MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE MOULOUD MAMMERI DE TIZI-OUZOU
FACULTE GENIE ELECTRIQUE ET INFORMATIQUE
DEPARTEMENT INFORMATIQUE

Mémoire

En vue de l'obtention du Diplôme de master en informatique

Thème

Intégration des commentaires dans la recherche d`information sociale

Proposé et dirigé par :

Melle : L. BOUGCHICHE

Réalisé et présenté par :

Melle: MEKSEM Ouarda

Melle: OUZEROUT Hayet

Promotion 2011-2012

Remerciement

Nous faillirons à la tradition si nous n'exprimions ici notre gratitude envers tous ceux qui ont collaboré à la réalisation de notre projet, pour cela :

- *Nous remercions d'abord le bon dieu qui nous avoir donné le courage, la patience, la santé et la volonté d'arriver à la fin de ce travail.*

- *Nous tenons à exprimer notre profonde reconnaissance et nos vifs remerciements à notre promotrice, M^elle L. BOUGCHICHE, pour tous ses conseils précieux et ses encouragements.*

- *Nous adressons nos remerciements également aux jurys, pour l'honneur qu'ils nous ont fait en acceptant de juger notre travail.*

- *Nous tenons à remercier, par ailleurs, tous les professeurs du département d'Informatique qui nous ont apporté leur aide, d'une manière ou d'une autre, tout au long de notre cursus*

- *Notre gratitude va également à nos parents, pour leur aide et leur soutien.*

- *Enfin, nous tenons à remercier tous nos frères et sœurs, nos proches, et nos amis, ainsi que tous les étudiants de notre promotion. Surtout notre sœur MESTOUR Linda.*

Dédicaces

Je dédie ce modeste travail à :

La mémoire de ma grand-mère, Rabie et ma tente *que je peux jamais oublier*

Celle que je garde dans mes pensées *ma chère mère adorée*

Son soutien m'a beaucoup inspiré *c'est son rêve que je viens de réaliser*

Mon père que je remercie *d'avoir avec moi patienté*

Ma sœur Iidia notre préféré *Que dieu la protège*

Mes adorables frères(kamel,hamid,mourad,hacene) *qui m'ont compris et aimé*

Mes grands parents *a qui je souhaite la langue vie*

Mes oncles *et leurs familles sans exception*

A notre adorable Ouannes *Que dieu le protege*

Ma tente aldjia et sa famille *a qui je souhaite la belle vie*

A Djazia,Radia,Nina,Kenza *a qui je souhaite un bon avenir*

Sans oublier Ioussif *a qui je souhaite la réussite et la santé*

Tous ceux que je connais *mes camarades et mes amis*

Farid,Idir,Amir,nounou, Azouaoue,Karim *Lila,Lyly,wissame, Adelia,Amel*

Mon binôme ouarda, mon amie *je lui espère la réussite et la belle vie*

Hayet

Dédicaces

Je dédie ce modeste travail à :

La mémoire de ma grand-mère *que je peux jamais oublier*

Celle que je garde dans mes pensées *ma chère mère adorée*

Son soutien m'a beaucoup inspiré *c'est son rêve que je viens de réaliser*

Mon père que je remercie *d'avoir avec moi patienté*

Ma sœur nabila qui m'a supportée *sa présence m'a beaucoup aidée*

Mon adorable frère Mohammed *qui m'a compris et aimé*

Mon frère :yahia notre préféré *a qui je souhaite la réussite et la santé*

Mon mari mouh *Grâce à lui et sa générosité*

j'ai su ce que rêver et espérer

Ma belle mère *a qui je souhaite la santé et la langue vie*

Mes belles sœurs :

Farida et son mari *son adorable fils Kamel*

Nadjia et son mari *a qui je souhaite le bonheur et la santé*

Zahia, djidji et Fazia *A qui je souhaite un bon avenir*

Razika, Moumouh, Chabane *Que dieu les protège et les bénis*

Mes grands parents *a qui je souhaite la langue vie*

Mon oncle nouri et sa femme *mes oncles rezak, youcef et leurs familles*

Mes tentes et leurs familles *Ma tente saliha a qui je souhaite la belle vie*

Tous ceux que je connais *mes camarades et mes amis*

Mon binôme Hayet, mon amie *je lui espère la réussite et la belle vie*

Ouarda

Sommaire

Chapitre III : La recherche d`information sociale

Chapitre IV : Intégration des avis sociaux dans la RI

Conclusion générale

Annexe

Bibliographie

Table des figures

La recherche d'information (RI) était déjà présente dans la vie quotidienne avant l'avènement de l'informatique et les technologies de communication, elle était réalisée par l'utilisation de moyens empiriques (répertoires ou annuaires d'index manuels) ainsi que des annotations et résumés pour accéder aux documents contenant l'information recherchée.

L'élaboration de systèmes automatisés pour gérer l'explosion du volume d'informations capitalisées, produites par des sources distribuées, est devenue dans un tel contexte une nécessité. Sur le Web, où de grandes sources d'informations sont immédiatement disponibles, l'accès à l'information répondant au besoin d'un utilisateur est devenu de plus en plus difficile car il se trouve noyé au centre de la masse d'informations disponibles, surtout avec l'apparition des réseaux sociaux qui contiennent des méta-informations qui sont une valeur ajoutée au contenu du Web.

L'objectif principal des Systèmes de Recherche d'Information (SRI) est de répondre au besoin en information des utilisateurs qui interrogent, à travers une requête, une base de documents et le système leur renvoie une liste de documents susceptibles de répondre à leur besoin.

Notre travail se situe dans le cadre de la recherche d'information contextuelle, qui exploite les informations produites dans un contexte social tels que les commentaires sociaux, avec un objectif d'améliorer les résultats de la RI classique c'est-à-dire intégrer ces commentaires au SRI afin de mieux répondre aux requêtes des utilisateurs.

Nous tenterons de répondre, dans ce présent mémoire, à certaines questions posées dans le domaine de la RI:
« Comment intégrer les commentaires sociaux pour améliorer la réponse apportée à l'utilisateur? Quel type de commentaire allons-nous utiliser? Comment les modéliser ?»

Pour ce faire nous avons structuré notre mémoire comme suit :

Le chapitre I où nous débuterons par la définition et quelques types des réseaux sociaux, ensuite identifiants les commentaires, leurs types enfin les domaines d`utilisation de ces commentaires.

Le chapitre II aborde la recherche d`information social nous y présenterons le modèle de domaine et le modèle générique pour la recherche d`information social ainsi que la pondération des relations sociales.

Ensuite Le chapitre III est consacré à la recherche d'information classique et le processus de recherche d'information. On présente par la suite les différents modèles de la RI.

Le chapitre VI clôt ce mémoire où, on présente l'implémentation et les résultats de l'évaluation.

À la fin de ce document, on présentera un bilan sur l'ensemble de cette étude et on indiquera les perspectives envisagées pour étendre et améliorer notre application.

Introduction :

La croissance d'Internet a permis de former différents types de réseaux sociaux (RS) à grande échelle, qui sont maintenant reconnus comme un moyen important pour la diffusion de l'information. Nous allons voir dans ce chapitre les différentes notions concernant les réseaux sociaux et les différentes interactions des individus au sein de ces communautés virtuelles. Nous décrirons aussi certaines informations sociales produites pendant leurs activités.

I. Les réseaux sociaux :

I.1. Définitions Réseau social :

Un réseau social et un ensemble d'entités sociales (individus, organisations) reliées par des liens créés lors des interactions sociales. Les liens peuvent être de type très variés.
Un réseau social représente une structure sociale dynamique se modélisant par des sommets et des arêtes (structure de graphe qui peut être très complexe).

Deux aspects se côtoient quand on parle de réseaux sociaux : d'un côté, l'aspect sociologique et communautaire et de l'autre aspect technologique et Internet.
D'un point de vue sociologique, selon Wassermann et Faust **[WF94]**, un réseau social est un ensemble de relations entre des entités sociales (individus). Les contacts entre ces individus peuvent être, par exemple, des relations de collaboration, d'amitié, ou des citations bibliographiques. Ces ressources sont donc aussi bien formelles qu'informelles, matérielles qu'immatérielles. Toujours selon Wasserman et Faust, trois concepts sont également retenus dans cette analyse des réseaux sociaux :

1. Les acteurs et leurs actions sont considérés comme des entités indépendantes.
2. L'environnement des acteurs procure des opportunités et exerce des contraintes sur leurs actions individuelles.
3. Les structures sociales, politiques, économiques, etc. ont une influence sur les formes de relations entre les acteurs.

Nous avons ainsi tous les concepts sociologiques pour définir un réseau social : les individus, leurs liens (contacts), leurs affinités et l'environnement les entourant.

I.2. Typologie des réseaux sociaux

Les réseaux sociaux peuvent se diviser en plusieurs catégories selon le type d'interactions, l'objectif du site (enseignement, rencontres…), des centres d'intérêts et des informations diffusées (music, vidéos, photos). De ce fait, plusieurs classifications sont proposées par différents auteurs. Nous allons en citer certaines :

1. Tout d'abord, voici celle que l'on peut trouver sur **Wikipédia** , qui est la classification la plus simple :

- **Réseaux ouverts :**

La plupart des réseaux sociaux sont des réseaux ouverts (non privés) qui ne nécessitent pas de s'identifier pour y accéder.

- **Réseaux sur invitation :**

Il faut être invité par l'un de ses membres, cela permet au site de maitriser sa croissance et aux individus de connaitre au moins une personne.

- **Services en ligne de réseautage professionnels :**

Ils favorisent les rencontres professionnelles, les offres des postes d`emplois et la recherche de profils.

2. Voici la classification proposée par Pascal Faucompré **[1]**

- **Les Networking** : le plus utilisé dans les milieux professionnels. C`est le type de réseau car 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'adoulèssants en mal de reconnaissance.
- **Les spécialisés** : ils regroupent des communautés autour d'un thème bien précis.
- **Le micro-blogging** : chat public, summum du narcissisme, on y met tout ce qu'on y fait minute par minute, histoire de montrer aux autres qu'on est très actif.
- **Les fourre-tout** : ce sont les inclassables qui se servent du collaboratif ou du participatif pour alimenter leur service.
- **Les open-sources** : ou plutôt les plateformes qui permettront de créer le propre réseau social.

3. Esther Dyson **[TMA, 02]** de part sa définition des réseaux sociaux <les réseaux sociaux fournissent des outils qui facilitent le processus de mise en relation autour d'un centre d'intérêt commun et permettent la prise de contact en linge.>, Amène une classification des réseaux sociaux selon différents critères :

- **Les réseaux plate-forme de partage** :

Les plates-formes permettent de diffuser du contenu, souvent multimédia (vidéo et son), aux internautes. La mise en ligne et le partage de vidéo par exemple deviennent plus faciles car accessible par tous les internautes de la communauté. Exemple : Youtub, Dailymotion…

- **Les réseaux personnels et généralistes** :

Ils sont les plus souvent orientés vers un centre d'intérêt commun (lecture,…), avec le but de faire partager ses passions au reste de la communauté. On peut noter que les mises en relation sont rares dans ce type de réseaux. Exemple : MySpace, Skyblog, …

- **Les réseaux personnels et thématiques** :

Ils ont la plupart du temps le même fonctionnent que ceux généralistes, mais ils sont orientés autour d'une thématique : les voitures… exemple : Boompa, Eonscom…

- **Les réseaux professionnels** :

Les réseaux professionnels sont ceux les plus aboutis. Ils donnent la possibilité de mise en relation entre utilisateurs ainsi que le partage d'informations (information sur l'entreprise, coordonnées, CV…). Exemples : 6nergies, Viaduc, LinkedIn, OpenBC…

I.3. Présentation des principaux réseaux sociaux

I.3.1. Facebook :

À l'origine, Facebook est un réseau conçu pour un usage personnel. On y créé un compte (aussi appelé « profil ») afin de rester en contact avec des membres de son entourage.

Chaque membre y dispose d'une page personnelle pouvant être personnalisée de plusieurs manières : photo, texte de présentation, date d'anniversaire, profession, goûts musicaux, message s'affichant à côté du pseudonyme et reflétant l'humeur du moment etc…. Cette page personnelle est appelée Wall. (voir l'interface d'accueil sur la figure 6).

Toute personne inscrite à Facebook peut choisir de rendre son compte public (accessible à tous) ou privé (accessible à ses amis uniquement). Il est également possible de personnaliser plus avant les paramètres de confidentialité et de rendre accessible certains contenus à certaines personnes uniquement.

Tous les contenus diffusés peuvent être signalés comme intéressants via la fonction « J'aime » (Like), ou en rédigeant un commentaire dessus en choisissant l'option « commenter ». L'utilisateur peut aussi recommander ce contenu qui l'intéresse à un ami, en cliquant sur le lien « partager ».

On distingue 3 types de présence sur Facebook : les comptes, les groupes, et les pages (fanpages).

Figure 1 : page d'accueil du réseau sociale Facebook**.**

I.3.2 Twitter :

Twitter est une plateforme de microblogging sur laquelle les messages (appelés tweets) sont limités à 140 caractères (plus ou moins l'équivalent du statut de Facebook). Cette spécificité en fait un réseau social à part, beaucoup plus épuré que Facebook en termes de contenu. Ici, l'identification d'un membre se fait uniquement via la photo, le nom, la localisation et éventuellement un lien vers un site Web. La plupart du temps publics, les profils peuvent également être privés (ils nécessitent un accord préalable de leur créateur pour être suivis). A chaque réseau sa terminologie : un membre de Twitter n'a pas d'amis ou de fans, il est abonné à un certain nombre de comptes (il s'agit du nombre d'Abonnements ou Following), et un certain nombre de personnes sont abonnées à son compte (il s'agit du nombre d'abonnés ou Followers). Pour suivre une personne, la démarche est très simple : il suffit de disposer d'un compte sur Twitter et de cliquer sur l'icône Suivre (Follow) en haut de la page du compte que l'on souhaite suivre. La personne en question est alors immédiatement ajoutée à mon réseau. Contrairement à Facebook, cette procédure n'est pas réciproque. Ce n'est pas parce que on ajoute une personne au réseau sur Twitter qu`on fait partie du sien.

Lorsque je me connecte à Twitter, le site s'ouvre ma page d'accueil. Celle-ci répertorie tous les tweets diffusés par les membres de mon réseau par ordre anti-chronologique.

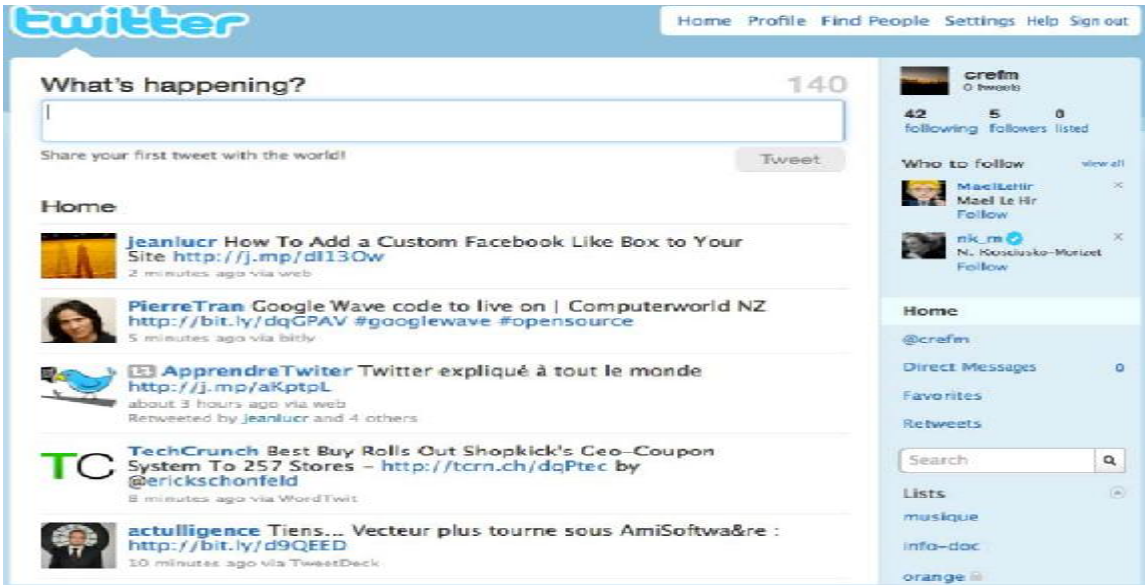


Figure 2 : page d'accueil du réseau sociale de twitter.

II. Les informations sociales

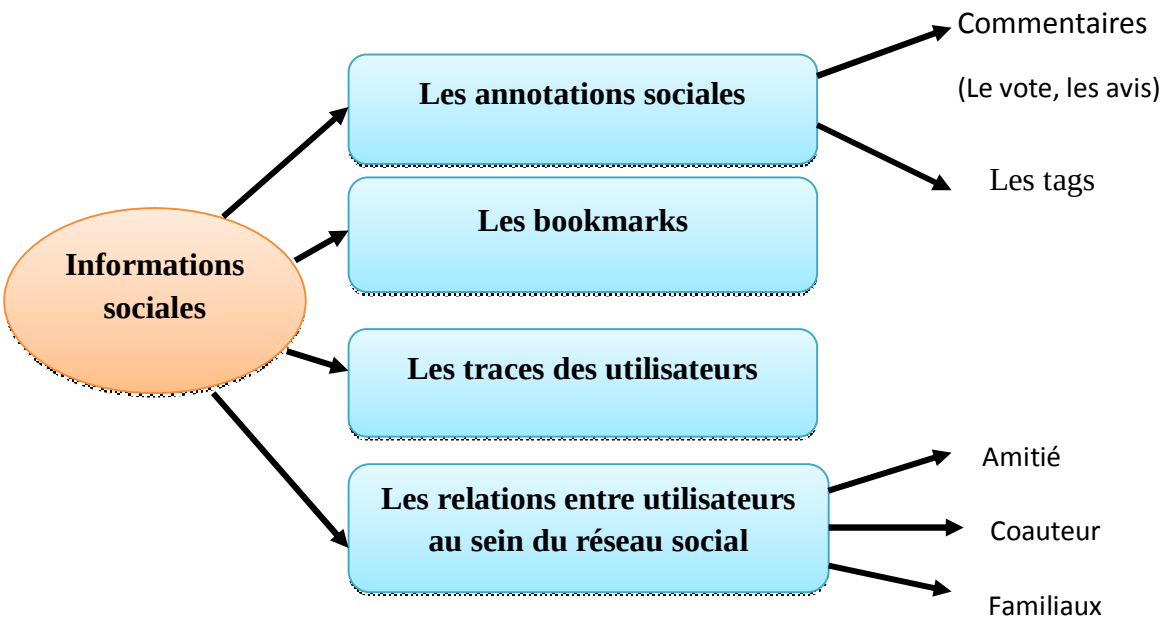


Figure 3 : Schéma représentant les informations sociales.

II.1. bookmarks :

Une solution pour alléger la surcharge d'information peut consister dans le développement par chaque utilisateur d'un système personnel d'information qui représente un sous-ensemble ciblé d'informations pertinentes pour lui. Le bookmark représente un outil simple pour construire ces sous-ensembles d'information personnalisés où des pages Web identifiées par des URLs utiles ou intéressantes peuvent être stockées afin de servir pour une utilisation ultérieure. Les utilisateurs gardent la trace des liens vers des pages en créant une liste des bookmarks : un espace personnel de stockage de l'information du Web **[ABC, 98].**

Pourquoi utiliser les bookmarks ?

la notion de bookmark a été adoptée pour permettre de marquer une page d'un livre, d'être capable de distinguer cette page parmi les autres afin de pouvoir la retrouver ultérieurement pour reprendre la lecture ou pour retrouver un passage ou une citation en particulier **[Szi,99]**.

Cette notion a été adoptée aussi dans nos jours avec un sens commun. Les bookmarks sont utilisés comme des « espaces d'information personnelle du web » pour aider les gens à se rappeler et à récupérer des pages Web.

Les bookmarks réduisent la surcharge physique et cognitive de gestion des URLs, en facilitant le stockage, la gestion et l'interprétation des liens (les utilisateurs ne doivent pas taper les longues adresses), en aidant la mémoire et en gardant l'historique.

Les limites qui s'imposent dans ce contexte ont trait à la difficulté de partager les ressources sauvegardées dans les navigateurs internet (favorits pour IE et bookmarks pour Mozilla) avec la communauté et avoir accès aux bookmarks sur n'importe quel ordinateur. Dans la section suivante nous présentons la notion de *social bookmarking* qui répond à ces deux limites.

II.2. les tags :

Selon Golder & Huberman **[GH,05]**, 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» **[HJS, 06]**. Elle est constituée par l'effort collectif des utilisateurs qui ont diverses connaissances et besoins quand ils interagissent avec un système de *social bookmarking* **[FKK, 09]**. De plus, le terme *folksonomy* est une combinaison de «folk» et «taxonomy», décrivant le phénomène de classification sociale **[Smi,04]**, 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.

II.3. commentaire :

Le mot COMMENTAIRE est un ensemble de mots (en langue naturelle) inséré dans une source pour expliquer ou apporter des informations, et non pour programmer.

On peut dire que le commentaire est une Explication, remarque sur un texte, un événement, une situation ou une Observation.

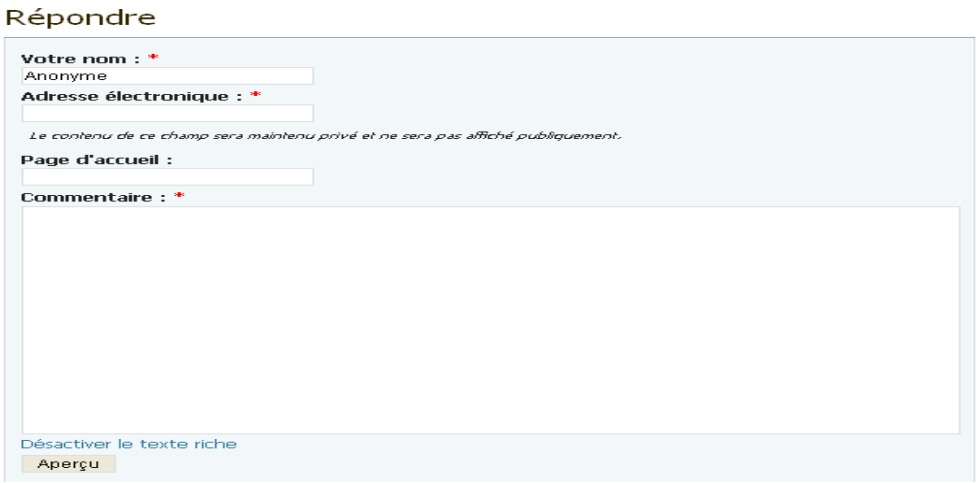
Le vote est un exemple d'application distribuée qui permet aux élections d'avoir lieu sur des réseaux informatiques ouverts **[SCH, 97]**. Dans cette application un ensemble de votants envoie leurs bulletins à travers le réseau à un centre de dépouillement virtuel responsable de la réception, validation, et classification des bulletins.

D'une manière générale, les participants impliqués dans une élection électronique sont un collectif d'électeurs et un ensemble d'autorités de vote. **[SCH, 97]**

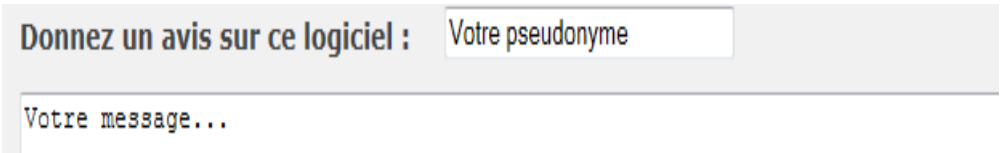
II.3.1. Les types des commentaires dans les réseaux sociaux :

- **Textuel** : texte brut.

On peut trouver des interfaces pour commenter en saisissons un texte pour donner un avis comme par exemple dans le E-commerce comme le montre l`image suivante :



Ou bien



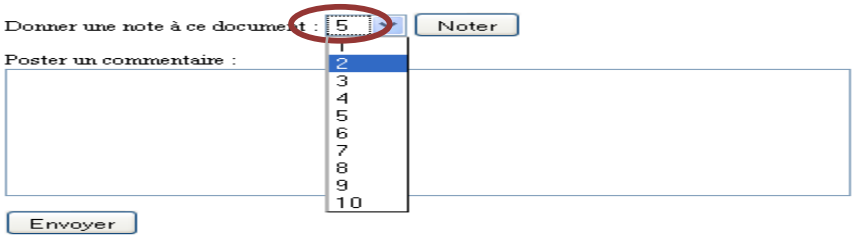
- **Numérique** : notes

C'est un autre type des commentaires qui sert à commenter ou donner un avis social par une note prise dans une échelle des valeurs, en utilise se type de commentaires dans l'évaluation des documents ou des fichiers pour voir l'importance de ces derniers par rapport aux utilisateurs.

Plus le nombre de notes et élevé plus la page a un grand intérêt (attire l`attention des visiteurs ou des lecteurs), si ce nombre n`est pas éleve cela peut vouloir dire que la thématique traitée ou le document lui-même (ou le site) n`est pas intéressant pour l`utilisateur.

La figure suivante montre un nombre de 4094 utilisateurs qui ont donnés la note 4/5

On peut aussi trouver ce type de commentaire sous cette forme (une liste de notes) :



Ou encore comme suit (des boutons radio) :

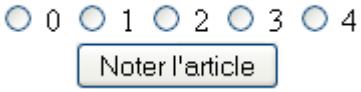


Image : ce sont des images statiques qui représentent le nombre de voix affecté à l`article. Par exemple dans la figure suivante le nombre de voix affecté est 0.

Ou bien

Plus le nombre de j`aime est élevé plus le document, le site ou le produit a une grande importance.

Ou comme suit :

Et aussi comme un accord sous la forme suivante :

Plus le nombre de LIKE ou le nombre de l`accord est élevé plus le document, le site ou le produit a un grand intérêt.

- **Pourcentage :**

Ou bien en le trouve sous la forme d'un pourcentage et vote

Le percentage est faible

Le percentage est èlevè

D`apres cette image plus le pourcentage est élevé plus le document, le site ou le produit gagne plus de voix.

- **Sonore :** commentaire laisser sur le site a fin d`exprimer l`appréciation ou la dèsappréciation.

On trouve d'autres types de commentaires sous les formes suivantes

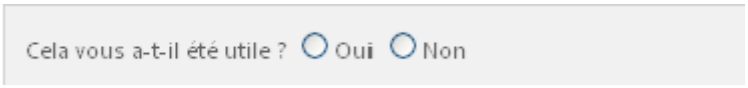
- **Un avis**

On trouve 4094 avis commentent le produit en utilisant les 4 étoiles c'est-à-dire valider la note 4/5, comme montré sur la figure suivante :

On trouve 2197 utilisateurs commentent le produit en utilisant les 5 étoiles c'est-à-dire valider la note 5/5, comme montré sur la figure si-dessous :

Une autre interprétation de ces étoiles, en terme de classement, voir l'illustration suivante :

- **L'utilité**



- **Le vote**

Conclusion :

Nous avons présenté dans ce chapitre les concepts généraux du réseau social. Et les différentes informations sociales en ns focalisant sur les commentaires, nous avons distingué plusieurs types de commentaires. Nous allons nous intéressé sur le type de commentaire image pour la recherche d`information ce qui fera l`objet du chapitre prochain.

« La vocation de tout système informatique documentaire est de raccourcir et de simplifier le chemin entre l'utilisateur et l'information : toute technologie qui répond à cet objectif doit être intégrée dans le processus de traitement de l'information. »
(**Vocation** : le temps consacré pour une affaire.) **[MAN, 85]**

Introduction

La Recherche d'Information (RI) s'intéresse à l'acquisition, l'organisation, la recherche et la sélection de l'information. La tâche principale d'un Système de Recherche d'Information (SRI) **[SM, 86]** est de sélectionner dans une collection de documents ceux qui sont susceptibles de répondre aux besoins en information de l'utilisateur. L'accès à ces documents peut s'effectuer de manière active à travers une requête, on parle alors de recherche active, ou bien de manière passive, en se basant sur le profil de l'utilisateur on parle alors d'un système de filtrage d'information **[TEB, 04] [TMA, 02]**

Ce chapitre a pour objectif de présenter les principaux concepts de la RI, les principaux modèles de recherche ainsi que les approches d'évaluation des SRI.

I. Définition d`un système de recherche d`information

Un système de recherche d`information est un système qui permet de retourner à partir d`un ensemble de documents ceux dont le contenu correspond le mieux a un besoin en information d`un utilisateur, exprimé à l`aide d`une requête.

Un système de recherche d`information 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`information.

II. Concepts de base de la recherche d'information

Les principaux éléments de la RI sont :

- Un utilisateur qui a un besoin en information à un moment donné.
- Un stock de données fixé au préalable (documents).
- Un système de recherche d'information(SRI) qui est l'interface entre l'utilisateur et le stock de données.

a) Le Document :

« Le document est l'élément centrale de SRI. C'est un objet complexe sans cesse en évolution car il est lié aux développements des technologies de la communication »

[sau, 05]. Un document peut être un texte, un morceau de texte, une page web, une image, une vidéo, etc. On peut appeler document toute unité qui peut constituer une réponse à un besoin informationnel de l'utilisateur. Nous nous intéressons uniquement, dans ce travail, aux documents textuels. Dans la suite de cette mémoire, nous utilisons indifféremment les termes document ou information pour designer l'unité documentaire retournée en réponse à la requête de l'utilisateur.

b) La requête de l'utilisateur :

Une requête est une formulation du besoin d'information d'un utilisateur. Elle peut être vue comme tant une description sommaire des documents ciblés par la recherche. Pour une recherche documentaire donnée, l'utilisateur doit soumettre une requête au moteur de recherche dans laquelle il spécifié les mots clés représentant son besoin en information.

c) Le système de recherche d'information (SRI) :

Il existe plusieurs définitions d'un système de recherche d'information. Citons par exemple :

« Un SRI est un ensemble de programmes informatique qui a pour but de sélectionner des informations pertinentes répondant à des besoins d'utilisateur, exprimés sous formes de requêtes » **[sau, 05]**

« Un SRI est un système informatique qui facilite l'accès à un ensemble de documents, pour permettre de trouver ceux dont le contenu correspond le mieux à un besoin d'information d'un utilisateur » **[Dah, 06]**.

La pertinence est un autre acteur de la RI, elle est défini comme suit :

d) La Pertinence :

La pertinence est la correspondance entre un document et une requête, une mesure de l'informativité du document à la requête, un degré de relation entre le document et la requête.

La notion de pertinence est le critère primaire pour l'évaluation des systèmes de recherche d'informations. Le processus de jugement de la pertinence de l'information est basé sur le degré de similitude de la représentation de la requête avec le contenu du document retrouvé par le système.

Il existe plusieurs « pertinences » possibles entre un document et un besoin, nous en citons

Les quatre (4) les plus importantes:

- **Pertinence algorithmique**: cette pertinence est traduite par une mesure algorithmique basée sur le calcul de la pertinence de l'information par rapport à la requête en utilisant les caractéristiques des requêtes d'une part et des documents d'autre part.
- **Pertinence thématique:** cette pertinence est traduite par la topicalité et est définie par le degré de couverture du document où se trouve le sujet de la requête.
- **Pertinence cognitive:** c'est la pertinence liée au thème de la requête, selon la perception ou les connaissances de l'utilisateur sur ce même thème, cette pertinence est caractérisée par une dynamique qui permet d'améliorer la connaissance de l'utilisateur via l'information renvoyée au cours de sa recherche.
- **Pertinence situationnelle:** cette pertinence est définie par l'utilisateur où l'usabilité de l'information est jugée relativement à la tâche de recherche et exprimée par le besoin en information selon la perception de l'utilisateur. C'est une pertinence potentiellement dépendante du contexte de recherche et est vue comme une pertinence dynamique.

  Il est à noter qu'un SRI idéal doit supporter un modèle de recherche d'information qui rapproche la pertinence algorithmique calculée par le système aux jugements de pertinence donnés par des vrais utilisateurs.

III. 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. Pour cela, le système de recherche regroupe un ensemble de méthodes et procédures permettant la gestion des collections de documents stockés sous forme d'une représentation intermédiaire permettant de refléter aussi fidèlement que possible leurs contenus sémantiques. L'interrogation de la collection de documents à l'aide d'une requête nécessite la représentation de cette dernière sous une forme unifiée et compatible avec celles des documents. Ces fonctionnalités sont représentées à travers le processus global de la RI, communément nommé processus en U et schématiquement représenté par la **figure I.1**.

v Avantages

- Fournir un ensemble plus grand de documents pertinents en cas de recherche.
- Réduction de la taille des index.
- Les relations entre les mots sont prises en considération, ce qui rapproche la capacité sémantique des systèmes informatiques de celle des humains ce qui améliore les performances.
- Cette approche fournit de nouvelles connaissances.

v Inconvénients

- approche coûteuse en calcul.
- Le temps de réponse de la recherche sémantique est supérieur à celui d'une recherche syntaxique.
- Complication lors de la mise à jour des index.

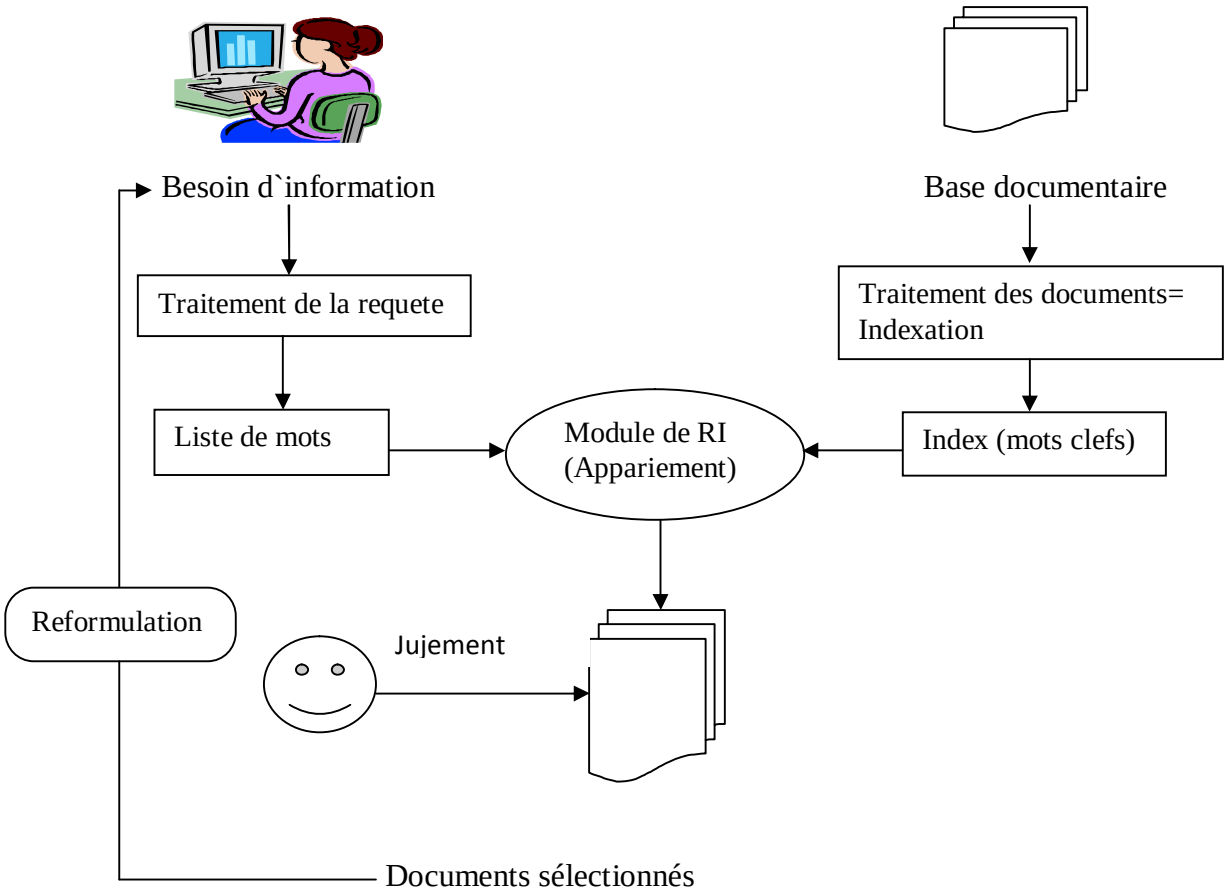


Figure 4: Processus en U de la RI

Ce processus consiste en deux principales phases: l'indexation et l'interrogation

III.1. L'indexation

Elle consiste à extraire et représenter le contenu des documents de manière interne sous forme d'index. Cette structure d'index permet de retrouver rapidement les documents contenant les mots clés de la requête.

Le processus d'indexation peut être manuel, automatique ou semi-automatique.

- **Indexation manuel** : chaque document est analysé par un spécialiste ou un documentaliste. Ce processus assure la pertinence dans les réponses puisqu'il permet de repérer d'une façon plus précise les mots clés décrivant un document; cependant, il est très coûteux en temps et en nombre de personnes et la subjectivité lie aux facteurs humains pose un problème de cohérence **[Sauv, 05].**
- **Indexation automatique :** chaque document est analysé à l'aide d'un processus entièrement automatisé. Cette méthode est la plus efficace pour les textes libres. Elle crée ce que nous appelons les index, qui relient les documents aux termes. Ces derniers sont de manière générale tirés des documents en utilisant des méthodes statistiques (notion du langage libre). Cette méthode est moins coûteuse en termes de temps, et la seule à pouvoir être utilisée dans l'environnement web ou dans les systèmes de recherche d'informations.
- **Indexation semi-automatique (mixte) :** c'est une combinaison des deux méthodes précédentes: un premier processus automatique permet d'extraire les termes du document.

  Cependant, le choix final reste au spécialiste du domaine ou au documentaliste pour établir les relations entre les mots clés et choisir les termes significatifs **[Hla, 07].**

L'indexation est un processus composé des étapes suivantes :

- **Lemmatisation :**

  La lemmatisation consiste à remplacer les différentes formes d'un mot par leur lemme (racine grammaticale). Car un mot peut avoir plusieurs formes dans un texte dont le sens est presque similaire, portes et du verbe portera qui possèdent la même racine port. Pour résoudre ce problème, on opère plutôt un traitement de lemmatisation qui consiste à ramener tous les mots à leur forme de base ou lemme.

Il y a plusieurs façons de lemmatiser des mots :

- Une première façon consiste à examiner seulement la forme du mot pour en déduire sa forme de base.
- Une autre approche consiste à utiliser un dictionnaire pour la lemmatisation. Son utilisation ajoute certains avantages, mais au prix de disposer d'un dictionnaire électronique qui est encore peu accessible.

Donc cette phase n'est pas obligatoire car dans certains cas ce passage supprime la sémantique des termes, par exemples le verbe « porter » et le mot « porte » seront indexés par le même radical « porte », ce qui augmentera le nombre de document non pertinents sélectionnés par le SRI **[ARK ,08].**

- **Simplification du texte :**

Cette étape consiste à supprimer dans un premier temps des mots de fortes fréquences, généralement en référence à une liste de mots appelée anti-lexiques ou plus fréquemment stoplist.

- **Pondération des termes**

Cette étape est généralement basée sur des formules de pondération qui affecte à chaque terme un degré d'importance (une valeur de discrimination) dans le document ou il apparait **[Hla, 07]**.

Il existe un grand nombre de formules de pondération :

- Fonction brute de tf_{ij} (term frequency) : correspond au nombre d'occurrences du terme T_i dans le document D_j.
- Fonction binaire : elle vaut 1 si la fréquence d'occurrence du terme dans le document est supérieure ou égale à 1, et 0 sinon.
- Fonction logarithmique : combine tf_{ij} avec un logarithme, elle est donnée par : $\alpha + \log(tfij)$, où α est une constante. Cette fonction a pour but d'atténuer les effets de larges différences entre les fréquences d'occurrence des termes dans le document.
- Fonction normalisée : permet de réduire les différences entre les valeurs associées aux termes du document. Elle est donnée par la formule suivante :

$$0.5 + 0.5 * \left(Tf_{ij} / \text{Max}\left(Tf_{ij}\right) \right)$$

Formule 1 : Fonction normalisée

Où Max (Tf_{ij}) est la plus grande valeur de Tf_{ij} des termes du document D_j. Dont la plus connue est basée sur deux facteurs: fréquence de terme (TF) et fréquence inverse de document (IDF), définies dans ce qui suit:

- **Fréquence des termes (TF)**

La fréquence du terme (term fréquence) est simplement le nombre d'occurrences de ce terme dans le document considéré. L'idée sous-jacente est que plus un terme est fréquent dans ce document, plus il est important dans la description de celui-ci. Soit le document *d* et le terme *t*, alors la fréquence TF du terme dans le document est souvent utilisée directement ou exprimée selon l'une des formules suivantes :

$]=$

$Tf = log\ (f(t,d))$
$Tf = log\ (f(t,d) + 1)$
$Tf = f(t,d) / max\ (f(t,d))$

Formule 2 : fréquence de terme (TF) **[Dah, 06]**

- **fréquence inverse du document (IDF)**

La fréquence inverse du document (inverse document frequency) est une mesure de l'importance du terme dans l'ensemble du corpus. Elle consiste à calculer le logarithme de l'inverse de la proportion de documents du corpus qui contiennent le terme. Cette mesure est exprimée selon l'une des déclinaisons suivantes:

$Idf = log\ \|N\|/n$
$Idf = log\ \|N-n\|/n$

Formule 3 : fréquence inverse de document (IDF) **[Dah, 06]**

Où n est la proportion des documents contenant le terme et N le nombre total de documents dans collection.

La fonction de pondération de la forme TF-IDF consiste à multiplier les deux mesures TF et IDF comme suit:

$Tf * idf = log\ (1+tf) * log\|N\|/n$

Formule 4 : fonction de pondération de la forme TF*IDF **[Dah, 06]**

III.2. Interrogation

L'interrogation est la deuxième phase de processus l'interaction d'un utilisateur final avec le SRI, une fois les documents sont représentés sous forme d'index. Suit à une requête utilisateur, le système calcule la pertinence de chaque document vis à vis de la requête utilisateur selon une mesure de correspondance du modèle de RI, et retourne la liste des résultats à l'utilisateur.

- **L'appariement document-requête**

Ce processus permet de mesurer la pertinence d'un document vis-à-vis d'une requête. De manière générale, à chaque réception d'une requête, le système crée une représentation de la requête qui soit similaire à celle des documents, puis calcule un score de correspondance entre la représentation de chaque document et celle de la requête. Le processus d'appariement est étroitement lié au processus d'indexation et de pondération des termes. Il existe deux méthodes d'appariement:

- **Appariement exact** « exact match retrieval »: Le résultat est une liste de documents respectant exactement la requête spécifiée avec des critères précis. Les documents retournés ne sont pas triés.
- **Appariement approché** « best match retrieval » : Le résultat est une liste de documents censés être pertinents pour la requête. Les documents retournés sont triés selon leurs scores de pertinence vis-à-vis de la requête.

- **La reformulation de la requête**

La requête initiale est vue en RI comme un moyen permettant d'initialiser le processus de sélection d'informations pertinentes. A ce titre, les SRI doivent intégrer des fonctionnalités permettant de prendre le relai. Ce relai est souvent effectué par le processus de reformulation de requêtes. Ce processus permet en fait de construire une nouvelle requête en se basant sur des informations/connaissances extraites des documents ou disponibles dans des ressources spécifiques. La reformulation rentre dans un processus plus général d'optimisation de la fonction de pertinence qui a pour but de rapprocher la pertinence système de la pertinence utilisateur **[Bou, 00].** Le principe de la reformulation de requêtes consiste à modifier la requête de l'utilisateur en rajoutant des termes significatifs et/ou en ré-estimant leurs poids associés. Ces termes peuvent provenir:

- De documents jugés par l'utilisateur. On parle alors dans ce cas de la réinjection de pertinence, communément appelée *Relevance feedback* ou *Retour de pertinence*.

- Des sources construites manuellement (thésaurus), ou automatiquement à partir des documents de la collection. Dans le cas où ces ressources sont construites automatiquement, elles sont souvent représentées sous forme de termes reliés entre eux. Ces liens sont mesurés en se basant sur la cooccurrence entre termes et sont construits, soit à partir des documents retrouvés par le système.

IV. Les principaux modèles de RI

Un modèle de recherche d'informations est formellement décrit par un quadruplet **[D, Q, F, R**

(q_i ,d_j)] où :

D : ensemble des représentants des documents de la collection,

Q : ensemble des représentants des besoins en informations,

F : schéma du support de représentation des documents, requêtes et relations associées.

R (q_i ,d_j) : fonction d'ordre associée à la pertinence.

Nous présentons ici les modèles les plus couramment utilisés pour la RI, notamment le modèle booléen, le modèle vectoriel et le modèle probabiliste. Les différents modèles de RI

IV.1. Le modèle booléen

C'est le premier modèle utilisé en RI **[Sal 71]** ; il est basé sur la théorie des ensembles et l'algèbre de Boole [Gessler 93]. Le modèle booléen propose la représentation d'une requête sous forme d'une équation logique. Les termes d'indexation sont reliés par des connecteurs logiques *ET*, *OU* et *NON*.

L'approche booléenne consiste à trouver les documents qui ont *exactement* les mêmes termes qu'une requête construite par mots clefs. Les requêtes peuvent être affinées grâce aux opérateurs *OR* ou *AND* ou encore au moyen d'opérateurs comme *NEAR*. Ce type de recherche est à la base des moteurs d recherche comme Altavista ou Google.

Cette approche est très efficace pour des requêtes utilisant des termes très spécifiques ou portants sur des domaines techniques particuliers avec leur vocabulaire propre mais son intérêt reste néanmoins limité.

L'inconvénient majeur du modèle booléen réside dans sa caractéristique de fournir une réponse binaire (les documents contiennent les termes demandés ou ne les contiennent pas). Ceci induit un volume de réponses important sans ordre spécifique des documents résultants.

v Les avantages :

- Le modèle booléen est très simple à mettre en œuvre.
- la raison pour laquelle un document est sélectionné par le système est claire.
- le modèle booléen répond exactement à la requête.

v Les inconvénients :

- Le système retourne un ensemble de documents non ordonnés comme réponse à une requête. Cependant, il n'est pas possible de dire quel document est plus pertinent qu'un autre.
- Les termes dans une requête ou dans un document sont pondérés de la même façon (simple 0 ou 1), il est ainsi difficile de distinguer les termes les plus importants.
- Les formules de requêtes sont complexes, non accessibles à un large public. Elles nécessitent une maîtrise parfaite des opérateurs booléens, car leur signification est déférente de celle qu'il est dans la langue naturelle.
- la sélection d'un document est basée sur une décision binaire qui est peu efficace.

Les deux modèles présentés ci-dessous largement utilisés en pratique, permettent de remédier à ces inconvénients.

IV.2. Modèle vectoriel (Vector Space Model) :

Suivant la proposition faite part **[Luh, 57],** le modèle vectoriel a été développé par **[Sal, 71] [Sal, 83]** dans le projet SMART (Salton's Magical Automatic Retriever of Text). Ce modèle repose sur les bases mathématiques des espaces vectoriels. Les requêtes et les documents sont représentés dans l'espace vectoriel engendré par les termes d'indexation. Dans ce modèle, le degré de pertinence d'un document vis-à-vis de la requête est proportionnel à la position des deux vecteurs dans l'espace. Elle est évaluée à l'aide d'un degré de corrélation entre les vecteurs associés. Le coefficient de similarité **RSV** (Retrieval Status Value) où Q est une requête et D un document de la collection, est calculé sur la base d'une fonction qui mesure la colinéarité des vecteurs documents et requête.

Chaque document est représenté par un vecteur : $Dj= (d_{1j} ,d_{2j},\ldots\ldots\ldots,d_{Tj})$

Chaque requête est représentée par un vecteur : $Q=(q_1,q_2,\ldots\ldots\ldots\ldots q_T)$

Avec

$\mathbf{d_{ij}}$:poids du terme t_i dans le documents D_j ,

$\mathbf{q_i}$:poids du terme t_i dans la requête Q.

On peut citer notamment les fonctions suivantes :

Produit scalaire : RSV(Q,D_j) = $\sum_{I=1}^{T} q_i * d_{ij}$

Mesure de Jaccard : RSV(Q,D_j) = $\dfrac{\sum_{I=1}^{T} q_i * d_{ij}}{\sum_{i=1}^{T} q_i^2 + \sum_{i=1}^{T} d_{ij}^2 - \sum_{i=1}^{T} q_i * d_{ij}}$

Mesure de cosinus : RSV(Q,D_j) = $\dfrac{\sum_{I=1}^{T} q_i * d_{ij}}{(\sum_{i=1}^{T} q_i^2)^{1/2} * (\sum_{i=1}^{T} d_{ij}^2)^{1/2}}$

- **Les avantages :**

- Le langage de requête est plus simple (liste de mots clés).
- Les performances sont meilleures grâce à la pondération des termes.
- Permet effectivement de trier les documents répondant à une requête.
- La mesure de similarité permet d'ordonner les documents selon leurs pertinences vis-à-vis de la requête.

- **Les inconvénients :**

- L'inconvénient principal de ce modèle lié à l'indépendance mutuelle des termes d'indexation (il ne prend pas en considération les relations entre les termes de l'indexation).

IV.3. Modèle probabiliste (Probabilistic Model)

Le modèle probabiliste a été proposé par Robertson et Sparck Jones il utilise un modèle mathématique fondé sur la théorie de la probabilité conditionnelle (appelé aussi modèle de la théorie de pertinence). Lors du processus d'indexation deux probabilités conditionnelles sont utilisées :

P(t/Pert) : La probabilité pour que le terme *t* apparaisse dans un document donné sachant que ce document est pertinent pour la requête.

P(t/NonPert) : La probabilité pour que le terme *t* apparaisse dans un document donné sachant que ce document est non pertinent pour la requête.

En supposant que la distribution des termes dans les documents pertinents est la même que leur distribution par rapport à la totalité des documents, et que les variables "documents pertinents" et "document non pertinent" sont indépendantes, la fonction de recherche est obtenue en calculant la probabilité de pertinence d'un document **P (Pert/D)**

$$\mathbf{P(Pert/D)} = \frac{P(D/Pert)*P(Pert)}{P(D)}$$

$$\mathbf{P(NonPert/D)} = \frac{P(D/NonPert)*PNon(Pert)}{P(D)}$$

P(D)=P(D/Pert)*P(pert)+P(D/NonPert)*P(NonPert)

Où :

P(D/Pert) (respectivement P(D/NonPert)) : Probabilité d'observer D sachant qu'il est pertinent (respectivement non pertinent).

P(Pert) (respectivement P(NonPert)) : Probabilité à priori qu'un document soit pertinent (respectivement non pertinent).

Le coefficient de similarité requête document (RSV) peut être calculé par différentes formules. Robertson et Spark-Jones [Robertson 96] proposent la formule suivante :

$$\mathbf{RSV}=\sum \log \left[\frac{(r+0,5)\ /(R-r+0,5)}{(n-r+0,5)/(N-n-R+r+0,5)} \right]$$

Où

N: nombre total de documents de la base,

n: nombre de documents contenant le terme,

R: nombre de documents connus comme étant pertinents,

r: nombre de documents connus comme étant pertinents et contenants le terme.

L'ajout de 0.5 à tous les membres s'explique par la nécessité d'écarter tous les cas limites qui entraîneraient des valeurs nulles de ces membres.

Ce modèle a donné lieu à de nombreuses extensions. Il est à l'origine du système OKAPI qui est l'un des systèmes les plus performants selon les compagnes d'évaluation TREC[3].

L'inconvénient majeur de ce modèle est que les calculs des probabilités sont complexes et que l'indépendance des variables n'est pas toujours vérifiée voir pas prise en compte.

- **Les avantages :**

- Les modèles probabilistes est largement répandu dans le domaine de la RI car Ils ont une base théorique saine et sont indépendants du domaine d'application.

- **Les inconvénients :**

- Un obstacle majeur avec les modèles de recherche d'information probabilistes est de trouver des méthodes pour estimer les probabilités utilisées pour évaluer la pertinence qui soient théoriquement fondées et efficaces au calcul**.**

Conclusion :

Nous avons présenté dans ce chapitre les concepts généraux de la recherche d`information avec une description de processus de recherche et les différents modèles permettant de mesurer la pertinence requêtes documents.

Introduction :

L'explosion des réseaux sociaux a permis l'émergence d'une nouvelle branche de la Recherche d'Information: la recherche d`informations sociale, 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.

I. Définition de la RI :

La recherche d'information selon **[Sal, 70]** « est l'ensemble des techniques permettant de gérer des textes ».

D'après **[Urf, 04],** « La recherche d'information (RI) est un ensemble de méthodes et de procédures ayant pour objectif d'extraire d'un ensemble de document, les informations voulues. Dans un sens plus large, la RI est toute opération(ou ensemble d'opérations) ayant pour objectif la recherche, la collecte et l'exploitation d'informations en réponse à une question sur un sujet précis ».

II. Définition de la RI social :

La recherche d'information sociale est définie comme l'incorporation d'information concernant les réseaux sociaux et les rapports dans le processus de la recherche d'information:

La recherche d'information social = réseaux sociaux + recherche d'information

La recherche d'information social a pour but: d'intégrer les concept de réseau social dans les processus de la recherche d'information avec exploration de réseau social pour accéder à l'information pertinente et estimer la pertinence d'un document à partir de son contexte social et Intégrer deux domaines la recherche d'information qui représente et compare les documents et les requetes et l'analyse des réseau sociaux qui représente les entités sociales et estime la centralité d'un individu.

III. De la RI classique vers la RI social :

- ***RI classique / RI sociale***

Cette section présente, par rapport au modèle de RI classique, les principales informations sociales apportées par les utilisateurs et pouvant être intégrées dans la RI sociale.

- ***Modèle de RI classique***

La RI traite différents aspects, notamment la représentation, le stockage, l'organisation et l'accès à l'information. Un système de RI se base sur une comparaison entre la représentation interne de la requête et la représentation interne des documents du corpus, comme montre la **figure 1**.

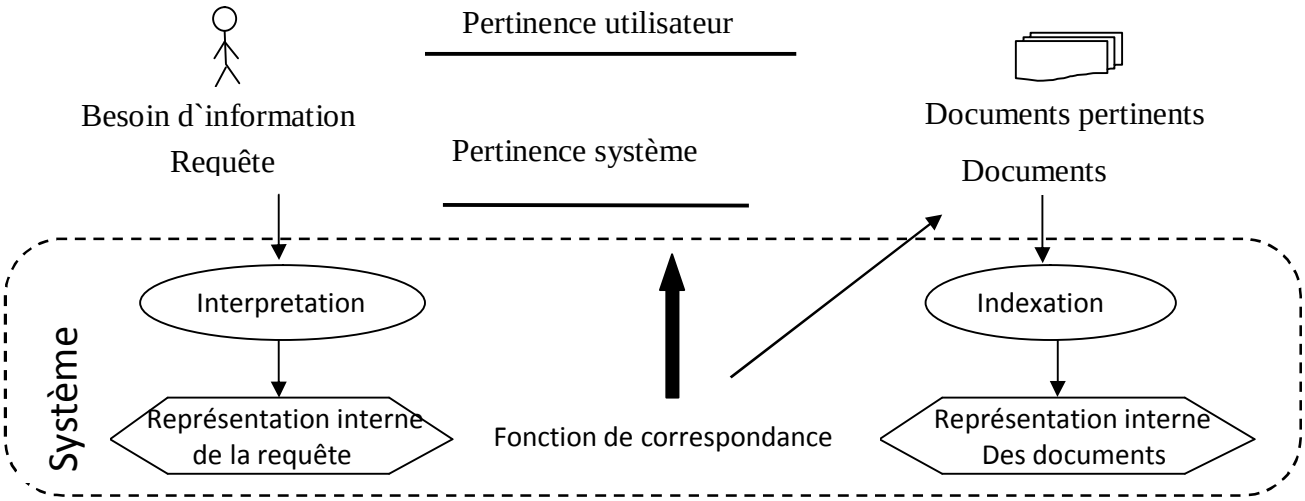


Figure 5 : Modèle de RI classique

- ***Modèle de RI social***

Les utilisateurs du Web produisent divers contenus, créent des annotations, manipulent des documents sur le Web et laissent des traces de leurs passages, etc. Par exemple, les *folksonomies*[1] 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.

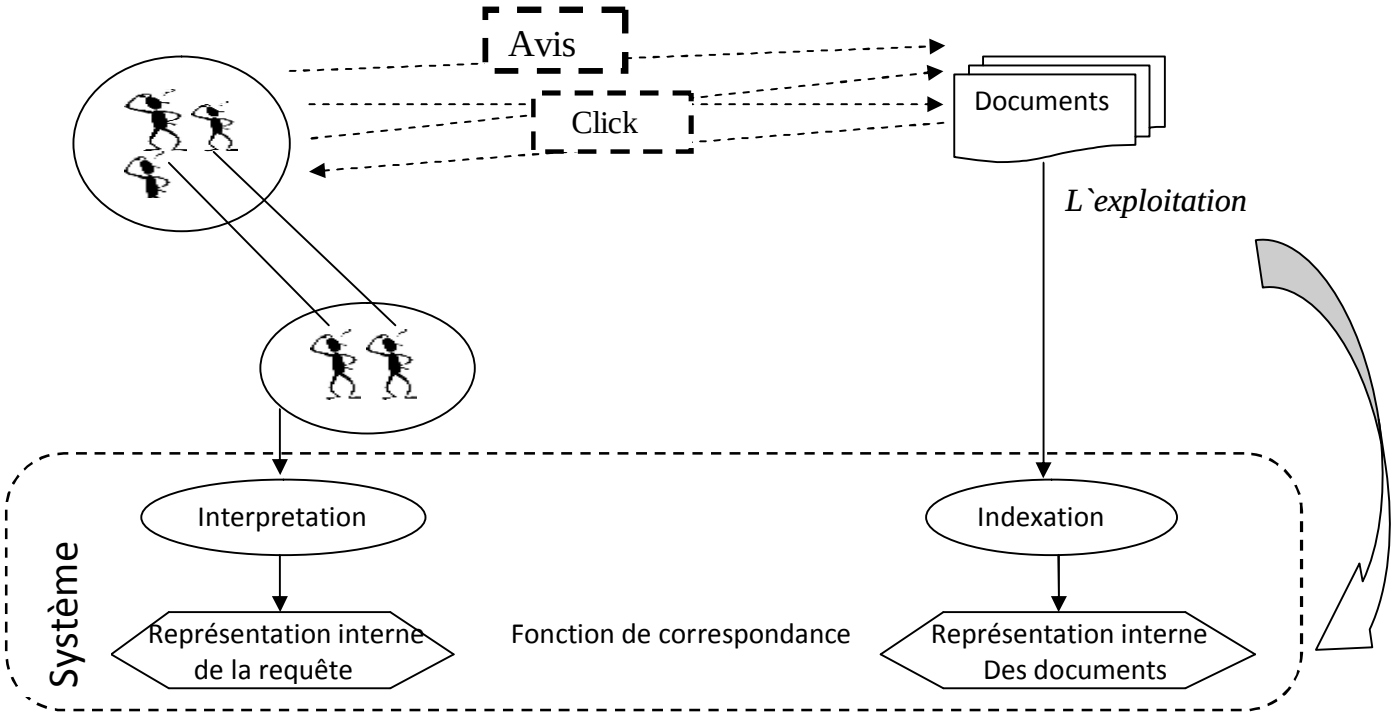


Figure 6 : Modèle de RI sociale : l'utilisateur producteur et consommateur d'informations

IV. Modèle de domaine pour RI social :

Les modèles traditionnels pour la recherche documentaire se concernent par des documents, questions, et leurs relations entre eux: Un document est approprié à une question, un document met en référence d'autres documents, une question est semblable à autre questions. De même, l'analyse de réseau sociale modèle des individus et leurs relations de l'un à l'autre. Les systèmes de recherche documentaire traditionnels ne modélisent pas des individus, ni leur rôle comme utilisateurs du système, ni comme auteurs du recherche des documents, et les réseaux sociaux n'incorporent pas le contenu recouvrable.

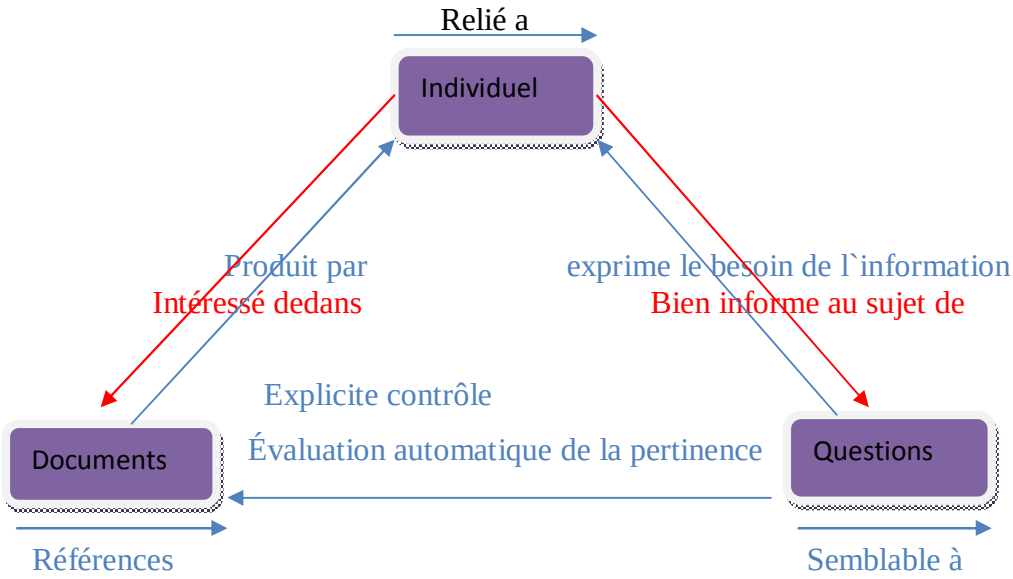


Figure 7 : Un modèle de domaine pour la recherche documentaire sociale

Un système de RI social est caractérisé par la présence de chacun des trois types d'entités: documents, questions, et individus. La plupart des systèmes emploieront seulement un sous-ensemble des associations possibles entre les entités, selon le domaine du système. Modeler les relations entre les individus est obligatoire pour un système de RI social ; tous autres types d'associations sont facultatifs, aussi long que chacune des trois entités ayant une association avec au moins un autre.

V. Le modèle générique de recherche d'information sociale :

Un modèle de recherche d'information offre un support théorique pour représenter des documents et des requêtes et mesure leur degré de similitude assimilée à la pertinence. Formellement, et en se basant sur la représentation proposée par Baeza-Yates et Ribeiro-Neto (1999), nous décrivons le modèle générique de recherche d'information sociale par un quin-tuplet [D,Q,G, F,R(qi, dj ,G)] où D est l'ensemble des documents, Q est l'ensemble des requêtes, G est le réseau d'information sociale, F est la fonction d'appariement des documents et des requêtes et R (qi, dj , G) est une fonction de classement qui intègre divers facteurs de la pertinence sociale et qui tient compte de la topologie du réseau. Cette fonction peut être définie par la combinaison de sous-ensemble des facteurs de la pertinence sociale suivants : la pertinence thématique, l'importance sociale des acteurs, la distance sociale, la popularité, la fraîcheur de l'information et le nombre de marque-pages reçus Amer-Yahia et al. (2007).

En ce qui concerne le réseau d'information sociale G, il représente les entités sociales qui interagissent au voisinage du document. Nous proposons donc d'y inclure tous les acteurs et les données qui permettent d'évaluer sa pertinence sociale comme illustré dans la figure 1. Les acteurs y représentent les producteurs et les consommateurs d'information (respectivement les Modèle de RIS pour l'Accès aux Ressources Bibliographique sauteurs et les utilisateurs) tandis que les données comprennent les documents et les annotations sociales (les tags, les votes, et les avis). Dans le cadre de leurs collaborations et interactions sociales, les acteurs participent à produire de l'information et à enrichir les documents par les annotations sociales.

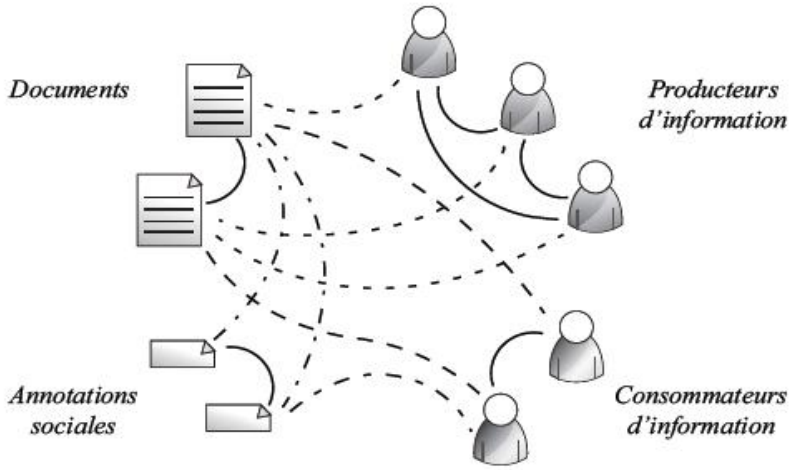


Figure 8 : Le réseau d'information sociale.

Le réseau d'information sociale peut être représenté par un graphe G = (V,E) où l'ensemble des nœuds V = A □ D □T représente les entités sociales avec A, U, D et T correspondent respectivement aux auteurs, aux utilisateurs, aux documents et aux annotations sociales. L'ensemble des arcs E inclus ou égal (V × V) représente les relations sociales reliant les différents types des nœuds (publier, co-auteur, amitié, citation, annotation...etc.).

Le réseau d'information sociale est appréhendé différemment. Du point de vue du producteur d'information, le réseau social regroupe les nœuds documents et auteurs et met en évidence le contexte social de la production des ressources. De même, la vue consommateurs d'information représente le contexte d'utilisation sociale des documents et l'interaction entre les utilisateurs. Cette vue intègre 3 types des nœuds à savoir : les documents, les annotations sociales et les utilisateurs.

Dans la suite, nous instancions ce modèle générique au cadre de la production et de la consommation de ressources bibliographiques.

VI. Pondération des relations sociales:

Les arcs reliant les nœuds sociaux expriment divers types de relations sociales et permettent d'optimiser d'une façon significative le processus d'exploration du réseau social. Si nous explorons le voisinage social d'un nœud, les poids sur les arcs nous permettent de sélectionner le nœud suivant à chaque saut. Dans cet article, nous nous intéressons essentiellement au réseau social des publications scientifiques, pour cela nous définissons un modèle de pondération pour les associations auteur-auteur e (ai, aj) 2 (A × A) et les associations auteur document e (ai, dj) 2 (A × D).

V.1. relation de coauteur :

Représentée par un arc dirigé, cette relation connecte deux coauteurs ayant collaboré pour produire un document. Les coauteurs ont souvent des contacts personnels directs néanmoins la multiplicité de leurs collaborations exprime la similarité et le partage d'intérêt entre eux. En effet, les auteurs des publications scientifiques ont tendance à s'échanger les connaissances et diversifier leurs collaborations. Pour cette raison et afin de quantifier la similarité entre les coauteurs, nous proposons de tenir compte de la totalité des collaborations. Nous proposons d'assigner des poids asymétriques aux relations de coauteur comme suit :

$$\boldsymbol{Co(i,j) = \frac{A(i,j)}{A(i)}} \quad \textbf{.....1}$$

Avec A (i, j) est le nombre de documents co-écrits par les auteurs ai et aj. A(i) représente le nombre des documents publiés par l'auteur ai.

V.2. La relation de citation :

Représentée par un arc dirigé, les liens de citation expriment le transfert de connaissances entres les auteurs des publications scientifiques. Par conséquent, plus un auteur cite les publications d'un second auteur, plus il est influencé par ses idées d'une manière que tous les deux partagent des sujets similaires. Cette relation est asymétrique et son importance est souvent proportionnelle au nombre des publications. Afin de mesurer l'importance dès cette association, nous tenons compte du nombre des citations entre les auteurs ainsi que le nombre total des citations énoncées par l'auteur source de la relation. Les relations de citation sont alors pondérées comme suit :

$$\boldsymbol{Ci(i,j) = \frac{C(i,j)}{C(j)}} \text{.....2}$$

Avec C (i, j) est le nombre de fois que l'auteur ai a cité l'auteur aj et C(i) représente le nombre de citations énoncées par l'auteur ai.

V.3. La relation de publication :

Un auteur sera plus affilié à un sujet S s'il l'a fréquemment abordé dans ses publications. Ainsi, un coauteur sera davantage associé à son document d que ses coauteurs s'il a publié plusieurs documents sur le même sujet de d. Pour estimer les connaissances d'un coauteur ak sur le sujet de son document d, nous proposons de comparer la quantité d'information importée via ses autres publications. Du point de vue des consommateurs, cela peut être estimé par la distribution des tags affectés au sous-ensemble des publications de chaque coauteur représenté par Ak avec ak 2 A est le coauteur que nous souhaitons mesurer l'affiliation au sujet de son document d. Nous calculons ainsi une distribution de probabilité de l'ensemble des *tags* T assignés au document d dans la sous-collection des documents publiés par les coauteurs du document d.

$$\mathbf{w(ak, d)} = \sum_{\boldsymbol{n=1}}^{\infty} \left(\frac{\mathbf{tf(ti,Ak)}}{\mathbf{tf(ti,A)}} \right) \text{.....3}$$

Avec T l'ensemble des tags assignés au document d. A = Smk=1 Ak représente la sous-collection des documents publiée par les m coauteurs du document d. tf(ti,Ak) est la fréquence de *tag* ti dans le sous-ensemble des documents Ak publiés par l'auteur ak. tf(ti,A) représente la fréquence de *tag* dans la sous-collection des documents publiés par les coauteurs du document d.

Certains algorithmes de centralité sociale ne supportent pas la multiplicité des arcs de même sens entre deux nœuds. Nous proposons donc de combiner les poids des relations de coauteur et de citation comme suit :

w(ai, aj) =$\frac{1}{4}$ (1 + Co(i, j)) _ (1 + Ci(i, j))…….4

Conclusion:

Les différentes approches de RI classique ignorent l'influence des relations sociales et des interactions de l'utilisateur au sein de son contenu social, sur le processus global de RI. Les travaux effectués dans la RI sociale montrent l'intérêt d'exploiter les informations sociales pour la RI. Nous avons vu dans le présent document deux catégories d'approches suivies pour les différents travaux d'état de l'art en RI sociale. La première catégorie consiste à exploiter les informations sociales relatives au contenu (annotations, traces, etc.). La deuxième catégorie permet de combiner ce contenu social avec les relations entre les utilisateurs des RS.

Ces travaux ouvrent de nombreuses perspectives. En particulier, nous pensons que cette combinaison ne peut être satisfaisante que si elle intervient au sein même du modèle de RI, ce qui nécessite une adaptation des composants de ce modèle dans un cadre social.

Un problème important réside dans le fait qu'il n'existe pas de collection de test standardisée pour la RI sociale. En effet, dans les travaux présentés, plusieurs jeux de données sont utilisés pour évaluer les approches (Del.icio.us, Citeulike.org, last.fm, etc.) mais aucun d'entre eux n'utilise une collection de test standardisée. La construction d'une telle collection est un problème fondamental de la RI sociale. Il existe différentes collections de tests basés sur les données du Web, par exemple *TREC Blog*. Cependant, ces collections ne tiennent pas compte des spécificités des RS, en particulier l'existence de relations explicites entre les utilisateurs, ou encore les communautés qui se forment entre eux.

Notons qu'une tâche "*Social Networks Search*" est proposée dans le cadre de la nouvelle piste *MicroBlog*de l'édition 2011 de TREC2, basée sur des données issues de Twitter.

Introduction :

Apres avoir présenté dans les chapitres précédant, les différente concepteurs nécessaires a l`accomplissement de notre travail, nous passons a la partie principale qui est l`implémentation.

Dans ce chapitre nous nous intéressant a la description de l`architecture d`ensemble du système, elle consiste en le développement d'une application basée sur les commentaires sociaux dans le processus de la recherche d'information.

I. L` objectifs de notre système

L'objectif principal de notre travail est d'améliorer la Recherche d'Information classique, ou en intégrant les commentaires sociaux.

Les documents commentés permettent aux utilisateurs de retrouver rapidement et d'extraire dans une base documentaire des documents qui répondent mieux à leurs besoins.

II. Description de notre approche

La solution proposée pour améliorer les performances du système de recherche d'information classique, est une approche qui permet en premier lieu d'appliquer un modèle de recherche basé sur le calcul du score classique des documents à base de la BM25. En second lieu, on évalue un score social qui est basé sur les documents commentés et finir par combiner ces deux résultats le score classique et le score social suivant cette formule :

$$\text{Score final} = \alpha\ (\text{score classique}) + \beta\ (\text{score social}) \ldots\ldots \text{Formule 1}$$

Telle que :

α : est un paramètre défini par apprentissage, on le fixe pour notre étude à 0,7

β = 1-α, **β**=0,3

Score Classique : est le score classique qui représente le modèle de recherche BM25.

Score Social : est le score social c`est le nombre de voix pour chaque document devisé par le nombre total de voix de toute la collection des documents.

Notre système est composé des modules suivants :

- **Le module de requête :** qui sert à représenter la requête utilisateur.
- **Le module de document :** qui sert à indexer les documents.
- **Le module des avis sociaux :** il s'agit d'extraire les avis sociaux qui leurs sont assignés afin de construire la base des avis.
- **Le module de recherche :** qui consiste à confronter les modèles de document, des avis avec le modèle de la requête en calculant des scores intermédiaires (classique et social) qui vont servir à évaluer le score final(le score de similarité global).

Notre contribution est au niveau des deux modules d'avis et de recherche.

III. Architecture du système :

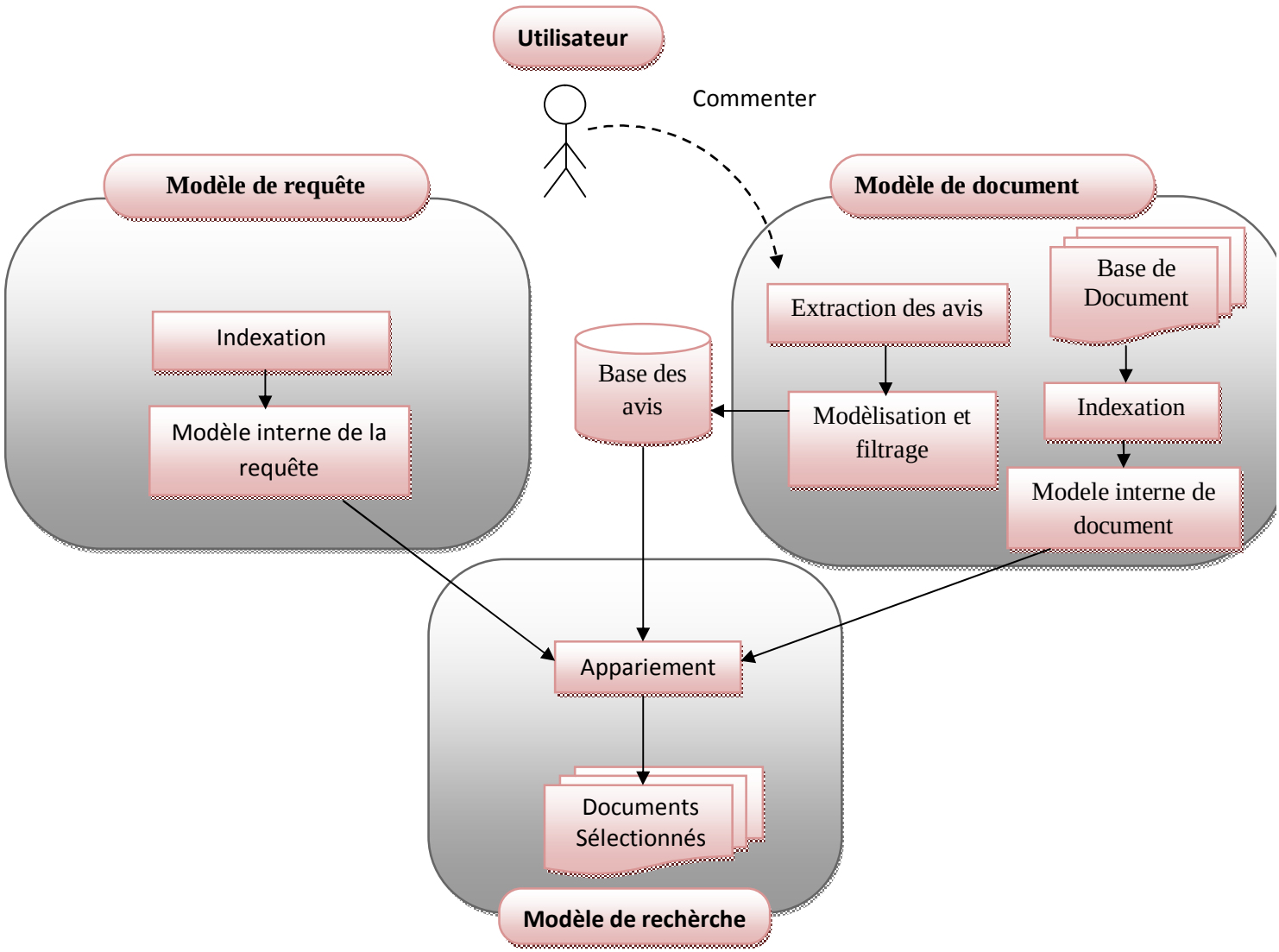


Figure 10 : Architecture général du système.

IV. Implémentation de l'approche :

Pour permettre une bonne mise en œuvre de notre modèle nous avons implémenté un ensemble de classes nous permettant au préalable d'indexer la collection ainsi que d'autres classes pour implémenter les différentes formules ainsi ces deux groupes de classes sont répartie selon deux niveaux :

Ø **Au niveau de l'indexation :**

Dans la phase de l'indexation, un ensemble de classes est mis en jeu, ces dernières nous permettent d'extraire à partir de chaque document les informations nécessaire à la mise en œuvre de notre modèle ces informations sont stockées dans une base de données.

Les classes utilisées sont les suivantes :

- ü **La classe *TestMotVide* :** qui permet d'extraire les mots, d'éliminer les mots vides en utilisant une StopList et de faire la troncature.
- ü **La classe *Indexation* :** cette dernière classe permet de sauvegarder dans la base de données les différentes informations telles que le numéro du document, les mots indexés… etc.

Les tables de base de donnée qu'on a crée au niveau de l'indexation sont :

- ü **La table document :** contient trois colonnes, numéro du document, titre du document (indexé), et le score du document qui est initialisé à 0.
- ü **La table tab_freq_doc:** contient les informations sur les termes de chaque document et les fréquences.

Ø **Au niveau de l'appariement :**

Les classes utilisées au niveau de l'appariement consistent à exploiter les différentes formules pour avoir un nouveau score pour chaque élément afin de réaliser l'ordonnancement.

Les classes crées à ce niveau sont :

- ü **La classe *FenetrePrincipale* :** qui permet de choisir la requête et les différents paramètres et récupérer les résultats du système en réponse à cette requête.
- ü **La classe *Score_classique* :** permet d'effectuer le calcul de score sur les éléments résultats en appliquant la formule BM_25 .
- ü **La classe *Score_sociale* :** permet de calculer le score social du document en appliquant la formule (score social) et insérer les résultats dans la base de données.

- **La classe *Score_final :*** permet d'effectuer le calcul final par la combinaison des deux classes précédentes en appliquant la formule décrite précédemment (Formule1).

Les tables de base de données créée à ce niveau sont :

- **La table *resultat_sc* :** contient les informations sur les résultats obtenus utilisant la formule de la BM_25.
- **La table *resulta_ss* :** contient les informations sur les résultats des documents en utilisant les nombres de voix pour chaque document. (Intégration des commentaires)
- **La table *resultat_sf* :** contient les informations sur les résultats finals obtenus en utilisant la formule 1.
- **La table *topic* :** contient le numéro et le titre de la topic.

V. Notre travail par étape :

Pour permettre une bonne mise en œuvre de notre modèle de recherche, nous avons répartie notre travail en plusieurs phases, dans chacune d'elle un ensemble d'outils et de programmes est utilisé pour réaliser différentes tâches :

- **Première phase :** L'indexation est la première étape dans le processus de recherche. S'occupe essentiellement de l'analyse des documents de la base afin de les représenter sous forme concise, porteuse d'information et exploitable par les autres sous-systèmes.

La représentation des documents selon un langage d'indexation est un travail délicat, et pour se faire nous nous sommes basée sur l'indexation automatique qui permet de décrire les documents par un ensemble de termes paramétrés, susceptible de représenter leurs contenus, en vue de leur exploitation ultérieure.les étapes de son processus sont :

-extraction des mots simple.

-élimination des mots vides.

-normalisation.

-pondération des termes de la liste.

-stockage de la liste des termes sélectionnés.

1. **extraction des mots simples :**

Dans cette étape, le document est parcouru mot par mot pour extraire tout les mots qui apparaissent dans le texte en comptant leur fréquence d'occurrences. Cette extraction est reliée grâce à la reconnaissance de séparateurs de mot (<<, >>, << ;>>, le blanc, point d'interrogation, etc.). Les mots extraits subiront en suite une conversion en minuscule et tous les mots comportant des caractères particuliers seront simplifiés (<<é>> <-> <<è>> <-> <<e>>) pour éviter tous comparaison de mots.

2. **élimination des mots vides :**

Dans cette étape ; on fait appel à un anti-dictionnaire (stoplist) pour éliminer les mots vides (pronom, article,…) qui peuvent exister dans les mots extraits précédemment.

3. **normalisation :**

Afin de réduire les différentes formes morphologique des mots sans perdre leurs sémantiques, nous utilisant la troncature droite à 7 caractères ; cette technique fournir généralement, les meilleurs résulta pour une recherche d'information probabiliste [soule 1990]

4. **pondération des termes :**

La pondération permet d'affecter à chaque terme son poids sémantique dans la base qui mesure son importance dans la caractérisation du contenu d'un document particulier ou d'une collection de documents.

Pour cela on utilise la fonction de pondération BM25, 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 fonctionBM25 suivante:

$$\boldsymbol{W = \frac{(k_1 + 1) \times tf}{k + tf} \times log \frac{N - n + 0.5}{n + 0.5} \times \frac{(k3 + 1) \times qtf}{k3 + qtf}}$$

Le tableau 1 illustre les variables de la fonction BM25 :

Variables	Description
N	Le nombre total de document dans la collection
N	Le nombre de document contenant le terme t, (**n<N***)*
Tf	La fréquence du terme t dans le document d
Qtf	La fréquence du terme t dans la requête q
Nq	Le nombre de terme dans la requête q
Dl	La longueur du document d
Avdl	La longueur moyenne des documents dans la collection
K	$k_1 \times ((1-b) + b \times \frac{dl}{avdl}) + tf$
k_1, k_3, d	Paramètres constants

Tableau. Paramètres de la fonction BM25.

5. stockage des résultats d'indexation :

La résultante des étapes 1 à 4 est un enregistrement descriptif pour chaque document de la base contenant les informations suivantes:

-terme : le terme extrait du document.

-fr : la fréquence d'apparition du terme dans le document.

-poids : la valeur du poids du terme dans le document.

Pour bien comprendre cette étape en va définir quelques algorithmes tels que l'algorithme d'indexation des documents.

- **Algorithme d'indexation**

Entrée : les documents.

Sortie : les documents indexés**.**

```
Début
    Initialiser une liste à vide;
    Pour un document à indexer faire
    Début
        Répéter
            extraction d'un mot simple();
            transformer les mots majuscules en mots minuscules;

            si (mot extrait ∈ Stoplist) alors
                éliminer ce mot vide;
            sinon
                troncature mot () ;

                si mot extrait ∈ liste() alors
                    fréquence du mot + =1;
                sinon
                    Début
                        insertion du mot dans la liste();
                        fréquence du mot =1;
                    Fin
                Finsi
            Finsi
        jusqu'à la fin du document.

            Pour chaque mot de la liste faire
            Début
                Si (terme ∈ table terme) alors
                    Début
                        Récupérer numéro du terme;
                        Table document. insérer (Num_terme, Num_doc, score);
                    Fin
                Sinon
                    Début
                        Table tab_freq_doc. insérer (terme, freq_doc);
                        Table document. insérer (Num_terme, Num_doc, score);
                    Fin
                Finsi
            Fin
                Vider liste;
    Fin
        Construire le fichier inverse();
        Pondération des mots avec la formule BM_25 ();
Fin
```

- **Deuxième phase :** La deuxième étape consiste à exploiter les éléments indexés pour appliquer les formules décrite précédemment.

Dans cette phase en va définir l'algorithme de recherche pour comprendre bien notre démarche.

- **Algorithme de recherche**

Entrée : requête à court terme et requête à long terme ;

Sortie : liste des documents pertinents ;

Début
-Initialiser à vide la liste L des documents pertinents ;
Procédure d'indexation de la requête :
Début
-Lire (requête) ;
-Initialiser à vide le vecteur « termerequête » des termes de la requête.
Répéter
-Extraire un mot de la requête ;
Si le mot contient des majuscules ***alors***
- Les transformer en miniscules ;
Finsi

Si le mot contient des caractères spéciaux ***alors***
-Le simplifier ;
Finsi

Si (le mot Î stopliste) ***alors***
-Eliminer le mot vide ;
Sinon

Si le mot contient plus de 7 caractères ***alors***
-Tronquer ce mot ;
Finsi
Si (le mot Ïtermerequête) ***alors***
Debut
-Insérer le mot dans termerequête ;
Freq_Term_Req (ti) =1 ;
Fin
Sinon
Freq_Term_Req (ti) = Freq_Term_Req (ti) + 1;
Finsi
Finsi

Jusqu'à la fin de la requête ;

Pour chaque terme ti de termerequête ***faire***
Début
- Récupérer la forme tronquée du terme ti ;
- Table Terme_Req.insérer (Term_Req, Freq_Term_Req);

Fin
Fin /* A ce niveau on obtient la requête interne R*/

- **Troisième phase** : La troisième partie consiste à réaliser des expérimentations sur la collection de test AP88. Un ensemble de requêtes issu de la compagne d'évaluation AP88 est exécuté tout en exploitant un ensemble initial de résultats obtenus par le calcule de score.

En calcul d'abord le score classique en utilisant la formule de la BM_25 , après le calcul du score social (nombre de voix d'un document/le nombre total des voix de toutes la collection AP88) et a la fin la combinaison des deux formules.

- **Quatrième phase :** La quatrième partie consiste à faire une évaluation de notre modèle.

VI. Outils de développement

Pour réaliser notre application nous avons utilisé :

- Un langage de programmation qui est JAVA ;
- Un environnement de développement qui est Eclipse ;
- un système de gestion de base de données relationnelle et objet MySQL ;
- Le pilote JDBC

VI.1 Le langage de programmation JAVA

Nous avons choisit pour l'implémentation de notre système d'exploitation des liens le langage de programmation JAVA, il a été mis au point par Sun Microsystems en 1991. C'est un langage de programmation à usage général, il est multi plate forme grâce à sa machine virtuelle appelée Java Virtual Machine (JVM) évolué et orienté objet de très haut niveau capable de s'exécuter sur n'importe quelle machine, parmi les caractéristiques les plus intéressantes de Java, on peut citer :

- Un langage orienté objet ;

- Indépendant vis-à-vis de la plate forme ;
- Avoir la capacité d'exécuter du code source extérieur de façon sécurisée permettant de compiler le code Java, c'est-à-dire de produire un exécutable capable de fonctionner hors de l'environnent Java ;
- La programmation peut se faire pour des exemples simples avec le compilateur Javac, mais pour avoir plus de confort il est préférable d'utiliser un environnement de développement intégré ou IDE, celui que nous avons utilisé est l'environnement Eclipse ;
- Doté d'un standard de bibliothèque de classes très riches comprenant la gestion des interfaces graphiques (fenêtres, boites de dialogues, contrôles, menus, graphismes), la programmation multithreads (multitâches), la gestion des exceptions, les accès aux fichiers et au réseau…. L'utilisation de ces bibliothèques facilite grandement la tache du programme lors de la construction d'applications complexes.

VI.2 Présentation d'Eclipse :

Eclipse est un projet open source fondé par SUN Microsystems. L'IDE Eclipse est un environnement de développement permettant d'écrire, de compiler, de déboguer et de déployer des programmes. Il est écrit en java, et il y'a un grand nombre de modules pour étendre l'IDE Eclipse.

L'IDE Eclipse est un produit gratuit, sans aucune restriction quant à son usage. L'installation de L'IDE Eclipse nécessite l'installation de la JDK (Java Développement Kit), le kit de développement java compatible avec la version d'IDE.

Pour concevoir notre application, nous avons utilisé la version Eclipse Europa

L'image suivante présente l'interface de travail sous Eclipse :

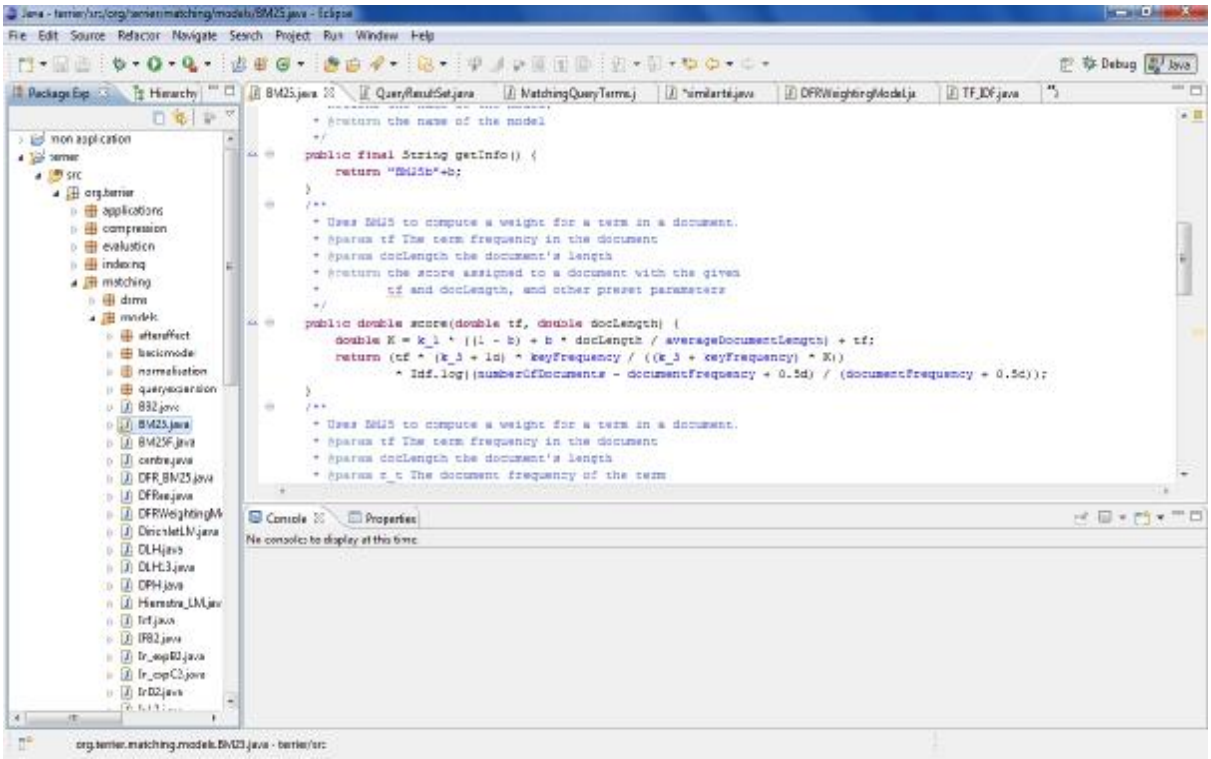


Figure 9 : capture d'écran présentant l'interface de développement d'Eclipse.

VII. Utilisation des commentaires pour le ré-ordonnancement des résultats

Pour réaliser un ré-ordonnancement des premiers résultats, nous avons utilisé dans nos expérimentations, les scores initiaux pour le ré-ordonnancement final des éléments.

Les expérimentations présentées dans cette section ont pour but d'évaluer l'impact des commentaires pour le ré-ordonnancement des résultats, la figure ci-dessous montre les résultats de la requête 283 avant et après intégration des commentaires sociaux.

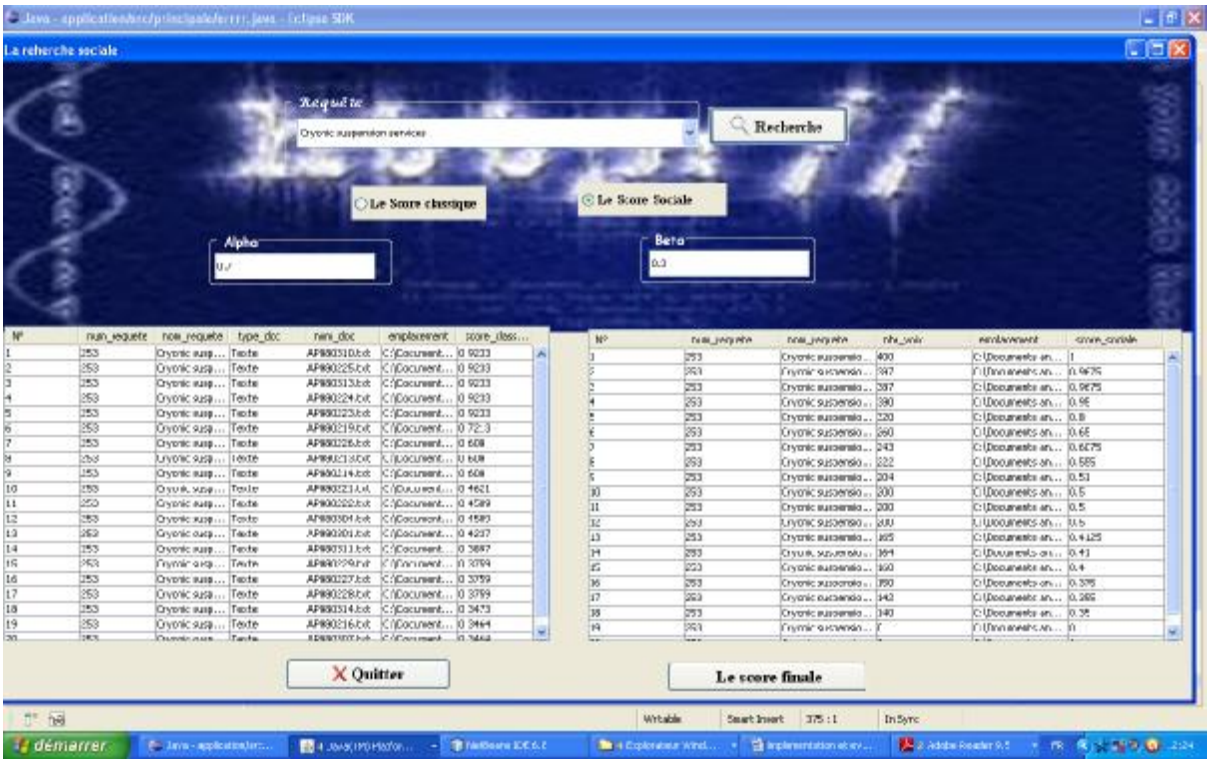


Figure 11 : Exemple de ré-ordonnancement des documents pour la collection AP88 requête 283 avant d'intégrer les commentaires sociaux.

Cette figure montre un résultat d'un ré-ordonnancement réalisé après l'application de la formule BM25. Les résultats sont classés par ordre décroissant.

En cliquant sur le bouton score finale en obtient l'interface suivante qui nous retourne le score finale.

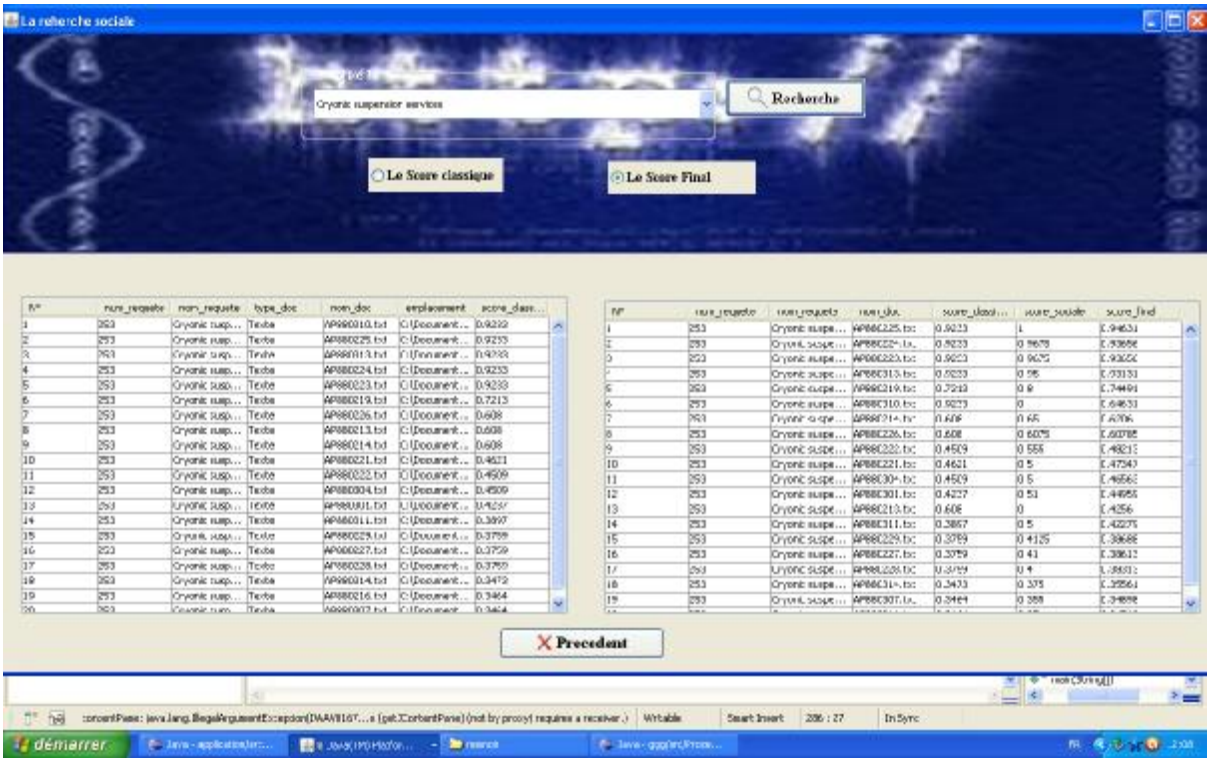


Figure 12 : Exemple de ré-ordonnancement des documents pour la collection AP88 requête 283.

La figure ci-dessus montre un résultat d'un ré-ordonnancement réalisé après l'application des différentes formules. Les résultats sont classés par ordre décroissant des valeurs des scores, ainsi certains éléments sont remontés dans le classement et d'autres on perdu de place dans le classement à l'image du document AP880223 qui s'est classé 5ème dans la première recherche sans intégration des commentaires et 3ème avec la deuxième recherche en intégrant les commentaires sociaux.

VIII. Expérimentation

v *La collection de test AP88*

Pour évaluer les différentes formules proposées [référence], nous nous appuyons sur une collection de test TREC AP88 (Associated Press 1988) qui est de taille moyenne et qui contient des documents plats, et un ensemble de requêtes qui se trouve dans le fichier Topics251-300.

Le tableau ci-dessous montre quelques statistiques sur la collection AP88

Collection	Taille	Documents	Topics
AP88	237 Mo	79 919	251-300

Le format d'un document dans la collection AP88 est le suivant :

<DOC>
<DOCNO> AP880212-0001 </DOCNO>
<FILEID>AP-NR-02-12-88 2344EST</FILEID>
<FIRST>u i AM-Vietnam-Amnesty 02-12 0398</FIRST>
<SECOND>AM-Vietnam-Amnesty,0411</SECOND>
<HEAD>Reports Former Saigon Officials Released from Re-education Camp</HEAD>
<DATELINE>BANGKOK, Thailand (AP) </DATELINE>
<TEXT>
More than 150 former officers of the overthrown South Vietnamese government have been released from a re-education camp after 13 years of detention, the official Vietnam News Agency reported Saturday...
</TEXT>
</DOC>

- ***Requête (topics251-300)***

L'ensemble des requêtes se trouvent dans le fichier Topics251-300 qui est un document plat.

Le format d'une requête est le suivant:

<top>

<num> Number: 251

<title> Exportation of Industry

<desc> Description:

<narr> Narrative:

</top>

IX. Tests et évaluation

1. Environnement d'évaluation

Dans cette section nous allons décrire l'environnement et les conditions expérimentales qui vont nous permettre d'évaluer les différentes requêtes que nous avons exécuté.

1.1. les requêtes utilisées :

Pour tester notre formule un ensemble de 6 requêtes illustrées dans la table ci-dessous est exécuté et voici un récapitulatifs de ces requêtes :

Numéro de la requête	Titre de la requête
253	Cryonic suspension services
257	Cigarette Consumption
280	Ban on Ivory Trade
283	China Trade
292	Worldwide Welfare
294	Animal husbandry for exotic animals

Table 6.2 Requêtes de la collection AP88.

1.2. Paramètre :

Nous n'avons pas utilisés plusieurs variations des valeurs des paramètres a, β, pour les tests puisque la collection est gigantesque et que chaque exécution prend beaucoup de temps,
Nous avons donc fixé ces valeurs comme suit:
$\alpha=0.7$, $\beta=0.3$

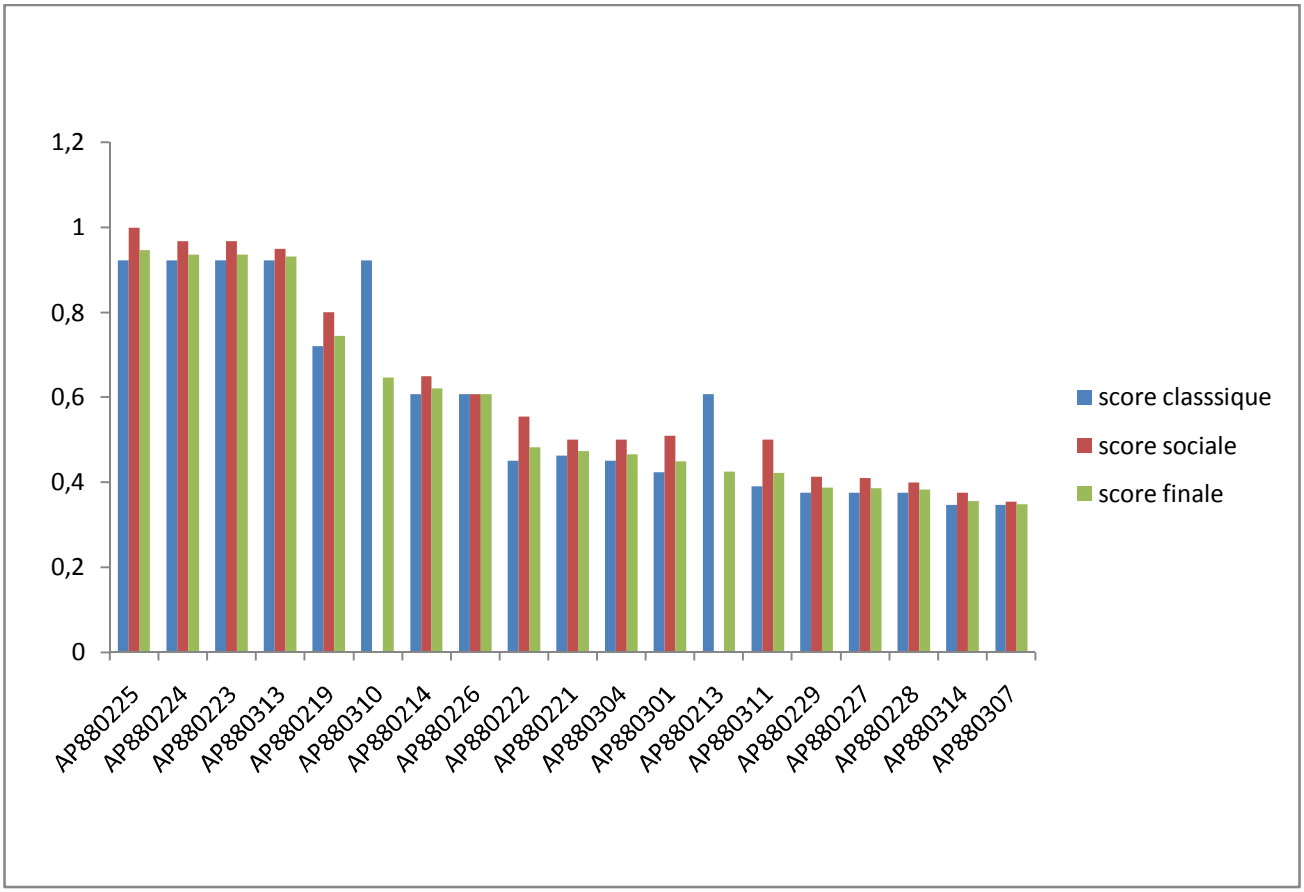
2. les résultats de l'évaluation

Les résultats présentés dans cette section ont été obtenus à partir de l'évaluation des requêtes montrées dans le tableau précédant.
Voici un graphe qui nous permet de voir l'importance d'intégrer les commentaires sociaux pour améliorer le classement des documents :

	score classique	score sociale	score finale
AP880310	0,9233	0	0,64631
AP880225	0,9233	1	0,94631
AP880313	0,9233	0,95	0,93131
AP880224	0,9233	0,9675	0,93656
AP880223	0,9233	0,9675	0,93656
AP880219	0,7213	0,8	0,74491
AP880226	0,608	0,6075	0,60785
AP880213	0,608	0	0,4256
AP880214	0,608	0,65	0,6206
AP880221	0,4621	0,5	0,47347
AP880222	0,4509	0,555	0,48213
AP880304	0,4509	0,5	0,46563
AP880301	0,4237	0,51	0,44959
AP880311	0,3897	0,5	0,42279
AP880229	0,3759	0,4125	0,38688
AP880227	0,3759	0,41	0,38613
AP880228	0,3759	0,4	0,38313
AP880314	0,3473	0,375	0,35561
AP880216	0,3464	0,35	0,34748
AP880307	0,3464	0,355	0,34898

Le graphe suivant montre les résultats obtenus en intégrant les commentaires sociaux (le score social dans cette figure)

En appliquant les différentes formules d'évaluation qu'on a définie précédemment les résultats du modèle que nous avons implémenté pour quelques requêtes issue de la collection AP88 nous avons réalisé une comparaison entre le score classique et le score final avant et âpres intégration des commentaires, cette comparaison est illustrée par le graphe suivant :



Le graphe montre une nette amélioration au niveau de l'ordonnancement des documents. Nous pouvons conclure que l'intégration des commentaires améliorent de manière significative les résultats de la recherche, leur prise en compte s'avère donc importante pour réaliser un système de recherche fiable répondant aux besoins et attentes des utilisateurs.

Conclusion :

Nous avons présenté dans ce chapitre l'implémentation et l'évaluation de notre approche, notamment l'indexation des documents et la recherche d'information.

Pour la recherche, notre système parcours les structures qui se trouvent dans la base des avis afin de déterminer les documents qui correspondent au mieux à la requête de l'utilisateur.

Notre travail se situe dans le contexte de la recherche d'information, plus particulièrement la recherche d'information sociale qui exploite les interactions sociales au sein des réseaux sociaux.

Les différentes approches de la recherche d'information classique ignorent l'influence des relations sociales et des interactions de l'utilisateur au sein de son contenu social, sur le processus global de la recherche d'information. Vu l'importance que peut apporter ce type de méta-informations sur la qualité de la recherche, nous avons pensé à une solution hybride qui tient compte des commentaires sociaux. Les résultats obtenus par notre solution montrent une amélioration considérable vis-à-vis de la recherche traditionnelle. On a pu, à travers les avis sociaux, faire apparaitre, parmi les résultats de la recherche, des documents pertinents mais qui étaient 'cachés'. Les avis sociaux ont contribué à leur découverte.

L'objectif de notre travail était d'améliorer la Recherche d'Information classique, ou en intégrant les commentaires. Les documents commentés permettent aux utilisateurs de retrouver rapidement et d'extraire dans une base documentaire des documents qui répondent mieux à leurs besoins.

Enfin, la réalisation de ce travail nous a permis de nous imprégner dans des domaines qui sont nouveaux pour nous et d'acquérir de nouvelles connaissances en RI et en RIS où l'on manipule diverses informations sociales. Nous avons approfondis nos connaissances dans le domaine de la programmation avec le langage orienté objet Java en implémentant l'indexation et les phases de recherche documentaire.

Comme suite à notre étude, on envisage d'intégrer d'autres types de commentaires sociaux puis étendre notre application à d'autres informations sociales (les tags…).

Pour les tests, il serait préférable de faire la simulation sur un cas réel de réseau social, nous avons l'intension de valider notre approche sur un champ réel.

Bibliographie

[WF94] *Wassermann et Faust : Methods and Applications* publié en 1994

[1] Http://www.id-blog.net/ras-le-bol-des-reseaux-sociaux-a289-html.

[TMA, 02] M. Tmar. Modele auto-adaptatif de filtrage d'information : apprentissage incremental du profil et de la fonction de decision. Thèse de doctorat, Université Paul Sabatier, Toulouse, France, juillet 2002.

[ABC98] Abrams D., Baecker R. et Chignell M. : Information archiving with bookmarks: personal web space construction and organization. *Dans CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems*, p. 41–48, New York, NY, USA, 1998. ACM Press/Addison-Wesley Publishing Co.

[Szi99] Szirmai J: *The Archaeology of Medieval Bookbinding*. Scolar Pr, août 1999.

[GH05] Golder S. A. et Huberman B. A. : The structure of collaborative tagging systems. *CoRR—Computing Research Repository*, abs/cs/0508082, 2005.

[HJS06] Hotho A., Jäschke R., Schmitz C. et Stumme G. : Information retrieval in folksonomies: Search and ranking. *Dans Proceedings of the 3rd European Semantic Web, 2006*

[FKK09] Fu W.-T., Kannampallil T. G. et Kang R. : A semantic imitation model of social tag choices. *Computational Science and Engineering, IEEE International Conference on*, 4:66–73, 2009.

[Smi04] Smith G.: Atomiq. folksonomy: Social classification. http://atomiq.org/ archives/2004/08/folksonomy_social_classification.html, août 2004.

[SCH, 97] B. Schneier. *Cryptographie appliquée*. International Thomson Publishing Company, Paris 1997.

[MAN,85] MANIEZ, Jacques ; MUSTAFA EL HADI, Widad. *Organisation des connaissances en vue de leur intégration dans les systèmes de représentation et de recherche d'information*. Lille : UL3 – travaux et recherche, 1999, 398 p.

[SM,86] G. Salton and M. J. McGill. Introduction to Modern Information Retrieval.McGraw-Hill, Inc., New York, NY, USA, 1986.]

[TEB, 04] H. Tebri. Formalisation et spécification d'un système de filtrage incrémental d'information. Thèse de doctorat, Université Paul Sabatier, Toulouse, France, décembre 2004.

[sau, 05] K.sauvagnat, « modèle flexible pour la recherche d'information dans des corpus de documents semi-structurés » thèse de doctorat de l'universitéPaul sabatier, spécialité informatique, 2005

[Dah, 06] F. DAHAK, « *Indexation des documents semi-structurés: Proposition d'une approche basée sur le fichier inversé et le Tree.* » Thèse de Magister, 2005.

[Hla, 07] L.Hlaoua, « reformulation de requêtes par réinjection de pertinences dans les documents semi-structurés » Thèse de doctorat de l'université Paul sabatier, spécialité informatique, 2007

[ARK, 08] D.ARKAM et Y.CHIBANE « système de recherche d'information dans des documents XML basé sur un modèle probabiliste » Mémoire d'ingénieur, Institut National de formation en informatique (I.N.I), Alger, 2008

[Bou, 00] M. Boughanem, « Formalisation et spécification des systèmes de recherche et de filtrage d'information », Hdr, Univ. Paul Sabatier de Toulouse, 2000

[Sal, 71] G.Salton, « The SMART Retrieval System-Experiments», in Automatic Document Processing. Prentice-Hall, Engle wood,Cliffs,New Jersey,1971.

[Luh, 57] H.P.Luhn, « A statistical approach tomechanized encoding and searching of literary information», Jounal of IBM-JRD,1(4):309-317,october1957.

[Sal, 83] G.Salton and M. McGill, « Introduction to Modern Information Retrieval», McGraw-Hill, NewYork, 1983.

[Sal, 70] G. Salton, The SMART retrieval system: Experiments in automatic document processing. Prentice Hall, 1970.

[Urf, 04] Urfist, Cours "Problématique Générale de la Recherche d'Information" URFIST Bretagne-Pays de Loire, Alexandre Serres, 2004 .

[DDP] (dictionnaire du droit privé).

[DLF] (Dictionnaire de la langue française

[DAF] Dictionnaire de l’academie francaise (8eme edition)

[DAF] Définition de l'Académie française (éd. 1986)

[DAF ed 32-35] Signification de l'Académie française (éd. 1932-35)

[DEL] Dictionnaire d'Emile Littré

[SEAF] Signification éditée en 1835 par l'Académie Française

[ADAF] Ancienne définition de 1798 (Académie Française)

JAVA

I. Historique du 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 *Patrick Naugthon*, *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 en fait et 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(garbage collector), 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 crée avec pour objectif 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 du 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 des bibliothèques 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 sur 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 Interfaces).

II. 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 capablesd'effectuer plusieurs actions (*thread*) en même temps. Un même programme peut parexemple à l'aide de trois *threads* faire un calcul, tout en chargeant une vidéo et endiffusant 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, …
- ***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 bytecode 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++.

III. Terminologie :

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

Le rôle des technologies orientées objet est de fournir un certain nombre de mécanismes destinés à organiser ces composants en objets. Dans la suite, nous allons présenter les différents concepts utilisées par une telle technologie :

Ø Un objet :

C'est une entité qui possède une identité et des caractéristiques, et qui est susceptible d'effectuer des actions.

Ø Une classe :

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 définie par une structure de données et un comportement. Les objets qui possèdent les mêmes caractéristiques (structure de données et comportement) sont regroupés en une classe. Dans un programme, l'objet est crée a partir d'une classe, on dit qu'il s'agit d'une instance de cette classe.

Ø Un attribut :

Un attribut est un élément de la structure de donnes d'une classe, lors de chaque instanciation d'une classe, une occurrence de chacun des attributs de la classe est crée pour l'objet instancie.

La valeur d'un attribut est une caractéristique de l'objet, et l'ensemble des valeurs de ses attributs représentant l'état de cet objet. Il est fortement recommande de ne modifier l'état d'un objet que via l'invocation des méthodes de celui-ci.

Ø Une méthode :

Une méthode est élément du comportement d'une classe. Elle peut en fait assimilée à une fonction effectuant un certain nombre de traitements, susceptibles ou non, de modifier la valeur des attributs d'un objet. Tout comme un attribut, une méthode est définie dans la classe dont est issu l'objet. L'ensemble des méthodes définit le comportement de la classe correspondante.

Ø 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 ou non à un même package, et d'une manière globale, rend l'ensemble plus lisible.

Ø L'héritage:

L'héritage permet à une classe d'objet 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 avoir à redéfinir ce que les deux classes ont de commun.

- **Le polymorphisme:**

Offre la possibilité d'associer à un comportement, une implémentation différente en fonction de l'objet auquel on se réfère (**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.

- **L'abstraction :**

Introduit des mécanismes favorisant la dissociation entre la déclaration d'une classe et son implémentation.

- **L'encapsulation:**

Permet de dissimuler les détails du fonctionnement interne d'une classe aux par apport autres classes.

IV. Types de programmes Java :

Java implémente principalement trois types de programmes :

- ***Les applications :*** Ce sont des programmes autonomes capables de fonctionner sur n'importe quel ordinateur possédant une machine virtuelle Java (JVM).
- ***Les applets :*** ce sont des petits programmes java intégrés dans des pages web, les applets sont transférés lors de l'appel de la page web par l'intermédiaire du navigateur, elles sont chargées et démarrées sur une machine virtuelle java intégrée au navigateur.
- ***Les servelets :*** ce sont des programmes java intégrés dans une page web. La principale différence avec les applets est le code java effectif qui n'est pas transféré sur le poste local, il est exécuté sur le serveur. Seuls les sorties sont transférées aux clients.

V. Les primitives du langage :

Java a été crée par la société SUN. C'est un langage orienté objet c'est-à-dire que les éléments manipulés sont des classes, ou plus exactement des objets c'est à dire des instances de classes.

Ces objets contiennent des données possédant un type (et une représentation). Ces données sont un ensemble d'éléments stockés en mémoire et baptisés *Primitive*. Les données manipulées avec java sont typées c'est-à-dire que pour chaque donnée que l'on utilise (dans les variables par exemple) il faut préciser le type de données, ce qui permet de connaître l'occupation mémoire (le nombre d'octets) de la donnée ainsi que sa représentation, ceci peut être :

- Des nombres entiers (Int) .
- Des nombres réels (Float).

De plus, le langage java introduit un type de données appelé *boolean*. Ce type de variable accepte deux états true (vrai) correspond à une valeur vraie. False (faux) correspond à une valeur fausse.

Remarque : Si une variable n'est pas initialisée, sa valeur par défaut est false.
Voici un tableau répertoriant les primitives (données) de java :

Primitive	**signification**	**Plages de valeurs acceptées**	**Taille (en octets)**
Char	caractère	Valeur du jeu de caractères Unicode (65000 caractères possibles)	**2**
Byte	Entier très court	-128 à 127	**1**
Short	Entier court	- 32768 à 32768	**2**
Int	Entier	-2147483648 à 2147483647	**4**
Long	Entier long	-9223372036854775808 à 9223372036854775807	**8**
Float	Flottant (réel)	$-1{,}9\ 10^{-324}$ à $1{,}710^{308}$	**4**
Double	Flottant double	$4{,}9\ 10^{-324}$ à $1{,}7\ 10^{308}$	**8**
boolean	Boolean	0 ou 1 (en réalité, toute autre valeur que 0 est considérée égale à 1).	**1**

VI. Le kit de développement Java :

L'écriture d'un programme java ne nécessite pas d'outils spécifiques : un éditeur de texte tel que bloc-notes suffit. Le programme java qui porte l'extension. Java doit être compilée afin de générer un fichier *bytecode* portant l'extension .class, qui sera interprétable par une machine virtuelle java.

Pour ce faire il est nécessaire de télécharger le JDK (Java Developement Kit) sur le site de *SUN (www.javasoft.com).* Ce kit est disponible pour différentes plates-formes de développement, il existe en outre plusieurs versions : la **1.0.x**, la **1.1.x**, la **1.2.x**, la **1.3.x**,…etc. Ce kit de développement comprend plusieurs outils, ne disposant pas tous d'interface graphique.

Voici les principaux outils :

- ***Javac*** : il s'agit du compilateur java. Il traduit un fichier source d'extension .java en un fichier bytecode 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- à-d 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 contrôle (**ex** : s'assurer que l'exécution d'un programme en provenance d'un serveur distant, ne porte préjudice à l'intégrité de la machine locale).
- ***Applet Viewer*** : 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 tout le JDK pour exécuter un programme compilé par javac. Il est destiné à des plateformes clientes qui ne développent pas d'applications java mais qui l'utilisent.
- ***Jbd*** : 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 les sources.
- ***Javap*** : cet utilitaire permet de désassembler un fichier compilé (bytecode) en code source (java).

- ***Jar*** : cet utilitaire permet d'archiver plusieurs classes java en un fichier d'archive Jar (à partir de la version ***1.1.x*** du JDK).

Parmi les utilitaires utilisés, on a aussi :
Javah, Jit, Native2ascii, Rmic, Rmiregistery, Javakey, Keytool,…etc.

VII. 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 oeuvre 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.

VIII. 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 *bytecode* en instructions exécutables par le processeur de la machine hôte, on trouve ainsi plusieurs implémentation 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 (bytecode verifier) :***

Si le compilateur est censé d'assurer que le code source temps qu'il est en train de traiter ne risque pas de violer un certain nombre de règles de sécurité, alors le vérificateur de bytecode 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.

Ø ***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.

Ø ***Le moteur d'exécution :***

C'est le coeur de la machine virtuelle, car il prend la charge de convertir le *bytecode* 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.