République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université de Larbi Tébessi -Tébessa



Mémoire de Master

Présenté à l'Université de Tébessa
Faculté des Sciences Exactes et Sciences de la Nature et de la Vie

Département de : **Mathématiques et Informatique**
Spécialité : **Informatique**
Option : **SI**

Par :

Mr: ABDERRAHMANE Faouzi
Mlle: BENAOUA Aouatef

Thème

Interrogation de l'Ontologie DBpedia par les mots-clés

Jury

Président	**Professeur**	**Laouar Med. Ridha**	**Université de Tébessa**
Rapporteur	**Docteur**	**Bourougaa Salima**	**Université de Tébessa**
Encadreur	**Docteur**	**Djeddai Ala**	**Université de Tébessa**

Remerciements

Nous remercions avant tout, Dieu de nous a prodiguée la force morale et physique et nous a permis d'achever ce travail.

Ce travail n'aurait jamais pu voir le jour sans la contribution et l'effort de Mr. DJEDDAI Ala (Docteur) à qui nous tenons à exprimer notre profonde gratitude pour avoir bien voulu accepter de diriger ce travail.

Nos respects et notre gratitude vont également à MR. LAOUAR Med. Ridha (Professeur) , pour avoir accepté de présider le jury de cette thèse.

Nos profonds remerciements à Mme BOUROUGAA Salima (Docteur) pour avoir accepté d'être membre du jury de cette thèse.

Nous remercions tous les enseignants du département d'informatique que nous respectons beaucoup.

Enfin, nous remercions toutes nos familles et nos amis.

Dédicaces

À la mémoire de mes très chers parents.
À ma femme et mes enfants.
A tous mes frères et mes sœurs.
À tous mes amis et mes collègues .
À tous ceux qui sont proches de mon cœur et dont je n'ai pas cité les noms.

Je dédie ce modeste travail.

" ***A. Faouzi*** "

Á mes très chers parents
Á mes frères et leurs familles
Á mes Amis.

Je dédie ce modeste travail.

B . Aouatef

Résumé – Abstract

Résumé :

Le Web sémantique et les ontologies jouent un rôle pivot notamment dans le contexte de la recherche d'information dans le but de pallier aux insuffisances des systèmes existants.

La sémantique d'une requête d'un utilisateur exigeant en termes de précision est difficile à extraire par une machine a l'aide des techniques classique. Le défi d'interroger la base de connaissances DBpedia par les mots clé nous obliges d'adapter une approche de recherche fiable pour résoudre les problèmes d'ambigüités entre les termes et les concepts.

Une solution proposée formalise le problème de la construction des requêtes formelles meilleures en traduisant les requêtes mots-clés en requêtes SPARQL en utilisant les connaissances disponibles dans l'ontologie DBpédia .

Abstract :

The Semantic Web and ontologies play a role particular in the context the search for information with the aim of mitigating the inadequacies of the existing systems.

User semantics of a request requiring in terms of precision is difficult to extract by a machine has the classic help(assistant) of techniques. The challenge to question the knowledge base DBpedia by keywords oblige us to adapt an approach of reliable search(research) to solve the problems of ambiguities between the terms and the concepts.

A proposed solution formalizes the problem of the construction of the better formal requests by translating the requests keywords into requests SPARQL by using the available knowledge in the ontology DBpedia .

Sommaire

	Page
Introduction Générale	**1**
Chapitre 1: Principes Généraux	
1.1. Le Web Sémantique	3
1.1.1. Définition	3
1.1.2. Le World Wide Web Consortium (W3C)	5
Protocoles utilisés	5
a- le protocole HTTP	5
b- les U R I	5
c- le langage XML	5
1.1.3. Annotation sémantique et concept de métadonnée	6
a- Annotation sémantique	6
b- Concept de métadonnée	7
1.1.4. Les Ressources Sémantiques (“ Les Ontologies”)	8
a- Taxonomie	8
b- Thesaurus	8
c- L'ontologie	9
Type d’ontologie	10
L’utilité des Ontologies	10
Ontologies Majeures	10
Web de données liées (Linked Data)	12
Web de données ouvertes et liées (Linking Open Data)	12
1.1.5. L'ontologie Dbpedia	12
1.2. Les Langages des Ontologies	14
1.2.1. RDF	14
a- Les syntaxes de RDF	15
b- Les graphes conceptuels (GCs)	16
1.2.2. RDFS (RDF Schema)	17
a- Les Classes	17
b- Les propriétés	18
c- Les Individus	18
1.2.3. OWL	19
1.2.4. SPARQL	19
a- Une requête SELECT	20
b- Une requête ASK	20
c- Une requête CONSTRUCT	20
d- Une requête DESCRIBE	21

1.3. La recherche d'information	21
1.3.1. Introduction	21
1.3.2. Définition	22
1.3.3. Modèles de recherche d'informations	23
a- Le modèle booléen	23
b- Le modèle vectoriel	24
c- Le modèle probabiliste	25
1.3.4. La recherche sémantique d'information	25
a- Indexation sémantique	26
b- Traitement sémantique de la requête	27
c- Appariement sémantique	27
1.4. Conclusion	27
Chapitre 2: Etat De L'Art	
Introduction	28
2.1. Description	28
- Interrogation en mode simple	28
- Interrogation en mode complexe	29
2.2. Détail des l'approches existantes	29
2.2.1. Interrogation en mode simple : (Approches Coopératives)	29
a - Suggestion de requêtes	29
b - Interrogation par l'exemple	30
c - Explications sur les réponses d'une requête	30
d - Raffinement de la requête par l'utilisateur (Users refinement)	30
e - Redéfinition des requêtes	31
2.2.2 Interrogation en mode complexe: (Approches basées sur la structure des ressources)	31
2.3 . Les approches selon le degré de formalité	32
2.4 . Modèles de Moteurs sémantique existants	34
2.5 . Conclusion	35
Chapitre 3 : La Conception	
Introduction	36
3.1 - Description	36
3.2 - Présentation du problème	37
. Exemple (1)	38
. Prefixes (URIs)	39
3.3 - Etapes du système	40
3.3.1. Mappage des mots clés aux éléments de la B.C	40
Définition	40

Le détail	41
. Le mappage morphologique	42
. Le mappage sémantique	42
. Suite de l'exemple (1)	42
3.3.2 Construction des Graphes Requête	44
a - Calcul des graphes de Connexions	44
. Explorer les liens entre les éléments de la B.C	44
. Calcul des graphes de connexions	47
b- Mappage des graphes de Connexions aux graphes requête	49
3.3.3 Classement des requêtes	51
• Algorithme de Classement	52
3.4 - Conclusion	53
Chapitre 4 : Implémentation et Evaluation	
Introduction	54
4.1. L'Implémentation	54
4.1.1. Les outils et les langages utilisés	54
4.1.2. L'Implémentation	55
a- Edition de l'ontologie DBpedia	55
b- Exploitation du code OWL dans le programme Java	55
c- Exécution de l'application	56
. Interface principale	56
. Interface après recherche sur le mot clé « book »	57
. Interface après la recherche sur la requête « writer misérable book »	57
4.2. Evaluation	58
4.2.1. Métriques utilisées	58
a- La précision	58
b- Le rappel	58
4.3 . Conclusion	59
Conclusion Générale	60

Liste des Figures

N°	Titre du Figure	Page
Figure 1.1	La pile des standards du web sémantique	4
Figure 1.2	Les groupes de la pile des standards du WS.	4
Figure 1.3	Exemple d'annotation Sémantique basée ontologie	6
Figure 1.4	Exemple de Métadonnées d'un fichier PDF	8
Figure 1.5	Vocabulaires et richesse sémantique	9
Figure 1.6	Le web de données en Avril 2017	11
Figure 1.7	L'Ontologie Dbpedia	12
Figure 1.8	Wikipedia avec infobox pour créer DBpedia	13
Figure 1.9	Représentation d'un triplet RDF (sujet , relation , objet).	14
Figure 1.10	Ensemble de triplets RDF	15
Figure 1.11	Représentation sous forme de graphe de l'ensemble de triplets RDF	15
Figure 1.12	Document représenté en figure 6 au format RDF/XML.	16
Figure 1.13	Exemple de graphe conceptuel	16
Figure 1.14	Exemple de l' hiérarchie de classes	17
Figure 1.15	Exemple de l'hiérarchie de propriétés	18
Figure 1.16	Exemple d'instances des types Homme, Femme et Lieu .	19
Figure 1.17	Exemple de requête SPARQL SELECT	20
Figure 1.18	Résultat de la requête SPARQL SELECT	20
Figure 1.19	Exemple de requête SPARQL ASK.	20
Figure 1.20	Exemple de requête SPARQL CONSTRUCT	21
Figure 1.21	Résultat de la requête SPARQL CONSTRUCT	21
Figure 1.22	Exemple de requête SPARQL DESCRIBE	21
Figure 1.23	Résultat de la requête SPARQL DESCRIBE	21
Figure 1.24	Productivité de la recherche	22
Figure 1.25	Système de recherche d'information	23
Figure 1.26	Appartenance des termes d'une requête dans un document	24
Figure 1.27	Le modèle vectoriel	24
Figure 1.28	Système de recherche sémantique de l'information	26
Figure 2.1	L'interface du moteur Corese	34
Figure 2.2	L'interface du moteur Evi	35
Figure 2.3	L'interface du moteur wolframalpha	35
Figure 3.1	Architecture générale de l'approche	37
Figure 3.2	Le Graphe Conceptuel extrait de l'ontologie DBpedia	39
Figure 3.3	Calcule de similarité	41

Figure 3.4	Exemple de mapping	43
Figure 3.5	Algorithme (1) Exploration de la B.C	45
Figure 3.6	Algorithme (2) Passé	46
Figure 3.7	Graphe résultat d'application de l'algorithme d'exploration	47
Figure 3.8	Algorithme (3) calcule des graphes de connexions	48
Figure 3.9	Graphe des Connexions 1	48
Figure 3.10	Graphe des Connexions 2	49
Figure 3.11	Graphe de requête 1	50
Figure 3.12	Graphe de requête 2	50
Figure 3.13	La requête Sparql du Graphe Requête (1)	50
Figure 3.14	La requête Sparql du Graphe Requête (2)	50
Figure 3.15	Classement des requêtes	51
Figure 3.16	Algorithme (4) - Classement des résultats.	52
Figure 4 .1	Vue de Librairies utilisés sur JAVA Eclipse	56
Figure 4 .2	Interface Principale de l'application.	56
Figure 4 .3	Résultat de recherche sur la requête « book »	57
Figure 4 .4	Résultat de recherche sur la requête « writer misérable book »	57

Liste des Tableaux

N°	Titre du Tableau	Page
Tableau 1.1	L'évolution de version Anglaise de DBpedia	14
Tableau 2.1	Classification des approches par KAUFMANN E. & BERNSTEIN A	33
Tableau 3.1	Les valeurs des similarités trouvées entre les mots clés et les ressources	43
Tableau 3.2	Classement des requêtes formelles selon leurs scores	52
Tableau 4.1	Ensembles des requêtes mots-clés	58
Tableau 4.2	Résultats de calcule des Rappels et Précisions	59

Introduction Générale

Suite à une évolution fulgurante avec le World Wilde Web et ses différents domaines, les chalenges posés aux premiers moteurs de recherche du marché dans les années 90 ne sont plus comparables aux défis posés aujourd'hui et avec l'avènement du Web sémantique et la description sémantique des ressources, les ontologies jouent un rôle pivot parce qu'elles sont utilisées dans plusieurs domaines dont notamment la recherche d'information afin de pallier aux insuffisances des autres modèles.

Aussi, la sémantique d'une requête utilisateur est difficile à extraire par une machine (un même mot peut désigner des contenus très différents et différents mots peuvent désigner un même contenu). Il devient alors très difficile d'avoir des résultats précis. Les techniques d'indexation et de référencement par mots clés ne suffisent plus à pallier au problème devant les besoins croissants des utilisateurs, de plus en plus exigeants en termes de précision face à leurs requêtes.

Dans ce contexte, nous y trouvons devant le défi d'interroger en utilisant les mots clés l'ontologie " DBpedia " afin de satisfaire les utilisateurs par des réponses pertinents .

Pour la plupart des utilisateurs, rechercher des informations dans une ontologie n'est pas une tâche facile. Cette difficulté est due aux trois principales raisons suivantes.

- ✓ Une ontologie est souvent construite automatiquement à partir d'un ensemble de sources de données hétérogènes. En conséquence, les utilisateurs ont souvent une connaissance partielle du contenu et de la structure d'une ontologie.
- ✓ Le schéma d'une ontologie est défini avec des langages basés sur les logiques de description dont la sémantique et les hypothèses sous-jacentes ne sont pas forcément bien comprises par les utilisateurs.
- ✓ Malgré leur taille très importante , les ontologies sont incomplètes.

Face à ce constat, notre objectif principale dans ce travail est d'adapter la recherche par mots clés dans l'ontologie DBpedia. Le travail comporte 3 étapes principales : chercher les ressources similaires aux mots clés, construire les graphes des requêtes et enfin générer et exécuter des requêtes Sparql sur DBpedia.

Ce mémoire est structuré en quatre chapitres détaillés comme suit :

- Chapitre 1: Présente un aperçu des concepts généraux de notre travail, il débute par quelques détails sur le web sémantique , le W3C , les ressources sémantiques, les ontologies et spécialement DBpédia. En suite un aperçu sur les langages de ce domaine (RDF , RDFS , SPARQL), et en finissant par la définition de la recherche d'information classique et sémantique .
- Chapitre 2 : Vise une étude de l'état de l'art spécifié concernant les différentes approches existantes dans ce contexte de la recherche sémantique, avec quelques détails sur les travaux connexes y compris quelques application de moteurs de recherche sémantique connus .
- Chapitre 3 : Ce chapitre présente notre choix de l'approche convenable pour faire fasse aux défis et à la problématique de notre sujet . On commence par l'architecture générale , description du problème et le détail des trois étapes suivi par des exemples explicatifs extraits de l'ontologie DBpedia
- Chapitre 4 : Réservé à la phase d'implémentation du programme . Nous citons les outils et les logiciels utilisés, puis l'exécution du code avec une visualisation de quelques images d'écran. Pour finir, nous faisant une simple évaluation des résultats obtenues par des métriques connues du domaine .

1 CHAPITRE

Principes Généraux
Sur Le web sémantique Et La recherche d'information

Introduction :

Ce chapitre présente un aperçu des concepts généraux de notre travail, il débute par quelques détails sur le web sémantique , le W3C , les ressources sémantiques , les ontologies et spécialement DBpédia , ensuite un aperçu sur les langages de ce domaine (RDF , RDFS , SPARQL), en finissant par la définition de la recherche d'information classique et sémantique .

1.1. Le Web Sémantique :

1.1.1. Définition :

Le Web classique se base essentiellement sur la structure du document et les liens entre les pages Web, ce qui rend l'exploitation des informations impossible par les machines. En revanche, le Web sémantique offre une représentation sémantique du contenu, ce qui permet aux agents logiciels l'extraction de l'information.

Dans son ouvrage Weaving the Web, Tim Berners-Lee (co-inventeur avec Robert Cailliau du World Wide Web) donna sa définition la plus aboutie [BER 01] « *Le web sémantique désigne une évolution du web syntaxique. Ce dernier est entièrement destiné à la consultation de documents par des utilisateurs humains. Dans le web sémantique, les contenus sont également compréhensibles et exploitables par des agents logiciels. Sa mise en place permettrait l'automatisation d'un grand nombre de tâches et un accès plus facile à la connaissance, en s'appuyant sur l'utilisation d'un langage de description commun, permettant d'exprimer à la fois des ontologies et des assertions (éventuellement sous formes d'annotations dans des documents web) utilisant le vocabulaire de ces ontologies* ».

Et selon Goble [CHR, KEI 07]: « *le Web sémantique vise a favoriser la découverte, l'échange d'information et l'intégration de l'information par l'étiquetage (tagging) ou le balisage (marking up), et a développer des technologies et infrastructures pour annoter le contenu du Web avec des informations additionnelles appelées " métadonnées " pouvant être traitées par des agents logiciels* ».

D'un point de vue technologique, le web sémantique peut se définir par la pile de standards qui le composent et fixent le cadre de sa mise en place. Un schéma largement diffusé, appelé " semantic web layer cake" (voir Figure 1.1 et Figure 1.2), décrit cette pile de standards et constitue à lui seul un résumé de la feuille de route dressée par Tim Berners-Lee. L'agencement en pile illustre l'organisation et les dépendances entre les standards, chaque couche exploitant les couches inférieures.

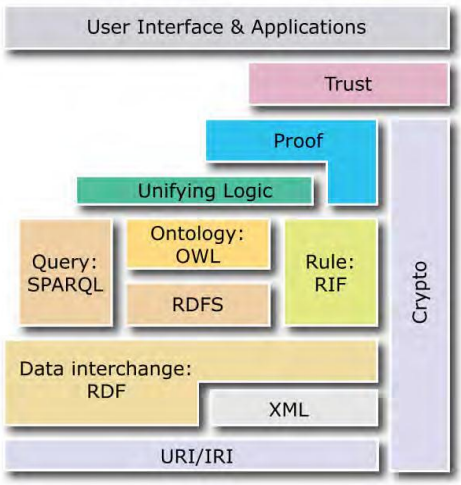


Figure 1.1 : La pile des standards du WS [01].

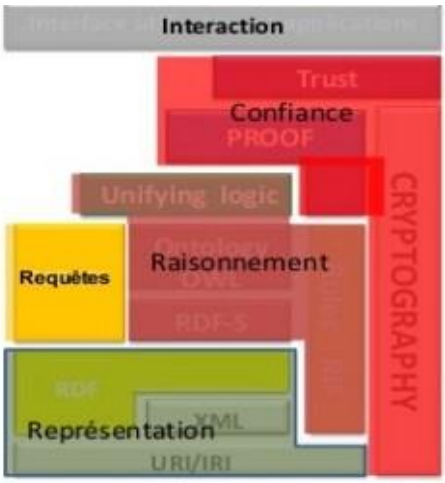


Figure 1.2 : Les groupes de la pile [01]

- URI : permet d'identifier les ressources sur les réseaux.
- XML : est utilisé comme couche de base syntaxique du web sémantique. Le langage XML est considéré comme un standard pour le transport de données sur le web.
- RDF : représente les métadonnées pour les ressources web.
- OWL : fondée sur une formalisation commune, spécifie la sémantique de méta-données fournies dans le web sémantique.
- Logique : s'appuie sur des règles d'inférence qui permettent le raisonnement intelligent exécuté par des agents logiciels.
- Preuve : les langages logiques permettent la mise en œuvre d'outils de raisonnement. Des outils de raisonnement sont disponibles pour des langages comme OWL et permettent par exemple de tester la cohérence des informations, de les classifier.
- Confiance : la couche confiance est située au haut de la pyramide. Elle concerne l'utilisation de signatures numériques et d'autres types de connaissances afin de garantir la fiabilité et l'origine des informations, par des recommandations d'agents de confiance, de la notation, des organismes de certification et des organismes de Consommateurs.

1.1.2. Le World Wide Web Consortium (W3C) :

Abrégé par le sigle W3C, est un organisme de standardisation à but non lucratif, fondé en octobre 1994, chargé de promouvoir la compatibilité des technologies du World Wide Web telles que : (HTML5 , HTML , XHTML , XML, RDF, SPARQL, CSS , XSL, PNG, XSL, PNG, SVG et SOAP). Fonctionnant comme un consortium international, il regroupe plusieurs entreprises partenaires.

Protocoles utilisés :

a- le protocol HTTP :

Le HTTP (*HyperText Transfer Protocol*), permet, à partir d'une adresse URL, de demander une représentation de la ressource identifiée (par exemple une page Web) et localisée par cet URL, et d'obtenir en retour soit les données de cette représentation, soit des codes d'erreur indiquant un problème rencontré. Nous avons tous déjà rencontré la célèbre « erreur 404 » qui indique que la page demandée n'a pas été trouvée.

b- les U R I :

Un URI (*Uniform Resource Identifier)* est un identifiant unique sur le Web (*une* adresse page web à propos de la ressource). Cet identifiant est le plus souvent sous la forme d'une URL de page web. Un URI est dit *" déréférençable*s " il désigne une page web existante à laquelle un utilisateur peut accéder via un navigateur Web.

c- le langage XML :

Le XML *(eXtensible Markup Language*), traduit ainsi (Langage *à balises étendu*), est un langage HTML amélioré permettant de définir de nouvelles balises. Il permet de mettre en forme des documents grâce à des balises (markup). XML peut être considéré comme un métalangage permettant de définir d'autres langages, c'est-à-dire définir de nouvelles balises permettant de décrire la présentation d'un texte. La force de XML est dans sa capacité de décrire n'importe quel domaine de données grâce à son extensibilité. Il va permettre de structurer, poser le vocabulaire et la syntaxe des données qu'il va contenir. Les balises XML décrivent le contenu plutôt que la présentation.

Le XML permet de séparer le contenu de la présentation .ce qui permet par exemple d'afficher un même document sur des applications ou des périphériques différents sans pour autant nécessiter de créer autant de versions du document que l'on nécessite de représentations .

1.1.3 Annotation sémantique et concept de métadonnée :

a . Annotation sémantique :

Le terme « *Annotation* » est défini dans le dictionnaire comme étant « *une note explicative ou critique qui accompagne un texte- une note de lecture que l'on inscrit sur un livre* ». Ce terme dérive du terme latin « *Annotare* » qui signifie « *accompagner un texte de notes, de remarques et commentaires* », et parce que nous annotons un sujet, une annotation seule n'a aucun sens, elle est associée a l'objet qui a été annoté.

Nous nous intéressons principalement à l'annotation de document par le contenu, l'un des aspects du web sémantique est de pouvoir créer et manipuler les annotations sémantiques, ce sont des Métadonnées de documents (voir l'exemple du Figure 1.3)

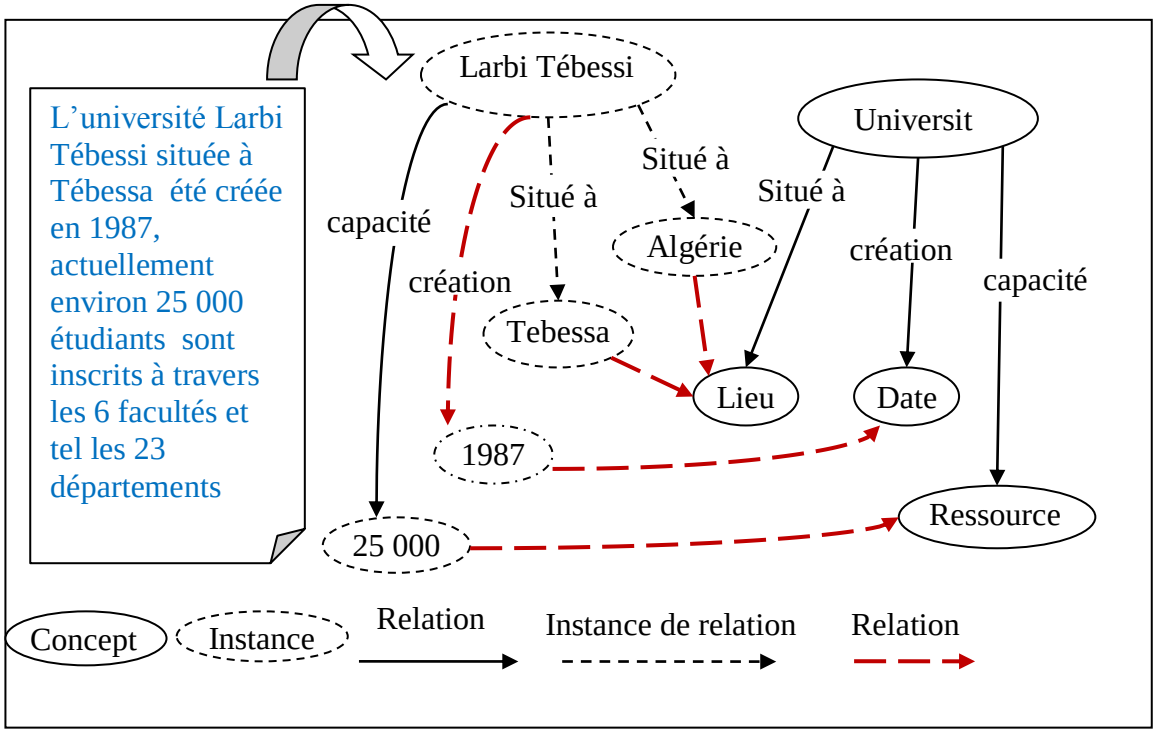


Figure 1.3 : Exemple d'Annotation Sémantique basée ontologie

La sémantique de l'annotation est fondée sur des vocabulaires dans les ontologies spécifies explicitement par un langage de représentation, par conséquent l'annotation sémantique est le processus le plus adapté pour le partage du contenu des documents.

Une annotation sémantique peut alors être définie selon Tuan Dung [TUA 06] comme :

« *Une représentation formelle d'un contenu, exprimée par des concepts, des relations, et instances décrits dans une ontologie reliée à le ressource documentaire* ».

Les approches d'annotation sémantique se distinguent par les types de connaissances ontologiques utilisés, et par la nature et la granularité des parties de document annotées. Les éléments ontologiques utilisés pour l'annotation peuvent être classés en cinq classes:

- Les concepts : termes définis dans une ontologie.
- Les instances de concepts.
- Les relations.
- Les instances de relations.
- Les valeurs littérales.

b. Concept de métadonnée :

Le Web sémantique est un moyen d'échange et de partage des ressources grâce à la représentation sémantique du contenu. Ces ressources sont décrites par des informations structurées additionnelles " *les métadonnées* ".

Les métadonnées selon Jorge Cardoso et Amit Sheth [JOR, ADM 06] sont : « *des données sur les données* ».

Le préfixe « *méta* » signifie que ces informations ne concernent pas le sujet traité dans le document mais qu'il s'agit de données portant sur le document lui même. Ces métadonnées peuvent être, suivant les technologies utilisées, intégrées dans le document, ou bien rassemblées en dehors des documents, dans une base de métadonnées jouant le rôle d'un catalogue de bibliothèque.

L'utilisation des métadonnées permet de décrire, identifier et définir le contenu de la ressource afin de rendre son contenu exploitable. Les métadonnées peuvent exister sur différents niveaux distincts. Les informations décrivant la syntaxe, la structure et le contexte sémantique.

Avec la diffusion de l'internet, et la croissance du Web, le terme " *métadonnées* " commence à être utilisé dans le contexte de la description des ressources numériques disponibles. Ainsi que les documents textuels qui sont compréhensibles par les humains, ils ont besoin des métadonnées pour que les machines puissent les manipuler, les contrôler, les rechercher et les récupérer.

La figure 1.4 illustre un exemple des métadonnées pour un fichier PDF.

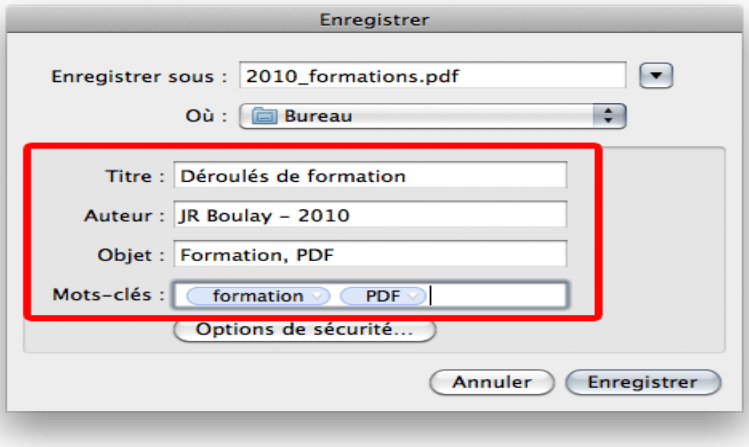


Figure 1.4 : Exemple de Métadonnées d'un fichier PDF. [02]

1.1.4 . Les Ressources Sémantiques (" Les Ontologies "):

Une ressource sémantique est un vocabulaire contrôlé. C'est une liste de termes d'un domaine ou de plusieurs domaines qui ont été énumérés explicitement. Cette liste est contrôlée par un expert. Chaque terme du vocabulaire doit avoir (théoriquement) une définition non ambigüe et non redondante (mais dans la pratique n'est pas toujours vrai).

Ils existent différents types de ressources sémantiques telles que les taxonomies, les thesaurus, les ontologies, glossaires et dictionnaires. Les gens utilisent dans la plupart du temps le mot ontologie pour indiquer ces différents concepts.

a. Taxonomie :

C'est la technique utilisée pour faire un classement dans un système hiérarchique des organismes. Une taxonomie est une collection de termes organisés en une structure hiérarchique. Les termes dans une taxonomie sont reliés entre eux par des relations parent-enfant.

Il peut y avoir différents types de relations parent-enfant dans une taxonomie (par exemple, tout-partie, genre-espèces, Type-Instance), mais une bonne pratique limite les relations parent-enfant (voir Figure 1.5).

b. Thesaurus :

Un thésaurus est un réseau de termes d'un vocabulaire contrôlé. Il utilise d'autres relations entre les termes, en plus de la relation parent-enfant.

Dans le domaine de recherche d'information un thésaurus cherche à dicter les relations sémantiques entre les termes pendant la phase d'indexation. Un thésaurus sert à minimiser

l'ambigüité sémantique en garantissant l'homogénéité et la cohérence dans le stockage et la récupération des termes du contenu.

Un thésaurus sert à guider à la fois la phase d'indexation et la phase de recherche en sélectionnant le meilleur terme ou la meilleure combinaison de termes qui représente un sujet donné. (Voir Figure 1.5)

c. Ontologie :

Face à l'émergence des sources d'informations disponibles de plus en plus nombreuses et complexes, il est nécessaire de permettre une description de ces informations non seulement en termes de structure (aspect syntaxique) mais également en termes de signification (aspect sémantique). La description de métadonnées sur les sources d'informations prenant en compte la description structurelle mais aussi sémantique des informations est un problème important. Cette sémantique peut être exprimée à l'aide d'ontologies.

Dès les début des années 90, Gruber proposait la définition suivante : " *Une ontologie est une représentation formelle des connaissances* " [GRU, 93]. Ensuite, il complète en précisant qu'une ontologie est une "*spécification formelle et explicite d'une conceptualisation partagée*" [GRU, OLS 94].

Et selon Gomez Carpio [GOM 10], " *Une ontologie permet la modélisation d'un domaine de connaissances et peut être vue comme un modèle conceptuel d'un domaine particulier, qui décrit les concepts de ce domaine et les relations entre ces concepts. Elle est généralement considérée comme une base de connaissances et est au centre des développements émergents du Web sémantique ".*

Les ontologies permettent la modélisation d'informations agréées par une communauté de personnes et accessibles par une machine pour développer des services automatisés et par conséquent, jouent un rôle de référence pour décrire la sémantique des informations à partager.

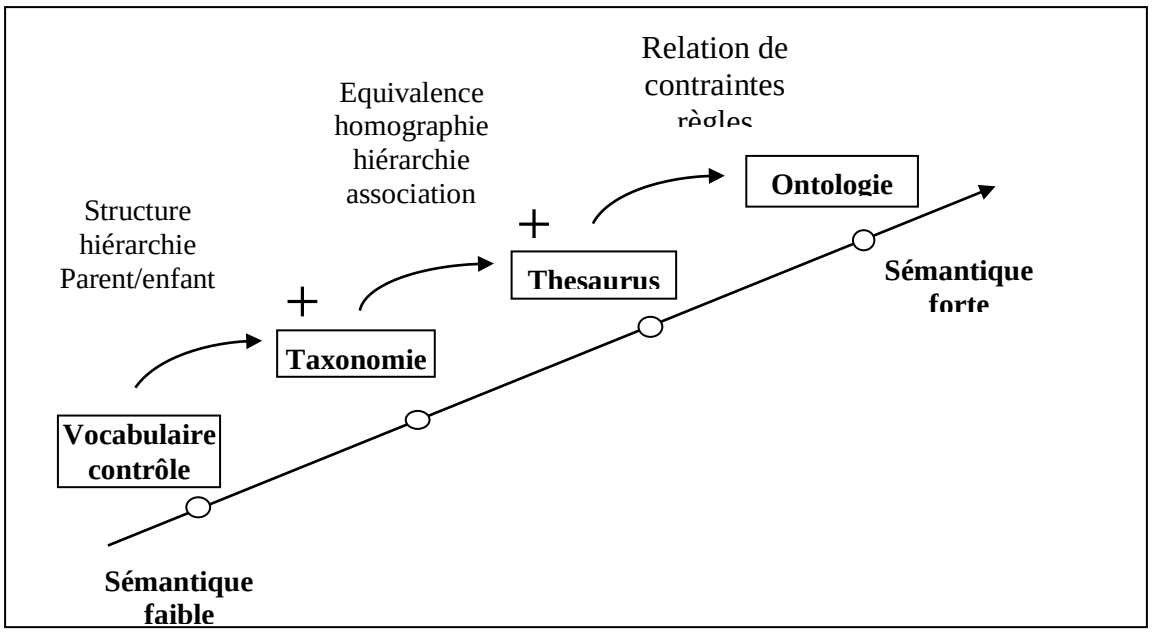


Figure 1.5 : Vocabulaires et richesse sémantique.

- **Types d'ontologies :**

Selon Van Heijst [VAN 97], les typologies d'ontologies fondées sur le sujet de la conceptualisation comportent quatre catégories :

- ✓ Les ontologies d'application : elles contiennent toutes les informations nécessaires pour modéliser les connaissances pour une application particulière.
- ✓ Les ontologies de domaine : elles fournissent un ensemble de concepts et de relations décrivant les connaissances d'un domaine spécifique.
- ✓ Les ontologies génériques (dites aussi de haut niveau) : elles sont similaires aux ontologies de domaine, mais les concepts qui y sont définis sont plus génériques et décrivent des connaissances tels que l'état, l'action, l'espace et les composants. Généralement, les concepts d'une ontologie de domaine sont des spécialisations des concepts d'une ontologie de haut niveau.
- ✓ Les ontologies de représentation (méta-ontologies) : elles fournissent des primitives de formalisation pour la représentation des connaissances. Elles sont généralement utilisées pour écrire les ontologies de domaine et les ontologies de haut niveau.

- **L'utilité des Ontologies :**

Les ontologies selon Domenico M. Pisanelli et ces co-équipiers [DOM, ALD, GER 02] servent à :

- ✓ Créer une compréhension partagée pour unifier différents points de vue,
- ✓ Faciliter la communication entre les personnes impliquées dans la construction du SI,
- ✓ Permettre la réutilisation des connaissances du domaine, faciliter la récupération,

l'intégration et l'interopérabilité entre les sources hétérogènes de la connaissance,

- ✓ Fournir une base de représentation des connaissances du domaine,
- ✓ Aider à identifier les catégories sémantiques de domaine.

- **Ontologies Majeures :**

Les Communautés actives autour du web sémantique sont reconnues par *" Les Bases Majeures* **"** , certaines de ces bases sont particulièrement connues et utilisées, elles sont liées à un très grand nombre d'autres bases, à un tel point qu'on peut les qualifier de « *majeures* » dans le W3C : (*DBpedia , YAGO , FOAF , Freebase ….. etc*) (voir Figure 1.6).

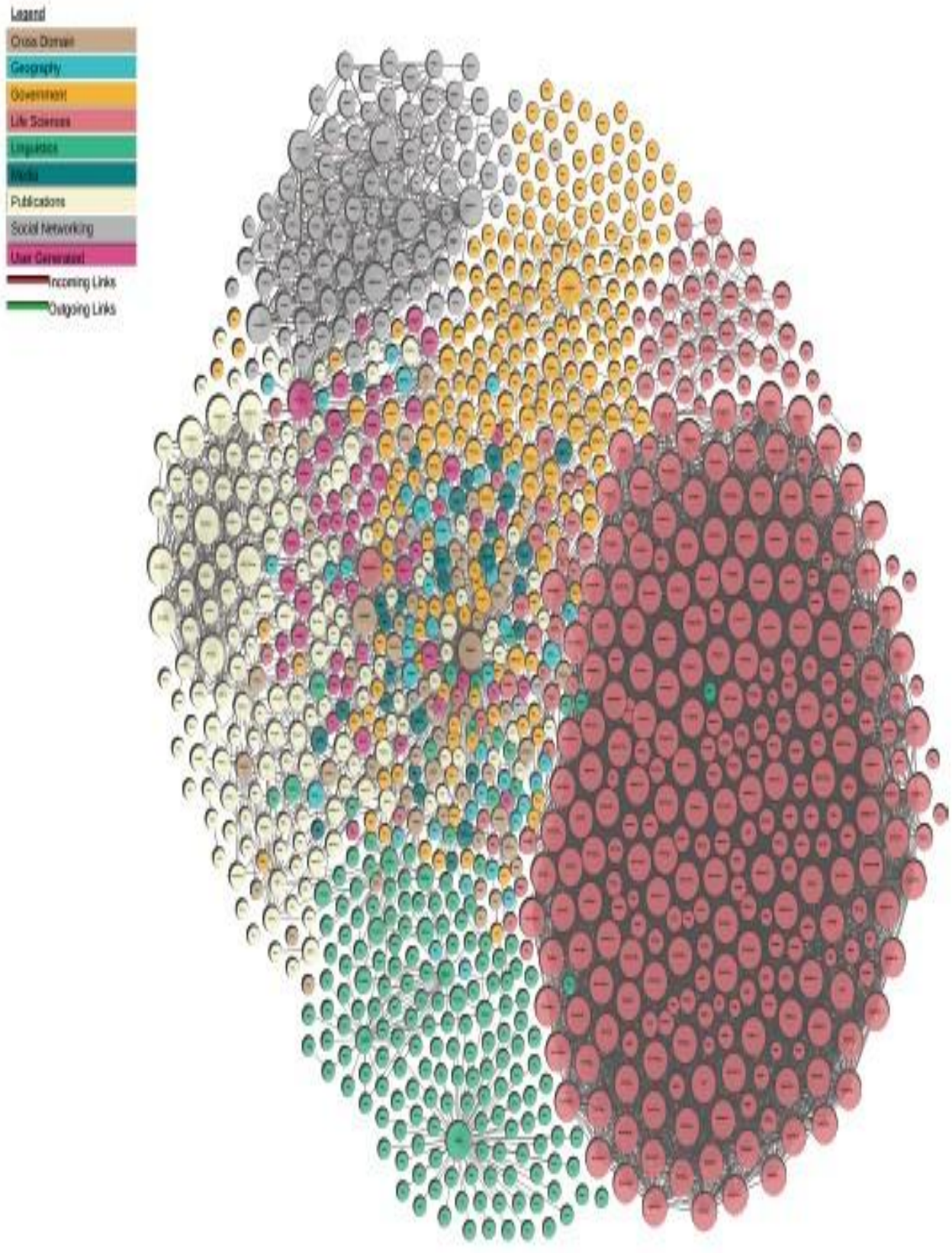


Figure 1.6 : Le web de données en Avril 2017. [03]

- **Web de données liées (Linked Data) :**

L'objectif de Linked Data est d'utiliser le Web pour faciliter la création de liens entre des données connexes qui n'étaient pas encore liées, ou qui l'étaient déjà via d'autres méthodes.

- **Web de données ouvertes et liées (Linking Open Data)** :

Linking Open Data est un projet communautaire collectif visant a publier un maxi--mum de données libres de droits sur le Web de données selon les principes de Linked Data, en interconnectant des données de différentes sources de données. Une fois ce réseau de sources de données créé, il est alors intéressant de développer des clients Linked Data permettant de naviguer parmi ces données interconnectées.

1.1.5 . L'ontologie DBpedia :

Depuis Juin 2007, une nouvelle application est en train de révolutionner la manière dont en peut rechercher et utiliser des données. DBpedia [LET, JEN, ISE 15] (voir Figure 1.7), est en effet une version base de données libre (open source) de certaines informations de Wikipédia avec une structure intelligible et partagée. Les données sont principalement extraites des boîtes d'informations (dites *infobox*) présentes dans les articles de Wikipédia.

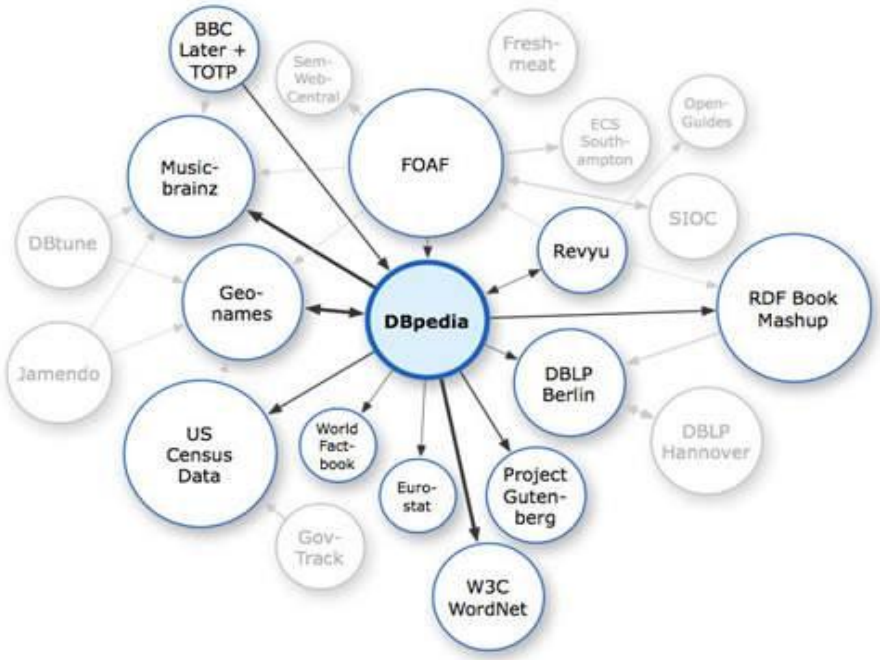


Figure 1.7 : L'Ontologie DBpedia . [04]

Le contenu de DBpedia [LET, JEN, ISE 15] est obtenu du moissonnage des pages et des mises à jours de Wikipedia. Le système de moissonnage à l'origine de DBpedia crée des triplets principalement à partir des cadres de résumés formatés, les « infoboxes», des pages de Wikipedia et en moindre proportion, du traitement du langage naturel des textes d'articles.



Figure 1.8 : Wikipedia avec infobox pour créer DBpedia. [05]

La base DBpedia est mise à jour annuellement (voir Tableau 1.1). Cette action est faite depuis la base DBpedia Live, miroir de DBpedia, qui est mise à jour directement à partir de Wikipedia.

La première version de DBpedia [LET, JEN, ISE 15] était extraite depuis la version anglophone de Wikipédia, les extracteurs d'informations n'étant conçus que pour être appliqués sur les infoboxes de cette version. Ce choix a eu pour conséquence que seules les entrées de la version anglophones bénéficiaient d'une représentation sous forme de données dans DBpedia .

Mais la création de données d'après une encyclopédie anglophone a rapidement montré des limites de couverture. Il a donc été décidé de poursuivre le développement d'éditions linguistiques de DBpedia.

En mars 2012, Une version entièrement francophone, agrémentée de fonctionnalités nouvelles, a été officiellement révélée. Elle est développée par l'Inria avec le soutien du Ministère de la Culture et l'expertise de l'association Wikimédia France.

Le but principal de cette base de connaissances est de combler ce manque de données gratuites dans la toile, aussi de développer les technologies (web sémantique) pour effectuer des recherches dans ces bases et/ou faire interagir et partager les données récupérées. DBpedia nous permet aussi de faire des recherches avancées sur la structure de Wikipedia. [06]

Années	Nombre Concepts (éléments)	Dont Personnes	Dont Places	Dont Organisations
2010	3.40 Millions	312 000	413 000	140 000
2012	3.70 Millions	416.000	526.000	190 .000
2014	4.58 Millions	1.450.000	735.000	241.000
2016	4.60 Millions	1.500.000	810.000	275.000

Tableau 1.1 : L'évolution de version Anglaise de DBpedia [09]

1.2 . Les Langages des Ontologies :

1.2.1 – RDF :

RDF (Resource *Description Framework)* est à la base des technologies du Web des données. Il s'agit d'un modèle de données simple pour la représentation de connaissances sur le Web dans des *documents RDF*.

Comme son nom l'indique, RDF sert à décrire des *ressources RDF*. Une ressource RDF est un concept ou un élément concret (toute sorte de chose, une personne, un pays, un évènement, ...). Chaque ressource RDF est – théoriquement – unique dans tout le Web des données. Les descriptions des ressources RDF sont composées de *triplets RDF*, (sujet, relation, objet), qui peuvent être représentés sous forme de graphe. Un triplet RDF (voir Figure 1.9) exprime une relation entre un sujet (la ressource décrite) et un objet. Autrement dit, un triplet décrit une propriété du sujet ayant pour valeur l'objet du triplet. La description d'une ressource correspond à l'ensemble des triplets qui contiennent la ressource en sujet.

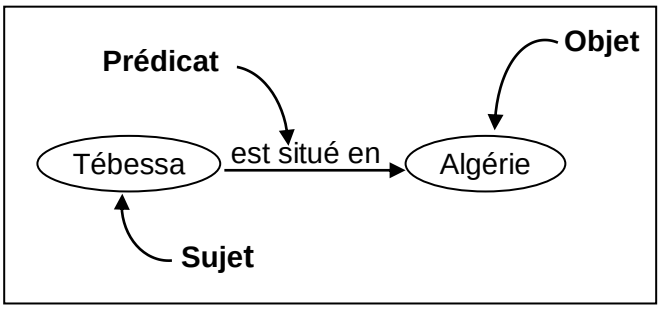


Figure 1.9 : Représentation d'un triplet RDF (sujet-relation-objet).

Sujet : est une ressource, soit identifiée par un URI, sinon correspondant à un nœud blanc.

Relation : est toujours une ressource identifiée (*URI*), aussi appelée "*prédicat*".

Objet : soit une ressource (identifiée ou non), soit une donnée brute, aussi appelée *littéral*.

Un document *RDF*, (voir Figure 1.10 et Figure 1.11), est également appelé " *ensemble de triplets RDF* " ou encore " *graphe RDF* " . Il peut en effet être représenté sous forme d'un multi-graphe orienté étiqueté.

< Ahmed><parent de ><Omar>
< Ahmed><parent de ><Lina >
< Ahmed><habite ><Algérie>

Figure 1.10 : Ensemble de triplets RDF.

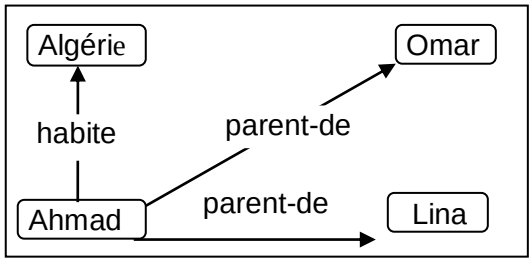


Figure 1.11 : Représentation graphique des triplets RDF du Figure 1.10

- **Les Syntaxes de RDF :**

Un document RDF peut être stocké dans des fichiers en suivant une des nombreuses syntaxes de RDF. Ces syntaxes permettent de représenter des documents RDF de différentes manières sans en changer le contenu de façons plus ou moins adaptées pour l'exploitation par la machine ou par l'humain.

La syntaxe RDF/XML est la première syntaxe à être apparue pour l'écriture de documents RDF. Elle a été introduite en même temps que RDF, basée sur XML et se place dans la continuité des technologies du Web. Son caractère très verbeux, hérité de XML, la rend peu adaptée à l'exploitation par l'humain. Les fichiers RDF/XML, dont un exemple est en Figure 1.12, utilisent les extensions ".rdf ".

```
<rdf:RDF>
<rdf:Descriptionrdf:about="http://www.exemple.com/rdf#Lina">
<ex:parentDerdf:resource="http://www.exemple.com/rdf#Ahmad"/>
<ex:parentDerdf:resource="http://www.exemple.com/rdf#Omar">
<ex:habiterdf:resource="http://www.exemple.com/rdf#Algerie"/>
</rdf:Description>
</rdf:RDF>
```

Figure 1.12 : Document représenté en Figure 1.10 au format RDF/XML.

- **Les Graphes Conceptuels (GCs) :**

Selon GOMEZ Carpio [GOM 10] , Le modèles des graphes conceptuels « G.C » introduits par Sowa en 1984 : « *est un modèle formel et expressif de représentation des connaissances d'un domaine d'application* ».

Les G.C est un langage intermédiaire entre des formalismes destinés aux traitements automatiques et le langage naturel.

Ils tiennent leurs origines des réseaux sémantiques de Quillian 1968 [GOM 10], qui depuis plusieurs extensions ont été proposées, aussi un graphe conceptuel n'a pas de sens propre, c'est au travers des réseaux sémantiques que les concepts et les relations sont rattachés au contexte. Le graphe conceptuel illustré par la Figure 1.13, correspond à la phrase « *Chaque employé est recruté par quelques managers à certaines dates* ».

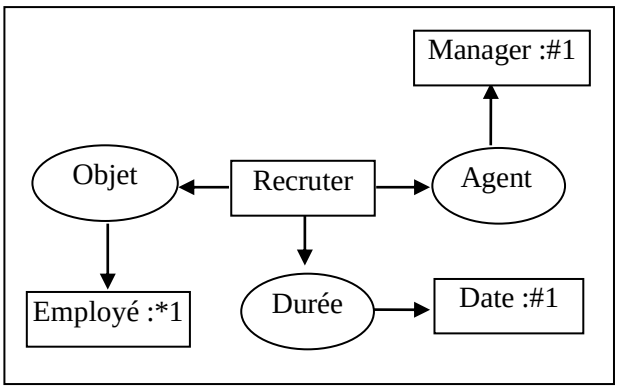


Figure 1.13 : Exemple de graphe conceptuel.

1.2.2- RDF(S) (RDF Schema) :

Les ontologies sont des ensembles de concepts permettant de définir des regroupements de ressources, les classes, de définir les relations utilisables pour les décrire, les propriétés, et de les hiérarchiser.

Le W3C a fourni en 2004 un premier langage basique RDFS (RDF *Schema*) , permettant de définir des ontologies simples. RDFS a ensuite été légèrement étendu en 2014 dans sa version 1.1, en même temps que RDF 1.1. Traditionnellement, on sépare dans un document RDF :

- ✓ les définitions des classes et des propriétés, qui forment l'ontologie,
- ✓ les définitions et mises en relations des instances des classes (aussi appelés individus), qui forment les faits du document.

Les bases RDF du Web des données peuvent utiliser plusieurs ontologies pour définir leurs données, cependant lorsque nous discutons des données d'une seule base, nous considérons l'union des ontologies utilisées dans la base comme formant « l'ontologie de la base ».

- **Les Classes :**

Une classe est un groupe de ressources ayant des propriétés similaires, Personne, Voiture, Pays, …… *etc.* On dit d'une ressource appartenant à une classe qu'elle est *instance* de la classe. Une instance d'une classe est une ressource liée par la relation rdf:type à une classe. Les classes définies en RDFS sont elles-mêmes des instances de rdfs:Class.

Les classes peuvent être organisées en hiérarchies pour indiquer des classes plus générales et des sous-classes plus spécialisées. Dans ces hiérarchies, on considère que les instances des sous-classes sont également instances des classes plus générales. Les hiérarchies de classes sont définies par la relation rdfs:subClassOf, tel qu'en Figure 1.14.

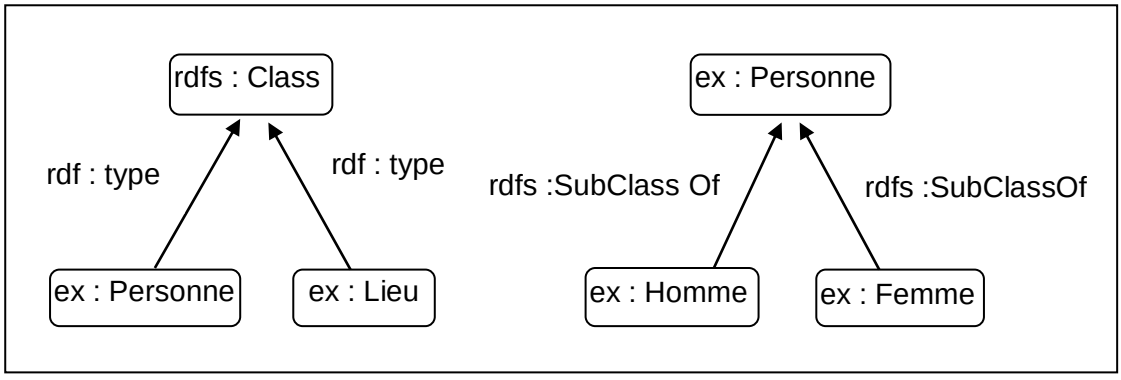


Figure 1.14 : Exemple de l'hiérarchie de classes.

Le langage RDFS définit des ressources utiles à la définition de classes, notamment :

- ✓ rdfs:Resource est la classe par défaut de toutes les ressources définies selon RDFS.
- ✓ tous les types de données des littéraux sont par défaut instances de la classe rdfs:- Datatype.
- ✓ toutes les relations sont instances de la classe rdf:Property. Nous considérons de façon équivalente les termes « relation » et « propriété ».

- ➢ **Les propriétés :**

Les propriétés sont les relations de triplet qui sont définies dans l'ontologie. Toute propriété d'une ontologie définie par le langage RDFS est instance de la classe rdf:Property. Tout comme les classes, les propriétés peuvent être organisées en hiérarchies par la relation rdfs:subPropertyOf, voir par exemple Figure 1.15. Une hiérarchie de propriétés définit des niveaux de généralisation ou de spécialisation entre relations. On considère que deux ressources liées par une propriété sont également liées par ses propriétés plus générales.

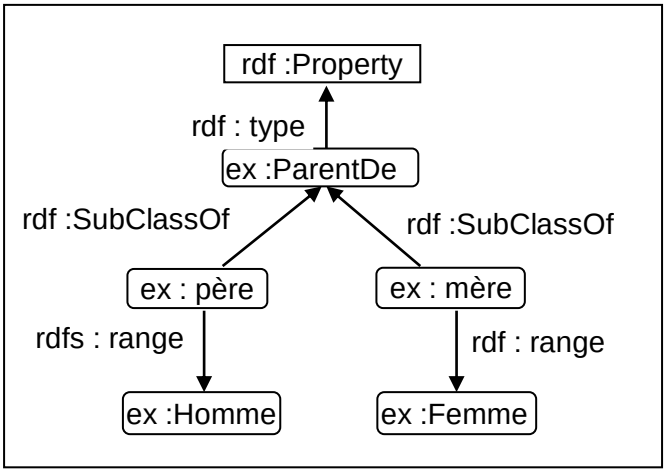


Figure 1.15 : Exemple de hiérarchie de propriétés

- ➢ **Les Individus :**

Un individu est une ressource instance d'une classe définie dans l'ontologie – en dehors des classes définies par RDFS – dont des exemples sont donnés en Figure 1.16. Pour un individu, la relation d'instanciation peut être traduite en langage naturel par « l'individu est un ... » ou « l'individu est une sorte ». Par définition, un individu est sujet d'au moins une relation d'instanciation, il est donc toujours représenté par un URI ou un nœud blanc, jamais par un littéral. Un individu instance d'une classe A est également appelé « individu de type A ».

Les individus sont les éléments de plus bas niveau d'un document RDF. En effet, les langages du W3C ne permettent pas de définir des relations de hiérarchisation entre individus. Selon le langage RDFS, toutes les ressources d'un document sont instances de rdfs:Resource. Ainsi, tous les individus d'un document ont toujours un type en commun.

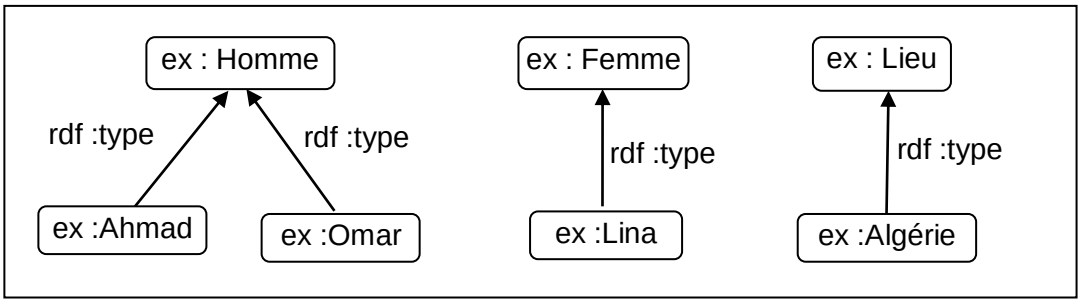


Figure 1.16 : Exemple d'instances définis en figures 1.14 et 1.15.

1.2.3- OWL :

Le langage OWL est apparu en 2004, en même temps que RDF et RDFS, il a ensuite été étendu en 2009-2010 dans la norme OWL 2 .

OWL fait étendre RDFS, il introduit dans les documents RDF une sémantique logique tirée des logiques de description. OWL apporte un langage plus riche pour la description de ressources en permettant la définition de relations entre les classes et entre les propriétés, comme l'équivalence, la différence, l'union, le complément, *etc*…., ainsi que des raisonnements suivants les logiques de description.

1.2.4 - SPARQL :

L'interrogation d'une base RDF se fait très généralement par le langage SPARQL – *SPARQL Protocol and RDF Query Language* – défini par le W3C pour la première fois en 2008 (4 ans après RDF), puis étendu en 2013. Ce langage est un langage d'interrogation de bases de connaissances – à comparer avec le langage d'interrogation des bases de données relationnelles SQL – adapté pour la structure sous forme de triplets des bases RDF.

Une requête SPARQL possède un entête et, le plus souvent, un corps de requête. L'entête contient le mot-clé de la requête – SELECT, ASK, INSERT, *etc.* – avec les éléments demandés en résultat de la requête. Le corps de requête contient un ensemble de « *triplets de recherche* ». Dans les triplets de recherche, un élément sujet, relation ou objet peut être remplacé par une variable, sous la forme d'un nom de variable préfixé du caractère ?. Un ensemble de triplets de recherche forme un *motif de graphe*. On appelle solution d'un motif de graphe un ensemble de données RDF qui forme un morphisme du motif. Ce morphisme substitue les variables du motif par des ressources de façon à ce que le motif de graphe et l'ensemble de données RDF soient identiques. Le corps d'une requête commence par le mot clé WHERE suivi d'un motif de graphe entre accolades ou de plusieurs motifs organisés par des opérateurs.

La syntaxe d'une requête d'interrogation SPARQL est comparable à celle du langage SQL pour les bases de données relationnelles.

Les requêtes d'interrogation SPARQL sont de 4 sortes différentes :

a- **Une requête SELECT** : extrait des résultats sous forme d'un tableau, comme avec SQL, contenant les ressources solutions à un motif de graphe donné et correspondantes aux variables présentes dans l'entête , tel que dans les Figures 1.17 et 1.18 si dessous .

```
SELECT ?adresse
WHERE {
ex:Jeanex:habite ?adresse .
        }
```

Figure 1.17 : Exemple de requête SPARQL SELECT.

```
---------------
| adresse   |
=========
| ex:Paris  |
```

Figure 1.18 : Résultat de la requête SPARQL SELECT

b- **Une requête ASK** retourne une valeur booléenne s'il existe au moins une solution à un motif de graphe donné, tel qu'en exemple de la Figure 1.19 (dont le résultat est " true ").

```
ASK
WHERE
 {
 ex:Paulex:parentDe ?enfant .
 }
```

Figure 1.19 : Exemple de requête SPARQL ASK.

c- **Une requête CONSTRUCT** extrait les solutions à un motif de graphe donné sous la forme d'un nouveau graphe RDF, tel que dans les exemples 1.20 et 1.21. Dans cette requête, les variables sont utilisées dans un motif de graphe en entête pour générer de nouveaux triplets RDF à partir des résultats.

```
CONSTRUCT {
  ex:Julieex:parentDe ?enfant .
  } WHERE {
  ex:Paulex:parentDe ?enfant .
  }
```

Figure 1.20 : Exemple de requête SPARQL CONSTRUCT.

```
ex:Julieex:parentDeex:Jean .

ex:Julieex:parentDeex:Marie .
```

Figure 1.21 : Résultat de requête SPARQL CONSTRUCT

d- **Une requête DESCRIBE** ce type de requête (voir Figure 1.22) retourne des informations sur les résultats de la requête. Les informations décrivant les résultats, qui vérifient les contraintes spécifiées dans la requête, sont retournées sous forme de graphe (voir Figure 1.23). Ce graphe est déterminé par le processus d'interrogation de SPARQL. L'utilisateur ne peut déterminer à l'avance la structure du graphe résultat ;

```
DESCRIBE ex:Paul
```

Figure 1.22 : Exemple de requête SPARQL DESCRIBE

```
SPARQL DESCRIBE.
ex:Paulex:habiteex:Paris .
ex:Paulex:parentDeex:Jean .
ex:Paulex:parentDeex:Marie .
```

Figure 1.23 : Résultat de requête SPARQL DESCRIBE

1.3. La Recherche d'Information :

1.3.1. Introduction :

Le domaine de recherche d'information a pris un grand essor avec l'apparition et le développement de l'internet, où une quantité exponentielle de documents de tout genre (texte, page web, image, vidéo ….) .

Les documents textuels peuvent êtres sans structuration (un texte plat) ou avec une structure tels que les documents XML. ce qui a aidé les systèmes de recherche à améliorer leurs résultats en prenant en compte cette structure.

Un axe récent dans le domaine de recherche d'information qui est la recherche sémantique a pris une place importante ces dernières années (voir Figure 1.24).

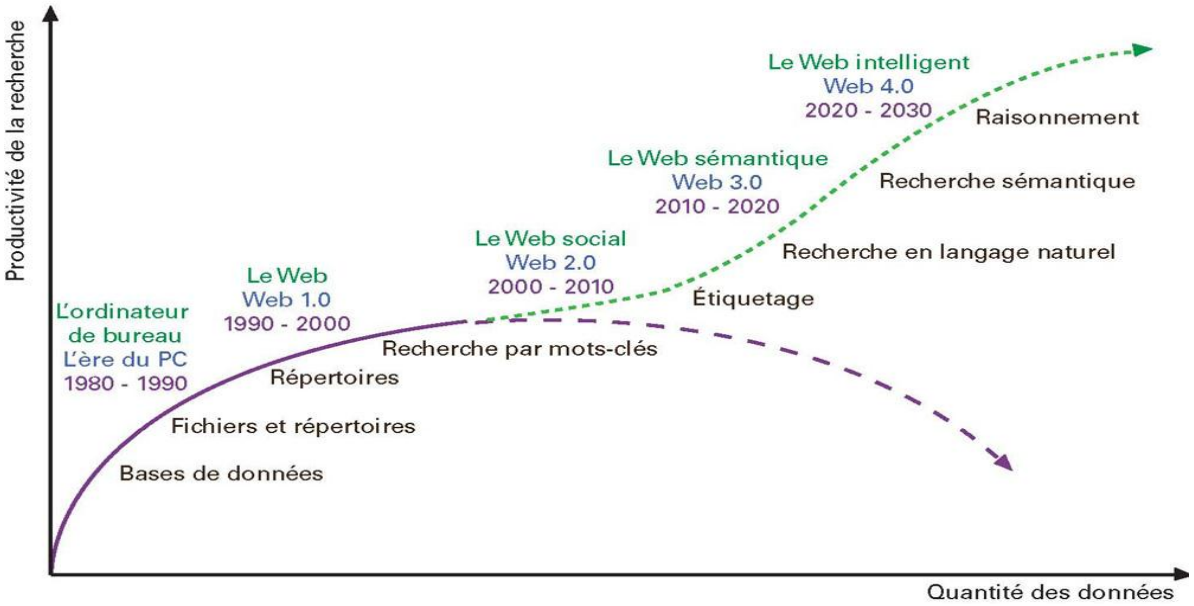


Figure 1.24 : Productivité de la recherche [07]

1.3.2. Définition :

La recherche d'information (RI) est définit par Christopher D. Manning et Prabhakar Raghavan [CHR,MAN,PRA 09] comme: " *c'est l'opération de trouver un matériels (habituellement un document) de nature non structuré (habituellement un texte) qui satisfait un besoin d'information à partir d'une grande collection (généralement stockée sur les ordinateurs)* ".

L'objectif des systèmes de recherche d'information est de retourner des informations (sous forme de document ou une partie de document) selon le besoin de l'utilisateur. Ce besoin est formulé sous forme d'une requête.

Le domaine de (RI) traite les concepts suivants : la représentation, le stockage et l'accès aux documents. La recherche d'information est devenue ces dernières années le moyen le plus utilisé pour l'accès à l'information, principalement en raison de l'augmentation considérable de documents dans le web.

Les systèmes de recherche d'information classique (voir Figure 1.25) sont composés de trois fonctions principales: l'indexation et la représentation des documents, la représentation de la requête utilisateurs, et l'appariement requête-document.

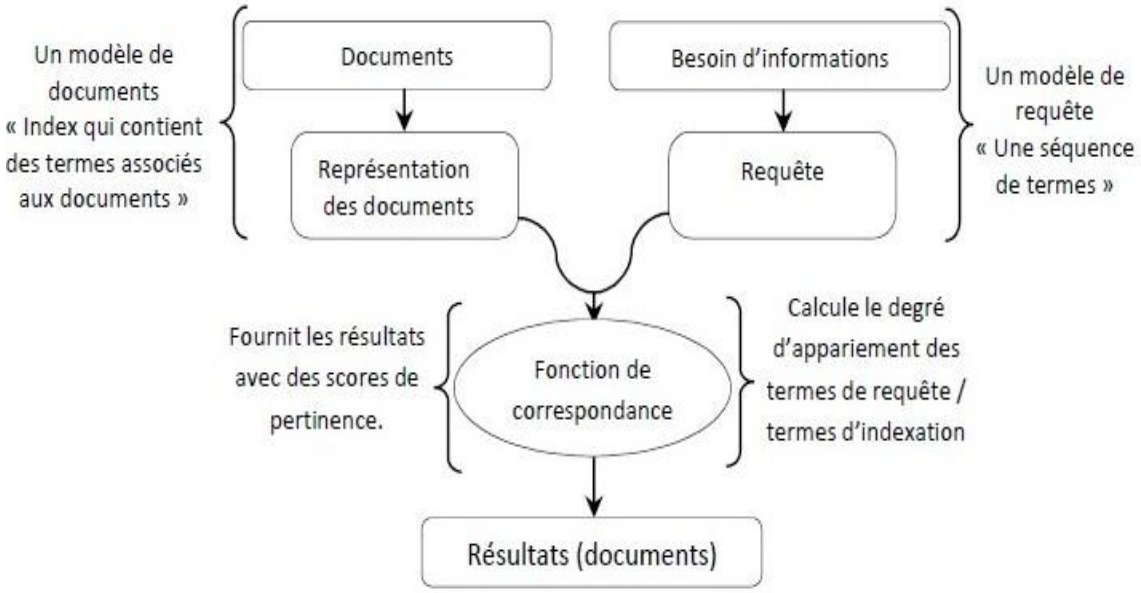


Figure 1.25 : Système de recherche d'information [BOU 11]

1.3.3. Modèles de recherche d'informations :

Un modèle de recherche d'information spécifie pour le système qui l'utilisera les

démarches à suivre pour l'accomplissement des étapes décrites précédemment. Ce sont la

représentation des documents et des requêtes, l'indexation, la recherche proprement dite,

l'évaluation des correspondances documents-requêtes et si besoin en est, la reformulation de la requête.

a. Le modèle booléen :

Le modèle booléen est le modèle classique le plus ancien pour la recherche d'information. Ce modèle est basé sur l'utilisation de la logique booléenne traditionnelle et la théorie des ensembles pour identifier les documents qui correspondent à une requête.

Les documents et les requêtes sont représentés sous formes des expression logique, ou les connecteurs logique sont utilisés (ET, OU, NON). Ce modèle permet de faire un appariement exact entre la requête et les documents en déterminant si les termes de la requête existent ou non (voir Figure 1.26).

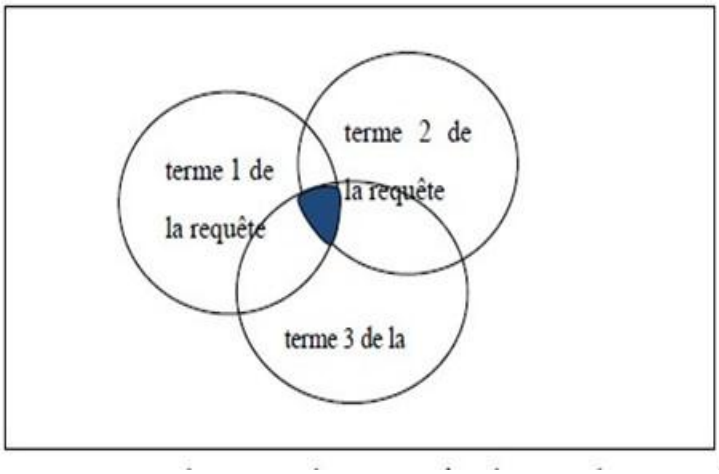


Figure 1.26 : Appartenance des termes d'une requête dans un document.

Ce modèle est simple et très facile à réaliser, mais son inconvénient majeur c'est qu'il ne retourne pas des documents pertinents qui correspondent partiellement à la requête, comme il ne permet pas de classer les documents selon leur degré de pertinence.

b. Le modèle vectoriel :

Le modèle vectoriel (voir Figure 1.27) est l'un des plus utilisé dans la recherche d'information. Dans ce modèle, les documents et les requêtes sont considérés comme des vecteurs dans un espace de terme.

Pour calculer la similarité entre le vecteur requête et le vecteur document, on utilise la fonction de distance euclidienne (similarité cosinus). Après le calcule de la distance entre chaque document et la requête, un document sera classé au premier rang s'il est plus proche du vecteur

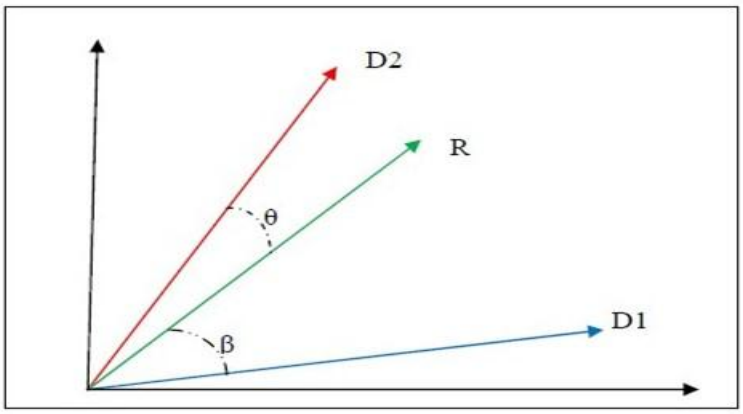


Figure 1.27 : Le modèle vectoriel.

Pour calculer la distance entre le vecteur du document D2 et le vecteur de la requête R, on doit calculer le cosinus de " θ " [BAZ 05] :

$$\text{Cos}\,\theta = \frac{D2 \,.\, R}{\bullet \,||D2|| \,.\, ||R||}$$

Ou : R est le produit scalaire entre les deux vecteurs (D2 et R)

et ||D2|| est la norme du vecteur D2.

c. Le modèle probabiliste :

Le modèle probabiliste de recherche est basé sur le principe de la probabilité. Il consiste à classer les résultats selon la probabilité de pertinence d'un document par rapport à une requête. Les travaux les plus connus et les plus exploités dans ce modèle son ceux qui se basent sur la notion de "principe de classement probabiliste" qui indique que les documents sont classés selon leur probabilité de pertinence. Cette probabilité est estimée sur la base de la distribution des termes pertinents et non pertinent dans les documents:

Propose deux classes de documents D_j: pertinents (classe R) ou non pertinents (classe $\bar{R}$) par rapport à une requête Q :

- P(R/D) est la probabilité que D est pertinent pour Q
- P($\bar{R}$ /D) est probabilité que D n'est pas pertinent pour Q.

Pour calculer le score d'un document le modèle utilise la formule suivante :

$$S = \frac{P(R / D)}{P(\bar{R} / D)}$$

1.3.4. La recherche sémantique d'information :

Les recherches massives engagées ces dernières années, ont abouti à la réalisation de plusieurs systèmes qui considèrent la recherche sémantique d'informations sous divers angles, qui dépendent des buts envisagés, des stratégies et modèles utilisées. La plus part des modèles proposés partagent des aspects communs qui se concentrent sur les suivants points:

- ✓ des outils d'annotation.
- ✓ des représentations sémantiques des connaissances de domaines.
- ✓ des stratégies de recherche (sémantiques, hybrides, …. etc.) basées ou non sur les agents.
- ✓ des algorithmes de classements des résultats obtenus selon leur pertinence.

Le problème majeur dans les systèmes de recherche classiques est que l'utilisation des mots-clés simples comme requêtes conduit au problème de silence, où des documents pertinents ne seront pas présentés comme résultats de la requête. Les documents peuvent décrire le même contenu et le même sens en utilisant des termes déférents.

Les chercheurs ont pensé à utiliser la notion du concept, au lieu d'un simple terme. Un concept regroupe autour de lui plusieurs mots clés. Pour cela, les ressources sémantiques sont devenues un composant principal dans les moteurs de recherche sémantique (voir Figure 1.28).

La ressource sémantique, intervient pratiquement, dans toutes les étapes de la recherche d'information: dans l'indexation des documents, dans l'analyse de la requête et dans l'étape d'appariement requête-documents.

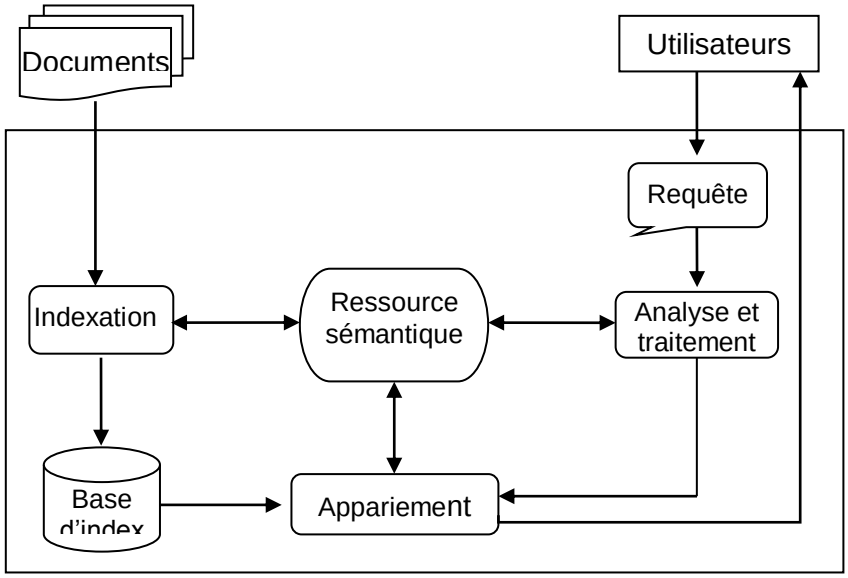


Figure 1.28 : Système de recherche sémantique de l'information [BOU 11].

a. Indexation sémantique :

Dans la recherche classique, l'unité textuelle dans l'indexation est représenté par un

mot- clés. Cela pose deux problèmes : l'ambigüité et la disparité des mots.

- L'ambigüité sémantique due au fait qu'un terme peut avoir plusieurs sens selon le contexte ou il se trouve.
- La disparité des mots est dû au fait que les termes qui sont syntaxiquement différents ils ont le même sens.

Ces deux problèmes mènent à l'effet du silence où des documents pertinents ne sont

pas retourné comme résultats pour une requête.

L'objectif de l'indexation sémantique est de désambigüiser le sens des termes. La tache consiste à représenter les documents et les requêtes par leurs sens. Dans ce type d'indexation l'unité textuelle sera le concept et non pas un simple terme.

Pour extraire le sens exact du terme, on doit utiliser les techniques de similarité sémantique pour trouver le meilleur concept qui représente le terme dans le contexte.

b. Traitement sémantique de la requête :

Le traitement sémantique de la requête d'un utilisateur peut résoudre deux problèmes, le premier est les erreurs dans l'orthographe de la requête par l'utilisation d'un dictionnaire et le deuxième est l'expansion automatique de la requête.

Pour l'expansion de la requête. il existe plusieurs méthode regroupé en deux techniques locale et globale. La méthode locale consiste à utiliser la notion de feedback, qui permet d'injecter de nouveaux termes dans la requête à partir des premiers documents retournés comme résultats.

La méthode globale permet de reformuler la requête en ajoutant d'autre termes proche sémantiquement aux termes initiaux de la requête (comme par exemple les synonymes...). ces termes sont calculés et extraits à partir d'une ressource sémantique.

c. Appariement sémantique :

L'appariement sémantique consiste à calculer la similarité sémantique entre les concepts de la requête et ceux des documents. Il permet de détecter les relations sémantiques entre les concepts, comme la relation de généralisation ou de spécialisation entre concepts. ces techniques permettent l'amélioration des résultats de recherche en retournant tous les documents qui ont une relation sémantique avec les concepts de la requête [BOU 11].

1.4. Conclusion :

Nous avons présenté dans ce chapitre un passage obligatoire sur les principes généraux du domaine qui englobe notre thème de recherche. On a débuté à définir le web sémantique et ces langages , puis on a enchainé par les ontologies avec un grand aperçu sur DBpedia , et pour finir, nous avons décrit la recherche d'information ainsi que les principales étapes d'un processus de recherche d'information .

2 CHAPITRE

Etat de l'Art

- Travaux connexes -

Introduction :

Après un détaille sur le web sémantique et les ontologies, nous visons dans ce chapitre à présenter un état de l'art spécifié aux différentes approches existantes représentant les travaux connexes du domaine, avec la citation de quelques modèles de moteurs de recherche sémantique connus par les experts.

2.1 .Description :

Un problème important des utilisateurs des bases de données, en général, est la formulation et la soumission des requêtes aux systèmes de gestion des données pour rechercher des réponses à leurs besoins. Pour formuler leurs requêtes, les utilisateurs ne disposent que d'une connaissance limitée des données à interroger. Ce problème devient encore plus complexe lorsque les utilisateurs se trouvent dans un environnement de bases de données distribuées avec des sources distantes et hétérogènes, dont les accès sont, très souvent, règlementés.

De plus, pour faciliter la communication et l'intégration des données entre les sources hétérogènes, des ontologies ont vu le jour. L'intégration de données est réalisée à partir des schémas des données représentés dans des ontologies. Cette évolution accroit également la complexité de l'interrogation et de l'exploitation de ces bases de données. Dans ce contexte de données volumineuses, distribuées, hétérogènes, dynamiques et liées, rend la tache de satisfaire l'utilisateur plus complexe.

Les systèmes d'interrogation existants offrent principalement aux utilisateurs les trois services suivants :

- ✓ l'interrogation avec des requêtes dans les langages tels que SQL ou SPARQL ;
- ✓ l'interrogation des données par des applications dédiées ;
- ✓ l'interrogation des données via des interfaces graphiques telles que des formulaires ou des langages graphiques de composition de requêtes qui sont plus intuitifs.

Les algorithmes d'interrogation tiennent compte des différents niveaux d'indexation offerts et du niveau de connaissances de l'utilisateur sur les ontologies gérées par la base d'ontologies. L'interrogation peut se faire de deux manières :

- **Interrogation en mode simple :** permet de faire des interrogations sans se soucier des champs sur lesquels va porter la recherche. Par défaut, le système reformule la requête de l'utilisateur en la propageant sur les différents champs de recherche disponibles.

- **Interrogation en mode complexe :** Ce mode d'interrogation est conçu pour une utilisation avec des connaissances a priori sur le contenu de la base d'ontologies. On spécifie de façon explicite les champs de recherche (termes, attributs, relations).

2.2. Détail des approches existantes :

2.2.1. Interrogation en mode simple (Approches Coopératives) :

L'objectif de ces approches est d'assister les utilisateurs dans l'interrogation des données. Ils ont été intégrées ensuite dans des systèmes d'interrogation tels que les systèmes de recommandation.

Ces approches permettent d'alerter les utilisateurs en cas de problème pendant l'exécution. Ces problèmes peuvent être :

- aucune réponse n'est retournée, cas des requêtes à réponse vide (empty answer query).
- un utilisateur, familier de la base de données et aux données, n'est pas satisfait des réponses retournées par la requête. On parle alors de réponses insatisfaisantes ;
- un utilisateur qui ne comprend pas pourquoi il a obtenu des réponses mais pas celles qu'il attendait, c'est un autre aspect des réponses insatisfaisantes ;
- un utilisateur est partiellement satisfait par les réponses retournées : réponses incomplètes.

Nous pouvant citer dans cette catégorie les approches suivantes :

a .Suggestion de requêtes :

La suggestion de requêtes aide les utilisateurs à formuler leurs besoins sous forme de requêtes syntaxiquement et sémantiquement correctes. Elle a initialement été proposée pour éviter aux utilisateurs de formuler des requêtes qui ne pourront pas être satisfaisantes à cause des fausses suppositions. Deux types de fausses suppositions peuvent conduire à l'insatisfaction des utilisateurs. Le premier type est l'ensemble des suppositions faites par les utilisateurs sur le schéma des données, on parle de présupposition. Le deuxième type est l'ensemble des suppositions sur les données que les utilisateurs pensent exister ou pas dans la base de données.

b. Interrogation par l'exemple :

Malgré l'aide apportée par les techniques de suggestion de requêtes dans la formulation des requêtes, il est montré que la conversion des besoins en requêtes est une tâche complexe. En effet, pour les utilisateurs n'ayant qu'une connaissance imprécise de leurs besoins, la formulation des requêtes correspondantes à ces besoins est très complexe. L'interrogation par l'exemple (Query by Example, QBE) permet d'alléger cette difficulté. L'approche d'interrogation par l'exemple a été initialement conçue par IBM comme un langage d'interrogation pour les données relationnelles. Le concepteur de cette approche, a eu l'idée d'un langage d'interrogation pour les non-programmeurs, afin de permettre aux utilisateurs de formuler leurs requêtes avec précision et d'éviter toutes fausses présuppositions. Cette nouvelle approche d'interrogation été conçu en primo pour les données relationnelles, et par la suite, elle été adaptée pour les autres types de données, notamment RDF et XML.

c. Explications sur les réponses d'une requête :

Les approches présentées dans cette section ont pour but de fournir aux utilisateurs des explications aux questions qu'ils peuvent se poser devant les réponses retournées par leurs requêtes. Ces approches étaient, initialement, destinées à traiter des questions provenant de l'insatisfaction des utilisateurs. Cette insatisfaction se traduisait par la question : (pourquoi n'ai-je pas obtenu la réponse X que j'attendais ?), plus connue sous le terme (Why-Not Anwsers) . Les utilisateurs se posent cette question lorsque l'ensemble des réponses retournées ne contient pas celles qu'ils attendaient. Ainsi, l'explication consiste à fournir aux utilisateurs les raisons pour lesquelles les réponses attendues ne se trouvent pas dans l'ensemble des réponses retournées. Par la suite, d'autres types d'explications de réponses de requête ont été développés. Il s'agit de l'explication de la présence de certaines réponses, inattendues, dans l'ensemble de réponses retournées (WhyAnswers ?), de l'explication des réponses vides (Why Empty Answers ?) et de l'explication des réponses pléthoriques (Why Too Many Answers ?). Cette section présente ces différentes approches d'explication des réponses d'une requête.

d. Raffinement de la requête par l'utilisateur (Users refinement) :

L'approche de raffinement de la requête par l'utilisateur (Users refinement) permet avant tout la réparation de la requête. Elle permet aux utilisateurs de redéfinir leurs requêtes de façon itérative jusqu'à être satisfait par les réponses retournées. Nous rappelons que

l'ensemble des réponses peuvent ne pas satisfaire les utilisateurs parce qu'il ne contient pas les réponses attendues ou parce qu'il contient des réponses indésirables. Les utilisateurs peuvent également être insatisfaits du nombre de réponses retournées.

Dans tous ces cas, la requête doit être réparée et le " Users refinement " peut le faire de deux façons :

- les techniques interactives font intervenir les utilisateurs et exploitent leurs retours.
- les techniques automatiques sont basées sur les requêtes, les données et une modélisation de l'utilisateur.

e. Redéfinition des requêtes :

La redéfinition d'une requête reformule les contraintes de cette requête afin de retourner des réponses satisfaisantes. Il existe trois types de reformulation de requête, ou modification de requête :

- ✓ la restriction de la requête : la requête issue de cette modification est plus restrictive que la requête initiale. Toutes ses réponses sont contenues dans celles de la requête initiale ;
- ✓ la relaxation de la requête : la requête issue de cette modification est moins contraignante que la requête initiale. Ses réponses incluent toutes celles de la requête initiale et en rajoutent ;
- ✓ l'approximation de la requête : la requête issue de cette modification est une approximation de la requête initiale. Elle contient une partie des réponses de la requête initiale et rajoute d'autres réponses qui ne sont pas inclues dans celles de la requête initiale.

Remarque**:** Il existe une relation entre l'approximation et la relaxation de requête. Certains travaux existants exploitent l'approximation d'une requête pour réaliser la relaxation de cette requête.

2.2.2. Interrogation en mode complexe (Approches basées sur la structure des ressources) :

Avec le développement des entrepôts de triplets RDF, un besoin d'interfaçage des moteurs SPARQL a émergé, puisqu'il est impossible pour un utilisateur final de maîtriser la complexité des schémas de ces connaissances : pour pouvoir exprimer une requête valide sur les connaissances du web sémantique, l'utilisateur doit connaître le langage SPARQL, les ontologies utilisées pour exprimer les triplets qu'il veut interroger ainsi que la "forme" des graphes RDF considérés.

Dans ce contexte, nous pouvons citer les approches suivantes :

- Le travail de Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., Sudarshan [BHA, HUL, NAK 02] démarche la recherche de chemin le plus court sur chaque élément correspondant afin de trouver une arborescence de jointure.
- L'approche DBExplorer de l'équipe de Chaudhuri [CHA, DAS, NAR 02] et l'approche Discover de l'équipe de Hristidis [HRI PAP 02], profitent du schéma de données pour calculer un ensemble de réseaux de jointure possibles.
- Les travaux de N.Athanasis et VC.Kotzinos [ATH VC KOT 04], proposent une recherche basée sur des graphiques ils ont construit des requêtes graphiques a travers la navigation et la sélection sur l'ontologie.
- Les auteurs comme COMPAROT C., HAEMMERLÉ O. & HERNANDEZ N. [COM, HAE, HER 10] ont proposé un moyen de construire des requêtes exprimées en graphes conceptuels à partir d'une requête utilisateur exprimée sous forme de mots-clés, puis ils ont étendu le système afin qu'il prenne en considération les relations exprimées par l'utilisateur entre les mots-clés utilisés dans sa requête, ensuite ils ont présenté de manière précise la notion de patron de requêtes.

2.3. Les approche selon le degré de formalité :

Depuis quelques années, plusieurs travaux se sont intéressés à l'utilisation des liens sémantiques entre les termes afin d'enrichir les requêtes. Ces travaux utilisent un vocabulaire structuré allant d'une simple taxonomie à l'utilisation d'ontologies complexes.

- ✓ D'après E. M. Voorhees [VOO 94], les méthodes statistiques ont moins de succès que les méthodes utilisant les relations sémantiques et linguistiques. Il propose une méthode d'enrichissement de requêtes basée sur les relations sémantiques entre les termes (synsets) dans Wordnet .
- ✓ Selon Zhiguo Gong, Chan Wa Cheang, and Leong Hou U. [GON, CHA, LEO 05] qui utilisent les relations hyperonymie, hyponymie et les relations de synonymie de Wordnet pour l'enrichissement de la requête.
- ✓ Selon Shuguang Wang and Milos Hauskrecht [WAN, MIL 10], qui proposent une nouvelle mesure pour calculer la similarité entre termes afin d'enrichir les requêtes. Une mesure de similarité est calculée en interprétant le graphe de termes comme un réseau électrique.
- ✓ Le travail de P. Sebillot V. Claveau [SEB, CLA 04] décrit comment les liens sémantiques entre les noms et verbes peuvent améliorer les résultats des systèmes de

recherche d'information, la requête étendue se compose donc des termes de la requête originale et des verbes qualia. Un lien qualia relie un nom et un verbe relies sémantiquement (par exemple couper est la fonction du terme couteau).

- ✓ Le travail de R. Navigli and P. Velardi [NAV,VEL 03] propose une approche qui consiste a extraire des concepts et relations d'une ontologie et les ajouter dans la requête initiale en se basant sur les liens entre les termes.
- ✓ Les chercheurs dans Aime X., Furst F., Kuntz P., and Trichet F [AIM,FUR,KUN,TRI 10] ont utilisé une ontologie personnalisée pour une recherche d'information sémantique en offrant a l'utilisateur une interrogation par concepts avec l'ensemble des termes décrivant le concept.
- ✓ Afin d'améliorer la recherche dans le Web sémantique et permettre a l'utilisateur de créer des requêtes simples, le système SPARK proposé par Zhou, Q., Wang, C., Xiong, M., Wang, H., Yu [ZHO,CHO,MIA,HAO,YON 07] traduit des mots clés d'une requête en requêtes Sparql.

La classification des approches proposée par KAUFMANN E. & BERNSTEIN A. [KAU, BER 10] (voir tableau 2.1) , selon un continuum du plus formel au plus permissif, en commençant par les langage de requêtes eux-mêmes :

Approches	Exemples	Observations
Les langages formels de requêtes graphes	SPARQL SeREQL	Inadaptés aux utilisateurs finals
Des moteurs qui assistent intégralement l'utilisateur pendant la formulation de la requête. (implémentent de simples fonctionnalités comme la coloration syntaxique et l'auto-complétion)	Flint SparQLed	Inadaptés aux utilisateurs finals
Approches qui reposent sur des interfaces graphiques (contribue également, en construisant des requêtes graphiques a travers la navigation et la sélection sur l'ontologie.)	specific-to-general pour SPARQL	Ne dépasse pas les limites d'utilisabilité des langages de requête formels.
Introduit la recherche par facette à base de requêtes (un paradigme combinant la recherche par facette et le requêtage)	Opérateur PICO dans le secteur de santé.	Facile à exploiter par la plus part des utilisateurs
Génération automatique des requêtes avec retour aux utilisateurs pour plus de raffinement	Moteur de recherche Evi	Plus adaptés aux utilisateurs finals

Tableau 2.1 : Classification des approches par KAUFMANNE. & BERNSTEIN A.

2.4. Modèles de Moteurs sémantique existants :

On peut citer quelques applications représentatives de la recherche sémantique à base des mots-clés :

- ✓ SemSearch [SEM 06]: a une petite interface structurée de requête mots-clés pour cacher la complexité de la recherche sémantique.
- ✓ Avatar SemanticSearch [SEM 06]: est un prototype de moteur de recherche qui exploite les annotations dans le contexte de la recherche par mot-clé classique.
- ✓ CORESE : Ce moteur est dédié à des applications de web sémantique communautaire ou d'entreprise. Le moteur Corese permet la recherche d'information basée sur des modélisations de connaissances, des modèles conceptuels ou des référentiels métier et organisationnel (voir figure 2.1).



Figure 2.1 : L'interface du moteur Corese [CHA, NAD, OLI,WIL 04]

- ✓ Evi: de la société Amazon, l'utilisateur peut enrichir l'outil avec ses propres connaissances, liens vers DBPedia, Wikipedia, Freebase, ... (voir figure 2.2)

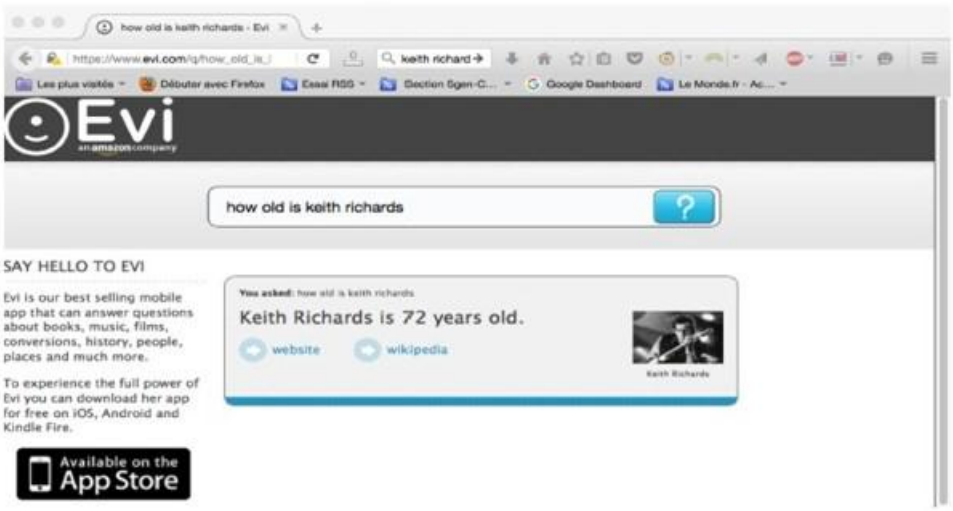


Figure 2.2 : L'interface du moteur Evi [08]

- ✓ Wolframalpha [09] : Contrairement à Google, Yahoo et consorts, WolframAlpha ne restitue pas des informations piochées sur le Web. Son rôle : compiler, croiser et présenter synthétiquement les données issues de bases de connaissances, dont celle de Mathematica, et d'autres sources communiquées à chaque résultat de recherche. Il fournit alors des « fiches pratiques » aux utilisateurs (voir figure 2.3)



Figure 2.3 : L'interface du moteur wolframalpha [09].

2.5 . Conclusion :

Nous avons vu au sein de ce chapitre un balayage sur les différentes approches représentées par des systèmes qui offrent aux utilisateurs des services remarquables afin de réussir à interroger les ontologies sur le web.

On a distingué deux modes d'interrogation par mots-clés (simple et complexe) suivi des travaux connexes justifiés par quelques exemples des moteurs de recherches spécialement sémantique .

3 CHAPITRE

La Conception

Introduction:

Ce chapitre présente notre choix de l'approche convenable pour faire fasse aux défis et a la problématique de notre sujet . On commence par l'architecture générale , description du problème et le détail des trois étapes suivi par des exemples explicatifs extraits de l'ontologie DBpedia .

3.1. Description :

Pour un aperçu global , notre système consiste à traduire ou transférer automatiquement des requêtes mots-clés en des requêtes formelles, afin de permettre aux utilisateurs finaux de se familiariser avec l'utilisation des mots-clés et d'utiliser ces mots-clés pour effectuer des recherches sémantiques. Nous avons adaptés les idées spécialement de l'approche SPARK [WAN,XIO,YU 07] et le travail remarquable du Docteur Bellatreche Abdelkrim et son équipe [LAT, LEH, BEN 09].

La recherche sémantique est un processus complexe qui devrait être achevé sur plusieurs étapes:

- Un processus d'annotation sémantique (Mapping des mots clés aux éléments de la Base de Connaissances BC)
- La phase de traitement de la requête (Construction des graphes requêtes)
- Une phase qui consiste à classer les résultats obtenus (Classement pour évaluer la pertinence.)

Après l'opération du prétraitement qui a comme rôle l'indexation automatique des entités de la base de connaissances pour obtenir un index très utile pour les étapes de notre travaille. L'approche appeler brièvement (*méthode de construction des requêtes formelles*), prend en entrée les mots-clés, et renvoie en sortie une liste classées de requêtes conjonctives SPARQL (voir figure 3.1).

Il est important de permettre aux utilisateurs d'effectuer des recherches sémantiques simplement en entrant des requêtes à base de mots-clés. Pour adapter la recherche par mots-clés à la recherche sémantique, nous avons à sur monter les obstacles suivants :

- ✓ Le fossé vocabulaire *:* Les utilisateurs du web traditionnel généralement n'ont pas de connaissances sur le contenu et la structure de la base de connaissances (annotations et ontologies sous-jacentes).

- ✓ Le manque de relations *:* les relations entre les ressources de la base de connaissances sont exigées pour être explicitement énoncées dans les requêtes formelles, qui sont souvent manquantes dans les requêtes mots-clés des utilisateurs.
- ✓ Classement des Requête: En raison de l'ambigüité de la recherche par mots clés, il peut y avoir de multiples requêtes formelles produites à partir d'une requête mots-clés.

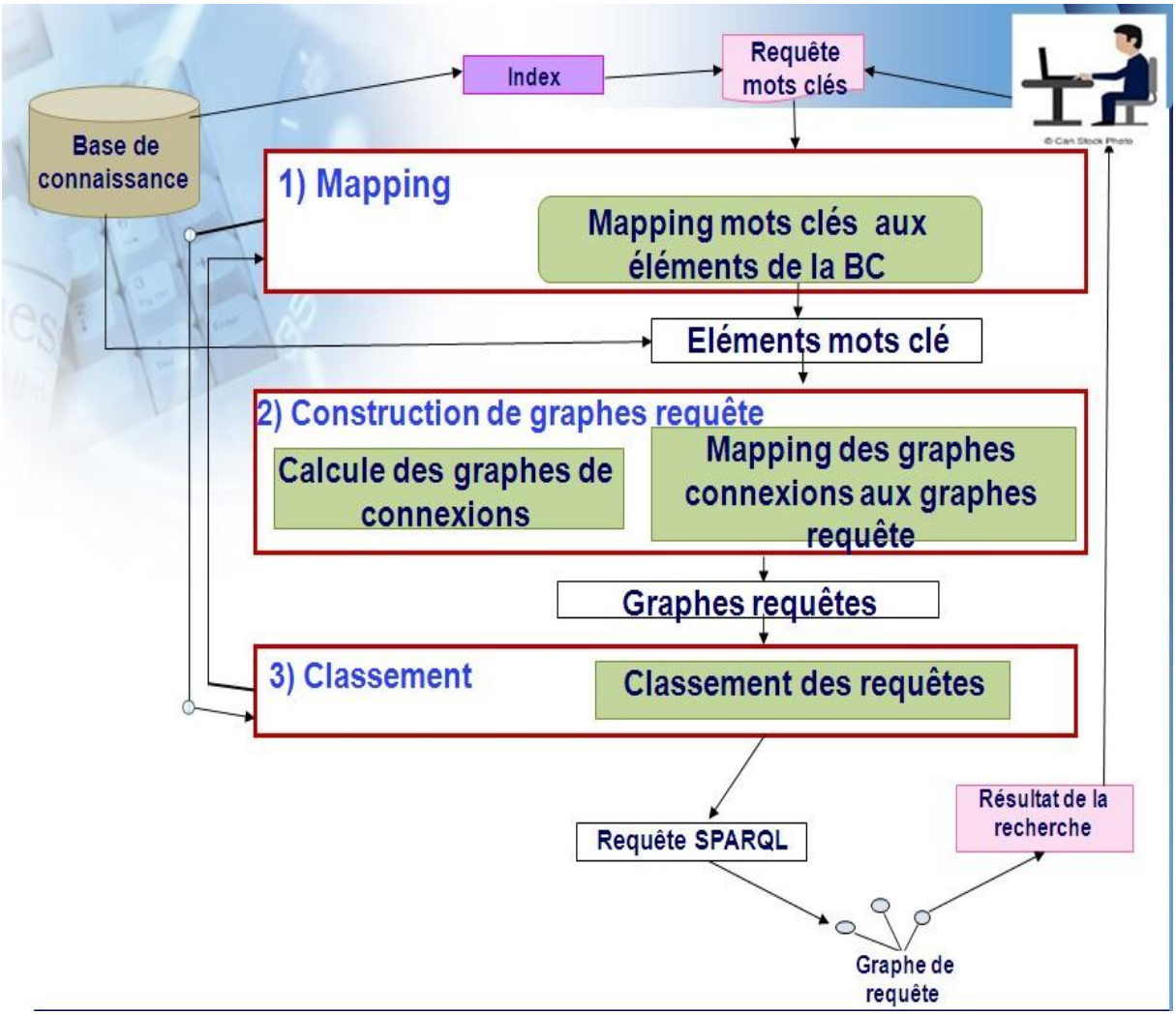


Figure 3.1 : Architecture générale de l'approche.

3.2. Présentation du problème :

Dans une base de connaissances RDF, une requête sémantique correspond à un graphe requête avec des nœuds objets et des arcs propriété. Ainsi, nous pouvons réduire le problème de la traduction des requêtes mots-clés en requêtes formelles conjonctives au problème de la

construction des graphes requêtes équivalent. Pour clarifier le problème, nous donnons la définition formelle comme suit:

- *Base de connaissances (B.C) D:(C, I, L, R, t)* est un graphe orienté *GD*

  où :

  C définit des ensembles de *concepts (classes)*

  R définit des ensembles de *relations (propriétés).*

  I définit des ensembles d*'individus* (r*essources*)

  L définit des *littéraux*.

  t une fonction $t:(C \cup I) \times (C \cup I \cup L) \rightarrow R$ définit tous les *triples* de *D.*

  En outre, nous utilisons le symbole éléments (entités) $\{e\}:\{C \cup R \cup I \cup L\}$ pour représenter toutes les concepts (classes), individus (ressources), relations (propriétés) et les littéraux.

- *Requête mots-clés : Qu*: est un ensemble de mots-clés *{k1,……..,kn}.*
- *Requête formelle : Qs: (C', R', I', L', V, t')* de *D* est un graphe *GF* subsumé par *GD.*

  où *:*

  V définit l'ensemble des *nœuds variable*.

  t' une fonction $t' (I' \cup C' \cup V) \times (I' \cup C' \cup V \cup L') \rightarrow R'$ définit l'ensemble des *triples* de *Qs.*

Exemple (1):

On navigant sur le web et spécialement on basant la recherche sur l'ontologie DBpedia , nous avons choisis une partie concernant les œuvres écrites (les livres) . Le graphe construit est représenté ci-dessous (voir figure 3.2):

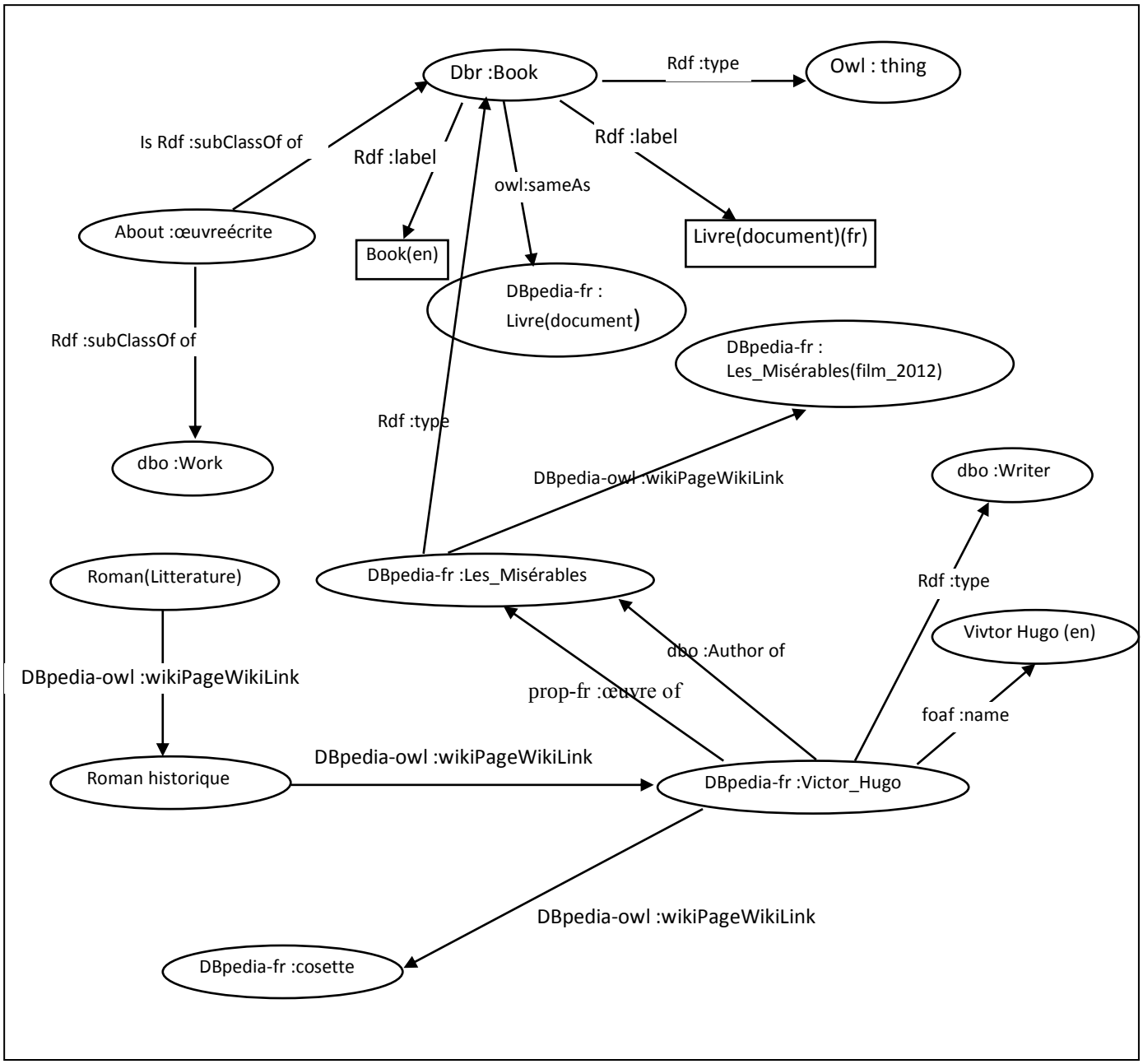


Figure 3.2 : Le Graphe Conceptuel extrait de l'ontologie DBpedia

Préfixes : (Les URIs)

Book → http://DBpedia.org/resource/Book

rdf :type → http://www.w3.org/1999/02/22-rdf-syntax-ns#type

rdfs :label → http://DBpedia.org/resource/Book

owl :thing → http://www.w3.org/2002/07/owl#Thing

DBpedia-Fr:Livre_(document) → http://fr.DBpedia.org/resource/Livre_(document)

rdf :label → http://www.w3.org/2000/01/rdf-schema#label

owl :sameAs → http://www.w3.org/2002/07/owl#sameAs

DBpedia-owl :WikiPageWikiLink → http://DBpedia.org/ontology/wikiPageWikiLink

Roman(litterature) → a#subClassOfhttp://DBpedia.org/ontology/WrittenWork

Roman(litterature)→ http://fr.DBpedia.org/resource/Roman_(litt%C3%A9rature)

Roman historique → http://fr.DBpedia.org/resource/Roman_historique

DBpedia-Fr :Victor_Hugo → http://fr.DBpedia.org/resource/Victor_Hugo

DBpedia-fr :Les_Misérables → http://fr.DBpedia.org/resource/Les_Mis%C3%A9rables

DBpedia-fr :Les_Misérables(film_2012)→

http://fr.DBpedia.org/resource/Les_Mis%C3%A9rables_(films,_1934)

Is Rdf :subClassOf of → http://www.w3.org/2000/01/rdf-schema#subClassOf

Rdfs :subClassOf → http://www.w3.org/2000/01/rdf-schema#subClassOf

About :œuvreécrite → http://DBpedia.org/ontology/WrittenWork

dbo :Work → http://DBpedia.org/ontology/Work

dbo :Writer → http://DBpedia.org/ontology/Writer

foaf :name → http://xmlns.com/foaf/0.1/name

Is prop-fr :œuvre of : → http://fr.DBpedia.org/property/oeuvre

DBpedia-fr :cosette→ http://fr.DBpedia.org/resource/Cosette

Is dbo :Author of → http://DBpedia.org/ontology/author

3.3 - Etapes du système :

3.3.1- Mappage des mots clés aux éléments de la B.C :

- **Définition** :

Le mappage *(mapping)* en informatique est l'association des données appartenant à un ensemble (modèle logique de données, base de données de production, champ source) avec les données appartenant à un autre ensemble (modèle physique de données, entrepôt de données, champ cible), de manière que les données du premier ensemble puissent se substituer à celles du second ensemble, ou encore que l'on puisse passer harmonieusement des premières aux secondes.

Les recherches montrent que le fait de comparer deux choses produit un savoir, Miner [LAT,LEH,BEN 09] écrit « *Il est manifestement impossible de comparer ce qui est identique. Des différences doivent exister ou alors nous identifions plus que nous comparons. De la même façon, si les différences sont trop grandes, la comparaison devient infaisable, les résultats logiques ou pratiques ne satisfont pas*.» Ainsi, il semble que comparaison et similarité soient étroitement liées.

Ce qui est important dans le mappage est la similarité sémantique qui exprime une liaison entre deux concepts est une capacité abstraite de l'homme, les machines n'ont pas la faculté de l'interpréter. Il est évident pour un humain que les concepts « *stylo* » et « *papier* » sont liés beaucoup plus que « *température* » et « *chaise* ». Cet état de fait, difficile à formaliser sans recours aux ressources sémantiques : les ontologies permettent de montrer les liens entre les concepts (hyperonymie, antonymie, etc.).

Le calcul des mesures de similarité entre concepts d'une ontologie, constitue aussi un autre défi auquel diverses solutions existent et qui sont classées en deux grandes catégories :

- ✓ Approches basées sur la structure de l'ontologie (edge counting): C'est une technique de calcul basée sur les liens taxonomiques « est un » « is a ». comme exemple " mesure de Wu-Palmer ".
- ✓ Approches utilisant le contenu informatif des concepts : comme exemple " mesure de Resnik" .

Dans cette étape (voir figure 3.3): nous détectons le meilleur concept pour chaque terme. Chaque terme peut avoir plusieurs sens, nous allons essayer de déterminer le meilleur sens (concept) dans le contexte (requête utilisateur) ou se trouve ce terme. Pour cela, une opération de désambigüisation est nécessaire.

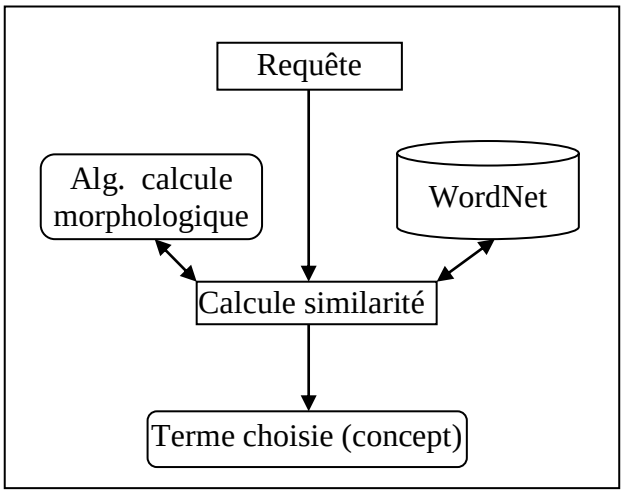


Figure 3.3 : Calcule de similarité.

- **Le Détail :**

Le mappage (*mapping*) comme étape représente " *une cartographie temporaire* " , son but est de trouver les éléments de la B.C. (concepts, individus, relations et littéraux) correspondants à chaque mot clé de la requête de l'utilisateur. Les noms et les étiquettes (labels) des éléments de la B.C. utilisées pour le mappage.

Dans notre système, deux types de méthodes de mappage peuvent être utilisées:

- ✓ Le mappage morphologique: emploie les techniques de comparaison de chaîne de caractères, tels que le stemming, Sub-String et Levenshtein-distance pour trouver les mots morphologiquement semblables, Par exemple : « author » et « author of ».
- ✓ Le mappage sémantique: se base sur la similarité sémantique et il utilise principalement des dictionnaires généraux comme WordNet [10],, afin de trouver les mots sémantiquement approprié (les synonymes, …… etc.), Par exemple : « author » et « writer ».

Nous assignons un score prédéfini ***Sm**(e) : (Sm(e)* ∈ *[0,1])* à chaque méthode de mapping pour déterminer la qualité du mapping. En règle générale, le score du mapping direct est supérieur au mapping à base des synonymes. En conséquence, le mapping des mots clés aux éléments de la B.C associe à chaque terme de la requête mots-clés un ou plusieurs sens selon la base de connaissances. En particulier, le mapping peut être définie comme fonction $f : Qu \rightarrow D$qui pour chaque terme de la requête mot-clés retourne un ensemble des éléments de la B.C.
(Pour des fins pratiques le moteur de recherche Lucene [11], est effectivement utilisée comme moteur d'index et de recherche). C'est à dire, les noms et les étiquettes des éléments de la BC. sont indexées, et en utilisant la fonction de recherche de Lucene, une requête est générée pour chaque mot-clé entré. Le moteur retourne les éléments de la B.C classées en fonction de leurs similarités syntaxique aux mots clés respectifs. Ces éléments appelés « éléments mots clés » ***Dm*** *: Dm = {f(ki) / Qu=(k1,....., Kn)}* , seront alors introduit dans l'étape de la construction des graphes requêtes.

Suite de l'Exemple (1):

On exploite le même exemple de la Figure 3.2 , et nous prenons la phrase suivante (en anglais), qui représente une requête d'un utilisateur :" **Author Miserable book** ". Nous avons trois noms qui ont des synsets dans WordNet . T = { Author, Miserable, Book }.
Chaque terme de T a plusieurs sens (synsets) dans WordNet . La figure 3.4 montre le résultat de mappage entre les éléments de T et les éléments dans le graphe de la figure 3.2. les valeur des similarités trouvées sont présenté dans le tableau 3.1.

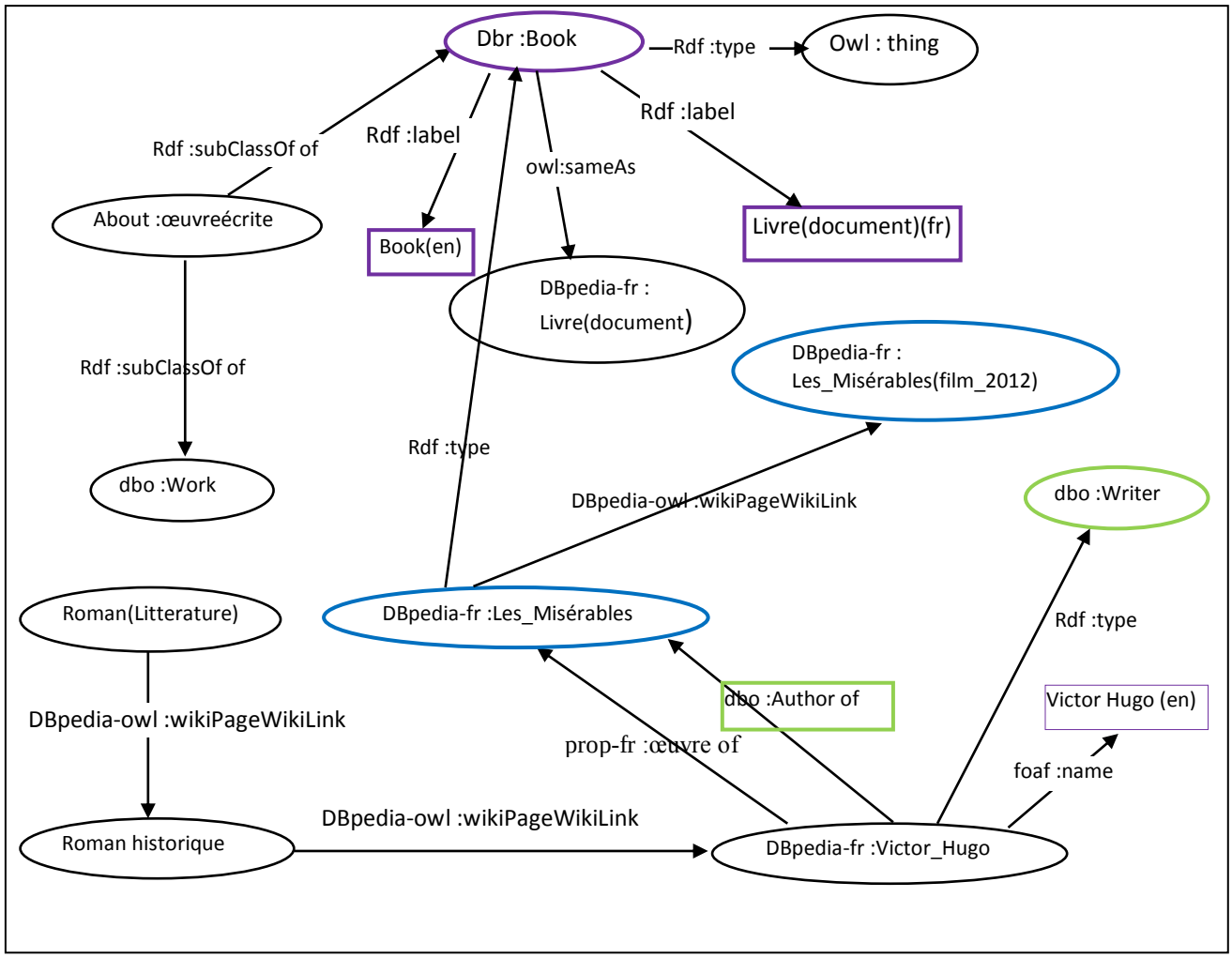


Figure 3.4 : Exemple de mapping

Mots clés	**Ressource**	**Mesure utilisées**	**Valeur de similarité**
Book	dbr :Book (c)	Levenshtein	1.00
Book	Book(en) (L)	Levenshtein	1.00
Misérable	dbfr :Les_Misérables(film_2012)(I)	TagLinkToken	0,49
Misérable	dbfr : Les_Misérables (I)	JaroWinkler	0.32
Author	dbo :Author of (P)	MongeElkan	0,66
Author	dbo :Writer (P)	WordNet	0,37

Tableau 3.1 : Les valeurs des similarités trouvées entre les mots clés et les ressources**.**

3.3.2 . Construction des Graphes Requête :

Cette opération représente la construction des graphes requête candidats avec les éléments mots clés trouvés dans l'étape précédente. Tout d'abord, les éléments mots clés sont répartis en différentes ensembles de requêtes. Le but de cette répartition est d'attribuer un sens à chaque requête formelle par l'énumération de toutes les combinaisons possibles des différents sens pour chaque terme ***ki*** de ***Qu***. Ensuite, un certain nombre d'algorithmes sont appliqués pour construire de possible graphes requêtes pour chaque ensemble de requêtes.

a- Calcul des graphes de Connexions :

A ce stade , l'exploration des relations se limite aux relations de type: (i , c) , (i_1, R, i_2) et (i , U, j) , où : I, i_1, $i_2 \in \mathcal{I}$, $j \in \mathcal{L}$, $c \in C$, $R \in \mathcal{R}$ et $U \in \mathcal{U}$

avec : $\mathcal{I}$(Individus), $\mathcal{L}$(littéraux), C (Concepts), $\mathcal{R}$ (propriétés d'objets) et $\mathcal{U}$ (propriétés de données).

Par l'utilisation de ces axiomes, nous explorons tous les éléments de la B.C liées aux éléments de ***Dm*** identifiés dans la première étape selon l'algorithme (1) (Figure 3.5)

Le processus du calcule des graphes de connexions se déroule comme suit :

- **<u>Explorer les liens entre les éléments de la B.C</u>** :

Elle comprend la traversée des voisins pour chacun des éléments de ***Dm***. Ensuite, selon le type de l'élément particulier ***em*** $\in$ ***Dm***, différents parcours sont effectués pour construire un graphe connectant ***em*** à tous ces voisins sur une distance spécifique ***d***. Soit un concept, tous les individus sont recherchés par l'intermédiaire de l'axiome ***(i, c)***. Compte tenu une relation, les axiomes **(i_1, R, i_2)** et **(i, U, j)** sont utilisées pour naviguer vers les individus et les littéraux, respectivement.

La Figure 3.5 montre le pseudo-code de l'algorithme pour le passage récursif d'un individu ou d'un littéral particulier à ses voisins concepts, individus et littéraux. La valeur de ***d*** est réduite de (**1**) à chaque étape de récusions pour assurer à ce que ce passage sera limité à une certaine distance. En fin de compte, nous obtenons un graphe ***g*** contenant tous les éléments de ***D*** qui ont une distance graphique qui ne dépasse pas ***d*** au moins à un des éléments de ***Dm***.

Ainsi, quelques éléments découverts peuvent vraiment ne pas être nécessaires pour connecter des éléments de ***Dm***. Par conséquent, à partir de ce graphe, on choisis seulement les chemins où le premier et le dernier sommet correspond est un élément de ***Dm***. En particulier,

Algorithme Exploration-BC (Dm, d) :

Entrée Dm ensemble des éléments mots clés et d distance du passage.
Sortie graphe contenant tous ou certaines éléments de D.
Initialisation d'un nouveau graphe vide g.
Pour e ∈ Dm
Faire Si e est un concept
Alors Pour Tous i étant instance de e
Faire Passé (i, d, 0, g)
Sinon Si e est une propriété d'objet (Individu-Individu)
Alors Pour Tous i, j avec <i, e, j>∈ D
Faire Passé (i, d, 0, g), Passé (j, d, 0, g)
Sinon Si e est une propriété de données (Individu-Littérale)
Alors Pour Tous i, j avec <i, e, j>∈ D
Faire Passé (j, d, 1, g)
Sinon Si e est un Individu
Alors Passé (e, d, 0, g)
Sinon Si e est un Littérale
Alors Passé (e, d, 1, g)
Retourné **g.**

Figure 3.5 : Algorithme (1) Exploration de la B.C [LAT,LEH, BEN 09]

L'algorithme précédant fait des appelles à l'algorithme (Passé) présenté ci-dessous **:** (Voir figure 3.6):

Algorithme Passé (i, d, t, g) :

Entrée i un individu ou un littérale à explorer, d distance du passage, g un graphe intermédiaire.

Sortie graph g mis à jour contenant les éléments connecté à i sur une distance d.

Si i n'est pas marqué comme visité et d > 0

Alors Marquer i comme visité dans D

Si t=0

Alors Ci :={c/ i instances de c}

Ajouter Arête (i, type, c) dans g pour tout c ∈ Ci

P :={(i, p, j) / <i, p, j>∈ D}

Pour Tous (i, p, j) ∈ P

Faire Si j n'est pas marqué comme visité dans D

Alors Ajouter nouveau Arête (i, p, j) dans g

Si j est un individu

Alors Passé (j, d-1, 0, g)

Sinon Passé (j, d-1, 1, g)

Si t=1

Alors P :={(j, p, i) / <j, p, i>∈ D}

Pour Tous (j, p, i) ∈ P

Faire Si j n'est pas marqué comme visité dans D

Alors Ajouter nouveau Arête (j, p, i) dans g

Si j est un individu

Alors Passé (j, d-1, 0, g)

Figure 3.6 : Algorithme (2) Passé [LAT, LEH, BEN 09]

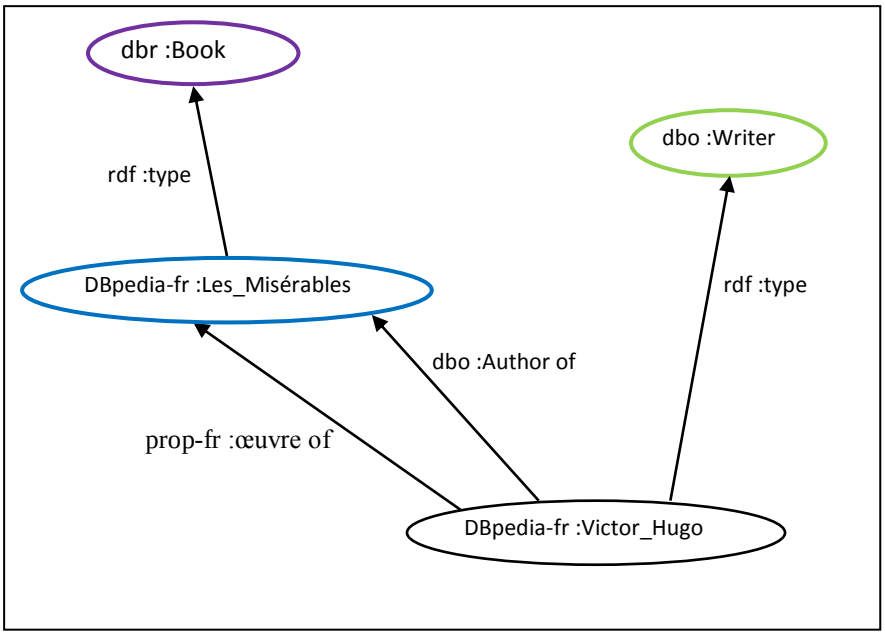


Figure 3.7 : Graphe résultat d'application de l'algorithme d'exploration sur le graphe de la Figure 3.4

Exemple : Si on utilise l'algorithme d'exploration avec comme entrées l'ensemble Dm={« dbr :Book », « dbfr : Les_Misérables », « dbo :writer » } et le graphe de la Figure 3.4 , alors on obtient le graphe présenté dans la Figure 3.7 . Par contre si on remplace « dbfr : Les_Misérables » par « dbfr :Les_Misérables (film_2012) » alors aucun graphe sera trouvé .

- **Calcul des Graphes de Connexions**:

Une requête peut être dérivée quand tous les éléments ***Dm*** identifiés dans la première étape sont connectés. En fusionnant tous les chemins ***P*** calculés dans l'étape précédente, nous obtenons toutefois un graphe qui peut contenir plusieurs différents sous-graphes connectant tous les éléments de ***Dm***. Le sous-graphe connectant les éléments de ***Dm*** sont calculés par une procédure récursive montrée dans l'algorithme de la Figure 3.8 . L'entrée de l'algorithme est l'ensemble des chemins ***P*** calculés précédemment. La récusions commence par la sélection d'un certain chemin connectant deux sommets arbitraires et entre d'autres récusions pour ajouter des sommets supplémentaires. De cette façon, tous les sous-graphes en forme d'arbre possibles connectant des éléments de ***D*** sont déterminés.

Algorithme Calcule-graphe-connexion (P, C, R, G, g) :

Entrée l'ensemble de chemins P calculer par DFS pour tous les éléments mots clés de Dm

Sortie Tous les sous-graphes connectant les sommets de Dm

```
Si G= Ø
  Alors Pour {i, j}  ⊆R
        Faire G = Nouveau Graphe
        Pour chaque chemin p entre i et j (calculer par DFS)
              Faire ajouter (i, p, j) dans G
              Calculer-graphe-connexion (P\p, C∪{i, j}, R\{i, j}, G)
  g=g ∪ G
  Sinon Pour i ∈  R
        Faire Pour j ∈ C
              Faire Pour chaque chemin p entre i et j
                    Faire ajouter (i, p, j) dans G
                    Calculer-graphe-connexion (P\p, C∪{i}, R\{i}, G)
```

Figure 3.8 : Algorithme (3) calcule des graphes de connexions [LAT,LEH, BEN 09]

Suite de l'Exemple: Le premier graphe de connexions (voir figure 3.9):

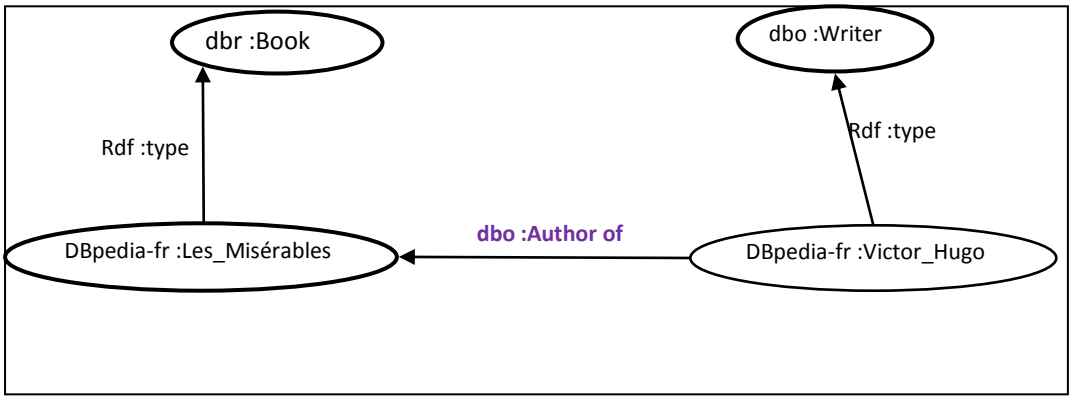


Figure 3.9 : Graphe de Connexions 1

Le deuxième graphe de connexions (voir figure 3.10) :

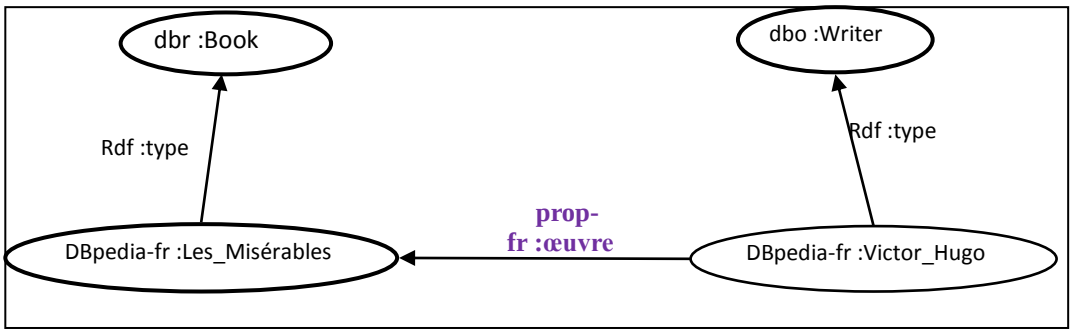


Figure 3.10 : Graphe de Connexions 2

b- Mappage des graphes de Connexions aux graphes requête :

Enfin, Chacun des graphes de connexion ***Gc*** trouvé précédemment est ensuite traduit en un graphe requête comme suit :

Une arête dans ***Gc*** de la forme *type* ***(vi, vc)*** (représentant la connexion *(i, c)*) est considérée comme arête dans le graphe requête de la forme *Type****(x, C)***,

où ***:*** ***vi*** est un sommet construit au moyen d'un individu,

vc est construit en utilisant un concept,

x est un individu ou une variable.

Lorsqu'un individu de ***vi*** correspond à certains éléments ***em*** $\in$ ***Dm***, il est considéré comme une constante, sinon une variable est utilisé pour ce terme. Comme le même individu peut être utilisé dans beaucoup d'arêtes, la même variable doit être utilisée pour le même individu. Les arêtes construit avec les axiomes de la forme ***R (vi, vj)***, où ***vi*** **(*vj*)** est construit en utilisant soit un individu ou un littérale (connexions (i_1, R, i_2) *et (i, U, j))*.

Ces arêtes sont considérés comme arêtes dans le graphe requête de la forme ***R(x, y)***, où un sommet construits au moyen d'un individu est associé à une variable ou une constante comme décrit ci - dessus. Lorsque ***vi*** **(*vj*)** est construit en utilisant un littérale, il est tout simplement considéré comme nœud associé à une constant.

Dans notre exemple de la Figure 3.7, il existe deux graphes de connexions donc deux graphes requêtes.

Le premier graphe de requete (voir figure 3.11) :

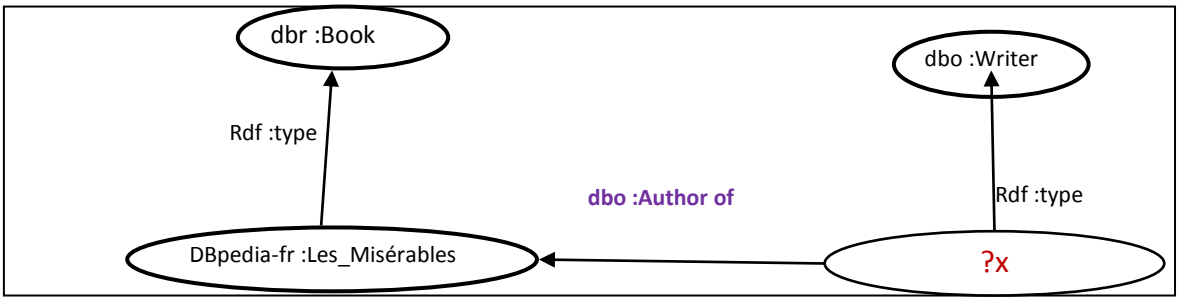


Figure 3.11 : Graphe de requête 1

Le deuxième graphe de requête (voir figure 3.12) :

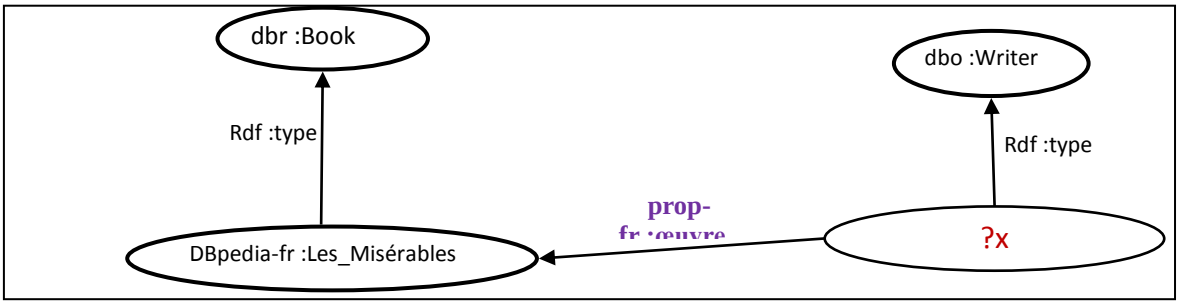


Figure 3.12 : Graphe de requête 2

Puisque SPARQL est un langage d'interrogation basé sur les motifs de graphes, il est facile de convertir un graphe requête en une requête SPARQL (voir figure 3.13 et figure 3.14)

```
SELECT    ? x
WHERE  {
  ? x   Type   dbo:Writer
  ? xdbo :Author_ofDbpedia-fr :Les_Misérables
Dbpedia-fr :Les_Misérablesrdf :type    dbr :book
 }
```

Figure 3.13 : La requête Sparql du Graphe Requête (1)

```
SELECT    ?x
WHERE  {
 ? xrdf:type   Writer
? x   oeuvreDbpedia-fr :Les_Misérables
Dbpedia-fr :Les_Misérablesrdf :type    dbr :book
  }
```

Figure 3.14 : La requête Sparql du Graphe Requête (2)

3.3.3. Classements des requêtes :

Après l'étape de la construction des graphes requêtes, plusieurs requêtes formelles candidates peuvent être produites à partir de la requête mots-clés initiale correspondant à toutes les interprétations possibles des mots clés. Un problème se pose: comment prendre la requête formelle qui correspond aux besoins de l'utilisateur? Le classement des requêtes est utilisé pour résoudre le problème (voir figure 3.15) .

Le calcule du score des réponses a été largement discuté dans le domaine de la recherche d'information. Dans le contexte des données structurées en graphe, des mesures de proximité sémantique ont été proposées dans la littérature (une douzaine) utilisant des structures de réseaux sémantiques ou hiérarchiques :

- Mesure basée sur le chemin (path based measures) entre les deux concepts à comparer telles que définies par Rada, Leacock ou Jiang en 1997.
- Mesure basée sur la notion de contenu d'information (Information Content ou IC) telle que celle définie par Wu et Palmer et Resnik.
- Mesure basée sur une combinaison du chemin et du contenu d'information (Mesure hybride) définie par D. Lin en 98.

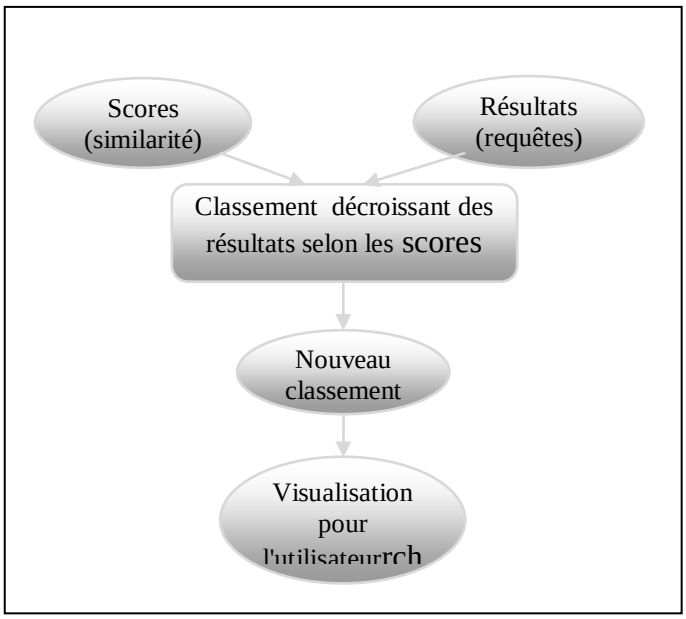


Figure 3.15 : Classement des requêtes.

A chaque graphe de requête est associé un score qui permet de le pondérer. Les graphes sont ensuite convertis en requêtes et exécutés dans l'ordre décroissant de ces scores. Ce score **S** est donné par la formule :

$$S = \sum_{i=1}^{n} Sei$$

Avec $\mathbf{Se_i}$ le score du l'élément i (valeur de similarité trouvée dans le procédure de mappage sinon il est égale à 0). La figure 3.16 présente l'algorithme de classement des requêtes.

Algorithme de Classement (R , Rc) :

Entrée R ensemble des requêtes trouvées (Ri / Ri ∈ R)

Sortie Rc liste des requêtes classées selon leurs scores.

Initialisation du score et la somme som.

Pour tous élément i ∈ Ri

Faire : som := som + Sei

positionner Ri dans Rc selon le score.

Figure 3.16 : Algorithme (4) - Classement des résultats.

Suite de l'Exemple (1) : on prenant compte des résultats de notre exemple durant l'étape (1) et (2) , on peut appliquer l'algorithme de classement pour les deux requêtes et nous aurons les résultats illustrés dans le tableau 3.2 :

Requête Sparql	**Score Similarité**	**Classement**
Requête 1 (Figure 3.13)	0,51	1
Requête 2 (Figure 3.14)	0,37	2

Tableau 3.2 : Classement des requêtes formelles selon leurs scores

3.4. Conclusion :

Dans ce chapitre on a opté pour une approche de formalisation du problème de la construction de requête formelle qui vise à traduire les mots-clés en requêtes SPARQL pour satisfaire les fins sémantiques utilisateurs.

On a détaillé les trois étapes du système (mapping - construction - classement) avec justifié et accompagné par des exemples concrets.

4

CHAPITRE

L'Implémentation et l'Evaluation

Introduction :

Ce dernier chapitre est réservé à la phase d'implémentation de la solution maintenue dans le chapitre précédent. Nous exploitons un code OWL extrait de l'ontologie DBpedia sur un programme Java équipé de plusieurs APIs nécessaires comme Jena, WordNet, Lucene. Pour finir, nous testons le nouveau système afin d'évaluer les résultats obtenues.

4.1. Implémentation :

4.1.1. Les outils et les langages utilisés :

- **Java** [1]**:** est un langage de programmation développé dans les laboratoires de *Sun Microsystems* et fourni gratuitement aux utilisateurs. Nous avons choisis ce langage pour de nombreuses raisons :

✓ Java est un langage orienté objet simple, qui réduit le risque des erreurs d'incohérences.
✓ Il est indépendant de toute plate forme, il est possible d'exécuter des programmes Java sur tous les environnements qui possèdent une Java Virtual Machine (JVM).
✓ Il est doté d'une riche bibliothèque de classes, comprenant la gestion des interfaces graphiques (fenêtres, menus, graphismes, boites de dialogue, contrôles)
✓ Il permet d'accéder d'une manière simple aux fichiers et aux réseaux (notamment Internet).
✓ Il est caractérisé aussi par la réutilisation de son code ainsi que la simplicité de sa mise en œuvre.
✓ Il est compatible avec L'API Jena, ce qui nous permet la manipulation, le parcours et la modélisation des documents OWL.

- **Jena** [2]**:** est un ensemble d'outils (API) open source développés par *HP Labs semantic Web programme,* permettant de lire et de manipuler des ontologies décrites en RDFS ou en OWL et d'y appliquer certains mécanismes d'inférences. Pour notre projet on utilise la version 2.6.4.

- **Lucene** [3]**:** est une bibliothèque open source écrite en Java qui permet d'indexer et de chercher du texte. Il est utilisé dans certains moteurs de recherche afin de profiter de ces détails techniques sur le traitement du langage naturel (snowball). Le concept est assai analogue à

[1] https://www.oracle.com/fr/java/
[2] https://jena.apache.org/
[3] https://lucene.apache.org/

l'index terminologique que l'on trouve généralement à la fin d'un livre, qui permet de trouver rapidement les pages qui traitent d'un sujet donné.

- **WordNet** [4]**: est** un thésaurus électronique, une référence lexicale largement utilisée pour l'anglais. Sa conception reposait sur des théories psycholinguistiques des mémoires lexicales humaines. Ce lexique comporte des termes organisés dans des synsets qui regroupent des noms, des verbes, des adjectifs et des adverbes organisés dans des ensembles de synonymes appelés « synsets », représentant les concepts lexicaux sous-jacents.

Pour l'implémentation de mesure de similarité de WordNet on a utilisé l'API JAWS (Java API for WordNet Searching) connectable avec WordNet.

Pour la similarité morphologique, on a utilisé le projet SecondString[5] qui un paquetage java ouvert qui contient plusieurs métriques basées sur la syntaxe comme : JaroWinkler, Levenshtein, QGramsDistance, etc.

4.1.2. L'Implémentation :

a- Edition de l'ontologie DBpedia :

La première étape de l'implémentation consiste à intégrer une copie non complète de l'ontologie DBpedia dans le répertoire principale de l'application , et on a basé sur la présence du concept " Work " et spécialement " Written work " . Nous pouvons accéder au contenu de notre base de connaisses à travers le logiciel (Protégé) ou directement par l'éditeur " Wordpad " afin d'ajouter des propriétés et des instances importantes pour enrichir la base. Il est a signaler que notre DBpedia prototype contient 556 classes, 282 propriétés et 8 instances relatives au concept des auteurs et des livres.

b- Exploitation du code OWL dans le programme Java :

Pour falloir exploiter notre code OWL (fichier DBpedia2014.owl) dans notre programme Java, on s'est rendu compte qu'il existe une API appelée Jena qui sert de lien entre le code OWL et le programme Java. Il faut donc s'assurer de l'implémentation des APIs de la bonne manière et dans le bon emplacement. (Voir Figure 4.1)

[4] https://wordnet.princeton.edu/
[5] http://secondstring.sourceforge.net/

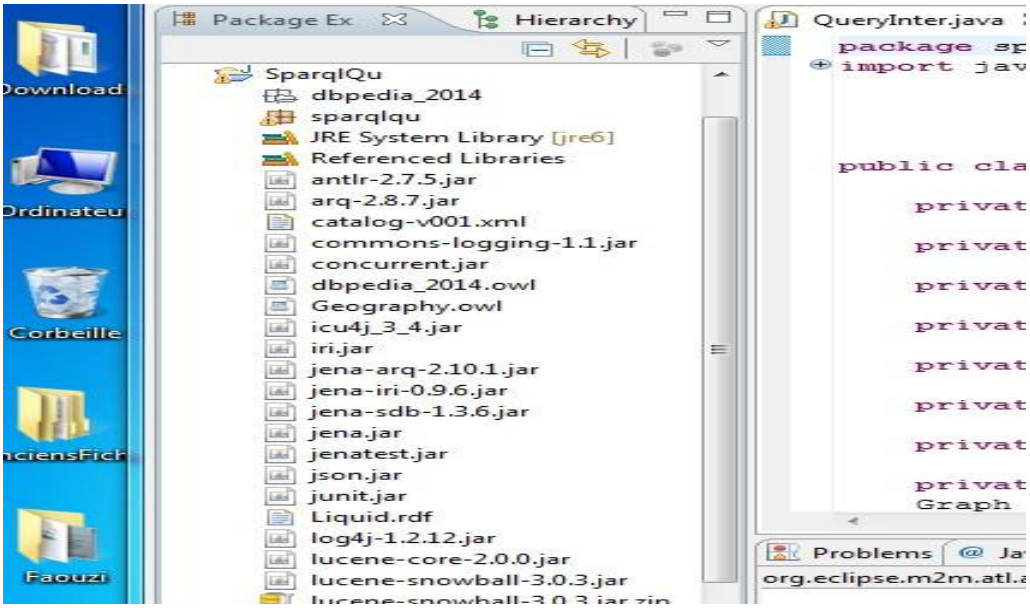


Figure 4.1 : Vue de Librairies utilisés sur JAVA éclipse

c- Exécution de l'application :

Après avoir installer le programme et ces accessoires sur un PC avec un Processeur : Intel Core i5 et Mémoire vive : 4 Go. Après l'exécuté du code , nous avons put capturer les états suivants (voir les figures 4.2 , 4.3 et 4.4) :

- **Interface principale :**

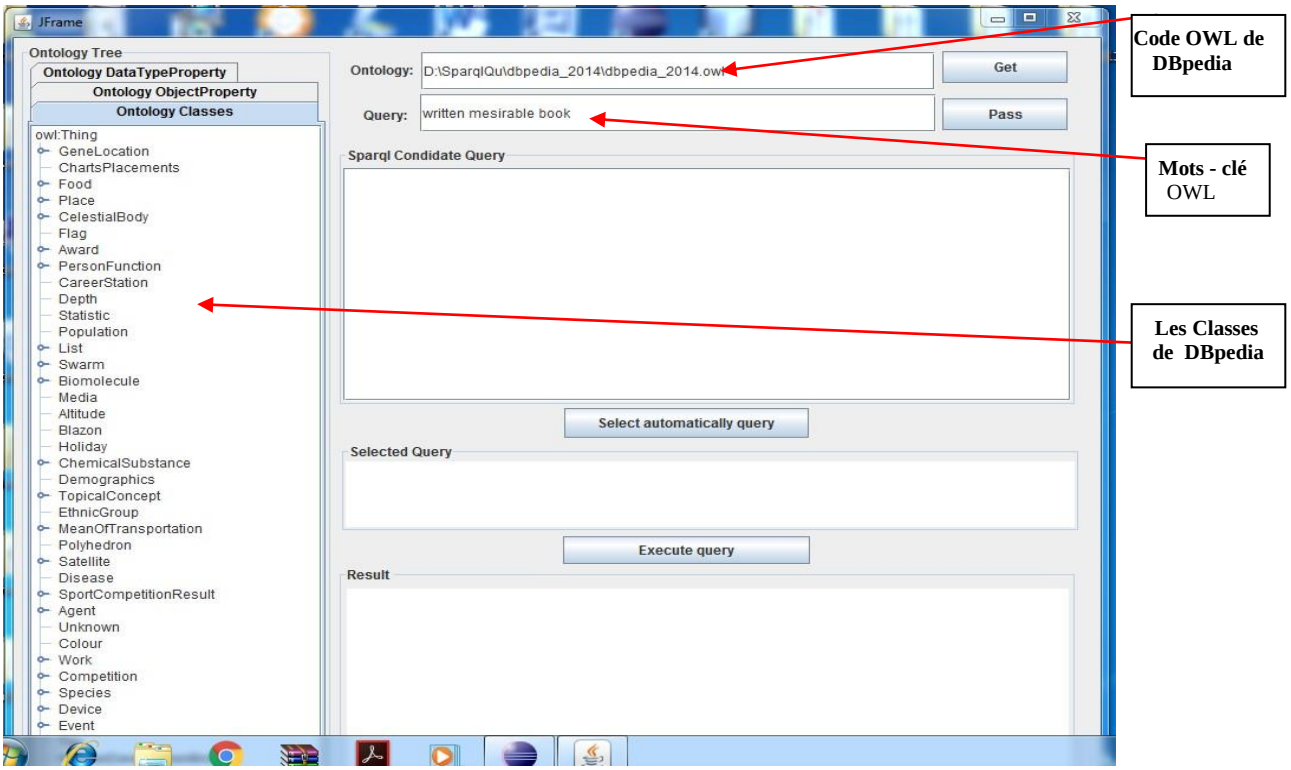


Figure 4.2 : Interface Principal de l'application

- **Interface après la recherche sur le mot –clé " book " :**

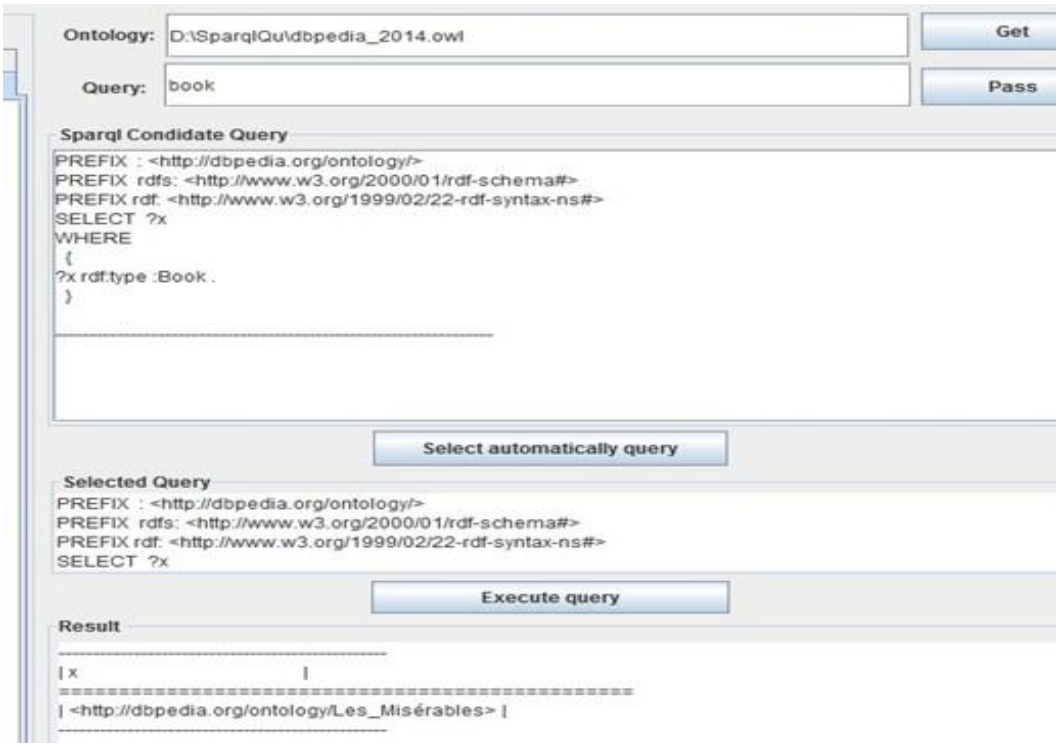


Figure 4.3 : résultat de recherche sur la requête "book"

- **Interface après la recherche sur la requête "** writer misérable book **" :**



Figure 4.4 : résultat de recherche sur la requête " writer misérable book "

4.2. Evaluation :

Pour l'évaluation, nous avons essayé quelques requêtes mots-clés qui sont mentionnées dans le tableau suivant :

N°	*Les Requêtes*
1	*book misérable*
2	*writer book*
3	*writer misérable book*
4	*book*
5	*author misérable book*

Tableau 4.1 : Ensembles des requêtes mots-clés

4.2.1. Métriques utilisées :

Il existe plusieurs facteurs pour mesurer la performance d'un système de recherche d'information tels que: le temps de réponse du système, le nombre de documents pertinent retournés, la méthode de présentation des résultats, ... [BAZ 05].

Mais les deux mesures les plus importantes pour l'évaluation d'un système de recherche est la " Précision " et le " Rappel ". Ces deux mesures sont largement utilisées et permettent de bien connaitre les performances du système de recherche.

a- La précision : permet de déterminer le nombre de Requêtes Pertinentes Retournées par le système de recherche comme résultats par rapport aux nombres Total Requêtes Retournées. La précision parfaite est égale à « 1 », qui veux dire que tous les documents retournés par le système sont pertinents.

$$\textbf{Précision} = \frac{\textbf{Nombre Requêtes Pertinentes Retournées}}{\textbf{Nombre Total Requêtes Retournées}}$$

b- Le rappel : est le Nombre des Requêtes Pertinentes Retournées par le système par rapport aux Nombre Total Requêtes Pertinentes. Le rappel parfait est « 1 », et qui veut dire que tous les documents pertinents sont retournés par le système.

$$\textbf{Rappel} = \frac{\textbf{Nombre Requêtes Pertinentes Retournées}}{\textbf{Nombre Total Requêtes Pertinentes}}$$

Apres l'enregistrement de tous les résultats des recherches faites une par une sur les mots-clés du Tableau 4.1 **,** et on appliquant les formules ci-dessus, nous aurons les résultats suivants (voir tableau 4.2) :

N°	*Requêtes mots-clés*	*Nombre Requête Pertinents Retournés*	*Nombre Total Requêtes Pertinents*	*Nombre Total Requêtes Retournés*	*Rappel*	*Precision*
1	*book misérable*	*1*	*3*	*2*	0.333	0.500
2	*writer book*	*3*	*3*	*4*	1	0.750
3	*writer misérable book*	*2*	*3*	*4*	0.666	0.500
4	*book*	*1*	*3*	*1*	0.333	1
5	*Author misérable book*	*1*	*3*	*2*	0.333	0.500

Tableau 4.2 : Résultats de calcule des Rappels et Précisions

On peut remarquer que notre exemple marque des rappels moyens avec des précisions remarquables.

4.3. Conclusion

Dans ce chapitre, nous avons présenté le détail de l'implémentation de notre application sous un langage Java renforcé avec les API nécessaires pour notre cas. Nous avons préparé la plate forme idéale pour l'exécution du système qui consiste à interroger une copie de la base de connaissances DBpedia par quelques requêtes mots clé choisis. Suite aux résultats obtenus, une évaluation est faite pour juger la pertinence de notre travaille.

Conclusion Générale

Cette étude formalise le problème de la construction des requêtes formelles meilleures en adoptant une approche efficace afin de réduire l'écart entre la recherche sémantique basé sur la logique formelle et les utilisateurs finaux habitué à utiliser les moteur de recherche classique. Elle vise globalement à traduire les requêtes mots-clés en requêtes SPARQL en utilisant les connaissances disponibles dans une base bien définie, c'est l'ontologie major DBpedia .

Notre approche peut se résumé ainsi : Une fois la requête mots-clés est lancée, le mapping des mots clés aux éléments de la base de connaissance débute en se basant sur le calcule de similarité pour trouver les éléments correspondants syntaxiquement et sémantiquement. Ensuite, l'étape de construction des graphes des requêtes qui applique un certain nombre d'algorithmes pour construire des graphes avec différents sens Enfin, l'étape de classement des requête évalue les requêtes formelles construites en utilisant un modèle de classement descendant des scores moyens des éléments des requêtes. Avec les premiers tests, notre système a réalisé des résultats encourageants.

Les perspectives de recherche dans ce domaine sont diverses, notamment :

- ✓ l'élargissement de l'exploitation et la portée des requêtes SPARQL par l'introduction de certains opérateurs structurés (par exemple : NOT, OR, FILTER … .. etc.) afin d'avoir des résultats meilleurs.
- ✓ Etendre le travail pour interroger DBpedia sur internet en utilisant des points d'accès SPARQL.

Abréviation - Glossaire

Abréviation

W3C : World Wide Web Consortium
WS : Web Sémantique
RI : Recherche d'Information
URI : Uniform Resource Identifiers
URL : Uniform Resource Locator
XML : eXtensible Markup Langage
GCs : Graphes Conceptuels
OWL : Ontology Web Langage
RDF : Resource Description Framework
RDF(S) : Resource Description Framework Schema
SPARQL: SPARQL Protocol and RDF Query Language
BC : Base de connaissances
SQR : Systèmes Questions Réponses
API : **Application** Programming Interface

Glossaire

Ontologie : permet la modélisation d'un domaine de connaissances et peut être vue comme un modèle conceptuel d'un domaine particulier, qui décrit les concepts de ce domaine et les relations entre ces concepts.

Ressource: certaines entités. Elle pourrait être une ressource Web comme une page Web, ou une chose physique concrète comme un arbre ou une voiture. Elle pourrait être une idée abstraite comme les échecs ou le football. Les ressources sont nommées par une URI.

Triplet : une structure contenant un sujet, un prédicat et un objet. Un autre terme pour une déclaration.

Sujet : la ressource qui est la source d'un arc dans un modèle RDF.

Objet : la partie d'un triplet qui est la valeur de la déclaration.

Prédicat : la partie de la propriété dans un triplet.

Propriété: une propriété est un attribut d'une ressource. Par exemple " RDF.type"

Littéral : une chaîne de caractères qui peut être la valeur d'une propriété.

Mot : la plus courte et la plus simple unité textuelle significative dans un document texte. Un mot est une séquence de lettres entourées par des séparateurs.

Concept : Concept se concentre plus sur la sémantique du texte. Il est une unité abstraite de connaissances qui explique le sens des mots.

Requête : demande, sollicitation par écrit adressée à quelqu'un ayant autorité pour prendre une décision.

Bibliographie

[AIM,FUR,KUN,TRI 10]:	Aime X., Furst F., Kuntz P., and Trichet F. "Enrichissement sémantique de requêtes au moyen d'ontologies de domain personnalisées". actes de l'atelier "personnalisation du web". In 10ième Journées francophones d'Extraction et de Gestion de Connaissance (EGC 2010), Hammamet., Tunisie, (2010).
[ATH VC KOT 04]:	N Athanasis, VC, Kotzinos, D . " Génération Sur les requêtes Fly pour la sémantique Web","L'ICS-FORTH graphique RQL Interface (GRQL) ". In Actes de ISWC . (2004)
[BAZ 05]:	Mustapha Baziz, "Indexation conceptuelle guidée par ontologie pour la recherche d'information", thèse de doctorat, IRIT, Toulouse. France. (2005)
[BER 01]:	T.Berners-Lee, J. Hendler, O. Lassila, the semantic web, " Scientific American" , 284(5), mai (2001)
[BHA,HUL,NAK 02]:	Bhalotia, G., Hulgeri, A., Nakhe, C., Chakrabarti, S., Sudarshan, S . " Key-mot recherche et la navigation dans les bases de données qui utilisent les banques". In: Actes de ICDE 2002, (2002)
[BOU 11]:	BOURAMOUL Abdelkrim. "Recherche d'Information Contextuelle et Sémantique sur le Web". thèse de doctorat. Université MENTOURI de Constantine. Algérie. (2011)
[CHA,DAS,NAR 02]:	Chaudhuri, S., Das, G., Narasaya, V . "DBExplorer: Un système de recherche par mot sur bases de données relationnelles". In: Actes de données Engineering (2002)
[CHA ,NAD,OLI,WIL 04]:	Chaker Djaiz, Nada Matta, Olivier Corby, William Léone, William Lesguillier , " Flexibilité de la recherche d'information dans les projets de conception", Université de Technologie de Troyes 12 rue Marie Curie, BP. 2060, 10010 Troyes Cedex, France (2004).
[CHR,MAN,PRA 09]:	Christopher D. Manning, Prabhakar Raghavan Hinrich Schütze, " Introduction to information Retrieval", Livre, Cambridge University Press. (2009)
[CHR,KEI 07]:	Christopher Baker and Kei-Hoi Cheung. "Knowledge discovery for biology with Taverna". Producing and consuming semantics in the Web of science. Semantic Web, chapter 16, pages 355–395. Springer-Verlag New York Inc, (2007).
[COM,HAE,HER 10] :	COMPAROT C, HAEMMERLÉ O & HERNANDEZ N "An easy way of expressing conceptual graph queries from keywords and query patterns". In ICCS, (2010a).
[DEK 94]:	DEKLEVA S. "Is natural language querying practical? ", ACM SIGMIS Database, (1994)
[DOM , ALD ,GER 02]:	Domenico M. Pisanelli, Aldo Gangemi, and Geri Steve. "Ontologiesand Information Systems: The marriage of the century".In Proceedings of LYEE Workshop, 2002.
[ELB,RAM,SCH,WEI 10]:	ELBASSUONI S., RAMANATH M., SCHENKEL R. & WEIKUM G. "Searching rdf graphs with sparql and keywords". IEEE Data Eng. Bull., 33(1), 16–24. (2010).
[GOM 10]:	GOMEZ CARPIO , " Enrichissement de requêtes et visualisation

	sémantique dans une coopération de systèmes ", thèse de doctorat, Université de Bourgogne, 14 Decembre (2010) .
[GON,CHA,LEO 05]:	Zhiguo Gong, Chan Wa Cheang, and Leong Hou U. " Web query expansion by WordNet". Lecture Notes in Computer Science 3588, editor, 16th International Conference on Database and Expert Systems Applications, DEXA'05, August (2005),
[GRU, 93] :	Gruber T., A translation approach to portable ontology specifications. Knowledge Acquisition Journal. 5:199 -220. (1993)
[GRU, OLS 94] :	Gruber T. et Olsen G.R., An ontology for enginering mathematics. Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasonning. Bonn, Germany, Morgan Kaufmann, pp. 258-269. (1994)
[HRI PAP 02]:	Hristidis, V., Papakonstantinou, Y .: DÉCOUVRIR, " Mot - clé Rechercher dans Relational Bases de données". Dans : Actes du VLDB (2002)
[JOR,ADM 06]:	Jorge Cardoso and AmitSheth. " The semantic web and its applications " . In Semantic Web Services, Processes and Applications, volume 3 of Semantic Web and Beyond, pages 3{33. SpringerUS, (2006).
[KAU, BER 10]:	KAUFMANN E. & BERNSTEIN A. "Evaluating the usability of natural language query languages and interfaces to semantic web knowledge bases". Web Semantics : Science, Services and Agents on the World Wide Web, (2010).
[LAT,LEH, BEN 09]:	LATRECHE Abdelkrim1, LEHIRECHE Ahmed1, BENYAHIA Kadda1, " Interrogation à base d'annotation sémantique", 1 Laboratoire EEDIS, UDL de Sidi Bel Abbes, ALGERIE, (2009) .
[LET, JEN, ISE 15]	Lehmann, Jens, Isele, Robert, Jakob, Max, Jentzsch, Anja, Kontokostas, Dimitris, Mendes, Pablo N., Hellmann, Sebastian, Morsey, Mohamed, van Kleef, Patrick, Auer, Sören and Bizer, Christian. "DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia." *Semantic Web Journal* 6 , no. 2 (2015)
[MIL,BEC,FEL,GRO 90]:	G. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. Miller. "Introduction to wordnet", An on-line lexical database. International Journal of Lexicography, (1990).
[NAV,VEL 03]:	R. Navigli and P. Velardi. " An analysis of ontology-based query expansion stratégies". In 14th European conference on machine learning (2003).
[SEB, CLA 04]:	P. Sebillot V. Claveau. "Extension de requêtes par lien sémantique nom-verbe acquis sur corpus". In Traitement automatique des langues naturelles, TALN'04, (2004).
[SEM 06]:	LEI Y., UREN V. S. & MOTTA E Semsearch , " A search engine for the semantic web". In EKAW. (2006).
[WAN, XIO, YU 07]:	Zhou, Q., Wang, C., Xiong, M., Wang, H., Yu, Y., " SPARK :Adapting keyword query to semantic search". In: ISWC/ASWC. (2007)
[TUA 06]:	Tuan Dung CAO, " Exploitation du web sémantique pour la veille technologique", Thèse de doctorat. Université Nice-Sophia Antipolis - UFR, Novembre (2006).

[VAN, SCH, WIE 97]:	G. Van Heijst, A. Th. Schreiber, and B. J. Wielinga. " Using explicit ontologies in KBS development". International Journale Human-Computer studies, 47(2-3) , (1997).
[VOO 94]:	E. M. Voorhees. " Query expansion using lexical - semantic relations ". In SIGIR : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, Springer-Verlag New York, (1994).
[WAN, MIL 10]:	Shuguang Wang and Milos Hauskrecht. " Eective query expansion with the existence distance based term similarity metric.In SIGIR 10" : Proceeding of the 33rd international ACM SIGIR conférence on Research and développement in information retrieval, ACM. (2010).

Sitographie

[01] https://fr.wikipedia.org/wiki/Web_s%C3%A9mantique
[02] https://www.abracadabrapdf.net/le-pdf-parfait/metadonne
[03] lod-cloud.net
[04] http://zone47.com/dozo/voyage-de-dbpedia-en-wikidata-a-bord-dun-bot
[05] http://zone47.com/dozo/voyage-de-dbpedia-en-wikidata-a-bord-dun-bot
[06] https://fr.wikipedia.org/wiki/DBpedia
[07] http://nt2.uqam.ca/fr/actualites/la-limite-des-mots-clefs
[08] https://www.evi.com
[09] https://www.wolframalpha.com/
[10] http://www.d.umn.edu/~tpederse/similarity.html
[11] http://dictionnaire.sensagent.leparisien.fr/Lucene/fr-fr/