Y., 1968

[Salton 1971] Salton G., The Smart Retrieval System : Experiments in Automatic Document Processing, G. Salton Editor, Prentice Hall Inc., Englewood Cliffs, New Jersey, 1971.

[Salton et al 1975] Salton G., Wong A., Yang C., (1975) A vector space model for automatic indexing. Communications of the ACM, 18, 1975 pp. 613-620.

[Salton 1989] Salton G. Automatic Text Processing. The Transformation Analysis and Retrieval of Information by Computer. Addison Wesley, Reading 1989

[Salton et Buckley 1990] Salton G., Buckley C., Improving retrieval performance by relevance feedback. In Sparck Jones and Willet, (Eds) Reading in information retrieval , San Francisco, CA, Morgan Kauffman, 1990.

[Salton et Allan 1994] Salton G., et Allan J. Automatic Text Decomposition and Structuring, Actes du Congrès RIAO'94, Intelligent Multimedia Information on Retrieval Systems and Management, New York (USA) pp 6-20, 1994

[Savoy et Desbois 1991] Savoy J., Desbois D. Bayesian Inference Networks in Hypertext, Intelligent Multimedia Information Systems and Management (RIAO), pp 662-681, 1991

[Savoy et Rasolofo 2000] Savoy J., Rasolofo Y. Recherche d'informations dans un environnement distribué. In actes "Traitement Automatique de la Langue Naturelle, TALN 2000 Lausanne (Suisse), octobre 2000, pp. 317-326.

[Scime et al 2000] Scime A., kershbergs L., WebSifter : An Ontologiy-based personalizable search agent for the Web. In Proceedings of International Conference on Digital Libraries: Research and practice 2000.

[Seo et Zhang 2000] Seo Y-W, Zhang T A Reinforcement Learning Agent for Personalized Information Filtering, Proceedings of the 2000 International Conference on Intelligent User Interfaces, New-Orleans, USA, Jan 9-12, 2000, ACM, pp248-251, 2000.

[Shahabi et Chen 2003] Shahabi C., Chen Y.S. Web information personalization : Challenges and approaches, Databases in networked information systems, 3rd International Workshop, 2822, 5-15, Japan September 2003.

[Si et Callan. 2003] L. Si, J. Callan. A Semi-Supervised learning method to merge search engine results. ACM Transactions on Information Systems.2003 pp. 457-491.

[Si et Callan 2004] L. Si, J. Callan. Unified Utility Maximization Framework for Resource Selection. Proceedings of the ninth nternational conference on Information and knowledge managementCIKM washington USA 2004, pp 32-41.

[Si et Callan, 2005] L. Si, J. Callan. Modeling Search Engine Effectiveness for Federated Search. Proceedings of the Twenty Seventh Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. Toronto, Canada: ACM 2005, pp 83-90

[Singhal et al 1995] Singhal A., Salton G., Mitra M., Buckley C. Document Lenght Normalisation, Rapport de recherche, 1995.

[Soltysiak et Crabtree 1998] Soltysiak S.J., Crabtree I.B. BT Technol J.Automatic learning of user profiles-towards the personalisation of agent services, Vol 16 No 3, July 1998, pp 110-117.

[Sparck Jones 1971] Sparck Jones K., Automatic keyword classification for information retrieval, 1971.

[Sparck Jones 1979] Sparck Jones K., Experiments in relevance weighting of search terms. IPM, 15(3), pages 133-144, 1979.

[Sparck Jones et Needham 1972] Sparck Jones K., Needham R. M. Automatic Theme Classification and Retrieval, Information Processing and Management, Vol 4, pp 91-100, 1972.

[Spink et al 1998] Spink A., Bateman J., Jansen B. User's searching behavior on the EXCITE web search engine. In Proceedings of Web-Net, 1998, World conference of the WWW, Internet and Intranet. AACE Press 1998.

[Spink et al 2001] Spink A., Wolfram D., Jansen B., Saracevic T. Searching the Web : The public and their queries, 2001.

[Su et Lee 2003] Su J., Lee M. An exploration in personalized and context-sensitive search, In proceedings of SIGIR Conference, In proceedings of the 7th annual CLUk (the UK special interest group for Computational Linguists) research colloquium 2003.

[Tamine 2000] Tamine L. Optimisation de requêtes dans un système de recherche d'information approche basée sur l'exploitation de techniques avancées de l'algorithmique génétique. Thèse de doctorat à l'université de Paul Sabatier Toulouse, 2000.

[Turtle et Croft 1991] Turtle H., Croft W.B. Evaluation of an Inference Network Based Retrieval Model. ACM Transactions on Information Systems July 1991.

[Vishnu 2004] Vishnu K. R C. Contextual information retrieval using ontology based user profiles, Master's thesis, January 2004.

[Voorhess 1995] : Voorhess M., Siemens trec-4 report : further experiments with database merging. In the fourth text Retrieval Conference (TREC), Gainthersburg, Maryland, 1995.

[Voorhees et al 1995a] Voorhees E.M., Gupta N.K., Johnson-Laird B. The Collection Fusion Problem. Proceedings of TREC'3, pages 95-104. Gaithersburg: NIST Publication #500-225, 1995.

[Voorhees et al 1995b] Voorhees E. M, Gupta N. K, Johnson-Laird B. Learning Collection Fusion Strategies. Proceedings of the ACM-SIGIR'95, 1995, pp. 172-179.

[Wilkinson et Hingston 1991] Wilkinson R., Hingston P. Using The Cosine Measure in A Neural Network for Document Retrieval, Conference on Research and Development in Information Retrieval (SIGIR), pp 202-210, Chicago (USA), 1991

[Wolfram et al 2001] Wolfram D., Spink A., Jansen B., Saracevic T. Vox populi : The public searching of the web. Journal of the American Society for information Science and Technology, pages 1073-1074, 2001.

[Wong et al 1985] Wong S.K.M., Ziarko W., Wong P.C. N. Generalized Vector Space Model in Information Retrieval. In Proc of the 8th ACM SIGIR Conference on Research and Development, New-York USA, 1985

[Xu et Croft 1996] Xu J., Croft B., Query expansion using local and global document analysis. In The Annual International ACM / SIGIR Conference on Research and Development in Information Retrieval, pages 4-11, 1996.

[Xu et Callan 1998] Xu J., Callan J. Effective retrieval with distributed collections. In The Annual International ACM / SIGIR Conference on Research and Development in Information Retrieval, Lebourne, Australia, 1998.

[Xu et Croft 1999] Xu J., Croft B. Cluster-based language model for distributed retrieval. In The Annual International ACM / SIGIR Conference on Research and Development in Information Retrieval, Berkley, CA USA, 1999.

[Yates et Neto 1999] Yates R.B., Neto R., Modern information retrieval. ACM Press, Addison Wesley, 1999.

[Yu et Singh 2003] Yu. B and Singh. M. Incentive mechanisms for agent-based peer-to-peer systems. Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, 2003.

[Yumono et Lee 1997] Yumono B., Lee D.L. Server ranking for distributed text retrieval system on the internet. In the fifth International Conference on Database Systems for Advanced Applications, Melbourne, Australia, April 1997.

[Zadeh 1965] Zadeh L.A. Fuzzy Sets, Information Control, 8 : p 338-353, 1965

[Zhang et al 2004] Zhang H, Croft W.B, Levine B, Victor Lesser. A Multi-agent Approach for Peer-to-Peer-based Information Retrieval Systems AAMAS'04, July 19-23, 2004, New York, New York, USA.Copyright 2004 ACM 1-58113-864-4/04/0007. 2004.