REPUBLIQUE DU CAMEROUN
Paix-Travail-Patrie
****
UNIVERSITE DE YAOUNDE 1
****
FACULTE DES SCIENCES
****
UNITE DE RECHERCHE ET DE FORMATION DOCTORALE MATHEMATIQUES, INFORMATIQUE, BIO-INFORMATIQUE ET APPLICATIONS

REPUBLIC OF CAMEROON
Peace-Work-Fatherland
****
UNIVERSITY OF YAOUNDE 1
****
FACULTY OF SCIENCES
****
RESEARCH AND POSTGRADUATE TRAINING UNIT FOR MATHEMATICS, COMUPUTER SCIENCES, BIOINFORMATICS AND APPLICATIONS

Mémoire de fin d'étude

Construction de ressources lexicales sémantiques basée sur le Web sémantique : cas de langues Sara-Baguirmiennes du Tchad.

Présenté et soutenu par :

NDALI NABIA Guy Serge

Matricule : 18Z2231

Etudiant en Master 2 Informatique

Option : Système d'Information et Génie Logiciel

En vue de l'obtention du :

« DIPLOME DE MASTER RECHERCHE EN INFORMATIQUE »

Membres du jury :

Président : Pr. ATSA ETOUNDI ROGER

Encadreur : Pr. ABESSOLO ALO'O GHISLAIN

Examinateur : Dr AMINOU HALIDOU, Charge de Cours, Université de Yaoundé 1

Année académique 2023-2024

Dédicace

En mémoire de mon grand-père Pasteur **NABIA Paul**.

Remerciements

Je viens par ces quelques lignes exprimer ma gratitude envers tous ceux qui, par leur présence, leur soutien, leur disponibilité et leurs conseils, ont permis d'accomplir ce projet.

Mes tout premiers remerciements s'adressent à **mon directeur de mémoire, Pr. ABESSOLO ALO'O Ghislain**, Maitre de Conférence et Chargé de Cours au Département d'Informatique de la Faculté des Sciences de l'Université de Yaoundé I. Sans ses orientations éclairées, sa rigueur scientifique et son implication constante, ce travail n'aurait certainement pas pu aboutir.

Je remercie également les membres du jury pour l'honneur qu'ils m'ont fait en acceptant d'évaluer ce travail. Mes sincères remerciements vont au **Professeur ATSA ETOUNDI Roger**, Président du jury, pour sa bienveillance et sa disponibilité, ainsi qu'au **Docteur AMINOU HALIDOU**, Chargé de Cours à l'Université de Yaoundé I, pour ses remarques pertinentes et enrichissantes.

Mes pensées reconnaissantes vont également à **ma famille**, pour son soutien indéfectible tout au long de mes études. Votre amour, votre patience et vos encouragements ont été une source précieuse d'énergie et de motivation. Je remercie tout particulièrement Alain Kimato, Leïla Nabia, Nabia Emeline, TOUMAN Clément, ainsi que tous les frères, sœurs, oncles et tantes de la grande famille. Tout ceci n'a été possible que grâce à vous.

Je tiens à rendre un hommage appuyé à **l'ensemble des enseignants du Département d'Informatique** pour leur dévouement, leur professionnalisme et les efforts constants qu'ils consentent au service de la formation des étudiants.

Enfin, mes remerciements s'adressent à mes camarades et amis, notamment Yves, Onguéné, Stéphane, ADALA Roger, BOBDA Armel, ainsi qu'à ceux devenus des frères : Francis, Danilo, Patrice, KEMNELDJI, et tous ceux que je n'ai pas pu nommer ici. Merci pour votre amitié, votre soutien au quotidien, et votre aide précieuse dans la relecture de ce document.

Table des matières

Liste figures

Abréviations

1. DOE : Differential Ontology Editor
2. GUI : Identifiant Universel Globalement Unique
3. ODE : Ontology Design Environment
4. OTK : On-To-Knowledge
5. PWN : Princeton WordNet
6. RDF : Resource Description Framework
7. TALN : Traduction Automatique des Langues Naturelles
8. UML : Unified Modeling Language

Résumé

Dans un contexte de la numérisation et de la préservation des langues, la construction de réseaux lexicaux sémantiques est une tâche cruciale pour le traitement automatique, la recherche et la préservation de ces langues. Pour construire ces réseaux lexicaux sémantiques, deux types d'approches sont utilisés à savoir la méthode par fusion et la méthode par extension. Ces approches bien qu'efficace pour les langues documentées, éprouvent des difficultés à être appliqués sur les langues peu dotées du fait de leur faible niveau de documentation. Pour répondre à cela, nous proposons une méthodologie pour construire un réseau lexical sémantique multilingue, cette méthode comprend la collecte et le traitement de corpus textuels, l'identification des concepts et des relations sémantiques, l'extraction et la représentation des concepts et des relations, la génération d'un graphe RDF des dictionnaires. Le résultat attendu est un réseau lexical sémantique multilingue pour les langues peu dotées, qui facilitera la recherche et la préservation de ces langues, et contribuera à la valorisation de la diversité linguistique et culturelle. Pour faciliter ce processus, Nous proposons également une plateforme web qui permet à un utilisateur de générer des fichiers RDF et enfin d'effectuer des recherches textuelles multilingues dans ces différentes ressources sous leur nouvelle structure de représentation.

Mots-clés : réseau lexical sémantique, description RDF, ontologie, Informatique linguistique, web sémantique, application web.

Abstract

In the context of digitization and language preservation, constructing semantic lexical networks is a crucial task for natural language processing, research, and the safeguarding of languages. To build these semantic lexical networks, two main approaches are employed: fusion-based and extension-based methods. While effective for well-documented languages, these approaches face challenges when applied to under-resourced languages due to their limited documentation. To address this, we propose a methodology for constructing a multilingual semantic lexical network. This method involves collecting and processing textual corpora, identifying semantic concepts and relations, extracting and representing these concepts and relations, and generating an RDF graph from dictionaries. The expected outcome is a multilingual semantic lexical network for under-resourced languages, facilitating research and preservation efforts while contributing to the appreciation of linguistic and cultural diversity. Additionally, we propose a web platform that allows users to generate RDF files and perform multilingual text searches within these resources using their new representation structure.

Keywords: semantic lexical network, RDF description, ontology, computational linguistics, semantic web, web application.

Introduction générale

I. Contexte

Depuis presque une vingtaine d'années, le nombre de travaux sur la construction de ressources lexicales sémantiques souligne l'intérêt, l'actualité et les enjeux de ce domaine pour les sciences cognitives et la traduction automatique[1].L'utilisation de ces ressources s'est considérablement accrue dans le domaine du traitement automatique des langues, ce qui conduit d'autres travaux à l'échelle intercontinentale pour la construction des ontologie multilingues de ces ressources. Parmi les exemples de réseaux lexicaux sémantiques multilingues, on peut citer le projet EuroWordNet [2] pour l'Europe et Asian WordNet [3] pour l'Asie. D'autre travaux émergent également pour les langues peu dotées où les défis restent énormes [4]. Une langue est dite peu dotée lorsque celle-ci dispose de peu de ressources linguistiques informatisées telle qu'un lexique, un correcteur orthographique, un analyseur syntaxique, et d'une Terminologie spécialisée. En Afrique, il existe plus de 2000 langues, dont la plupart disposent peu de ressources et sont orales[5]. Malgré le nombre, très peu de langues ont un statut officiel, quelques-uns ont disparues et d'autre sont en voie de disparition [6].Conscient de ces dangers, plusieurs efforts ont déjà fait dans ce sens notamment la numérisation et la création des plateformes d'apprentissages de ces langues, afin de sauvegarder cette richesse culturelle inhérente à la transmission du savoir local.

L'utilité des réseaux lexicaux sémantiques a été démontrée pour des tâches telles que la traduction automatique [7] ou la désambiguïsation lexicale[8], entre autres. Les travaux de recherches sur ces sujets se sont repandus sur tous les continents, a l'exception du continent Africain qui comptes un WordNet pour le swahili, et un WordNet multilingues pour les langues locales en Afrique du sud [9].

II. Problématique

La construction de réseaux lexicaux sémantiques pour les langues peu dotées demeure un défi majeur. Ce problème s'explique par la faible disponibilité de ressources linguistiques numériques (dictionnaires, corpus, annotations), l'ambiguïté des sens, ainsi que l'hétérogénéité des données existantes. En conséquence, ces langues restent peu représentées dans les technologies du langage, ce qui freine leur traitement automatique, leur accessibilité, et compromet leur préservation face au risque de disparition.

Par ailleurs, la construction et l'enrichissement de ressources lexicales sémantiques est une tâche complexe, chronophage et coûteuse, nécessitant des compétences pluridisciplinaires en linguistique, ontologie et sciences cognitives [10]. Les approches classiques — telles que la méthode par fusion ou par extension se révèlent efficaces pour les langues bien dotées, mais rencontrent de sérieuses limites lorsqu'elles sont appliquées à des langues faiblement documentées. Ces difficultés sont accentuées par la polyvalence des sens et la diversité des formats ou structures linguistiques disponibles [11], [12].

Face à ces constats, nous posons la question suivante : comment construire un réseau lexical sémantique multilingue pour des langues peu dotées en s'appuyant sur les technologies du Web de données, et comment l'enrichir efficacement malgré le manque de ressources directes ?

III. Objectifs

L'objectif principal de ce mémoire est de construire un réseau lexical sémantique multilingue pour les langues peu dotées. Plus spécifiquement, de proposer un générateur de graphe RDF, permettant de Représenter ces ressources lexicales du groupe de langue Sara-Baguirmien de manière à réduire les ambiguïtés et une facilité de recherche dans ces lexiques à travers des interfaces de manipulation de ces ressources.

IV. Méthodologie

La méthodologie adoptée dans ce travail repose sur une combinaison des approches par fusion et par extension, adaptées au contexte des langues peu dotées. Elle s'articule en plusieurs phases complémentaires.

Dans un premier temps, il s'agit de constituer un lexique bilingue ou multilingue, en exploitant les ressources disponibles ou en les créant si nécessaire. Ensuite, une étape d'alignement sémantique est réalisée pour rapprocher les entrées lexicales issues de différentes langues, en identifiant des correspondances conceptuelles. Une mesure de confiance est alors appliquée pour sélectionner les correspondances les plus fiables et écarter les associations incertaines.

Sur cette base, la démarche se décline selon les étapes suivantes :

1. Collecte et traitement de corpus textuels en langues Sara-Baguirmien afin d'extraire les unités lexicales pertinentes ;
2. Identification des concepts et des relations sémantiques à partir de ces corpus ;

3. Extraction et formalisation des concepts et relations dans une structure exploitable ;
4. Génération d'un graphe RDF à partir des lexiques structurés, conformément aux normes du Web sémantique ;
5. Évaluation du réseau lexical sémantique en termes de cohérence, couverture et interopérabilité ;
6. Diffusion du réseau lexical via une plateforme accessible permettant la consultation et l'exploitation multilingue des données.

Cette méthodologie vise à produire une ressource interconnectée, évolutive et réutilisable pour la recherche, l'enseignement et les technologies du langage appliquées aux langues peu dotées.

V. Résultat attendu

A l'issue de ces travaux, le résultat principal envisagé est un réseau lexical sémantique multilingue pour les langues peu dotées. Hormis ce résultat principal, les résultats secondaires escomptés sont :

- Un outil pour la construction de réseaux lexicaux ;
- Une plateforme permettant d'effectuer des recherches textuelles multilingues ;
- Un format de données ouvert, accessible et pouvant servir de base a des travaux de traitement automatique.

VI. Organisations du document

Outre ce premier chapitre introductif qui a permis de présenter le contexte, la problématique et la méthodologie, ce mémoire est structuré comme suit :

- Chapitre 1 : Revue de la littérature sur les ontologies et les ressources lexiques sémantiques. Ce chapitre a pour objectif de faire une étude des différentes méthodologies existantes et les travaux sur les différents concepts de base tel que les ontologies, le réseau lexical sémantique et les langues peu dotées.
- Chapitre 2 : Conception et Implémentation des réseaux lexicaux sémantiques. Ce chapitre présentes les différentes phases de la méthodologie de construction du réseau lexical, la conception et l'implémentation de cette méthodologie.
- Conclusion générale et Perspectives.

Chapitre 1 : Revue de la littérature sur les ontologies et les ressources lexicales sémantiques.

1.1. Introduction

L'évolution de l'informatique et des technologies de l'information a permis la création de vastes quantités de données et de connaissances dans divers domaines. Cette croissance exponentielle de l'information a créé des défis pour l'organisation et la gestion de ces données. Ensuite, les avancées de l'intelligence artificielle et du traitement du langage naturel ont permis de développer des techniques pour extraire et analyser l'information de manière plus sophistiquée. Cependant, il est devenu de plus en plus difficile de trouver des façons de connecter ces informations de manière significative, afin de faciliter l'automatisation et l'analyse des données. C'est dans ce contexte que la notion d'ontologie a commencé à prendre de l'importance en informatique. L'ontologie est une notion qui remonte à l'Antiquité grecque. Elle a évolué et a été adaptée à d'autres domaines, tels que l'informatique, la traduction automatique des langues naturelle (TALN) et bien d'autre. Les ontologies ont été utilisées depuis longtemps pour améliorer les systèmes de traduction automatique en modélisant les concepts et les relations entre eux dans un domaine spécifique. Afin de mieux situer nos différentes approches, Il est important de rappeler quelques concepts fondamentaux associés aux ontologies et la construction de ressources lexicales.

1.2. Les ontologies

1.2.1. Définition

Le terme ontologie couvre deux usages dont le premier appartient à la philosophie classique et le second plus récent, aux autres sciences cognitives.

En philosophie, **Aristote** le définit comme la science de l'Être en tant qu'être. En d'autres termes, l'Ontologie est la Science ou théorie de l'être. Elle est une branche fondamentale de la Métaphysique, qui s'intéresse à la notion d'existence, aux catégories fondamentales de l'existant et étudie les propriétés les plus générales de l'être.

En **Informatique**, la définition la plus citée dans la littérature est celle proposée par [13]: « une ontologie est une spécification explicite d'une conceptualisation ». Elle a été étendue par [14] : « une ontologie est une spécification formelle d'une conceptualisation partagée » puis par Studer : « Une ontologie est une spécification formelle, explicite d'une conceptualisation

partagée ». Une discussion complète de la signification des termes de cette définition est proposée en introduction du livre de S. Staab [15].

Formellement [16] définit une ontologie par un quadruplet :

O = {C, P, Sub, Applic}, avec :

- C : ensemble des classes utilisées pour décrire les concepts d'un domaine donné (comme les services de voyages, les pannes des équipements, les composants électroniques, etc.). Chaque classe est associée à un identifiant universel globalement unique (GUI) ;
- P : ensemble des propriétés susceptibles d'être utilisées pour décrire les instances de l'ensemble des classes C. Une propriété peut aussi bien avoir comme co-domaine un type (on la qualifie alors aussi d'attribut) qu'une classe, ou un ensemble de classes (on la qualifie aussi de relation). Chaque propriété est associée à un GUI ;
- *Sub* : C → 2C est la relation de subsomption qui, à chaque classe *ci* de l'ontologie, associe ses classes subsumées directes, c'est-à-dire les classes vérifiant la propriété ∀*c*1, *c*2 ∈ C, *c*1subsume *c*2 si et seulement si ∀*x* ∈ *c*2, *x* ∈ *c*1. *Sub* définit un ordre partiel sur C et 2C désigne l'ensemble de partie de C ;
- *Applic* : C → 2P associe à chaque classe de l'ontologie les propriétés qui sont applicables pour chaque instance de cette classe.

Les ontologies visent à capturer la connaissance du domaine et fournissent un consensus sur la compréhension commune du domaine qui peut être réutilisée et partagée à travers des applications ou des groupes. Elles ont un rôle fondamental dans la création du Web sémantique, un outillage qui permet aux ordinateurs d'accéder au sens des données disponibles sur le Web pour assister les utilisateurs de façon plus "intelligente".

1.2.2. Les types d'ontologie

Les ontologies peuvent être classifiées selon plusieurs niveaux, dont les plus courantes sont celle de Gómez-Pérez [17]. Nous en examinerons quatre : Objet de conceptualisation ; Niveau de détail ; Niveau de complétude ; Niveau de formalisme de représentation.

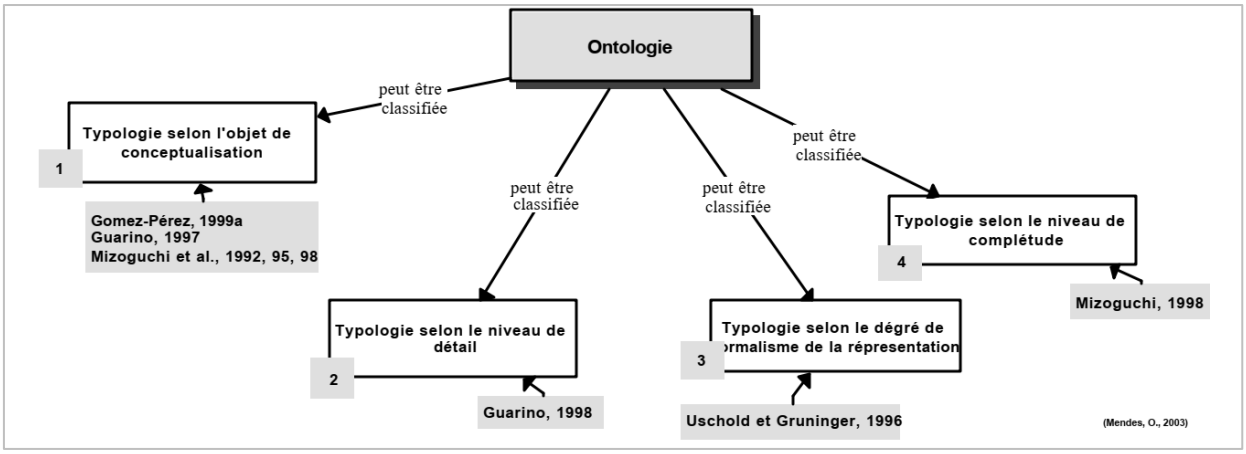


Figure 1 : Typologies d'ontologies selon quatre dimensions de classification [18]

1.2.2.1. Typologie selon l'objet de conceptualisation

La classification selon l'objet de conceptualisation se fait de la façon suivante :

- Représentation des connaissances ;
- Supérieure/ Haut niveau ;
- Générique ;
- Domaine ;
- Tâche ;
- Et Application.

1.2.2.1.1. Ontologie de représentation des connaissances

Ce type d'ontologies regroupe les concepts (primitives de représentation) impliqués dans la formalisation des connaissances. Un exemple est l'*ontologie de Frame* qui intègre les primitives de représentation des langages à base de *frames* : classes, instances, facettes, propriétés/*slots*, relations, restrictions, valeurs permises, etc.[17]

1.2.2.1.2. Ontologie supérieure ou de Haut niveau

Cette ontologie est une ontologie générale. Son sujet est l'étude des catégories des choses qui existent dans le monde, soit les concepts de haute abstraction tels que : les entités, les événements, les états, les processus, les actions, le temps, l'espace, les relations, les propriétés. L'ontologie de haut de niveau est fondée sur : la théorie de l'identité, la méréologie (*theory of whole and parts role*) et la théorie de la dépendance. Guarino et Sowa ont poursuivi chacun indépendamment des recherches sur la théorie de l'ontologie. Tous deux intègrent les fondements philosophiques comme étant des principes à suivre pour concevoir l'ontologie de

haut niveau ou supérieure. Sowa introduit deux concepts importants, *Continuant* et *Occurrent*, et obtient douze catégories supérieures en combinant sept propriétés primitives. L'ontologie supérieure de Guarino consiste en deux mondes : une ontologie des *Particuliers* (choses qui existent dans le monde) et une ontologie des *Universels* comprenant les concepts nécessaires à décrire les Particuliers. La conformité aux principes de l'ontologie supérieure a son importance, lorsque le but est de standardiser la conception des ontologies.

1.2.2.1.3. Ontologie Générique

Cette ontologie aussi appelée, **méta-ontologies** ou ***core ontologies***, véhicule des connaissances génériques moins abstraites que celles véhiculées par l'ontologie de haut niveau, mais assez générales néanmoins pour être réutilisées à travers différents domaines. Elle peut adresser des connaissances factuelles (*Generic domain ontoloy*) ou encore des connaissances visant à résoudre des problèmes génériques (connaissances procédurales) appartenant à ou réutilisables à travers différents domaines (*Generic task ontoloy*). Deux exemples de ce type d'ontologies sont : l'ontologie méréologique [14] contenant des relations, *Partie-de* et l'ontologie topologique contenant des relations, *Associé-à*.

1.2.2.1.4. Ontologie du Domaine

Cette ontologie régit un ensemble de vocabulaires et de concepts qui décrit un domaine d'application ou monde cible. Elle permet de créer des modèles d'objets du monde cible. L'ontologie du domaine est une méta-description d'une représentation des connaissances, c'est-à-dire une sorte de méta-modèle de connaissance dont les concepts et propriétés sont de type déclaratif. La plupart des ontologies existantes sont des ontologies du domaine. Selon Mizoguchi, l'ontologie du domaine caractérise la connaissance du domaine où la tâche est réalisée.

1.2.2.1.5. Ontologie de Tâches

Ce type d'ontologies est utilisé pour conceptualiser des tâches spécifiques dans les systèmes, telles que les tâches de diagnostic, de planification, de conception, de configuration, de tutorat, soit tout ce qui concerne la résolution de problèmes. Elle régit un ensemble de vocabulaires et de concepts qui décrit une structure de résolution des problèmes inhérente aux tâches et indépendante du domaine. Selon Mizoguchi [19], l'ontologie de tâche caractérise l'architecture computationnelle d'un système à base de connaissances qui réalise une tâche.

1.2.2.1.6. Ontologie d'Application.

Cette ontologie est la plus spécifique. Les concepts dans l'ontologie d'application correspondent souvent aux rôles joués par les entités du domaine tout en exécutant une certaine activité.

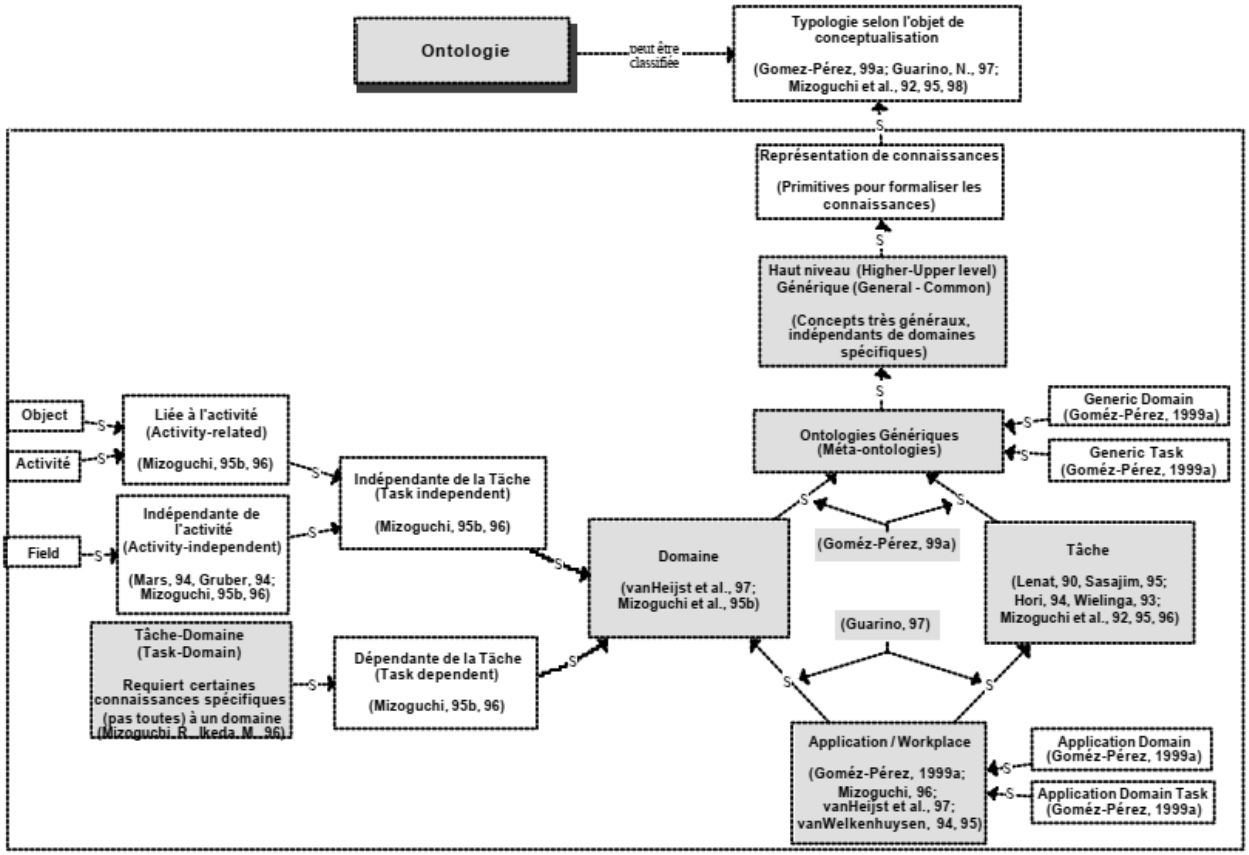


Figure 2 : Typologies selon le niveau de conceptualisation[18]

1.2.2.2. Typologie selon le Niveau de détail de l'ontologie

Par rapport au **niveau de détail** utilisé lors de la conceptualisation de l'ontologie en fonction de l'objectif opérationnel envisagé pour l'ontologie, deux catégories au moins peuvent être identifiées :

- **Granularité fine** : correspondant à des ontologies très détaillées, possédant ainsi un vocabulaire plus riche capable d'assurer une description détaillée des concepts pertinents d'un domaine ou d'une tâche. Ce niveau de granularité peut s'avérer utile lorsqu'il s'agit d'établir un consensus entre les agents qui l'utiliseront ;

- **Granularité large** : correspondant à un vocabulaire moins détaillé comme par exemple dans les scénarios d'utilisation spécifiques où les utilisateurs sont déjà préalablement d'accord à propos d'une conceptualisation sous. Les ontologies de haut niveau possèdent une granularité large, compte tenu que les concepts qu'elles traduisent sont normalement raffinés subséquemment dans d'autres ontologies de domaine ou d'application.

1.2.2.3. Typologie selon le niveau de complétude

Le niveau de complétude a été abordé par Mizoguchi R. et Bachimont. À titre d'exemple, nous décrivons la typologie de [20]. Ce dernier propose la classification sur trois niveaux suivants :

- **Niveau 1 - Sémantique** :

Tous les concepts (caractérisés par un terme/libellé) doivent respecter les quatre principes différentiels :

- Communauté avec l'ancêtre ;
- Différence (spécification) par rapport à l'ancêtre ;
- Communauté avec les concepts frères (situés au même niveau) ;
- Différence par rapport aux concepts frères (sinon il n'aurait pas lieu de le définir).

Ces principes correspondent à l'**engagement sémantique** qui assure que chaque concept aura un sens univoque et non contextuel associé. Deux concepts sémantiques sont identiques si l'interprétation du terme/libellé à travers les quatre principes différentiels aboutit à un sens équivalent.

- **Niveau 2 - Référentiel** :

Outre les caractéristiques énoncées au niveau précédent, les concepts référentiels (ou formels) se caractérisent par un terme/libellé dont la sémantique est définie par une extension d'objets. L'**engagement ontologique** spécifie les objets du domaine qui peuvent être associés au concept, conformément à sa signification formelle. Deux concepts formels seront identiques s'ils possèdent la même extension.

- **Niveau 3 - Opérationnel** :

Outre les caractéristiques énoncées au niveau précédent, les concepts du niveau opérationnel ou computationnel sont caractérisés par les opérations qu'il est possible de leur

appliquer pour générer des inférences (**engagement computationnel**). Deux concepts opérationnels sont identiques s'ils possèdent le même potentiel d'inférence.

1.2.2.4. Typologie selon le niveau de formalisme

On peut distinguer les ontologies selon le formalisme utilisé pour les exprimer

- **Informelle :** l'ontologie est exprimée en langage naturelle. Cela peut permettre de rendre plus compréhensible l'ontologie pour l'utilisateur, mais cela peut rendre plus difficile la vérification de l'absence de redondances ou de contradiction ;
- **Semi-informelle :** l'ontologie est exprimée dans une forme restreinte et structurée de la langue naturelle ; cela permet d'augmenter la clarté de l'ontologie tout en réduisant l'ambiguïté ;
- **Semi-formelle :** l'ontologie est exprimée dans un langage artificiel définit formellement ;
- **Formelle :** l'ontologie est exprimée dans un langage artificiel disposant d'une sémantique formelle, permettant de prouver des propriétés de cette ontologie. L'intérêt d'une ontologie formelle est la possibilité d'effectuer des vérifications sur l'ontologie : complétude, non redondance, consistance, cohérence, etc.

1.2.3. Les composantes d'une ontologie

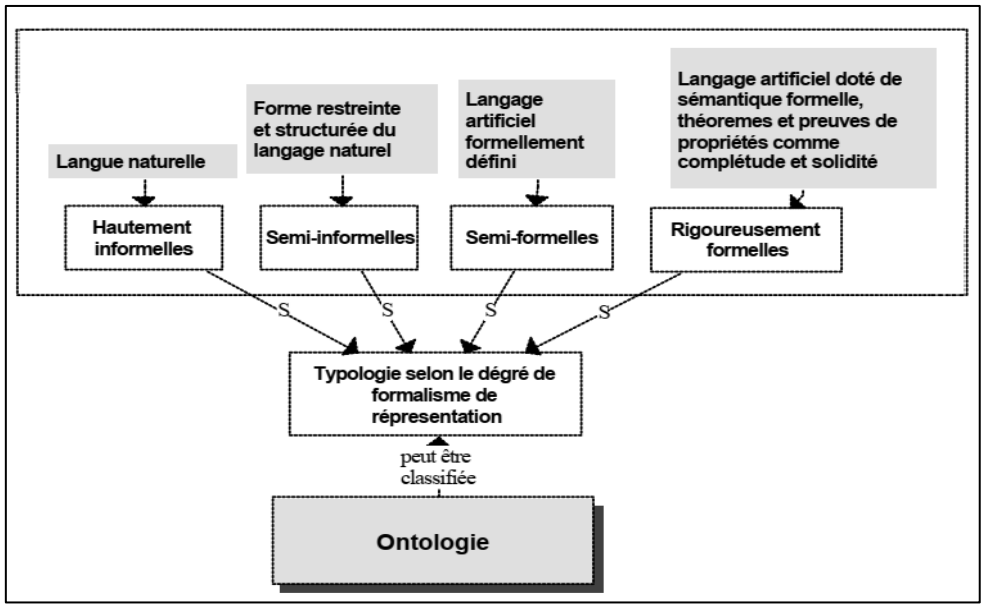


Figure 3 : typologie selon le degré de formalisme)

Les ontologies fournissent un vocabulaire commun d'un domaine et définissent la signification des termes et des relations entre elles. La connaissance dans les ontologies est principalement formalisée en utilisant les cinq types de composants[15] à savoir : concepts (ou classes), relations (ou propriétés), fonctions, axiomes (ou règles) et instances (ou individus).

1.2.3.1. Les concepts :

Aussi appelés termes ou classes, les concepts constituent les objets de base manipulés par les ontologies. Ils correspondent aux abstractions pertinentes du domaine du problème, retenues en fonction des objectifs qu'on se donne et de l'application envisagée pour l'ontologie, par exemple, la description d'un ensemble d'objets, d'une tâche, d'une fonction, d'une stratégie, d'un processus de raisonnement, etc.

Un concept est caractérisé par un ensemble de propriétés :

- Un concept est générique s'il n'admet pas d'extension. La vérité, par exemple, n'a pas d'extension.
- Un concept porte une propriété d'identité si cette propriété permet de différencier deux instances de ce concept.
- Un concept est rigide s'il ne peut pas être une instance d'autres concepts.

Par exemple, l'être vivant est un concept rigide, mais un " être humain " n'est pas un concept rigide, car l'humain est une instance du concept " être vivant ".

- Un concept est anti-rigide s'il peut être une instance pour d'autres concepts.

1.2.3.2. Les relations :

Une relation permet de lier des instances de concepts ou des concepts génériques.

Elles sont caractérisées par un terme ou plusieurs, et une signature qui précise le nombre d'instances de concepts que la relation lie, leurs types et l'ordre des concepts, c'est – à – dire la façon dont la relation doit être lue.

1.2.3.3. Les fonctions :

Les fonctions sont des cas particuliers de relations dans lesquelles un élément de la relation est défini à partir des autres éléments.

1.2.3.4. Les axiomes :

Permettent de modéliser des assertions toujours vraies, à propos des abstractions du domaine

traduites par l'ontologie. Ils permettent de combiner des concepts, des relations et des fonctions pour définir des règles d'inférences et qui peuvent intervenir, par exemple, dans la déduction, la définition des concepts et des relations, ou alors pour restreindre les valeurs des propriétés ou les arguments d'une relation.

1.2.3.5. Les instances :

Ou individus constituent la définition extensionnelle de l'ontologie. Ils représentent des éléments singuliers véhiculant les connaissances (statiques, factuelles) à propos du domaine du problème. Trois grands types de caractéristiques nous permettent de préciser ce qui peut être représenté dans une ontologie considérée en tant qu'objet informatique :

- Les concepts ;
- Les propriétés ;
- Les relations ;

1.2.4. Cycle de vie

Le développement d'ontologies, tout comme le génie logiciel, nécessite une approche structurée et itérative pour garantir que le produit final répond aux besoins des utilisateurs et est techniquement solide. Voici un résumé des étapes clés du cycle de vie d'une ontologie, inspiré des principes du génie logiciel [21] :

- **Évaluation des Besoins** : Comprendre les objectifs opérationnels et les exigences des utilisateurs pour définir le périmètre de l'ontologie.
- **Construction** :
  - **Conceptualisation** : Création d'un modèle conceptuel à partir de ressources textuelles et/ou multimédia, qui capture les connaissances du domaine.
  - **Ontologisation** : Structuration et formalisation de la conceptualisation pour créer une ontologie avec une terminologie et une sémantique formelle.
  - **Opérationnalisation** : Transformation de l'ontologie en une représentation formelle adaptée à une application spécifique.
- **Diffusion** : Partage de l'ontologie avec les utilisateurs et les parties prenantes.
- **Utilisation** : Mise en œuvre de l'ontologie dans des systèmes ou des applications.

- **Réévaluation et Extension** : Après chaque utilisation significative, l'ontologie est réévaluée et peut être étendue ou partiellement reconstruite pour mieux répondre aux besoins évolutifs.
- **Intégration** : Importation d'autres ontologies dans l'ontologie en construction pour enrichir ou compléter les connaissances.
- **Documentation et Évaluation** :
  - **Documentation** : Fournir des informations détaillées sur l'ontologie pour faciliter son utilisation et sa maintenance.
  - **Évaluation** : Comprend deux niveaux :
    - ✓ **Vérification** : Assurer que l'ontologie est construite correctement selon un modèle formel.
    - ✓ **Validation** : Confirmer que l'ontologie correspond bien au domaine de connaissance qu'elle est censée modéliser.

Cette approche cyclique et évolutive permet de s'assurer que l'ontologie reste pertinente et utile, tout en minimisant les erreurs et en maximisant la qualité et la fiabilité. La documentation et l'évaluation sont des activités continues qui soutiennent chaque étape du processus de construction de l'ontologie.

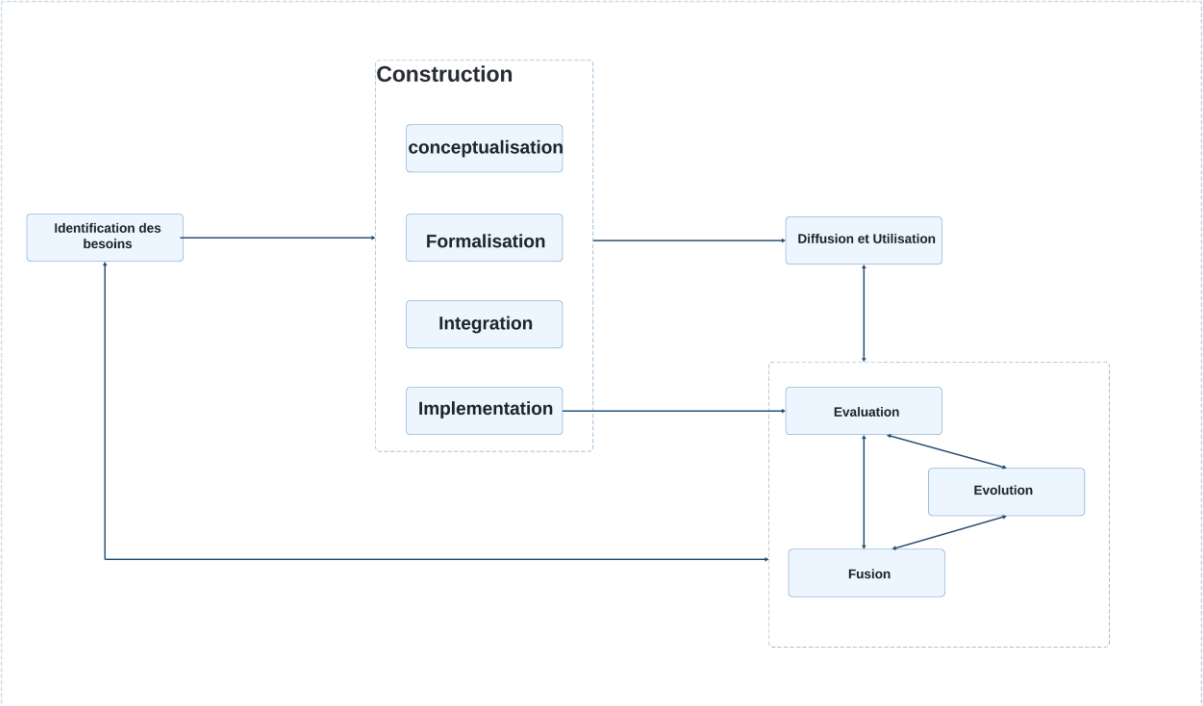


Figure 4 : Cycle de vie d'une ontologie

1.2.5. Méthodologie et outils de construction d'ontologies

Il existe plusieurs méthodologies et des outils pour la création des ontologies. Nous allons décrire les plus marquant selon les travaux de [22].

1.2.5.1. Les méthodologies

- Approche de construction d'ontologie de domaine à partir de grandes ontologies (SENSUS, Cyc, AKT, …) : Construire des ontologies de domaine à partir des grandes ontologies déjà existantes.
- Méthode de Uschold et King's : Basé sur l'expérience acquise lors du développement de l'ontologie d'Entreprise, the enterprise ontology.
- La méthodologie On-To-Knowledge1 (OTK) : La méthodologie On-To- Knowledge propose de construire une ontologie en tenant compte de comment l'ontologie va être utilisée par l'application plus tard (cette méthodologie est très dépendante de l'application).
- METHONTOLOGY : Cette méthodologie a été développée au sein du groupe d'ontologie à l'université polytechnique de Madrid.

1.2.5.2. Les outils

- TERMINAE, intègre des outils linguistiques, elle permet la visualisation des résultats des extracteurs de candidats-termes Lexter et/ou Syntex. Ces concepts doivent ensuite être triés par un expert et organisés hiérarchiquement, puis la sémantique du domaine est précisée à travers des axiomes.
- DOE (DIFFERENTIAL ONTOLOGY EDITOR) est un éditeur d'ontologie.
- ODE (ONTOLOGY DESIGN ENVIRONMENT), construction des ontologies au niveau connaissances (Méthodologie METHONTOLOGY).
- ONTOEDIT (ONTOLOGY EDITOR), est indépendant de tout formalisme. Il permet l'édition des hiérarchies de concepts et de relations et l'expression d'axiomes algébriques.
- PROTEGE 2000 : Il permet l'édition, le contrôle, la visualisation et l'extraction d'ontologie à partir des textes.
- ONTOLINGUA qui constitue une extension du langage KIF (KNOWLEDGE INTERCHANGE FORMAT). Ce serveur d'édition permet la fusion d'ontologie. L'ontologie est immédiatement représentée dans un formalisme.

- OILED (OIL EDITOR) est un éditeur d'ontologie s'inspirant du formalisme OIL. Un éditeur de petites ontologies avec un moteur d'inférence pour tester la cohérence de l'ontologie construite.

1.3. Ontologies lexicales Wordnets

L'une des ressources sémantiques lexicales les plus connues et les plus utilisées dans les domaines du traitement automatique des langues et du web sémantique est le Princeton **WordNet** (PWN) . Elle dispose également des équivalents pour d 'autres langues. Parmi ces derniers, on peut citer les wordnets développés dans le cadre des projets EuroWordNet [2], BalkaNet [23], ainsi que le Open Multilingual Wordnet, qui normalise et fusionne tous les wordnets dont la redistribution est autorisée par leurs auteurs, et inclut à ce jour des wordnets pour 27 langues. Initialement, le PWN était pourtant développé dans un contexte psycho lexicographique, inspiré par des travaux sur les processus cognitifs d 'accès au lexique.

WordNet (PWN) se construit à partir de la relation de synonymie. Pour les concepteurs du thesaurus, deux expressions sont synonymes si le remplacement de l'une par l'autre ne change pas la valeur de vérité de la proposition. Pour autant, la substituabilité et des synonymes dans tous les contextes n'est pas nécessaire : il suffit que les synonymes soient remplaçables dans certains contextes. Cela permet d'admettre qu'un même lexème puisse être associé à plusieurs synsets, ce qui correspond à la flexibilité des langues naturelles.

Le *synset* (ensemble de synonymes) est la composante atomique sur laquelle repose WordNet. Un synset correspond à un groupe de mots interchangeables, dénotant un sens ou un usage particulier. Un synset est défini d'une façon différentielle par les relations qu'il entretient avec les sens voisins.

Les **noms** et **verbes** sont organisés en hiérarchies. Des relations d'hyperonymie « est-un » et d'hyponymie relient les concepts généraux des noms et des verbes avec leurs « spécialisations ». Au niveau racine, ces hiérarchies sont organisées en types de base. Le réseau des noms est bien plus profond que celui des autres parties du discours. A titre indicatif, les deux premiers niveaux de la hiérarchie des noms se constituent des concepts abstraits suivants :

- **Abstraction**: attribute, measure/quantity/amount, relation, set, space, time…
- **Human action**: activity, communication, distribution, inactivity, judgment, leaning, legitimation, motivation, proclamation, production, speech act…

- **Entity**: anticipation, causal agent, enclosure, expanse, location, physical object, sky, substance, thing…
- **Event**: group action, natural event, might-have-been, migration, miracle, nonevent, social event…
- **Group, grouping**: association, biological group, people, collection, aggregation, community, ethnic group, kingdom, multitude, population, race, rare-earth element…
- **Phenomenon**: effect/result, levitation, fortune/chance, rebirth, natural phenomenon, process, pulsation…
- **Possession**: assets, circumstances, property/material possession, transferred property, treasure…
- **Psychological feature**: cognition/knowledge, feeling, motivation/need…
- **State**: action/activity, existence, state of mind, condition, conflict, damnation, death, degree, dependency, disorder, employment, end, freedom, antagonism, immaturity, imminence, imperfection, integrity, maturity, omnipotence, perfection, physiological state, relationship, state of affairs, status, temporary state, natural state…

L'organisation des **adjectifs** est différente. Un sens « tête » joue un rôle d'attracteur ; des adjectifs « satellites » lui sont reliés par des relations de synonymie. On a donc une partition de l'ensemble des adjectifs en petits groupes. Les **adverbes** sont le plus souvent définis par les adjectifs dont ils dérivent. Ils héritent donc de la structure des adjectifs.

La version 3.0, la plus récente (janvier 2007) compte 117 597 synsets et 207 016 lemmes.

Relations sémantique (entre synsets)

Relation	**Entre**	**Nombre**
Hypernym/Hyponym	Verbe / verbe	13 124
	Nom / nom	75 134
Instance Hyponym	Nom / nom	8 515
Part	Nom / nom	8 874
Member	Nom / nom	12 262
Substance	Nom / nom	793
Attribute	Adjectif / nom	643
Verb Group	Verbe / verbe	1 748

Verb Entailment	Verbe / verbe	409
Verb Cause	Verbe / verbe	219
Adjective Similar	Adjectif / adjectif	22 622
Topic Domain	Nom / adjectif	1 108
	Nom / nom	4 146
	Nom / adverbe	37
	Nom / verbe	1 236
Region Domain	Nom / adjectif	75
	Nom / nom	1 246
Usage Domain	Nom / adjectif	227
	Nom / nom	563
	Nom / adverbe	73
	Nom / verbe	14
See Also	Adjectif / adjectif	2 683

Tableau 1: Comptage des relations sémantiques de WordNet 2.

1.3.1. Méthodologie de construction des Wordnets

Les techniques automatiques de développement de wordnets se répartissent selon celle des deux approches principales qu'elles mettent en œuvre : l 'approche par fusion et l 'approche par extension [2]. Dans le cas de l 'approche par fusion, un wordnet pour une langue donnée est créé indépendamment des autres wordnets existants, en exploitant au mieux des ressources monolingues disponibles ; dans un second temps, le wordnet ainsi créé peut être aligné avec les wordnets disponibles pour d 'autres langues. Dans le cas de l 'approche par extension, l'inventaire de sens du PWN est conservé (mêmes identifiants de synsets, mêmes relations entre synsets) et on cherche à peupler les synsets avec des littéraux de la langue cible, par exemple par désambiguïsation et traduction des littéraux anglais présents dans le PWN.

1.3.2. Le développement automatique de wordnets

L 'approche par extension repose sur l 'approximation selon laquelle les concepts (les sens) et les relations entre eux sont indépendants de la langue, au moins pour une bonne part. l 'approche par extension est ainsi très utilisée pour le développement de wordnets, par exemple

dans les projets BalkaNet [23] et MultiWordnet. Le premier avantage est naturellement un coût de développement bien plus bas que pour l 'approche par fusion. Le second avantage est que les wordnets produits sont alignés sur le PWN et donc également alignés entre eux, ce qui permet d'envisager leur utilisation dans des applications multilingues, telles que la traduction automatique ou l'extraction d 'informations.

Les mises en œuvre de l 'approche par extension varient selon le type de ressources qui sont disponibles pour la construction d 'un wordnet dans une langue donnée. Les premiers travaux en ce sens utilisaient directement des dictionnaires électroniques bilingues, en essayant d'aligner les entrées des dictionnaires avec wordnet. Le problème qui apparaît immédiatement est la difficulté de la désambiguïsation des entrées des dictionnaires bilingues. De plus, les dictionnaires bilingues ont souvent une couverture limitée et ne sont pas nécessairement disponibles pour toutes les paires de langues.

Une façon de surmonter ces difficultés est d 'utiliser des lexiques bilingues ou multilingues extraits de corpus parallèles. L 'hypothèse principale qui sous-tend ces travaux est que les différents sens d'un même mot qui est ambigu dans une langue sont souvent traduits par des mots différents dans une autre langue. De plus, si deux mots distincts, voire plus, sont traduits par le même mot dans une autre langue, ils sont souvent sémantiquement reliés, voire synonymes. Ceci permet de désambiguïser les mots polysémiques ou à l 'inverse de créer des liens synonymiques. Les corpus parallèles ont été utilisés pour induire des synsets pour une nouvelle langue par divers auteurs.[24]

La troisième famille de travaux, plus récente, cherche à exploiter au mieux les ressources libres et collaboratives telles que Wikipedia. Wikipedia est une ressource encyclopédique libre disponible dans de nombreuses langues. Chaque article peut notamment être muni de catégories et être relié à des articles qui lui correspondent dans les Wikipedia d 'autres langues. De nouveaux wordnets ont été induits en associant des pages de Wikipedia avec des synsets de wordnet, en utilisant des informations structurelles pour associer des catégories Wikipedia aux synsets ou en extrayant des mots clés à partir des articles de Wikipedia. Un modèle vectoriel permettant d 'associer des pages Wikipedia à wordnet a été proposé par divers auteurs; Les approches les plus abouties utilisent Wikipedia et d'autres ressources wiki, notamment Wiktionary (version française : le Wiktionnaire) pour produire des wordnets dans de multiples langues. [24]

La plupart des méthodes de développement de wordnets relevant de l 'approche par extension peuvent être décomposées en deux étapes :

- Une première étape consistant à construire, si l 'on n 'en dispose pas déjà, un lexique bilingue ou multilingue ; l 'utilité de disposer de lexiques impliquant d 'autres langues en plus de la langue source et de la langue cible.
- Une deuxième étape consistant à rapprocher les entrées de ces lexiques bilingues ou multilingues, dont l 'une des langues est la langue source avec les synsets du wordnet source (PWN), afin de construire le plus grand nombre possible de couples (littéral de la langue cible, synset) candidats ; l 'idéal est de disposer d 'une mesure de confiance sur ces candidats afin de pouvoir sélectionner les meilleurs d 'entre eux et éliminer les autres.

1.4. Les langues peu dotées [4]

Une langue peu dotée est une langue qui dispose de peu de ressources linguistiques informatisées. Ces ressources sont entre autres un lexique, un correcteur orthographique, un analyseur syntaxique, et une Terminologie spécialisée.

- Lexique : Un lexique est une liste de mots d'une langue avec leurs informations associées, telles que les définitions, les catégories grammaticales et les relations sémantiques.
- Correcteur orthographique : Un correcteur orthographique est un outil qui détecte et suggère des corrections pour les erreurs d'orthographe dans un texte. Pour les langues peu dotées, il est souvent nécessaire de créer des dictionnaires spécifiques et d'entraîner des modèles adaptés.
- Analyseur syntaxique : Un analyseur syntaxique décompose les phrases en unités grammaticales qui sont les mots, groupes nominaux et verbes et identifie leurs relations.
- Terminologie spécialisée : se réfère aux termes techniques et spécifiques à un domaine particulier. Pour les langues peu dotées, la création de terminologies spécialisées peut être cruciale pour des applications telles que la traduction automatique ou la recherche d'information.

Une langue est considérée comme peu dotée si elle a un lexique restreint, c'est-à-dire un nombre limité de mots documentés. Si ne dispose pas d'un correcteur adapté, cela peut indiquer qu'elle est peu dotée. Si une langue manque d'analyseurs syntaxiques pré-entraînés ou spécifiques, cela peut être un signe de faible dotation. Si une langue ne dispose pas de

terminologie adaptée à des domaines tels que la médecine, la technologie ou le droit, elle peut être considérée comme peu dotée. Cependant, il est important de noter que ces mesures ne sont pas absolues.[4], [25]

1.5. Conclusion

Dans ce chapitre, nous avons exploré les différents concepts d'ontologie et étudié les différents types d'ontologies, notamment les ontologies de représentation des connaissances, les ontologies supérieures, génériques, de domaine, de tâches et d'application. Chaque type d'ontologie sert un objectif spécifique et fournit un cadre pour modéliser et organiser les connaissances d'une manière structurée et significative. De plus, nous avons passé en revue plusieurs méthodologies et outils de construction d'ontologies, tels que la construction à partir de grandes ontologies existantes, la méthode de Uschold, la méthodologie On-To-Knowledge et METHONTOLOGY. Ces méthodologies fournissent des lignes directrices et des techniques pour créer des ontologies robustes et adaptées aux besoins spécifiques d'une application.

En ce qui concerne les ressources lexicales sémantiques, toute la littérature sur les ressources lexicales sémantiques est unanime, le Princeton WordNet (PWN) est une référence dans ce domaine. Le PWN a inspiré le développement de wordnets équivalents dans d'autres langues, tels que EuroWordNet, BalkaNet, AsianWordnet et Open Multilingual Wordnet. Ces wordnets sont des ressources précieuses pour le traitement automatique des langues et le web sémantique. Pour la construction de ces ressources, la littérature révèle deux approches principales : l'approche par fusion et l'approche par extension. L'approche par extension, qui présume que les concepts et relations sont indépendants de la langue, a été largement utilisée pour développer des wordnets efficaces et multilingues. Les méthodes de développement de wordnets impliquent généralement la construction de lexiques bilingues ou multilingues et le rapprochement de ces lexiques avec les synsets du wordnet source pour créer des couples littéral-synset candidats.

Ce chapitre fourni une appréhension des ontologies et des ressources lexicales sémantiques ; fourni un panorama des outils et des méthodes de constructions de ces ressources et jette ainsi les bases de la deuxième partie de notre travail sur la conceptualisation et la construction de ces ressources.

Chapitre 2: Conception et Implémentation des réseaux lexicaux sémantiques

2.1. Introduction

Ce chapitre, en première partie, traite de la méthodologie et de la conception d'un réseau lexical sémantique pour les langues peu dotées en utilisant le générateur RDF. Nous proposons une approche générale visant à générer une représentation RDF à partir d'un dictionnaire bilingue, en analysant aussi bien sa structure que son contenu pour aboutir à une ressource plus riche. Le modèle repose sur les concepts généraux de WordNet, mais pas tous, centrés sur l'identification des classes et des relations présentes dans leurs dictionnaires bilingues. La deuxième phase consiste à décrire la conception et l'implémentation de la plateforme pour les langues peu dotées. Nous présentons l'architecture globale de la solution, les différents diagrammes de conceptions, les choix technologiques effectués et les procédures de mise en œuvre.

2.2. Conception du réseau lexical sémantique

La construction automatique d'un réseau lexical sémantique est un processus complexe qui implique la compréhension et l'analyse des mots, de leurs significations et de leurs relations dans un contexte donné. Cette tâche est cruciale pour divers domaines, notamment le traitement automatique des langues et l'intelligence artificielle. Ce chapitre rapporte sur la méthodologique, les ressources, l'architecture du générateur RDF des dictionnaires bilingues.

2.2.1. Démarche générale

L'approche que nous proposons consiste à produire une représentation RDF à partir d'un dictionnaire bilingue, en analysant à la fois sa structure et son contenu pour obtenir une ressource plus riche que ne fournirait une analyse prise indépendamment. Notre modèle sera base sur les concepts généraux de wordnet représenté par la figure 1. Faute de ressources disponible, Nous optons pour la construction d'une ressource indépendante ; une approche similaire à la méthode par fusion, mentionnée dans le chapitre 2 sur l'état de l'art. La méthode de génération est répartie comme suit :

- Indentification des classes et les relations dans nos dictionnaires bilingues,
- Prétraitement des ressources dictionnairiques,

- Générer un graphe RDF vide avec les classes et les relations identifiés, peuples ce graphe avec le dictionnaire prétraiter.
- Et enfin générer le fichier RDF final contenant la nouvelle structure du dictionnaire.

A la différence de l'approche par fusion, notre RDF final ne sera pas aligné sur les ressources existantes, et donc non conforme à PWN.

a) Identification des classes et des relations dans le document représentant le dictionnaire conforment au modèle RDF de PWN [1]:

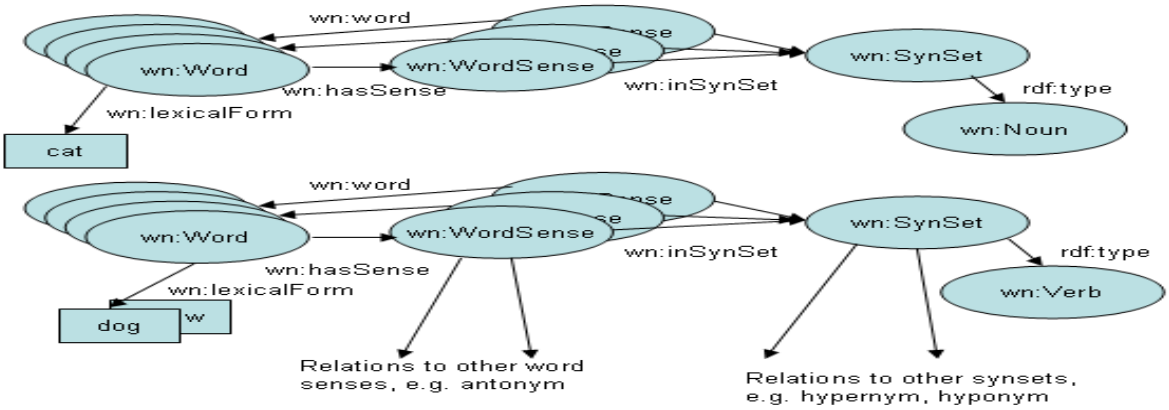


Figure 5: représentation des concepts de PWN

Les concepts et les relations qui ressortent de l'analyse des fichiers sont suivants :

- **Mot** : Cette classe représente un mot dans le dictionnaire. Il a une propriété **a_pour_définition** pour stocker la définition du mot et une propriété **est_en_langue** pour indiquer la langue du mot.
- **Langue** : Cette classe représente une langue dans le dictionnaire. Il a une propriété **a_pour_définition** pour stocker le nom de la langue.
- **Traduction** : Cette classe représente la traduction d'un mot dans une autre langue. Il a une propriété **a_pour_définition** pour stocker la traduction du mot.
- **Exemple_de_phrase** : Cette classe représente un exemple de phrase dans laquelle le mot est utilisé. Il a une propriété **a_pour_définition** pour stocker l'exemple de phrase.
- **Synonyme** : Cette classe représente un synonyme d'un mot. Il a une propriété **a_pour_définition** pour stocker le synonyme du mot.

[1] https://www.w3.org/2001/sw/BestPractices/WNET/wordnet-sw-20040713.html

- **Expression** : Cette classe représente une expression ou une phrase idiomatique associée à un mot. Il a une propriété **a_pour_définition** pour stocker l'expression.
- **PartOfSpeech** : Cette classe représente la partie du discours d'un mot (par exemple, nom, verbe, adjectif, etc.). Il a une propriété **a_pour_définition** pour stocker la partie du discours du mot.
- **a_pour_définition** : Cette propriété relie un mot à sa définition. La définition est stockée sous forme de littéral.
- **Est_en_langue** : Cette propriété relie un mot à la langue dans laquelle il est utilisé. La langue est représentée par une instance de la classe **Langue**.
- **est_traduit_par** : Cette propriété relie un mot à sa traduction dans une autre langue. La traduction est représentée par une instance de la classe **Traduction**.
- **est_utilisé_dans** : Cette propriété relie un mot à un exemple de phrase dans lequel il est utilisé. L'exemple de phrase est représenté par une instance de la classe **Exemple_de_phrase**.
- **est_une** : Cette propriété est utilisée pour indiquer que le mot est une instance d'une certaine classe ou propriété.
- **est_synonyme_de** : Cette propriété relie un mot à son synonyme. Le synonyme est représenté par une instance de la classe **Synonyme**.
- **a_pour_partie_du_discours** : Cette propriété relie un mot à sa partie du discours. La partie du discours est représentée par une instance de la classe **PartOfSpeech**.

b) Prétraitement des ressources dictionnairiques : cela consiste à :

- Convertir les fichiers en **plain texte** s'ils sont dans un autre format (généralement en PDF) ;
- Nettoyer les numérotations de pages, les entêtes, pieds de page et les retours à la ligne ;
- Formatter le document afin de le faire correspondre au paterne Regex du script python devant servir à l'extraction des instances ;

c) Générer un graphe RDF vide avec les classes et les relations identifiés :

Nous utilisons la librairie **rdflib**[2] de python pour la manipulation du graphe RDF. La création du graphe par rdflib suit les étapes suivantes :

[2] https://rdflib.readthedocs.io/en/stable/: RDFLib est un package Python pur pour travailler avec RDF .

1. Initialiser un graphe RDF vide G.

2. Définir l'espace de noms pour l'ontologie O.

3. Définir l'espace de noms pour RDF Schema S.

4. Lier l'espace de noms O au graphe G.

5. Initialiser un ensemble de classes C à vide.

6. Initialiser un ensemble de propriétés P à vide.

7. Ajouter les classes prédéfinies à C :

"Mot", "Langue", "Traduction",

"Exemple_de_phrase", "Synonyme",

"Expression", "PartOfSpeech".

8. Ajouter les propriétés prédéfinies à P : "a_pour_définition",

"est_en_langue","est_traduit_par","est_utilisé_dans",est_une",

"est_synonyme_de", "a_pour_partie_du_discours".

9. Pour chaque classe c dans C :

9.1. Ajouter c en tant que classe RDF dans G avec le type $S:Class$.

10. Pour chaque propriété p dans P :

10.1. Ajouter p en tant que propriété RDF dans G avec le type $rdf:Property$.

11. Pour chaque propriété p dans P :

11.1. Si p a un domaine d, ajouter (p, d) en tant que domaine dans G.

11.2. Si p a une plage r, ajouter (p, r) en tant que plage dans G.

12. Définir la fonction $createLanguageEntities(G, O, C, P, l1, l2)$:

12.1. Créer des nœuds $n1$ et $n2$ pour les langues $l1$ et $l2$ en utilisant l'espace de noms O.

12.2. Ajouter $n1$ et $n2$ en tant qu'instances de la classe "Langue" dans G.

12.3. Ajouter les définitions de $l1$ et $l2$ aux nœuds respectifs dans G.

12.4. Renvoyer $n1$ et $n2$.

L'initialisation produit la structure suivante :

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix onto: <http://www.example.org/onto/dictionary#> .

onto:Mot a rdfs:Class .
onto:Langue a rdfs:Class .
onto:Traduction a rdfs:Class .
onto:Exemple_de_phrase a rdfs:Class .
onto:Synonyme a rdfs:Class .
onto:Expression a rdfs:Class .
onto:PartOfSpeech a rdfs:Class .

onto:a_pour_définition a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range rdfs:Literal .

onto:est_en_langue a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range onto:Langue .

onto:est_traduit_par a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range onto:Traduction .

onto:est_utilisé_dans a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range onto:Exemple_de_phrase .

onto:est_une a rdf:Property ;
    rdfs:domain onto:Mot .

onto:est_synonyme_de a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range onto:Synonyme .

onto:a_pour_partie_du_discours a rdf:Property ;
    rdfs:domain onto:Mot ;
    rdfs:range onto:PartOfSpeech .
```

Figure 6: représentation RDF des classes et relation des dictionnaires

d) Extraction et peuplement de la représentation RDF :

Cette phase implique l'extraction d'informations d'un fichier d'entrée, la création de nœuds RDF pour représenter les mots, les traductions, les exemples de phrases et les parties du discours, et l'établissement de relations entre ces nœuds en fonction des propriétés définies dans l'ontologie. Le résultat est un graphe RDF qui représente le dictionnaire bilingue. Elle se décompose en étapes suivantes :

1. Ouverture du fichier :
   - Ouvrir le fichier d'entrée spécifié par le chemin de fichier fourni en argument
2. Lecture des lignes :
   - Traiter les lignes par paires, car chaque paire de lignes représente un mot et ses informations associées.
3. Extraction du mot et de l'abréviation :
   - Pour chaque paire de lignes :
     - Extraire le mot de la première ligne.
     - Extraire l'abréviation de la deuxième ligne. L'abréviation indique la partie du discours du mot.
4. Correspondance de modèle regex :
   - Utiliser le paterne regex `r'([A-Z]+)\s+(.*?)(?=\.)(.*)'` pour extraire la traduction et les exemples de la deuxième ligne :
     - `[A-Z]+` : Capture une ou plusieurs lettres majuscules, représentant l'abréviation.
     - `.*?` : Capture tout ce qui suit l'abréviation jusqu'au premier point (`.`), représentant la traduction.
     - `(.*)` : Capture tout ce qui suit le point, représentant les exemples de phrases.
5. Création du nœud Mot :
   - Pour chaque mot extrait :
     - Créer un nœud RDF représentant le mot.
     - Ajouter le nœud au graphe RDF.
     - Définir les propriétés du nœud :
       - `RDF.type` : Défini à `onto:Mot`.
       - `RDFS.label` : Défini au mot extrait.
       - `properties["a_pour_définition"]` : Défini au mot extrait.
       - `properties["est_en_langue"]` : Défini au nœud de langue correspondant.
6. Création du nœud Traduction :
   - Pour chaque traduction extraite :
     - Créer un nœud RDF représentant la traduction.
     - Ajouter le nœud au graphe RDF.
     - Définir les propriétés du nœud :
       - `RDF.type` : Défini à `onto:Traduction`.
       - `RDFS.label` : Défini à la traduction extraite.

- `properties["a_pour_définition"]` : Défini à la traduction extraite.
- `properties["est_en_langue"]` : Défini au nœud de langue correspondant.

7. Création des nœuds Exemple_de_phrase :
   - Pour chaque exemple de phrase extrait :
     - Créer un nœud RDF représentant l'exemple de phrase.
     - Ajouter le nœud au graphe RDF.
     - Définir les propriétés du nœud :
       - `RDF.type` : Défini à `onto:Exemple_de_phrase`.
       - `RDFS.label` : Défini à l'exemple de phrase extrait.
       - `properties["a_pour_définition"]` : Défini à l'exemple de phrase extrait.
8. Création du nœud PartOfSpeech :
   - Pour chaque abréviation extraite :
     - Déterminer la partie du discours correspondante en utilisant un tableau de correspondance.
     - Créer un nœud RDF représentant la partie du discours.
     - Ajouter le nœud au graphe RDF.
     - Définir les propriétés du nœud :
       - `RDF.type` : Défini à `onto:PartOfSpeech`.
       - `RDFS.label` : Défini à la partie du discours.
       - `properties["a_pour_définition"]` : Défini à la description de la partie du discours.
9. Établissement des relations :
   - Pour chaque mot et ses nœuds associés :
     - Relier le nœud Mot au nœud Traduction en utilisant la propriété `properties["est_traduit_par"]`.
     - Relier le nœud Mot aux nœuds Exemple_de_phrase en utilisant la propriété `properties["est_utilisé_dans"]`.
     - Relier le nœud Mot au nœud PartOfSpeech en utilisant la propriété `properties["a_pour_partie_du_discours"]`.
10. Répétition pour chaque paire de lignes :
    - Répéter les étapes 3 à 9 pour chaque paire de lignes dans le fichier d'entrée.
11. Fermeture du fichier:
    - Une fois toutes les lignes traitées, fermer le fichier d'entrée.

Ces différentes étapes devraient être exécute sur un fichier structuré comme suit :

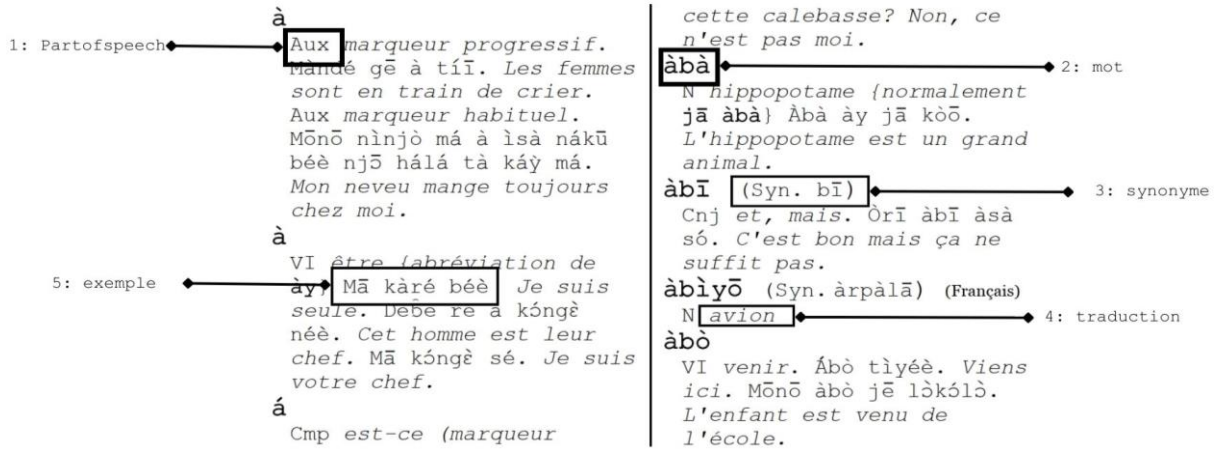


Figure 7: extrait d'un dictionnaire bilingue KabaNa-Français (JM. Keegan, 2014)

Le resultat fourni en sortie un fichier RDF dont voici un extrait:

```
<rdf:Description rdf:nodeID="Nf20ac4a8c15e4826b74c489945cde5f2">
    <rdf:type rdf:resource="http://www.example.org/onto/dictionary#Mot"/>
    <onto:a_pour_définition>dìsā</onto:a_pour_définition>
    <rdfs:label>dìsā</rdfs:label>
    <onto:est_en_langue rdf:resource="http://www.example.org/onto/dictionary#kabaNa"/>
    <onto:est_traduit_par rdf:nodeID="N367108b98d71434e85b6e071bf491904"/>
    <onto:est_utilisé_dans rdf:nodeID="N58a0b84234f44e58a45fd93296958f63"/>
    <onto:est_utilisé_dans rdf:nodeID="Ndfe2d495c702449c9da27a64a0168105"/>
    <onto:est_utilisé_dans rdf:nodeID="Nd1a79e482bcf4c27b195440b86b29bc9"/>
    <onto:a_pour_partie_du_discours rdf:nodeID="N7e2770e618c6452a800215639e630686"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="N33a06f4e13074498954b264ba4b26bb6">
    <rdf:type rdf:resource="http://www.example.org/onto/dictionary#Exemple_de_phrase"/>
    <onto:a_pour_définition>Yē í àkàń</onto:a_pour_définition>
  </rdf:Description>
```

2.3. Conception de la plateforme

2.3.1. Étude des besoins

2.3.1.1. Besoins fonctionnels :

- **Gestion des données structurées** : La plateforme doit être capable de gérer des données structurées sous forme de graphes RDF. Cela implique la création, la manipulation et la sérialisation de graphes RDF.
- **Interactions avec les fichiers** : La plateforme doit être capable de lire des fichiers contenant des données de dictionnaire et de les transformer en graphes RDF. Elle doit également être capable de sérialiser ces graphes en fichiers RDF.
- **Services Web** : La plateforme doit fournir des services web pour la recherche d'entités et la récupération de toutes les entités et leurs traductions. Elle doit également fournir un service pour le téléchargement de fichiers.
- **Interface utilisateur** : La plateforme doit avoir une interface utilisateur qui permet aux utilisateurs de rechercher des entités, de voir les résultats de la recherche, de télécharger des fichiers et de visualiser les classes et les liens.

2.3.1.2. Besoins non fonctionnels :

- **Authentification** : La plateforme doit avoir un système d'authentification pour sécuriser certaines requêtes HTTP.
- **Internationalisation** : la plateforme doit prendre en charge l'internationalisation. Cela signifie qu'elle doit être capable de présenter l'interface utilisateur et les données dans différentes langues.
- **Performance** : La plateforme doit être capable de gérer un grand nombre de requêtes et de données. Elle doit également être capable de répondre rapidement aux requêtes des utilisateurs.
- **Gestion des erreurs** : La plateforme doit être capable de gérer les erreurs de manière appropriée, par exemple lorsqu'un fichier n'est pas trouvé ou lorsqu'une requête HTTP échoue.

2.3.2. Architecture logicielle[3]

Compte tenu de la complexité de notre système, nous avons opté pour une architecture oriente service ou Service Oriented Architecture (SOA). Le résultat de l'analyse des besoins

[3] Architecture Microservices : Guide complet (marjory.io)

impose une séparation claire des responsabilités entre les différentes couches et composants du système.

L'architecture de micro-services est une approche de conception logicielle qui transforme la façon dont les applications sont intégrées et déployées. Contrairement à l'architecture monolithique traditionnelle, qui regroupe toutes les fonctionnalités dans un seul bloc de code, l'architecture micro-services divise une application en un ensemble de services et d'outils plus petits et indépendants. Ce système dispose des caractéristiques suivantes :

- Services à composants multiples : Les micro-services sont constitués de services de composant à couplage lâche individuels qui peuvent être développés, déployés, exploités, modifiés et redéployés sans compromettre le fonctionnement des autres services ou l'intégrité d'une application. Cela permet un déploiement rapide et facile des fonctionnalités individuelles d'une application.
- Hautement maintenable et testable : Les micro-services permettent aux équipes de tester de nouvelles fonctionnalités et de revenir en arrière en cas de problème. Cela facilite la mise à jour du code et accélère la commercialisation de nouvelles fonctionnalités. De plus, cela simplifie le processus d'isolement et de correction des défaillances et des bugs dans les services individuels.
- Développée par des petites équipes : Les petites équipes indépendantes développent généralement un service au sein de micro-services, ce qui encourage les équipes à adopter des pratiques Agile et DevOps. Les équipes sont habilitées à travailler de manière autonome et à agir rapidement, ce qui réduit les temps de cycle de développement.
- Organisée en fonction des capacités métier : Une approche de micro-services organise les services en fonction des capacités métier. Les équipes sont transverses et disposent de toutes les compétences requises pour le développement, travaillant à la réalisation d'une fonctionnalité individuelle.
- Infrastructure automatisée : Les équipes qui développent et maintiennent des micro-services utilisent généralement des pratiques d'automatisation de l'infrastructure telles que l'intégration continue (CI), la livraison continue (CD) et le déploiement continu (CD également). Cela permet aux équipes de développer et de déployer chaque service indépendamment sans impact pour les autres équipes, mais aussi de déployer une nouvelle version d'un service côte à côte avec la version précédente.

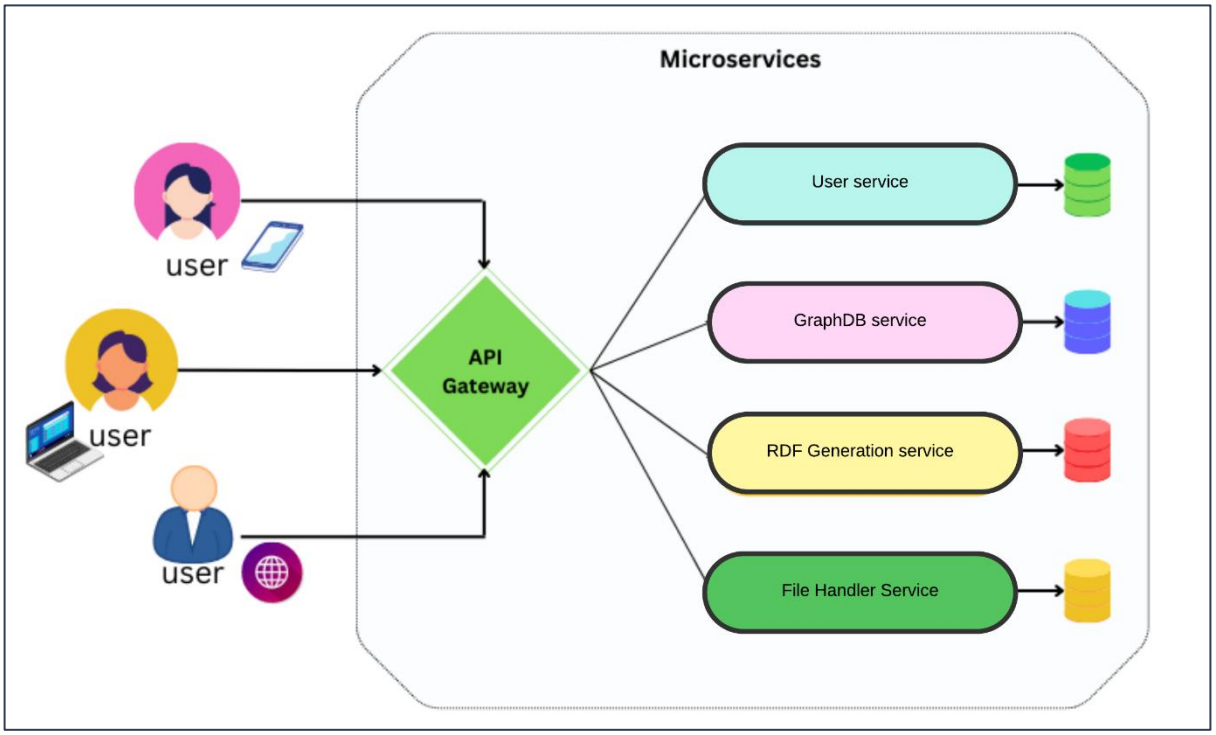


Figure 8: Architecture micro-service du système

2.3.3. Modélisation UML du générateur RDF

a. Diagramme de cas d'utilisation :

La plateforme est conçue pour être utilisée par deux types d'utilisateurs : toute personne disposant d'un accès a internet et les administrateurs.

- Les clients ont la possibilité de :
  - Faire une recherche globale d'un mot dans toutes les langues disponibles ;
  - Faire une recherche spécifique à une langue ;
- D'autre part, les administrateurs ont la capacité de générer un fichier RDF :
  - Visualiser les informations sur le graphe ;
  - Importer des fichiers.
  - Générer des dictionnaires RDF
  - S'authentifier sur la plateforme.

- Un générateur qui est un outil utilisé par la plateforme.

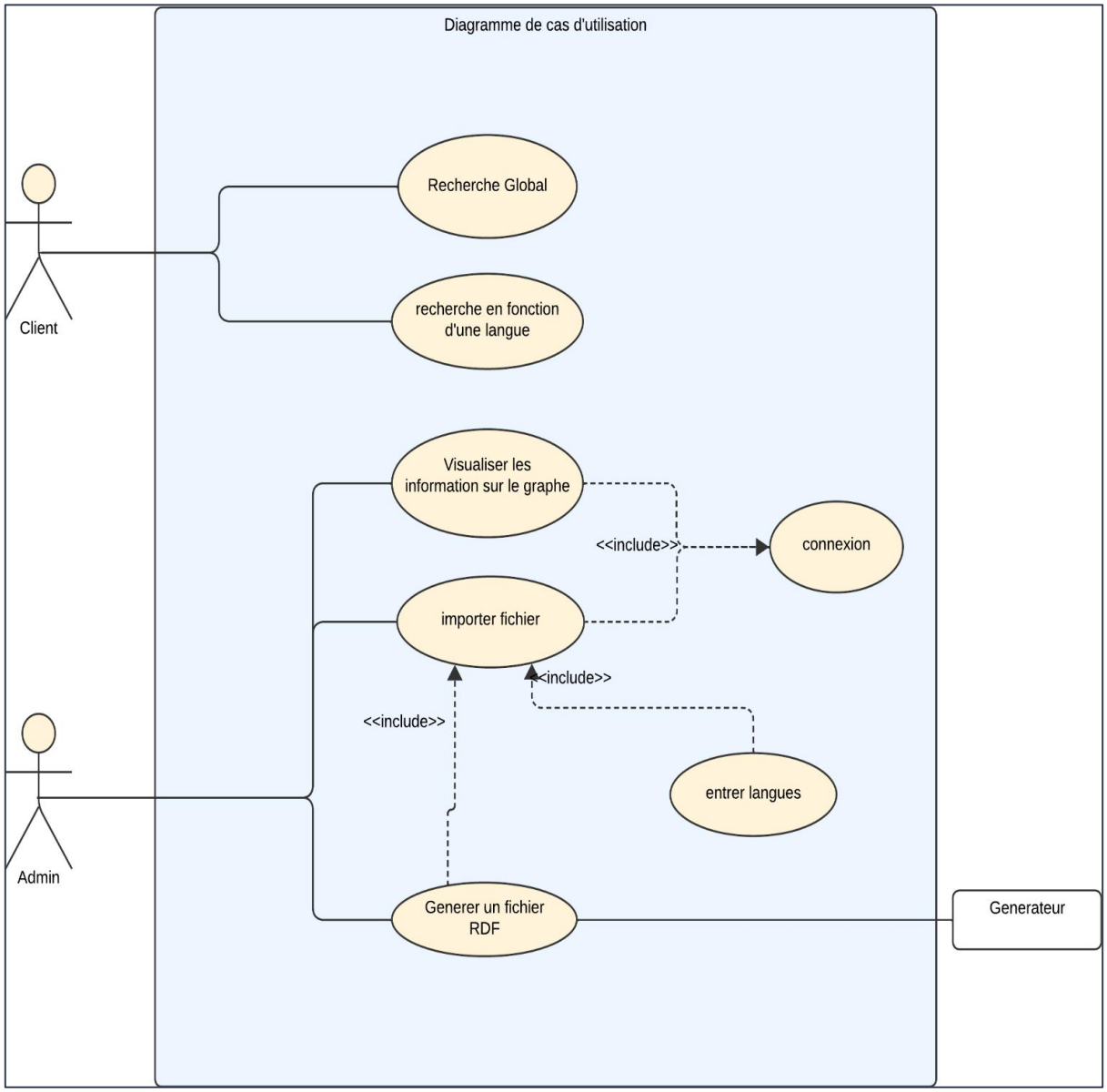


Figure 9:diagramme de cas d'utilisation du système

b. Diagramme d'activité

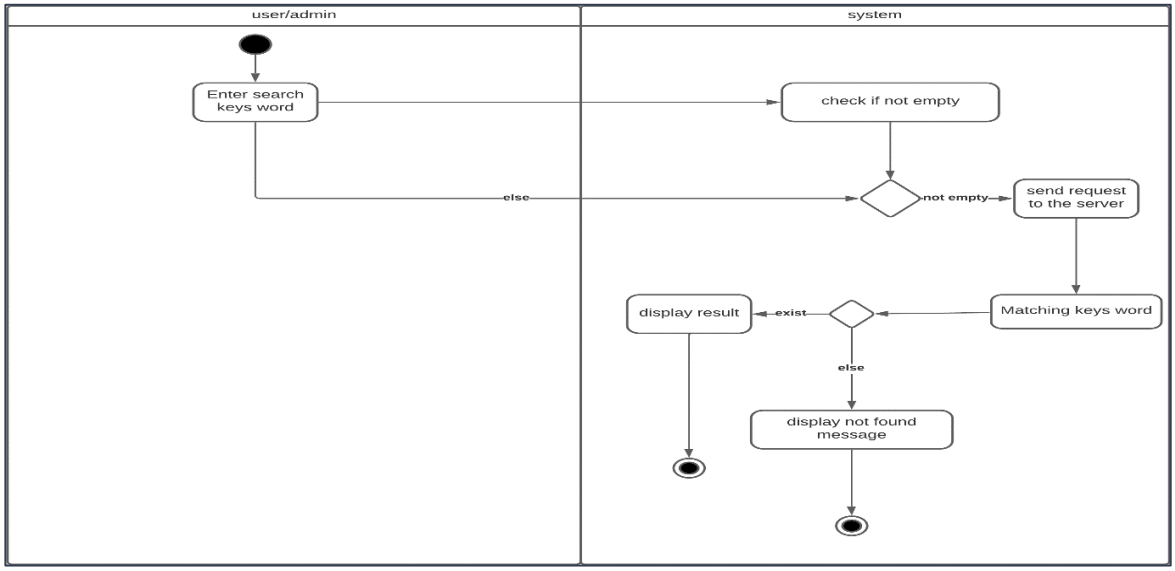


Figure 10: diagramme d'activité pour la recherche.

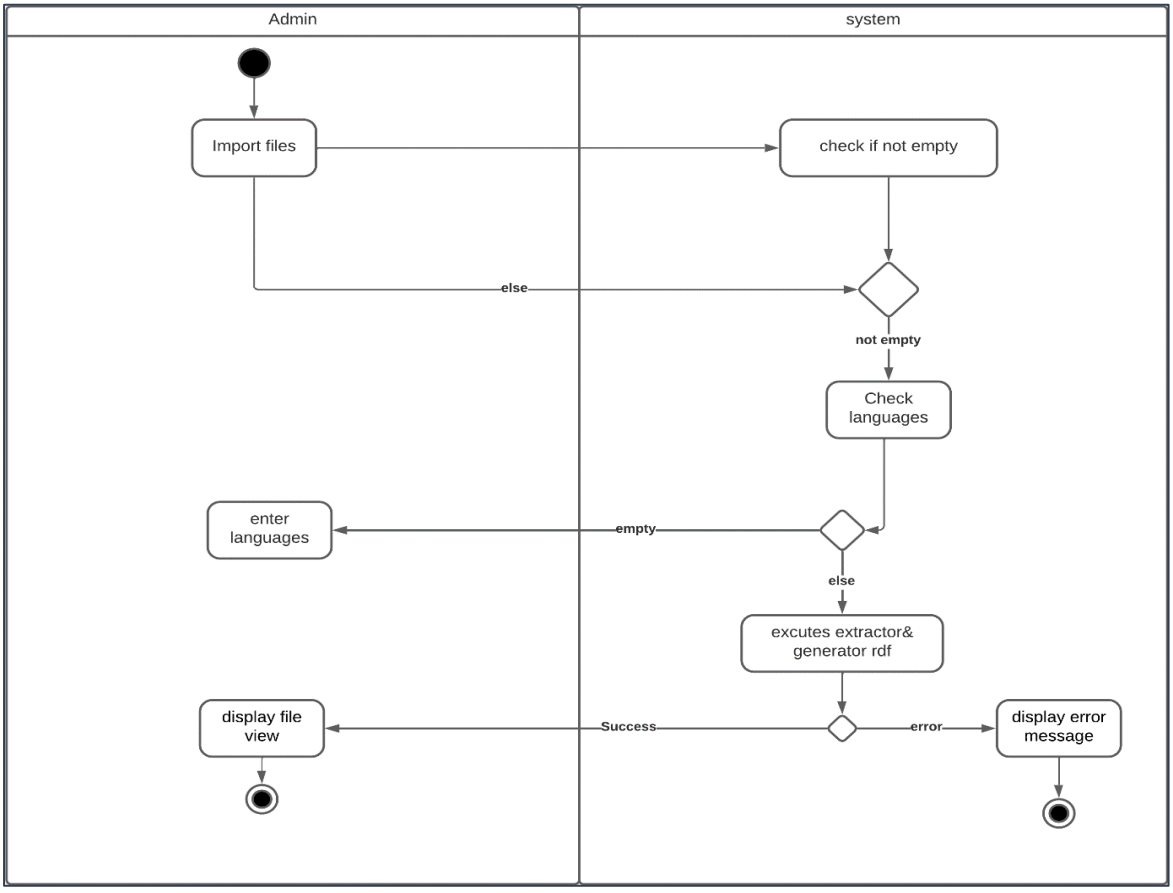


Figure 11: diagramme d'activité pour la génération de fichier RDF

c. Diagrammes de séquence

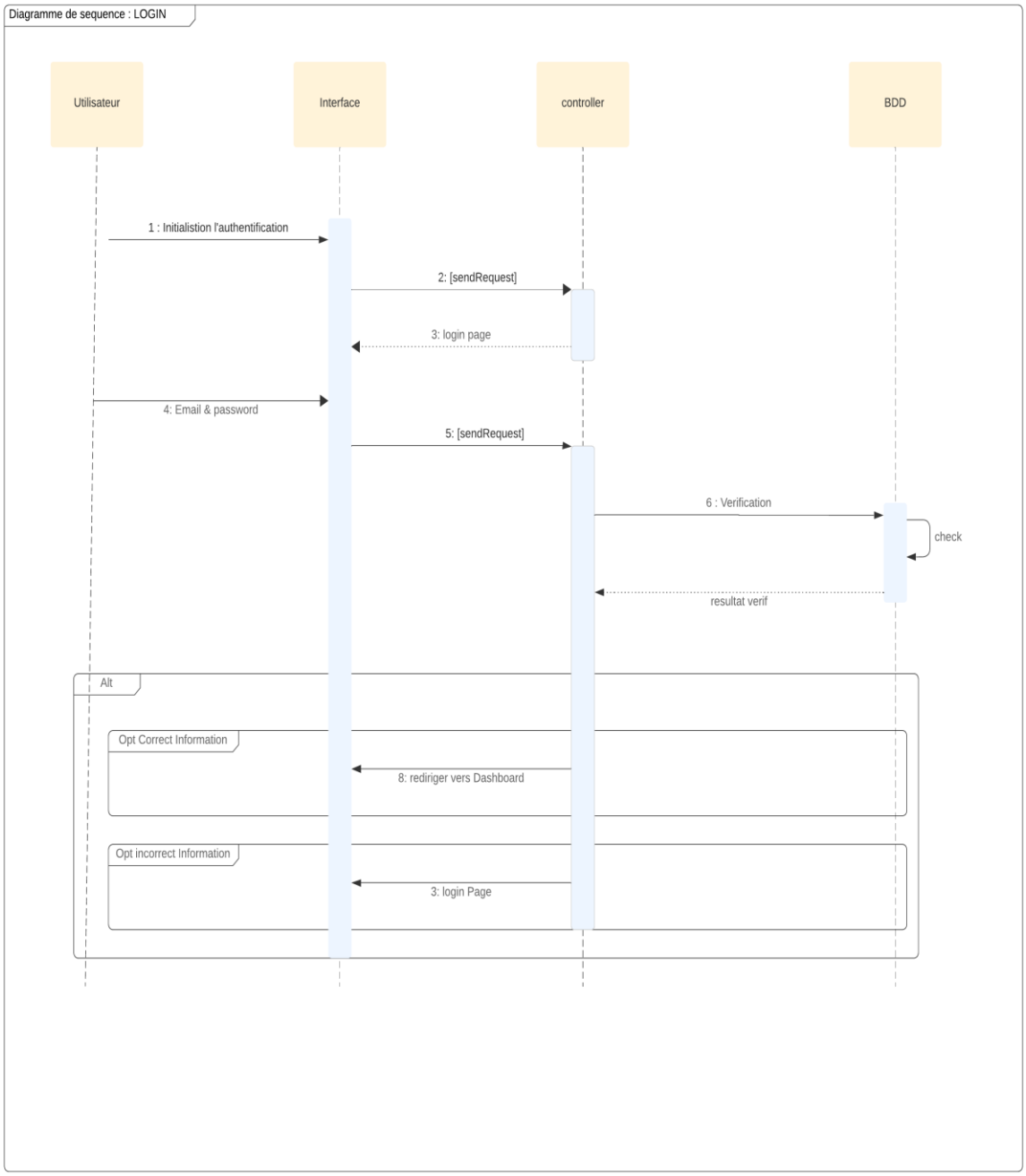


Figure 12: Diagramme de séquence : Login

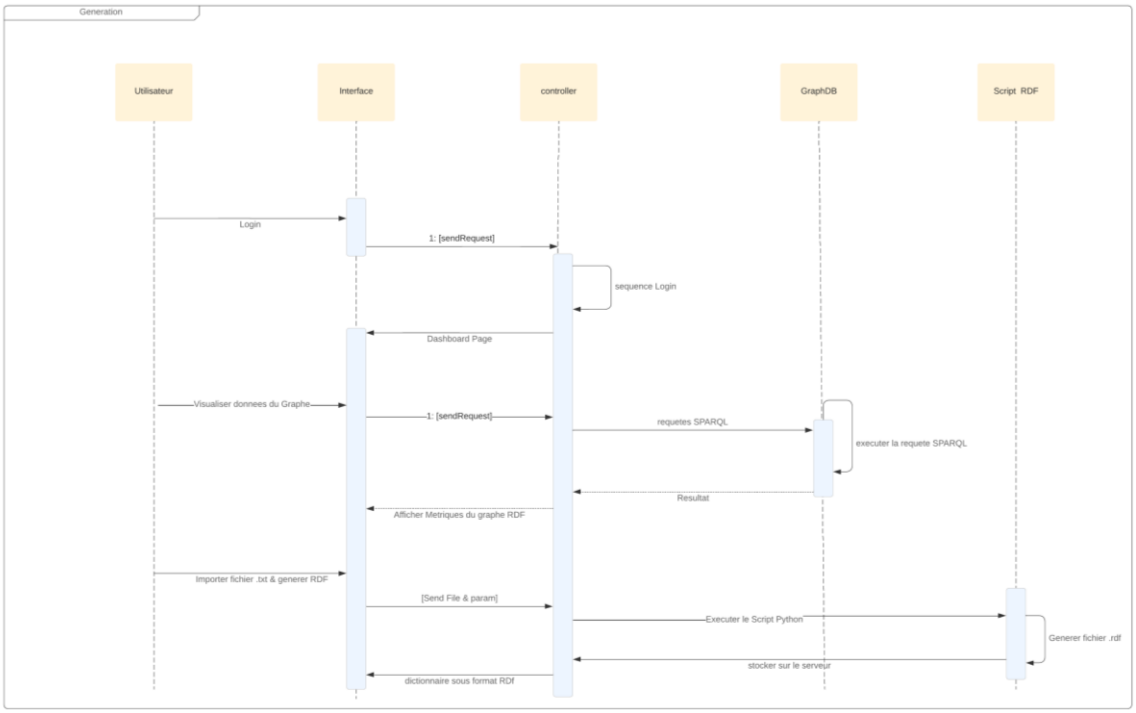
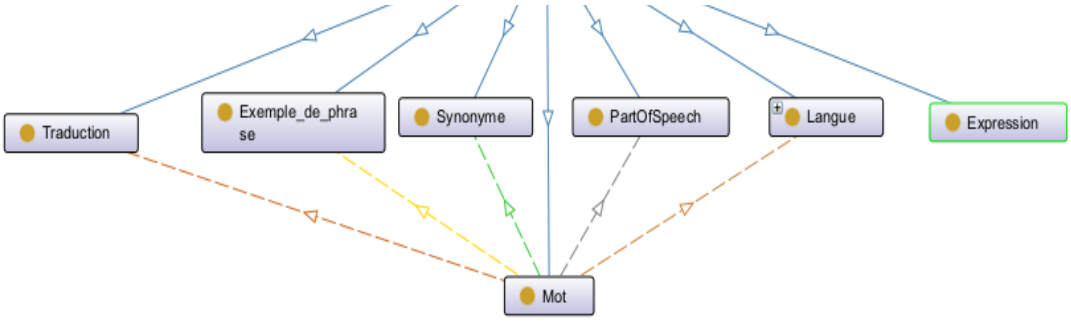


Figure 13: Diagramme de séquence pour la Génération

d. Représentation du Dictionnaire RDF avec protégé

Figure 14: visualisation du graphe avec protégé



2.3.4. Implémentation

2.3.4.1. Technologies :

- **Angular** 1: Un framework de développement web pour la création d'applications single-page. Il permet de créer des interfaces utilisateur riches et interactives.
- **Python** 2: Un langage de programmation de haut niveau largement utilisé pour le développement web, l'analyse de données, l'apprentissage automatique, et plus encore. Il est connu pour sa syntaxe claire et lisible.
- **Node.js** 3: Un environnement d'exécution JavaScript côté serveur qui permet de développer des applications réseau efficaces et évolutives.
- **Express.js4** : Un framework web minimaliste pour Node.js, utilisé pour construire des applications web et des API rapidement et facilement.
- **GraphDB5**: Une base de données graphique qui permet de stocker, organiser et gérer les données sous forme de graphes. Elle est particulièrement utile pour gérer les données RDF et SPARQL.
- **Protégé6** : Un éditeur de connaissances gratuit et open source qui permet de construire des ontologies RDF et OWL. Il est largement utilisé pour la visualisation et la gestion des données RDF.

2.3.4.2. Bibliothèques :

1. **RDFLib** : Une bibliothèque Python pour travailler avec RDF. Elle fournit des interfaces pour la création, la manipulation et la sérialisation des graphes RDF.
2. **jsonwebtoken** : Une bibliothèque Node.js pour travailler avec les JSON Web Tokens (JWT). Elle est utilisée pour l'authentification et la sécurisation des routes.
3. **multer** : Un middleware Node.js pour la gestion des téléchargements de fichiers. Il est utilisé pour gérer le téléchargement de fichiers sur la plateforme.

2.3.4.3. Captures d'écran des Interfaces

1. Pages d'accueil de la plateforme

Figure 15: Page d'accueil Page de chargement de fichier et d'extraction de données

2. Page de chargement de dictionnaire et de génération de RDF

Figure 16: chargement d'un dictionnaire sur la page de génération

3. Résultat d'une recherche

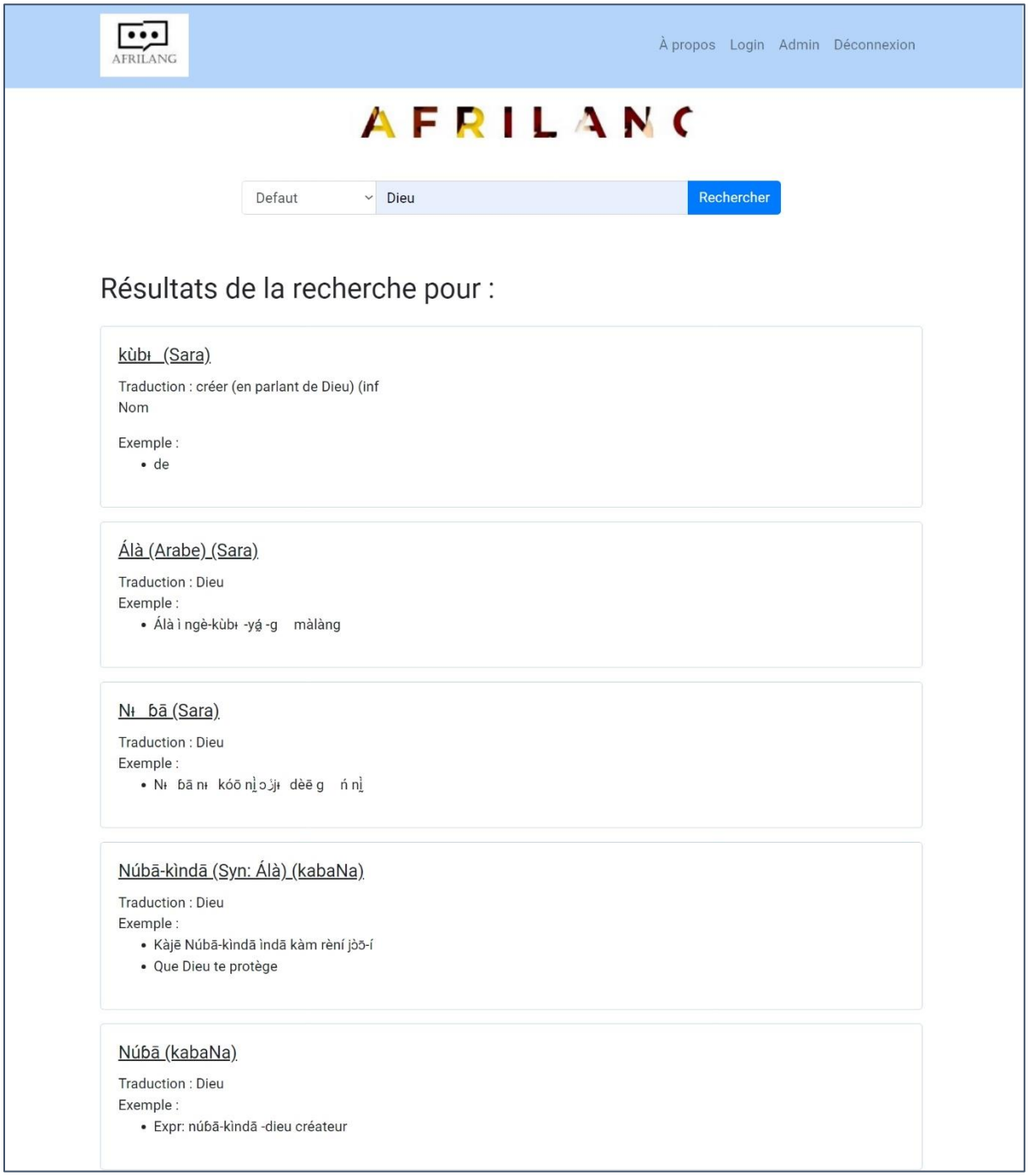


Figure 17: résultat de recherche pour le mot "Dieu"

4. Visualisation des données dans graphDB

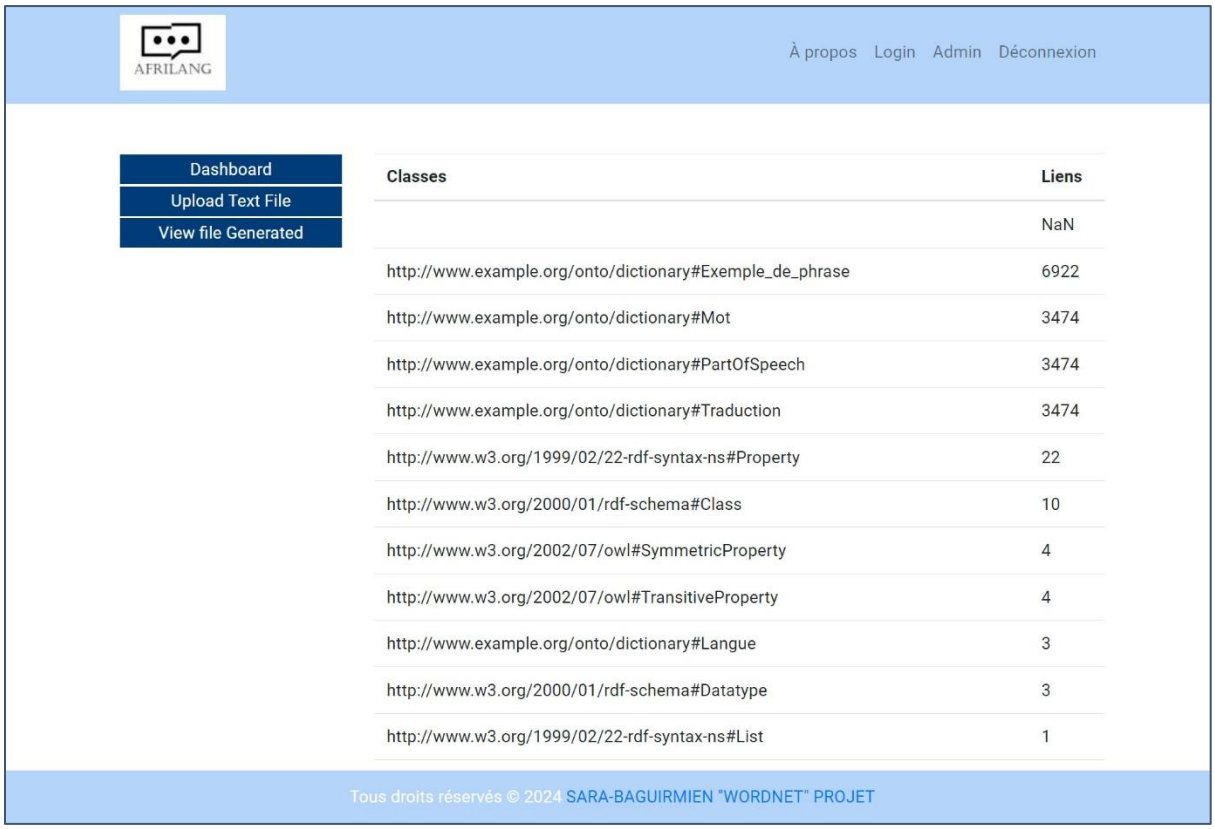


Figure 18: données récupérer sur graphDB

2.3.4.4. Visualisation des dictionnaires sur GraphDB

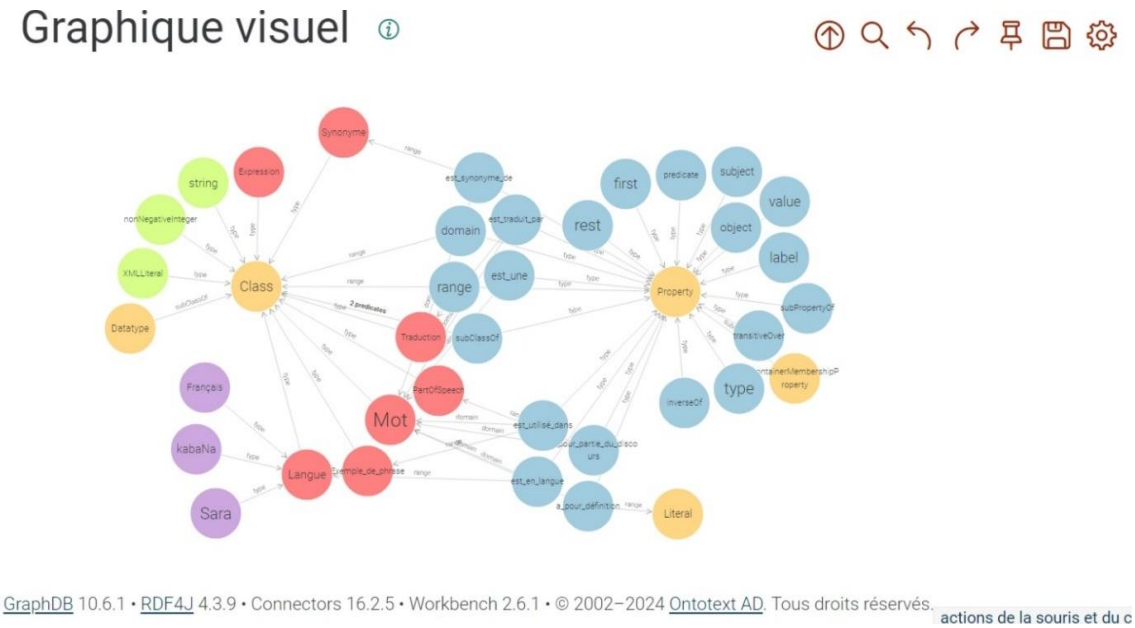


Figure 19: visualisation des dictionnaires sur graphDB

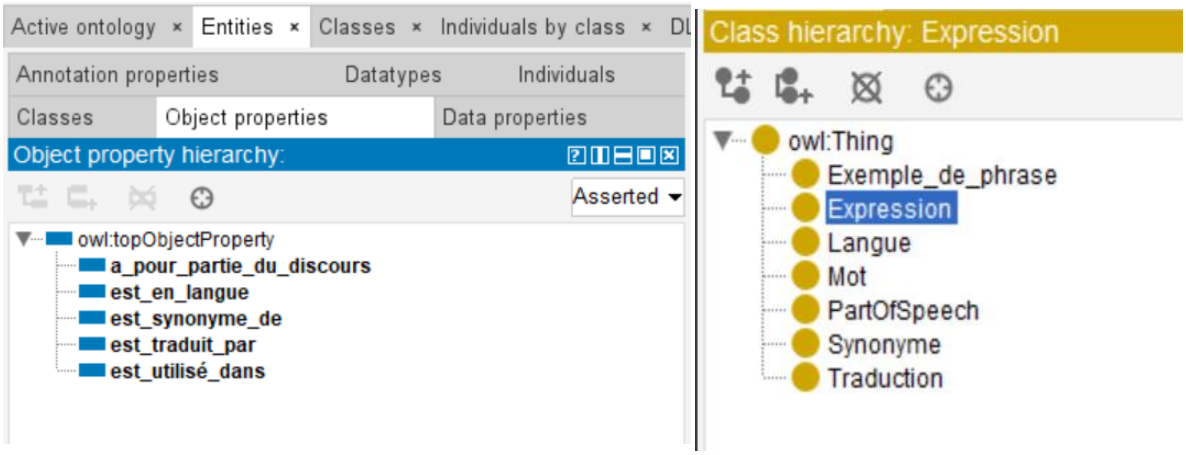


*Figure 20: classes et relation du graphe sur **Protégé2000***

2.4. Conclusion

Dans ce chapitre, nous avons présenté une approche pour la modélisation et l'implémentation de la construction des réseaux lexicaux sémantiques pour les langues peu dotées en utilisant le générateur RDF. Nous avons décrit la conception du réseau lexical sémantique, y compris l'identification des classes et des relations, le prétraitement des ressources dictionnairiques, la génération d'un graphe RDF vide et l'extraction et le peuplement de la représentation RDF. Nous avons également présenté la conception de la plateforme, y compris l'étude des besoins, l'architecture logicielle, la modélisation UML et l'implémentation utilisant des technologies telles que Angular, Python, Node.js, Express.js, GraphDB et Protégé. Les captures d'écran des interfaces et la visualisation des dictionnaires sur GraphDB ont également été présentées.

Conclusion générale et perspective

Les travaux dans le domaine de traitement des langues ont connu une avancée majeure depuis ses pionniers jusqu'à nos jours. En marge, les défis pour les langues peu dotées restent bien grands. Ainsi ce mémoire consiste donc à relever quelques-uns de ces défis en explorant la construction de réseau lexical sémantique pour les langues peu dotées grâce au web sémantique. Face à la rareté des ressources pour ces langues, nous avons formulé l'objectif de générer un réseau lexical sémantique multilingue à partir des dictionnaires bilingues. En outre, nous avons cherché à développer un outil qui facilite le processus de prétraitement, génération des fichiers RDF et permettre des recherches textuelles multilingues dans ces ressources sous leur nouvelle représentation. Partant de ces défis, nous avons proposé une méthode de traitement automatisé des ressources lexicales des langues peu dotées, et afin nous avons intégré cet outil sur la plateforme décrites dans le chapitre 2.

Les résultats premiers de notre travail visent à contribuer à la préservation et à la valorisation de la diversité linguistique et culturelle, mais surtout à offrir de nouvelles perspectives pour le développement de technologies linguistiques pour les langues peu dotées dont le groupe Sara-Baguirmien fait partie. Bien que notre recherche ait atteint ses objectifs principaux, il est important de reconnaître certaines limitations. Premièrement, la méthodologie utilisée repose sur l'exploitation d'un dictionnaire bilingue, ce qui peut ne pas couvrir l'ensemble du vocabulaire et des nuances linguistiques des langues peu dotées. Deuxièmement, la taille de l'échantillon, c'est-à-dire le nombre de langues peu dotées pour lesquelles nous avons construit des réseaux lexicaux sémantiques, pourrait être élargie pour une représentation plus complète. Enfin, des problèmes peuvent survenir lors de la génération des fichiers RDF et fournir un résultat biaisé. Ces limitations donnent des ouvertures pour des recherches futures afin d'améliorer et d'élargir notre travail.

Nous avons été confrontés à des défis, notamment la rareté des ressources et la complexité de la construction d'une ontologie adaptée aux spécificités linguistiques de ces langues. Cependant, ces défis ont renforcé notre détermination à poursuivre cette recherche. Nous espérons que notre travail trouvera un bon aboutissement et contribuer à la richesse de la diversité linguistique. Enfin, nous tenons à souligner que la langue est plus qu'un simple moyen de communication ; elle est le reflet de notre identité, de notre culture et de notre histoire. En préservant nos langues, nous préservons notre patrimoine culturel pour les générations futures.

Dans son état actuellement, notre système est conçu pour fonctionner dans un contexte linguistique spécifique qui est le groupe de langue Sara-Baguirmien. Cependant, il existe un potentiel considérable pour adapter la plateforme à d'autres langues et contextes culturels. Cela nécessiterait une recherche approfondie pour comprendre les nuances linguistiques, ainsi que le développement de nouvelles fonctionnalités pour répondre aux besoins spécifiques de ces contextes. La plateforme a également un potentiel significatif pour des applications dans le domaine du traitement automatique des langues (TAL). Par exemple, les données recueillies par la plateforme pourraient être utilisées pour entraîner des modèles d'apprentissage automatique pour la reconnaissance de la parole, la traduction automatique, ou la génération de texte.

Références

[1] R. Abdi and G. Lapalme, “21 ème Traitement Automatique des Langues Naturelles,” 2014.

[2] P. Vossen, “EUROWORDNET: A MULTILINGUAL DATABASE OF AUTONOMOUS AND LANGUAGE-SPECIFIC WORDNETS CONNECTED VIA AN INTER-LINGUALINDEX,” *International Journal of Lexicography*, vol. 17, no. 2, pp. 161–173, Jun. 2004, doi: 10.1093/ijl/17.2.161.

[3] S. Thoongsup *et al.*, “Thai WordNet Construction,” pp. 139–144, Sep. 2009, doi: 10.3115/1690299.1690319.

[4] D. Bernhard and C. Soria, “Traitement automatique des langues peu dotées,” *Revue Traitement Automatique des Langues*, vol. 59, pp. 7–14, Jan. 2018.

[5] Tchindjang, Bopda, Athanase, Ngamgne, Louise Angéline, and Mesmin, “Langues et identités culturelles en Afrique,” *Museum international, LX(60), 3 / 239, p. 37-50, illus., maps*, 2008.

[6] UNESCO, “Projet UNESCO: Atlas des langues en danger dans le monde,” 2011.

[7] M. Carpuat and D. Wu, “Improving Statistical Machine Translation Using Word Sense Disambiguation,” in *Conference on Empirical Methods in Natural Language Processing*, 2007. [Online]. Available: https://api.semanticscholar.org/CorpusID:135295

[8] M. Cuadros and G. Rigau, “Quality Assessment of Large Scale Knowledge Resources,” in *Conference on Empirical Methods in Natural Language Processing*, 2006. [Online]. Available: https://api.semanticscholar.org/CorpusID:841957

[9] S. E. Bosch and M. Griesel, “African Wordnet: facilitating language learning in African languages,” in *Global WordNet Conference*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:4417550

[10] N. Gala, “Approches multidisciplinaires pour l’étude du lexique et la création de ressources lexicales nouvelles. Mémoire d’Habilitation à Diriger des Recherches.,” 2015.

[11] G. Djiako, *Lexical ambiguity in machine translation and its impact on the evaluation of output by users*. 2019.

[12] D. Xu and S.-Q. Wen, *The Silk Road: language and population admixture and replacement*. 2017.

[13] T. R. Gruber, “A Translation Approach to Portable Ontology Specifications,” 1993.

[14] W. N. Borst, “Construction of Engineering Ontologies for Knowledge Sharing and Reuse,” *CTIT*, Jan. 1997.

[15] S. Staab and R. Studer, *Handbook on Ontologies*. 2009. doi: 10.1007/978-3-540-92763-3.

[16] G. Pierra, H. Dehainsala, Y. A. Ameur, and L. Bellatreche, “A paraître dans : Ingénierie des systèmes d’information, Hermès, 2005 Ingénierie des systèmes d’information. Volume X-n° X/2005, pages 1 à X Base de données à base ontologique : principe et mise en oeuvre,” 2005.

[17] A. Gomez-Perez, “Ontological Engineering: A state of the Art,” *Expert Update*, vol. 2, Jan. 1999.

[18] V. Psyché, O. Mendes, and J. Bourdeau, “Apport de l’ingénierie ontologique aux environnements de formation à distance,” *Sciences et Technologies de l’Information et de la Communication pour l’Éducation et la Formation*, vol. 10, Jun. 2003, doi: 10.3406/stice.2003.858.

[19] R. Mizoguchi, K. Kozaki, T. Sano, and Y. Kitamura, *Construction and Deployment of a Plant Ontology*. 2001. doi: 10.1007/3-540-39967-4_9.

[20] B. Bachimont, “Engagement sémantique et engagement ontologique : conception et réalisation d’ontologies en Ingénierie des connaissances,” 2000.

[21] M. Blázquez, M. Fernández-López, J. M. García-Pinar, and A. Gomez-Perez, “Building Ontologies at the Knowledge Level using the Ontology Design Environment,” May 1998.

[22] B. Benaissa, D. Bouchiha, A. Zouaoui, and N. Doumi, “Building Ontology from Texts,” *Procedia Comput Sci*, vol. 73, pp. 7–15, 2015, [Online]. Available: https://api.semanticscholar.org/CorpusID:61835959

[23] D. Tufis, D. Cristea, and S. Stamou, “BalkaNet : Aims , Methods , Results and Perspectives . A General Overview,” 2004. [Online]. Available: https://api.semanticscholar.org/CorpusID:10411845

[24] B. Sagot, “Représentation de l’information sémantique lexicale : le modèle *wordnet* et son application au français,” *Revue française de linguistique appliquée*, vol. Vol. XXII, no. 1, pp. 131–146, 2017, doi: 10.3917/rfla.221.0131.

[25] H. Ahmed Assowe, “Approche pour un premier étiqueteur morphosyntaxique d’une langue très peu dotée : le cas du somali,” *CEC-TAL’13*, Sep. 2013.