République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et
De la Recherche Scientifique
Université Mouloud Mammeri. Tizi-Ouzou
Faculté Génie Electrique et Informatique
Département d'Informatique

Mémoire de fin d'études

En vue de l'obtention du diplôme du master
en Informatique
Option : conduite de projets informatique

Thème

Intégration de La Fraicheur d'Information sociale dans la Recherche d'Information Sociale

Proposé et dirigé par :

Melle BOUGCHICHE

Réalise par :
Melle : GOUDJIL Nadia
Melle : ABDELLI Hayat

2011_ 2012

Remerciements

On tient à remercier en tout premier lieu Mme BOUGCHICHE, notre encadreur qui a su orienté notre travail et pour sa disponibilité pour prodiguer des conseils et des orientations, pour sa confiance et pour sa précieuse aide, on la remercie du fond du cœur.

Nos remerciements s'adressent aussi aux membres du Jury pour nous avoir honorés en consentant à juger notre modeste travail.

Nos sincères remerciements sont adressés à tous ceux qui, sans avoir été impliqués directement dans ce travail, ont toujours été d'un grand support :nos enseignants, nos familles, nos amis, nos collègues chacun son nom.

Dédicaces

Je dédie ce travail
A la mémoire de mon cher père et ma chère grand-mère
A La lumière de ma vie qui n'a jamais cessé de m'éclairer et à qui je dois tout : ma très chère mère

A mes frères et mes sœurs (Saïd, mehand yiyi, hamid, lynda, sabiha, karima,chafia)

A mon bien aimer Saïd
A toutes les personnes qui pensent à moi de près ou de loin.

Nadia

Dédicaces

Louanges à dieu le seul et l'unique

A mes très chères parents en reconnaissance de leur précieux soutient toutes au long de mes études ;

A mes sœurs Samia, Sylia et Kamelia ;

A ma sœur Kahina et son mari Mourad ainsi qu'à sa fille Anais

A mon adorable frère Abdenour

A Loucif qui n'a jamais hésité à me venir en aide et qui n'a jamais cessé de m'encourager qu'il trouve ici l'expression de ma reconnaissance et de ma sincère gratitude ;

A Naouel, Zaki, Mohammed, Samira et à tous mes amis(es)

A mon binôme et sa famille

Je dédié ce modeste travail

Hayat

Résumé

Depuis quelques années, les services des réseaux sociaux attirent l'attention des internautes. Cet attrait peut s'expliquer par la facilité et la rapidité avec lesquelles les internautes peuvent partager des informations, le plus souvent en temps réel. Les internautes cherchent souvent à collecter des informations récentes sur les derniers sujets d'intérêt. Trouver les meilleurs résultats pour un sujet dépond de la fraicheur des informations.

Dans ce rapport de master, nous proposons d'intégrer la fraicheur d'information dans la recherche d'information sociale. Nous considérons un modèle de recherche basé sur les informations sociales (tags) et qui comporte trois modules. Le premier consacré à l'indexation des données, le deuxième vise à déterminer le score classique et le score sociale pour une requête donnée et enfin le dernier module vise à combiner linéairement les deux scores obtenus de façon à déterminer les documents les plus récents.

Mots clés :

Recherche d'information, recherche d'information sociale, réseaux sociaux, tags, fraicheur d'information, BM25.

SOMMAIRE

Chapitre III: Conception

Chapitre IV: Réalisation

Table des figures

Chapitre I : le processus de la recherche d'information classique

Chapitre II : la recherche d'information sociale et la fraicheur d'information

Chapitre III : conception

Chapitre IV : la réalisation

Liste des tables

Introduction générale :

Les plates formes communautaires actuelles, connues sous le nom de réseaux sociaux, sont en prolifération et nécessitent de plus en plus d'utilisateurs autour de centres d'intérêts communs.ces réseaux sociaux tels que Facebook, Delicious ,Youtube ou Twitter offrent à leurs membres des espace individuels pour déposé des documents de toutes natures(texte, photos, musique, vidéo,…), les organiser, les partager, leur attribuer des étiquettes(des tags) et les sauvegarder dans des bookmarks pour des usages ultérieurs .

L'activité la plus fréquente et qui attire l'attention des usagers de ces sites est l'étiquetage(le 'tagging') du fait que les tags apportent une valeur ajoutée aux informations existantes sur le réseau. Vu cette masse d'informations, la recherche d'une information précise n'est pas toujours efficace surtout que les systèmes de la recherche d'information actuels se basent sur des modèles de RI traditionnels (vectoriel,…).

L'objectif de notre étude est de prendre en considération cette information (tags) produite dans un contexte social afin d'améliorer les systèmes de RI actuels, du fait, ces tags constituent une forme indirect de validation sociale de toutes les informations auxquelles ils sont assignées surtout lorsqu'ils sont récemment i.e. à la mode, d'actualité ou encore frais. Nous allons, également, tenir compte de la fraicheur de ces tags dans notre étude vu que dans la vie quotidienne, la notion de fraicheur est important, la préférence de tout ce qui est frais et d'actualité, est l'inné chez l'homme. Dans le domaine alimentaire, si les articles proposés, du jour ils sont vite consommés par les clients, tandis que les produits non frais, sont déjà flétris et attirent de moins en moins les clients. Dans le domaine de journalisme plus une information est récente plus elle vaut cher. De ce fait, elle se voit dépassés s'elle n'est pas du jour ou encore même du matin.

Pour ce faire, nous avons organisé notre mémoire en quatre chapitres, le premier chapitre est consacré à la description générale et l'architecture des systèmes de recherche d'informations (SRI), le processus d'indexation des documents et des requêtes ainsi qu'une brève description des principaux modèles de recherche utilisés.

Le second aborde la recherche d'information sociale, les types d'informations sociales, la qualité d'information, nous présentons quelques expériences sur la qualité de l'information, son évaluation ainsi que son intégration dans la recherche d'information sociale.

Le troisième chapitre décrit notre contribution en explicitant la démarche à suivre pour prendre en considération l'influence da la fraicheur d'informations sur le processus de recherche

Le dernier chapitre, est consacré à la description de la réalisation de notre système de recherche d'information sociale avec le langage Java en appliquant un modèle de recherche basé sur la BM25 puis le combinant au score social basé sur la fraicheur des tags.

CHAPITRE 1

Processus de la recherche d'information classique

I. Introduction :

La Recherche d'Information (RI) est un domaine de l'informatique qui s'intéresse à l'organisation, au stockage et à la sélection d'informations répondant aux besoins des utilisateurs. Ce domaine manipule différents concepts : la requête, le besoin en information, les documents, la pertinence…etc. D'un point de vue utilisateur, l'accès à l'information peut être effectué de manière délibérée à travers un Système de Recherche d'Information (SRI) (on parle alors de collecte active de l'information), ou bien de manière passive à travers un système de filtrage d'information (c'est un processus qui permet d'extraire à partir d'un flot d'informations « News, e-mail, actualités journalières, etc. », celles qui sont susceptibles d'intéresser un utilisateur ou un groupe d'utilisateurs ayant des besoins en informations).

II. Les notions de base d'un système de recherche d'informations :

Les trois concepts clés de la RI :

1. Le document :

Élément central du SRI, c'est un objet complexe sans cesse en évolution car lié au développement des technologies de communication. Un document peut être un texte, un morceau de texte, une page web, une image, une bande vidéo, etc.

Il est ainsi important de signaler la difficulté de trouver une définition précise du terme document. Les dictionnaires donnent souvent une définition très rapprochée.

Citons par exemple le Petit Larousse qui définit le document comme étant un écrit ou objet servant d'information, de témoignage ou de preuve. On peut ainsi dire qu'un document est toute unité qui peut constituer une réponse à une requête utilisateur.

2. Les requêtes:

Une requête correspond à la traduction du besoin en information de l'utilisateur dans un langage d'interrogation du SRI. Elle est constituée d'une liste de mots clés, de termes du langage naturel, booléen ou d'un graphique.

3. Pertinence :

La pertinence est la notion centrale de la RI c'est la mesure d'informativité du document recherché pour la requête et on distingue deux types de pertinence.

a) pertinence système :

Il s'agit d'une évaluation de la similarité entre deux représentations interne: celle du document et celle de la requête, il est donc nécessaire de créer des

représentations internes pour la requête et pour les documents. La pertinence système est donc la correspondance que fait le système entre ces deux représentations.

b) pertinence utilisateur :

À ce niveau, le système repend à la requête formulée par l'utilisateur, par un ensemble de documents retrouvés (jugés pertinents) dans la base de documents. Un document sera jugé pertinent par un utilisateur si le document contient des informations qui correspondent à ce qu'il souhaite.

III. Définition d'un système de recherche d'information :

Un Système de Recherche d'Informations (SRI) est un système informatique qui permet de retourner à partir d'un ensemble de documents, ceux dont le contenu correspond le mieux à un besoin en informations d'un utilisateur, exprimé à l'aide d'une requête.

Un SRI inclut un ensemble de procédures et d'opérations qui permettent la gestion, le stockage, l'interrogation, la recherche, la sélection et la représentation de cette masse d'informations.

IV. Architecture générale d'un SRI :

L'architecture générale d'un SRI est représentée dans la **figure 1,** on y distingue les éléments de base suivants :

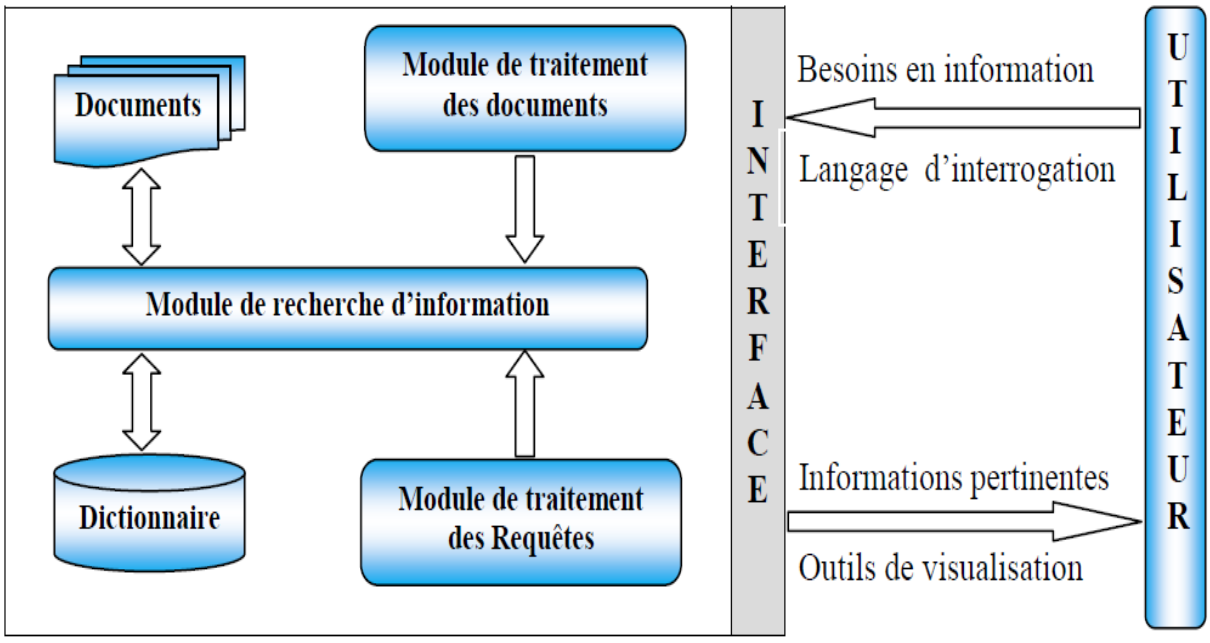


Figure 1 : Architecture générale d'un SRI [TAM, 98].

- **Interface :**

L'interface a pour objectif d'assurer la communication entre la base documentaire et l'utilisateur. Les SRI étant dédiés à un large public, des efforts se sont alors portés sur la conception d'interfaces ergonomiques et conviviales facilitant l'accès des utilisateurs à l'information voulue.
La communication SRI/utilisateur nécessite la mise en œuvre de langages d'interrogation pour l'expression des requêtes d'une part, et d'outils de visualisation pour la visualisation de l'information pertinente d'autre part.

- **Langage d'interrogation :**

Plusieurs types de langages d'interrogation ont été mis au point dans les SRI. Les langages les plus répandus sont les suivants:

a. Le langage booléen :

Dans ce type de langage, une requête est considérée comme étant une expression booléenne. Les opérateurs **ET**, **OU** et **NON** relient les termes de la requête et les parenthèses sont utilisées pour exprimer l'ordre de priorité.
Par exemple :
<<Pollution AND mers>>
<<Mers OR océans>>
<<Pollution NOT AND NOT marine>>
L'avantage de ce langage est l'absence d'ambigüité dans les requêtes, mais son inconvénient majeur c'est que seuls les documents répondant exactement à la requête seront sélectionnés (les documents qui répondent seulement à une partie de la requête ne seront pas inclus dans la sélection), aussi il est difficile de formuler la requête lorsque le nombre des opérateurs est important.

b. Le langage naturel :

Ici, l'utilisateur possède toute la liberté dans le choix des termes de sa requête, c'est le cas des moteurs de recherche web.
Par exemple :
<<Donnez-moi tous les ouvrages d'Albert Camus>>
L'avantage est que l'accès à la recherche est mis à la disposition du grand public avec un temps d'apprentissage réduit, cependant l'inconvénient résident dans l'ambigüité des langages naturels.

c. Le langage graphique:

Dans ce type de langage, le système assiste l'utilisateur pendant la formulation de la requête en lui proposant graphiquement une liste de termes qui représentent le contenu sémantique de la collection des documents. La requête sera donc formulée avec un vocabulaire contrôlé ce qui améliorera la pertinence. Ce type de langage est conçu pour les petites collections de documents.

- **Outils de visualisation:**

Les outils de visualisation mis en œuvre dans les SRI, offrent des possibilités de consultation de l'intégralité de documents, de passages de documents pertinents ou d'identifiants de documents.

- **Modules de traitement des documents et requêtes:**

La structure d'un SRI intègre un module de traitement des documents et un module de traitement des requêtes :

- **Module de traitement des documents:**

Comprend des outils de gestion de la base documentaire : représentation, organisation et stockage des documents.

- **Module de traitement des requêtes:**

A pour objectif de représenter les requêtes utilisateur dans un modèle prédisposant à la recherche d'information.

➢ **Module de recherche d'information :**

C'est le module noyau du système. Il comprend la procédure fondamentale permettant d'effectuer un « rapprochement » entre requête utilisateur et documents de la base, afin de restituer les informations pertinentes. Notons que le modèle de recherche d'information est étroitement lié aux modèles de représentation des documents et requêtes, mis en œuvre dans les précédents modules.

➢ **Document et base documentaire:**

Le document constitue le potentiel d'information élémentaire d'une base documentaire. Le contenu de la base documentaire diffère d'une base documentaire à une autre selon le domaine d'application considéré.
On distingue principalement deux types de bases documentaires **:** les référothéques et les bibliothèques :

a) Les référothéques:

Une référothéque est constituée d'un ensemble d'enregistrements faisant référence au document dans lequel se retrouve l'information intégrale. Une unité d'information est composée d'un résumé du texte intégral (*abstract*) et de données factuelles complémentant la description du document.

b) Les bibliothèques:

Ce sont des bases documentaires composées des textes intégraux de documents (*full texts*).

- **Dictionnaire:**

Le dictionnaire comprend principalement les mots clés du domaine de la base documentaire ainsi que des mots outils nécessaires au traitement des requêtes pour aboutir à leur représentation intermédiaire.

V. Le processus de recherche d'information :

Pour répondre aux besoins en information de l'utilisateur, un SRI met en œuvre un certain nombre de processus pour réaliser la mise en correspondance des informations contenues dans un fonds documentaire d'une part, et les besoins en information des utilisateurs d'autre part. Un SRI est défini par ses modèles de représentation des documents, des besoins de l'utilisateur et sa fonction d'appariement.

Ce processus est composé de trois fonctions principales :

- ✓ L'indexation des documents et des requêtes : l'indexation a pour rôle d'extraire à partir d'un document ou d'une requête, une représentation paramétrée qui couvre au mieux son contenu sémantique.
- ✓ L'appariement requête-document ou l'interrogation : la comparaison entre le document et la requête revient à calculer un score, supposé représenter la pertinence du document vis-à-vis de la requête.
- ✓ La reformulation de la requête : qui intervient suite à une itération de recherche, et consiste à modifier les requêtes en fonction des résultats présentés et le jugement de l'utilisateur.

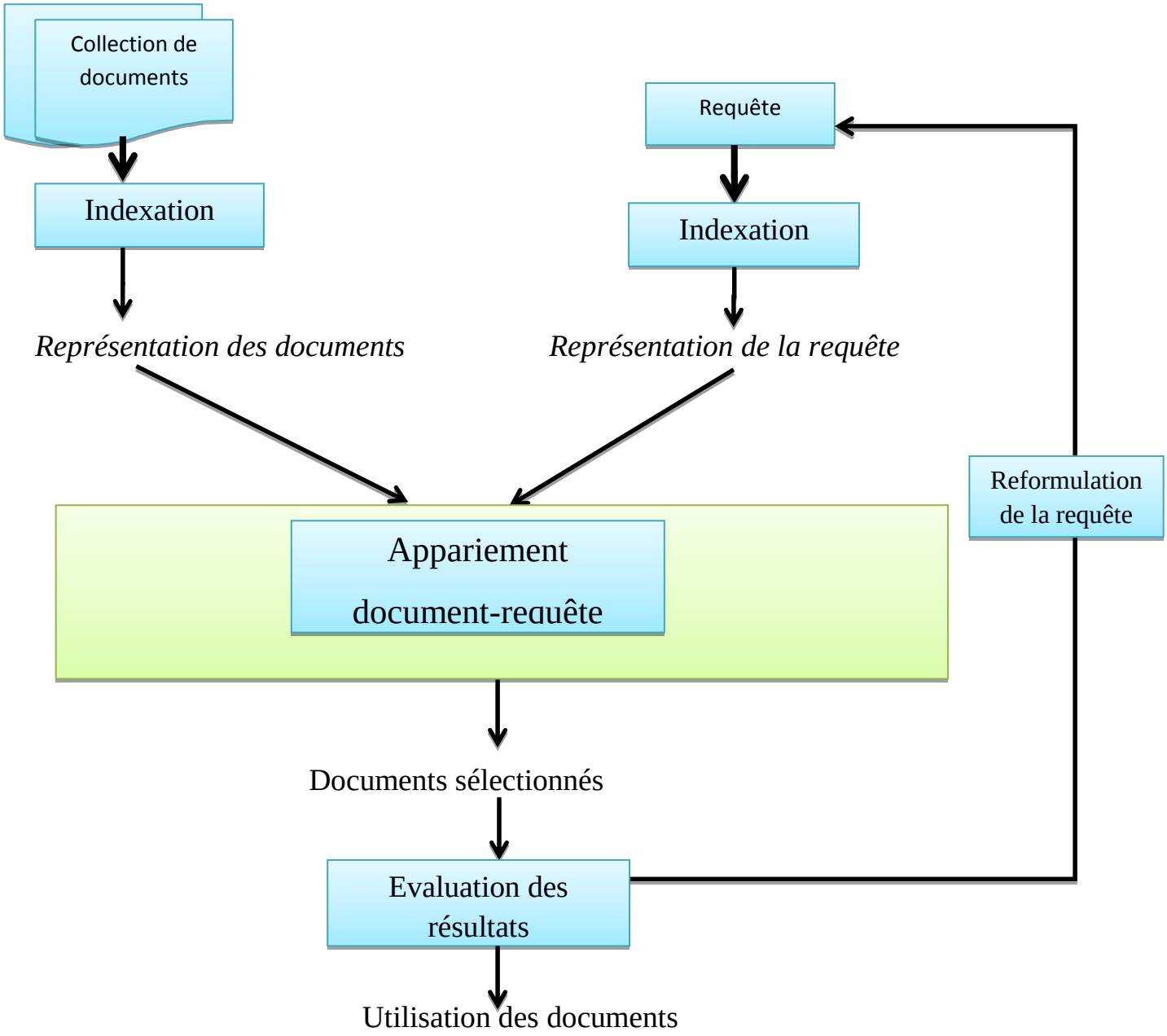


Figure 2: *Processus en U de recherche d'informations.*

1. Principales phases du processus de RI :

L'objectif fondamental d'un processus de RI est de sélectionner les documents ‘‘les plus proches’’ du besoin en information de l'utilisateur décrit par une requête. Ceci induit deux principales phases dans le déroulement du processus : indexation et appariement requête/documents.

1.1. L'indexation des documents et des requêtes :

Dans un processus de RI le coût de recherche doit être acceptable, il convient alors de procéder à une phase fondamentale pour optimiser le temps d'exécution de ce processus.

Cette phase consiste à analyser chaque document de la collection, et de créer un ensemble de mots-clés, on parle alors de l'indexation. Le rôle de l'indexation est fournie une présentation synthétique du contenu du document, on distingue trois types d'indexation :

- manuelle : la représentation du document se fait par un spécialiste (documentaliste) ;
- automatique : le processus d'indexation est complètement informatisé ;
- semi-automatique : le processus d'indexation se fait en premier lieu d'une manière automatique, le documentaliste intervient seulement pour ajouter des mots-clés qu'il trouve intéressants pour représenter un document.

L'indexation manuelle permet d'assurer une bonne représentation des documents, cependant elle présente un inconvénient majeur vu que la tâche du documentaliste se veut souvent influencée par son point de vue, or celui-ci est très subjectif : aucune règle universelle ne peut être appliquée à cette fin. L'indexation semi-automatique est un processus partiellement automatisé, le documentaliste intervient pour apporter les rectifications qu'il voit nécessaires. L'indexation automatique est la plus communément utilisée. Ce type d'indexation regroupe un ensemble de traitements automatisés sur un document :

- ✓ l'extraction automatique des termes du document.
- ✓ l'élimination des mots vides.
- ✓ la lemmatisation ou la radicalisation des mots, la pondération.
- ✓ la création de l'index.

1.1.1. Les étapes d'indexation :

L'indexation passe par les étapes suivantes :

- **Extraction des mots-clés :**

Consiste à extraire à partir du texte des documents, les termes clés appelé aussi descripteurs. Les descripteurs représentent l'information atomique d'un index. Ils sont censés indiquer de quoi parle le document **[SAL, 83]**. Repose sur l'analyse linguistique de texte du document. Plusieurs niveaux d'analyse peuvent être distingués : niveau lexical, syntaxique et sémantique.

- **Elimination de mots-vides :**

Un des problèmes majeurs de l'indexation consiste à extraire des termes significatifs et à éviter les mots vides.

- Les termes qui apparaissent dans la plus part des documents d'une collection ne sont pas discriminants. Ces termes sont qualifiés de mots vides et sont rangés dans un anti-dictionnaire (stoplist en anglais) ces mots sont souvent des prépositions (" de ", " à "), pronom (" aucun ", "

tout ", " on "), certain adverbes (« ailleurs", " maintenant "), adjectifs (" certain ", " possibles "), etc.

- Les termes dépassant un certain nombre d'occurrence (fréquence d'occurrence trop élevée) et les termes de fréquence quasi-nulle peuvent être éliminé.

Même si l'élimination des mots vides a l'avantage évident de réduire le nombre de termes d'indexation, elle peut cependant réduire le taux de rappel, c'est à dire la proportion des documents pertinents renvoyé par le système par rapport à l'ensemble des documents pertinents (loi de Zipf) **[NGA ,09]**.

- **Radicalisation(Lemmatisation**) :

Consiste à représenter les différentes variantes d'un mot par une forme unique (racine grammaticale). On peut par exemple citer économie, économiquement, économétrie, etc. appartiennent au même champ lexical. Il n'est pas forcément nécessaire d'indexer tous ces mots alors qu'un seul suffirait à représenter le concept véhiculé. Pour résoudre le problème, une substitution des termes par leur racine est utilisée.

- **La pondération :**

La pondération des termes permet de mesurer l'importance d'un terme dans un document. Consiste à associer un poids d'importance (ou valeur de représentativité) à chaque terme d'un document. Cette mesure est souvent calculée en se basant sur des propriétés et interprétations statistiques. D'après Zipf **[KAR, 05],** si on dresse une liste de l'ensemble des mots différents d'un texte quelconque, classés par ordre de fréquences décroissantes, on constate que la fréquence d'un mot, est inversement proportionnelle à son rang de classement dans la liste. Formellement, ceci peut être traduit par la formule suivante :

Fréquence * rang = constante … (1)

La loi de distribution des termes suit alors la courbe présentée dans la figure suivante :

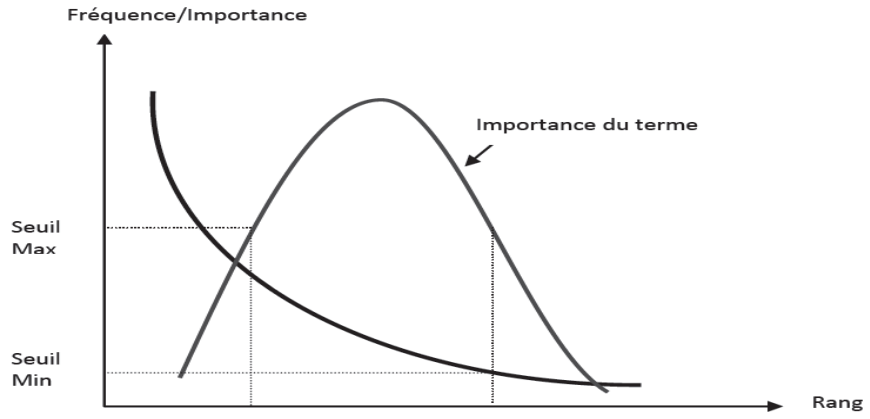


Figure3 : rapport entre la fréquence d'un terme et son importance.

Les termes représentatifs vont être triés en définissant un seuil maximum en dessous duquel les termes à rang×fréquence élevé ne sont pas conservés et un seuil minimum au-dessus duquel les termes à rang×fréquence faible ne sont pas conservés non plus.

La fréquence relative d'un terme dans un document est représentative du pouvoir de représentation du terme pour le document, dans le même temps, la fréquence absolue d'un terme dans la collection est caractéristique du pouvoir de discrimination du terme pour les documents. Il est donc important de prendre en compte la fréquence relative et la fréquence absolue d'un terme lors de sa pondération. La pondération c'est l'association d'une valeur appelée poids à un terme. Pour associer un poids à un terme on peut procéder de différentes manières :

- *0 ou 1* : exprime la présence (1) ou l'absence (0) d'un terme dans le document.
- *tf* : *term-frequency* est la fréquence du terme dans le document c'est-à-dire le nombre d'occurrences d'un terme dans le document.
- *idf* : *Inverse of Document Frequency* est la fréquence absolue inverse. C'est un facteur qui varie inversement proportionnel au nombre *n* de documents où un terme apparaît dans une collection de *N* documents.

La fréquence absolue inverse est égale à:

$$Idf = log\ (N/n) \ldots (2)$$ **[SAL, 87]**

Avec *N :* le nombre total de documents dans la collection

n : le nombre de documents où le terme apparaît.

Le poids d'un terme *j* dans le document *i* s'écrit alors généralement

$$Poids_i(j) = tfij \times id \quad \ldots(3)$$ **[SPA, 72]**

Où tf_{ij} *:* est la fréquence d'apparition du terme *j* dans le document *i*

Idf_j *:* est la fréquence absolue inverse du terme *j* dans la collection.

Ainsi le poids d'un terme augmente si celui-ci est fréquent dans le document et décroît si celui-ci est fréquent dans la collection.
La formule *tf×idf* fournit une bonne représentation du poids pour les corpus dont les documents sont de taille homogène c'est-à-dire composés de documents de tailles similaires. Dans le cas de corpus non homogènes, il peut être intéressant de procéder à une normalisation du poids. Pour cela une normalisation est incorporée à la formule du poids **[SAL, 87]**

$$poids_i(j) = tf_{ij} \times idf_j / \sum_{k=1}^{t} tf_{ik}.idf_k \qquad (5)$$

$$\text{Ou } poids_i(j) = tf_{ij} * idf_j / \sqrt{\sum_{k=1}^{t} (tf_{ik}.idf_k)^2} \qquad (6)$$

Avec : $\sum^{k} tf_i .Idf_{ik}$: la somme des poids des termes du document *j*
t : le nombre de termes dans le document. Il existe d'autres types de normalisation développés dans **[SAL, 87]**.

1.2. Appariement document-requête :

L'objectif du SRI est le calcul de la pertinence d'un document par rapport à une requête, c'est une fonction d'appariement qui détermine le degré de ressemblance d'un document par rapport à une requête, et permet éventuellement de classer les documents par ordre de pertinence. Cette fonction est notée rsv (q, d) (Retrieval Status Value), où q est une requête et d est un document de la collection.

La fonction d'appariement est indépendante de l'indexation et de la pondération des termes, par contre elle caractérise le SRI plus que le modèle d'indexation : la plupart des approches de recherche inspirent leurs noms à partir de la façon dont ils entament l'appariement.

1.3. Reformulation de la requête : avant évaluation

La reformulation de la requête consiste à exprimer ou transformer le besoin utilisateur en une information. De nos jours retrouver des informations pertinentes en utilisant la seule requête initiale de l'utilisateur est aujourd'hui quasi-impossible, et ce à cause du volume croissant des bases documentaires. Afin de faire correspondre au mieux la pertinence utilisateur et la pertinence du système, une étape de reformulation de la requête est souvent utilisée.
La reformulation de la requête se fait en deux étapes principales : trouver des termes d'extension à la requête initiale, et répondre les termes dans la nouvelle requête. La reformulation de la requête peut être automatique ou manuelle :

- **La reformulation manuelle :**

Il s'agit de la stratégie de reformulation de la requête la plus populaire. On la nomme aussi ***réinjection de la pertinence*** ou ***relevance feedback***.
L'idée principale de la réinjection de pertinence est de sélectionner les termes importants appartenant aux documents jugés pertinents par l'utilisateur, et de renforcer l'importance de ces termes dans la nouvelle formulation de la requête.

- **La reformulation automatique:**

Dans ce type de reformulation l'utilisateur n'intervient pas.
L'extension de la requête est faite à partir d'un thésaurus qui définit les relations entre les différents termes de l'index et permet de sélectionner de nouveaux termes à ajouter à la requête initiale.

Le thésaurus regroupe plusieurs informations de type linguistique (équivalence, association, hiérarchie) et statistique (pondération des termes).

VI. Les modèles de la recherche d'information :

1. Objectif des modèles :

Il s'agit de construire des formalismes permettant :

- de représenter les documents et les requêtes,
- de calculer la similarité entre un document et une requête ainsi représentés.

2. Principaux modèles :

De façon générale, les modèles de RI peuvent être classé en trois principales classes ou modèles qui sont :

- **les modèles ensemblistes :**

Ils sont bases sur la théorie dans des ensembles, ainsi une requête est représentée par un ensemble de termes sépares par des opérateurs logiques (OR, AND, NOT). Le document est représenté par une liste de mots-clés. Ces modèles permettent d'effectuer des opérations d'union, d'intersection et de différence lors de l'interrogation, et sont les premiers à avoir été utilisés. On peut citer le modèle booléen.

- **les modèles algébriques :**

Les modèles algébriques regroupent tous les modèles de RI qui utilisent une représentation vectorielle des documents et des requêtes,

Dans ces modèles, la pertinence d'un document vis-à-vis d'une requête est définie par des mesures de distance dans un espace vectoriel.

- **les modèles probabilistes :**

Reposant sur la théorie des probabilités. Pour ces modèles, la pertinence d'un document vis-à-vis d'une requête est vue comme une probabilité de pertinence document/requête.

La figure 4, inspirée de **[MOR, 06]** et **[TEB, 04]**, illustre les principaux modèles de RI existant repartis sur les trois courants cites ci-dessus :

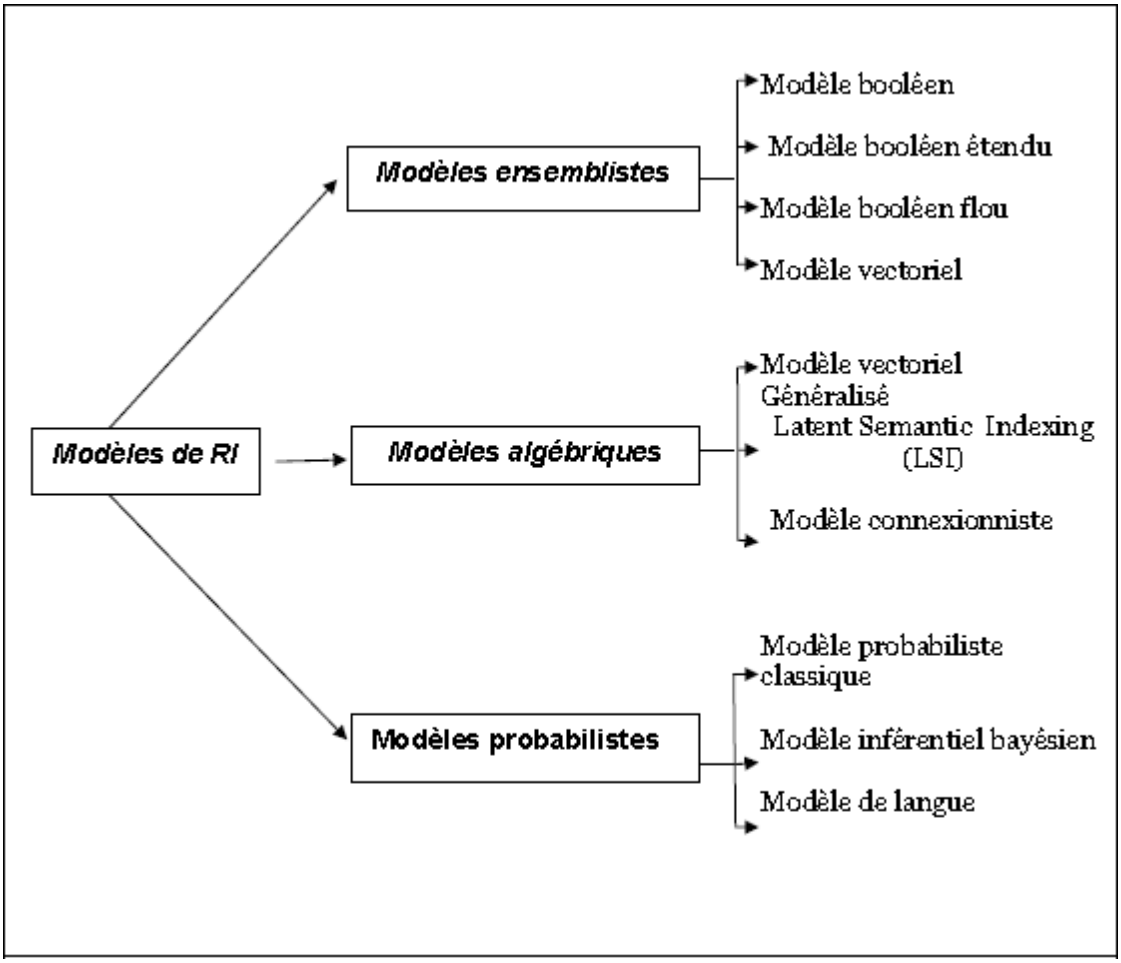


Figure 4: Les principaux des modèles de RI.

Nous décrivons dans la suite, de manière générale, un exemple de modèle issu de chacun de ces trois groupes.

2.1 Le modèle booléen de base :

Dans ce modèle un document D est représenté comme une conjonction logique des termes (non pondéré), par exemple D=t1^ t2^ t3^…^ tn.
Une requête est une expression logique quelconque des termes. On peut utiliser les opérateurs (et ^), (ou v) et (non ¬).

Exemple :

Q= (t1^ t2) v (t3^¬t4).

Pour qu'un document corresponde à une requête il faut que l'implication D =>Q soit valide cette évaluation peut être défini de la manière suivante : un document est représenté comme un ensemble de termes, et une requête comme une expression logique de termes.

La similarité RSV (Q, D) entre une requête et un document est déterminée de la façon suivante :

RSV (ti, D)=1 si ti ^ D ; 0 sinon,
RSV (Qi ^ Qj, D)=1 si RSV (Qi, D)=1 et RSV (Qj, D)=1 ; 0 sinon,
RSV (Qi ^ Qj, D)=1 si RSV (Qi, D)=1 ou RSV (Qj, D)=1 ; 0 sinon,
RSV (¬Qi, D)=1 si RSV (Qi, D)=0 ; 0 sinon.

- **Avantage :**

Simple à mettre en œuvre.

- **Inconvénients :**

Les documents pertinents dont la représentation ne correspond qu'approximativement à la requête ne sont pas sélectionnés.

Mais encore deux autres inconvénients sont dus au fait que :
(1) dans ce modèle, tous les termes ont la même importance.
(2) ce modèle est incapable de trier les documents pertinents.

2.2 Le modèle vectoriel (Vector Space Model) :

Dans ce modèle, les requêtes et les documents sont représentés dans l'espace vectoriel engendre par les termes d'indexation **[SAL, 83].** L'espace est de dimension N (N étant le nombre de termes d'indexation de la collection de documents).

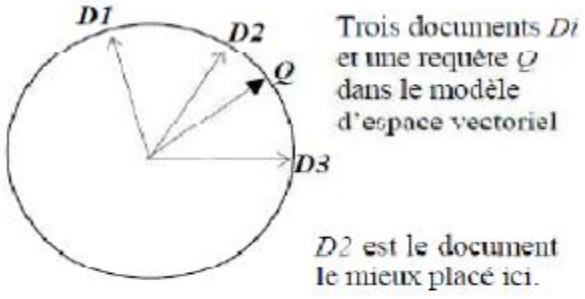


Figure 5 : Le modèle vectoriel **[SAU, 05]**

Le système SMART est base sur ce modèle. Chaque document est représenté par un vecteur (voir schéma de la Figure 5) :
$Dj = (d1j, d2j, d3j, \ldots, d_Nj)$,
Chaque requête est représentée par un vecteur :
$Q = (q1, q2, q3, \ldots, q_N)$,
Avec dij : poids du terme ti dans le document Dj
qi : poids du terme ti dans la requête Q.

La pondération des composantes de la requête est soit la même que celle utilisée pour les documents, soit donnée par l'utilisateur lors de sa formulation.
Le mécanisme de recherche consiste à retrouver les vecteurs documents qui s'approchent le plus du vecteur requête. Les principales mesures de similarité utilisées sont

Produit scalaire :

$$RSV\ (Q, D_j) = \sum_{i=1}^{N} q_i . d_{ij}$$

Mesure de Jaccard :

$$RSV\ (Q, D_j) = \frac{\sum_{i=1}^{N} q_i . d_{ij}}{\sum_{i=1}^{N} q_i^{\,2} + \sum_{i=1}^{N} d_{ij}^{\,2} - \sum_{i=1}^{N} q_i . d_{ij}}$$

La mesure cosinus :

$$RSV\ (Q, D_j) = \frac{\sum_{i=1}^{N} q_i . d_{ij}}{\left(\sum q_i^{\,2}\right)^{1/2} \cdot \left(\sum d_{ij}^{\,2}\right)^{1/2}}$$

- **Avantage :**

La mesure de similarité permet d'ordonner les documents selon leurs degrés de pertinence vis-à-vis de la requête.

- **Inconvénients :**

La représentation vectorielle suppose l'indépendance entre les termes, ce qui n'est pas souvent le cas, un concept étant souvent représenté par un groupe de mots. En outre, l'espace à considérer possède une grande dimension.

2.3 Modèles probabilistes :

Les modèles probabilistes abordent le problème de la recherche d'information dans un cadre probabiliste. Le principe de base consiste à présenter les résultats retournes par le SRI dans un ordre base sur la probabilité de pertinence d'un document vis-à-vis d'une requête **[SAU, 05].**

Il fonctionne selon le principe suivant :

Soient R et NR représentant respectivement la pertinence et la non pertinence (ou de façon équivalente, l'ensemble de documents pertinents et l'ensemble de documents non pertinents) pour une requête donnée Q.

L'idée de base est de tente de déterminer les probabilités P(R /D) et P (NR/D). Ces deux probabilités signifient respectivement : si on trouve le document D, quelle est la probabilité qu'on obtient l'information pertinente et non pertinente respectivement. On définit une fonction
O(D) telle que :

O (D) = P (R/D) / P (NR/D)

Plus O(D) est élevé pour un document, plus ce document doit être classé en haut lors de l'acquisition de résultats.

- **Le modèle Okapi BM25 :**

La fonction de pondération BM25 est part de modèle probabiliste de base, elle est traitée par **Robertson et al en 1994** qui ont adapté la formule de BM25 afin de tenir compte de la structure des documents HTML **[KHA, 08]**. Les éléments de la structure sont pondérés suivant leur importance. Par exemple, ils considèrent le titre, le « heading », et par suite leur affectent un poids supérieur à celui du body **[GEO, 07]**. Ainsi, dans BM25, le poids de chaque terme est calculé à partir du nombre d'occurrences du terme dans le document, de la distribution du terme dans le reste du corpus, de la taille du document, etc.…

Lors du traitement d'une requête, la pertinence d'un document est évaluée par la somme des poids BM25 des termes de la requête qu'il contient.

Etant donné un terme t, une requête q est un document d, le poids w de d selon q et le terme est calculé par la fonction suivante **[KHA, 08]** :

$$w = \frac{(k_1+1)\times tf}{k+tf} \times log\frac{N-n+0.5}{n+0.5} \times \frac{(k_3+1)\times qtf}{k_3+qtf} \oplus k_2 \times nq \times \frac{avdl-dl}{avdl+dl}$$

Le tableau1 illustre les variables de la fonction BM25 :

Variables	Description
N	Le nombre total de documents dans la collection
n	Le nombre de document contenant le terme t, (n<N)
tf	Fréquence du terme t dans le document d
qtf	Fréquence du terme t dans la requête q
nq	Nombre de terme dans la requête q
dl	Longueur du document d
$avdl$	Longueur moyenne des documents dans la collection
k	$k_1 \times \left(1_b + b \times \frac{dl}{avdl}\right)$
k_1, k_2, k_3, b	Parameters constants

Table1 : Paramètres de la fonction BM25.

- **Avantage:**

Les documents retrouvés sont classés selon l'ordre de pertinence décroissant.

- **Inconvénients :**

Difficulté de calcule de probabilités conditionnelles Les jugements de pertinence étant propres a un utilisateur particulier, l'apprentissage effectue à partir de ces données ne sont pas valables que pour cet utilisateur.

VII. Evaluation de SRI :

La démarche de validation en RI se base sur l'évaluation expérimentale des performances du modèle ou du système proposé. L'évaluation des performances d'un modèle de RI, permet de paramétrer le modèle, d'estimer l'impact de chacune de ses caractéristiques et de fournir des éléments de comparaison entre modèles. Cette évaluation peut porter sur plusieurs critères : le temps de réponse, la pertinence, la qualité et la présentation des résultats, etc. Le critère le plus important est celui qui mesure la capacité du système à satisfaire le besoin en information de l'utilisateur, c'est à dire la pertinence. Deux facteurs permettent d'évaluer ce critère. Le premier est le taux de rappel, il mesure la capacité du système à sélectionner tous les documents pertinents. Le second est le taux de précision, il mesure la capacité du système à rejeter tous les documents non pertinents. On calcule :

$$Taux\_rappel = R_d = \frac{\text{Nombre de documents pertinents restitués}}{\text{Nombre de documents pertinents}}$$

$$Taux\_précision = P_d = \frac{\text{Nombre de documents pertinents restitués}}{\text{Nombre de documents restitués}}$$

Ce type de mesure est effectué sur des collections de tests. Une collection de tests est composée d'un ensemble de documents, d'un ensemble de requêtes et d'un ensemble de jugements de pertinence.

L'initiative la plus importante actuellement pour la construction de collections de tests est sans conteste TREC (Text REtrieval Conference, http://trec.nist.gov) **[Harman 92]**. TREC est plus qu'une collection de tests, c'est un programme d'évaluation des SRI, initié par le NIST (National Institute of Standards and Technology) aux USA. TREC fournit une plate-forme comportant des collections de tests, des tâches spécifiques et des protocoles d'évaluation pour chaque tâche, pour l'évaluation et la comparaison d'expérimentations sur des collections volumineuses de textes. Il faut noter que les collections TREC représentent aujourd'hui un référentiel incontournable en RI [TREC-9 00].

Conclusion :

Dans ce chapitre, nous avons présenté un état de l'art sur la recherche d'information classique. Nous y avons abordé la définition et l'architecture générale d'un SRI, les techniques d'indexation utilisées en RI, les principaux modèles de recherche, ainsi, que l'évaluation adoptées pour attester de la qualité d'un SRI.

CHAPITRE 2

La recherche d'information sociale et la fraicheur d'information

I. Introduction :

La croissance d'Internet a permis de former différents types de réseaux sociaux(RS) à grande échelle. L'explosion des RS a permis l'émergence d'une nouvelle branche de la Recherche d'Information (RI) : la RI sociale (RIS). Il s'agit d'adapter les modèles et les algorithmes de la RI classique afin d'exploiter les informations sociales propres à ce nouveau cadre.

Dans le présent chapitre nous nous focalisons sur l'évaluation de la fraicheur d'informations, avec un objectif de présenter les différents travaux de recherche sur la fraicheur d'information ainsi que l'intégration de la fraicheur d'information dans la RIS.

II. La notion d'information :

L'information est un concept ayant plusieurs sens. Il est étroitement lié aux notions de contrainte, communication, contrôle, donnée, formulaire, instruction, connaissance, signification, perception et représentation.

L'information désigne à la fois le message à communiquer et les symboles utilisés pour l'écrire ; elle utilise un code de signes porteurs de sens tels qu'un alphabet de lettres, une base de chiffres, des idéogrammes ou pictogrammes.

Au sens étymologique, *l'information* est ce qui donne une *forme* à l'esprit. Elle vient du verbe Latin informare, qui signifie "donner forme à" ou "se former une idée de".

Hors contexte, elle représente le véhicule des données comme dans la théorie de l'information et, hors support, elle représente un facteur d'organisation.

On touche là à un sens fondamental, où l'information est liée à un projet. Il peut être construit, comme un programme, ou auto-construit, comme la matière.

1. Informations sociales :

Avec l'explosion des technologies du Web 2.0, les utilisateurs du Web produisent divers contenus, créent des annotations, manipulent les documents sur le Web et laissent des traces de leurs passages, etc. Par exemple, les *folksonomies* représentent des informations d'une grande importance. La plupart des RS sont modélisés par des structures de graphe social dont les nœuds sont les utilisateurs du réseau et les arcs représentent les relations entre ces utilisateurs. Dans le cas des *folksonomies*, la structure sous-jacente peut être

considérée comme un graphe tripartite : utilisateurs, annotations et nœuds de ressources. On observe dans les RS l'existence des informations suivantes :

- Les bookmarks : un espace personnel de stockage de l'information du Web [**ABC ,98]**

  - les tags, utilisés pour annoter des bookmarks, des pages Web, des images, etc.
  - les traces des utilisateurs (navigation sur le Web, visualisation des pages Web et documents, etc.). L'exploitation de ces traces permet également d'extraire des informations, éventuellement thématiques, sur les préférences des utilisateurs.
  - les relations entre utilisateurs au sein du réseau social : amis, co-auteurs, etc.

D'une manière générale, ces informations sociales peuvent être exploitées pour améliorer l'évaluation des ressources sur le Web par rapport à un thème donné suite aux retours positifs que donne l'utilisateur.
Elles peuvent être prises en compte dans un modèle de RI sociale à différents niveaux.

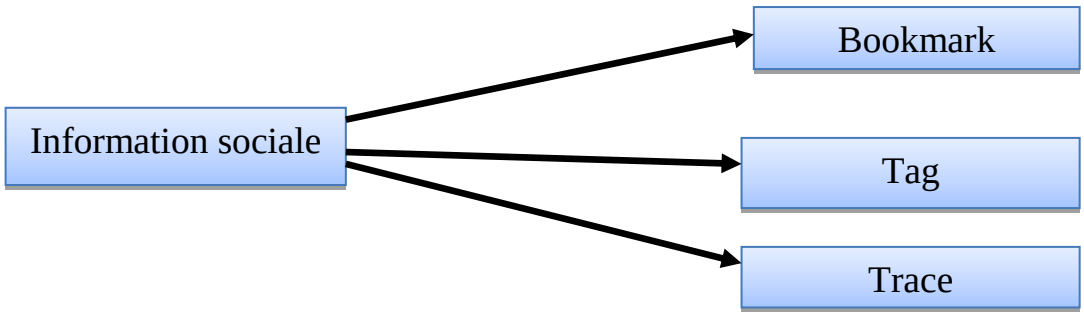


Figure 5 : les informations sociales

Parmi ces informations sociales on s'intéresse aux tags :

1.1. Les tags :

Selon [**Golder&Huberman]**, le terme « tag » (en français « étiquette ») représente un mot-clé ou une expression associée ou assignée aux ressources. Il décrit ainsi l'objet et lui permet d'être retrouvé par navigation, par filtrage ou par recherche. L'activité des individus consistant à attribuer des métadonnées

aux ressources s'appelle «tagging». Ces dernières années, ce processus a gagné beaucoup en popularité sur le Web.

L'ensemble des tags d'un individu forme une collection qui s'appelle « personomy ». L'ensemble des personomies constitue la «folksonomy». Elle est constituée par l'effort collectif des utilisateurs qui ont diverses connaissances et besoins quand ils interagissent avec un système de *social bookmarking*. De plus, le terme *folksonomy* est une combinaison de «folk» et «taxonomy», décrivant le phénomène de classification sociale, les ressources, les utilisateurs, les tags, ainsi que l'activité d'un individu. Cette activité consiste à créer une association entre les utilisateurs et les ressources à travers des étiquettes.

1.1.1. Les plates-formes à base de tags :

Les systèmes de tagging sociaux sont généralement utilisés pour faciliter l'indexation collaborative de quantité massive d'informations et améliorer leur accès **[MIL. 06]**. Par exemple, les services en ligne de gestion des références de bookmarking social Connotea2 et CiteULike3 sont utilisés par les chercheurs, scientifiques et académiciens pour stocker, organiser, partager et découvrir des liens vers des papiers académiques et de recherche. Le système ASK – LOST 2.0 **[KAL & AL. 09]** propose d'utiliser les tags pour indexer tous types de ressources pédagogiques digitales (images, vidéos, textes, URL). Les tags sont également utilisés comme outil d'indexation et de recherche d'informations dans des communautés d'enseignants, comme le propose le site Web Couldworks **[CON & CUL 10]** créé pour les enseignants afin de discuter de leurs pratiques et idées de design pédagogique. L'outil de partage de signets SemanticScuttle **[HUY, 09]** propose également aux communautés de structurer leurs tags en créant des relations explicites d'inclusion et de synonymie entre tags.

Certains travaux s'intéressent aux tags à d'autres fins pédagogiques que la simple indexation et recherche d'information. Les outils d'annotation sociale OATS **[BAT & AL. 07]** et SparTag.us **[NEL & AL. 09]** permettent notamment de surligner des passages de textes et de leurs associer des tags. Pour chaque passage annoté, l'utilisateur peut voir les tags qu'il a associés ainsi que les tags les plus associés par les autres utilisateurs. De façon plus globale, il est possible de voir le nuage des tags les plus utilisés.

Cependant, ces outils ne permettent pas un travail collaboratif, chaque étudiant annotant individuellement le texte. Il n'y a par exemple pas de création collective de tags, de négociation ou de critique.

[CHE & AL. 10] proposent le système collaboratif en ligne TACO, basé sur les tags, comme support à l'apprentissage des langues. Ce système a été conçu pour améliorer la compréhension de l'anglais écrit et plus précisément pour développer des capacités de pensée critique, un forum étant associé à chaque tag pour permettre aux apprenants de les critiquer et échanger des idées (tous les tags étant visibles par tous les apprenants). Dans ce système, l'accent est surtout

mis sur l'aide apportée aux enseignants pour évaluer précisément les étudiants à partir de leurs tags grâce à un mécanisme de score automatique.
Seul l'environnement TACO est utilisé à de réelles fins d'apprentissage collaboratif, avec négociation et critique des tags(Figure6), mais il est très spécifique à l'apprentissage des langues.

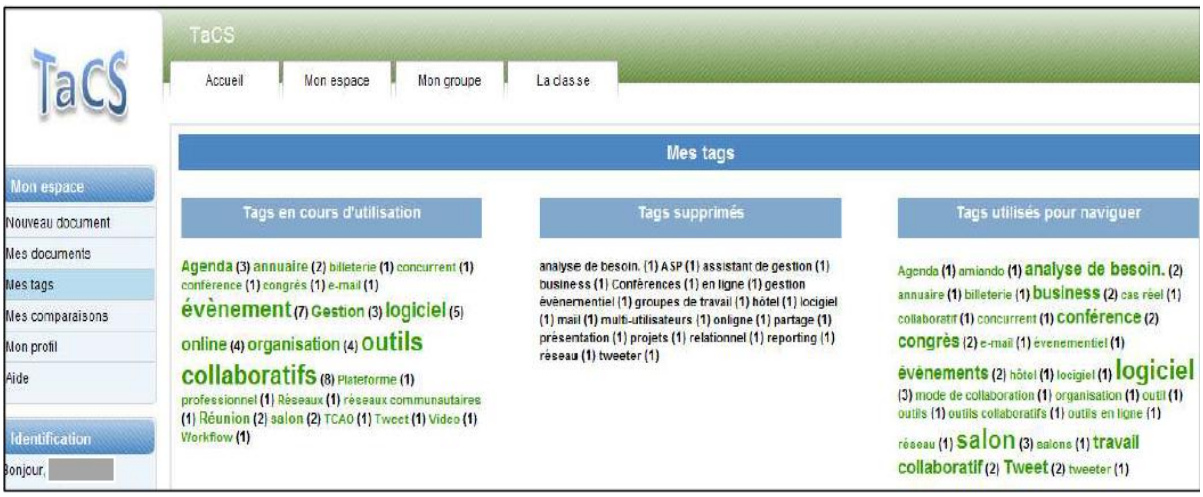


Figure 6 : *Nuages de tags*

2. La qualité d'une information:

La qualité de l'information a fait l'objet de nombreuses études mettant en évidence différents facteurs de qualité dont les définitions sont souvent nuancées par l'usage de termes aussi variés que catégorie, dimension, attribut, critère, mesure, paramètre, etc. S'il est souvent possible de rapprocher certaines notions, la standardisation des concepts et du vocabulaire reste à faire. Outre la définition des concepts, l'évaluation de la qualité constitue un des principaux problèmes de recherche; elle concerne la spécification et l'implémentation des procédures de mesure des différents facteurs caractérisant les sources de données ainsi que la combinaison de ces mesures pour obtenir une valeur agrégée interprétable par l'utilisateur.

Certaines approches se focalisent sur la définition de facteurs, par exemple **[RED, 96]**. Par contre, les facteurs définis sont pertinents uniquement dans le domaine pour lequel ils ont été conçus, laissant relativement peu de possibilités de réutilisation ou de définition d'un sous-ensemble cohérent adapté à une application donnée. Certains travaux analysent et classifient les techniques de mesure, les métriques et les fonctions d'agrégation **[PIP, 02], [NAU, 00] , [BAL, 98]**. D'autres travaux montrent leur usage dans des contextes applicatifs précis comme, par exemple, la fraîcheur des données **[CHO, 00].**

Il est nécessaire de combiner plusieurs valeurs d'un facteur de qualité (mesurées pour les différentes sources) pour obtenir une valeur caractéristique

du résultat de la requête. Par exemple, dans une requête qui intègre des données de différentes sources, la disponibilité des résultats peut être calculée comme une fonction de la disponibilité des différentes sources. Malgré l'existence de techniques pour mesurer certains facteurs de qualité, la combinaison de valeurs a été peu traitée dans la littérature. Dans **[NAU, 99]**, les auteurs proposent de combiner les valeurs de qualité des sources via des opérateurs arithmétiques (minimum, maximum, moyenne, somme, produit). D'autres travaux analysent la combinaison de valeurs de qualité pour des facteurs spécifiques, par exemple l'exactitude et la complétude des données **[MOT & AL, 98].**

D'autres paramètres du contexte d'application, comme la nature des données, le domaine d'application, l'architecture du système, etc., peuvent influencer la qualité des résultats **[PER, 06].** Par exemple, les coûts de traitement des données influencent le temps de réponse et la fraîcheur des données. De plus, différents utilisateurs peuvent avoir besoin de différentes stratégies de combinaison de valeurs de qualité. Par exemple, un utilisateur peut exiger que toutes les données soient exactes alors qu'un autre peut tolérer un certain taux d'erreur sur certains attributs (calcul d'une moyenne de l'exactitude).

2.1. Quelques expériences de recherche sur la qualité de l'information:

Les premiers travaux réalisés concernent l'évaluation de la qualité des publications académiques, en particulier la qualité scientifique, la lisibilité, la fraîcheur des données contenues, le degré d'autorité, la disponibilité, la popularité et la qualité de l'identification. Ils ont donné lieu à l'élaboration d'un modèle exhaustif **[DEN, 00]** qui a été mis en regard d'un contexte applicatif précis (archives de prépublication) pour établir la faisabilité pratique de l'exploitation de ces facteurs de qualité**[DEN, 00]**. Suite à cette mise en œuvre, le retour des utilisateurs a été collecté, analysé et pris en compte pour affiner le modèle opérationnel de qualité pour cette application **[DEN, 02].**

Un canevas a été développé par **[BOU & AL, 04]. et [Per, 06]**, il fournit un support formel pour l'évaluation de la qualité des données, en particulier la fraîcheur et l'exactitude des données. Ce canevas permet d'analyser les différentes définitions et métriques de ces facteurs, d'analyser les paramètres du système d'information qui les influencent, et de développer des algorithmes d'évaluation qui tiennent compte de ces paramètres.

Des critères de qualité d'un document ont été proposés par **[CAL ,98]** s'appuyant sur les critères de qualité des données. Ces critères sont : facilité d'exploitation, crédibilité, traçabilité, compréhensibilité, réutilisabilité, portabilité et flexibilité.

[HAR, 06] suggère un modèle de qualité hiérarchique orienté utilisateur. Cette hiérarchie compte trois niveaux :

(1) niveau système qui décrit les métriques de qualité calculables au niveau du système ;

(2) niveau utilisateur qui décrit les facteurs de qualités désirées par l'utilisateur qui sont calculés ou agrégés à partir du métrique système ;

(3) niveau source de la qualité qui classifie ces facteurs dans 3 catégories (sources, support et usage). Par exemple, la fréquence de mise à jour (métrique système) s'utilise pour calculer la fraîcheur (facteur utilisateur) qui est un facteur de la catégorie support**.**

2.2. Evaluation de la fraîcheur d'information :

Intuitivement, le concept de fraîcheur introduit l'idée d'âge des données : Les données sont-elles suffisamment fraîches pour les utilisateurs ? Les données sont-elles obsolètes ? Une certaine source de données a-t-elle les données les plus récentes ? Quand les données ont-elles été produites ?

La fraîcheur représente une famille de facteurs de qualité, chacun représentant un certain aspect de fraîcheur et ayant ses propres métriques. Nous distinguons deux facteurs de fraîcheur :

2.2.1. Actualité [SEG & AL, 90]:

L'actualité mesure la distance ou le décalage entre l'extraction et la livraison de données (la date actuelle), voir la formule suivante:

Actualité (information)=date_livraison – date_extraction

Par exemple, en regardant un solde bancaire, nous voulons savoir le moment où il a été extrait de la banque, peu importe quand il a été mis à jour.

2.2.2. Age [WAN & AL, 96]:

L'Age mesure la distance ou le décalage entre la création (ou mise à jour) et la livraison des données. Il est indépendant du moment de l'extraction des données. Par exemple, si nous obtenons une recommandation des 10 meilleurs CDs, nous nous intéressons à la date de création de cette liste, peu importe quand elle a été extraite.

Age (information)=date_livraison – date_création

- **Les métriques de la fraîcheur d'information :**

La figure 1 montre les métriques de fraîcheur proposées dans la littérature ; une étude détaillée est présentée dans **[BOU& AL, 04].**

Facteur	**Métrique**	**Définition**
Actualité	Actualité	Le temps passé depuis l'extraction des données (borné par la fréquence d'extraction)
	Obsolescence	Le nombre de transaction de mise ajour d'une source depuis l'extraction des données
	Ratio de fraîcheur	Le taux de données qui sont à jour
Age	Age	Le temps passé depuis la création ou la mise à jour des données (borné par de mise à jour)

Table 1 : Métriques de fraîcheur d'information

Il existe d'autres travaux sur la fraîcheur d'information tels que :

2.2.3. Algorithme de la fraîcheur de Peralta : [PER & AL, 04]

Le temps passé depuis la création des données. La fraîcheur des données produites dépend de la fraîcheur réelle des données sources (actualfreshness), des coûts d'exécution des opérations de la requête (cost) et les délais que peuvent exister entre l'exécution des opérations (delay) engendrés par exemple par la synchronisation entre ces opérations.

Peralta considère les propriétés suivantes et estime la fraîcheur atteinte par chaque nœud du graphe, en utilisant les règles suivantes :

- Pour chaque nœuds source A: Freshness (A) = SourceActualFreshness (A)
- Pour chaque nœud non source A et l'ensemble de ses prédécesseurs P:

$$\text{Freshness (A)} = \text{combine } \{\text{Freshness (B)} + \text{Delay (B, A)}\} + \text{Cost (A)}$$

Pour les nœuds source la fraîcheur de données est la fraîcheur réelle des données source. Pour les autres nœuds, la fraîcheur des données produites est calculée comme la fraîcheur des données d'entrée à laquelle on ajoute le délai et le coût.

Quand un nœud a plusieurs prédécesseurs, la valeur de fraîcheur d'entrée est dérivée en utilisant une fonction de combinaison spécifique ; par exemple la valeur maximum parmi des valeurs d'entrée.

Les résultats de l'évaluation peuvent être utilisés d'une façon informative, pour avoir une mesure de la qualité des données produites par chaque requête et éventuellement comparer et trier les requêtes par rapport aux facteurs de qualité auxquels l'utilisateur s'intéresse.

[PER & AL, 04] implémentent un outil d'évaluation de la qualité qui permet de choisir les propriétés les plus pertinentes pour une application donnée, associer des propriétés aux requêtes, incorporer dynamiquement des nouveaux algorithmes d'évaluation et les exécuter. La Figure7 montre l'interface de l'outil.

Dans la partie gauche, on peut gérer les différents composants du cadre de travail lequel inclut un catalogue de facteurs de qualité avec leurs métriques, des requêtes utilisateur, des sources de données, des algorithmes d'évaluation et des requêtes. Dans la partie droite, on peut gérer les requête, acquérir et modifier des valeurs des propriétés, évaluer la qualité en exécutant des algorithmes et faire apparaître les requêtes pour lesquelles les besoins des utilisateurs ne peuvent pas être satisfaits. L'outil permet d'évaluer en parallèle la qualité de plusieurs requêtes de médiation. Il présente les résultats sous forme graphique, permettant ainsi de comparer les valeurs de qualité obtenues avec les préférences des utilisateurs exprimés dans leurs profils.

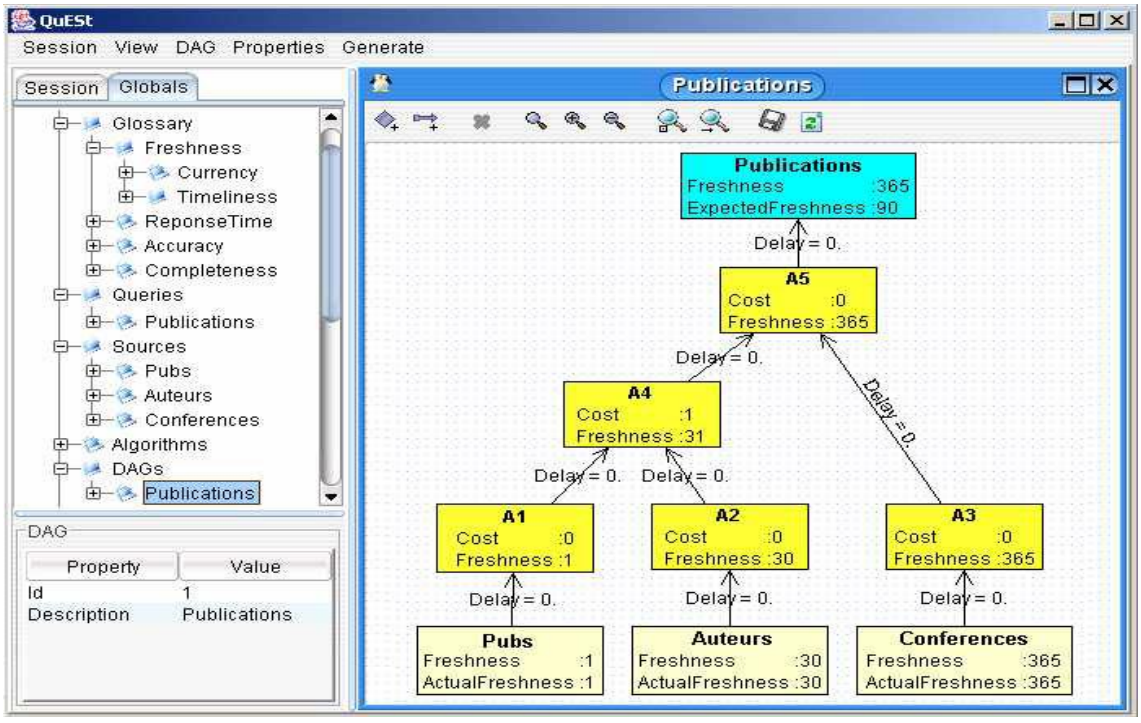


Figure 7**:**Outil d'évaluation de la qualité.

2.2.4. Fraîcheur d'information [HUO,] :

Une interprétation simple de fraîcheur est la première date où l'article a été signalé.

Fraîcheur (info)=première date où l'article a été signalé

2.2.5. Fraîcheur de la source : [Lynda tamine,]

La fraîcheur d'un document est définie par son âge calculé comme la période séparant sa date de création de la date (instant) d'évaluation de la requête. Dans cette optique, on relativise la fraîcheur de la source d'information par rapport au profil de la requête en cours d'évolution, en ce sens que la fraîcheur de la source est une grandeur relative et non absolue, dépendant du besoin en information courant. Plus précisément, soit une source d'information *Si*, on caractérise sa fraîcheur *Ht*(*q*) à la période *t* relativement à la requête *q*, en calculant la densité temporelle des *n* premiers documents retournés soit $D_{top}(q)$ comme suit :

$$H_t(q) = \frac{|< D^t, D_{top}(q) >|}{|S_i^t|}$$

Où S_i^t : est le nombre total de documents datés de la période *t*,
$<D_t, D_{top}>$: est le nombre de documents datés de la période *t* parmi les documents D_{top} retournés par la requête *q*. comme montrer dans la **figure**

Dans le but de caractériser la distribution de la fraîcheur de l'information permettant de déterminer la sensibilité de la requête à ce facteur considérant le contenu de la source *Si*, on calcule une mesure de l'assymétrie de la distribution temporelle de la densité H_t comme suit :

$$Ass_i(Q) = Skew(H_t(q))$$

Où *Skew* est une fonction statistique qui est utilisée pour mesurer l'assymétrie d'une distribution de probabilités, soit :

$$Skew\,(Y_1, Y_2, \ldots, Y_l) = \frac{\sum_{i=1}^{l}(Y_i - \overline{Y})}{(l-1)\times s^2}$$

Avec *s :* est la valeur de déviation standard, égale à 0.2
Y: est la moyenne des valeurs Y_i ($i = 1 \ldots l$).

Lorsque la valeur d'Assi (q) est élevée, la source d'information *Si* est sensible à la fraîcheur de l'information en accord avec la requête *q*. Le nombre de pics observés au niveau de chaque source devient alors un critère pour leur sélection.

- **Exemple :**

Pour n = 10

Si =20

$\overline{Ht}(q)=(0+0.6+0.5+0.75+0.65+0.55+0.7+0.95+0.85+1)/9=0.405$

$\sum Ht(q)- \overline{Ht}(q)=(0-0.405)+(0.6-0.405)+(0.5-0.405)+(0.75-0.405)+(0.65-0.405)+(0.55-0.405)+(0.7-0.405)+(0.95-0.405)+(0.85-0.405)+(1-0.405)=2.5$

$Skew\ (Ht\ (q)) = \sum Ht(q)- \overline{Ht}(q)\ /(l-1)*S^2=2.5/(10\ -1)*(0.2)^2=2.5/0.36=6.94$

<Dt,Dtop>	0	12	10	15	13	11	14	19	17	20
Si	20	20	20	20	20	20	20	20	20	20
Ht(q)	0	0.6	0.5	0.75	0.65	0.55	0.7	0.95	0.85	1
Skew (Ht(q))	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94
Assi(Q)	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94	6.94

Table 2 : Exemple de fraîcheur de la source d'information

Assi(Q)=Skew (Ht(q))=6.94 est élevée donc la source d'information *Si* est sensible à la fraîcheur de l'information en accord avec la requête *q*.

III. Le Systeme De Recherche D'information Sociale :

1. Définition de la recherche d'information sociale :

- ✓ Il s'agit de localiser des personnes qui savent où trouver l'information sur les documents pertinents à la requête.
- ✓ Il s'agit d'adapter les modèles et les algorithmes de la RI classique afin d'exploiter les informations sociales.
- ✓ La recherche sociale d'information est fondée sur le fait que l'on ne peut pas séparer le producteur d'une information de son produit **[kIR, 05]**.

2. Architecture de système de recherche d'information sociale :

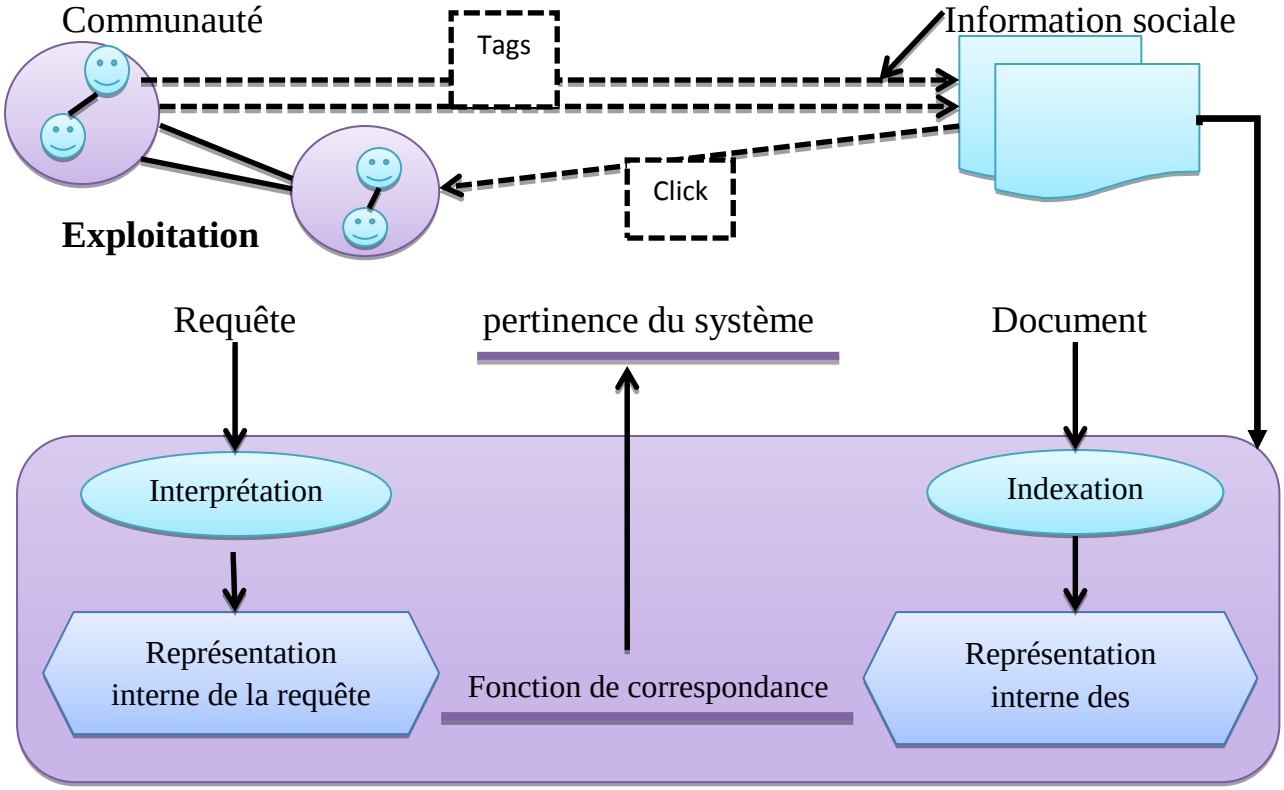


Figure8: *l'architecture de système de recherche d'information sociale.*

IV. La fraicheur d'information dans la RIS :

Il n'existe pas, dans la littérature, beaucoup de travaux sur l'intégration de la fraicheur d'information dans la RIS

1. Le modèle de Huo [Huo,] :

Les travaux précédents dans l'étiquetage social ignorent la plupart du temps l'information temporelle, considérant seulement trois facteurs: *Utilisateur article* et *étiquettes* Huo prolonge le comportement d'étiquetage a ajoutant des horodateurs: *<utilisateur, article, étiquettes, horodateur >* qui indique à l'utilisateur d'annoté un article avec les étiquettes arbitraires à un moment donné

Dans les emplacements d'étiquetage sociaux, les utilisateurs participent généralement aux déférents réseaux sociaux.

a. Les réseaux sociaux globaux :

La signification de chacun se relie avec n'importe qui d'autre sur l'enchaînement entier.

b. Les réseaux sociaux d'amitié :

Un utilisateur peut choisir d'ajouter tous autres utilisateurs comme amis. La plupart d'entre eux pourrait être des connaissances dans la vraie vie amie, camarades de classe, contacts d'affaires, …etc; certains peuvent être connus par Internet.

c. Les réseaux sociaux d'intérêt commun :

Basé sur les comportements d'étiquetage semblables. Les articles signalés et les étiquettes utilisées par un utilisateur peuvent être considérés des indicateurs d'intérêts de cette personne. Enchaînement des personnes ensemble dont en étiquetant les comportements recouvrez de manière significative peut implicitement former réseaux d'intérêt commun.

1.1. Fonctions De Marquage :

1) Score statique :

La fonction de marquage statique globale doit agréger trois composants sociaux de réseau: amitié, réseau d'intérêt commun, et raccordement global.

Étant donné un utilisateur *u*, le score du composant amitié d'un élément *i* pour un *t* tags est défini comme le nombre d'amis des utilisateurs u qui taggés i avec l'étiquette t.

$$S_{CF}(i, u, t) = |Amis(u) \cap \{marquage\ (v, i, t)\}| \qquad (1)$$

De même, les points du réseau d'intérêt commun sont définis comme nombre des liens utilisateurs u qui ont étiqueté *i* avec le tag *t.*

$$S_{CL}(i, u, t) = |Lien(u) \cap \{marquage\ (v, i, t)\}| \qquad (2)$$

Le score *global,* dans l'utilisateur est indépendant, définie comme le total nombre d'utilisateur dans le poste ensemble du site étiqueté i avec le tag t

$$S_{CG}(i, t) = |\{marquage\ (v, i, t)\}| \qquad (3)$$

En conséquence, le score global statique du point i pour l'utilisateur u avec une étiquette *t* est une fonction globale des scores pesés des trois composants:

$$S_{CO}(i, u, t) = w_1 * S_{CG}(i, t) + w_2 * S_{CF}(i, u, t) + w_3 * S_{CL}(i, u, t) \qquad (4)$$

$\sum_{j=1}^{n} W_i$ *est le poids de chaque composant et* $\sum S_{CO}(i, u, tj)=1$

Depuis une requête contient plusieurs balises, le *SCORE* statique globale de l'élément *i* pour utilisateur u avec toute requête *Q* = *t*1,….tn que la somme des scores à partir du tag individuel, qui est la fonction d'agrégation monotone:

$$SCORE\ (i, u) = \sum Sc_O\ (i, u, tj)$$

2. Score Temporel (dynamique) :

Les résultats du classement seront plus attrayants pour les utilisateurs non seulement sur la base de leur pertinence, mais aussi sur la popularité et la fraîcheur. Par exemple, un élément peut être plus intéressant si elle est récemment ajoutée. Dans ce cas, une interprétation simple de fraîcheur est la première date de l'envoi. Cependant, d'une manière plus subtile peut considérer combien les comportements de marquage récentes ont ciblé un point. Approche de base de Huo est de diviser les comportements de marquage en tranches de temps multiples, sur la base de leurs horodatages. Il utilise *m* pour désigner le nombre de tranches de temps et d'ajuster les poids de tranches de temps différentes en fonction de leur récence. Un facteur de décroissance *a* ($0 < a < 1$) est utilisé pour sanctionner le score comptage à partir de tranches de temps anciens. Le score temporel de la composante globale de *i* article avec étiquette *t* peut être défini comme :

$$TSc_G(i,t) = \sum_{s=1}^{m} Sc_G(i,t,s) * a^{m-s}$$

Où : $Sc_G\ (i, t)$est le score global de l'élément *i* avec tag t à *s* tranche de temps, avec *s* = *m* étant la tranche de temps actuelle.

Les fonctions de scoring temporelles pour les *Amis* et les composants sont définis *Liens* même avec les même facteurs dans Global :

$$TSc_F(i,u,t) = \sum_{s=1}^{m} Sc_F(i,u,t,s) * a^{m-s}$$

$$TSc_L(i,t) = \sum_{s=1}^{m} Sc_L(i,u,t,s) * a^{m-s}$$

La fonction de notation temporelle globale de l'élément i et l'utilisateur u avec le tag t est :

$$TSc_O(i,u,t) = w_1 * TSc_G(i,t) + w_2 * TSc_F(i,u,t) + w_3 * TSc_L(i,u,t)$$

Par conséquence, la notation temporelle pour la requête entière est :

$$TSCORE(i,u) = \sum_{j=1}^{n} TSc_O(i,u,t_j)$$

V. Conclusion :

Dans ce chapitre, nous avons présenté :

Premièrement la notion d'information. Nous y avons abordé la définition et les types d'information sociale, la qualité d'information, quelques expériences sur la qualité de l'information, ainsi, qu'une évaluation de la fraicheur d'information.

Deuxièmement un état de l'art sur la **RIS**. Nous avons abordé en premier la définition de la RIS, puis, L'architectures globales de SRIS.

Et enfin on a présenté la fraicheur d'information dans la RIS.

CHAPITRE

3

Conception

I. Introduction :

Ce chapitre est consacré à la description du SRIS que nous allons réaliser, base sur le modèle BM25 de recherche et le modèle de la fraicheur .D'abord nous allons présenter la conception de notre système en spécifions son architecture générale et décrivant le rôle de chacun de ses modèles. Rappelons ici que le principal objectif de ce système est de retourné à l'utilisateur les documents pertinents et frais adapté à ses préférences et ses besoins précis.

II. Description de notre approche :

La solution que nous avons proposée pour améliorer la performance du système de recherche d'information permet de prendre en charge la fraicheur d'information.

Pour cela, on a proposé un modèle de recherche d'information qui comporte plusieurs modules, comme l'indique la figure10 qui décrit son architecture globale. Le premier est consacré à l'indexation des collections de documents ou tags et des requêtes, composés seulement d'une partie textuelle.

Le second module, vise à déterminer, pour une requête donnée, un score classique pour les documents et un score social pour les tags.

Enfin, le dernier module vise à combiner linéairement les deux scores obtenus pour chaque collection de façon à déterminer les documents les plus récents répondant le mieux à la requête.

La formule de score final c'est :

$$\alpha * \text{Score classique} + (1-\alpha) * \text{Score sociale}$$

Tel que $\alpha \in (0.3, 0.5, 0.8)$

Score classique :

$$\text{Score}(D_j) = \sum \left[\log \frac{N}{Df} \right] \cdot \frac{(k1+1)TFj}{k1\,(1-b) + b.\,(\text{Taille}/TF_j)} \cdot \frac{(k3+1)\,tfq}{(k3+tfq)}$$

K1=1.2 ; k3=8 ; b=0.75 ; N=3 ; tfq=1 ;

N : le nombre de documents dans la collection.

Df : le nombre de document ou le terme apparait.

TF_J : la fréquence du terme j dans le document.

Score sociale = 1/p

p=date_actuelle– date_création

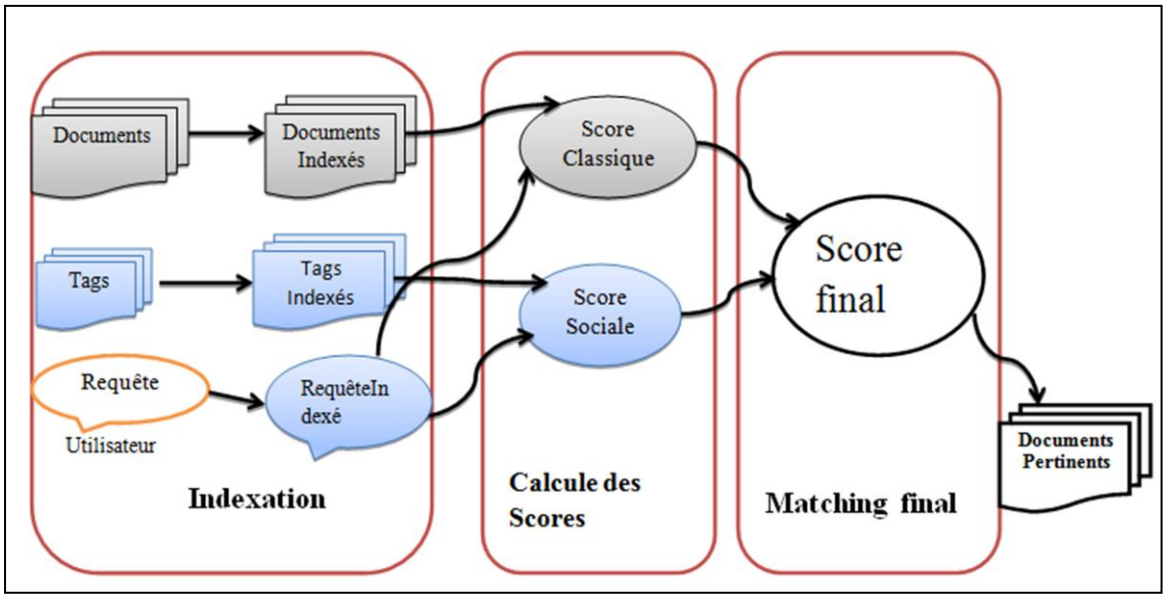


Figure10 : Architecture globale de notre approche

1. **Exemple illustratif :**

En suppose qu'en a trois documents D_1, D_2, D_3 tel que :
$D_1 = (t_1, t_2)$;
Tagdoc1 (t_1, 03/08/2012) ;
$D_2 = (t_2, t_3)$;
Tagdoc2 (t_2, 02/09/2012) ;
$D_3 = (t_2, t_4)$;
$D_4 = (t_3, t_4)$;
Tagdoc4 (t_1, 10/10/2011)

Soit la requête q= (t1, t2, 06/10/2012)

- **Calcule de score classique :**
  - Score classique du D_1=Score(D_1)= 0.024
  - Score classique du D2= Score(D_2)=0.014
  - Score classique du D3= Score(D_3)=0.045
  - Score classique du D4= Score(D_4)=0

- **Calcule de score social :**
  - Score sociale (q, Tagdoc1)=0.46
  - Score sociale (q, Tagdoc2)=0.88
  - Score sociale (q, Tagdoc3)=0
  - Score sociale (q, Tagdoc4)=0.08

- **Calcule de score final :**

On a trois cas :

Cas1 : α=0.3

Score final (q, D_1) =α*score classique (q, D_1) + (1- α)*score social (q, tagdoc1)=0.3*0.024+ (1-0.3)*0.46=0.32

Score final (q, D_2) = α*score classique (q, D_2) + (1- α)*score social (q, tagdoc2)=0.014*0.3+0.7*0.88=0.62

Score final (q, D_3) = α*score classique (q, D_3) +0=0.045*0.3+0.013
Score final (q, D_4) = (α-1)*score social (q, tagdoc4) +0=0.08*0.7+0.056

Le tableau ci-dessous résume les résultats des scores de cas α=0.3

Documents/Tags	Score Social	Score Classique	Score Final
D1	0.46	0.024	0.32
D2	0.88	0.014	0.62
D3	0	0.045	0.013
D4	0.08	0	0.056

Table3 : Exemple illustratif dans le cas α=0.3

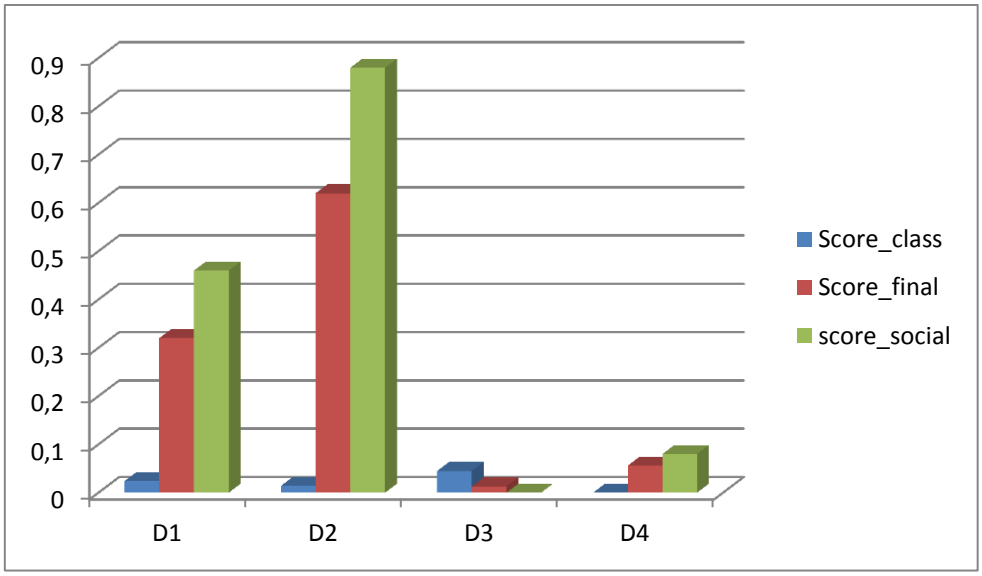


Figure 11: diagramme des scores avec α=0.3

Cas2 α=0.5 :

Score final (q, D_1) =α*score classique (q, D_1) + (1- α)*score social (q, tagdoc1)=0.24

Score final (q, D_2) = α*score classique (q, D_2) + (1- α)*score social (q, tagdoc2)=0.44

Score final (q, D_3) = α*score classique (q, D_3) + 0=0.022
Score final (q, D_4) = (1-α)*social (q, tagdoc4) + 0=0.04

Le tableau ci-dessous résume les résultats des scores maximums de cas α=0.5

Documents/Tags	Score Social	Score Classique	Score Final
D1	0.46	0.024	0.24
D2	0.88	0.014	0.44
D3	0	0.045	0.022
D4	0.08	0	0.04

Table : Exemple illustratif dans le cas α=0.5

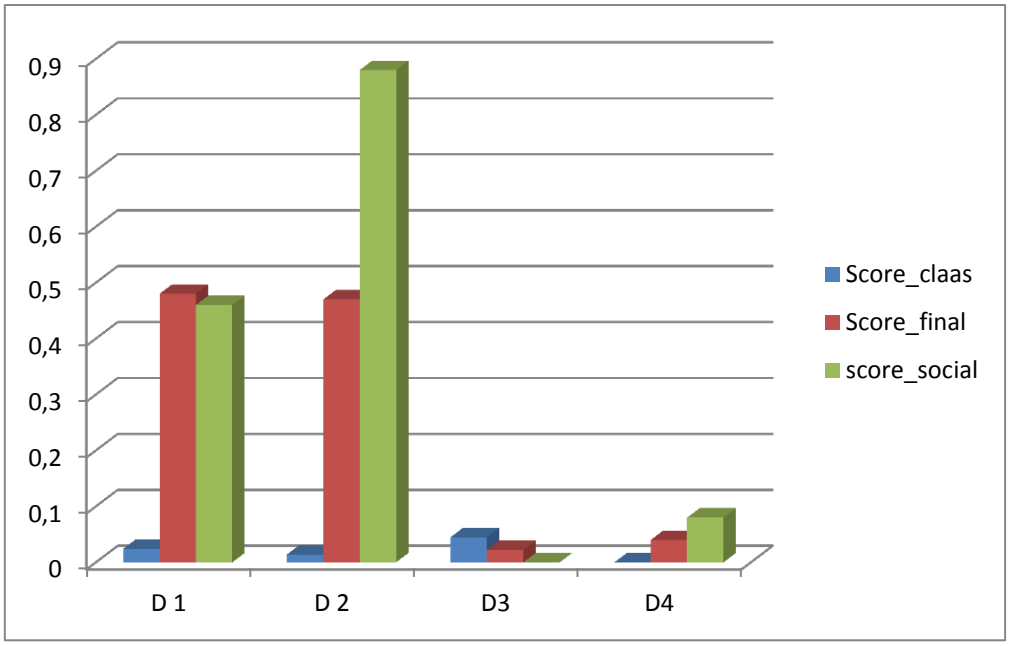


Figure 12: diagramme des scores avec α=0.5

Cas α=0.8:
Score final (q, D_1) =α*score classique (q, D_1) + (1- α)*score social (q, tagdoc1)=0.11

Score final (q, D_2) = α*score classique (q, D_2) + (1- α)*score social (q, tagdoc2)=0.18

Score final (q, D_3) = α*score classique (q, D_3) +0=0.036
Score final (q, D_4) = (1-α)*score social (q, tagdoc4) +0=0.016

Le tableau ci-dessous résume les résultats des scores maximums de cas α=0.8

Documents/Tags	Score Social	Score Classique	Score Final
D1	0.46	0.024	0.11
D2	0.88	0.014	0.18
D3	0	0.045	0.036
D4	0.08	0	0.016

Table: Exemple illustratif dans le cas α=0.8

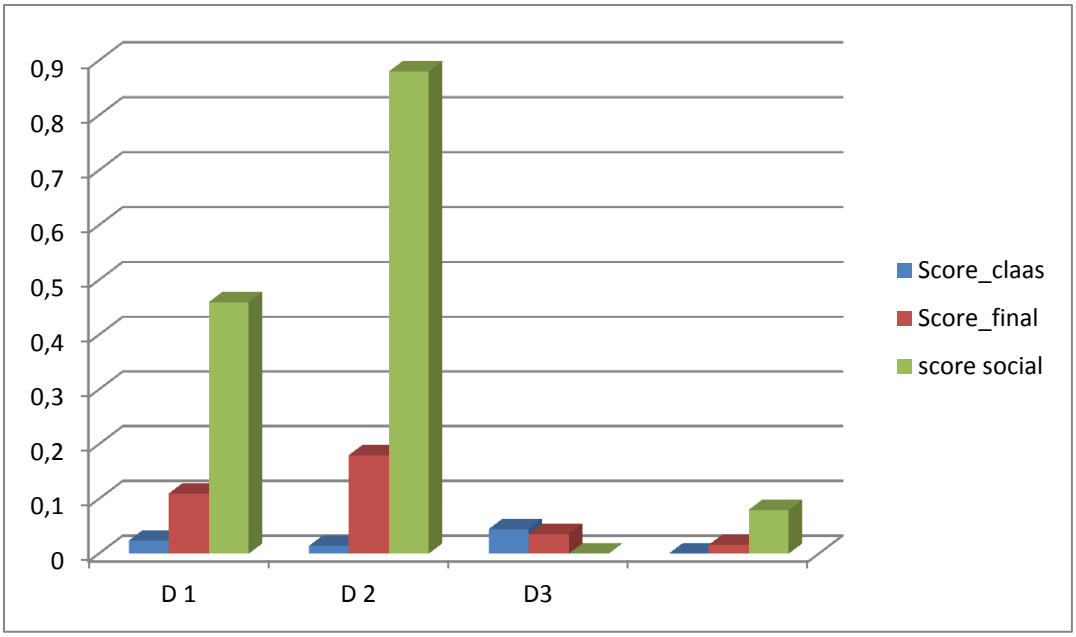


Figure 13 : diagramme des scores avec α=0.8

D'après les tests précédents plusieurs cas se présentent :

- Plus le score social est élevé plus le document remonte dans la liste des documents restitués (son score final est élevé) c'est le cas de document D1 et D2
  Où score classique(D1) > score classique(D2)
  Mais score sociale(D1) < score sociale(D1)
  Ce qui amène à score final(D2) > score final(D1)
  Cela signifie que le document D2 a des informations sociales qui correspondent à la requête plus récentes que celles de document D1.

- Les documents qui ne contient pas les termes de la requête ne sont pas restituer par le système avec une recherche traditionnelle mais comme ils ont été récemment tagués par une communauté en utilisant des mots clés qui sont dans la requête ils apparaissent dans le score final avec notre approche c'est le cas de document D4.

- Les documents qui contient les termes de la requête sont restitués par le système avec une recherche classique mais comme ils ont pas été récemment tagués par une communauté ils descendent dans la liste des documents restitués avec notre approche(leurs score final est diminué) c'est le cas de document D3.

Parmi les trois cas de α on a trouvé que le premier cas ou α=0.3 c'est le meilleur cas pour l'amélioration du score final.

C'est pour cela qu'on a pris α=0.3 pour réaliser notre approche.

III. Conception du système :

1. Architecture du système :

Dans notre système on trouve quatre parties essentielles, « le modèle document et tags», « le modèle requête », « le modèle recherche », « le modèle interface »

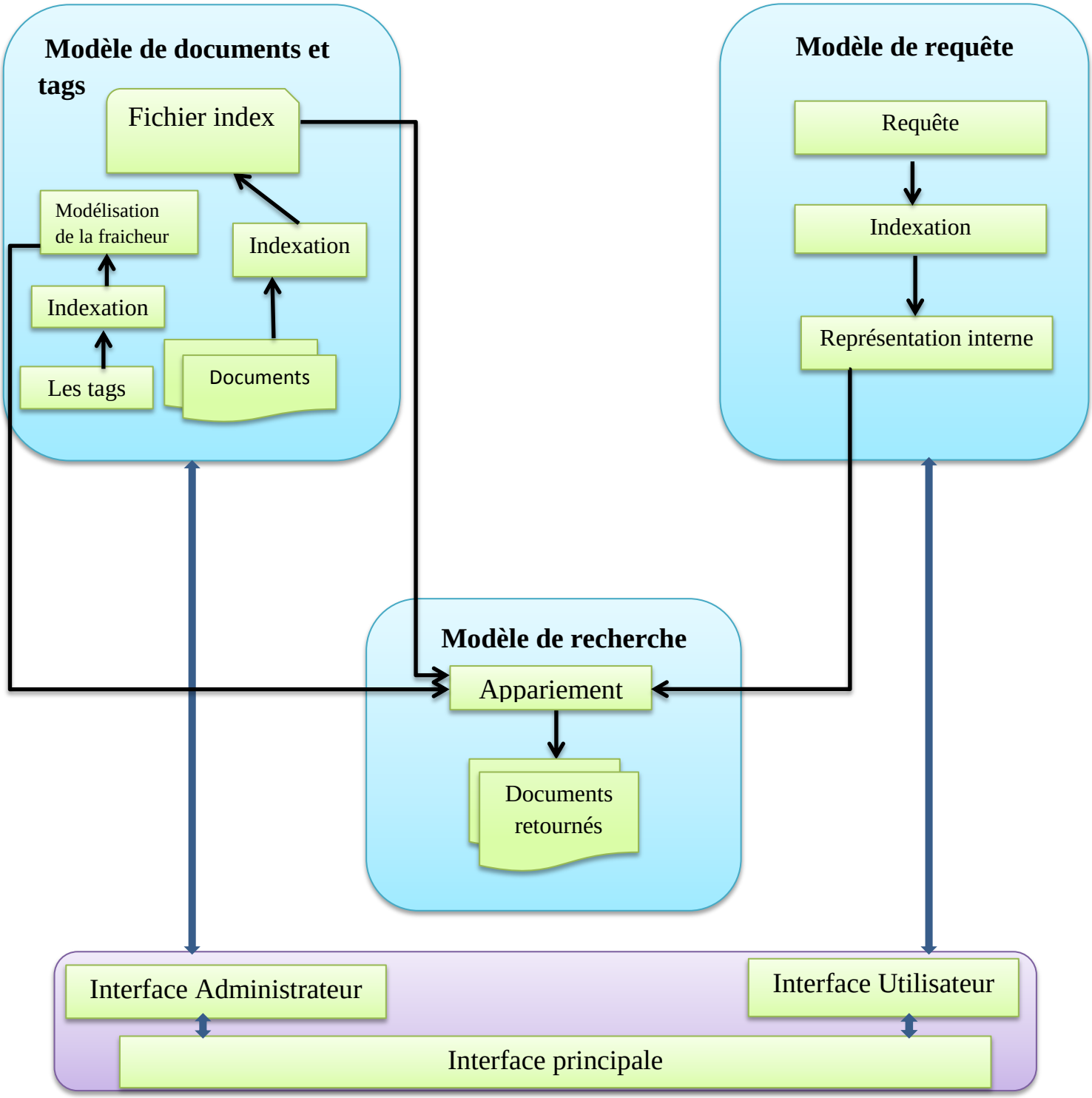


Figure 14 : Architecture globale du système

1.1. Description de la décomposition du système :

L'architecture générale du système que nous avons adopté est représentée selon trois niveaux comme l'illustre la figure suivante :

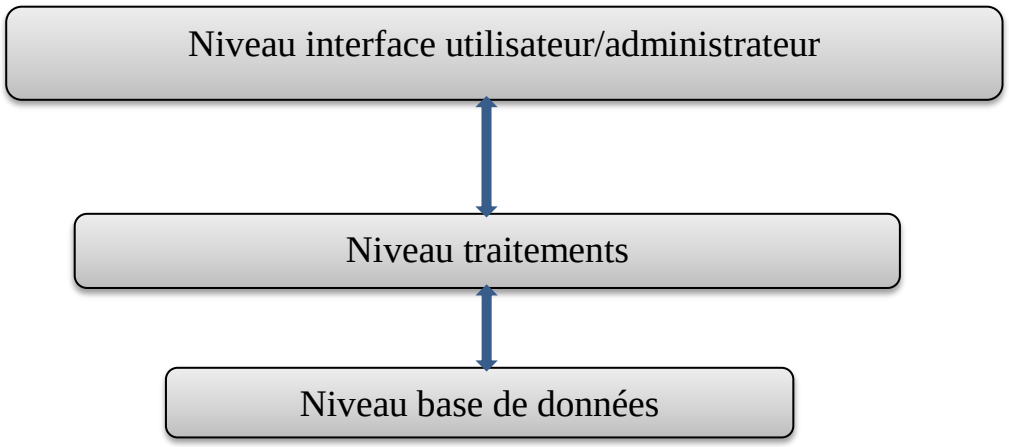


Figure 15 : les trois niveaux de système

Ces trois niveaux sont décrits comme suit :

1) Niveau interfaces utilisateur/administrateur :

Ce niveau regroupe les différentes interfaces de notre système à savoir :

- ✓ l'interface utilisateur appelée également interface de recherche dans laquelle l'utilisateur effectue des recherches d'information.
- ✓ l'interface administrateur dans laquelle l'administrateur pourra effectuer les opérations d'indexation des documents et des tags non encore indexés et gérer la base documentaire et base de tags.

2) Niveau traitement :

Ce niveau regroupe les trois modèles définis explicitement dans l'architecture globale du système, qui sont :

- Modèle de document et tags
- Modèle de requête
- Modèle de recherche.

Ces modèles coopèrent pour la réalisation des différentes tâches nécessaires pour le fonctionnement du système.

- **Modèle de document et tags:**

Il regroupe le traitement portant sur l'indexation des documents et l'indexation des tags et la modélisation de la fraicheur. Ce module a pour fonction principale la construction d'une représentation interne des textes intégraux. Cette fonction consiste à décrire les documents et tags par un ensemble de descripteurs susceptibles de représenter au mieux le contenu des documents et le contenu des tags. Cet ensemble est utilisé ultérieurement durant la recherche.

- **Modèle de requête :**

Ce modèle consiste en l'analyse de la requête afin de lui construire une représentation interne similaire à celles des documents. L'analyse de requête est préalable à toute recherche effective dans la base.

- **Modèle de recherche :**

Ce modèle permet d'effectuer l'appariement entre la requête interne et les représentations des documents de la base afin de restituer les informations pertinentes.

3) Niveau base de données :

Il concerne l'organisation physique des données relatives aux documents textuels.

Donc, c'est à ce niveau que s'effectue la manipulation des informations qui constituent le contenu des documents.

- **Architecture de la base de données :**

Le résultat d'analyse des documents sert de base pour construire les tables de la base de données. Les tables que nous avons implémentées dans notre base sont les suivantes :

- **Table Liste des documents :**

Cette table regroupe tous les documents de la base documentaire. Chaque enregistrement (ligne) de la table contient : le nom de document, la taille, l'adresse du document.

- **Table Liste des termes :**

Cette table regroupe tous les termes d'indexation de la base documentaire. Chaque enregistrement contient: la forme tronqué du terme le nombre de documents ou le terme apparait.

- **Table fichier direct :**

Cette table contienne tous les termes associer aux documents et leurs fréquences.

- **Table fichier inverse :**

Cette table contient tous les documents dans lequel le terme apparait avec leur fréquence.

- **Table freq_doc :** contienne le nombre de documents ou le terme apparait.

- **Table liste des tags *:***

Cette table regroupe tous les tags de la base tags. Chaque enregistrement (ligne) de la table contient : la date de création, nom de tag, mot-clé, adr_tag.

- **Table tag direct :**

Cette table contient tous les termes associer aux fichiers tags et leurs dates de création

- **Table tag inverse :**

Cette table contient tous les fichiers tags dans lequel le terme apparait avec leur date de création.

En plus de ces tables, d'autres tables sont prévues pour la base de données, on citera :

- **Table mot de passe**: pour assurer le stockage de la valeur du mot de passe.

2. La conception d'objet :

Le résultat de cette étape nous présentons les algorithmes de quelques opérations telles que les algorithmes d'indexation des documents et de la requête et des tags ainsi que l'algorithme de recherche classique, l'algorithme de recherche sociale et l'algorithme de recherche finale

1. Algorithme d'indexation de document:

Entrée : collection des documents

Sortie : Table fichier inverse, Table fichier directe, Table terme et Table document

Début

La table des documents tab_Doc vide (Dopc_id, Taille_doc,Doc_addrs) ;

La table des termes de documents Tab_T vide (Terme_id, TF) ;

Répéter

Extraire un document D de la collection ;

N=0 ; // compteur de nombre de termes dans le document D ;

Répéter

Extraire un mot t_i dans le document D;

Le transformer en minuscule ;

Le simplifier ;

Si ($t_i \in$ stoplist) **alors**

Eliminer ce mot vide ;

Sinon

Si ti contient plus de 7 caractères **alors**

Le tronquer ;

Fin si

Si $t_i \notin$ TAB_T **alors**

Début

TF (t_i)=0;//initialiser la fréquence du terme t_i

TF (t_i)= TF(t_i)+1 ;

(1) (2)(3) (4) (5) (6) TAB_T.insérer(t_i , TF(t_i));

(1) (2)(3) (4) (5) (6) N= N+1; //incrémenter le nombre de termes.

Fin

Sinon

TAB_T.insérer [t_i] [TF (t_i)] +1 ;

Fin sinon ;

Fin si ;

Fin si ;

Jusqu'à la fin du document ;

Tab_Doc.insérer (Doc_id, Doc_Taille, Doc_Addres) ;

Jusqu'à la fin de la collection ;

// Remplir la table de la fréquence des termes de la collection Tab_DF//

Pour chaque Doc_id ∈ Tab_Doc **faire**

Pour chaque terme t_i de TAB_T **faire**

Si ($t_i \notin$ Tab_DF) **alors**

DF=0 ;

DF=DF+1 ;

Tab_DF.insérer (t_i, DF) ;

Sinon

Tab_DF [t_i][DF]+1 ;

Fin si ;

Fait ;

Fait ;

(1) // cette boucle parcoure toute la collection pour remplir les tables fichier direct et inverse//

(1) Pour chaque D∈Tab_Doc **faire**

Récupérer le Doc_id ;

Récupérer la TAB_T ;

// Remplissage de la table des fichiers directe//

Si Doc_id ∉ Tab_fic_dir **faire**

Tab_fic_dir.inserer(Doc_id,TAB_T) **;**

Fin si

// Remplissage de la table de fichier inverse//

Pour chaque terme t_i de TAB_T **faire**

Si t_i ∈fichier inverse **alors**

Récupérer TF (t_i)

Si Doc_id ∉index_inverse(t_i) **alors**

Tab_fic_invers.inserer [term_id] (Doc_id, TF (t_i));

Fin si;

Sinon

Si t_i ∉ fichier_inverse **alors**

Tab_fic_invers.inserer(term_id (Doc_id, TF (t_i));

Fin si;

Fait;

Fait;

FIN.

2. Algorithme d'indexation de tag:

Entrée : collection des tags

Sortie : Table tag inverse, Table tag directe et Table tag

Début

La table des tags tab_tag vide (tag_id, mot_cle,date_crea,tag_adrs) ;

Répéter

Extraire un tag T de la collection ;

Pour chaque ligne du tag T **faire**

Répéter

Extraire un mot t_i dans la ligne;

Le transformer en minuscule ;

Le simplifier ;

Si ($t_i \in$ stoplist) **alors**

Eliminer ce mot vide ;

Sinon

Si ti contient plus de 7 caractères **alors**

Le tronquer ;

Fin si

Dat_cr (t_i)= premier élément de la ligne ;

TAB_T.insérer(t_i ,Dat_cr);

Fin si

Jusqu'à la fin du Tag ;

Tab_tag.insérer (tag_id, mot_cle,dat_cr, tag_Adres) ;

Jusqu'à la fin de la collection ;

// Cette boucle parcoure toute la collection pour remplir les tables tag direct et inverse//

(1) Pour chaque T∈Tab_tag **faire**

Récupérer le tag_id ;

Récupérer la TAB_T ;

// Remplissage de la table des tags directe//

Si tag_id ∉ Tab_tag_dir **faire**

Tab_tag_dir.inserer(tag_id,TAB_T) **;**

Fin si

// Remplissage de la table de tag inverse//

Pour chaque terme t_i de TAB_T **faire**

Récupérer dat_cr (t_i)

Si tag_id ∉tag_inverse(t_i) **alors**

Tab_tag_invers.inserer (tag_id,t_i ,dat_cr (t_i));

Fin si ;

Fait;

Fait ;

FIN.

3. Algorithme d'indexation de la requête :

Entrer : requête Q
Sortie : liste des termes de la requête
Liste des fréquences des termes de la requête

Début
Initialiser le vecteur des termes vect_terme_req à vide ;
Initialiser le vecteur des fréquences des termes vect_freq _req à vide ;
Tan que non fin de la requête **Faire**
Début
Extraire un terme de la requête ;
Transformer les majuscules en minuscules
Si terme Stoplist **alors**
Eliminer le terme
Sinon
Troncature du terme
Si terme vect_terme_req **alors**
Début
Insérer le terme dans vect_terme_req ;
nbr_occurence =1 ;
Insérer nbr_occurnce dans vect_freq_req
Fin
Sinon
nbr_occurence = nbr_occurence +1 ;
Mettre à jour le vet_freq_req
Fsi
Fsi
Fin
Fait
Pour chaque terme de vect_term_req **Faire**
Calculer tf(terme) ;
Remplacer nbr_occurence du terme par tf dans le vect_freq_req ;
Fait
Fin

4. Algorithme de Recherche :

1. Algorithme Recherche classique ;

Entrée : Requête saisi par l'utilisateur Q ;
Sortie : Liste des documents pertinents ;
Début
-lire la requête Q;
Procédure d'indexation de la requête
Début
Initialiser à vide le vecteur vect_term des termes de la requête
Répéter
-Extraire un terme (q_i) de la requête ;
Si q_i en majuscule **alors**
-Transformer en minuscule ;
Fin si
Si $q_i \in$ stopliste **alors**
-Eliminer ce mot vide ;
Sinon
Si q_i contient plus de 7 caractères **alors**
-Tronquer ce mot ;
Fin si
Si $q_i \notin$ vect_term_req **alors**
vect_term_req.inseree (q_i);
Fin si;
Fin sinon;
Fin si;
Jusque à la fin de la requête ;
Fin procédure d'indexation de la requête;
Procédure de Matching
Début
-Initialiser à vide le vecteur des documents vect_doc_id ;
Pour chaque terme $q_i \in$ vect_term_req **faire**
-appeler Tab_fic_inv (q_i) ;
Si Tab_fic_inv contient q_i **alors**
Tant que Tab_fic_inv (q_i) pas vide **faire**
- Récupérer Doc_id ;
- Récupérer TF;
Vect_doc_id.inserer(Doc_id) ;
Vect_doc_tf.inserer(TF) ;
Fait ;

Pour chaque $D_j \in$ Vect_doc_id **faire**
Pour chaque TF $\in$ Vect_doc_tf **faire**
- Récupérer la taille Taille=Doc_taille (q_i) ;
-Récupérer le df df=Df (q_i);

$$Score(D_j) = \sum \left\lceil \log \frac{N}{Df} \right\rceil . \frac{(k1+1)TFj}{k1\,(1-b) + b.\,(Taille/TF_j)} . \frac{(k3+1)\,tfq}{(k3+tfq)}$$

Avec k1=1.2; k3=8; tfq=1; b=0.75
Pour chaque Score(Dj) de vect_score **faire**
Si vect_score(Dj) vide alors
Vect_score.inserer (Dj) ;
Fin si ;
Sinon
Vect_score [(Dj)] +Score(Dj) ;
Fin sinon ;
Fait ;
Fait ;
Fait ;
Fin si ;
Fait ;
Fin procédure Matching ;
Ordonner le vecteur vect_Score
Afficher les documents pertinents ;
Fin ;

2. Algorithme Recherche sociale ;

Entrée : Requête saisi par l'utilisateur Q ;
Sortie : Liste des tags pertinents ;
Début
-lire la requête Q;
Procédure d'indexation de la requête
Début
Initialiser à vide le vecteur vect_term des termes de la requête
Répéter
-Extraire un terme (q_i) de la requête ;
Si q_i en majuscule **alors**
-Transformer en minuscule ;
Fin si
Si $q_i \in$ stopliste **alors**
-Eliminer ce mot vide ;
Sinon

Si q_i contient plus de 7 caractères **alors**
-Tronquer ce mot ;
Fin si
Si $q_i \notin$ vect_term_req **alors**
vect_term_req.inserer (q_i);
Fin si;
Fin sinon;
Fin si;
Jusque à la fin de la requête ;
Fin procédure d'indexation de la requête;
Procédure de Matching
Début
-Initialiser à vide le vecteur des tags vect_tag_id ;
Pour chaque terme $q_i \in$ vect_term_req **faire**
-appeler Tab_tag_inv (q_i) ;
Si Tab_tag_inv contient q_i **alors**
Tant que Tab_tag_inv (q_i) pas vide **faire**
- Récupérer Tag_id ;
- Récupérer dat_cr;
Vect_tag_id.inserer(Tag_id) ;
Vect_tag_dat.inserer(dat_cr) ;
Fait ;
Pour chaque $T_j \in$ Vect_tag_id **faire**
- Récupérer adr_tag
- Vect_tag_adr.inserer(adr_tag) ;
-recuperer la date actuelle date_act=la date de la mise ajour
Score (T_j)=1/date_at-date_crea;
Pour chaque Score(Tj) de vect_score **faire**
Si vect_score(Tj) vide alors
Vect_score.inserer (Tj) ;
Fin si ;
Sinon
Vect_score[(Tj)]+Score(Tj) ;
Fin sinon ;
Fait ;
Fait
Fin si ;
Fait ;
Fin procédure Matching ;
Ordonner le vecteur vect_Score
Afficher les tags pertinents ;

Fin ;

1. Algorithme Recherche finale ;

Entrée : Requête saisi par l'utilisateur Q ;
Sortie : Liste des documents pertinents ;
Début
-lire la requête Q;
Procédure d'indexation de la requête
Début
Initialiser à vide le vecteur vect_term des termes de la requête
Répéter
-Extraire un terme (q_i) de la requête ;
Si q_i en majuscule **alors**
-Transformer en minuscule ;
Fin si
Si $q_i \in$ stopliste **alors**
-Eliminer ce mot vide ;
Sinon
Si q_i contient plus de 7 caractères **alors**
-Tronquer ce mot ;
Fin si
Si $q_i \notin$ vect_term_req **alors**
vect_term_req.inseree (q_i);
Fin si;
Fin sinon;
Fin si;
Jusque à la fin de la requête ;
Fin procédure d'indexation de la requête;
Procédure de Matching
Début
-Initialiser à vide le vecteur des documents vect_doc_id ;
Pour chaque terme $q_i \in$ vect_term_req **faire**
-appeler Tab_doc_inv (q_i) ;
Si Tab_doc_inv contient q_i **alors**
Tant que Tab_doc_inv (q_i) pas vide **faire**
- Récupérer doc_id ;
- Récupérer TF;
Vect_doc_id.inserer(doc_id) ;
Vect_doc_tf.inserer(TF) ;
Fait ;
Pour chaque terme $q_i \in$ vect_term_req **faire**
-appeler Tab_tag_inv (q_i) ;

Si Tab_tag_inv contient q_i **alors**
Tant que Tab_tag_inv (q_i) pas vide **faire**
- Récupérer Tag_id ;
-Recuperer la date_cre
Vect_tag_id.inserer(Tag_id) ;
Vect_tag_date.inserer(date_cre);

Fait ;
Fin si ;
Pour chaque $T_j \in$ Vect_tag_id **faire**
- Récupérer la date date_crea=Tag_date (q_i) ;
-recuperer la date actuelle date_act=la date de la mise ajour
Score (T_j)=1/date_at-date_crea;
Pour chaque $D_j \in$ Vect_doc_id **faire**
Pour chaque TF $\in$ Vect_doc_tf **faire**
- Récupérer la taille Taille=Doc_taille (q_i) ;
-Récupérer le df df=Df (q_i);

$$Score(D_j) = \sum \left[\frac{\log N}{Df}\right] \cdot \frac{(k1+1)TFj}{k1\,(1-b) + b.\,(Taille/TF_j)} \cdot \frac{(k3+1)\,tfq}{(k3+tfq)}$$

Avec k1=1.2; k3=8; tfq=1; b=0.7
Score=0.3*Score(D_j)+(1-0.3)*Score(Tj)
Pour chaque Score de vect_score **faire**
Si vect_score(Dj) vide alors
Vect_score.inserer (Dj) ;
Fin si ;
Sinon
Vect_score[(Tj)]+Score(Tj) ;
Fin sinon ;
Fait ;
Fait ;
Fait ;
Fait ;
Fin si ;
Fait ;
Fin procédure Matching ;
Ordonner le vecteur vect_Score
Afficher les documents pertinents ;
Fin ;

IV. Conclusion :

Dans ce chapitre, nous avons décrit notre solution proposée pour réaliser notre approche et la conception de notre SRIS selon la démarche suivante :

Durant l'étape de description de notre solution, nous avons pu décrire les deux modules (score sociale, score classique) qu'on a utilisé.

Durant l'étape de conception, nous avons présenté l'architecture globale de notre système, la décomposition du système en sous-systèmes, ainsi, nous avons présenté quelques méthodes de classes (Rechercher, Indexer).

Dans le chapitre suivant, nous allons présenter l'implémentation de notre système avec la solution proposé.

CHAPITRE

4

Réalisation

I. Introduction :

L'objectif de notre travail est de réaliser sur machine un SRIS qui répond aux besoins des utilisateurs. Après avoir présenté l'analyse et la conception de notre système, nous allons dans ce chapitre décrire son implémentation dans un environnement orienté objet, nous commençons par présenter l'environnement de développement, puis nous présenterons les interfaces de notre système.

II. L'environnement de travail :

L'application que nous avons réalisée a été développée sur un micro-ordinateur
Ayant les principales caractéristiques suivantes :
– Fréquence d'horloge : 2.10 GHz;
– RAM de capacité : 3 GO ;
– Disque dur de capacité : 178 GO.
Nous avons travaillé sous le système d'exploitation Windows 7 et pour la programmation, on a opté pour le langage java sous l'environnement de NETBEANS.

III. Les outils de développement utilisés :

1. Le langage de programmation Java :

Afin de réaliser l'interface permettant aux utilisateurs de manipuler notre système, nous avons utilisé le langage java développé par les laboratoires *Sun Microsystems.*
Java est un langage de programmation apparu en 1991, dont la syntaxe est très proche de celle de C++. Java est aussi une synthèse de plusieurs langages de programmation.
Le choix de ce langage est pour les raisons suivantes :

- Java est un langage multi-plate-forme, qui permet aux concepteurs d'écrire un code capable de fonctionner dans tous les environnements. Pour cela, il suffit que l'environnement possède une JVM (Java Virtual Machine).
- Java est un langage orienté objet simple, qui réduit le risque d'incohérence.
- Java est doté d'une riche bibliothèque de classes, comprenant la gestion des interfaces graphiques (fenêtres, boites de dialogues, contrôles, menus, graphismes), et la gestion des exceptions.
- Le JDK (Java Developpement Kit) fournit gratuitement par *Sun*, regroupe l'ensemble des éléments permettant le développement, la mise au point et l'exécution des programmes Java.

2. Accès aux bases de données :

JDBC est l'acronyme de Java Data Base Connectivity et désigne une API définie par Sun pour permettre un accès aux bases de données Java.

2.1. Les outils nécessaires pour utiliser JDBC :

Pour pouvoir utiliser JDBC, il faut un pilote qui est spécifique à la base à laquelle on veut accéder. Avec le JDK, Sun fournit un pilote qui permet l'accès aux bases de données via ODBC (Open Data Base Connectivity).
Les classes de JDBC sont regroupées dans le package java.sql et sont incluses dans le JDK à partir de sa version 1.1.
Il y'a 4 classe importantes, chacune correspond à une étape de l'accès aux données :

- ***DriverManager*** :

Charge et configure le driver de la base de données.

- ***Connections :***

Réalise la connexion et l'authentification à la base de données.

- ***Statement (et Prepared Statement)*** :

Contient la requête SQL et la transmet à la base de données.

- ***ResultSet :***

Permet de parcourir les informations retournées par la base de données dans le cas d'une sélection de données.

2.2. La connexion à une base de données :

- **Le chargement du pilote :**

Pour se connecter à une base de données via ODBC, il faut tout d'abord charger le pilote JDBC-ODBC qui fait lien entre les deux.

- **L'établissement de la connexion :**

Pour se connecter à une base de données, il faut instancier un objet de la classe Connexion en lui précisant sous forme d'URL la base à laquelle on accède.

3. Présentation de l'environnement de Netbeans :

Netbeans est un projet open source fondé par Sun Microsystems. L'IDE Netbeans est un environnement de développement permettant d'écrire, compiler, déboguer et déployer des programmes. Il est écrit en Java – mais peut supporter n'importe quel langage de programmation. Il y a également un grand nombre de modules pour étendre l'IDE Netbeans.

L'IDE Netbeans est un produit gratuit, sans aucune restriction quant à son usage. L'installation de l'IDE Netbeans nécessite l'installation de la JDK (Java Developpement Kit) le kit de développement java compatible avec la version d'IDE. Pour concevoir notre système nous avons utilisé la version Netbeans 7.0.1 et son interface principale est donnée dans la figure suivante Figure.

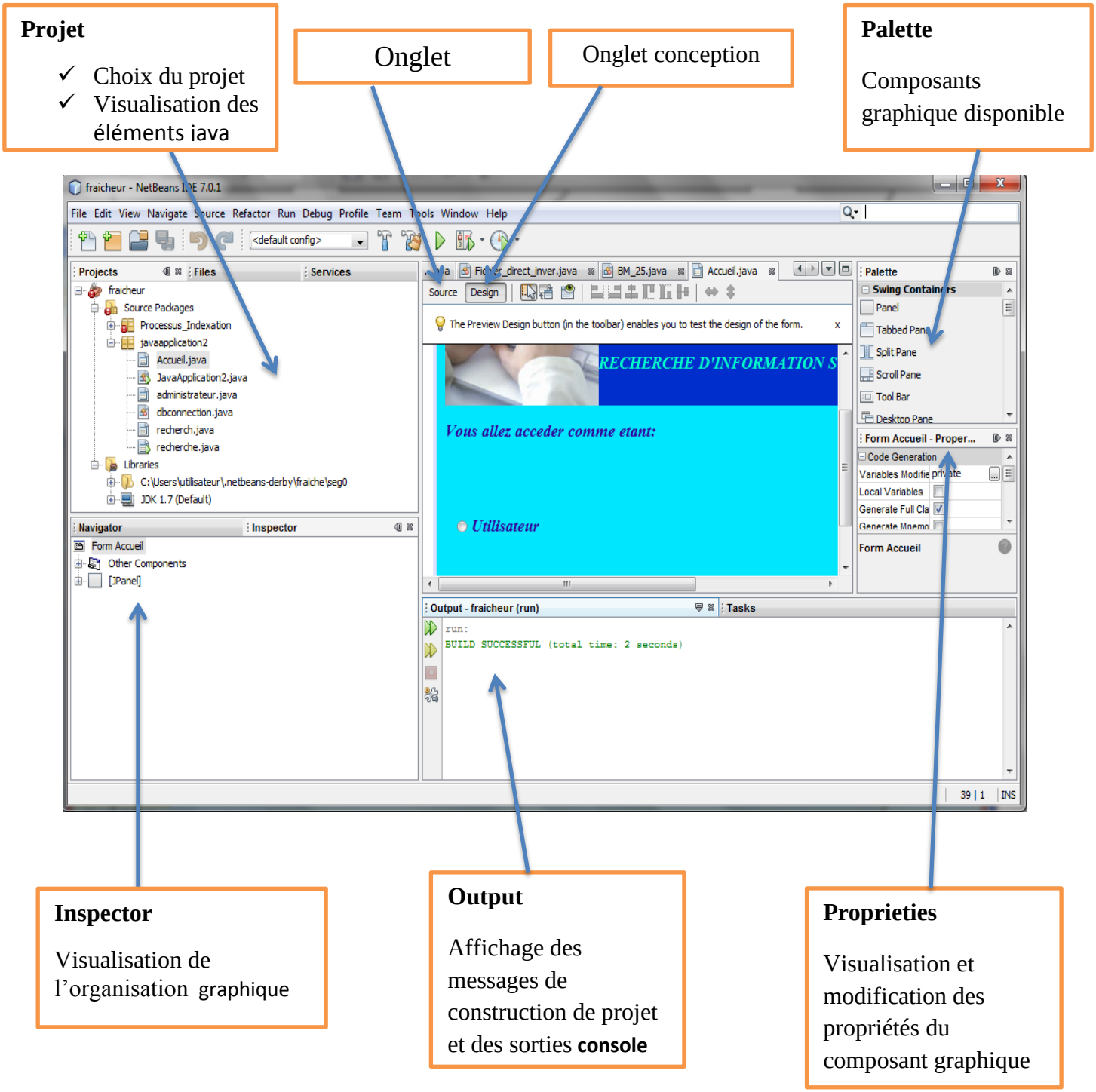


Figure 16 : l'interface de Netbeans IDE

4. Présentation des interfaces de notre SRIS :

Dans notre système, la première interface qui s'affiche est la page d'accueil, qui figure comme suit :

- **Interface d'accueil :**

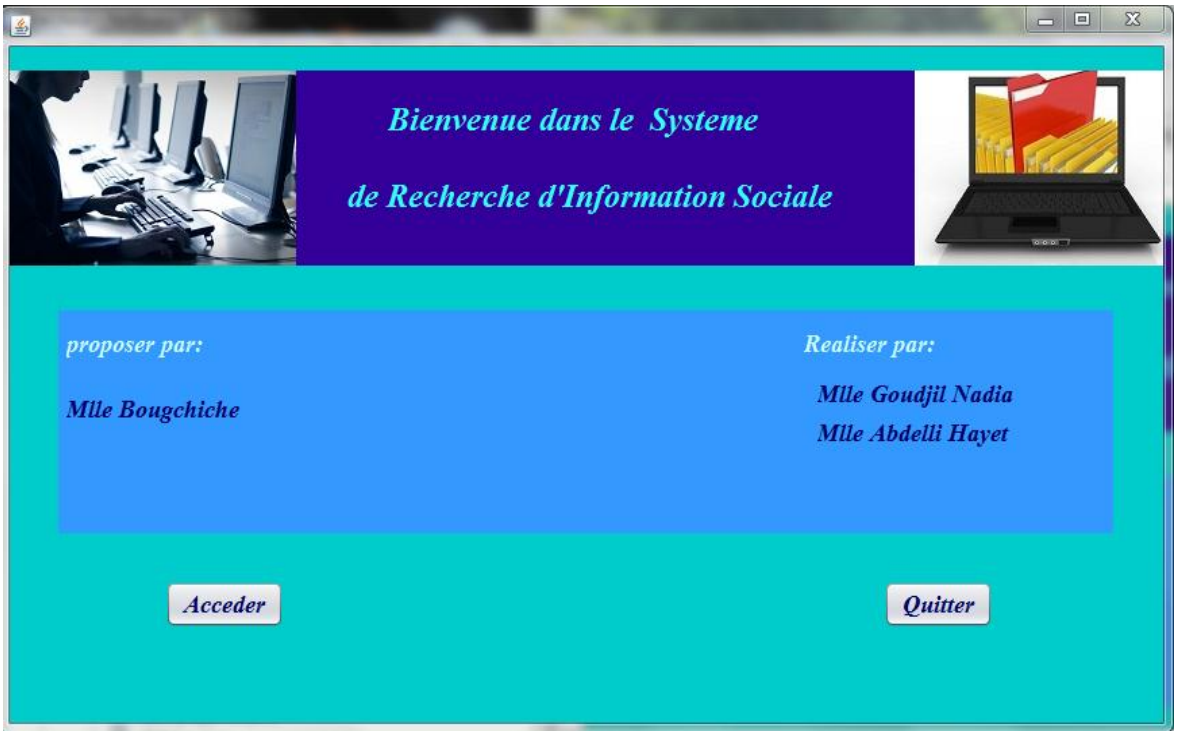


Figure 17: interface d'accueil.

Via cette interface, les acteurs de notre application peuvent accéder à notre système de recherche d'information sociale. Une fois le choix d'accès au système est effectué, une deuxième fenêtre apparait sur écran, dans le but de faire choisir à l'acteur son mode d'accès (administrateur ou utilisateur). Cette fenêtre sera affichée comme suit :

Figure 18: Interface principale.

- **Mode utilisateur :**

Figure19 : Interface Utilisateur.

Cette interface permet à l'utilisateur d'effectuer des recherches dans la base documentaire ou base de tags. Pour ce fait, l'utilisateur doit saisir la requête dans la zone et lance la recherche en cliquant sur l'un des boutons ***Recherche_Sociale, Recherche_Classique, Recherche_Finale***. S'il trouve des documents satisfaisant la requête, le système lui retourne, par ordre de pertinence décroissant, les liens vers ces documents.

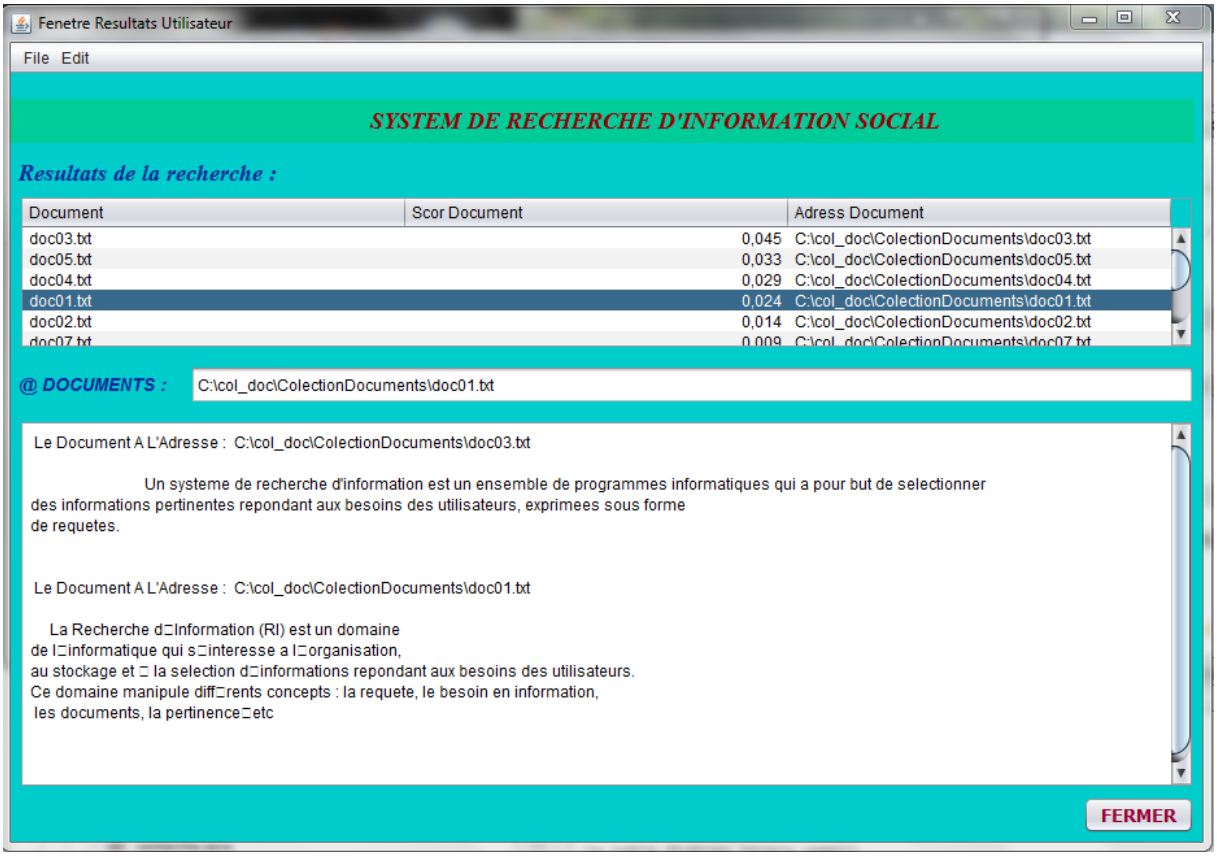


Figure20 : Résultats de la recherche classique.

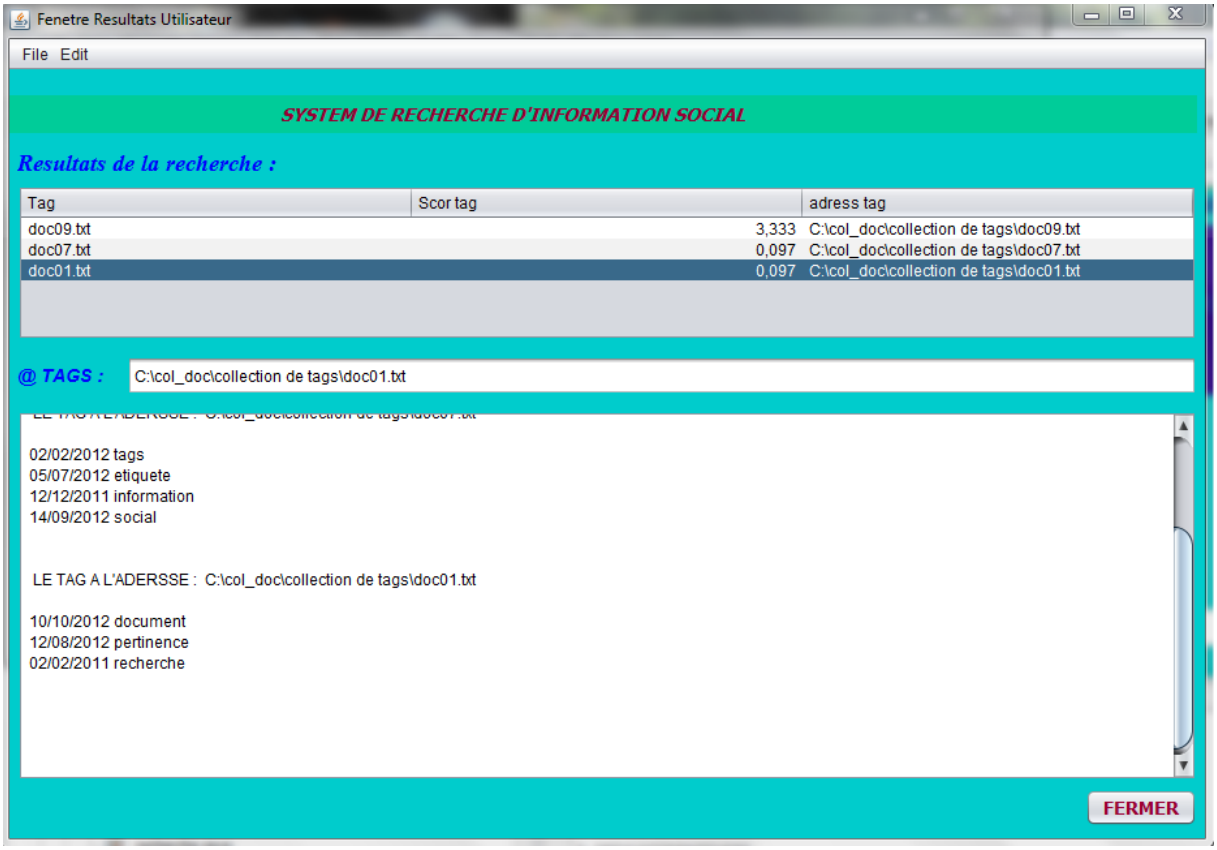


Figure21 : Résultats de la recherche sociale

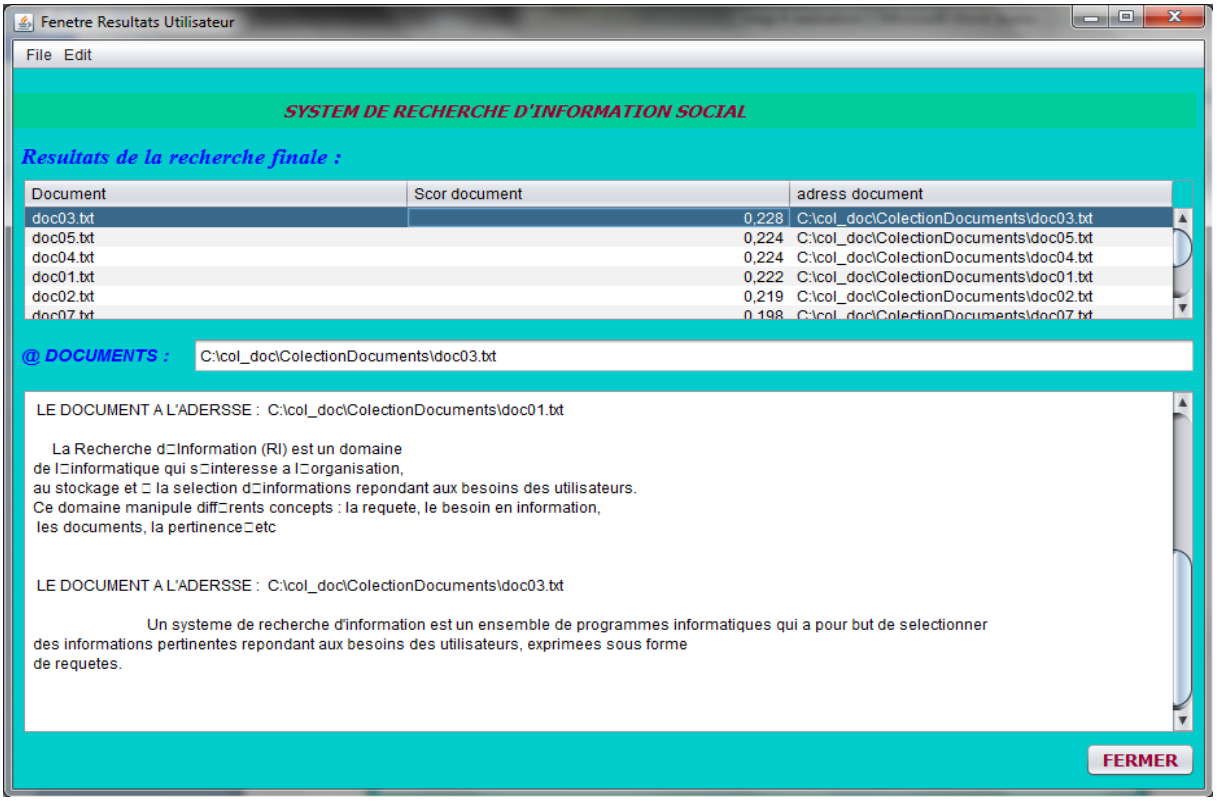


Figure22 : Résultats de la recherche finale.

Dans le cas de la recherche finale le classement des documents peut être changé selon leurs tags :

1. Les documents qui n'ont pas un score élevé peuvent se descendre car leurs tags ne sont pas récents.
2. La liste reste inchangée si aucun terme de la requête ne figure dans la collection de tags.

- **Mode administrateur :**

Si le choix se porte sur l'Administrateur, une fenêtre d'authentification sera affichée comme la montre la figure suivante :

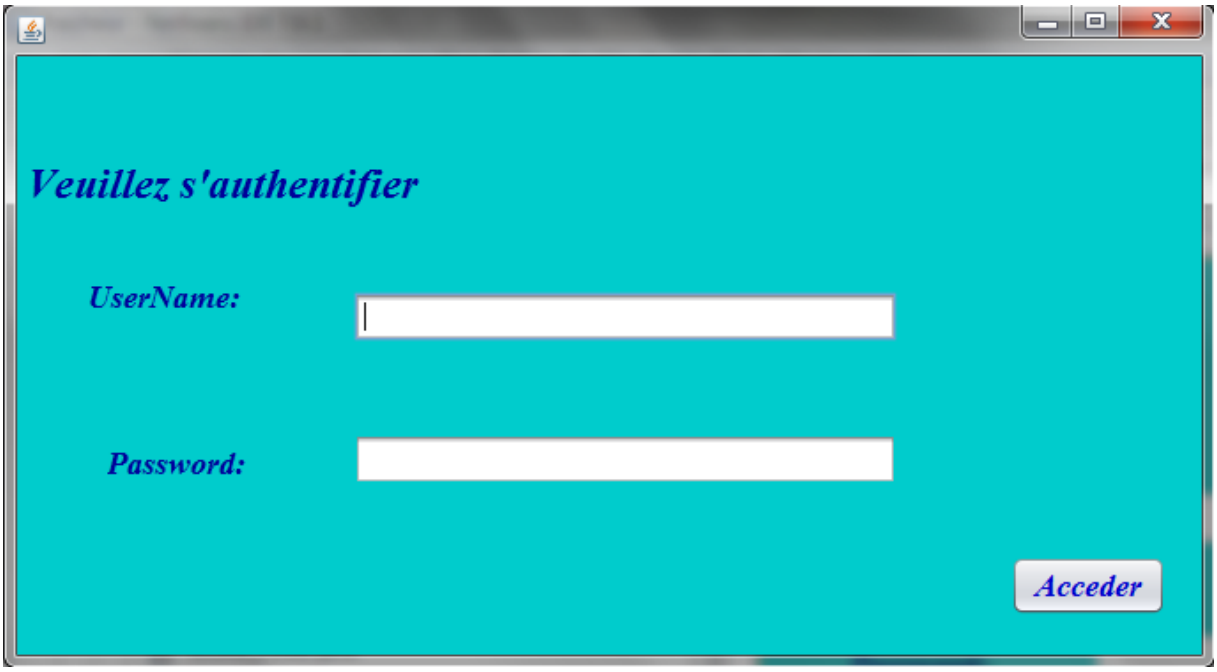


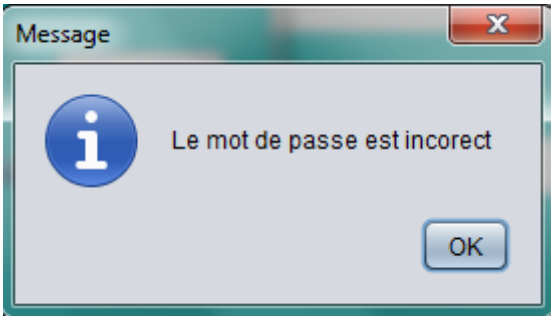
Figure23 : Interface authentification.

Pour des raisons de sécurité des données de la base, et avant de pouvoir accéder à l'interface administrateur, l'administrateur doit saisir son nom et son mot de passe.

- ✓ Le bouton « Annuler » permet de fermer la fenêtre d'authentification

Deux cas seront possibles:

Cas1 : Si le mot de passe introduit est incorrect, alors un message d'erreur sera affiché comme suit :



Cas 2 : Si le mot de passe saisi est correct, l'administrateur accède à son espace.

Figure 24: Interface Administrateur.

- ✓ Dans le cas « **Indexation_Document** » une fenêtre s'affiche pour s'sélectionner la collection de documents a indexé. Comme suit.

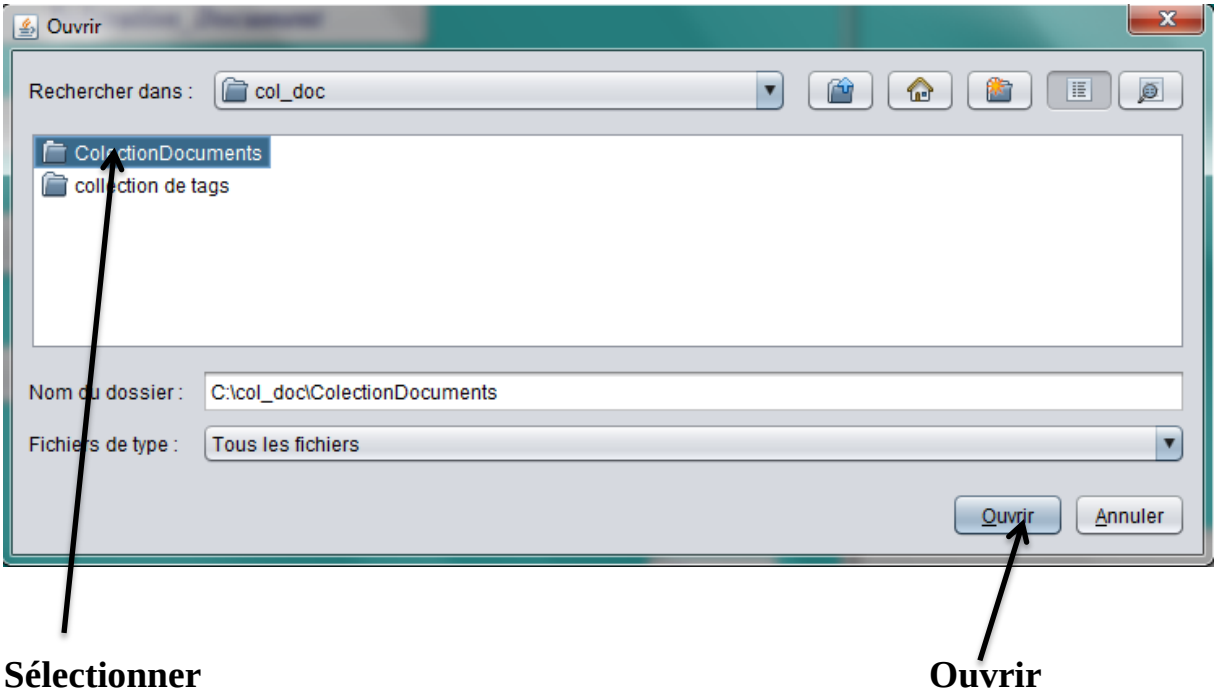


Figure25 : Interface de choix de la collection de documents à indexé.

En cliquant sur le bouton ouvrir, la base de données est remplie.

- ✓ Dans le cas « **Indexation_Tag** » une fenêtre s'affiche pour s'sélectionner la collection de tags a indexé. Comme suit.

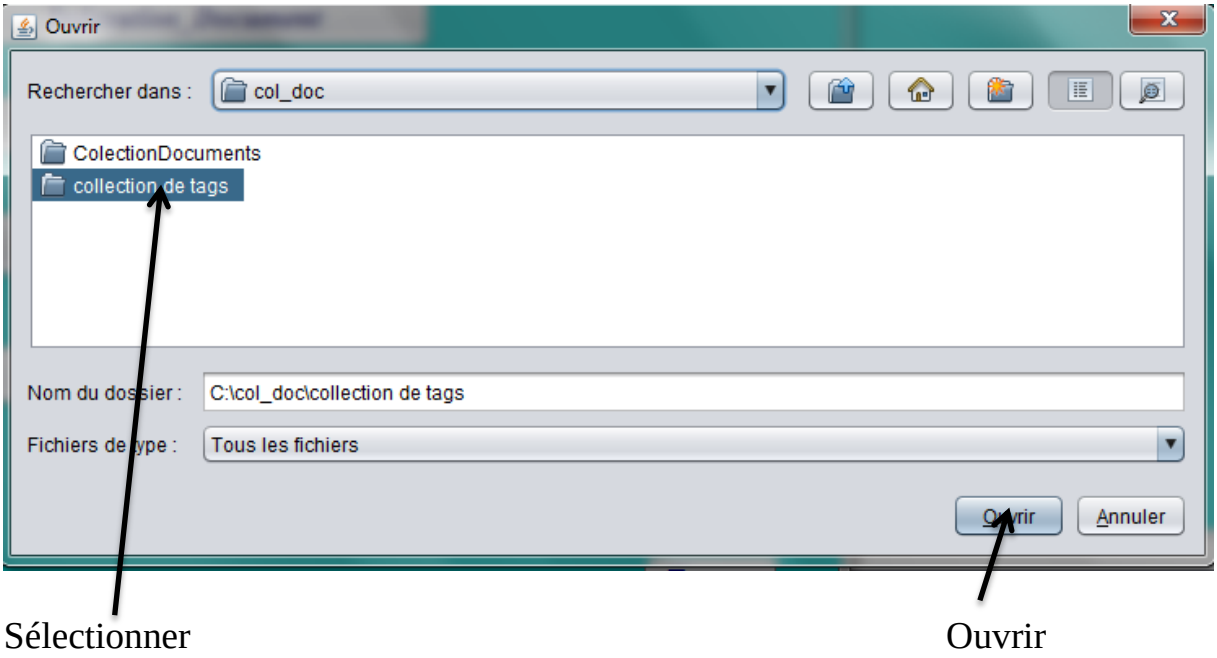
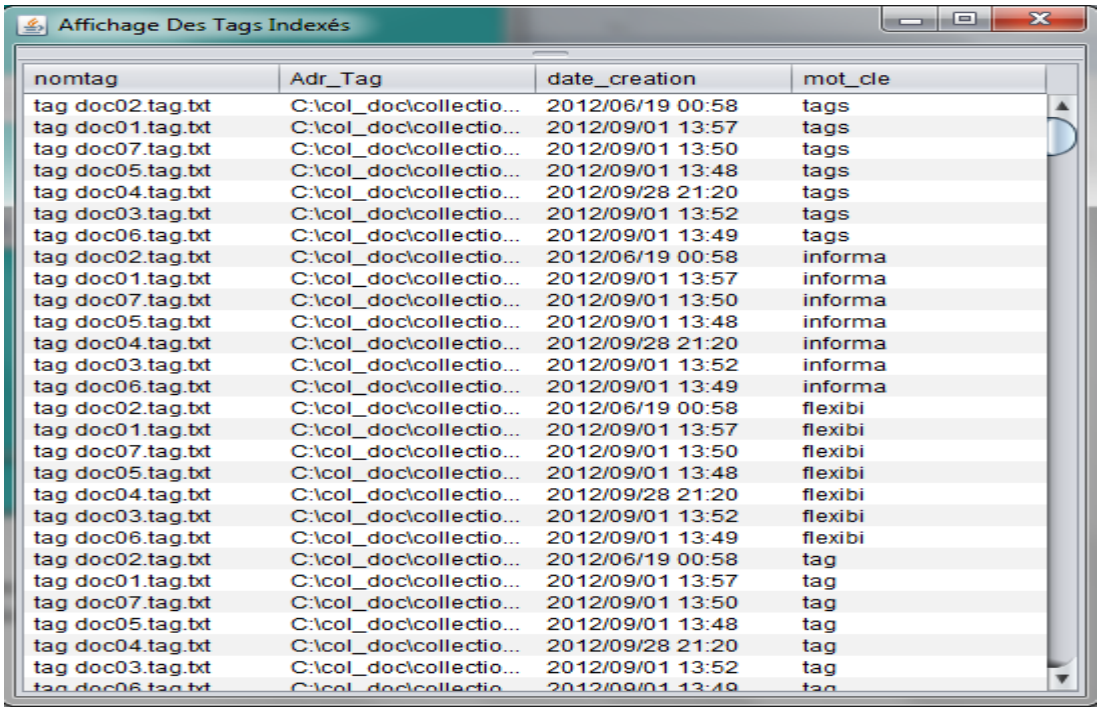


Figure26 : Interface de choix de la collection de tags à indexé.

- ✓ Dans le cas de choix de la « **consultation** »une fenetre de resultats de l'indexation s'affiche. Comme suit.

Figure27 : Interface de consultation de tables.

✓ le bouton « Tags Indexés »permet d'afficher une liste des tags de la base de données. Comme suit.



nomtag	Adr_Tag	date_creation	mot_cle
tag doc02.tag.txt	C:\col_doc\collectio...	2012/06/19 00:58	tags
tag doc01.tag.txt	C:\col_doc\collectio...	2012/09/01 13:57	tags
tag doc07.tag.txt	C:\col_doc\collectio...	2012/09/01 13:50	tags
tag doc05.tag.txt	C:\col_doc\collectio...	2012/09/01 13:48	tags
tag doc04.tag.txt	C:\col_doc\collectio...	2012/09/28 21:20	tags
tag doc03.tag.txt	C:\col_doc\collectio...	2012/09/01 13:52	tags
tag doc06.tag.txt	C:\col_doc\collectio...	2012/09/01 13:49	tags
tag doc02.tag.txt	C:\col_doc\collectio...	2012/06/19 00:58	informa
tag doc01.tag.txt	C:\col_doc\collectio...	2012/09/01 13:57	informa
tag doc07.tag.txt	C:\col_doc\collectio...	2012/09/01 13:50	informa
tag doc05.tag.txt	C:\col_doc\collectio...	2012/09/01 13:48	informa
tag doc04.tag.txt	C:\col_doc\collectio...	2012/09/28 21:20	informa
tag doc03.tag.txt	C:\col_doc\collectio...	2012/09/01 13:52	informa
tag doc06.tag.txt	C:\col_doc\collectio...	2012/09/01 13:49	informa
tag doc02.tag.txt	C:\col_doc\collectio...	2012/06/19 00:58	flexibi
tag doc01.tag.txt	C:\col_doc\collectio...	2012/09/01 13:57	flexibi
tag doc07.tag.txt	C:\col_doc\collectio...	2012/09/01 13:50	flexibi
tag doc05.tag.txt	C:\col_doc\collectio...	2012/09/01 13:48	flexibi
tag doc04.tag.txt	C:\col_doc\collectio...	2012/09/28 21:20	flexibi
tag doc03.tag.txt	C:\col_doc\collectio...	2012/09/01 13:52	flexibi
tag doc06.tag.txt	C:\col_doc\collectio...	2012/09/01 13:49	flexibi
tag doc02.tag.txt	C:\col_doc\collectio...	2012/06/19 00:58	tag
tag doc01.tag.txt	C:\col_doc\collectio...	2012/09/01 13:57	tag
tag doc07.tag.txt	C:\col_doc\collectio...	2012/09/01 13:50	tag
tag doc05.tag.txt	C:\col_doc\collectio...	2012/09/01 13:48	tag
tag doc04.tag.txt	C:\col_doc\collectio...	2012/09/28 21:20	tag
tag doc03.tag.txt	C:\col_doc\collectio...	2012/09/01 13:52	tag
tag doc06.tag.txt	C:\col_doc\collectio...	2012/09/01 13:49	tag

Figure28 : table des tags indexés.

- ✓ le bouton « Documents Indexés » permet d'afficher une liste des Documents de la base de données.

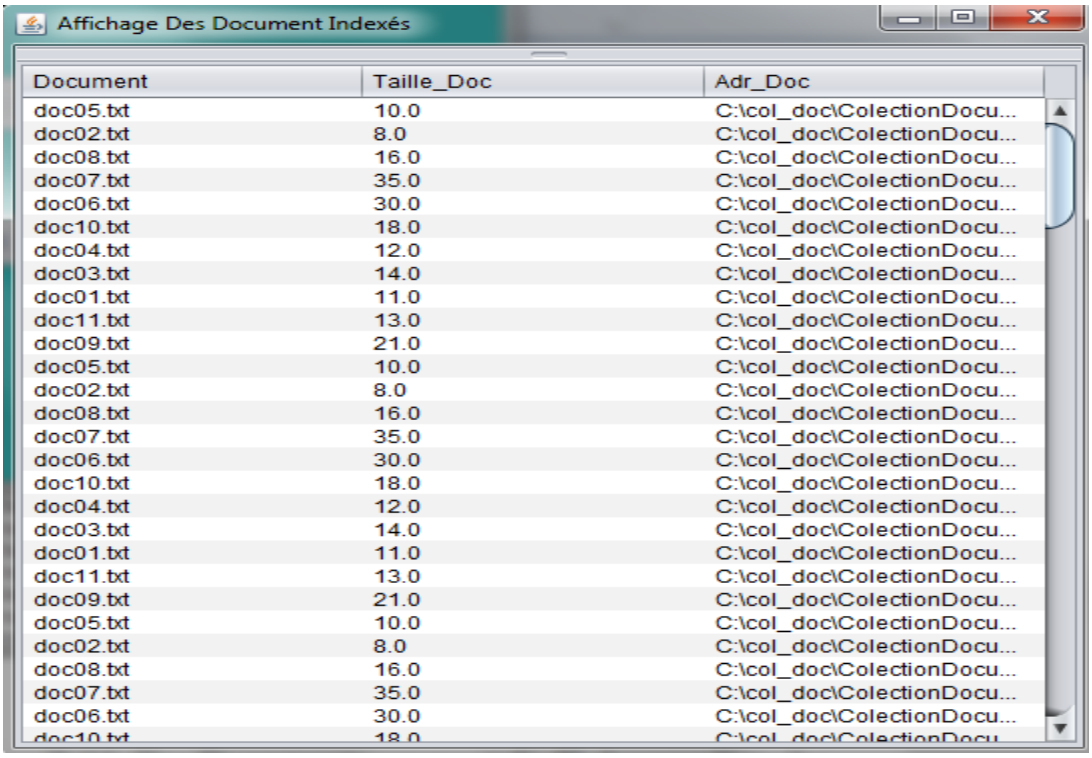


Figure29 : table des documents indexés.

Conclusion :

Dans ce chapitre nous avons présenté l'environnement de développement de notre application. Ensuite nous avons présenté les principales interfaces en interaction avec les différents acteurs de notre système de recherche.

Conclusion Générale :

L'objectif principal de notre travail est d'améliorer les systèmes de RI actuels et de mettre en œuvre un système de recherche d'information sociale qui permet de renvoyer les documents pertinents et frais en réponse à une requête utilisateur tout en orientant la recherche vers le contexte social en utilisant les tags sociaux.

Durant notre étude nous avons acquis de nouvelles connaissances concernant des domaines stratégiques et d'actualité qui sont la recherche d'information dans les réseaux sociaux où nous nous sommes familiarisés avec les notions de tags, de fraicheur d'informations et de modèles de recherche tels que la BM25.

Nous avons également approfondi nos connaissances sur le langage Java et son utilisation pour l'indexation de documents et pour la recherche.

Nous avons l'intension d'étendu le score social à d'autres facteurs autres que les tags et d'expérimenter notre approche sur un cas réel avec une collection de tags issue d'un réseau réel pour estimer vrai influence de la fraicheur d'informations.

Annexe A

Les réseaux sociaux

1. Définition :

Un réseau social (RS) est un ensemble d'entités sociales (tel que des individus ou des organisations sociales) reliées entre elle par des liens créés lors des interactions sociales. Il se représente par une structure ou une forme dynamique d'un groupement social. L'analyse des réseaux sociaux, basée sur la théorie des réseaux, l'usage des graphes et l'analyse sociologique représente le domaine étudiant les réseaux sociaux. Des réseaux sociaux peuvent être créés stratégiquement pour agrandir ou rendre plus efficient son propre réseau social (professionnel, amical,). Il existe des applications internet aidant à se créer un cercle d'amis, à trouver des partenaires commerciaux, un emploi ou autres. Il s'agit de services de réseautage social

Un réseau social représente une structure graphique se modélisant par des sommets et des arêtes. Les sommets désignent généralement des gens et/ou des organisations et sont reliées entre elles par des interactions sociales**. [JOH &BAR, 54].**

2. Fonctionnement d'un réseau social :

Le fonctionnement d'un réseau social est souvent caractérisé par une même et unique procédure : création d'un profil, recherche d'autres profils en relation avec ses centres d'intérêt et enfin la mise en relation (directe ou non). Optionnellement, la prise de contact (commentaire, mail etc.) peut être intégrée. Bien entendu, en fonction de la typologie des réseaux certaines différences apparaissent.

D'un côté les réseaux sociaux de type plate-forme, mettent à disposition des contenus sans création de profil, sans qu'il y ait besoin d'avoir une appartenance à la communauté. Cependant, si l'internaute souhaite faire partie de cette communauté, en diffusant ses propres vidéos par exemple, il doit obligatoirement se créer une fiche d'identité (profil). Ainsi, l'internaute est reconnu en tant que membre de la dite communauté. Une fois membre, il est possible de rechercher des profils, de diffuser et de partager des contenus. Par contre, il n'y a pas de mise en relation à proprement parler. Les membres du réseau ne peuvent interagir entre eux qu'en réagissant, sous forme de commentaires, à des contenus publiés. Ainsi les réseaux de type plate-forme sont essentiellement orientés vers le partage et la diffusion de contenu (souvent des vidéos ou des photos)

De l'autre côté, nous avons les réseaux sociaux personnels (généralistes ou thématiques) qui se rapprochent réellement plus de l'univers social et partage de contenu. La lecture du contenu est toujours libre d'accès, comme pour les réseaux de type plate-forme. Cependant, la création d'un profil est nettement

plus avancée. Ce profil, véritable carte d'identité, est essentiel pour entrer dans la communauté mais surtout pour y participer et créer des liens avec les autres membres. En général, ce profil est présenté par un espace personnel (page web) visible par tous les internautes. Il s'agit d'un espace réservé aux membres où ils ont la possibilité d'y mettre tout ce qu'ils souhaitent : textes, histoires, journal intime, photos de vacances ou encore vidéos. La mise en relation des membres se fait de manière très simple : soit par un lien vers l'autre profil que l'internaute insère manuellement dans *blogroll* soit en invitant l'autre membre à se joindre à son cercle d'amis. Les réseaux sociaux personnels sont donc plus orientés vers la diffusion d'informations que vers la relation entre membres.

Enfin, les réseaux sociaux professionnels sont les plus avancés d'un point de vue des fonctionnalités proposées pour la gestion de sa communauté. La création de son profil est primordiale pour pouvoir profiter de tous les services associés aux réseaux professionnels. L'objectif de ce type de sites est clairement de se construire un réseau le plus grand et pertinent possible. Que ce soit dans le cadre d'une recherche d'emploi, pour trouver des capacités de financement ou encore des opportunités de partenariat, les réseaux sociaux peuvent s'avérer bien utiles pour ce genre de demande. De ce fait, la création de sa fiche personnelle (son curriculum vitae ici) est vitale. Si vous voulez être trouvé ou trouver du monde, il faut que cette fiche soit la plus complète possible tout en définissant bien ses objectifs professionnels. En fait, l'utilisation d'un réseau social en ligne doit se faire à l'image d'un véritable réseau. D'un point de vue de la mise en relation entre membres, les réseaux professionnels sont certainement les plus aboutis tout comme la recherche de profil. Elles aboutissent souvent sur les profils recherchés et en quelques clics, avec un message de demande de mise en contact, l'invitation est envoyée par courrier électronique. Par la suite, la personne invitée a le choix de refuser ou d'accepter la mise en relation. Si elle l'accepte, elle sera en contact direct avec la personne et aura accès à toutes ses informations professionnelles mais également, et surtout, verra le réseau du membre ainsi que son degré de proximité avec ses autres contacts. La création de son réseau n'est pas plus compliquée que cela et permet d'être rapidement en relation avec « le monde entier »…

3. Caractéristiques essentielles d'un réseau social

Un réseau social répond à un besoin primaire d'appartenance sociale de tout individu. S'ajoute à ce besoin d'appartenance, le besoin de tout individu à être estimé par les autres membres de sa communauté. La première

caractéristique du réseau social est de regrouper des personnes autour de concepts communs. Toutes sortes de centres d'intérêts se retrouvent à un même endroit et tout le plaisir est de rencontrer d'autres personnes qui les partagent.

Tout réseau social fonctionne de la même manière : pour devenir membre du réseau chaque personne doit créer son propre profil en procédant à une inscription en ligne sur le site. Ce profil lui servira de clé d'entrée au site et lui permettra d'être identifié par tous les autres membres tout au long de sa navigation sur le site. Chaque membre peut par la suite soit intégrer un groupe existant, soit créer sa propre communauté avec qui il partagera les centres d'intérêts et les motivations liées au site, à travers un système d'invitations et de recherche d'individus membres de la communauté. Des applications sont ensuite proposées par le site pour inciter les membres à être actifs et ne pas seulement se servir du site comme vitrine.

Une remarque doit être faîte en ce qui concerne les modalités d'inscription au site. En effet, tout membre du réseau peut en toute liberté s'inscrire dans les groupes ou communautés qu'il désire sans aucune restriction de nombre.

4. Les enjeux :

Les enjeux dans le domaine des réseaux sociaux sont multiples tant pour les utilisateurs que pour les acteurs.

Du côté des utilisateurs, les réseaux sociaux regroupent toutes sortes d'individus provenant de tous secteurs. Ainsi, sur le réseau *LinkedIn*, les utilisateurs du service viennent de tous les horizons : plus de 120 secteurs d'activités sont représentés et aucun secteur ne représente plus de 11% de la base des inscrits. Scott Allen indiqueque 11,8% des utilisateurs de *LinkedIn*sont PDG, 10,2% sont vice-présidents ou directeurs généraux et 1,3% sont membres d'un conseil d'administration. Cela ne vaut pas que pour les réseaux sociaux professionnels. Sur *MySpace*, par exemple, il est possible de trouver la plupart des grands groupes de musique qui ont créé leur page *MySpace* officielle (U2, Metallica, Prodigy, Black EyedPeas…). De ce fait, il est possible de retrouver quasiment n'importe quelle personne quel que soit sa situation géographique ou son poste au sein de la société. En plus de pouvoir visualiser des informations sur les personnes, les réseaux sociaux offrent l'opportunité d'entrer en contact avec toutes ces personnes. Les liens entre tous les membres d'un réseau sont les profils personnalisés, ce que l'on peut appeler la carte d'identité numérique. Ainsi, les réseaux sociaux permettent de gérer son identité numérique ainsi que sa réputation en ligne. Lorsque l'utilisateur remplit sa fiche, il a le choix d'y intégrer les informations qu'il souhaite et de cacher celles qu'il estime privées.

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

Du côté des grands acteurs, il existe également de nombreux enjeux. Il s'agit essentiellement là d'enjeux économiques mais aussi de visibilité. Ainsi les milliers d'utilisateurs inscrits offrent, indirectement, une source de revenu importante. *MySpace* compte entre 80 et 100 millions de profils créés[14] dont un million rempli en détail. Il est ainsi possible de les cibler très précisément pour leur proposer du contenu publicitaire en adéquation avec leurs passions et leurs centres d'intérêts. Outre la publicité directe, les réseaux sociaux, grâce à leurs nombres importants d'utilisateurs, offrent aux grands groupes (audiovisuels, musicaux, informationnels…) un beau support de diffusion avec un large public qu'ils peuvent toucher de manières très pertinentes. Ainsi, des chaînes comme celles du groupe *Fox* ou encore des maisons de disques profitent des réseaux sociaux pour diffuser des contenus adaptés au profil des membres de communautés. De plus, il est très facile d'infiltrer la communauté (création de page personnelle pour un utilisateur fictif, mise en ligne de vidéo marketing…) afin de mettre en place une opération de marketing viral.

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

5. Classement des réseaux sociaux :

Plusieurs catégories de réseaux sociaux existent. Aussi est-il tentant de classifier, de catégoriser les réseaux sociaux

On peut effectuer un classement des réseaux sociaux selon 3 catégories comme le fait Wikipédia :

- Réseaux ouverts (favorisent toutes rencontres sans conditions ou sans abonnement)
- Réseaux sur invitation (il faut être invité par l'un de ses membres)

- Services en ligne de réseautage professionnels (favorisent les rencontres professionnelles, les offres de poste et la recherche de profils)

Mais d'autres classifications sont possibles :

- Les networkings : les plus utilisés dans les milieux professionnels. Ils permettent des échanges entre professionnels sur des plateformes en évolution perpétuelles.

- Les bloglikes : ils ressemblent vaguement à des blogs. Ils sont souvent le refuge d'adolescents en mal de reconnaissance.
- Les spécialisés : ils regroupent des communautés autour d'un thème bien précis
- Le micro-blogging : chat public, on y met tout ce qu'on y fait minute par minute, histoire de montrer aux autres qu'on est très actif.
- Les fourres-tout : ce sont les inclassables qui se servent du collaboratif ou du participatif pour alimenter leur service. On peut y trouver, les sites de partage d'avis
- Les open-sources : ou plutôt les plateformes qui permettront de créer son propre réseau social

D'autres auteurs on classer les réseaux sociaux en 7 catégories :

- Les réseaux sociaux d'affaires et d'emplois
- Les réseaux sociaux de jeunes, «bloglikes»
- Les réseaux sociaux « privés » (sur invitation)
- Les réseaux sociaux spécialisés : vidéo, images…
- Les réseaux sociaux communautaires et thématiques, y compris les digg-likes
- Les réseaux sociaux «identité numérique» : MyBlogLog
- Les réseaux sociaux micro : micro-blogging, micro-vidéo, etc.

6. Le développement des réseaux sociaux :

Dans un environnement en perpétuelle évolution comme le Web, tous les acteurs ont besoin de s'adapter et de trouver le bon compromis entre un modèle économique rentable mais non contraignant pour les utilisateurs et une communauté toujours avide de nouveaux services. Cette partie propose une étude des points clés et primordiaux pour un bon développement d'un réseau social : communauté, modèle économique, technologie et fonctionnalités

6.1. La communauté :

Le développement des réseaux sociaux ne peut se faire sans une communauté solide et grandissante. Il faut qu'ils évoluent en même temps que leurs utilisateurs. Comme pour *MySpace*, à la mode aujourd'hui, rien ne dit que demain, les adolescents auront d'autres modes et, par conséquent, changeront de supports.

Ainsi, il est important qu'un réseau social en ligne soit « ouvert » et « multi-culturel ». Il se doit de refléter la société : des origines, des cultures, des langues et des intérêts différents. Avec ces caractéristiques premières, le réseau social peut ainsi regrouper un public bien plus large. N'est-ce pas là, le but de tous réseaux sociaux que de rassembler un maximum d'utilisateurs ? Nous savons maintenant, avec un peu plus de 3 années d'expérience sur le phénomène, que le seuil critique, pour une communauté spécialisée, est d'environ 100 000 membres. Ce seuil, au-delà duquel l'effet viral commence à entraîner, de manière exponentielle le système. Par exemple, *Viaduc*, réseau social orienté professionnel, a mis près de deux ans à atteindre 100.000 membres. Moins de 8 mois plus tard, *Viaduc* avait multiplié par 5 son nombre de membres.

Il est également essentiel, tout comme sur le marché « classique », de répondre aux besoins des utilisateurs. Comme le confirme Bertrand Duperrin, « dès lors que les réseaux sociaux répondent à un besoin tant de l'individu (aspiration à exister, communiquer, échanger) que de l'entreprise (créer du lien, favoriser le partage et la collaboration)…», il y a de fortes chances pour qu'ils réussissent à s'imposer. En effet, un nouveau système est plus facilement assimilé s'il correspond à un besoin plutôt qu'à un simple service supplémentaire. Un utilisateur adoptera ainsi plus rapidement un outil répondant à ses attentes.

C'est notamment ainsi que beaucoup de réseaux sociaux de niche ont réussi à se faire une place contre les géants généralistes.

6.2. La technologie :

Un point qui peut sembler être un détail mais qui, au contraire, s'il n'est pas bien analysé peut faire beaucoup de dégâts.

Il faut tout d'abord savoir que le système d'un réseau social est bien plus complexe que n'importe quels autres sites, forums ou blogs. L'algorithme de mise en relation et de navigation au sein de ces liens demande beaucoup de ressources. De même les bases de données doivent gérer une énorme quantité d'informations et supporter une charge importante d'enregistrement. Dès lors, il est essentiel de bien dimensionner la puissance de son parc de serveurs et d'estimer correctement la bande passante nécessaire.

Effectivement, la technologie n'est pas critique pour débuter mais elle est indispensable pour atteindre le succès. Si 10 000 ou 100 000 utilisateurs sont gérables, quelques grands noms comme *Friendster*ou *Orkut*se sont effondrés

une fois le million d'utilisateurs inscrits. Par ailleurs, l'architecture doit être bien réfléchie dès le début de l'aventure car tous changements de structure (langage, serveurs etc.) seraient difficilement envisageables.

6.3. Les fonctionnalités :

Certaines personnes se demandent comme *MySpace* a pu atteindre ce niveau avec une interface aussi peu agréable et austère. La réponse doit se situer au niveau des fonctionnalités et de la navigation sur le site. Effectivement, *MySpace* propose un nombre impressionnant de fonctionnalités et de services à valeurs ajoutées notamment le fait de pouvoir personnaliser entièrement son espace personnel et cela de manière très simple et sans aucune connaissance technique. C'est là, la preuve que les fonctionnalités et la simplicité d'utilisation sont primordiales dans le succès d'un réseau social.

Nul besoin de fioritures technologies mais il est vrai que la méthodologie de développement *AJAX* peut simplifier l'expérience de l'utilisateur. Imaginez-vous entrain de pouvoir classer vos contacts (amis, relations professionnelles…) comme vous rangeriez des photos de vacances. La gestion des liens entre vos contacts pourrait être simplifiée si d'un clic vous pouviez vous connecter à une personne. Tout cela est possible avec *AJAX*. De même, lors de la création de votre page personnelle ou curriculum vitae, il vous suffirait de glisser/déposer les éléments désirés sur la page et de les renseigner ensuite.
D'autres fonctionnalités peuvent être imaginées comme la création de *widgets*. Ces petits modules ne proposant qu'une fonctionnalité (recherche, gestion photo, lecteur audio ou de flux d'informations…) peuvent être ajoutés ou supprimés selon les besoins. Il est également envisageable de coupler certaines fonctionnalités comme celle de partage de fichiers via les réseaux de P2P. Ainsi les artistes ou même les utilisateurs peuvent facilement se partager et s'envoyer des fichiers en les mémorisant sur leur propre ordinateur.

Enfin, il y a des fonctionnalités plus utiles pour les machines que pour les utilisateurs mais qui, au final, rendent bien service à ces derniers. Il s'agit des micro-formats qui permettent de normaliser du contenu afin d'être plus facilement identifiable par les robots, agents intelligents ou moteurs de recherche. Ces micro-formats font entrer une nouvelle notion : la sémantique. Ils vont permettre de décrire et de donner du sens à un curriculum vitae, à une fiche d'identité ou à un espace personnel de manière totalement compréhensible par les machines pour qu'elles puissent les ressortir de façon pertinente lors d'une requête précise effectuée par les utilisateurs. Plusieurs micro-formats sont ainsi apparus comme *FOAF* (Friend Of A Friend permet de décrire un individu et ses

connaissances), *Hcard (*format dédié à la description des individus, des entreprises et organisations) ou encore *OpenID*(format et service permettant la gestion décentralisée de l'identité). D'hors et déjà, ces micro-format sont utilisés dans différents services comme *Technorati*(annuaire universel de blogs), *Naimz*ou encore *ClaimID*qui sont des services de gestion d'identité numérique.

7. Présentation de quelques réseaux :

Nous allons vous présenter les réseaux les plus importants et à connaître dans une recherche d'emploi ou de stage. Nous avons décidé d'ajouter Facebook parce que c'est un réseau incontournable aujourd'hui.

- **Facebook :**

- Il est créé par Mark Zuckerberg en 2004 à Harvard aux Etats-Unis
- L'inscription est gratuite
- Il a 500 millions de membres actifs
- C'est un outil de partage et d'échange

- **Twitter :**

- Il est crée en 2006 à San Francisco
- C'est un outil de réseau social et de microblogging
- Il a 11.5 million de membres

«Discover what's happening right now, anywhere in the world»

- **Linked ln:** est un réseau professionnel en ligne, vous pourrez présenter votre carrière professionnelle mais également être en relation avec des personnes issues du même univers professionnel que vous et il permet une gestion de réputation.

- **Viadeo :**

- Il est crée en 2003 en Allemagne
- C'est Réseau Social Professionnel International (n°1 en Europe)
- C'est un Leader des marchés germanophones (Allemagne, Suisse, Autriche…).

Annexe B

Présentation de *langage java*

1. Historique de java :

Dans le but de trouver des solutions qui simplifient et fiabilisent l'exploitation des ordinateurs, la société ***Sun Microsystems*** sous l'impulsion de *Scott McNealy*, instaura en 1990 un groupe de travail dénommé Green, dont les principaux acteurs étaient *PatrickNaugthon*, *Gosling James* et *Mike Sheridan*.
Les axes de réflexion de ce groupe se conduisirent vers la conception d'un environnement d'exploitation indépendant du hardware, pouvant être implémenté aussi bien dans des appareils variés que les PDA (assistant numérique personnel), les téléviseurs, les magnétoscopes, les téléphones mobiles…etc.

Ce système devrait permettre de contrôler de manière conviviale et interactive ces différents appareils, et leur faire échanger des informations via des réseaux informatiques.la réalisation d'un tel système nécessitait un langage de programmation compact, portable, robuste, performant, sécurisé et répondant aux contrains des applications en temps réel.
Au début les membres de green optèrent pour le langage C++, ils s'accrochèrent des 1991 sur la création d'un prototype réalisé avec un langage choisi par *James Gosling*(***OAK***).ce dernier a fait une synthèse de plusieurs langages de programmation : il s'inspire de la syntaxe du langage C++, et techniques éprouvées en ***Smaltalk*** et autres langages de programmation (organisation en classes, utilisation de ramasse-miettes (garbagecollector), exécution à l'aide d'une machine virtuelle, gestion des exceptions…etc.). Et c'est en supprimant la plupart des mécanismes qui diminuent la fiabilité du code (la gestion des pointeurs, la gestion de la mémoire).

En plus ce langage a été enrichi par des mécanismes qui lui sont propres, et ce afin de mieux répondre aux fortes contraintes imposées par l'exécution de programmes transmis par internet.
En ***aout 1992*** le prototype a été présenté au public qui l'accueillit avec beaucoup d'enthousiasme. L'avenir d'***OAK*** semblait tellement prometteur que la filiale First Person Inc, fut créé avec objectif de l'exploitation du seul potentiel offert par l'***OAK***. En ce moment l'internet été l'un des événements qui marquait le monde informatique, et qui fut certainement la sortie de la première version du navigateur ***Mosaic*** qui ouvrait les portes de world wide web.
Le ***23 mai 1995***, Java son apparition officielle dans la communauté internet qui pouvait désormais être téléchargé gratuitement : compilateur, machine virtuelle librairies, documentations et spécifications sur le site de ***SUN***.
Très rapidement, ***Netscape***, puis ***Microsoft*** intégrait Java à leurs propres produits, alors que ***Symantec*** développait un environnement de développement complet pour Java.

Maintenant les qualités de la maturité du langage ainsi que la liste grandissante desbibliothèques de classes disponibles, ont permet à *Java* d'être

un acteur de plus incontournable dans la conception de solutions dédiées à internet ou non.

Pour conclure on dira que Java est à la fois :

Un *langage de programmation* orienté objet.

Une *plate-forme*: un environnement pour l'exécution et le développement des programmes Java.

Une *machine virtuelle* : la JVM (Java Virtual Machine).

Des *APIs* (Application Programming

2. *Les caractéristiques du Java :*

Le langage Java a été conçu à partir des langages objets *SmallTalk*et *C++*. Ses caractéristiques sont résumées ci-dessous :

- ***Simplicité*** :

D'une syntaxe familière aux programmeurs C et C++, il en a été dépouillé de tous les mécanismes complexes (gestion des pointeurs, gestion de la mémoire, l'héritage multiple, …)

- ***Orienté Objet*** :

Tout est objet, Il faut concevoir ainsi. Contrairement au C++ qui autorise l'écriture du code des fonctions en dehors des classes, Java oblige le programmeur à définir les méthodes comme partie intrinsèque des classes.

- **Orienté réseau et distribué** :

Développement d'applications réparties. Java permet d'accéder facilement à des données distantes sur le réseau, et de réaliser des applications client/serveur grâce aux classes paquetages de communication (java.net).

- ***Multitâche (multi-thread)*** :

Java permet de concevoir des applications capables d'effectuer plusieurs actions (*thread*) en même temps. Un même programme peut par exemple à l'aide de trois *threads* faire un calcul, tout en chargeant une vidéo et en diffusant un son.

- ***Dynamique*** :

Un programme Java charge les classes qu'il utilise (y compris les classes de base) en cours d'exécution. La modification d'une classe ne nécessite pas une recompilation de l'application pour prendre en compte les changements effectués, car les liens entre les objets sont résolus lors de l'exécution.

- ***Robuste*** :

L'objectif de java est de permettre la réalisation de logiciels fiables. Les caractéristiques suivantes permettent d'atteindre cet objectif :

Typage fort, pas d'accès aux pointeurs (réduisant les risques d'erreurs), Gestionnaire de la mémoire, mécanisme de gestion d'exception.

- ***Sécurisé*** :

Eviter d'exécuter du code dommageable. De plus java possède des API destinés au cryptage, à la gestion de l'authentification, … Interfaces).Java

propose dans son paquetage **JAXP** (Java API for XmlProcessing) les outils nécessaires pour exploiter facilement des définitions XML.

- ***Interprété, Indépendant des architectures matérielles*** :

Le code produit par le compilateur n'est pas du code machine. C'est un code intermédiaire (*bytecode*) simple et rapide (plus rapide qu'un langage de script entièrement interprété) à traduire en langage machine par un interpréteur (*Java Virtual Machine - JVM*) gérant pour sa part les spécificités du système hôte. Il est cependant possible de recompiler le byte code afin de produire du code natif et d'atteindre les performances identiques à celles du langage C.

- ***Portable :*** étant indépendant des architectures matérielles, java est par conséquent portable.
- ***Performant :***

Un compilateur générant directement du code machine a été ajouté aux outils java, il s'agit du compilateur Just in time, qui permet d'obtenir des temps d'exécution comparable à ceux obtenus par un programme en C++.

3. Terminologie :

En Java, à l'exception de quelques primitives, tout est objet. Cette organisation des composants logiciels en objets, contribue à faciliter la modélisation et l'implémentation d'une solution ainsi que la réutilisabilité des composants développés.

Le rôle des technologies orientées objets, est de fournir un certain nombre de mécanisme destinés à organiser ces composants en objets. Dans ce qui suit nous allons présenter les différents concepts utilisés par une telle technologie.

1. Encapsulation :

✓ Notion d'objet :

C'est une entité qui possède une identité et des caractéristique, et qui est susceptible d'effectuer des actions. Un objet est caractérisé par plusieurs notions :

- **L'identité :**

L'objet possède une identité, qui permet de le distinguer des autres objets, indépendamment de son état. On construit généralement grâce à un identifiant découlant naturellement du problème (par exemple un employé pourra repérer par un numéro de sécurité sociale).

- **Les attributs** :

Il s'agit des données caractérisant l'objet .ceux sont des variables stockant des informations sur l'état de l'objet ;

- **Les méthodes** :

Les méthodes d'un objet caractérisent son comportement, c'est-à-dire l'ensemble des actions (opérations) que l'objet est amené à réaliser. Ces

opérations permettent de faire réagir l'objet aux sollicitations extérieures (ou d'agir sur les autres objets). De plus, les opérations sont étroitement liées aux attributs car, leurs actions peuvent dépendre des valeurs des attributs ou bien les modifier.

- ✓ **Notion de classes :**

Elle peut être définie comme le type d'un objet, et comme un objet est capable de gérer à la fois ses données et ses actions, alors une classe doit être définir par une structure de données et un comportement. Les objets qui possèdent les mêmes caractéristiques (structures de données et comportements) sont regroupés en une classe.
Dans un programme, l'objet est créé à partir d'une classe on dit qu'il s'agit d'une instance de cette classe.

- ✓ **Un package :**

C'est une bibliothèque, servant à structurer un ensemble de classes. Cette organisation facilite à la fois la recherche de l'emplacement physique des classes quand cela est nécessaire et la distinction entre différentes classes de même nom. Elle permet en outre, de nuancer des niveaux d'accès aux membres (attributs et méthodes) d'une classe, selon que ces membres appartiennent au non au même package, et d'une manière globale, rend l'ensemble plus lisible.

2. **Le polymorphisme :**

Offre la possibilité d'associer à un comportement, une implémentation différente de l'objet auquel on se refaire (**ex** : dessiner à l'écran un carré et un cercle implique dans les deux cas de manipuler un certain nombre de pixels, cependant la manière d'implémenter ce tracé se fera différemment). Le polymorphisme augmente la généricité, et donc la qualité du code.

3. **L'héritage :**

L'héritage permet à une classe d'objets de bénéficier de la structure de données et du comportement d'une classe « mère », afin de prendre en compte les spécificités de la classe « fille », sans savoir à redéfinir ce que les deux classes ont de commun.

4. **L'abstraction :**

Introduit le mécanisme favorisant la dissociation entre la déclaration d'une classe et son implémentation.

5. **Le Kit de Développement Java (JDK) :**

Ecrire un programme Java ne nécessite pas d'outils spécifiques : un éditeur de texte tel Notepad suffit largement. Le programme Java, qui porte l'extension **.java** doit être compilé afin de générer un fichier byte code, pourtant l'extension **.class,** qui sera interprétable par une machine virtuelle Java.

Pour se faire, il est nécessaire de télécharger sur le site de Sun

- ✓ **Javac :**

Il s'agit du compilateur java.il traduit un fichier source d'extension**.java** en un fichier byte code d'extension **.class.** une partie de ses fonctions concerne la sécurité afin que le code généré ne risque pas d'effectuer des instructions illégales.

- ✓ **Java** :

C'est un interpréteur java, c'est-à-dire une implémentation de la machine virtuelle java.il traduit en langage machine les fichiers déjà compilés par **Javac,** après avoir effectué un certain nombre de contrôles (**ex :** s'assurer que l'exécution d'un programme en prévenance d'un serveur distant, ne porte préjudice à l'intégrité d'une machine locale).

- ✓ **AppletViewer:**

Ce programme est un interpréteur java qui possède la spécificité de ne pouvoir exécuter que des applets. Disposant de sa propre interface graphique, il permet de tester ces derniers sans avoir recours à un navigateur web.

- ✓ **JRE :**

C'est un interpréteur allégé de java qui n'a pas besoin d'installer tous le JDK pour exécuter un programme compilé par le javac. Il est destiné à des plates-formes clientes qui ne développent pas d'application Java mais qui les utilisent.

- ✓ **Jdb**:

Ce débogueur permet de détecter les erreurs de programmation.

- ✓ **Javadoc :**

Cet utilitaire permet de construire, à partir des commentaires insérés dans des sources Java et sous condition que ces derniers respectent une certaine syntaxe, des fichiers HTML documentant les classes, méthodes,…développées dans des sources.

- ✓ **Javap**:

Cet utilitaire permet de désassembler un fichier compilé (byte code) en code source (java).

- ✓ **Jar :**

Cet utilitaire permet d'archiver plusieurs classes Java en un fichier d'archive Jar ou exécutable Jar (à partir de la version 1.1 du JDK).

Parmi les utilitaires utilisés, on a aussi : Javah, Jit, Native2ascii, Rmic, Rmirejistry, Javakey,Keytool, Jarsigner, Policytool…….

6. Les API Java :

Le JDK fourni les API de base, ces derniers sont structures en packages, elles contiennent de classes pouvant être réutilisées pour construire des programmes plus complexes. Les classes ressemblaient dans le même package présentent généralement des fonctionnalités fortement connexes. Voici maintenant quelques packages :

- ***Java.lang***:

Ce package fournit des classes de base pour la conception du langage de programmation Java.

- ***Java.util***:

Ce package contient les classes qui permettent de manipuler les vecteurs, les dates, les fichiers Zip, les fichiers d'archives Jar.

- ***Java.applet***:

Ce package fournit les classes nécessaires à la création d'une applet ainsi qu'à la communication entre une applet et sont contexte.

- ***Java.awt***:

Ce package contient toutes les classes pour créer des interfaces graphiques très complètes, et pour manipuler des graphiques et des images.

- ***Java.io*** :

Ce package fournit des classes pour les entrées/sorties du système.

- ***Java.net*** :

Ce package fournit des classes pour implémenter des applications réseaux.

- ***Java.math***:

Ce package fournit des classes utilisées pour l'arithmétique des nombres entiers et des nombres décimaux à précision arbitraire.

- ***Java.security***:

Ce package fournit des classes permettant de mettre en œuvre des procédés de sécurité, tel que la gestion des droits d'accès à différentes ressources (fichiers, réseau,…).

- ***Java.sql***:

Ce package fournit un certain nombre d'interface et de classes, permettent la connexion à une base de données ainsi que la soumission de requêtes SQL.

- ***Java.text***:

Ce package fournit des classes permettant la manipulation de données (texte, date, nombres,…) indépendamment de la langue et de la région de l'utilisateur.

- ***Javax.accessibility***:

Ce package fournit un certain nombre de composant de construire des interfaces utilisateurs, utilisant des technologies d'assistance pour les terminaux en braille, ou dans le cas de système à reconnaissance vocale.

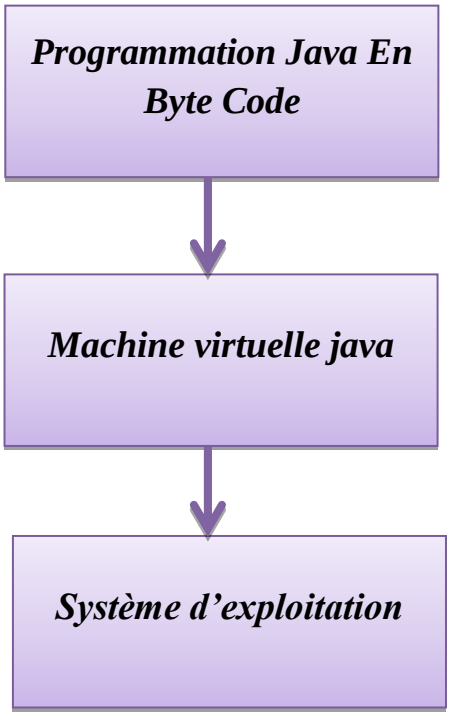
- ***Javax.swing***:

Tout comme le package java.awt, ce package contient de nombreuse classes permettant de crées des interfaces graphiques très complètes. Swing est en fait construit au-dessus de java.awt, et offre une palette plus complète et plus travaillée de composants.

Il existe encore d'autre package à fonctionnalité différentes *:*
Java.beans, Java.rmi, Javax.naming, Javax.rmi, Javax.sound, …etc.

7. La machine virtuelle Java :

Afin d'exécuter une application java, la présence sur la plate-forme d'exécution de l'implémentation d'une machine virtuelle java est indispensable. Cette dernière aura la charge de traduire le *byte code* en instructions exécutables par le processeur de la machine hôte, on trouve ainsi plusieurs implémentations de la machine virtuelle, pour chaque plate-forme (une pour *Microsoft*, une pour linux…etc.) .mais la JVM détient aussi d'autres responsabilités que nous allons rencontrer en détaillant ses principaux composants.



- ***Le chargeur de classes dynamiques (class loader) :***

Ce module charge les classes nécessaires à l'exécution d'un programme en mémoire. Il est capable d'aller chercher des classes de l'API Java, et aussi celles crées par un développeur sous réserve que leur emplacement ait été spécifié dans la variable *CLASSPATH*. Les navigateurs possèdent en outre un chargeur spécifique, capable d'aller rechercher une classe via le réseau.

- ***Le vérificateur de bytecode (bytecodeverifier) :***

Si le compilateur est censé d'assurer que le code source temps qu'il est entrain de traiter ne risque pas de violer un certain nombre de règles de sécurité, alors le vérificateur de byte code est intégré dans la machine virtuelle, pour s'assurer que le code source en provenance de serveurs distants, ne contient pas un code qui risque de saturer les ressources de la JVM, ou d'effectuer des

opérations illicites sur le système (manipulation d'adresses mémoire via des pointeurs, appel des méthodes avec des paramètres non-conforme…etc.

- ***Le gestionnaire de sécurité :***

Ce module à la charge de veiller à ce que l'exécution d'un programme ne vient pas remettre en cause l'intégrité des ressources de la machine hôte. Il considère les classes selon deux catégories : les classes trusted, comme les classes locales, qu'ils s'agissent de classes issues de l'API java ou de classes crées par le développeur sur la machine locale, et les classes untrusted qui proviennent d'un serveur distant. Ce module pourra interdire à une applet transmise par un serveur distant de lire un fichier sur la machine hôte, si aucune autorisation ne lui est accordée.

- ***Le moteur d'exécution :***

C'est le cœur de la machine virtuelle, car il prend la charge de convertir le *byte code* en instruction exécutables par le processeur hôte.

Ce moteur d'exécution, un véritable processeur virtuel, dispose de son propre jeu d'instructions (environ 200), comme par exemple des instructions pour les opérations arithmétiques et logiques les sauts conditionnels.

Références Bibliographiques

Bibliographies:

[ABC, 98]
Abrams D, Baecker R. et ChignellM : Information archiving with bookmarks:personal web space construction and organization. *Dans CHI '98: Proceedings ofthe SIGCHI conference on Human factors in computing systems*, p. 41–48, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[BAT & AL. 07]
Bateman, S., Brooks, C., Mccalla, G., Brusilovsky, P., « ApplyingCollaborative Tagging to E-Learning ». *Proceedings of the 16th International World Wide WebConference (WWW2007)*, Banff, Alberta, Canada, 8-12 mai 2007.

[BOU & AL, 04]
A Framework for Analysis of Data Freshness", Actes du 1st Int.Workshop on Information Quality in Information Systems (IQIS), Paris, France, 2004.

[CAL, 98]
De la qualité de l'information à la qualité de la documentation", Document Numérique, vol.12, nº.1, 1998, p. 37-52.

[CHE & AL. 10]
Chen, J.-M., Chen, M.-C., Sun, Y. S., « A novel approach for enhancing studentreading comprehension and assisting teacher assessment of literacy », *Computers &Education*,vol. 55, n°3, 2010, in press.

[CHO, 00]
"Synchronizing a database to improve freshness", Actes de la ACM Int. Conf. on Management of Data (SIGMOD), Dallas, USA, 2000, p. 117-128.

[CON & CUL 10]
Conole, G., .Culver, J., « The design of Cloudworks: Applying socialnetworking practice to foster the exchange of learning and teaching ideas and designs », *Computers & Education*, vol. 54, n° 3, 2010, p. 679-692.

[DEN, 00]
Quality Control Tools User Requirements - Part 1, Deliverable D2.1.1, Report URR1-QCT1 on WP2 Task 2.1, TIPS European project IST-1999-10419, July

[DEN, 02]
"QCT and SF services in Torii: Human Evaluations of Documents Benefit to the Community", Actes du Workshop on Personalization Techniques in

Bibliographies:

Electronic Publishing on the Web: Trends and Perspectives, Malaga, Spain, 2002, p.105-114.

[GEO, 07]
Georges BADR, Recherche d'Information sur le Web: Impact de la structure des documents sur la pertinence des résultats, Université Paul Sabatier & Institut National Polytechnique de Toulouse, 2007.

[GER, 04]
GerReport on the Dagstuhl Seminar: Data Quality on the Web", SIGMOD Record, vol. 33, nº 1, March 2004.

[HAR, 06]
Un modèle de qualité de l'information", Actes des JournéesExtraction et Gestion de Connaissances (EGC), Lille, France, 2006, p. 299-304.

[HUY, 09]
Huynh-Kim-Bang, B., Indexation de documents pédagogiques :fusionner les approches du Web Sémantique et du Web Participatif, Thèse de doctorat, Université Henri Poincaré - Nancy I, 2009, 274p.

[KAL & AL. 09]
Kalamatianos, A., Zervas, P., Sampson, D.G., « ASK-LOST 2.0: AWeb-Based Tool for Social Tagging of Digital Educational Resources », *Proceedings of theNinth IEEE International Conference on Advanced Learning Technologies (ICALT 2009)*, Riga, Latvia, 15-17 juillet 2009, p. 157-159.

[KAR, 05]
Karen SAUVAGNAT, Thèse Doctorat, "Modèle flexible pour la Recherche d'Information dans des corpus de documents semi-structurés", Université Paul Sabatier Toulouse, 2005.

[MIL, 06]
Millen, D., Feinberg, J., Kerr, B., « Dogear: Social Bookmarking in the enterprise »,*Proceedings of the 24th international conference on Human Factors in computing systems (CHI2006),* Montreal, Canada, 22-27 avril 2006, p. 111-120.

[KHA, 08]
Mohamed KHAZRI, Mohamed TMAR, Mohamed ABID, Mohand BOUGHANEM, Proposition pour l'intégration des réseaux petits mondes en recherche d'information, 2008.

Bibliographies:

[MOR, 06]
Moreau F. Revisiter le couplage traitement automatique des langues et recherche d'information, These de doctorat de l'Universite de Rennes 1, 2006.

[NAU, 99]
Quality-driven integration of heterogenous information systems", Actes de la Int. Conf. on Very Large Databases (VLDB), Edinburgh, 1999.

[NAU, 00]
Assessment Methods for Information Quality Criteria", Actes de la Int. Conf. on Information Quality (IQ), Cambridge, USA, 2000.

[NEL & AL. 09]
Nelson, L., Held, C., Pirolli, P., Hong, L., Schiano, D., Chi, Ed H, « With a little help from my friends: examining the impact of social annotations in sensemaking tasks », *Proceedings of the 27th international conference on Human factors in computing systems (CHI2009)*, Boston, MA, USA, 4-9 avril 2009, p. 1795-1798.

[NGA ,09]
NGASSA Hubert Landry, SINI Sarah, M_emoire de licence "Indexation de document
XML par l'approche XPATH ACCELERATOR", 2009

[PER, 06]
Data Quality Evaluation in Data Integration Systems, PhD Thesis, Université de Versailles, France and Universidad de la República, Uruguay, 2006.

[PIP, 02]
Data Quality Assessment", Communications of the ACM, vol. 45, nº 4, April 2002.

[RED, 96]
Data Quality for the Information Age, Artech House, 1996.

Bibliographies:

[TAM, 98] :
L. TAMINE, les Systèmes de recherche d'informations : Reformulation de Requête et Apprentissage basés sur l'algorithme génétique, thèse de Magister en Informatique, Université de Mouloud Mameri de Tizi-Ouzou ,1998**.**

[SAL, 87]
Gerard Salton *Approaches in Automatic Text Retrieval*, Cornell University, Ithaca, NY, 1987.

[SAL, 83]
SALTON.G et MCGILL.M.J, Extended Boolean information retrieval, 1983

[SAU, 05]
Sauvagnat K. Modèle flexible pour la recherche dans des corpus de Documents semi-structures, Thèse de doctorat, Toulouse 3, 2005.

[SCH & AL, 73]
Schurtz, al, Structures of the Life World. Northwestern University Press, Evanston, Ill., Ed. ACM, New York, Sept. 1990, p. 45-61, 1973.

[SPA, 72]
Sparck Jones, K. *A statistical interpretation of term specificity and its application.* In Retrieval Journal of Documentation 28:1; p. 11-21, march 1972.

[TEB, 04]
Tebri H. Formalisation et spécification d'un système de filtrage incrémentale D'information. Thèse doctorat de l'université Paul Sabatier, Toulouse,
2004.

[WAN & AL, 96]
Wan, al, Beyond accuracy: What data quality means to data consumers", Journal on Management of Information Systems, vol. 12, nº 4, 1996, p. 5-34.