République Algérienne Démocratique et Populaire

Ministère de L'enseignement Supérieur et de la Recherche Scientifique

Université Kasdi Merbah Ouargla

Faculté des Nouvelles Technologies de l'Information et de la Communication

Département d'Informatique et Technologie de l'Information

Mémoire

MASTER ACADEMIQUE

Domaine : Informatique et Technologie de l'Information

Filière : Informatique

Spécialité : Informatique Industrielle

Thème

Amélioration de la précision et du temps de réponse d'un moteur de recherche de texte

Présenté par :

- **Sabiha BRAHIMI**
- **Hamza KOUADRI**

Soutenu publiquement
Le : 15 / 06 /2014

	Président	MEFLAH Salim
	Examinatrice	KORICHI Mariam
Dr. Mohammed Lamine Kherfi	Encadreur/ Rapporteur	Mohammed Lamine Kherfi

Année universitaire 2013/2014

Remerciement

Nous tenons tout d'abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d'accomplir ce Modeste travail.

En second lieu, nous tenons à remercier notre encadreur Mr : Docteur Mohammed LAMINE KHERFI, pour son précieux conseil et son aide durant toute la période du travail.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail Et de l'enrichir par leurs propositions. Nous exprimons notre gratitude à tous les enseignants du département de mathématique et informatique – Ouargla- qui nous ont donnés beaucoup de connaissance. Un grand merci à nos parents pour leur contribution, leur soutien et leur patience. Je tiens à exprimer ma reconnaissance envers mon mari DJAMEL LABIDI qui sans lui je ne suis pas là devant vous. Enfin, nous adressons nos plus sincères remerciements à tous nos proches et amis, qui nous ont toujours soutenue et encouragée au cours de la réalisation de ce mémoire.

{رَبِّ أَوْزِعْنِي أَنْ أَشْكُرَ نِعْمَتَكَ الَّتِي أَنْعَمْتَ عَلَيَّ وَعَلَى وَالِدَيَّ وَأَنْ أَعْمَلَ صَالِحًا تَرْضَاهُ وَأَصْلِحْ لِي فِي ذُرِّيَّتِي إِنِّي تُبْتُ إِلَيْكَ وَإِنِّي مِنَ الْمُسْلِمِينَ} [الأحقاف: 15].

Dédicaces

À mon père qui nous a quittés soudainement l'an passé et qui accompagne mes Pensées.

À ma mère que dieu me la garde, mes deux filles RACHA et MOUNIA, mon fils ABDELDJALLIL mes frères et mes sœurs, mes neuves et mes nièces, mes belles sœurs et mes beaux-frères et toutes la famille LABIDI

À toi Djamel mon mari. À Tous ceux qui m'ont connu, soutenu et aimé.

À tous mes amis et camarades et collèges.

Madame BRAHMI

Résume

On constate une grande évolution des moteurs de recherche, et de grands efforts sont faits pour les améliorer. Comme nous le savons, il y a une quantité énorme d'informations disponibles avec différents formats et expansion à la disponibilité du document d'information qui rend la recherche normale (la recherche par navigation) insuffisante pour récupérer des informations de différentes sources et domaines. Il y a plusieurs domaines scientifiques en évolution, y compris la recherche sémantique, qui ont pour objectif d'améliorer la précision de recherche par la compréhension du but de la recherche et la signification contextuelle des termes. De nombreuse techniques permettant d'extraire des informations à partir de sources de données structurées comme une ontologie et c'est le cas du Web sémantique, car le Web d'aujourd'hui contient des informations dispersées écrite par l'homme en langage naturel.

Ces technologies permettent l'articulation formelle des connaissances de domaine à un niveau élevé de l'expressivité et pourraient permettre à l'utilisateur de préciser ses intentions plus en détail lors de la requête. Le but de ce travail est de construire un moteur de recherche semi-intelligent, en exploitant la recherche sémantique et l'ontologie de l'information. Nous allons essayer d'intégrer les concepts du moteurs de recherche, système de gestion électronique des documents (GED) et la programmation orientée objet afin de créer un système intégré. L'utilisateur aura un index privé pour son information il devra la rechercher dans son contenu, mais la structure de l'ontologie utilisé par la recherche sémantique sera partagé entre les objets (utilisateurs). L'utilisateur pourra contribuer à la modernisation et la modification de la structure de l'ontologie afin de rendre la recherche sémantique plus précise après chaque utilisation.

Mot clés : moteur de recherche, ontologie, GED, annotation sémantique. Ontologie, GED, annotation sémantique.

ملخص:

هناك بحوث كثيرة في مجال أنظمة البحث بشكل خاص أو استرجاع المعلومات بشكل عام وقد بذلت مجهودات كبيرة في سبيل تطوير نظم البحث، لأنه كما نعلم أصبحت المعلومات متوفرة بكم هائل وبأشكال مختلفة فهذا التوسع في كم المعلومات المتوفرة جعل من البحث العادي (بحث الملاحية) وهوا الانتقال من وثيقة لأخرى غير كافي لاسترجاع المعلومات باختلاف مصادرها ومجالاتها.

وقد ظهرت عدة مجالات علمية في صياغ هذا التطور منها البحث الدلالي سعيا لتحسين دقة البحث من خلال فهم نية الباحث والمعنى السياقي للمصطلحات كما تظهر في فضاء البيانات للبحث .

أيضا مجموعة من التقنيات لاسترجاع المعرفة من مصادر بيانات منظمة غنية مثل الانطولوجيا كما وجدت على الويب الدلالي نظرا لأن الواب اليوم وسط غني بالمعلومات المبعثرة تلقائيا والمكتوبة من طرف بشر بلغة بشرية لها مقصود ودلالة. وهذه التكنولوجيات تمكن من صياغة رسمية لمجال المعرفة على مستوى عال من القدرة على التعبير وتمكين تحديد نية المستخدم بمزيد من التفاصيل في وقت الاستعلام.

الهدف من هذا العمل هو بناء محرك بحث دلالي شبه ذكي، باستغلال البحوث في مجال البحث الدلالي وأنطولوجيا ألمعلومات سوف نحاول دمج تقنية محرك البحث وإدارة المحتويات الرقمية (GED) والبرمجة الكائنية بهدف خلق نظام متكامل بحيث سوف يكون المستخدم ككائن مستقل من حيث فهرسة المعلومات التي سوف يحتاج لاسترجاعها فيما بعد، لكن بنية الأنطولوجيا التي يعتمد عليها البحث الدلالي سوف تكون مشتركة والمستخدم في حد ذاته من يسهم في تحديثها وتغيير بنيتها ليكون البحث أكثر دقة بكثرة الاستخدام.

ABSTRACT:

There is a lot of research about search engines. A great effort has been made in the development of research systems, because as we know, there is tremendous amount of information available in different forms and this expansion in the number of documents available makes the normal search (navigational search) moving from document to another not convenient to retrieve information for different sources and domains.

There is an evolution in all scientific fields including semantic search to improve search accuracy by understanding the intent of the user and the contextual meaning of terms

Also a set of techniques for retrieving knowledge from richly structured data sources like ontologies as found on the Semantic Web, because Web today Contains scattered information written by humans in natural language, has a meaning.

Such technologies enable the formal articulation of domain knowledge at a high level of expressiveness and could enable the user to specify his intent in more detail at query time.

The purpose from this work is to build a search engine semi-intelligent, exploiting semantic search and the information ontology, we will try to integrate the search engine concepts, content management system (GED) and object-oriented programming in order to create an integrated system, the user will be independent object has private indexed for his information that will need to makes search in its content , but the semantic search's ontology structure will be shared and the user in itself will be able to contributes to modernization and change its structure in order to semantic search more precise after frequently used.

Keywords: search engine, ontology, GED, semantic annotation

Table des Matères

Table des figures

Table des tableaux

Introduction générale

1. Introduction.

Avec l'apparition des premiers ordinateurs naquirent l'idée d'utiliser des machines pour automatiser la recherche d'information dans les bibliothèques. Cela fut notamment popularisé en 1945 par Vannevar Bush dans son célèbre article « As We May Think ».Les premiers systèmes utilisés par des bibliothèques permettent d'effectuer des recherches booléennes, c'est-à-dire la recherche d'un terme dans un document conduit à la sélection du document où la présence (ou l'absence) du terme.

La recherche d'information est le domaine qui étudie la manière de retrouver des informations dans un corpus. Le volume d'information disponible électroniquement est toujours plus important et trouver des documents pertinents est une tâche de plus en plus délicate. L'ambigüité du langage naturel contribue à cette difficulté tant en termes d'expression du besoin d'information que de l'évaluation de la correspondance entre documents et besoins.

La recherche d'information consiste à mettre en œuvre une stratégie permettant de trouver de l'information pertinente en réponse à un besoin d'information. L'information sélectionnée doit être fiable et de qualité. La recherche doit être menée avec efficacité sans perte de temps.

Notre projet consiste à développer un outil de recherche, qui doit chercher des informations pertinentes au sein de documents. Il se base sur deux aspects fondamentaux qui sont l'indexation des documents et leurs interrogations à l'aide des requêtes formulées par les utilisateurs.

Pour l'indexation nous avons opté pour LUCENE, et pour avoir une recherche fructueuse nous devons mettre les ressources textuelles ou multimédias dans un index, qui seront sémantiquement étiquètes par des métadonnées afin que les agents logiciels puissent les exploiter.

2. Problématique :

De jour en jour les informations augmentent au point où elles deviennent ingérables. Selon le cabinet d'études Nielsen Netratings, environ 70% des visites d'un site web proviennent d'un moteur ou service de recherche, le reste provient de « bonnes adresses » données par un proche, ou de la publicité [1].Quand on sait que la Toile est devenue le vecteur principal des échanges commerciaux entre entreprises, et un canal majeur dans la vente et les services aux particuliers, on comprend mieux l'enjeu considérable de la recherche d'informations sur le net. C'est pour cela qu'il y a eu l'apparition du premier logiciel de gestion de documents d'entreprise. De manière générale, les moteurs de recherche sont des services qui aident leurs utilisateurs à trouver des informations sur internet. Il existe beaucoup de logiciel de gestion de documents d'entreprise qui fonctionne comme les GED (Gestion électronique des documents). Malgré l'émergence de nouvelles technologies très prometteuses (Collaboration, Cloud Sync…). Le GED manque d'outils flexibles de recherche pour gérer une énorme quantité d'informations et qui doit être récupérée. La diffusion de l'information au niveau du GED engendre un besoin d'accéder aux informations disponibles sur le domaine de sources différentes et hétérogènes. Ceci ne peut se faire que si les tâches suivantes sont accomplies :

1- Identification et gestion des différents domaines.
2- faciliter l'accès à ces informations.
3- rendre ces informations accessibles et disponibles sachant qu'ils proviennent de domaines hétérogènes

Dans notre mémoire nous allons nous attaquer seulement au premier et au second problème.

1- L'identification et la gestion l'information des documents selon leur domaine :

- L'identification de domaine : dans les systèmes d'information la méthode la plus utilisée pour décrire un domaine d'information c'est la méthode représentation formelle du

[1] http://www.secrets2moteurs.com/page/6?s=nielsen

contenu, exprimée à l'aide de concepts, relations et instances décrits dans une ontologie informatique.

- La gestion des documents **:** La gestion met principalement en œuvre des systèmes d'acquisition, d'indexation, de classement, de gestion et stockage…, mais ces tâches sont associées à un domaine décrit par une ontologie, alors l'index présente une ressource bien construit, sinon il demeure pratiquement inexploitable et impossible de retrouver notre recherche. L'annotation sémantique des documents est une solution à ce problème. L'annotation sémantique est une représentation formelle d'un contenu, exprimée à l'aide de concepts, relations et instances décrits dans une ontologie, reliant les ressources documentaires à la source.

2- Faciliter l'accès à ces informations :

Dans ce problème, nous allons parler généralement de la façon de fabriquer un moteur de recherche et comment exploiter l'index du premier problème, ici nous allons définir deux problèmes principaux :

a. Silence : On parle de silence lorsque des réponses pertinentes ne sont pas proposées par le système d'interrogation, sachant qu'elles existent. Cela peut arriver notamment avec les systèmes de recherche binaire.

Exemple. "Barcelone" est une ville de «l'Europe» et il existe un document qui parle du bureau de Yahoo à Barcelone par contre lorsqu'un utilisateur demande "entreprises de IT en Europe" il recevra aucun résultat. Mais imaginez votre moteur de recherche comprend que "Barcelone" est une ville de «l'Europe», il peut répondre à une requête de recherche sur les «entreprises de IT en Europe" avec un lien vers un document sur Office Yahoo à Barcelone, bien que les mots exacts "Barcelona" ou "Yahoo" ne se produisent jamais dans votre requête de recherche.

b. Bruit : On parle de bruit lorsque des réponses non-pertinentes sont proposées par le système. Ces réponses sont mêlées à des réponses pertinentes mais ces dernières risquent de ne pas être vues par l'utilisateur. Cela peut arriver notamment avec les systèmes de recherche qui manquent de crédibilité de traitement automatique du langage (TAL).

Exemple. Dans l'exemple précédent nous reformulons la requête de recherche par «entreprises de IT en Europe mais pas espagnol » alors toujours il répondre avec un lien vers un document sur Office Yahoo à Barcelone, mais normalement ce résultat est non-pertinentes.

Nous allons essayer de minimiser ces valeurs comme une valeur de valorisation d'augmentation. [1]

Organisation du mémoire.

Le mémoire est organisé en trois parties :

Partie I : Description générale des outils existants.

Chapitre 01 : Nous présenterons les moteurs de recherche en donnant des exemples des moteurs de recherches sur le web, moteurs de recherche d'entreprise, moteurs de recherche open-source.

Chapitre 02 : dans ce chapitre, on parlera des GED ; un petit historique suivit d'une définition ainsi que le fonctionnement de GED et en fin la mise en place d'une solution de GED.

Partie II : Description des outils.

Chapitre 01 : La collecte d'informations (CRAWLING & INDEXATION)

Dans ce chapitre nous allons parler du crawling et ifilter nous allons présenter Lucene et son fonctionnement, ses classes sans oublier lucene.net.

Chapitre 02 : La recherche basique dans ce chapitre nous présentant la Recherche avec Lucene et nous allons parler des principales classes de Lucene.

Chapitre 03 : Sémantique Nous allons parler du web sémantique L'annotation sémantique en pratique ainsi l'indexation avec une terminologie orientée ontologie.

Chapitre 04 : Cette partie concernera, la phase de construction de l'ontologie du domaine, la phase de conception du système qui exploite cette ontologie et les différents modules de ce système.

Partie III : Conception, Réalisation.

Dans cette partie nous allons parler de la conception et la réalisation de notre application.

En fin La **conclusion générale**.

[1] http://fr.wikipedia.org/wiki/Bruit_et_silence

PARTIE I :

Description générale des outils existants

CHAPITRE I :

Les moteurs de recherche

1. Introduction.

Un moteur de recherche est une application qui peut d'acquérir des informations sur un sujet précis de diverses ressources. Les moteur de recherche sur le web est doté de « robot», encore appelés bots, spiders, crawlers ou agents qui parcourent les sites régulièrement et automatiquement pour découvrir de nouvelles adresses (URL). Ils suivent les liens hypertext rencontrés sur chaque page visitée. Chaque page identifiée est alors indexée dans une base de données, qui pourra être consulté par les internautes grâce à de mots clés. On peut retrouver des moteur de recherche sur les PC appelé desktop qui fait la recherche les fichiers stokes dans la mémoire du PC.

2. Un peu d'historique.

Le premier moteur de recherche apparait en 1990, crée par Adam Emtage, étudiant à Mc Gill(Québec). Ce moteur, dénommé **Archie**, comportait les principes de base du moteur de recherche : on remplissait une base de données, que le moteur faisait correspondre aux requêtes des utilisateurs. Le Web de l'époque comportait seulement quelques centaines de sites, et Archie resta un projet universitaire.

Mais le saut technologique le plus important fut introduit par **Wanderer** (« le Vagabond ») en 1993 par Matthew Gray. Il fut le premier moteur à déployer des robots d'indexation (spiders). L'idée de base, qui était de mesurer la croissance du Web, fut rapidement remaniée pour arriver au premier moteur de recherche à indexation automatique (Bot search) Ce moteur a d'ailleurs causé un certain nombre de problèmes, car il retournait plusieurs centaines de fois par jour sur certains sites et les ralentissait.

En octobre 2003, le successeur d'Archie fait son apparition : Aliweb (Archie-like indexing the web). Ce moteur repose sur la soumission manuelle de sites. Le moteur se basait sur les mots clés et les descriptions fournies au moment de l'inscription pour effectuer la recherche.

Le premier moteur intelligent fut Excite (1993). Construit par six étudiants de Stanford, il se base sur l'analyse statistique des mots.

Enfin, en 1994, c'est la naissance de Yahoo, le premier « grand » service de recherche, crée également par des étudiants de Stanford. Mais à la différence des outils de l'époque,

Yahoo se base sur un annuaire, pas sur un moteur de recherche. Les résultats sont sélectionnés et indexés par l'homme. En quelques mois, Yahoo devient le plus important portail du Web.

Les années 1995-1997 voient l'apparition des grands moteurs de recherche (Excite, Hotbot, Lycos…). Altavista, crée par un français et jugé efficace et rapide, deviendra la star des moteurs de recherche du moment jusqu'aux années 2000, détrôné par Google.

De son coté, Inktomi développe la première activité de recherche destinée aux entreprises. C'est la première fois que les moteurs de recherche ciblent les professionnels.

Enfin, c'est en 1998 que naît Google, crée par Sergei Brin et Larry Page, encore une fois étudiante de Stanford. Google va littéralement révolutionner le monde de moteurs de recherche grâce à sa simplicité et son efficacité. L'interface dépouillée se charge instantanément sur les connexions bas-débit de l'époque, et la technologie d'indexation est inédite : Google se base sur le nombre de liens pointant sur une page pour en déterminer sa pertinence.

Vers 2001-2002, l'éclatement de la bulle internet fait disparaitre les premiers moteurs de recherche, et seuls les plus grands survivent. C'est l'ère moderne de la recherche internet. [1]

3. Définition :

Le moteur de recherche permet à l'internaute de trouver un ou plusieurs sites web qui répondront à ses attentes. L'internaute demande au moteur de recherche des informations sur ce qu'il souhaite trouver, et le moteur de recherche lui fournit une liste de sites classés par pertinence. La pertinence des résultats est un calcul complexe calculé par l'algorithme du moteur de recherche. L'algorithme analyse plusieurs centaines d'éléments sur tous les sites web afin de les classer, et de proposer à l'internaute le meilleur résultat possible selon une requête.

4. Des exemples moteurs de recherche.

4.1. Moteurs de recherche Web.

4.1.1 Google : Google est le principal moteur de recherche du marché, et également une des plus grosses entreprises informatique du monde. www.google.com vous permet de faire une recherche sur la globalité des sites du monde entier et vous permet de trouver des sites, des images, des vidéos et des actualités, classées par pertinence.

[1] http://oseox.fr/referencement/histoire-moteurs.html

4.1.2. Yahoo **:** Moteur concurrent de Google, il offre sensiblement la même qualité de réponse lors des requêtes simples et un index presqu'aussi conséquent, mais est moins pertinent dans le cas d'une requête complexe. L'index est sensiblement de la même taille que Google.Yahoo propose l'alternative d'une recherche par annuaire. Yahoo était à l'origine un portail web, une sorte de page d'accueil où l'on y retrouve tout un tas d'actualités, d'information utile. Yahoo propose les mêmes services que ses deux concurrents en plus de la recherche sur Internet

4.1.3. Bing **:** c'est le moteur de recherche de Microsoft (pour rappel l'éditeur de Windows). On y retrouve les mêmes services que Google, mais le moteur est moins utilisé que son concurrent

4.1.4. Exalead **:** est un moteur de recherche conçu en France, basé sur la spécificité du langage français et fonctionnant sur le clustering pour générer des termes associés, mais aussi pour catégoriser à partir d'une liste définie. Il permet de pré-visualiser les pages web grâce à des vignettes et des fonctions avancées proposées pour affiner la recherche (termes associés, type de ressources, langue, annuaire…).

4.2. Moteurs de recherche d'entreprise **:**

4.2.1. Autonomy **:** La société exploite plusieurs technologies issues des travaux de recherche de l'université de Cambridge. Elle développe des applications de recherche d'entreprise et de gestion des connaissances grâce à des techniques de reconnaissance de formes centrées sur l'inférence bayésienne en conjonction avec les méthodes traditionnelles. En mars 2009 Autonomy a acquis la société de gestion de contenu Interwoven, maintenant Autonomy Interwoven et Autonomy iManage. Auparavant indépendante, Autonomy a été acquise par Hewlett-Packard en octobre 2011[1]

4.2.2. Kartoo : est un méta-moteur qui présente les résultats des recherches sous forme de carte heuristique. L'affichage des résultats se fait sous forme de pages vignettes reliées entre elles par liens sémantiques générés dynamiquement. Kartoo a été un moteur de recherche capable d'exécuter simultanément une recherche sur plusieurs moteurs et annuaire de recherche sur le web.

4.2.3. PolySpot **:** PolySpot Enterprise Search est une solution de recherche documentaire permettant aux utilisateurs d'obtenir une vision globale de l'information

[1] http://fr.wikipedia.org/wiki/Autonomy

provenant de sources hétérogènes, sécurisées et stockées à l'intérieur ou à l'extérieur de l'entreprise. PolySpot Enterprise Search facilite l'accès aux documents aidant ainsi les utilisateurs à capitaliser sur le fonds d'information et à partager leur connaissances. Ses principales fonctions sont : recherche plein texte, Social Search (ou moteur de recherche collaboratif), navigation par facettes, réglage de la pertinence, suggestions d'expressions. Le moteur de recherche PolySpot dispose de fonctions sémantiques (extraction d'entités nommées : identification de personnes, lieux, organisation, produits...) et permet l'analyse des sentiments/de la tonalité des données indexées.

4.3. Moteurs de recherche open-source **:**

4.3.1. DataparkSearch **:** DataparkSearch est un moteur de recherche open source écrit en C. Il est distribué sous la licence publique générale GNU et conçu pour effectuer des recherches dans un site web, un groupe de site web, ou intranet ou sur un système en local. DataparkSearch peut indexer nativement des données text/plain, text/html et text/xml, et beaucoup d'autres types de données en utilisant des parsers (analyseur) externes.

4.3.2. OpenSearchServer (**OSS**) : est un serveur d'applications en open source permettant le développement d'applications reposant sur des index comme les moteurs de recherche. Disponible depuis avril 2009 en téléchargement sur SourceForge.net, OpenSearchServer, disponible sous licence GNU GPL v3, propose une série d'analyseurs syntaxiques et peut être installé sur différents systèmes (Windows, Linux, Macintosh).[1]

Ses principales fonctionnalités sont : un crawler pour base de données, pages web et documents riches ; une interface conviviale Zkoss permettant le développement de la plupart des applications en quelques clics ; extraits de textes ; facettes ; un outil de restitution html pour intégrer les résultats de la recherche dans une page et des fonctions de monitoring et d'administration.

4.3.3. Lucene **:** est un moteur de recherche libre écrit en JAVA qui permet d'indexer et de rechercher du texte. C'est un projet open sources de la fondation APACHE mis à disposition sous licence APACHE. Il est également disponible pour les langages Ruby, Perl, C++, PHP, C#.

[1] http://fr.wikipedia.org/wiki/Opensearchserver

CHAPITRE II :

La gestion électronique des documents (GED)

Chapitre II —— La gestion électronique des documents (GED)

Introduction

GED signifie Gestion Electronique des Documents. Il désigne le processus de gestion de l'ensemble du cycle de vie d'un document électronique. Le terme GED étant jugé trop généraliste, il est souvent nommé GEIDE pour Gestion Electronique de l'Information et du Document. Existant. En effet le document n'est plus seulement un document papier transformé, mais il est souvent déjà créé de façon électronique.

Un peu d'historique

Gestion électronique des informations et des documents, apparue avec l'apparition des imprimantes virtuelles (distiller / writer / …) qui peuvent transférer les dossiers informatiques (DOC. / XLS / doc…etc.) Aux systèmes électroniques (SPOOL) eux-mêmes permettant le transfert des dossiers informatiques à travers les imprimantes virtuelles (post script) pour être traités comme tout autre document (GED COLD : computer output on laser disk). GED : gestion électronique des documents, apparue en 1985 dans le but de gérer les archives actives, semi actives et inactives. La GED est basée sur une architecture client/serveur avec un système d'exploitation UNIX, un SGBDR (oracle) et un réseau local.

3. Qu'est-ce que la GED

3.1 Définition

La **Gestion Electronique des Documents (GED)** désigne un procédé informatisé visant à organiser et gérer des informations et des documents électroniques au sein d'une organisation. Le terme GED désigne également **les logiciels permettant la gestion de ces contenus documentaires.**

3.2 Objectifs

Un système GEID a pour objectifs : les gains d'espace de stockage le gain du temps de recherche le gain du coût de classement le gain du coût de diffusion .ainsi la maîtrise de la qualité de la productivité et la rapidité d'accès et aussi la sûreté et sécurité et enfin la confidentialité.

4. La Chaine du document numérique

Le cycle de vie du document numérique

Les logiciels de GED permettent la prise en compte de tout le cycle de vie du document qui passe par quatre étapes majeures : l'acquisition, le classement, le stockage, et

la diffusion. Desquelles découlent un certain nombre de fonctionnalités dans les logiciels de GED.

4.2 L'acquisition des documents

L'acquisition d'un document provient d'un processus automatique ou humain

Les différentes phases de l'acquisition : la création, l'enregistrement, le classement et enfin l'indexation

4.3 L'indexation des documents numériques

L'indexation constitue la description du document et de son contenu en vue de faciliter son exploitation. On distingue à ce titre :

- **l'indexation par type** offre une description formelle du document en utilisant ses métadonnées (type, auteur, titre, source, date, etc.) dont le vocabulaire est standardisé afin de permettre l'utilisation de ces métadonnées par le plus grand nombre d'outils de recherche.
- **l'indexation par concepts ou mots-clés** qui visent plutôt le contenu du document pour faciliter les opérations de recherche. Il peut s'agir ici, pour le concepteur du système ou le créateur du document, de recenser les termes qui apparaissent le plus souvent, on parle alors d'indexation statistique.

4.4 Le stockage

Il est important de réfléchir sur le système de stockage des documents. Il doit être adapté le mieux possible avec le volume des documents et choisit en fonction de la fréquence de consultation et de l'importance des données, et doit offrir un temps d'accès fiable.

5. Mise en place d'une solution de GED

5.1 Analyse préalable

Pour mettre en place un GED .En premier analysez les besoins, cela nous permet d'avoir une appréciation globale des tâches et des fonctions de chacun dans l'organisme.

- de quels documents avez-vous le plus souvent besoin ?
- quels types ou formats de documents traitez-vous le plus souvent ?

Cette analyse vise également à prendre en considération les différentes contraintes organisationnelles, les techniques, juridiques, budgétaires

Une fois les besoins recensés, il faut les formaliser dans un cahier des charges.

.

5.2 Les écueils à éviter

Pour garantir le succès lors de la mise en place d'un GED on doit prendre en considération quelque risque .Car sa mise en place est couteuse en temps et en ressources humaines .C'est pour cela il faut une étude préalable. Lors de la mise en place d'une solution de GED, plusieurs difficultés peuvent être rencontrées comme le manque d'implication du personnel ainsi la réticence des futurs acteurs du système face à l'arrivée d'un nouvel Si on prend en considération des mesures le résultat sera garanti. Comme la formation du personnel, et l'implication de la hiérarchie.

5.3 Choix du logiciel

5.3.1 Fonctionnalités

On distingue deux types de GED : la GED bureautique et la GED documentaire.

La GED bureautique

Cette GED gère les documents bureautiques de l'entreprise. Elle vient compléter les applications bureautiques. Elle intervient au niveau des échanges et des flux entre les postes de travail producteurs. Ce type de GED doit présenter les caractéristiques suivantes - La GED doit être fortement intégrée à l'environnement de productivité. C'est un élément fédérateur des échanges, car elle se trouve au niveau de la gestion et de la traçabilité des flux. Elle permet notamment la gestion de l'archivage des documents ainsi que le contrôle des différentes versions d'un document

La GED documentaire

Ce type de GED gère les documents de référence de l'entreprise. Elle permet de conserver la mémoire de l'activité de l'organisme. Elle nécessite un langage d'indexation (thésaurus). Ce langage doit être assez générique afin d'être partagé par tout le monde, mais aussi spécifique pour que le moteur de recherche soit pertinent et précis. Le rôle du professionnel de l'information prend toute son ampleur ici dans la construction d'un langage d'indexation.

5.3.2 Analyse du logiciel

La mise en place d'une solution de GED passe par l'analyse du logiciel. Plusieurs éléments doivent être analysés :

Les fonctionnalités offertes :

- acquisition et/ou création de document
- stockage et archivage
- recherche

- consultation des documents
- diffusion
- fonctions spécifiques à l'entreprise

L'architecture technique :

- relations stations serveur / stations clientes, réseau, protocole
- développement spécifique du logiciel de GED
- intégration à l'environnement bureautique

L'offre de service de l'éditeur :

- Paramétrage, installation, formation, maintenance, support technique
- Reprise de l'existant, conception d'un thésaurus, numérisation

6. Conclusion

Bien que la gestion électronique des documents soit coûteuse, elle permet de gagner un temps énorme. Au sein d'une organisation, le choix de gérer ses documents ou pas peut relever d'une stratégie d'entreprise. En effet, même si la GED favorise le travail collaboratif, elle peut s'avérer être une perte financière énorme pour les petites, moyennes et grandes organisations qui ne l'envisageraient que sous l'angle "logiciel". Nous avons présenté GED car notre produit et un GED.

PARTIE II :

Description des outils

CHAPITRE I :

La collecte d'informations (CRAWLING & INDEXATION)

Chapitre I—— La collecte d'informations (Crawling & Indexation)

CRAWLING

Le crawler est un logiciel d'analyse structurelle, syntaxique et sémantique de page Web (parser en anglais). Pour chaque page, il extrait les éléments jugés significatifs et pertinents, afin de se constituer une base de mots-clés relatifs à la page analysée. Lorsque le spider détecte des liens vers d'autres pages, il les garde en mémoire dans une base de données contenant des adresses restant à analyser. Une fois la page analysée, le spider regarde dans sa base de données la prochaine page à visiter, et ainsi de suite. Mais le robot n'agit pas pour autant à l'aveuglette. Lorsqu'il arrive sur une page, il détermine tout d'abord s'il « connait » la page, autrement dit s'il l'a déjà indexé. Si c'est le cas, il fera un passage plus rapide, se contentant de relever les modifications effectuées depuis sa dernière visite. Le robot doit aussi déterminer s'il est autorisé ou non à indexer la page. Cela se fait au moyen de directives standards mises au point par Google, et contenues dans un fichier intitulé « robots.txt ». Ce fichier permet de limiter ou modifier la façon dont les moteurs de recherche référencent un site. Il est ainsi possible de préserver les fichiers sensibles de la divulgation.

Les statistiques concernant les robots sont jalousement tenues secrètes. Mais selon toute vraisemblance, les spiders les plus puissants sont capables d'analyser des centaines de milliers de pages par jour. [25]

1.1 IFilter C#

Les IFilters sont des produits qui autorisent les services d'indexation de Microsoft à lire différents formats de fichier. Sans un IFilter approprié, le contenu du fichier ne sera pas indexé, et quand on cherche ces contenus, on ne trouve rien .Les IFilters peuvent apporter une aide à l'interopérabilité.

1.2 DocumentCloud Projects

DocumentCloud est une plate-forme logicielle basée sur le Web créé pour les journalistes afin de permettre la recherche, l'analyse, l'annotation et la publication de documents de source primaire utilisées dans les rapports. Il permet aux utilisateurs de rechercher, télécharger, modifier et organiser des documents

1. Lucene

1.1 Historique

. Depuis son lancement, Lucene a connu une évolution fulgurante et de nombreuses mises à jour ainsi que des améliorations réalisées par la création de plusieurs sous-projets. Le projet Lucene est créé en 1997 par Doug Cutting, développeur et créateur mais aussi spécialiste des technologies de recherche textuelle chez Xerox puis Apple. Celui-ci l'a mis en

téléchargement en Mars 2000 sur le site *SourceForge.net*. Sa toute première version publique 0.01 en Java date de Mars 2000. Lucene n'a connu que 3 versions sous SourceForge.net. Il est devenu en Septembre 2001 un projet officiel d'Apache Jakarta, qui gère des projets de logiciels libres, écrits en langage java. C'est en Juin 2002 que sa première version, sous le projet Apache Jakarta, voit le jour. Dès lors, sa popularité s'accroît, attirant plus de développeurs et d'utilisateurs. Le projet se développe réellement après la sortie du livre

« Lucene in action » écrit par Erik Hatcher avec la collaboration de Doug Cutting en Décembre 2004. A la suite de la publication de cet ouvrage, Lucene compte à son actif 11 versions différentes sur lesquelles ont travaillé une douzaine de développeurs dont les deux auteurs du livre [1] .D'une version à une autre, les changements, correction de bugs ont été plus ou moins importants, des fonctionnalités nouvelles sont apparues : la seconde version Lucene issue du projet Apache Jakarta en Décembre 2003 étend le type de formats possibles pour l'indexation comme Word, PDF, etc. Les versions suivantes font apparaître de nouveaux types de requête, les vecteurs de terme, l'optimisation de la classe IndexSearcher, la suppression et le rajout de documents, la pré-analyse des champs et l'amélioration des performances de Lucene. D'autres changements plus techniques sont apparus. La nouvelle version *Lucene in Action*. D'une version à l'autre, les changements, correction de bugs, ont été plus ou moins importants. Le projet Lucene est aujourd'hui maintenu par une équipe de développeurs bénévoles dont Erik Hatcher et Doug Cutting, qui continuent d'y travailler.

1.2 Fonctionnement de Lucene

Lucene, est un moteur de recherche et d'indexation. Il faut savoir qu'un moteur de recherche est un logiciel construisant des indices sur le texte et répondant aux requêtes utilisant cet index. Il offre la pertinence, l'adaptation, et peut intégrer des sources de données différentes comme le courrier électronique, les pages web, les fichiers et les bases de données. Lucene agit en quelque sorte comme une couche intermédiaire entre les données à indexer et vos programmes. Pour ce faire, il indexera des objets appelés des documents et, à partir des index, il permettra une recherche rapide et efficace dans ces documents. Notez ici le terme document pourrait être un texte Word, un fichier PDF, un ensemble de fichiers, une page web sur un serveur distant, des informations stockées dans une base de données, etc. Lucene n'a qu'une seule exigence : le document original doit pouvoir être converti en fichier texte. Concrètement, vous pourrez utiliser Lucene dans un programme Java en faisant appel à

[1] cf. *Lucene in Action d'Erik Hatcher*

des classes créées à l'avance qui effectueront tout le travail lié à l'indexage et à la recherche dans un index. Lucene repose sur quatre packages principaux : indexation, analyse, recherche et résultats. Ses packages contiennent un ensemble de classes spécifiques qui permettent un grand nombre de possibilités. Voici un schéma présentant la structure et l'architecture de Lucene, qui sont expliquées point par point dans les parties suivantes.

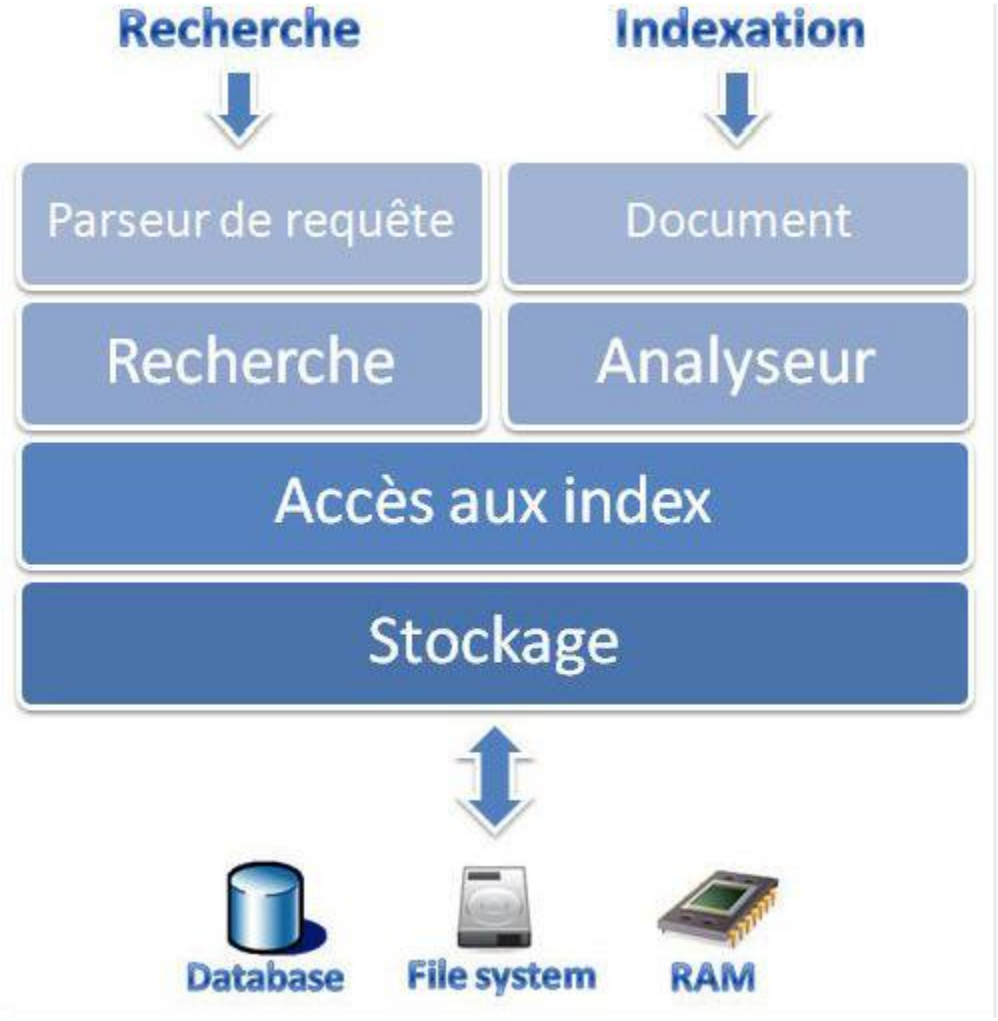


Fig.1 : Architecture et organisation de Lucene

1.2.1 Indexation

Lucene a recours au package nommé *org.apache.lucene.index* contenant les classes IndexWriter et IndexReader. L'index est une structure de données stockée sur le système de fichiers. Les documents de l'utilisateur vont être ajoutés à l'index, contient une série de documents, les termes retenus après l'analyse, les champs et les segments. Tout document de Lucene est composé de champs divers : titre, auteur, contenu (contents). Chaque

champ contient un nom et une valeur. A l'intérieur du champ, on retrouve une séquence de termes.

Exemple : titre : langage de programmation

Nom du champ : titre valeur du champ : langage de programmation

Lucene est capable de lire un très grand nombre de formats : PDF, Word, HTML, XML et TXT. Au moment de l'indexation, il ne traitera uniquement que le contenu textuel des documents. Voici un schéma qui montre comment sont exploités les documents au moment de l'indexation. Avant d'être indexée, la structure syntaxique et le texte des documents sont analysés .De plus, lors de l'indexation, il va assigner à chaque document de l'index un identifiant unique (Document ID).

Après la création d'un index, il est possible de rajouter ou supprimer des documents avec l'instance IndexWriter. Les données de l'index sont lues par le biais de la classe IndexReader. Il est stocké dans un répertoire unique. Son emplacement, déterminé par la classe Directory provenant du package org.apache.lucene.store, est situé dans le système de fichiers. L'utilisateur aura recours à l'implémentation : FSDirectory comme pour la base de documents choisie pour la mise en œuvre

L'index est composé de segments, pouvant être considérés comme des sous-index bien qu'ils ne soient pas entièrement indépendants. Lucene va assigner à chaque document de l'index un identifiant unique (Document ID). Les segments conservent les éléments suivants :

- Les noms des champs utilisés dans l'index,
- Un dictionnaire des termes : les termes contenus dans chaque champ,
- La fréquence des termes : numéros de tous les documents contenant ce terme,
- Proximité des termes : la position de chaque terme,
- Les documents supprimés.

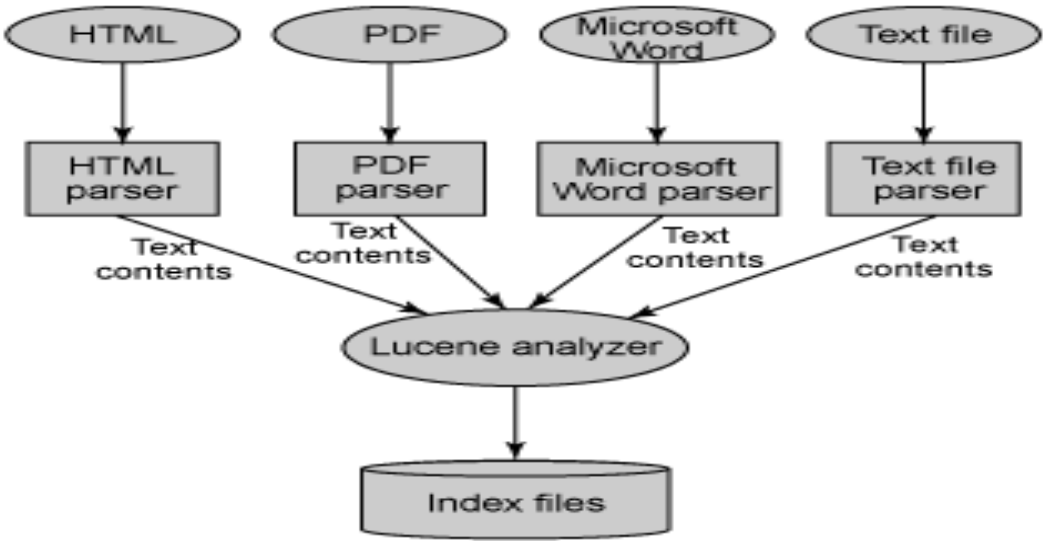


Fig 2 : ***Architecture d'indexation***

1.2.2 Recherche

Le modèle standard espace vectoriel (the vector space model) est utilisé par le moteur de recherche Lucene. Il a pour but de donner plus d'importance aux termes apparaissant souvent (term frequency) dans le document, mais qui sont relativement rares dans l'ensemble de la base de documents. Les documents et requêtes sont représentés comme des vecteurs. Si un terme apparaît dans un document, sa valeur dans le vecteur est non-nulle. Le vecteur se présente sous cette formule : V = [w1, w2,….,wn] où w est le poids de chaque terme.
Ces informations proviennent de l'ouvrage suivant : [[1]]**.**

Néanmoins, on peut constater que le modèle booléen est inclus dans Lucene dans le sens où le document correspond ou ne correspond pas à la requête demandée. C'est lui qui attribue la pertinence des documents. Si la requête est bonne, il retourne les scores et un ensemble de documents sinon il retourne « false » c'est-à-dire qu'aucun résultat ne s'affiche.

1.3 Classes

1.3.1 Classes d'indexation

- **IndexWriter**

La classe IndexWriter est le composant central du processus d'indexation. Cette classe crée un nouvel index et ajoute des documents à un index existant. On peut se la représenter comme un objet par lequel on peut écrire dans l'index mais qui ne permet pas de le lire ou de le rechercher.

[1] *Introduction to Information Retrieval* écrit par Christopher D. Manning

- **Directory**

La classe Directory représente l'emplacement de l'index de Lucene. IndexWriter utilise une des implémentations de Directory, FSDirectory, pour créer son index dans un répertoire dans le Système de fichiers. Une autre implémentation, RAMDirectory, prend toutes ses données en mémoire. Cela peut être utile pour de plus petits indices qui peuvent être pleinement chargés en mémoire et peuvent être détruits sur la fin d'une application.

- **Analyzer**

Avant que le texte soit dans l'index, il passe par l'Analyser. Celui-ci est une classe abstraite qui est utilisée pour extraire les mots importants pour l'index et supprime le reste. Cette classe tient une part importante dans Lucene et peut être utilisée pour faire bien plus qu'un simple filtre d'entrée.

- **Document**

La classe Document représente un rassemblement de champs. Les champs d'un document représentent le document ou les métadonnées associées avec ce document. La source originelle (comme des enregistrements d'une base de données, un document Word, un chapitre d'un livre, etc.) est hors de propos pour Lucene. Les métadonnées comme l'auteur, le titre, le sujet, la date, etc. sont indexées et stockées séparément comme des champs d'un document.

- **Field**

Chaque document est un index contenant un ou plusieurs champs, inséré dans une classe intitulé Field. Chaque champ (field) correspond à une portion de donnée qui est interrogé ou récupéré depuis l'index durant la recherche.

1.3.2 Classes de recherche

- **IndexSearcher**

La classe IndexSearcher est à la recherche ce qu'IndexWriter est à l'indexation. On peut se la représenter comme une classe qui ouvre un index en mode lecture seule.

- **Term**

Un terme est une unité basique pour la recherche, similaire à l'objet field. Il est une chaîne de caractère : le nom du champ et sa valeur. Notez que les termes employés sont aussi inclus dans le processus d'indexation.

- **Query**

La classe Query est une classe abstraite qui comprend Boolean Query, Phrase Query, PrefixQuery, PhrasePrefixQuery, RangeQuery, FilteredQuery, et SpanQuery.

- **TermQuery**

C'est la méthode la plus basique d'interrogation de Lucene. Elle est utilisée pour égaliser les documents qui contiennent des champs avec des valeurs spécifiques.

- **QueryParser**

La classe QueryParser est utilisée pour générer un décompositeur analytique qui peut chercher à travers un index.

- **Hits**

La classe Hits est un simple conteneur d'index pour classer les résultats de recherche de documents qui apparaissent pour une interrogation donnée. Pour des raisons de performances, les exemples de classement ne chargent pas depuis l'index tous les documents pour une requête donnée, mais seulement une partie d'entre eux.

1.4 Lucene.NET

Lucene.Net est une bibliothèque de haute performance d'Information (RI), aussi connu comme une bibliothèque de moteur de recherche. Lucene.Net contient de puissantes API pour créer des index de texte intégral et la mise en œuvre des technologies de recherche avancée et précis dans vos programmes. Certaines personnes peuvent confondre Lucene.net avec un prêt à utiliser l'application comme une recherche web / robot, ou une application de recherche de fichiers, mais Lucene.Net n'est pas une telle demande, c'est une bibliothèque de cadre. Lucene.Net fournit un cadre pour la mise en œuvre de ces technologies difficiles

vous. Lucene.Net ne fait aucune discrimination sur ce que vous pouvez index et de recherche, qui vous donne beaucoup plus de puissance par rapport aux autres indexations de texte intégral / recherche implications ; vous pouvez indexer tout ce qui peut être représenté sous forme de texte. Il y a aussi des façons d'obtenir Lucene.Net à l'index HTML, des documents Office, fichiers PDF, et bien plus encore.[1]

[1] http://www.codeproject.com/Articles/29755/Introducing-Lucene-Net

CHAPITRE II :

La recherche basique (moteur de recherche de base)

Chapitre II ___ La recherche basique (moteur de recherche de base)

1. Recherche Lucene

Lucene est un moteur de recherche textuelle Open Source et passant bien à l'échelle fourni par la fondation Apache. Vous pouvez utiliser Lucene dans des applications commerciales ou Open Source. Les puissantes APIs de Lucene se concentrent surtout sur l'indexation et le recherche. Il peut être utilisé pour ajouter des capacités d'indexation à des applications comme des clients de courrier, des listes de diffusion, des applications effectuant des recherches sur Internet ou dans une base de données, etc Lucene a beaucoup d'atouts :

- Il utilise des algorithmes de recherche puissants exacts et efficaces.
- Il calcule un score pour chaque document qui correspond à des critères données et retourne les documents les plus appropriés classés par score
- Il propose de nombreux types de requêtes, telles que PhraseQuery, WildcardQuery, RangeQuery, FuzzyQuery, BooleanQuery, et d'autre score.
- Il permet l'analyse d'expression de requetage riche du type de celle qui est saisies par des êtres humains
- Il permet à l'utilisateur d'étendre le comportement de la recherche en utilisant des tris personnalisés, des filtres et l'analyse d'expression de requetage.
- Il utilise un mécanisme de verrouillages basé sur les fichiers pour empêcher la modification d'index concurrent.
- Il permet d'indexer et rechercher simultanément.

Construction d'application avec lucene :

Comme la montre **Figure 3**, construire une application de recherche complète avec Lucene implique principalement l'indexation et la recherche des données, et l'affichage des résultats d'une recherche permettent d'indexer et de rechercher simultanément.

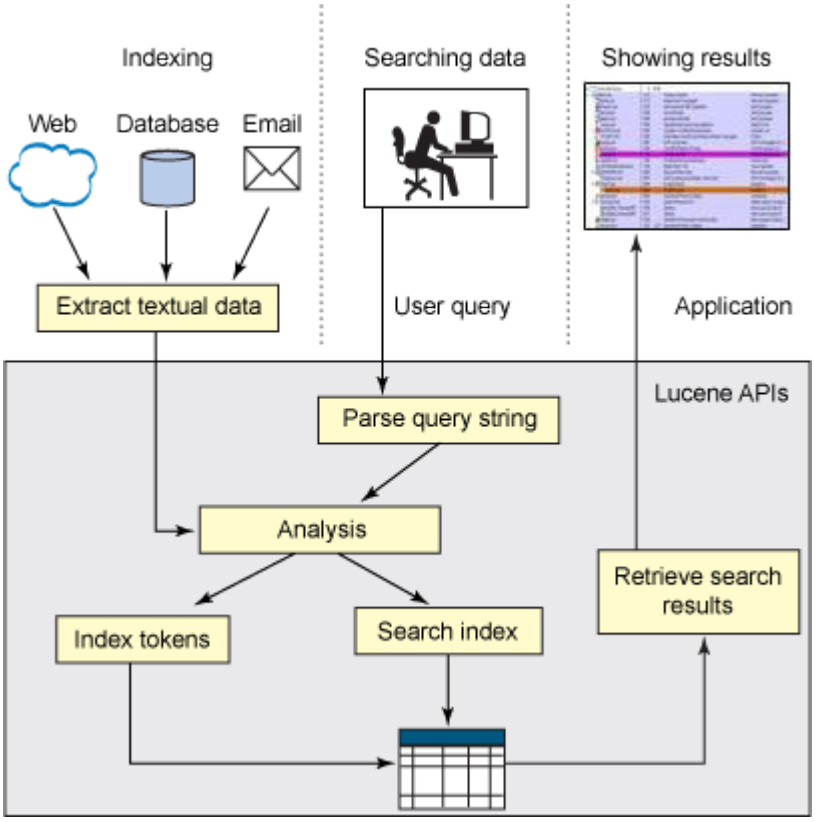


Fig 3 : les étapes de la construction d'application avec lucene

2. Indexation des données

Lucene vous permet d'indexer toute donnée disponible dans un format texte. Il peut être utilisé avec presque n'importe quelle source de données tant que de l'information textuelle peut en être extraite. La première étape dans l'indexation des données est de les rendre disponibles sous un format textuel simple. Pour ce faire, on utilise des analyseurs personnalisés et des convertisseurs

2.1 Processus d'indexation

L'indexation est le processus de conversion des données textuelles vers un format qui facilite une recherche rapide. Une analogie simple est celle de l'index que l'on trouve a la fin d'un livre : cet index indique la localisation des différents sujets qui sont traites dans le livre Lucene stocke les données en entrée dans une structure de données appelée index *inversé*, qui est lui-même stocké dans le système de fichiers ou en mémoire comme un ensemble de fichiers d'index. La plupart des moteurs de recherche Internet utilisent un index inversé. Cela permet aux utilisateurs d'exécuter des recherches par mot-clé rapides et de trouver les documents répondant à une requête données. Avant que la donnée textuelle ne soit ajoutée à l'index, elle est traitée par un analyseur

2.2 L'analyse

L'analyse est la conversion des données textuelles en unîtes fondamentales de recherche appelées termes. Pendant l'analyse, le texte subit plusieurs opérations : extraction des mots, suppression des mots les plus courants de la ponctuation, réduction des mots à leurs racines, passage en minuscules, etc. L'analyse du texte est effectuée avant l'indexation et l'analyse des requêtes .Elle convertit les données textuelles en tokens, et ces tokens sont ajoutés en tant que termes à l'index lucene Lucene offre une grande variété d'analyseurs, tels que SimpleAnalyzer, StandardAnalyzer, StopAnalyzer, SnowballAnalyzer, sans être exhaustif. Ils diffèrent dans leur manière de découper le texte et d'appliquer des filtres. Comme l'analyse supprime des mots avant l'indexation, elle diminue la taille de l'index, mais peut avoir un effet négatif sur la précision du traitement des requêtes.

3. Conclusion

Lucene.net une librairie de recherche Open Source très populaire provenant d'Apache, fournit des fonctions d'indexation et de recherche très puissantes. Il propose une API simple d'utilisation qui ne requiert qu'une compréhension minimale des mécanismes d'indexation et de recherche. Dans cet article, vous avez découvert l'architecture de Lucene et ses APIs de base. Lucene motorise différentes applications de recherche utilisées par de nombreux sites Internet et organisations bien connus. Il a été porté dans d'autres langages de programmation. Lucene bénéficie d'une communauté large et active. Si vous cherchez une librairie de recherche Open Source, facile d'emploi, qui passe à l'échelle, et très performante, Apache Lucene est un excellent choix. [1]

[1] http://www.ibm.com/developerworks/java/library/os-apache-lucenesearch/index.html Using Apache Lucene to search text

CHAPITRE III :

Recherche sémantique

1. Introduction

L'objectif de ce chapitre est de dresser le portrait de l'annotation sémantique .Nous soulignerons tout d'abord les différentes facettes et définitions de l'annotation. Puis, nous étudierons le lien existant entre l'annotation sémantique et les ressources terminologiques ou ontologiques existantes

2. Web sémantique

Ou toile sémantique, est un mouvement collaboratif mené par le World Wide Web Consortium(W3C) qui favorise des méthodes communes pour échanger des données. Le Web sémantique vise à aider l'émergence de nouvelles connaissances en s'appuyant sur les connaissances déjà présentes sur Internet. Pour y parvenir, le Web sémantique met en œuvre le Web des données qui consiste à lier et structurer l'information sur Internet pour accéder simplement à la connaissance qu'elle contient déjà. Selon le W3C, « le Web sémantique fournit un Modèle qui permet aux données d'être partagées et réutilisées entre plusieurs applications, entreprises et groupes d'utilisateurs ». L'expression a été inventée par Tim Berners-Lee, l'inventeur du *World Wide Web* et directeur du World Wide Web Consortium (« W3C »), qui supervise le développement des technologies communes du Web sémantique. Il définit le Web sémantique comme « un web de données qui peuvent être traitées directement et indirectement par des machines pour aider leurs utilisateurs à créer de nouvelles connaissances ». Alors que ses détracteurs ont mis en doute sa faisabilité, ses promoteurs font valoir que les recherches dans l'industrie, la biologie et les sciences humaines ont déjà prouvé la validité du concept original. Les chercheurs ont exploré le potentiel sociétal du web sémantique dans l'industrie et le secteur de la santé. L'article original de Tim Berners-Lee en 2001 dans le *Scientific American* a décrit une évolution attendue du Web existant vers un Web sémantique, mais cela n'a pas encore eu lieu. En 2006, Tim Berners-Lee et ses collègues ont déclaré : « Cette idée simple… reste largement inexploitée. » [[1]]

2.1 Annotation Sémantique

Quelques définitions

Le Petit Robert définit le terme annotation comme une « note critique ou explicative qui accompagne un texte – une note de lecture qu'on inscrit sur un livre ».

Le Dictionnaire Larousse définit le terme annotation comme une. « Action de faire des remarques sur un texte pour l'expliquer ou le commenter »

[1] http://fr.wikipedia.org/wiki/Web_s%C3%A9mantique

Ainsi, le terme annotation réfère à une note, une critique, une explication ou encore à un commentaire. Or, nous rédigeons une note sur un sujet ou bien nous critiquons, expliquons, commentons un sujet. Une annotation seule ne fait pas sens, elle est toujours associée à l'objet qui aété annoté. C'est pourquoi les annotations sont considérées comme des **métadonnées**. Comme le souligne Handschuh [1], si une métadonnée est une donnée sur une donnée, une annotation constitue un cas particulier d'une métadonnée puisqu'elle représente une nouvelle donnée attachée à une ressource documentaire. Prié & Garlatti distinguent une métadonnée comme une description normalisée attachée à une ressource identifiée (sur le Web notamment) et une annotation comme un commentaire libre situé à l'intérieur de la ressource documentaire [2].

Il est important ici de préciser la notion de ressource documentaire : elle peut correspondre à l'ensemble d'un document ou bien seulement à un fragment de celui-ci et contenir du texte, de l'image, du son, de la vidéo ou une combinaison de ces contenus. L'annotation de ressources documentaires est une vieille tradition dans le monde de la documentation et des bibliothèques. La Digital Library Federation (DLF), une association constituée des quinze bibliothèques américaines les plus importantes aux Etats-Unis, a défini trois sortes d'annotations qui peuvent s'appliquer aux ressources documentaires d'une bibliothèque numérique [3]

- L'annotation administrative, ou annotation documentaire, indique les informations associées à la création et à la maintenance de la ressource documentaire telles « qui, quoi, où et comment ». Depuis l'avènement du Web, le langage DublinCore fait office de standard pour l'annotation avec des descripteurs tels que l'auteur, le titre, la source, l'éditeur, la date de publication, la langue, etc. Le DublinCore est un schéma de métadonnées générique qui permet de décrire des ressources numériques ou physiques et d'établir des relations avec d'autres ressources.

Il comprend officiellement 15 éléments de description formels (titre, créateur, éditeur), intellectuels (sujet, description, langue…) et relatifs à la propriété intellectuelle. [4]

[1] HABERT B., Instruments et ressources électroniques pour le français, Collection "L'essentielFrançais", Ophrys, Paris, 2005, 169 p

[2] PRIE Y. & GARLATTI S., Méta-données et annotations dans le Web sémantique, in Le Web Sémantique, CHARLET J., LAUBLET P. & REYNAUD C. (Ed.), Hors série de la Revue Information - Interaction - Intelligence (I3), 4(1), Cépaduès, Toulouse, 2004, pp. 45-68.

[3] HABERT B., Instruments et ressources électroniques pour le français, Collection "L'essentiel Français", Ophrys, Paris, 2005, 169 p.

[4] http://fr.wikipedia.org/wiki/Dublin_Core

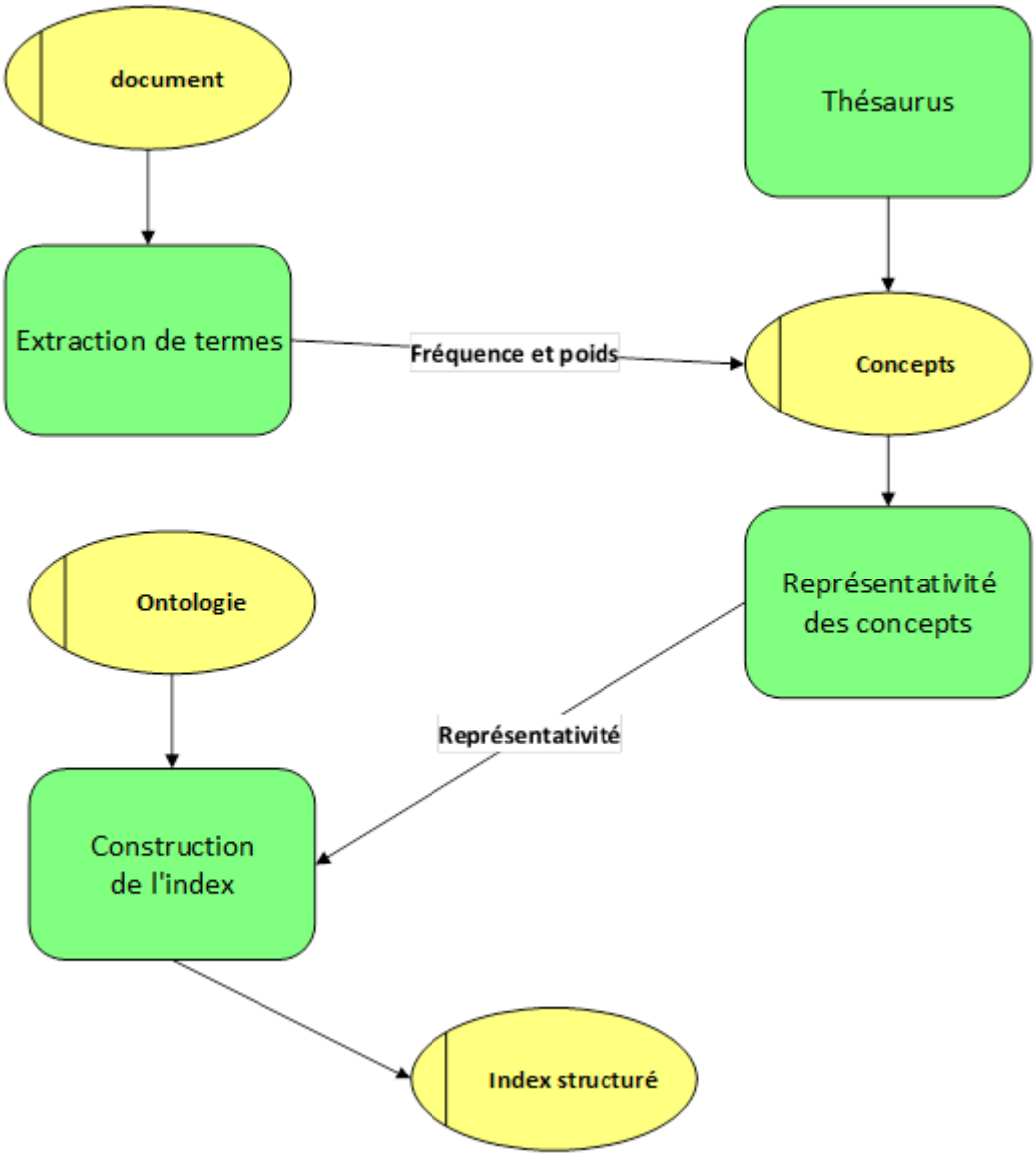
- L’annotation structurelle relie des parties de ressources documentaires entre elles afin de constituer une représentation logique d’un document
- *L’annotation descriptive* décrit une ressource documentaire vis-à-vis de son contenu c'est-à dire qu’elle va dégager les concepts mentionnés dans la ressource documentaire, les relations entre ces concepts ainsi que leurs instances

2.2 L'annotation sémantique en pratique

L'indexation sémantique elle prend en compte la sémantique des mots au travers des relations entre les termes indexés. Il existe plusieurs travaux traitant de l’utilisation de l’indexation sémantique. Nous en avons retenus l'indexation avec une terminologie orientée ontologie que nous avons estimée pertinents afin de montrer l’utilité d’une telle approche.

2.3 L’indexation avec une terminologie orientée ontologie :

Le but est de construire un index associé à une ontologie. Les différentes étapes de ce processus, illustrées dans la figure sont les suivantes :



.Figure 4 : Le processus d'indexation

Pour chaque document, un index est construit. Chaque terme de cet index est associé à sa fréquence pondérée. Ce coefficient dépend au nombre d'occurrences du mot dans le document. Un thésaurus permet de générer tous les concepts candidats qui peuvent être marqués par un terme à partir de l'index de la première étape. Dans notre implémentation, nous utilisons le thésaurus Wordnet. La représentativité de chaque concept candidat dans le document est déterminée. Ce calcul est fait à partir du poids des termes et de leurs relations avec les autres concepts. Cela permet de choisir le meilleur sens du concept dans le document. Plus un concept est proche des autres concepts et plus il est significatif dans le document,

autrement dit : Chaque concept de candidat d'une page est étudiée pour déterminer la représentativité de ce contenu de le document. Cette évaluation est basée sur la fréquence pondérée et sur les relations avec les autres concepts. Il permet de choisir le meilleur sens du terme (concept) d'un terme par rapport au contexte. Par conséquent, plus un concept entretient de solides relations avec d'autres concepts de sa document, plus cette notion est importante dans son document. Cette relation contextuelle minimise le rôle de la fréquence pondérée par le poids croissant des concepts fortement liés et en affaiblissant les concepts isolés (même avec une forte fréquence pondérée). Parmi ces concepts candidats, un filtre est réalisé via l'ontologie et la représentativité des concepts. A savoir, un concept sélectionné est un concept de candidat qui appartient à l'ontologie et a une grande représentativité du contenu de la page.

CHAPITRE IV :

L'Ontologie

1. Introduction

L'exploitation de connaissances en informatique a pour objectif de ne plus faire manipuler en aveugle des informations à la machine mais de permettre un dialogue (une coopération) entre le système et les utilisateurs. Alors, le système doit avoir accès non seulement aux termes utilisés par l'être humain mais aussi à la sémantique qui leur est associée, afin qu'une communication efficace soit possible. Actuellement, la connaissance visée par ces ontologies est un sujet de recherche populaire dans diverses communautés .Elles offrent une connaissance partagée sur un domaine qui peut être échangée entre des personnes et des systèmes hétérogènes. Elles ont été définies en intelligence artificielle afin de faciliter le partage des connaissances et leur réutilisation. La définition explicite du concept ontologie soulève un questionnement qui est tout à la fois d'ordre philosophique, épistémologique, cognitif et technique.

2. Qu'est-ce qu'une ontologie ?

En informatique, une ontologie est un ensemble structuré de concepts. Les concepts sont organisés dans un graphe dont les relations peuvent être :

- des relations sémantiques.
- des relations de composition et d'héritage (au sens objet).

L'objectif premier d'une ontologie est de modéliser un ensemble de connaissances dans un domaine donné. Les ontologies informatiques sont des outils qui permettent précisément de représenter un corpus de connaissances sous une forme utilisable par une machine.

Une des définitions de l'ontologie qui fait autorité est celle de Gruber :

Une ontologie est la spécification d'une conceptualisation d'un domaine de connaissance. Cette définition s'appuie sur deux dimensions :

- Une ontologie est la conceptualisation d'un domaine, c'est-à-dire un choix quant à la manière de décrire un domaine.
- C'est par ailleurs la spécification de cette conceptualisation, c'est-à-dire sa description formelle.

Approche opérationnelle : une autre définition, plus opérationnelle, peut être formulée ainsi :

Une ontologie est un réseau sémantique qui regroupe un ensemble de concepts décrivant complètement un domaine. Ces concepts sont liés les uns aux autres par des relations taxonomiques (hiérarchisation des concepts) d'une part, et sémantiques d'autre part. Cette définition rend possible l'écriture de langages destinés à implémenter des ontologies. [1]

[1] http://fr.wikipedia.org/wiki/Ontologie_(informatique)

Une ontologie définit un vocabulaire commun pour les chercheurs qui ont besoin de partager l'information dans un domaine. Elle inclut des définitions lisibles en machine des concepts de base de ce domaine et de leurs relations.

La littérature d'Intelligence Artificielle contient plusieurs définitions pour une ontologie ; un bon nombre d'entre elles sont même contradictoires. Pour les besoins de ce guide une ontologie est une description formelle explicite des concepts dans un domaine du discours (classes (appelées parfois concepts)), des propriétés de chaque concept décrivant des caractéristiques et attributs du concept (attributs (appelés parfois rôles ou propriétés)) et des restrictions sur les attributs (facettes (appelées parfois restrictions de rôles)). Une ontologie ainsi que l'ensemble des instances individuelles des classes constituent une base de connaissances. Il y a en réalité une frontière subtile qui marque la fin d'une ontologie et le début d'une base de connaissances. Les classes constituent le centre d'intérêt de plusieurs ontologies. Les classes décrivent les concepts dans le domaine.

.En termes pratiques, développer une ontologie inclut :

- définir les classes dans l'ontologie,
- arranger les classes en une hiérarchie taxinomique (sous-classe – super-classe),
- définir les attributs et décrire les valeurs autorisées pour ces attributs
- renseigner les valeurs pour les attributs des instances

Nous pouvons créer une base de connaissances en définissant les instances individuelles de ces classes, en précisant les valeurs spécifiques des attributs ainsi que les restrictions des attributs

2.1 Pour quelles raisons développer une ontologie ?

En voici quelques-unes :

- Partager la compréhension commune de la structure de l'information entre les personnes ou les fabricants de logiciels.
- Permettre la réutilisation du savoir sur un domaine.
- Expliciter ce qui est considéré comme implicite sur un domaine.
- Distinguer le savoir sur un domaine du savoir opérationnel.
- Analyser le savoir sur un domaine

2.2 Cycle de vie d'une ontologie

Il comprend une étape initiale de détection et de spécification des besoins qui permet notamment de circonscrire précisément le domaine de connaissances, une étape de conception qui se subdivise en trois phases, une étape de déploiement et de diffusion, une

étape d'utilisation, une étape incontournable, d'évaluation, et enfin, une sixième étape consacrée à l'évolution et à la maintenance du modèle. Après chaque utilisation significative, l'ontologie et les besoins doivent être réévalués et l'ontologie peut être étendue et, si nécessaire, en partie reconstruite. La validation du modèle de connaissances est au centre du processus et se fait de manière itérative. Nous insiste sur le fait que les activités de documentation et d'évaluation sont nécessaires à chaque étape du processus de construction, l'évaluation précoce permettant de limiter la propagation d'erreurs. Le processus de construction peut et doit être intégré au cycle de vie d'une ontologie comme indiqué en figure si dessous.

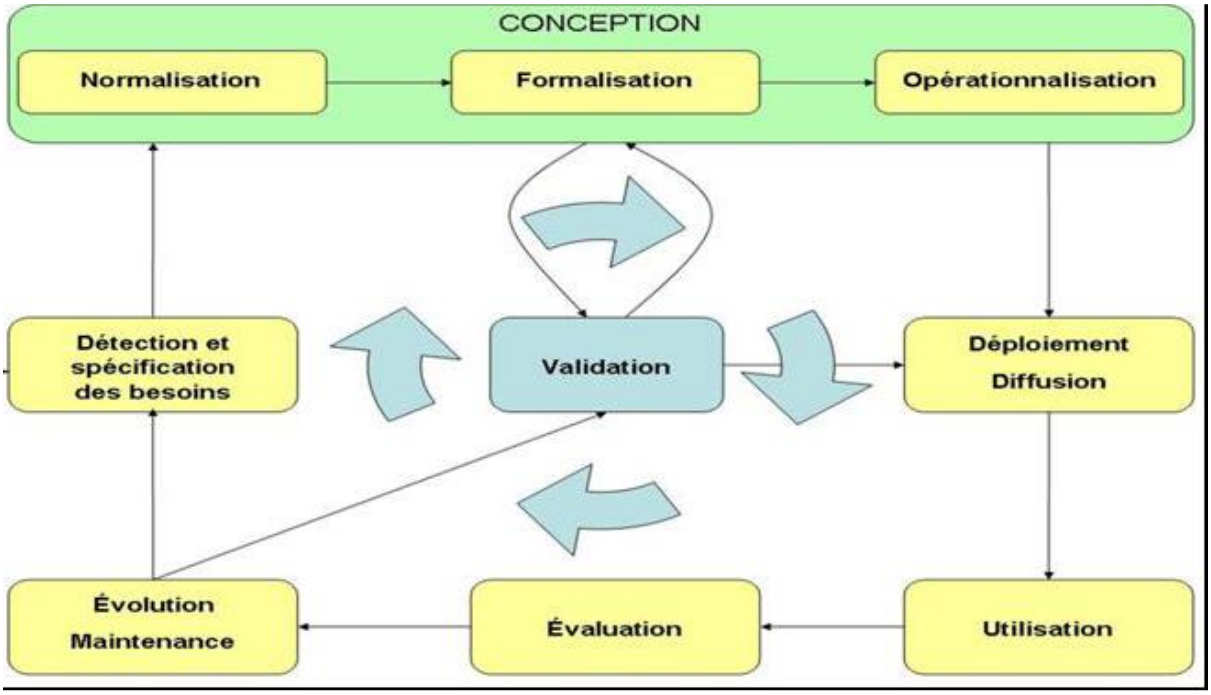


Fig :5 Cycle de vie d'une ontologie.

3. Composantes d'une ontologie

On peut caractériser une ontologie comme une structuration des concepts d'un domaine. Ces concepts sont rassemblés pour fournir les briques élémentaires et exprimer les connaissances dont on dispose dans ce domaine. Comme nous l'avons abordé, les ontologies fournissent un vocabulaire commun d'un domaine et définissent la signification des termes et des relations entre elles. La connaissance dans les ontologies est principalement formalisée en

utilisant les cinq types de composants à savoir : concepts (ou classes), relations (ou propriétés), fonctions, axiomes (ou règles) et instances (ou individus).

- Les concepts, aussi appelés termes ou classe de l'ontologie, correspondent aux abstractions pertinentes d'un segment de la réalité (le domaine du problème) retenus en fonction des objectifs qu'on se donne et de l'application envisagée pour l'ontologie;
- Les relations traduisent les associations (pertinentes) existant entre les concepts présents dans le segment analysé de la réalité. Ces relations incluent les associations suivantes :
    - Sous classes de (généralisation-spécialisation) ;
    - Partie de (agrégation ou composition) ;
    - Associe à ;
    - Instance de, etc.

Ces relations nous permettent d'apercevoir la structuration et l'interrelation des concepts, les uns par rapport aux autres ;

- Les fonctions constituent des cas particuliers de relations, dans laquelle un élément de la relation, (le nième) est défini en fonction des N-1 éléments précédents ;
- Les axiomes constituent des assertions, acceptées comme vraies, à propos des abstractions du domaine traduites par l'ontologie.
- Les instances constituant la définition extensionnelle de l'ontologie ; ces objets véhiculent les connaissances (statiques, factuelles) à propos du domaine du problème.

4. La méthodologie de construction d'une ontologie :

On entend par méthodologie, les procédures de travail, les étapes, qui décrivent le pourquoi et le comment de la conceptualisation puis de l'artefact construit. L'ingénierie ontologique ne propose à l'heure actuelle, aucune méthode normalisée ou méthodologie générale de construction d'ontologies, ce qui rend le processus d'élaboration des ontologies long et coûteux. Cependant certains auteurs ont proposé des méthodologies inspirées de leur expérience de construction d'ontologies **[17] [22].** Ces méthodologies proposent à travers un ensemble d'étapes, un cycle de développement d'ontologies qui peut être adopté lors de la construction d'une nouvelle ontologie. Quelque soit la méthodologie adoptée, Le processus de construction d'ontologies doit respecter certains principes de base qui permettent d'obtenir

une ontologie susceptible de répondre aux objectifs de l'ontologie. Gruber **[9],** présente certains critères pour la construction d'un projet d'ontologie :

La clarté : La définition d'un concept doit faire passer le sens voulu du terme, de manière aussi objective et complète que possible.

La cohérence : Une ontologie cohérente doit permettre des inférences conformes à ces définitions.

L'extensibilité : Il doit être possible d'ajouter de nouveaux concepts sans avoir à toucher aux fondations de l'ontologie.

Modularité : Ce principe vise à minimiser les couplages entre les modules.

Une déformation d'encodage minimale : l'ontologie doit être conceptualisée indépendamment de tout langage d'implémentation. Le but est de permettre le partage des connaissances contenues dans l'ontologie, entre différentes applications utilisant différents langages de représentation.

Un engagement ontologique minimal : L'ontologie devrait spécifier le moins possible la signification de ses termes, donnant aux parties (qui partager la connaissance) qui s'engagent dans cette ontologie la liberté de spécialiser et d'instancier l'ontologie comme elles le désirent.

Nous citons à titre d'exemple la méthode proposée par l'université de Stanford :

La méthode développée par l'université de Stanford :

Cette méthode comporte sept étapes qui sont les suivantes :

Etape 1 : déterminer le domaine et la portée de l'ontologie :

Cette étape se fait en répondant aux questions ci-dessous tout au long de la conception de l'ontologie et qui aident à définir la portée du domaine de l'ontologie :

Quel est le domaine que va couvrir l'ontologie ?

Dans quel but utiliserons-nous l'ontologie ?

A quels types de questions l'ontologie devra-t-elle fournir des réponses ?

Qui va utiliser et maintenir l'ontologie ?

Etape 2 : envisager une éventuelle réutilisation des ontologies existantes :

Dans tout domaine de recherche, il est utile de profiter de ce que les autres ont fait afin d'en tirer les informations et ainsi permettre d'élargir le travail et l'affiner pour répondre à nos propres besoins.

Il peut être intéressant d'importer des ontologies déjà existantes (dans le même domaine) et les raffiner et les perfectionner pour aboutir à une ontologie plus complète et étendue.

Etape 3 : énumérer les termes importants dans l'ontologie :

Il est important d'établir en premier lieu une liste complète des mots et termes concernant le domaine d'intérêt, et cela sans se soucier de la catégorisation de ces derniers dans des classes, hiérarchie, chevauchement,… etc. les questions à se poser pour établir cette liste sont :

Sur quels termes souhaiterons-nous discuter ?

Quelles sont les propriétés de ces termes ?

Que veut-on dire sur ces termes ?

Etape 4 : définir les classes et la hiérarchie des classes :

A partir de la liste de l'étape précédente, on commence par définir les classes en sélectionnant les termes qui décrivent des objets ayant une existence indépendante. Ce sont ces termes qui constitueront les classes (appelées parfois concepts) de l'ontologie. Il faut ensuite organiser ces classes dans une taxonomie hiérarchique.

Etape 5 : définir les propriétés des classes (les attributs ou rôles) :

Dans cette étape, on devra décrire la structure interne des concepts tires pendant l'étape précédente. Les propriétés définissent la structure interne et les caractéristiques des classes.

Etape 6 : définir les facettes des attributs :

Les attributs peuvent avoir plusieurs facettes (appelées parfois restrictions de rôles). Les facettes les plus communes décrivent :

Le type de valeur des attributs : désigne le type de valeur pouvant être affecte a un attribut. Les plus typiques sont les suivants : chaine de caractère, nombre ou entier, booléen…

Le nombre de valeurs ou cardinalité : désigne le nombre de valeurs qu'un attribut peut avoir. Une cardinalité peut être unique ou multiple. Il est utile de spécifier pour un attribut une cardinalité minimale et une cardinalité maximale. Le rang et le domaine d'un attribut : l'étendue ou le rang d'un attribut représente les classes autorisées pour les attributs de type « Instance ». Le domaine d'un attribut représente les classes auxquelles cet attribut est rattache ou les classes dont l'attribut décrit les propriétés.

Etape 7 : créer les instances :

Cette étape consiste à créer les instances qui représentent des entités

PARTIE III :

Conception, Réalisation

CHAPITRE I: Conception

1. Un aperçu du système :

1.1 Diagramme des cas d'utilisation

Nous pouvons dès ce stade de l'analyse représenter le diagramme des cas d'utilisation **Figure 6 :**

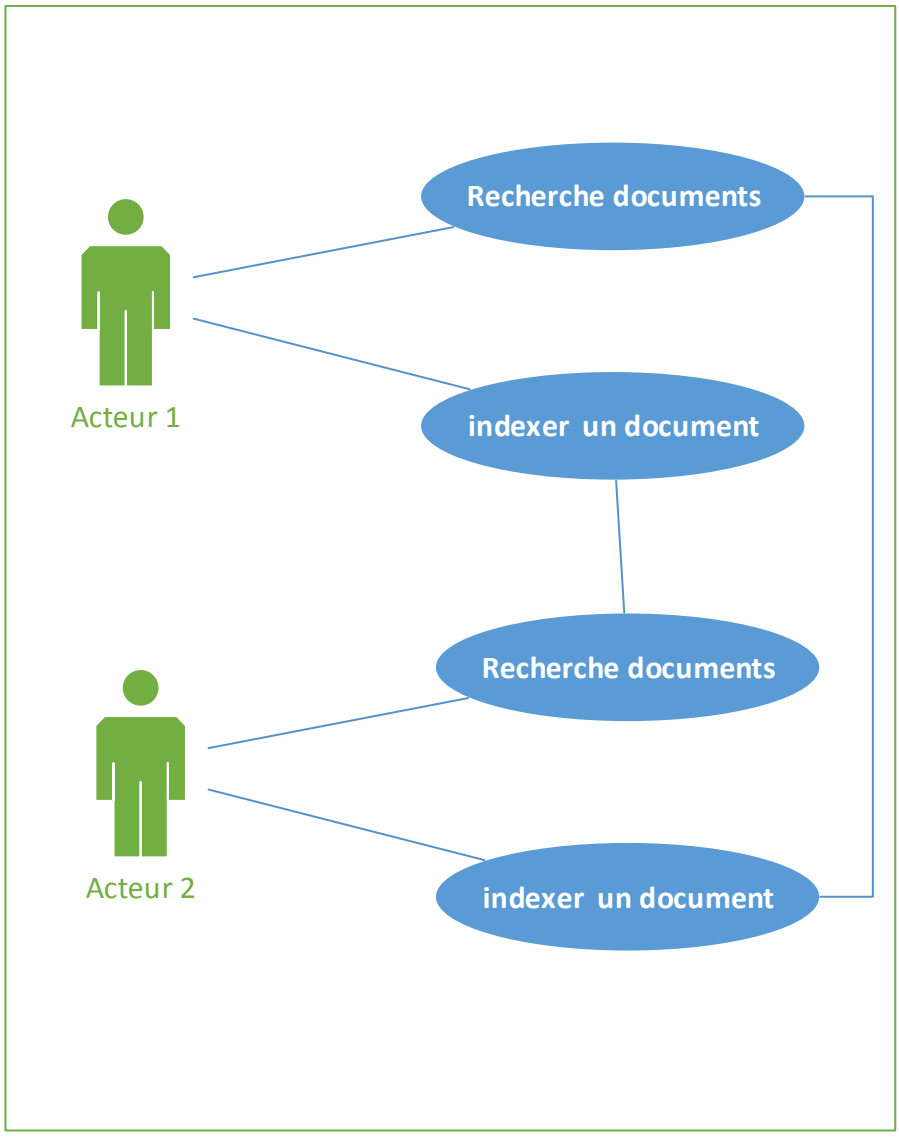


Figure 6: diagramme des cas d'utilisation

2. Description du système :

2.1 Diagramme de séquence :

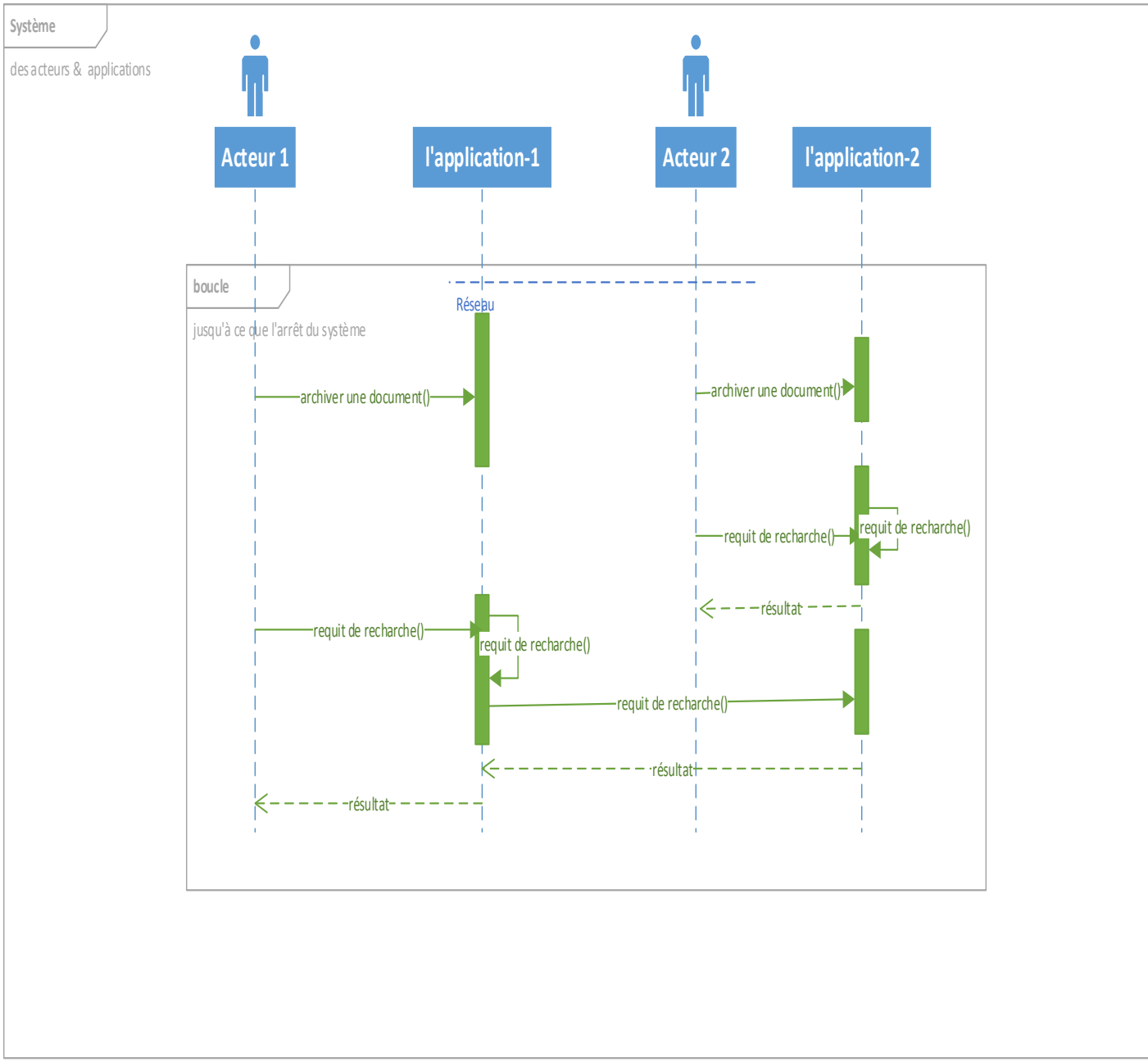


Fig 7: Diagramme de séquence

2.2 Diagramme d'activité

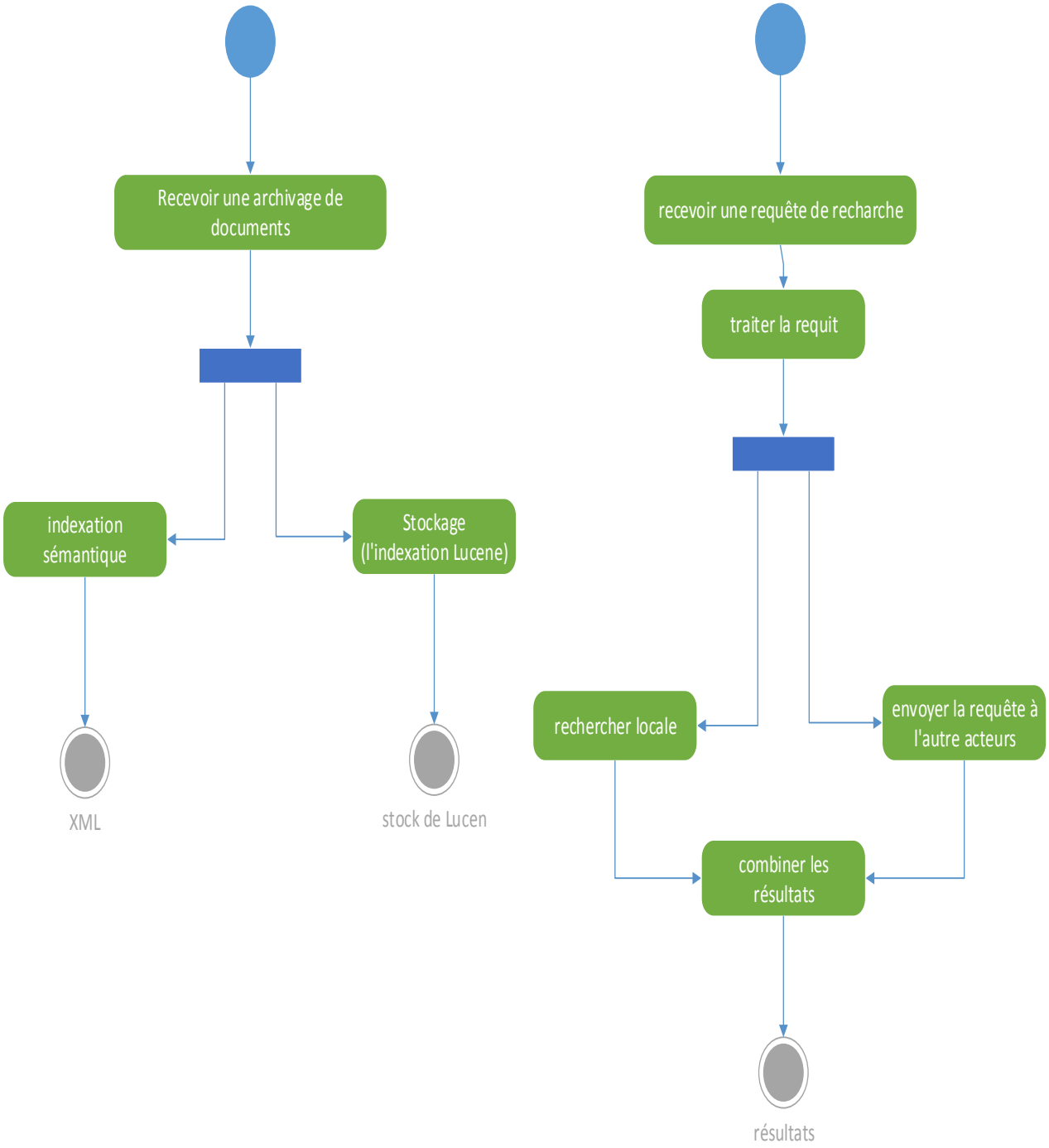


Fig8 : diagramme d'activité

2.3 Diagramme de classe :

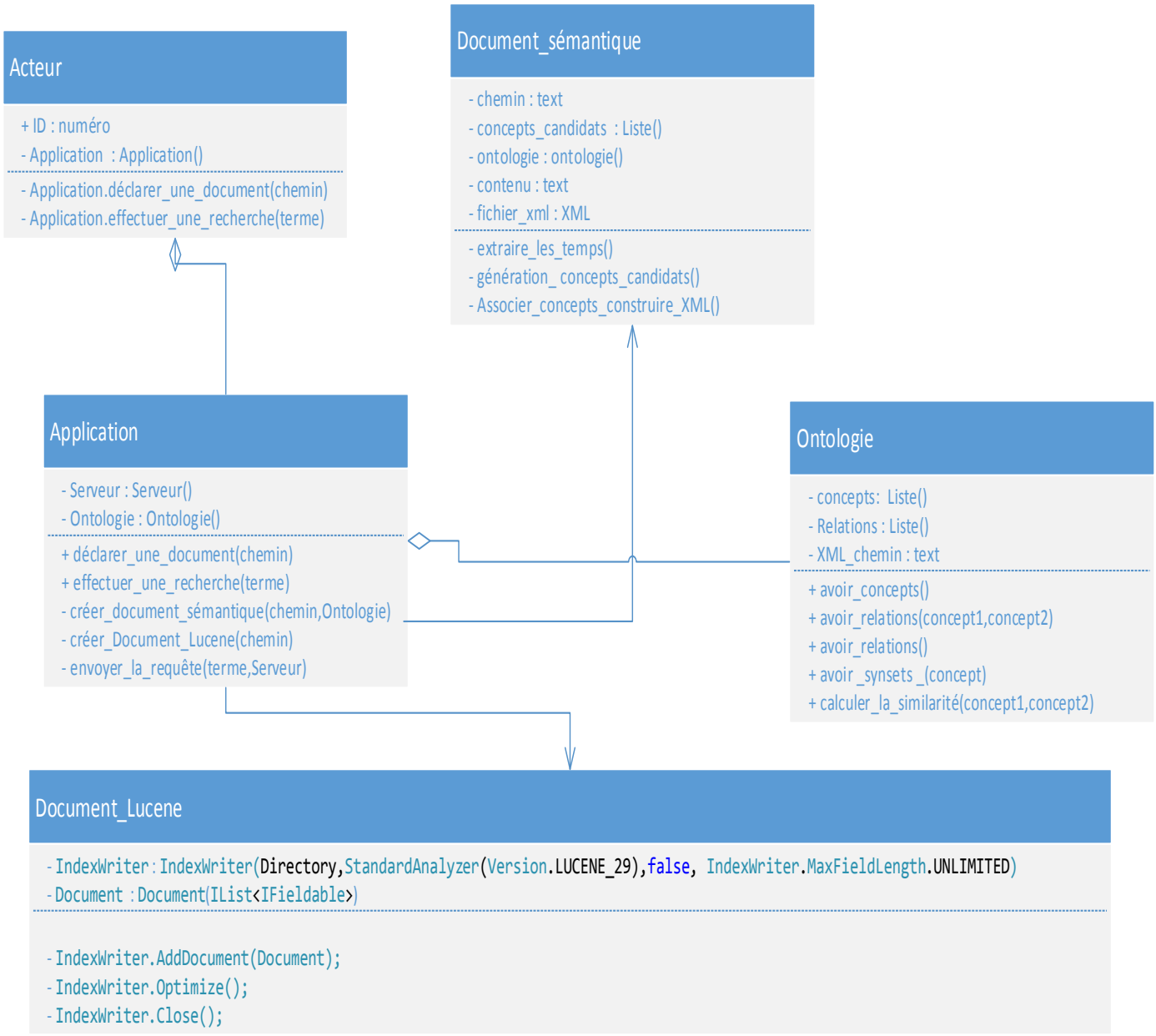


Fig 9: Diagramme de classe

3 Description des éléments de la modélisation :

3.1 Diagramme De Paquetage :

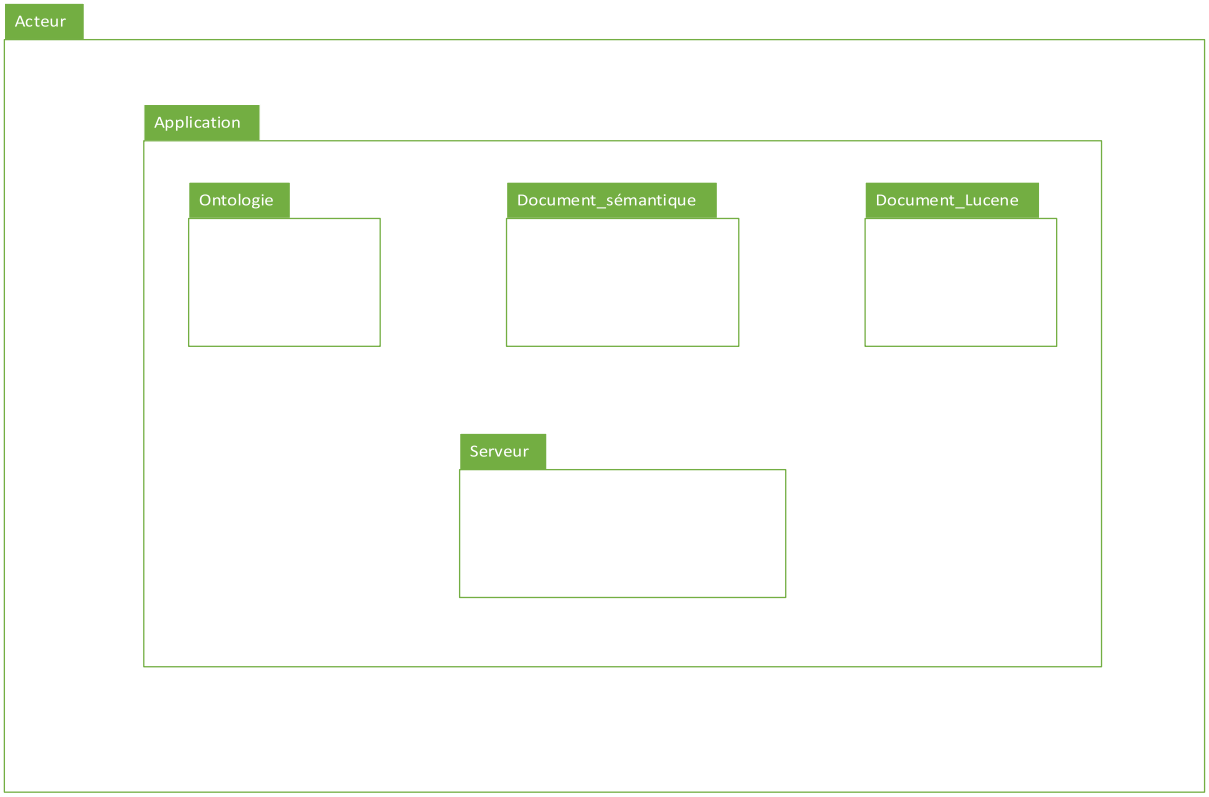


Figure 10: Diagramme De Paquetage

4 Conclusion

Comme on a déjà expliqué comment utiliser le Lucene, alors dans les chapitres suivants nous allons nous concentré sur l'ontologie et le document sémantiques donc le travail se compose de deux phases principales **Figure 10**:

- la phase d'annotation : qui a pour but de représenter au mieux le contenu du document.
- la phase de recherche : qui consiste à restituer à un utilisateur les réponses les plus pertinentes par rapport à sa requête, en utilisant l'annotation des documents.

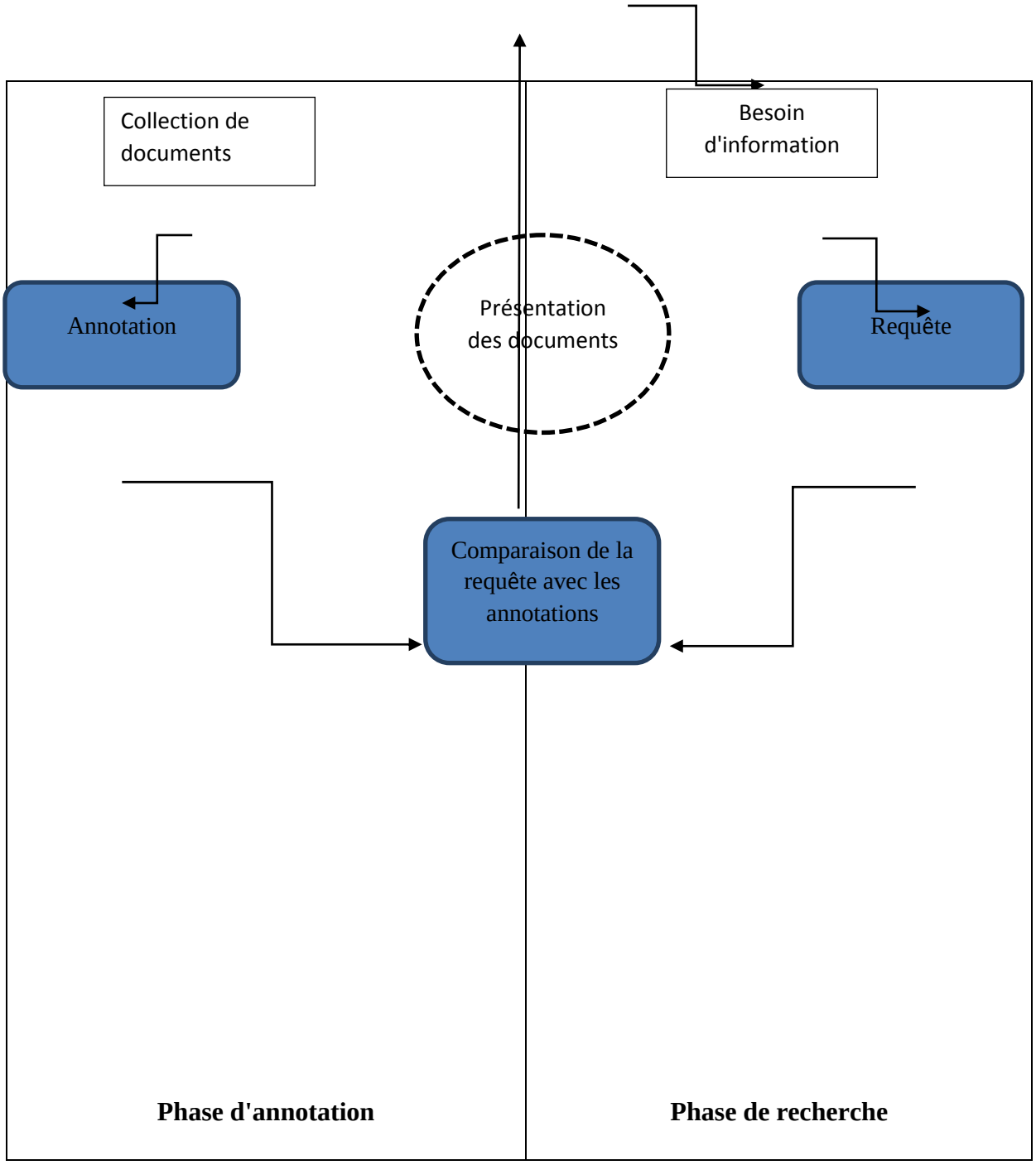


Figure 11: Les phases dans les processus l'annotation et recherche d'informations.

CHAPITRE II :

La construction de l'ontologie

1. **Etape 1 : la définition de domaine** :
   a. Le but de l'utilisation de l'ontologie par l'utilisateur est de définir ses différents domaines ainsi de les lui regrouper
   b. Quant à la construction de l'ontologie, la spécification de domaine lui impose de répondre de questions types
      i. le type de ce concept.
      ii. ancêtre de ce concept.
      iii. Les synsets (significations) de concept.[1]

2. **Etape 2 : Envisager une éventuelle réutilisation des ontologies existantes** :

Notre système a besoin d'interagir avec des ontologies spécifiques des vocabulaires contrôlés, pour ces question il y a une ontologie déjà disponible sous forme électronique et peut être importée dans notre propre environnement, WordNet « *WordNet est une base de données lexicale développée par des linguistes du laboratoire des sciences cognitives de l'université de Princeton depuis une vingtaine d'année1. Son but est de répertorier, classifier et mettre en relation de diverses manières le contenu sémantique et lexical de la langue anglaise* »[2].

3. **Etape 3 : Enumérer les termes importants dans l'ontologie :**

Les termes importants de notre ontologie sont énumérés dans l'ontologie de WordNet : WordNet « *offre pour chaque mot, une liste de synsets correspondant à toutes ses acceptions répertoriées. Mais les synsets ont également d'autres usages : ils peuvent représenter des concepts plus abstraits, de plus haut niveau que les mots, qu'on peut organiser sous forme d'ontologies* »[3]. En bref, le cœur d'un concept dans l'ontologie de WordNet ensemble est un de mots et leur sens [Figure 12]

[1] http://fr.wikipedia.org/wiki/WordNet#Les_synsets
[2] http://wordnet.princeton.edu/wordnet/
[3] http://fr.wikipedia.org/wiki/WordNet#Les_ontologies_et_les_relations_s.C3.A9mantiques

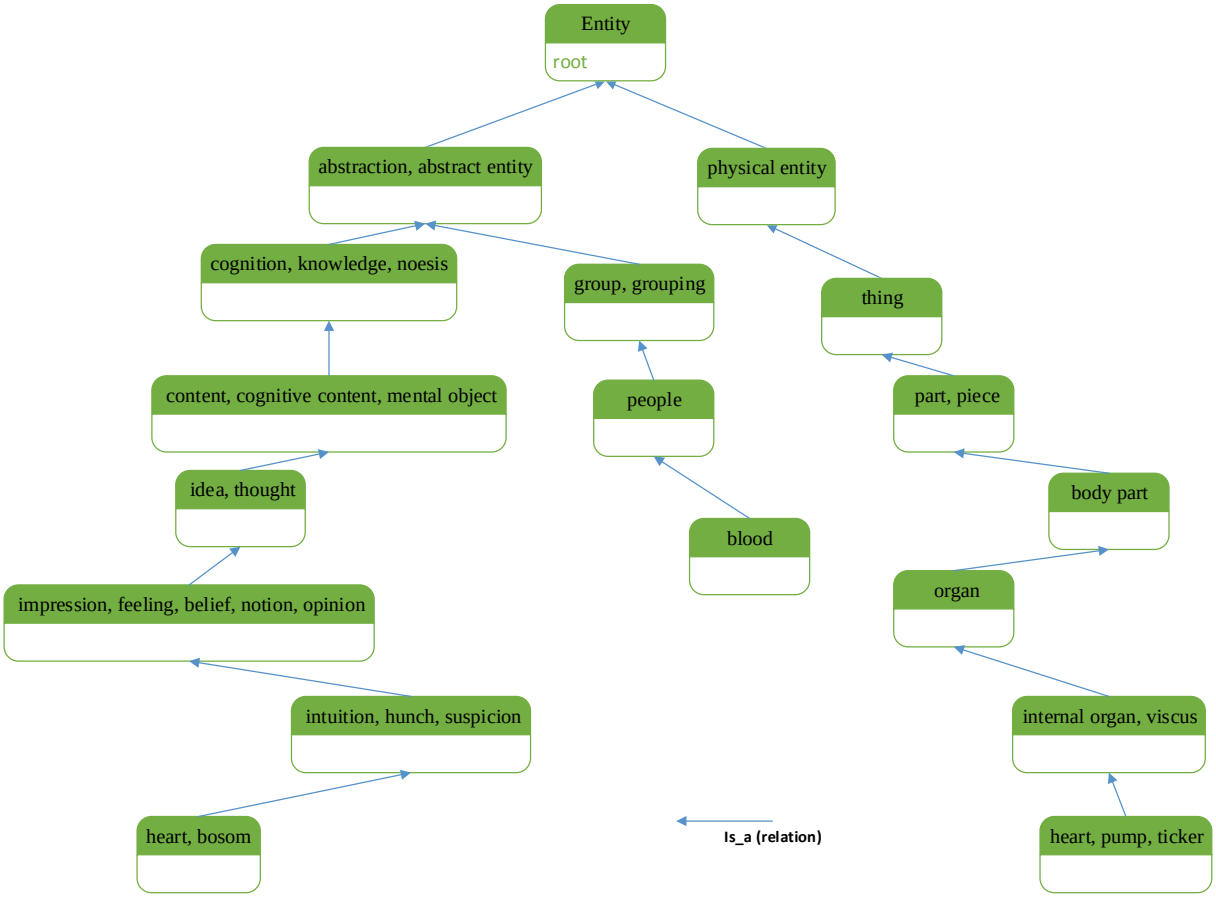


Figure 12 : Un exemple présent les concepts dans l'ontologie de WordNet.

Dans l'exemple suivant : On veut chercher le mot «heart (cœur)» on trouve deux sens 'heart, bosom' et 'heart, pump, ticker' : dans le premier sens le mot 'heart' avec son synset(bosom) donne le sens de sentiment de mot 'heart' par contre, dans le deuxième sens le mot 'heart' avec son synsets (pump, ticker) donne le mot qui est un organe interne dans un corps humain.

4. Définir les classes et la hiérarchie des classes :

Concept	Description
Entity	le concept Root de l'ontologie et le subsumé de tous les concepts.
abstraction, abstract entity	un concept général subsumé des concepts formé par des concepts abstraits.
cognition, knowledge, noesis	un concept général subsumé des concepts de

	résultat psychologique de la perception et de l'apprentissage et le raisonnement
……..	……

Tableau 3 : Les classes de l'ontologie

Les relations :

La relation hyperonyme (IS_A) [1] entre les synsets peut être interprétée comme des relations de spécialisation entre les catégories conceptuelles. La relation d'hyperonymie définit un arbre de concepts de plus en plus généraux, selon l'exemple précédent:

heart, pump, ticker
=> internal organ, viscus
=> organ
=> body part
=> part, piece
=> thing
=> physical entity
=> entity

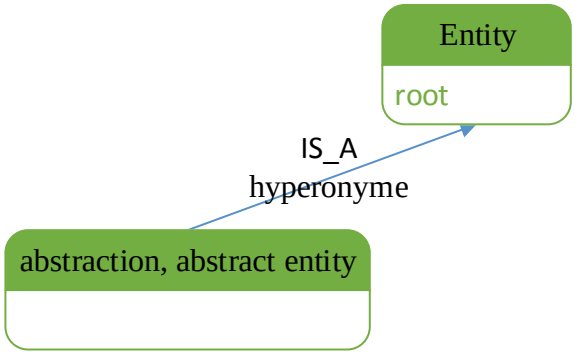


Figure 13 : Un exemple de la relation d'hyperonymie

[1] http://fr.wikipedia.org/wiki/Hyperonymie

CHAPITRE III :

L'annotation sémantique

L'annotation sémantique (l'indexation sémantique) :

L'indexation sémantique prend compte la sémantique des mots à travers des relations entre les termes indexés.

Il existe plusieurs travaux traitant l'utilisation de l'indexation sémantique.

Nous en avons retenus l'indexation avec une terminologie orientée ontologie que nous avons estimée pertinents afin de montrer l'utilité d'une telle approche.

1. L'indexation avec une terminologie orientée ontologie :

Le but est de construire un index associé à une ontologie. Les différentes étapes de ce processus, illustrées dans la figure 4 :

- Pour chaque document, un index est construit. Chaque terme de cet index est associé à sa fréquence pondérée. Ce coefficient dépend au nombre d'occurrences du mot dans le document avec * 4/3 si le terme est un composants des noms du document.
- un thésaurus permet de générer tous les concepts et les significations candidats qui peuvent être marqués par un terme à partir de l'index de la première étape. Dans notre implémentation, nous utilisons le thésaurus Wordnet.
- la représentativité de chaque concept candidat dans le document est déterminée par le calcul qui à partir du poids des termes et de leurs relations avec les autres concepts. Cela permet de choisir le meilleur sens du concept dans le document. Plus un concept est proche des autres concepts et plus il est significatif dans le document, autrement dit : Chaque concept de candidat d'une page est étudiée pour déterminer la représentativité de ce contenu du document. Cette évaluation est basée sur la fréquence pondérée et sur les relations avec les autres concepts. Il permet de choisir le meilleur sens du terme (concept) par rapport au contexte.

Par conséquent, plus un concept entretient de solides relations avec d'autres concepts du document, plus cette notion est importante dans ce document. Cette relation contextuelle minimise le rôle de la fréquence pondérée par le poids croissant des concepts fortement liés et en affaiblissant les concepts isolés (même avec une forte fréquence pondérée).

- Parmi ces concepts candidats, un filtre est réalisé via l'ontologie et la représentativité des concepts. A savoir, un concept sélectionné est un concept

candidat qui appartient à l'ontologie et a une grande représentativité du contenu de la page.

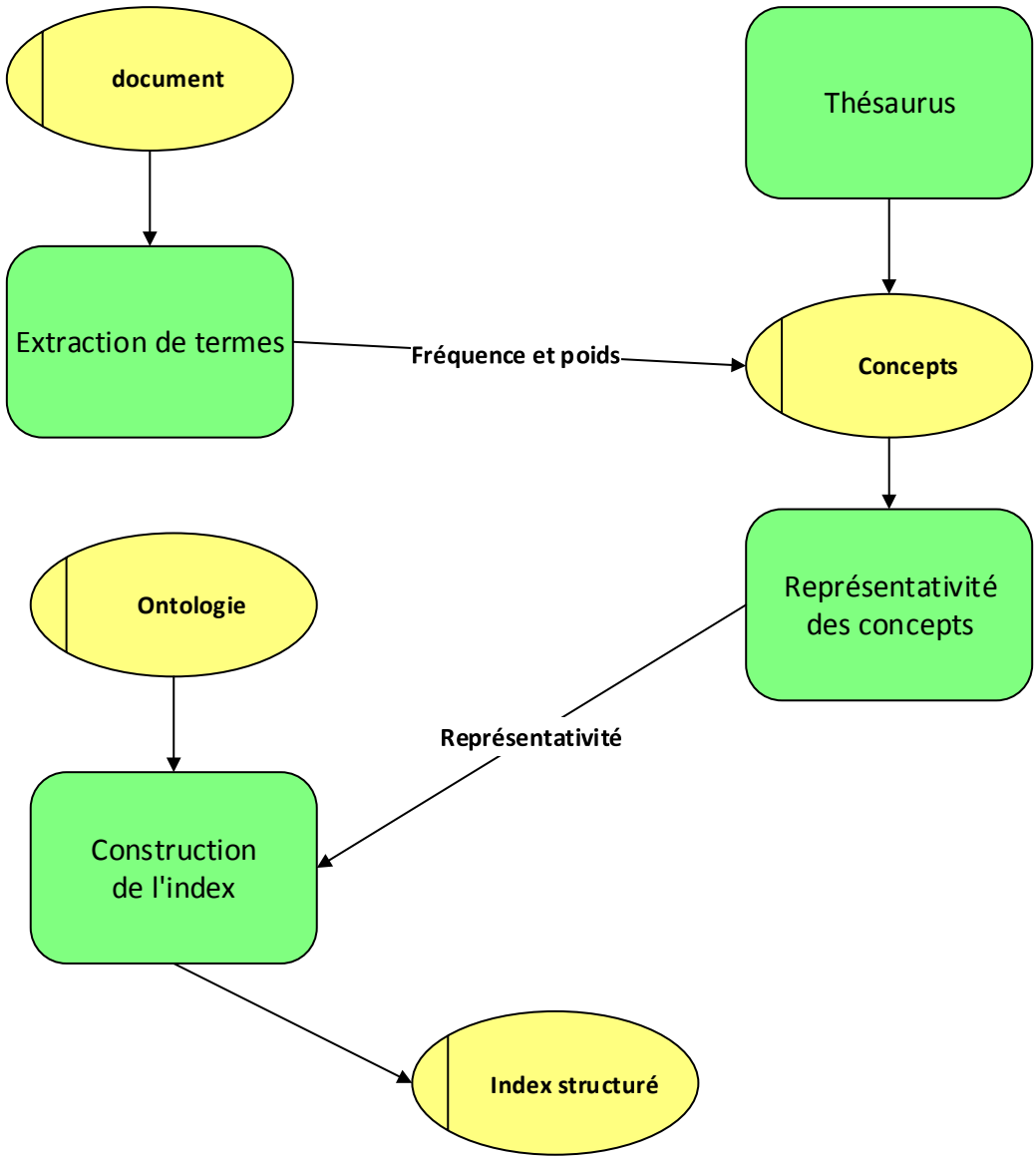


Figure 4 : Le processus d'indexation

3. Extraction de termes :

La première chose à faire est de charger tous les mots qui existent dans le document en suit :

- l'élimination de tous les caractères spéciaux ainsi que les chiffres.
- l'élimination de les conjonctions, les déterminants, les prépositions, les pronoms : ils sont d'aucune utilité dans cette étape, le résultat sera une matrice M constitué de mots pas forcément différents.

$$M = \begin{pmatrix} Blue \\ Cardiology \\ System \\ heart \\ Blood \\ flow \\ diagram \\ human \\ heart \\ Blue \\ components \\ indicate \\ oxygenated \\ blood \\ . \\ . \\ . \end{pmatrix}$$

- calcul de la fréquence : premièrement nous allons calculer le nombre d'occurrences de chaque mot par la formule indiquée (1), après on calcule sa fréquence par la formule (2).

(1) $oc(m_i) = \sum_{i=1}^{y} 1 \times \alpha \qquad \begin{cases} \propto = 1 \; si \; m_i = f(i) \\ \propto = 0 \; si \; m_i \neq f(i) \\ m_i \in M \end{cases} \qquad f(i) = m \begin{cases} \forall \; i = 1..y \\ m \in M \end{cases}$ f(i) retourne l'élément **m** par son index **i** dans la matrice **M,** le résultat sera une matrice **OC** de nombre **n* 2** des mots avec leurs occurrences :

$$OC = \left(\begin{array}{c|c} Blue & 3 \\ Cardiology & 29 \\ System & 10 \\ heart & 113 \\ Blood & 19 \\ flow & 1 \\ diagram & 1 \\ components & 2 \\ . & . \\ . & . \end{array}\right)$$

$Tit = \begin{pmatrix} Cardiology \\ definition \end{pmatrix}$ ***Tit*** matrice du titre de document

(2) $FP(m_i) = \frac{oc(m_i)}{n} \times \propto \begin{cases} \propto = \frac{4}{3} \; si \; m_i \in Tit \\ \propto = 1 \; si \; m_i \notin Tit \\ i = 1..n \end{cases}$ FP : fréquence pondérée

Mot	**fréquence pondérée**

Blue	0,04534314
Cardiology	0,06219363
System	0.01225490
Heart	0,13480390
.	.

Table 4 : Termes extraits et leur fréquence pondérée.

4. génération des significations et les concepts candidats

Au cours du processus d'extraction des mots bien formés et le calcul de leur fréquence pondérée. Les mots bien formés dans les différentes formes représentent un concept particulier. Le processus pour générer des concepts candidats est assez simple : à partir des mots extraits initialement nous représentons les concepts candidats, toutes les significations sont générées à l'aide d'un thésaurus, nos expériences utilisent le thésaurus WordNet. Un concept est représenté par une liste des significations (cette liste est unique pour un concept donné).

Heart	
Signification – 1	heart, bosom
Signification – 2	heart, pump, ticker
Signification – 3	center, centre, middle, heart, eye
Signification – 4	kernel, substance, core, center, centre, essence, gist, heart, heart and soul, inwardness, marrow, meat, nub, pith, sum, nitty-gritty
Signification – 5	affection, affectionateness, fondness, tenderness, heart, warmness, warmheartedness, philia

Table 5: exemple de terme et ses significations

Ensuite, pour chaque concept candidat, la représentativité est calculée en fonction de la fréquence pondérée et de la similarité cumulée du concept avec les autres concepts dans le document. Cette dernière est basée sur la similarité entre deux concepts.

Nous définissons d'abord la mesure de similarité entre deux concepts qui permet d'évaluer la distance sémantique entre ces deux concepts.

Dans notre contexte, nous utilisons la mesure de similarité défini par [20], Ils proposent une mesure de similarité liée à la distance des arêtes de la façon dont il prend en compte le subsumer la plus précise des deux concepts. Sa mesure est représentée dans la formule (3).

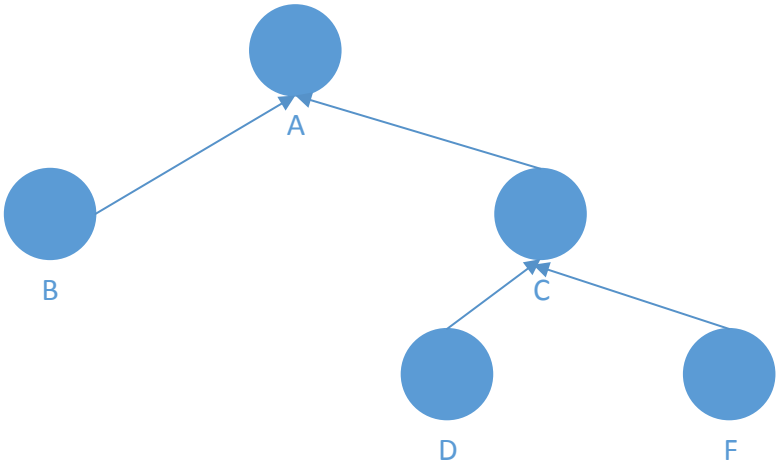


Figure 14 : A c'est le subsumer la plus précise du concept B et F

Où **C** est la subsumer plus spécifique de c_1 et c_2, $depth(c)$ est le nombre d'arêtes de **c** à la racine de la taxonomie, et $depth_c(c_i)$ avec $i\ dans\ \{1,2\}$ c'est le nombre d'arêtes de c_i à la racine de la taxonomie à travers c :

$$(3) \quad sim(c_1, c_2) = \frac{2 \times depth(c)}{depth_c(c_1) + depth_c(c_2)}$$

Pour évaluer l'importance relative d'un concept dans un document, nous définissons sa ressemblance cumulative. La mesure de similarité cumulée associée à un concept dans un document, c'est la somme de toutes les mesures de similarité calculées entre ce concept et de tous les autres concepts inclus dans le document étudié. Dans cette formule, un concept spécifique est unifié avec les significations correspondant (ensemble de significations) dans

[20] Z. Wu and M. Palmer, "verb semantics and lexical selection", In Proceedings of the 32nd annual meeting of the association for computational linguistics, Las Cruses, New Mexico, 1994

WordNet. La mesure est représentée dans la formule (4) où l_k sont associées à un concept C_k , et il existe m concepts dans le document étudié.

(4) $\widehat{sim}(synsets_i(C_k)) = \sum_{j\in[1,k-1]\cup[k+1,m]} \sum_{l=1}^{l_j} sim(synsets_i(C_k), synsets_i(C_j))$

Dans ce calcul, les similitudes n'ont pas été prises en compte afin de discriminer les résultats mais un seuil est appliqué. Enfin, nous déterminons un coefficient de représentativité qui détermine la représentativité d'un concept dans un document. Le coefficient est une combinaison linéaire pondérée de la fréquence et de la similarité cumulée d'un concept (formule (5)). Ce coefficient est le plus important pour qualifier la réponse à une demande.

(5) $représentativité(synsets_i(C_k)) = \frac{2 \times FP(synsets_i(C_k)) + \widehat{sim}(synsets_i(C_k))}{3}$

FP : Fréquence pondérée

5. Associer les concepts:

Dans la prochaine étape, les concepts sont évalués pour un seuil appliqué sur la représentativité. Si un concept est supérieur au seuil de 0.5, le chemin de ce document et de sa représentativité est ajouté à l'index sémantique.

La figure ci-dessous illustre la méthodologie d'annotation l'indexation avec une terminologie orientée ontologie:

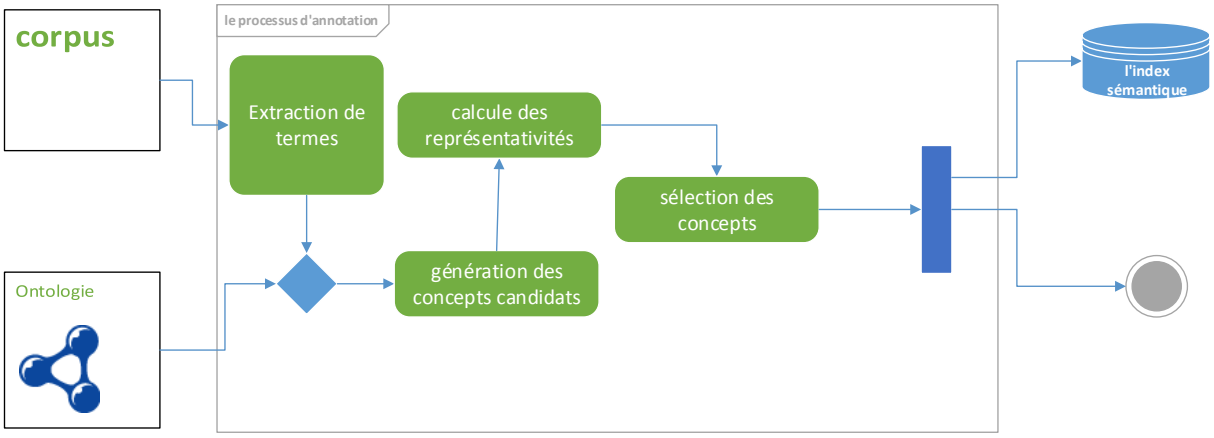


Figure 15 : le processus d'annotation

CHAPITRE IV :

La recherche

Introduction

Améliorer la recherche en interprétant les requêtes syntaxiquement est une idée répandue. Il existe beaucoup de façons pour ajouter la sémantique a la recherche, et aucune approche n'a encore fait ses preuves en tant que c'est la solution.

Dans cette étape nous allons essayer d'exploiter les index construit (l'index de Lucene et l'index sémantique) dans chapitre précédent **Figure 16**.

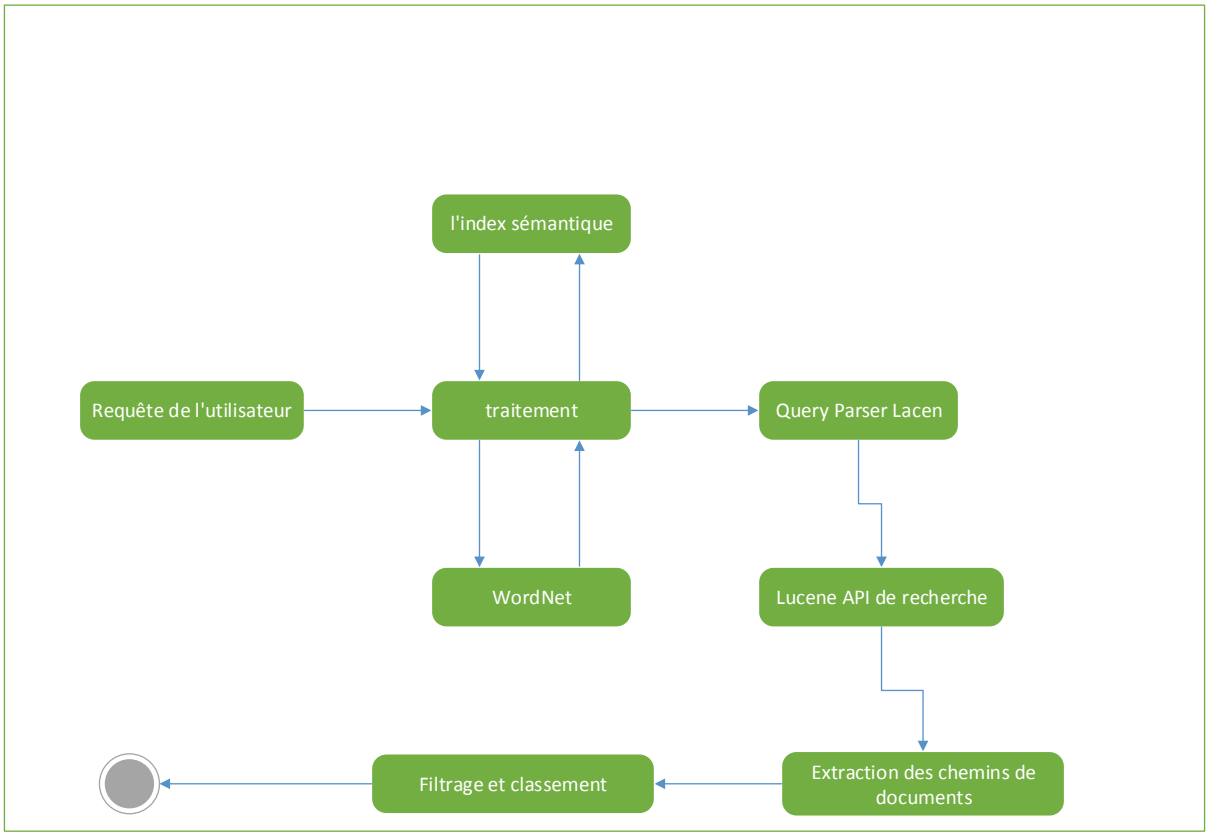


Figure 16 : Architecture de recherche

Recherche par mots clés a ses limites .Par exemple, considérez la requête " which animals belong to the desert ". Cela nécessite une recherche pas pour le mot «animals » littérale, mais plutôt pour les instances de la classe qu'il représente.

Déjà cette simple requête met en évidence deux des principaux défis de la recherche sémantique : (1) obtenir l'information sémantique nécessaire, dans ce cas, d'identifier chaque animal du désert à notre collection de documents et ici, nous allons montrer l'utilité de l'index sémantique, et (2) rendre cette information recherché dans un moyen pratique et efficace.

1. l'identification de l'information sémantique nécessaire :

1.1 Affiner la forme des requêtes :

La requête de l'utilisateur est raffinée pour donner un meilleur résultat de la recherche. La requête est analysée d'où sont extrait les mots de la requête. Les mots-clés du domaine qui sont sémantiquement liés à la demande de recherche sont extraits de l'ontologie existante. Cette étape se traduit par la récupération de plus de nombre de mots sémantiquement liés. Ces mots-clés de domaine sont ensuite utilisés pour la former la requêtes raffinés. Donc il en résulte des requêtes raffinés avec des mots-clés et qui ont plus de pertinence sémantique. Pour la requête de l'exemple précédent, après l'application de cette étape le requête sera reformer comme ça « which animals animal animate being beast brute creature fauna belong to the desert ».

Cette étape dépend à l'extraction des Synsets de mots.

Mot	Synsets
Which	
animals	animal, animate being, beast, brute, creature, fauna
belong	
desert	desert

Table 6 : les Synsets de mots[1]

2. Rendre cette information recherchée dans un moyen pratique et efficace :

2.1 Reconnaissance du Concept :

Nous allons utiliser le thésaurus. Notre ontologie est composé par des concepts qui est un ensemble de Synsets, alors l'idée est simple, un mot a un Synsets c'est déjà un concept dans l'ontologie, nous continuons avec notre

[1] L'extraction de Synsets faite par WordNet (Line de Command « **wn desert – synsn** »)

exemple : les concepts = {(animal, animate being, beast, brute, creature, fauna), (desert)}

2.1 Concepts supporté par la requête :

Les Concepts supporté ont tous les concepts produits par la relation hyponym appliquée sur les concepts reconnus.

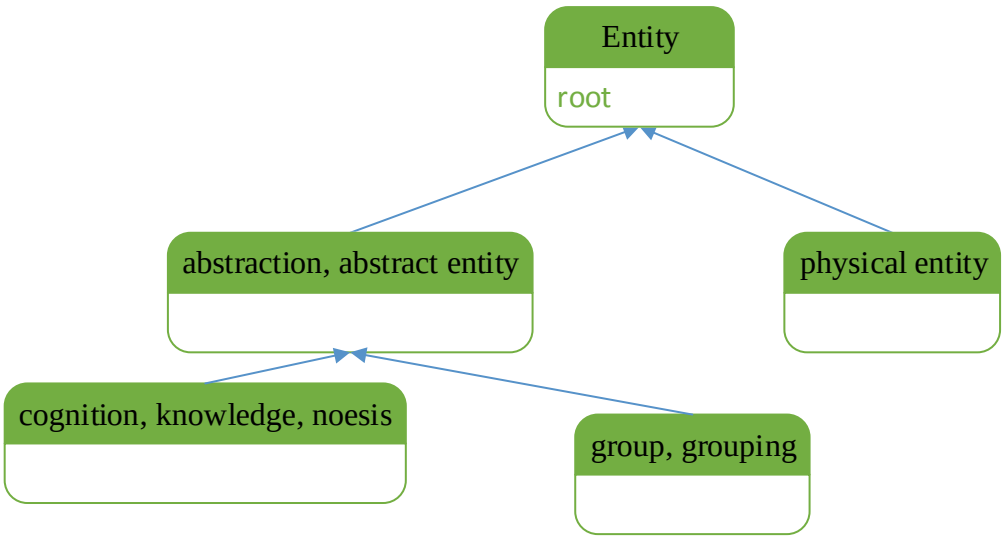


Figure 17 : les hyponyms de concept « Entity » est : {(*abstraction, abstract entity***),(** *physical entity***),(** *cognition, knowledge, noesis***),(** *group, grouping***)}**

Si la requête est un arbre avec M arêtes, on peut montrer que les hyponyms de base à plus de 4 M sont nécessaires.

La requête précédente sera : Si nous trouvons l'intersection entre les hyponymes directement nous exclurons les autres.

3 Query Parser Lucene :

Avec la requête affiné nous devons ouvrir l'index lucene et créer Query Parser Lucene, et effectuer une recherche Lucene, donc le résultat est une ensemble des chemins des documents.

4 Filtrage et classement :

Premièrement on fait un filtre qui élimine les chemins qui n'appartient pas à l'index sémantique. Dans la prochaine étape, les concepts de la requête sont jumelés aux concepts de l'index sémantique. Pour évaluer la pertinence d'un index sémantique selon

un ensemble de documents du résultat, cinq coefficients typiques sont calculés. Ces coefficients sont normalisés. Les quatre premiers coefficients définissent :

Nous devons définir quelques détails :

- QSET l'ensemble de ***m*** concepts dans la requête.
- $DSET_i$ l'ensemble de concepts dans le document i.
- $DQSET = QSET \cap DSET_i$.
- $\overline{DQSET} = (QSET \cup DSET_i) - QSET$.
- n nombre de documents de résultat.

4.1 Le taux de concepts directement impliqués dans le document, appelé Concepts Degree ou **CD.**

$$\mathbf{CD_i} = \frac{|DQSET| * \sum_{j=1}^{|DQSET|} représentativité\left(synsets_j(C_j)\right)}{\sum_{k=1}^{|\overline{DQSET}|} représentativité\left(synsets_k(\acute{C}_k)\right)} \begin{cases} \boldsymbol{C} \in DQSET \\ \acute{C} \in \overline{DQSET} \end{cases}$$

$représentativité(C)$ $représentativité$ de C $dans$ le $document$ i

4.2 Le taux de document concernés par les concepts de la requête, appelée the query Cover Degree ou **QCD**, qui donne le nombre de documents qui impliquent au moins un concept de la requête avec la représentativité :

$$\sum_{i=1}^{n} CD_i$$

4.3 Requête-Set Adéquation Degré : QSAD Le ratio entre chacun des CD et QSD, les coefficients sont évalués de façon expérimentale. L'équation donne la présente évaluation où D est un document et Q est une requête.

$$\boldsymbol{QSAD}_{\boldsymbol{D,Q}} = \boldsymbol{CD_i} \times \frac{\mathbf{1}}{\boldsymbol{QCD}}$$

5 Scénarios d'exécution avec captures d'écrans :

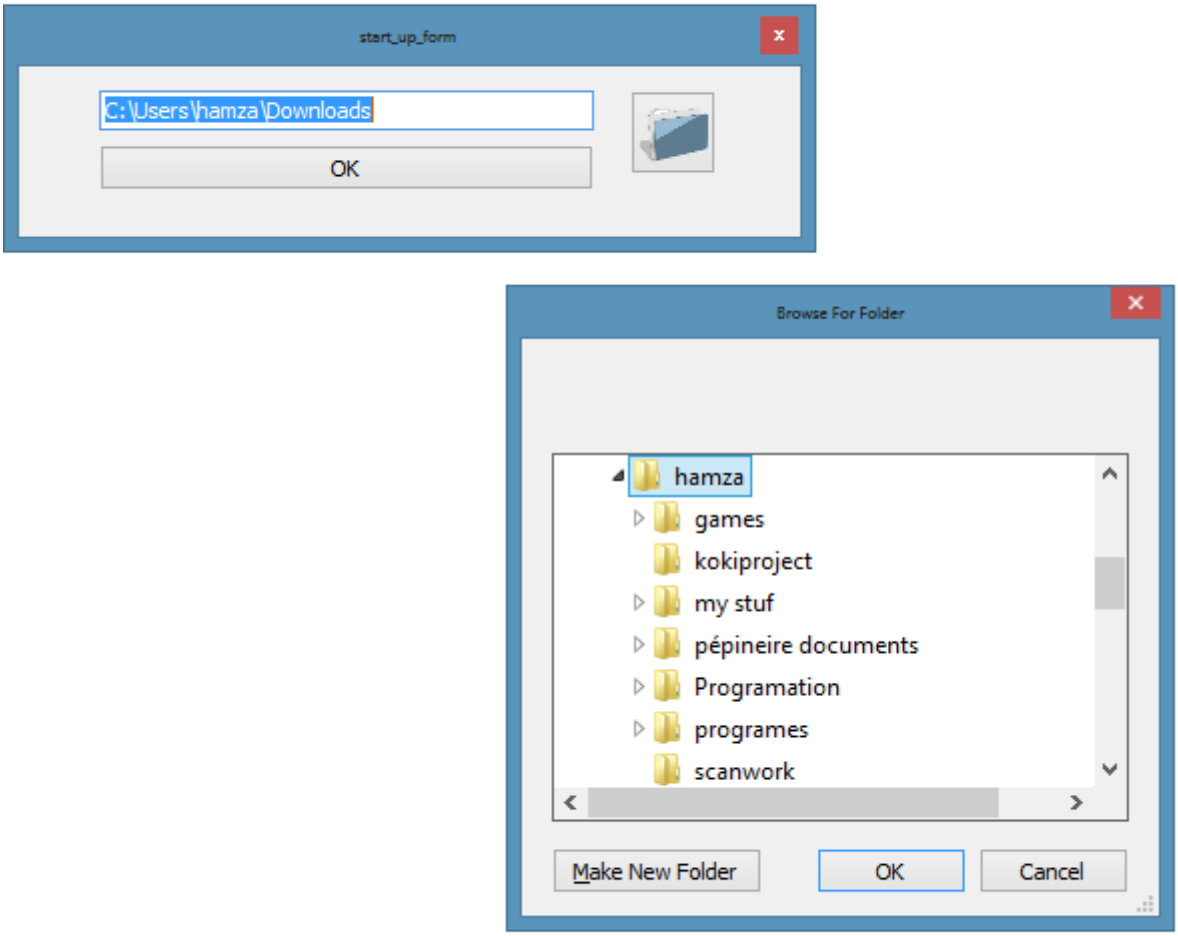


Figure 18 : première fenêtre la première exécution : Le programme demande à l'utilisateur de spécifier un chemin de répertoire.

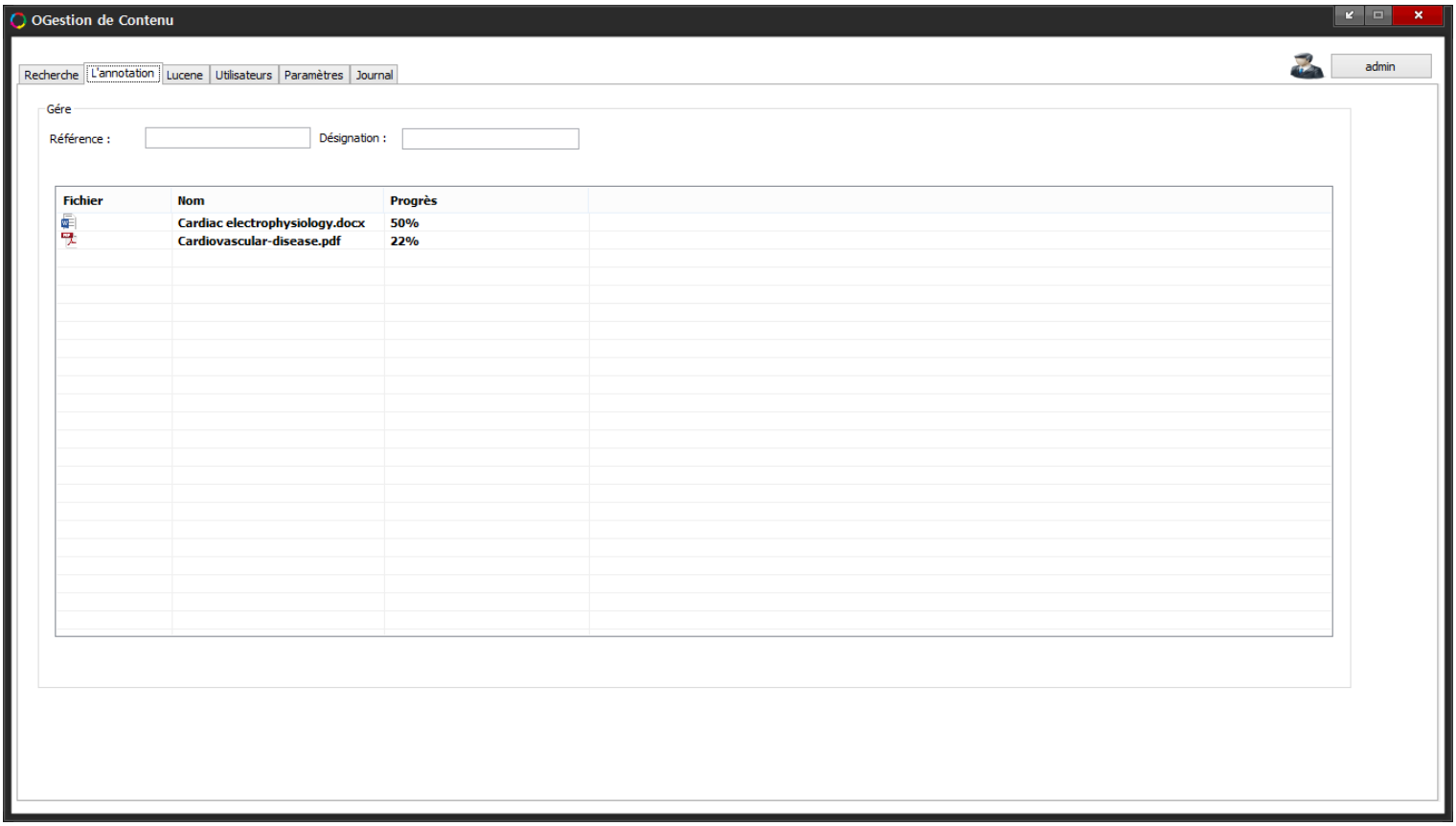


Figure 19 : l'annotation des documents retrouvés dans le répertoire.

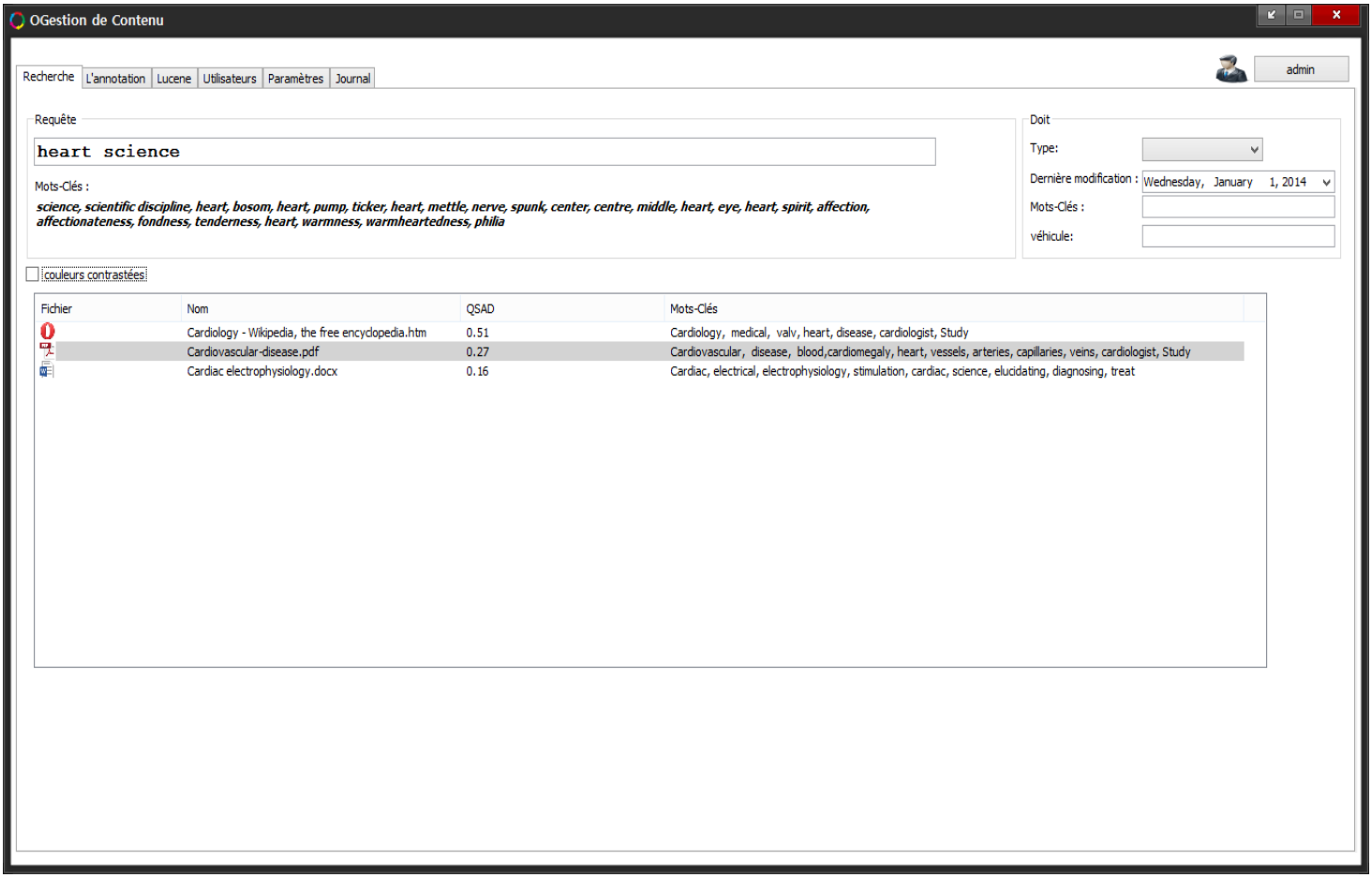


Figure 20 : exemple d'une requête de recherche : dans la partition de requête, nous pouvons vu les synsets trouvées, les résultats présentés avec ses mots-clés.

Conclusion

A la fin de cet humble travail, rappelons tout d'abord la problématique qui nous a été assigné à savoir

- Construire un GED faisant l'indexation d'une manière sémantique.

- Un outil de recherche efficace et intelligent.

C'est-à-dire un outil de recherche qui comprend la requête de l'utilisateur écrit avec un langage naturel .Et en plus il peut donner la réponse à la requête en cherchant dans l'index qu'il a déjà été fait grâce aux textes des documents. Le volume d'information disponible électroniquement est toujours plus important. Alors trouver des documents pertinents est une tâche de plus en plus délicate. L'ambigüité du langage naturel contribue à cette difficulté. Les multiples sens des termes et leurs multiples utilisations dans des domaines très variés participent également à rendre la tâche de recherche délicate. Nous avons présenté trois parties dans ce mémoire : la première partie a été consacrée à des généralités sur les moteurs de recherche ainsi que sur la gestion électronique des documents. La seconde partie : on a pris soin de bien décrire les outils de recherche et leur fonctionnement. On a bien mis en évidence les deux étapes qui sont le crawling et l'indexation dans cette partie on a présenté l'annotation sémantique et nous avons présenté le processus de création d'une ontologie, La troisième partie qui est la « conception » .nous avons présenté le processus de création de notre système intégré de recherche, ainsi que son fonctionnement .c'est a dire les étape a suivre après la requête de l'utilisateur jusqu' 'au retour du résultat de la requête .Ainsi on a atteint notre objectif en répondant à la problématique posée dans ce travail. Mais nous avons rencontré un petit souci dans le temps de réponse qui dépasse notre espérance. On doit y remédier. Grâce à ce mémoire nous avons eu la chance de manipuler et d'acquérir une certaine maitrise des outils de création et de manipulation des ontologies. Il nous a permis aussi de toucher et comprendre le monde des moteurs de recherche.

Merci.

Bibliographie

[1] Micheal Dewing , *les medias sociaux – introduction Bibliothèque de parlement de Canada*, 2012 ;

[2] William Ory, *Léa Hasgeyer , d'Arnaud Verchère, les médias sociaux , wellcom* , 2012 ;

[3] Rémi Bachelet , *réseaux sociaux, cours licenceCreativeCommons,centrale Lille*, mars2013 ;

[4] Wasserman et Faust , *Social Network Analysis: Methods and Applications*, Cambridge *University Press* , 1994;

[5] Uschold, M., M. King, S. Moralee, Y. Zorgios. *The enterprise ontology Knowledge Engineering Review*, 13(1), 31–90, 1996;

[6] Fox, M.S. *The TOVE project: a common-sense model of the enterprise,industrial and engineering applications of artificial intelligence and expert systems*. In F.Belli and F.J. Radermacher (Eds.), Lecture Notes in Artificial Intelligence, 604.Springer–Verlag. pp. 25–34.1992;

[7] GRUBER T., *A translation approach to portable ontology specifications, Knowledge Acquisition 5(2)*, 1993;

[8] LENART, Michèle. *La Gestion documentaire : évolutions fonctionnelles et description de dix logiciels*. ADBS éditions : Paris, 2004. 185p ;

[9] **Christopher D. Manning** , *Introduction to Information Retrieval* écrit par;

[10] **PRIE Y. & GARLATTI S**., *Méta-données et annotations dans le Web sémantique, in Le Web sémantique*, LAUBLET P. & REYNAUD C. (Ed.), Hors série de la Revue Information - Interaction - Intelligence , Cépaduès, Toulouse, 2004, pp. 45-68 ;

[11] HABERT B., *Instruments et ressources électroniques pour le français*, Collection "L'essentiel Français", Ophrys, Paris, 2005, 169 p ;

[12] Tim Berners-Lee , *Weaving the Web, HarperCollins*, new York, 1999

[13] Tim Berners-Lee , James Hendler , Ora Lassila ,*The Semantic Web*, Scientific American May 2001;

[14] Alexandre BERTAILS, Ivan HERMAN, Sandro HAWKE, *RÉALITÉS INDUSTRIELLES* , NOVEMBRE 2010 , de la page 84 a 89 ;

[15] Borst, W. N. *Construction of Engineering Ontologies. Center for Telematica and Information Technology*, University of Tweenty, Enschede, NL, 1997;

[16] GRUBER T., *A translation approach to portable ontology specifications, Knowledge Acquisition 5(2),* pages 199-220, 1993;

[17]… USCHOLD, M.KING, M.MORALEE, S. zORGIOS, The Enterprise Ontology. In The Knowledge Engineering Review, vol. 13, Special Issue on Putting Ontologies to Use,1998;

[18] Gómez-Pérez, A. Ontological Engineering: A state of the Art. Expert Update. BritishComputer Society. Autumn, 1999;

[19] McGuinness, D.L., Fikes, R., Rice, J. and Wilder, S. (2000). An Environment forMerging and Testing Large Ontologies. Principles of Knowledge Representation and Reasoning: Proceedings of the Seventh International Conference (KR2000). A. G. Cohn, F.Giunchiglia and B. Selman, editors. San Francisco, CA, Morgan Kaufmann Publishers;

[20] Guarino, N. Formal Ontology and Information Systems. In Proc. of Formal Ontology and Information Systems, Trento, Italy. IOS Press, 1998;

[21] Uschold, M., M. King, S. Moralee, Y. Zorgios. The enterprise ontology Knowledge Engineering Review, 13(1), 31–90, 1996;

[22] Fox, M.S. The TOVE project: a common-sense model of the enterprise,industrial and engineering applications of artificial intelligence and expert systems. In F.Belli and F.J.Radermacher (Eds.), Lecture Notes in Artificial Intelligence, 604.Springer–Verlag. pp. 25–34.1992;

[23] G. McLlachlan, Discriminant Analysis and Statistical Pattern Recognition. Wiley Interscience, Canada, 2004;

[24] C.Bishop, Neural Networks for Pattern Recognition, Ox;

[25] MICHAEL SCHRENK. Webbots, Spiders, and Screen Scrapers. No Starch Press, 2007.

[26] Joseph Gabay & David Gabay, *Uml 2 Analyse Et Conception*, Dunod 2008.

Webographie

http://fr.wikipedia.org/wiki/Web_s%C3%A9mantique

http://wordnet.princeton.edu/

http://fr.wikipedia.org/wiki/Ontologie_(informatique)

http://www.lesmoteursderecherche.com

http://docs.abondance.com/portails.html

http://www.webrankinfo.com/

http://www.dsi-info.ca/moteurs-de-recherche/langages/operateurs-logiques.html

http://www.bius.jussieu.fr/web/recherch.html

http://www.asktibbs.com/php/article.php3?id_article=11

http://www.uhb.fr/ccb/moteurs.htm

http://www.commentcamarche.net/utile/recherch.php3

http://searchenginewatch.com/

http://www.indicateur.com

http://www.search-marketing.info/search-engine-history/

http://www.answers.com/topic/archie-search-engine

http://www.owil.org/lexique/r.htm