UNIVERSITE DE FRANCHE-COMTE
ECOLE DOCTORALE «LANGAGES, ESPACES, TEMPS, SOCIETES»

Thèse en vue de l'obtention du titre de docteur en

SCIENCES DU LANGAGE
SPÉCIALITÉ : TRAITEMENT AUTOMATIQUE DES LANGUES

INTERROGATION DES SOURCES DE DONNEES HETEROGENES : UNE APPROCHE POUR L'ANALYSE DES REQUETES.

Présentée et soutenue publiquement par

Ibrahim SOUMANA

Le 7 Juin 2014

Sous la direction de Mme le Professeur Sylviane CARDEY-GREENFIELD

Membres de jury:

Sylviane CARDEY-GREENFIELD, Directrice de recherche, Professeur, Université de Franche-Comté, France

Bohdan Krzysztof BOGACKI, Professeur, Université de Varsovie, Varsovie, Pologne, Rapporteur

Rodolfo DELMONTE, Professeur, Università Ca' Foscari, Venise, Italie, Rapporteur

Christophe ROCHE, Professeur, Université de Savoie, Chambéry, France

Peter GREENFIELD, HDR, Université de Franche-Comté, Besançon, France

Résumé

Le volume des données structurées produites devient de plus en plus considérable. Plusieurs aspects concourent à l'accroissement du volume de données structurées. Au niveau du Web, le Web de données (Linked Data) a permis l'interconnexion de plusieurs jeux de données disponibles créant un gigantesque hub de données. Certaines applications comme l'extraction d'informations produisent des données pour peupler des ontologies. Les capteurs et appareils (ordinateur, smartphone, tablette) connectés produisent de plus en plus de données. Les systèmes d'information d'entreprise sont également affectés. Accéder à une information précise devient de plus en plus difficile. En entreprise, des outils de recherche ont été mis au point pour réduire la charge de travail liée à la recherche d'informations, mais ces outils génèrent toujours des volumes importants.

Les interfaces en langage naturel issues du Traitement Automatique des Langues peuvent être mises à contribution pour permettre aux utilisateurs d'exprimer naturellement leurs besoins en informations sans se préoccuper des aspects techniques liés à l'interrogation des données structurées. Les interfaces en langage naturel permettent également d'avoir une réponse concise sans avoir besoin de fouiller d'avantage dans une liste de documents, mais actuellement, ces interfaces ne sont pas assez robustes pour être utilisées par le grand public ou pour répondre aux problèmes de l'hétérogénéité ou du volume des données. Nous nous intéressons à la robustesse de ces systèmes du point de vue de l'analyse de la question. La compréhension de la question de l'utilisateur est une étape importante pour retrouver la réponse.

Nous proposons trois niveaux d'interprétation pour l'analyse d'une question : domaine abstrait, domaine concret et la relation domaine abstrait/concret. Le domaine abstrait s'intéresse aux données qui sont indépendantes de la nature des jeux de données. Il s'agit principalement des données de mesures. L'interprétation s'appuie sur la logique propre à ces mesures. Le plus souvent cette logique a été bien décrite dans les autres disciplines, mais la manière dont elle se manifeste en langage naturel n'a pas fait l'objet d'une large investigation pour les interfaces en langage naturel basées sur des données structurées. Le domaine concret couvre le domaine métier de l'application. Il s'agit de bien interpréter la logique métier. Pour une base de données, il correspond au niveau applicatif (par opposition à la couche des données). La plupart des interfaces en langage naturel se focalisent principalement sur la couche des données. La relation domaine abstrait/concret s'intéresse aux interprétations qui chevauchent les deux domaines.

Du fait de l'importance de l'analyse linguistique, nous avons développé l'infrastructure pour mener cette analyse. L'essentiel des interfaces en langage naturel qui tentent de répondre aux

problématiques du Web de données (Linked Data) ont été développées jusqu'ici pour la langue anglaise et allemande. Notre interface tente d'abord de répondre à des questions en français.

Mots clés : Interface en langage naturel, Web de données, Linked Data, Système de Question-Réponse, Ontologie, Web Sémantique, Traitement Automatique des Langues.

Remerciements

Je tiens d’abord à remercier le Professeur Sylviane Cardey de m'avoir accueilli au sein du Centre de recherche en linguistique et traitement automatique des langues, Lucien Tesnière, de m'avoir proposé ce sujet passionnant et d'avoir mis à ma disposition les moyens nécessaires à son aboutissement. Son aide précieuse a été déterminante sur le plan scientifique. Je n'oublierai pas ces qualités humaines qui ont grandement contribué au plaisir que j'ai eu à réaliser ce travail.

Je tiens à exprimer toute ma reconnaissance à Monsieur Peter Greenfield pour sa disponibilité et ses conseils tout au long de ces années.

J’adresse mes remerciements à tous les membres du Centre Tesnière pour les échanges scientifiques et humains conviviaux.

Je tiens à remercier l’Ecole Doctorale, le Bureau Doctoral pour la résolution rapide des problèmes administratifs.

J’adresse un grand merci à toute ma famille en particulier à mes parents, à ma femme qui m’a apporté son soutien pendant ces années, à mes frères et sœurs, à mes amis et une pensée émue pour ma sœur disparue.

Enfin, je tiens à remercier toutes les personnes qui ont contribué, de près ou de loin, à la réalisation de ce travail de thèse.

SOMMAIRE

CHAPITRE 1 : INTRODUCTION 1

1.1 Contexte général 1

1.2 Motivation 3

1.3 Limites de la recherche 3

1.4 Contribution 4

CHAPITRE 2 : ETAT DE L'ART 6

2.1 Introduction 6

2.2 Systèmes utilisant un jeu de données restreintes 7

2.2.1 PRECISE 7
2.2.2 NaLIX 9
2.2.3 NLP Reduce 12
2.2.4 AQUALOG 13
2.2.5 QALLME 15
2.2.6 QEURIX 18
2.2.7 PANTO 18
2.2.8 SWAT 22
2.2.9 GINSENG 23
2.2.10 MMS (Méta Modèle Sémantique) 26
2.2.11 Catégorisation hiérarchique 28

2.3 Systèmes utilisant un jeu de données larges et hétérogènes 32

2.3.1 FREyA 32
2.3.2 CASIA 34

2.4 Conclusion 36

CHAPITRE 3 : LA DIMENSION DOMAINE 37

3.1 Volume et hétérogénéité des données 38

3.2 Couches d'une base de données 40

3.2.1 Expressivité des différents modèles de données 42

3.3 Type de questions traitées 47

3.4 Décomposition de domaine 47

3.5 Conclusion 58

CHAPITRE 4 : LA MESURE DU TEMPS 59

4.1 Introduction 59

4.2 TimeML un format émergeant 60

4.3 Formats ad-hoc 61

4.4 Caractéristique des données temporelles 61

4.5 Catégorisation des valeurs temporelles 65

4.6 Schémas d'annotation TimeML 71

4.6.1 La balise EVENT 71
4.6.2 La balise SIGNAL 72
4.6.3 La balise TIMEX3 72
4.6.4 Les liens 75

4.7 Les méthodes d'extraction 77

4.7.1 Approches symboliques 78
4.7.2 Approches statistiques 80

4.8 Analyse temporelle d'une question pour des données structurées 81

4.8.1 Calculs liés à la nature des données 81
4.8.2 Compostions et expressions temporelles en contexte 81
4.9 Notre approche de traitement du temps pour l'interrogation des données structurées 82
4.9.1 Définition de l'information temporelle 83
4.9.2 Etapes prises en compte pour le traitement de la temporalité 84
4.9.3 Caractéristique des expressions temporelles 85
4.9.4 Catégorisation des expressions temporelles 87
4.10 Conclusion 93
CHAPITRE 5 : EXPERIMENTATION **94**
5.1 Introduction 94
5.2 Corpus 94
5.3 Architecture du Système 94
5.2.1 Etiquetage et désambigüisation 95
5.2.2 Analyse syntaxique 99
5.2.3 Représentation intermédiaire et Génération de code 101
5.4 Conclusion 105
CONCLUSION GENERALE ET PERSPECTIVES **107**
Bibliographie : 109

Table des figures

FIGURE 1. MOTEUR DE RECHERCHE AU-DELA DES MOTS CLES 2
FIGURE 2. VARIATION DES PERFORMANCES DES INTERFACES EN LANGAGE NATUREL 4
FIGURE 3 PRECISE TRADUCTION EN SQL 8
FIGURE 4 NALIX : CLASSIFICATION DE REQUETES 11
FIGURE 5 ARCHITECTURE AQUALOG 14
FIGURE 6 ARCHITECTURE PANTO 19
FIGURE 7. PROCESSUS DE TRADUCTION EN SPARQL 20
FIGURE 8. STRUCTURE DE LA BASE DE DONNEES 21
FIGURE 9. PROCESSUS DE GENERATION DE REQUETE POUR SWAT 22
FIGURE 10. EXTRAIT DE RÉSEAU SÉMANTIQUE 27
FIGURE 11. DETERMINATION DES VALEURS ET VARIABLES DE MMS 27
FIGURE 12. ARCHITECTURE DU SYSTÈME 29
FIGURE 13. MOVIE & CINEMA ONTOLOGY (OU ET AL. 2008) 30
FIGURE 14. INVENTAIRE DES ELEMENTS DE LA QUESTION 31
FIGURE 15. FORMATION DE L'ARBRE 31
FIGURE 16. ARCHITECTURE DE FREYA 33
FIGURE 17. **ARCHITECTURE** DE CASIA 35
FIGURE 18. INTERCONNEXION DES DONNEES STRUCTUREES (OCTOBRE 2007) 38
FIGURE 19. INTERCONNEXION DES DONNEES STRUCTUREES (JUILLET 2009) 39
FIGURE 20. INTERCONNEXION DES DONNEES STRUCTUREES (JUILLET 2011) 40
FIGURE 21. DIFFERENTE PARTIE D'UNE BASE DE DONNEES 41
FIGURE 22. SYSTEME DE QUESTION-REPONSE BASE SUR DES DONNEES STRUCTUREES 41
FIGURE 23. HIÉRARCHIES DES CLASSES 49
FIGURE 25. REPARTITION DE TYPES DE BASE DANS UNE BASE DE DONNEES 51
FIGURE 26. GRANULARITE ET RELATIONS DES UNITES TEMPORELLES 92
FIGURE 27. ARCHITECTURE DU SYSTEME 95
FIGURE 28. VUE GÉNÉRALE DU LEXIQUE 96
FIGURE 29. REPRÉSENTATION D'UNE QUESTION 102
FIGURE 30. EXEMPLE DE REPONSE D'UN MOTEUR DE RECHERCHE MATHEMATIQUE 103
FIGURE 31. EXEMPLE DE REPONSE D'UN MOTEUR DE RECHERCHE DE MOTS-CLES 104
FIGURE 32. EXEMPLE DE REPONSE D'UNE BASE DE DONNEES 105

Liste des tableaux

TABLEAU 1. TOKEN NALIX 10
TABLEAU 2. MARKER NALIX 10
TABLEAU 3. CLASSIFICATION DES VARIABLES DES CARDINAUX 52
TABLEAU 4. EXEMPLE DE VOCABULAIRE D'INSTANCIATION DES CARDINAUX 54
TABLEAU 5. OPERATEURS DE GENERATION DES INTERVALLES 54
TABLEAU 6. OPÉRATEURS DE PROPORTION 55
TABLEAU 7. OPÉRATEURS DE COMPARAISON 56
TABLEAU 8. OPERATEURS DE LOGIQUE 57
TABLEAU 9. OPÉRATIONS ARITHMÉTIQUES 57
TABLEAU 10. DECLENCHEURS LEXICAUX DU TIMEML 63
TABLEAU 11. NON MARQUABLE 64
TABLEAU 12. VALEURS DE L'ATTRIBUT MODE 75
TABLEAU 13. RELATIONS TEMPORELLES D'ALLEN 76
TABLEAU 14. DONNEES DE BASES (OU DONNEES ELEMENTAIRES) 86
TABLEAU 15. OPÉRATEURS 87
TABLEAU 16. OPÉRATEURS D'ANCRAGE 89
TABLEAU 17. OPERATEURS DE LA DUREE 90
TABLEAU 18. OPÉRATEURS DE FRÉQUENCE 91
TABLEAU 19. INSTANCES DES UNITES TEMPORELLES 91

CHAPITRE 1 : INTRODUCTION

1.1 Contexte général

L'accroissement des données produites par les entreprises, les particuliers, les scientifiques et les acteurs publics, couplé au développement d'outils informatiques, offre de nouvelles perspectives. Le volume de données numériques augmente de manière exponentielle : 90 % de l'ensemble des données aujourd'hui disponibles ont été créées ces deux dernières années.

Cette augmentation s'explique principalement par les évolutions techniques et d'infrastructures. Entre 1990 et 2011, le pourcentage des utilisateurs d'internet et de téléphones mobiles au niveau mondial est passé respectivement de 0,05 % à 32,7 % et de 0,21 % à 85,5 %. Entre les troisièmes trimestres de 2011 et de 2012, les ventes mondiales de tablettes numériques et de smartphones ont pour leur part augmenté de 45,2 %. Ericsson prédit qu'il y aura 50 milliards d'objets connectés dans le monde d'ici à 2020, contre environ 12 milliards aujourd'hui. Le développement d'applications et de réseaux sociaux liés à ces nouvelles technologies explique aussi la création de données. L'avènement d'outils comme le cloud computing permet par ailleurs de stocker des données à moindre coût (Hamel et Marguerit, 2013).

Ces données peuvent être de type :

- structuré : ce sont des données qui ont une représentation logique. Les données peuvent être interrogées ou analysées par un langage informatique.
- non structuré : ce sont des données qui n'ont pas de format logique prédéfini, elles sont généralement du texte ou des données multimédia (audio, vidéo ou texte). Mais la tendance actuelle est de structurer de plus en plus ces données multimédia pour faciliter leur analyse.
- semi-structuré : ce sont des données qui ont un modèle logique flexible. Ces données peuvent également être interrogées par un langage informatique.

Au fur et à mesure que les données s'accumulent, leur analyse en vue de tirer de la valeur s'accentue. Pour (Spivack, 2009) voire Figure 1, la croissance du volume de données sur le Web nécessite de nouveaux paradigmes de recherche pour maintenir la productivité de la recherche d'informations. Le Web évolue progressivement de la recherche en mots clés vers une recherche en langage naturel.

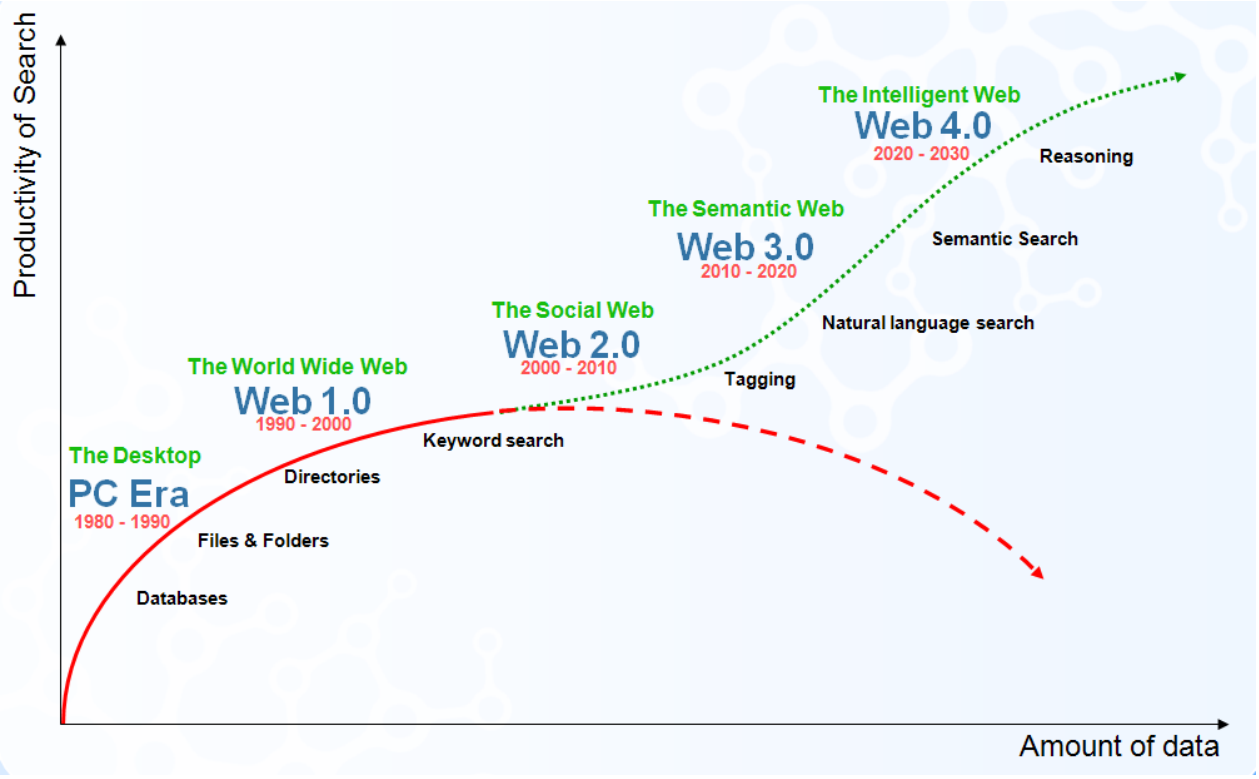


Figure 1. Moteur de recherche au-delà des mots clés

Notre travail porte sur l'interrogation des données en langage naturel dans le cadre de Système de Question-Réponse plus précisément sur l'analyse de la question de l'utilisateur. Les Systèmes de Question-Réponse peuvent être classés en trois catégories selon la nature des données traitées :

- système basé sur des données non structurées : la base de données utilisée pour répondre aux questions est du texte brut.
- système basé sur des données structurées : la base de données d'où la réponse est extraite est structurée. Elle dispose d'un langage informatique comme SQL[1], SPARQL[2] pour manipuler les données. Le Système de Question-Réponse traduit la question dans le langage formel. Puis la question traduite est utilisée par la base de données pour extraire la réponse.
- système hybride : ce type de système utilise à la fois des données structurées et non structurées.

Dans notre thèse nous nous basons sur les systèmes utilisant des données structurées ou semi-structurées ayant un langage d'interrogation informatique. Ces systèmes sont également appelés des interfaces en langage naturel pour des bases de données.

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

[2] http://fr.wikipedia.org/wiki/SPARQL

1.2 Motivation

La production des données structurées provient de plusieurs technologies :

- Le Web de données (Linked Data) : le Web Sémantique a développé des standards qui permettent de structurer, puis interconnecter les bases de données formant un nuage de données. Il contient plus de 31 634 213 770 faits[3] en 2011.
- Le Web profond (Deep Web) : le Web profond ou Web invisible est constitué des bases de données des entreprises, particuliers, administrations qui ne sont pas indexées par les moteurs de recherche. Ces données ne sont pas facilement accessibles aux moteurs de recherche. Le Web profond contient 400 à 550 fois plus de ressources que le Web indexé (Bergman, 2001).
- Les objets connectés : les objets connectés sont en grande partie responsable de la production de données.

Ces données constituent une mine d'informations dont l'intérêt est croissant. Ces données ne sont pas accessibles à la plupart des utilisateurs du fait qu'ils ne sont pas familiers des langages informatiques qui permettent de manipuler ces données. Une interface en langage naturel permet un accès convivial à ces données.

1.3 Limites de la recherche

Plusieurs problèmes freinent le déploiement des interfaces en langage naturel. Ces problèmes se situent à deux niveaux principalement (Damljanovic, 2011) Figure 2:

- au niveau de la question : une question complexe peut être mal interprétée. La complexité d'une question n'est pas encore définie dans la littérature.
- au niveau du volume et de la variété des domaines : même si la question est bien comprise, il n'est pas évident que la réponse puisse être repérée dans une base de données volumineuse constituée de plusieurs domaines.

Plus les domaines sont nombreux et la question est complexe, plus la performance des systèmes diminue. Nous nous intéressons plus au premier point qui est la compréhension des questions. L'analyse de la question est une étape importante pour extraire la réponse. Le bouleversement de l'univers de données change aussi la manière de concevoir les interfaces en langage naturel. Les méthodes utilisées pour des interfaces à un seul jeu de données et un nombre restreint d'utilisateurs montrent leurs limites dans un environnement comme le Web. A l'échelle du Web, une question peut être posée de plusieurs manières contrairement à l'échelle d'une entreprise où les utilisateurs sont restreints et ont une connaissance des données qu'ils manipulent. Nous avons donc orienté le travail

[3] http://lod-cloud.net/state/

vers des approches susceptibles de concilier à la fois la diversité des domaines et la complexité linguistique sans entrer dans les détails de leurs résolutions. Le travail ne vise donc pas la résolution des problèmes qui interviennent en conciliant la diversité des domaines et la complexité linguistique.

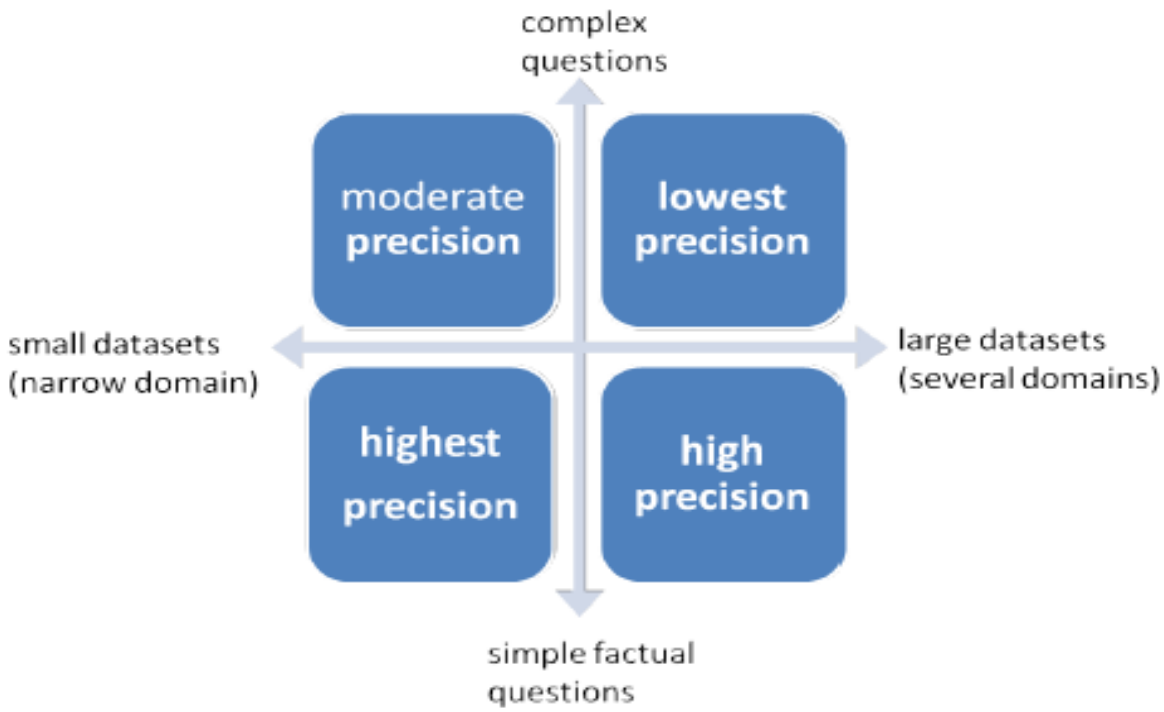


Figure 2. Variation des performances des interfaces en langage naturel

1.4 Contribution

La contribution de la thèse est une approche méthodologique qui permet de concilier la diversité des domaines et la complexité linguistique (variation). Cette approche décompose le domaine en 3 niveaux :

- domaine abstrait : il s'agit ici d'étudier le comportement de la donnée hors du contexte d'une base de données. Selon la nature de la donnée, quelles sont les opérations qui lui sont applicables. La plupart des méthodes se concentrent juste sur l'identification du vocabulaire de la base de données et du langage formel (comme SQL ou SPARQL). L'identification de ces deux vocabulaires ne suffit pas. La prise en compte de la manière dont ces données peuvent être manipulées permet d'accroitre la robustesse des systèmes.
- domaine concret : c'est le domaine métier.
- domaine abstrait/concret : le système doit être à mesure d'interpréter la logique métier et le domaine abstrait.

La thèse est organisée comme suit : le chapitre 2 présente l'état de l'art. Le chapitre 3 s'intéresse à la décomposition du domaine en domaines abstrait et concret. Le chapitre 4 est un exemple d'étude d'un domaine abstrait. Il s'agit du temps. Le chapitre 5 traite de l'expérimentation. Il présente le travail effectué et des exemples.

CHAPITRE 2 : ETAT DE L'ART

2.1 Introduction

Le premier système à interface en langage naturel date de 1972 (Woods et al. 1972). C'est une base de données de minéraux recueillis sur la lune. Plusieurs systèmes ont été développés par la suite, basés principalement sur le langage SQL (Structured Query Langage). Dans les années 80, TEAM (Grosz et al., 1987) a été développé et a même été commercialisé.

Dans les années 90, la recherche universitaire sur les interfaces en langage naturel n'était plus à la mode comme elle l'était dans les années 80 (Androutsopoulos et al., 1995).

La décennie 2000 a vu un regain d'intérêt pour les interfaces en langage naturel. La recherche a porté principalement sur des bases de données limitées et locales. Le problème fréquent signalé concerne la portabilité : la possibilité de déployer rapidement la même interface pour une autre base de données.

La masse de données structurées générées par le Web, a suscité de nouveaux problèmes à partir de la décennie 2010 notamment un besoin d'accéder aux données et la capacité des systèmes à s'adapter au volume des données. Il y a également un besoin croissant d'élargir l'accès à ces données au grand public. A ce niveau se pose le problème de robustesse vis-à-vis de la langue et de la diversité des jeux de données. Au delà des données (Soumana et al., 2012a), les interfaces en langage naturel peuvent jouer un rôle intéressant dans le domaine des objets connectés notamment lorsqu'il y aura un standard largement accepté.

Différentes approches sont utilisées par les interfaces en langage naturel. Elles ne sont pas nécessairement exclusives. C'est possible donc de trouver plusieurs approches dans un même système. Les différentes approches sont :

- basées sur le schéma : il s'agit de faire correspondre les mots de la question au schéma de la base de données. Il n'y a pas nécessairement une analyse des relations entre les mots de la question. Des exemples de systèmes de ce type sont : (Popescu et al. 2004 ; El Abed, 2001; Yahya et al., 2012),

- basées sur une langue contrôlée: ce type de système se caractérise par une restriction de la syntaxe des requêtes en langage naturel que l'utilisateur est autorisé à introduire dans le système. SWAT (Bernstein et al. 2005a) et GINSENG (Bernstein et al. 2006) sont des systèmes à langue contrôlée.

- basées sur des patrons : Ces systèmes identifient une expression ou une simple phrase à laquelle ils font correspondance une requête en langage formel. QUERIX (Kaufmann et al. 2006) et QALLME (OU et al. 2008) peuvent être classés comme systèmes basés sur des patrons.

- basées sur les triplets : ces systèmes concilient une analyse de la question étroitement liée au schéma (triplet de l'ontologie). En général au moins un analyseur syntaxique de surface est utilisé pour analyser la relation entre les mots. Ces relations sont généralement des relations de dépendance. Mais il ne s'agit pas vraiment d'une analyse linguistique complète. (Soumana, 2010), PANTO (Wang et al. 2007), CASIA (He et al., 2013), ORAKEL (Cimiano et al., 2008), TREO (Freitas et al., 2011) sont des exemples.

  Certains systèmes comme ORAKEL, C-PHRASE (Minock, 2010) utilisent une approche compositionnelle.

- Interaction avec l'utilisateur : l'utilisation est sollicitée à travers un dialogue pour lever certaines ambigüités. Un exemple de cette approche est (Damljanovic, 2011).

Dans les paragraphes qui suivent nous nous limitons aux données structurées. D'autres systèmes comme IBM Watson (David et al., 2010) utilisent principalement des données non structurées (Kalyanpur et al., 2012 ; Brown et al., 2013).

En pratique, un système peut combiner plusieurs de ces approches. La gestion de la complexité de la question et la gestion du volume de données sont des problèmes actuels pour les interfaces en langage naturel. La complexité de la question est difficile à définir alors que le volume de données et son hétérogénéité sont quantifiables. De ce fait les paragraphes suivants présentent les systèmes selon le volume de jeux de données. Les performances des systèmes sont données à titre indicatif (du fait de la difficulté de faire une comparaison objective).

2.2 Systèmes utilisant un jeu de données restreintes

2.2.1 PRECISE

PRECISE est un système qui traduit une phrase en langage naturel en langage formel SQL. Il a été développé par (Popescu et al. 2004) à l'Université de Washington. Le système PRECISE introduit la notion de phrase sémantiquement traitable (semantically tractable sentences). Ce concept suppose qu'une requête en langage naturel n'a de sens que si elle peut avoir une réponse dans la base de données. L'essentiel de la méthode consiste à analyser si une réponse peut être obtenue de la base de

données avant de générer la requête en SQL. Les terminologies utilisées par PRECISE sont relation, attribut, valeur (relation, attribute, value). Une relation est une entité de la base de données. La relation peut être également une relation entre tables de la base de données. L'attribut est une propriété d'une entité. Dans une base de données l'attribut correspond à une colonne de la table. La valeur correspond à une valeur d'un attribut d'une relation. Le lexique (lexicon) permet de faire la correspondance entre les termes de la phrase de l'utilisateur et les éléments de la base de données.

Le principe de question sémantiquement traitable est effectué comme suit (Popescu et al. 2004):

a) Correspondance termes utilisateur (Sentence Tokens T) – éléments de la base de données (Database Elements E). Il existe une correspondance un à un entre les termes de l'utilisateur T et les éléments de la base de donnée E.

b) Correspondance Attribut (Attribute Token)-Valeur (Value Token). Chaque attribut est lié à une valeur unique.

c) Attributs implicites (Implicite Attributes)

Certains éléments de la base de données peuvent ne pas avoir de correspondants dans les termes de l'utilisateur. Ces éléments sont appelées attributs implicites.

d) Correspondance Relation (Relation Token) - Attribut/Valeur (Attribute/Value Token). Chaque relation correspond à une valeur ou un attribut.

Un exemple (Popescu et al. 2004) pour la question «What are the HP jobs on Unix System?» est traité comme suit:

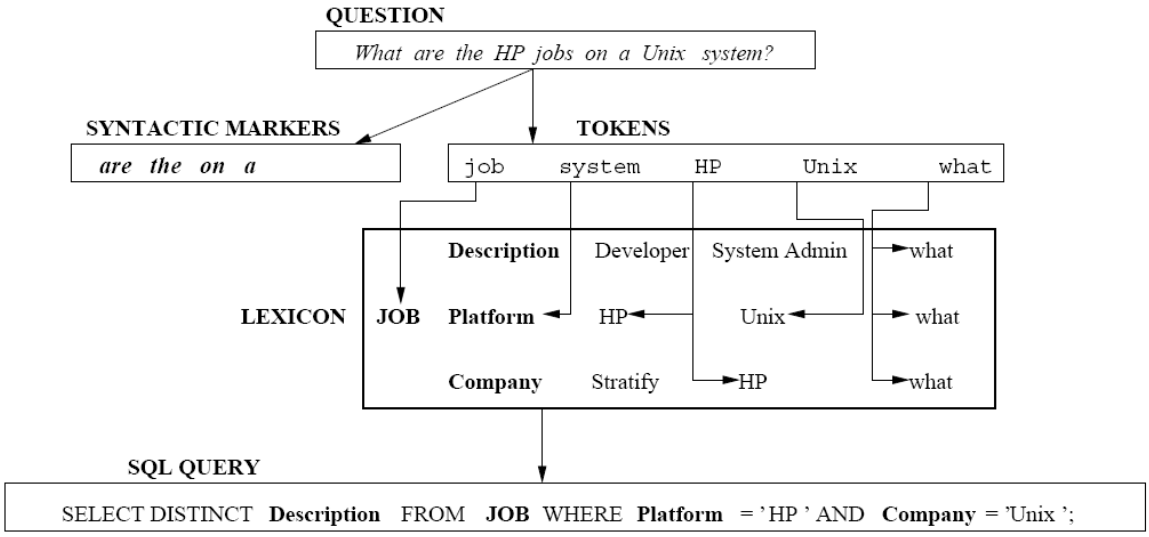


Figure 3 PRECISE traduction en SQL

La table référencée dans la base de données est JOB avec les attributs Description, Platform, Company. La correspondance est assurée par maxflow algorithme (Uno, 1997) qui traite également les ambigüités comme celle de la valeur HP dans cet exemple qui peut être une valeur pour l'attribut Platform et pour l'attribut Company. La restriction de la génération des requêtes aux phrases sémantiquement traitables assure à PRECISION une haute précision mais le rappel est faible (Popescu et al. 2004).

(Popescu et al. 2004) reporte une précision de 97% dans le domaine des restaurants, 88% dans le domaine d'offre d'emploi (Jobs) et 77,5% dans le domaine de la géographie. Le nombre de questions de test n'est cependant pas précisé.

2.2.2 NaLIX

NaLIX (Natural Language Interface for an XML Database) (Li et al 2004) est une interface interactive en langage naturel pour des bases de données XML, développée à l'Université de Michigan. L'architecture de NaLIX est constituée de deux parties. La première partie traduit les phrases en langage formel, le second interagit avec l'utilisateur pour le guider pour introduire une phrase susceptible d'avoir une réponse. La traduction des phrases se réalise en trois étapes (Li et al 2005):

- classification de l'analyse syntaxique (Parse Tree Classification),
- validation de l'arbre syntaxique,
- traduction en requête Schema Free XQuery.

Schema Free XQuery a l'avantage de s'affranchir des contraintes liées à la structure de la base de données. Cette indépendance de la structure sous jacente de la base de données est assurée en recherchant le nœud parent le plus significatif. Ce nœud correspond au nœud qui contient le plus d'attributs et classes contenus dans la question en langage naturel: meaningful lowest common ancestor structure (MLCAS).

NaLIX utilise l'analyseur syntaxique MINIPAR (Lin, 1998). La classification de l'analyse syntaxique identifie les mots ou expressions qui correspondent aux différentes parties d'une requête XQuery. Ces mots ou expressions sont appelés token. Les autres termes qui ne correspondent pas à un token sont classés comme marker.

Tableau 1. Token NaLIX

Token	Query Component	Description	Example
Command Token(CMT)	Return clause	Top or wh-phrase (WHNP)	Return, what is
Order by Token	Order by clause	Enum set of phrases	Sorted by
Function Token(FT)	Function	Enum set of adjectives and noun-phrase	maximun
Operator Token (OT)	Operator	Enum set of preposition phrase	More than
Value Token(VT)	Value	Noun(NN), Noun Phrase(NP), Proper Noun(NNP), Number(CD)	Traffic
Name Token (NT)	Basic variables	A non VT noun or Noun Phrase(NP)	director
Negation (NEG)	Function not	Adjective « not »	not
Quantifier Token	Quantifier	Enum set of adjectives serving as determiner	every

Tableau 2. Marker NaLIX

Type of Marker	Semantic Contribution	Description	Example
Connection Marker (CM)	Connect tow related tokens	A preposition from an enumerated set, or non-token main verb	Of, directed
Modifier Marker(MM)	Distinguish two NTs	Pronouns	Many, popular
Pronoun Marker (PM)	None due to parser's limitation	Pronoum	This, that, him
General Marker(GM)	None	Auxiliary verbs, articles	A, an , the

La requête "Return every director, where the number of movies directed by the director is the same as the number of movies directed by Ron Howard." est traitée come suit:

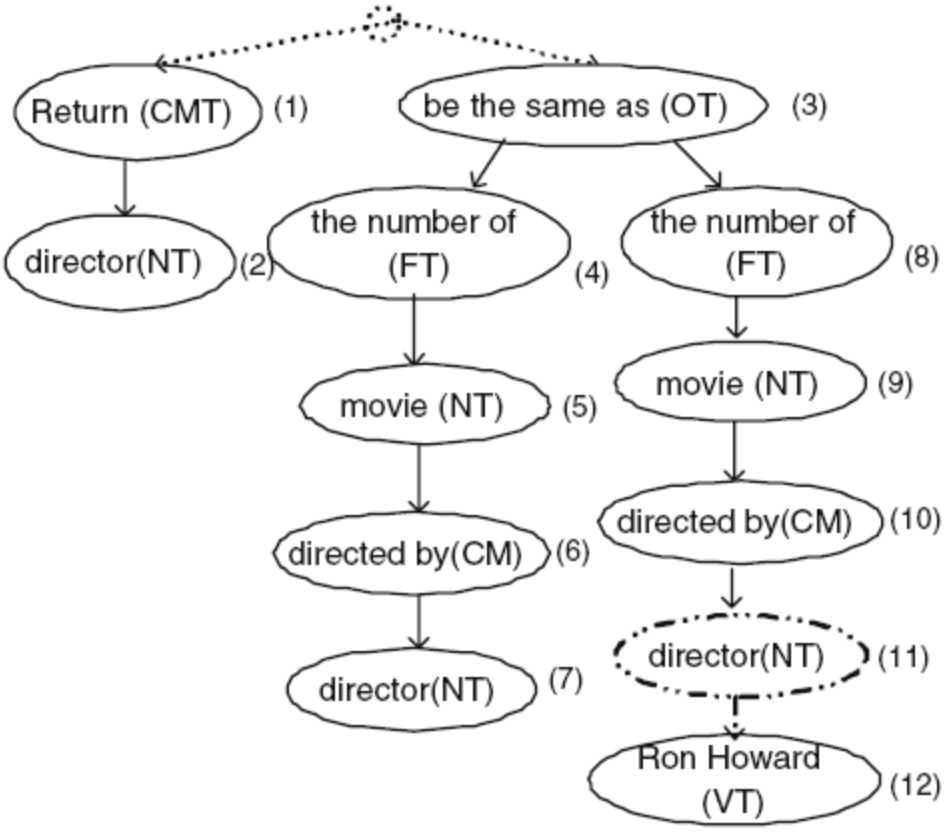


Figure 4 NaLIX : classification de requêtes

La validation de l'arbre syntaxique ainsi obtenu consiste à vérifier si l'arbre syntaxique peut correspondre à une requête XQuery et si les valeurs contenues dans l'arbre existent dans la base de données.

Si l'arbre syntaxique est valide, la traduction est autorisée. Cette traduction consiste à regrouper dans une même fonction MQF (Meaningful Query Focus) les tokens liés (identifiés lors de l'analyse syntaxique). La fonction MQF retrouve automatiquement la relation entre ces tokens dans la base de données (Li et al 2004).

De ce fait, Nalix dépend de XQuery. Si la question en langage naturel n'est pas correctement traduite en XQuery, une réponse n'est pas possible. Ceci entraine que Nalix peut répondre à une question ayant un sens donné avec une certaine syntaxe et ne peut pas répondre à une autre question ayant le même sens mais avec une syntaxe différente. La réponse dépend donc de la syntaxe de la phrase plutôt que du sens.

(Li et al 2004) reporte une précision de 95,1% et un rappel de 97,6 sur 112 questions syntaxiquement correctes d'après l'analyseur syntaxique Minipar (Lin, 1998).

2.2.3 NLP Reduce

NLP Reduce est une interface en langage naturel pour le langage SPARQL (Kaufmann et al. 2007). NLP-Reduce opère par mots clés. L'architecture du système est composée de 5 parties: l'interface utilisateur (user interface), le lexique (lexicon), un composant de prétraitement des requêtes (input query processor), un générateur de requêtes SPARQL (SPARQL query generator), et la couche d'accès à l'ontologie (ontology access layer).

Le lexique est généré automatiquement à partir de l'ontologie. L'algorithme Porter Stemmer (Martin et al, 1980) est utilisé pour améliorer le rappel. Le composant de prétraitement de la requête élimine les mots vides et les marques de ponctuation. Le résultat du prétraitement (les racines des mots) est ainsi passé au générateur de requête SPARQL. Ce dernier est le composant principal du système. Il essaie d'établir une correspondance entre les mots du lexique (lexicon) et les termes de la requête de l'utilisateur. Considérons l'exemple suivant: *Where is a restaurant in San Francisco that serves good French food?* Le générateur de requête suit les étapes suivantes (Kaufmann et al. 2007):

a) Le générateur de requêtes recherche d'abord les triplets dans le lexique (lexicon) dans lesquels au moins un des mots de la requête existe. Pour cet exemple les triplets <isIn>, <isInCity>, <isInCounty> and <isInRegion> seront retrouvés comme ils contiennent le mot is and in. Le générateur range les relations trouvées selon un critère qui dépend des mots contenus dans la requête et les triplets. Les triplets ayant une couverture exacte de la requête sont favorisés. Par exemple <isIn> est préférable à <isInCity>.

b) Le générateur cherche ensuite dans le lexique, les propriétés qui peuvent joindre les triplets trouvés à l'étape 1 par le reste des mots de la requête utilisateur. Ici il s'agit de "where," "restaurant," "San Francisco," "serve," "good," "French," and "food". Le mot where indique la localisation, les triplets contenant le prédicat location sont retrouvés. Les mots serve and food récupèrent les triplets ayant la propriété <foodtype>. Si un mot récupère plusieurs triplets, les triplets avec le score le plus élevé est préféré. L'ensemble des triplets de l'étape 2 sont ensuite combinés avec les triplets de l'étape 1 où la joncture est assurée par la classe restaurant.

c) Le générateur cherche maintenant les propriétés de type datatype qui correspond aux mots restant de la requête. Pour les mots "San Francisco," "good," and "French," les triplets [<Restaurant> <isIn> 'sanFrancisco'], [<Restaurant> <rating> 'good'], and [<Restaurant> <foodType> 'french'], sont récupérés. Le système ordonne ensuite le résultat.

d) Une fois tous les mots de la requête traités, le générateur établit la requête SPARQL avec les triplets ayant le score le plus élevé de l'étape 1 à 3. Il supprime les duplications. La requête est générée comme suit (Kaufmann et al. 2007) sans le préfixe :

```
SELECT distinct * WHERE {

?Restaurant <#location> ?Location.

?Restaurant <#rating> 'good'.

?Restaurant <#foodType> 'french'.

?Restaurant <#isIn> ?City.

?City <#label> 'sanFrancisco'.

?Restaurant <#type> <#Restaurant>.

?City <#type> <#City>.}
```

NLP reduce utilise une approche bag of words pour traiter une question. Il n'existe pas de traitement sophistiqué de la langue à l'exception de l'utilisation de Wordnet et de l'algorithme Porter Stemmer. La dépendance entre la relation et les mots de la question est seulement obtenue à partir de la base de données.

(Kaufmann et al. 2007) reporte une précision de 70,7% et un rappel de 76,4% pour 308 questions du domaine géographique.

2.2.4 AQUALOG

AQUALOG est un système de Questions-Réponse basé sur une ontologie au format RDF (Lopez et Motta, 2004 ; Lopez et al., 2005) développé par Knowlegde Media Instutite[4] . La démarche d'Aqualog est basée sur le fait que plusieurs questions peuvent être représentées sous forme de triplets RDF <sujet predicat objet>. L'architecture du système est composée principalement des composants linguistiques qui transforment une question en langage naturel en query-triple, d'un composant RSS (Relation Similarity Service) qui transforme le résultat du query-triple en onto-triple. La représentation query-triple est un format proche des triplets RDF obtenu essentiellement grâce à

[4] http://kmi.open.ac.uk/technologies/aqualog/

une analyse linguistique de la question. L'onto-triple est une représentation équivalente aux schémas des bases de données RDF.

L'onto-triple est utilisé pour retrouver la réponse dans la base de données. Aquelog ne convertit pas le langage naturel en langage formel comme SQL, SPARQL ou SeRQL.

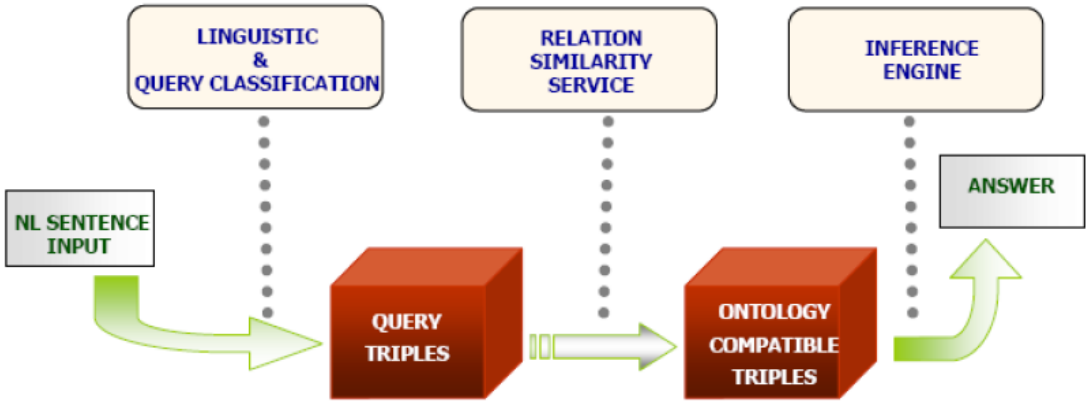


Figure 5 Architecture Aqualog

Le composant linguistique utilise l'infrastructure GATE (Cunningham et al. 2002) pour faire l'analyse syntaxique de la question en langage naturel. GATE produit une série d'annotations syntaxiques. Des extensions ont été ajoutées à GATE pour récupérer les entités, les relations, les indicateurs de questions comme (what, who, when…) et les types de questions. Ces tâches sont réalisées en utilisant JAPE[5] grammars qui permet à travers des expressions régulières l'analyse linguistique. Cette dernière est indépendante du domaine de l'ontologie. C'est une étape fondamentale pour la portabilité du système. Par exemple la question «who is the secretary in KMi?». Le composant linguistique produit le query-triple suivant <person/organization, secretary, kmi>, (Lopez et al. 2005).

Le composant RSS permet de donner une signification au query-triple selon la sémantique de l'ontologie. Les ressources utilisées par le composant RSS sont des métriques de similarité pour les chaines de caractères (string similarity matching), Wordnet, le lexique du domaine à travers un apprentissage automatique. Les ambigüités sont gérées d'abord en utilisant les ressources du domaine. Si l'ambiguïté n'a pu être levée, le module RSS interagit avec l'utilisateur.

[5] http://gate.ac.uk/sale/thakker-jape-tutorial/GATE%20JAPE%20manual.pdf

Par exemple pour le triplet <*person/organization, secretary, kmi*> le module RSS cherche d'abord à identifier l'objet *kmi*. Dans l'ontologie, *kmi* correspond à la classe *«research institute»* dénommée *"Knowledge-Media-Institute"*. Le module RSS cherche les relations entre la classe *«research institute»* ou ses superclasses et la classe personne/organisation ou une sous classe de personne/organisation. Même si une relation identique n'est pas trouvée, un triplet équivalent est accepté. Ceci est fréquent car la structure de l'ontologie peut ne pas correspondre à la relation qui lit les éléments du triplet. Dans cet exemple le triplet <secretary,works-for,kmi> est identifié. Bien que *secretary* soit une sous-classe de personne au lieu d'une relation, ce triplet est maintenu.

Cependant il est possible d'avoir plusieurs triplets candidats. Dans ce cas, les ressources lexicales, Wordnet sont utilisées pour déterminer la relation vraisemblable. Au cas où ces ressources ne peuvent pas identifier la relation, le module RSS propose une liste de candidats à l'utilisateur.

Aqualog ne supporte pas les expressions temporales. (Lopez et al. 2007) argumentent que ces expressions dépendent du domaine. Du fait que les composants linguistiques ne traitent pas les quantificateurs (many, all, ou nombre par exemple), Aqualog ne permet pas de traiter des questions de type *« how much »*. Aqualog ne semble pas aussi traiter des questions complexes car selon (Lopez et al. 2007), il ne peut traiter que des questions ayant au plus 2 triplets. Pour que le traitement soit possible, il faut que les entités qui interviennent dans la question soient directement liées dans l'ontologie (forment un triplet de l'ontologie). Aqualog ne traite pas non plus les noms ou adjectifs composés. (Cunningham et al. 2002) reporte une précision de 48,68% pour 76 questions.

2.2.5 QALLME

QALLME (Question Answering Learning technologies in a multiLingual and Multimodal Environment) est un projet ayant pour but de développer une infrastructure multilingue. L'infrastructure QALLME est adaptable à divers domaines (OU et al. 2008). Le projet a développé un System de Questions Réponse dans le domaine du tourisme. Le système est basé sur des patrons (*pattern generation*) et l'implication textuelle (*textual entailment*). Pour une question donnée de l'utilisateur, des questions similaires sont générées *automatiquement* ainsi que des patrons qui sont utilisés pour retrouver la réponse. L'implication textuelle est utilisée pour résoudre le problème de la variabilité de la langue (une idée peut être exprimée de plusieurs façons).

Soit un texte *(T)* et une hypothèse *(H)* on dit que *T* implique *H* si le sens de *H* peut être dérivé de T. (Negri et al. 2008).

Deux types de template sont générés selon qu'il s'agit du *nom* d'une classe ou d'une *propriété* de la classe (OU et al. 2008):

T1: Question relative au nom d'une instance de classe avec une ou plusieurs de ses propriétés. Si seulement une propriété <xxx> est prise en compte en plus de la propriété nom, le modèle est défini comme suit:

What is the name of the <class> which has the <xxx> [<xxx>_value]?

T2 : Question relative à une propriété autre que le nom.

What (or When, Where, How long) is the <xxx> of [<class>_name]?

Les variables entre crochets représentent le nom des entités et celles entre guillemets le nom des propriétés et/ou des classes. Les questions générées dépendent de la structure des triplets. Par exemple pour la structure suivante:

```
<Cinema, name, string>
<Cinema, hasPostalAddress, PostalAddress>
            <PostalAddress, street, string>
            <PostalAddress, postalCode, string>
            <PostalAddress, isInDestination, Destination>
<Destination, name, string>
```

Les questions suivantes sont générées selon le type de la question (T1 ou T2):
a) Question de type T1: What is the name of the movie which has the genre [genre_value]?
-Exemple de patron associé à la question:

```
SELECT ?movieName
WHERE {
?Movie prefix:name ?movieName.
?Movie prefix:genre "[genre_value]"^^<xsd:string>.}
```

-Questions générées:

T1-2: What is the name of the cinema which is in [street_value]?
T1-3: What is the name of the cinema which has the postal code [postalCode_value]?
T1-4: What is the name of the cinema which is in [Destination_name]?

b) Question de type T2: T2-1: What is the genre of [Movie_name]?

-Exemple de patron associé à la question :

```
SELECT ?genreValue
WHERE {
?Movie prefix:name "[Movie_name]"^^<xsd:string>.
?Movie prefix:genre ?genreValue. }
```

-Questions générée :

T2-2: What is the street of [Cinema_name]?

T2-3: What is the postal code of [Cinema_name]?

T2-4: Where is the destination of [Cinema_name]?

T2-5: What is the postal address of [Cinema_name]?

Le patron (pattern) suivant est généré :

```
SELECT ?streetValue ?postalCodeValue ?DestinationName
WHERE {
?Cinema prefix:name "[Cinema_name]"^^<xsd:string>.
?Cinema prefix:hasPostalAddress ?PostalAddress.
?PostalAddress prefix:street ?streetValue.
?PostalAddress prefix:postalCode ?postalCodeValue.
?PostalAddress prefix:isInDestination ?Destination.
?Destination prefix:name ?DestinationName. }.
```

Après la génération des questions et des patrons, l'implication textuelle est utilisée pour déterminer quelle question correspond à la question initiale de l'utilisateur. La requête SPARQL est ensuite exécutée sur l'ontologie.

L'approche de génération automatique de questions et de patrons révèle certaines insuffisances (OU et al. 2008). La première est que le patron ne peut pas couvrir toutes sortes de questions de l'utilisateur. Particulièrement pour des questions qui disposent de plusieurs classes et propriétés, il est difficile de faire une combinaison exhaustive des propriétés et de régénérer tous les patrons. La seconde insuffisance est que certaines questions générées n'ont pas de sens. Par exemple certaines questions générées pour «name of cinema» correspondent à des patrons des cordonnées GPS. Mais dans la pratique, les utilisateurs ne demandent pas cette information.

Le moteur d'implication textuelle en anglais a une précision de 81,3% pour 228 questions.

2.2.6 QEURIX

Querix (Kaufmann et al. 2006) est un système indépendant du domaine qui traduit des questions en langage naturel en SPARQL. Querix utilise Wordnet (Miller et al. 2004) pour avoir les synonymes afin de faire correspondre (matching) les mots de l'utilisateur et les éléments de l'ontologie. Il utilise également l'analyseur syntaxique StanfordParser (Klein et al. 2003). Querix procède en termes de pattern matching. Il localise les catégories grammaticales importantes de la question pour déterminer les triplets. L'ensemble de ces catégories est appelé query skeleton. Ce dernier est une séquence de nom, verbe, préposition, marque de question, conjonction ordonnée selon leur apparition dans la question.

Par exemple pour la question:

What are the population sizes of cities that are located in California?

Querix produit le "query skeleton" suivant : Q-V-N-P-N-Q-V-P-N

Où Q = question word, N = Noun, V = verb, P = preposition (Kaufmann et al. 2006)

Une fois le «query skeleton» généré, l'identification des triplets suit au moyen des synonymes générés avec Wordnet. Par exemple pour la séquence N-P-N est associée à la portion de la question «population sizes of cities » et le triplet suivant est identifié :

Sujet	Predicat	Objet
`querix:City`	`querix:hasPopulationSize`	`xsd:integer`

Les triplets sont ensuite traduits en SPARQL.

En cas d'ambiguïté, s'il n'est pas en mesure de la lever, Querix fait appel à l'utilisateur. Querix dépend de la qualité et du choix du vocabulaire de l'ontologie.

(Kaufmann et al. 2006) reporte une précision de 86,08 % et 87,11% de rappel sur 201 questions correctement analysées par l'analyseur syntaxique StanfordParser.

2.2.7 PANTO

PANTO (A Portable Natural Language Interface to Ontologies) est une interface générique hautement portable (Wang et al. 2007). La portabilité de PANTO est due au fait qu'il peut construire automatiquement le lexique (lexicon) d'une ontologie à partir d'un composant, le lexicon Builder. C'est une étape importante pour adapter le système à une nouvelle ontologie (Wang et al. 2007). A l'instar de Querix, PANTO utilise également Wordnet et StanfordPaser. PANTO utilise 2

représentations intermédiaires (query Triple et Onto Triple) avant la traduction en SPARQL. A l'image d'Aqualog, les «query triples» de PANTO sont obtenus à partir de l'analyse linguistique sans référence à l'ontologie. L'onto-triple est obtenu à partir des query-triples et du lexique. L'onto-triple est ensuite traduit en SPARQL.

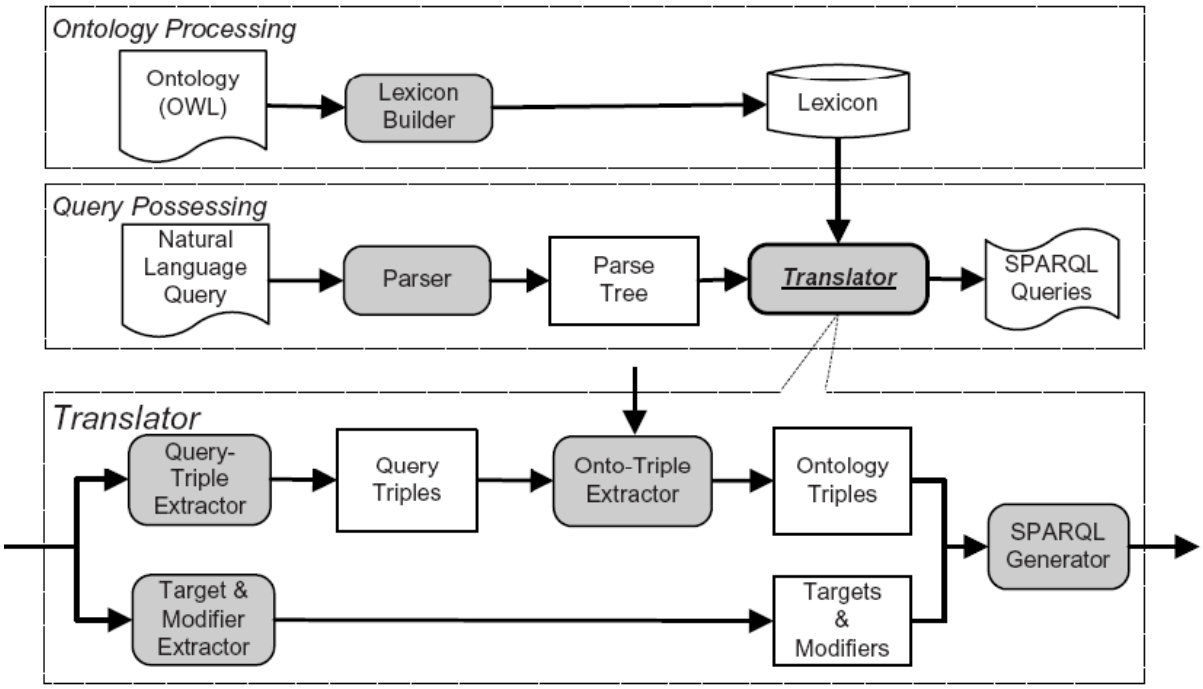


Figure 6 Architecture PANTO

Par exemple pour la question:

Which is the longest river that flows through Mississippi?

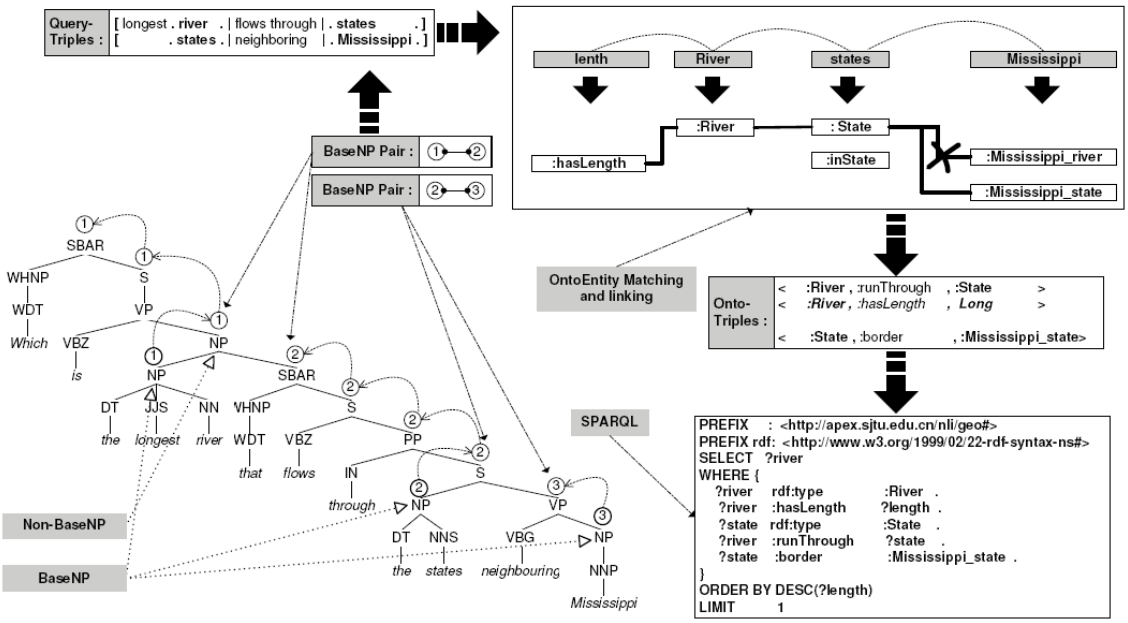


Figure 7. Processus de traduction en SPARQL

Pour extraire les query-triples, PANTO introduit la notion de BaseNP. BaseNP est un groupe nominal non récursif. Une BaseNP correspond soit au sujet ou l'objet d'un triplet. Le prédicat correspond à une préposition, un verbe. Il est possible que le prédicat soit omis également. Le détail de l'identification du triplet suit l'algorithme de (Collins, 1999) avec l'exception que lorsque le mot a une relation de conjonction, il est traité comme un noyau (de groupe nominal) (Wang et al. 2007).

A ce stade, pour cet exemple les query-triples suivants sont générés :

```
[longuest river | flows through| state]

[states| neighboring| Mississipi]
```

La formation des onto-triples suit deux étapes:

-PANTO établit une correspondance (matching) entre les mots de la requête de l'utilisateur et les «onto entities» (abréviation d'entité de l'ontologie)

-Une correspondance (matching) est ensuite établie entre les query-triples et les «onto entities». Pour générer un onto-triple à partir d'un query-triple toutes les onto-triples possibles avec le sujet ou l'objet d'un query-triple sont générés en respectant le domaine et le codomaine de l'onto-triple. Si le prédicat du query-triple n'est pas vide, il est utilisé pour éliminer les onto-triples ayant un prédicat non concordant.

Pour cet exemple le schéma de l'ontologie est la suivante (Wang et al. 2007) :

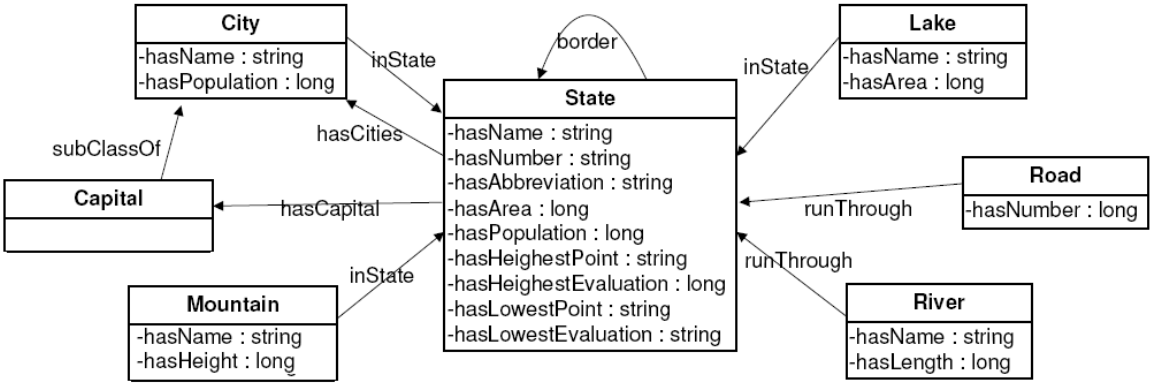


Figure 8. Structure de la base de données

Avec les «onto-entities» de la Figure 7 précédente et le schéma de l'ontologie les «onto triples» suivantes sont générés pour le «query-triple».

```
[longuest river | flows through| state]:

<:Mississippi, rdf:type, :River> et

<:River, :runThrough, :Mississippi>.
```

Puisque le schéma de l'ontologie a le prédicat <runThrough> qui correspond en domaine et en codomaine au prédicat <flows through>, l'onto-triple <:Mississippi, rdf:type, :River> est éliminé.
Les onto-triples sont ensuite reliés pour refléter l'arbre syntaxique (Wang et al. 2007).
Les limitations de PANTO (Wang et al. 2007) se situent au niveau de la restriction de la couverture des requêtes, la faiblesse de l'interaction avec l'utilisateur, et l'évolutivité.
Bien que l'analyse en 3 étapes permette de couvrir plusieurs requêtes en langage naturel, le champ d'application pris en charge est encore faible. Du fait que PANTO utilise l'analyseur syntaxique StanfordParser (qui n'est pas conçu peut être pour ses besoins spécifiques), il est limité à la capacité de cet analyseur qui dépend des techniques de traitement automatique des langues utilisées. Autrement dit, la question doit être analysée correctement par StanfordParser.

L'ontologie de test est relativement petite. (Wang et al. 2007) reporte une précision de 88,05% et 85, 86% de rappel pour 877 questions du domaine géographique. Les questions testées sont correctement analysées par l'analyseur syntaxique de Standford (StanfordParser). PANTO est l'un des systèmes ayant un corpus de test important avec une performance significative.

2.2.8 SWAT

SWAT (Bernstein et al. 2005a) est un système qui permet de formuler une question dans une langue contrôlée spécifique à savoir ACE[6] (Attempto Controlled English). ACE est une langue contrôlée issue de l'anglais, conçue pour une communication relevant d'un domaine de communication, y compris des spécifications techniques. SWAT a la particularité d'utiliser un langage prédéfini indépendamment de la base de données.

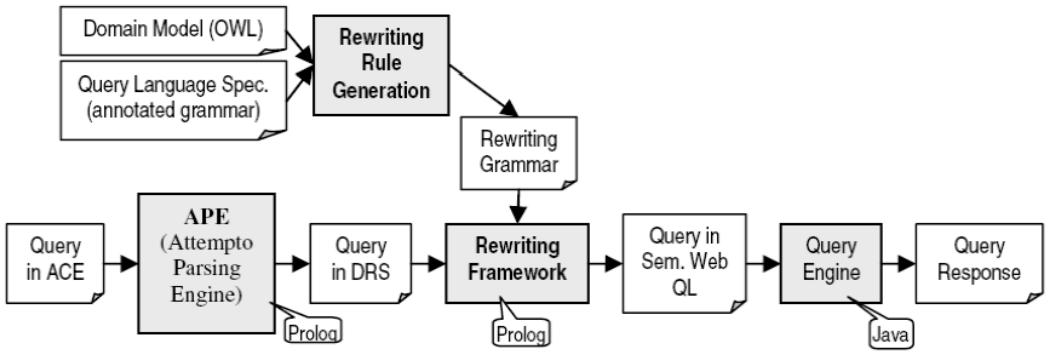


Figure 9. Processus de génération de requête pour SWAT

La traduction se déroule comme indiquée à la figure ci-dessus. La requête en ACE est analysée par l'APE (l'analyseur syntaxique d'ACE). Le résultat de l'analyse produit une requête présentée sous la forme de DRS (Discours Représentation Structure) qui est une variante de la logique du premier ordre. Chaque requête DRS est convertie en PQL (Process Query Language) en utilisant des structures basées sur des ontologies. A la Figure 9, le résultat de cette étape correspond à la phase «*Query in Sem. Web QL*». PQL est le langage d'interrogation, il ne s'agit pas d'un langage donné. Il peut être n'importe quel langage. Le moteur du langage concerné est ensuite utilisé pour retrouver la réponse dans la base de données.

Trois limitations sont présentées par (Bernstein et al. 2005a). La première limitation concerne l'usage de la langue contrôlée. Les utilisateurs doivent apprendre la langue contrôlée, ce qui est moins motivant pour les utilisateurs. (Bernstein et al. 2005a) argumente qu'il est plus facile d'apprendre une langue contrôlée qu'un langage logique comme SQL par exemple. Selon (Bernstein et al. 2005a), le développement des éditeurs en langue contrôlée permet de contourner le problème de l'apprentissage. Le langage utilisé pour interroger une source de données ne consiste pas en un langage compliqué que les humais utilisent entre eux lors d'une communication et ce langage peut être modélisé par une langue contrôlée.

[6] http://attempto.ifi.uzh.ch/site/

La seconde limitation est que les règles de réécriture des métadonnées ont besoin d'être changées lors d'une adaptation du système à un nouveau domaine.

La dernière limitation concerne la cadre de l'évaluation. Cette dernière a été conduite entièrement par des informaticiens et des linguistes qui peuvent avoir une idée du fonctionnement du système.

(Bernstein et al. 2005a) reporte une précision de 100% et un rappel de 90% sur 30 questions écrites dans le langage contrôlée ACE.

2.2.9 GINSENG

GINSENG (Guided Input Language Search Engine) permet aux utilisateurs d'interroger une ontologie en utilisant une langue contrôlée (Bernstein et al. 2005b; Bernstein et al. 2006). GINSENG utilise 2 types de grammaire pour contrôler les questions de l'utilisateur. Une grammaire dynamique et une grammaire statique. La grammaire dynamique est conçue à partir de l'ontologie. Lorsqu'un utilisateur démarre l'application, toute l'ontologie est chargée dans l'environnement d'exécution et un compilateur de grammaire est utilisé pour générer une grammaire dynamique pour chaque classe, propriété ou instance (Smart, 2008). La grammaire dynamique est spécialement destinée à contrôler la saisie de l'utilisateur. Seules les phrases valides sont acceptées par la grammaire active. De cette manière, l'utilisateur est empêché de saisir des phrases qui ne peuvent pas être traitées par le système. Ceci est une approche puissante dans la mesure où elle limite la gamme des requêtes que l'utilisateur peut rentrer aux requêtes qui peuvent avoir une réponse valide en RDQL. Cependant, toute question que l'utilisateur a pu rentrer devra être convertie en RDQL.

L'analyse syntaxique du système est composée de la grammaire statique et de la grammaire dynamique. Pour rendre l'approche compréhensible, considérons l'exemple suivant:

What state borders New York State?

Naturellement, on suppose que la phrase a été acceptée par la grammaire dynamique et statique définissant l'ontologie. Examinons un échantillon de la grammaire qui a permis la génération de la phrase:

```
(1) <START> ::=<OQ> ?
                              | SELECT <<OQ>>

                              | WHERE (<<OQ>>)

 (2)  <OQ> ::= what <subject> <verb>
```

```
     |  <<subject>>

     |<<subject:1>>  <<subject>>)(<<subject:1>> <<subject>>

(3) <subject>  ::= state

                            |  ?state

                            | <rdf#type> <geo#state>

(4)  <verb> ::= borders <object>
                            | -

                            | <geo#borders>  <<object>>
(5)  <object> :: New York State
                            |-

                            | <geo#newYorkState>
```

La grammaire suit la notation BNF[7] . Les non terminaux sont notés entre <>. Le symbole | désigne la disjonction «ou». L'axiome de la grammaire est la règle (1) qui retourne un non terminal <OQ> (partie droite de la règle). La règle correspondant à <OQ> est ensuite identifiée dans la partie gauche des règles. La règle correspondante est la règle (2) (c'est la seule règle qui a un symbole <OQ> à gauche). La partie droite de cette règle, dans ce cas est 'what <subject> <verb>'. Le symbole <subject> et <verb> sont encore des non terminaux, leur définition est encore rechercher dans la partie gauche des règles de la grammaire, mais 'what' est un symbole terminal, et il est présenté à l'utilisateur comme un symbole valide de la question en langage naturel. La règle générale est que tous les symboles non terminaux sont définis dans la partie gauche et donc recherchés dans cette partie alors que les symboles terminaux sont présentés à l'utilisateur comme des symboles valides. Les règles (1) et (2) sont des règles de la grammaire statique. Le mot «what» est souvent utilisé pour introduire une question, il est concevable de l'inclure comme une partie indépendante de l'ontologie, le restant des règles sont cependant dynamiques. Ces règles sont générées par GINSENG à tous les chargements de l'ontologie par un utilisateur.

Continuant avec l'exemple, une fois que l'utilisateur a choisi le mot « what» de la liste proposée par GINSENG, la grammaire cherche la définition du non terminal <subject> de la règle (2). Ce non terminal <subject> est défini par la règle (3). On se déplace au symbole <verbe> de la règle (2). Cette

[7] http://cui.unige.ch/db-research/Enseignement/analyseinfo/AboutBNF.html

règle est définie par la règle (4) qui donne le mot «borders». Le symbole <object> donne à son tour New York.

Maintenant considérons le processus de génération de la requête formelle RDQL (RDF Query Language). RDQL est un langage similaire à SPARQL. C'est un langage candidat à la recommandation pour l'interrogation de bases de données basées sur RDF. Pour chaque règle de la grammaire, il existe 2 lignes, chacune précédée par le symbole |. La première de ces 2 lignes représente la partie SELECT de la requête RDFQL ; le second représente la partie WHERE de la requête. Les règles ci-dessus sont l'ensemble des règles minimales pour former une requête RDQL c'est-à-dire pour pouvoir former un triplet dans la clause WHERE. Examinons d'abord la ligne qui correspond à la clause SELECT. Les règles (1) et (2) contribuent à la formation de la clause SELECT en donnant:

```
SELECT   ?state
```

Maintenant qu'en est-il de la clause WHERE ? Dans ce cas toutes les règles sont utilisées et leur exécution séquentielle donne :

```
WHERE
   (?state   <rdf#type>  <geo#state> )
   (?state    <geo#borders>  <geo#newYorkState>)
```

La première chose qu'il faut noter ici est que le symbole <<subject : 1>> dans la règle (2) est utilisé pour faire une référence arrière à la première occurrence de object dans la règle. Cette référence arrière permet de résoudre le symbole <<subject>> en formant le symbole '?state' qui correspond au premier élément des 2 triplets de la clause WHERE. La seconde chose qu'il faut noter est que la clause WHERE de la règle (2) comprend 2 symboles non terminaux séparés par des parenthèses. C'est ce qui nous donne les 2 patrons de triplets associés à la clause WHERE de la requête RDQL[8]. Finalement, on peut combiner la partie SELECT et la partie WHERE pour former une requête RDQL syntaxiquement valide qui peut retrouver l'information recherchée dans la base de données.

```
SELECT   ?state
WHERE
   ( ?state   <rdf#type>  <geo#state> )
   (?state    <geo#borders>  <geo#newYorkState>)
```

[8] http://www.w3.org/Submission/RDQL/

Comme on peut le constater, le vocabulaire utilisé par GINSENG est limité par l'ontologie ou les ontologies à partir desquelles les règles de la grammaire sont créées. Si le système est restreint seulement aux noms de classes, propriétés ou instances de l'ontologie alors les mots disponibles pour l'utilisateur pour poser une question seront limités. Pour contourner cette limitation GINSENG permet aux utilisateurs d'annoter une ontologie avec des synonymes. Ces synonymes sont traités et utilisés pour créer des mots alternatifs aux mots avec lesquels l'ontologie est créée.

En résumé GINSENG utilise une forme de langue contrôlée, qui, contrairement au système SWAT n'est pas liée à la langue en tant que tel mais plutôt à l'ontologie cible. Contrairement à l'interface en langage naturel comme PANTO et Querix, GINSENG ne traite pas une véritable question en langage naturel comme entrée. Etant donné que les questions sont limitées à celle que la grammaire peut traiter, la précision et le rappel de GINSENG sont considérablement élevés.

Les limitations de GINSENG sont pareilles à celle de SWAT (voir paragraphe 2.2.8). (Bernstein et al. 2005b) reporte une précision de 92.8% et un rappel de 98.4% sur 352 questions.

2.2.10 MMS (Méta Modèle Sémantique)

Le MMS (El Abed, 2001 ; Cardey et al. 2001) permet la modélisation de la structure sémantique d'une base de données quelconque. La modélisation inclut la classification des domaines de la source de données avec la classification des mots clés contenus dans la requête exprimée en langage naturel. La classification comporte des mots clés spécifiques et des mots clés génériques. Les mots clés génériques sont des mots indépendants du domaine. Ils peuvent être des accessoires ou des mots qui introduisent une contrainte (condition) imposée dans la clause where de SQL. Les mots clés spécifiques dépendent du domaine. Il existe 2 catégories de mots clés spécifiques: informationnel et classe (classifying). Un mot clé de type «classifying» correspond à un nom de table. Un mot clé de type informationnel correspond à un attribut d'une table. Considérons l'exemple suivant:

Give me the tariffs for the museums in Besançon

Les mots clés sont regroupés comme suit :

- accessoire (accessory): Give, me, the, for, in.
- informationnel (informational): tariffs
- classe (classifying): museums

Dans le modèle de données, Besançon est une occurrence de l'attribut "town" de la classe "museum". L'analyseur peut retrouver l'information grâce au méta modèle suivant:

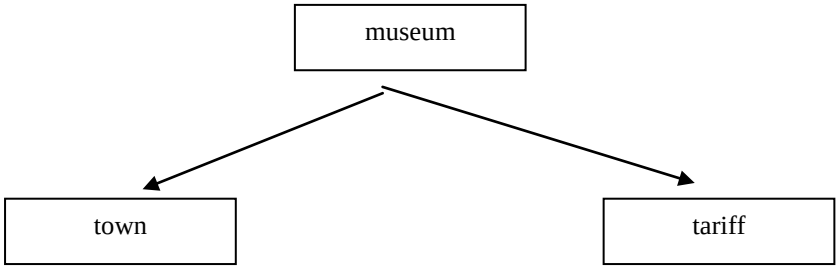


Figure 10. Extrait de réseau sémantique

Grace à ce graphe, l'analyseur déduit les informations manquantes. Cette déduction permet d'attribuer la valeur «Besançon» à l'attribut town. Le processus de détermination des valeurs et des variables suit le schéma suivant:

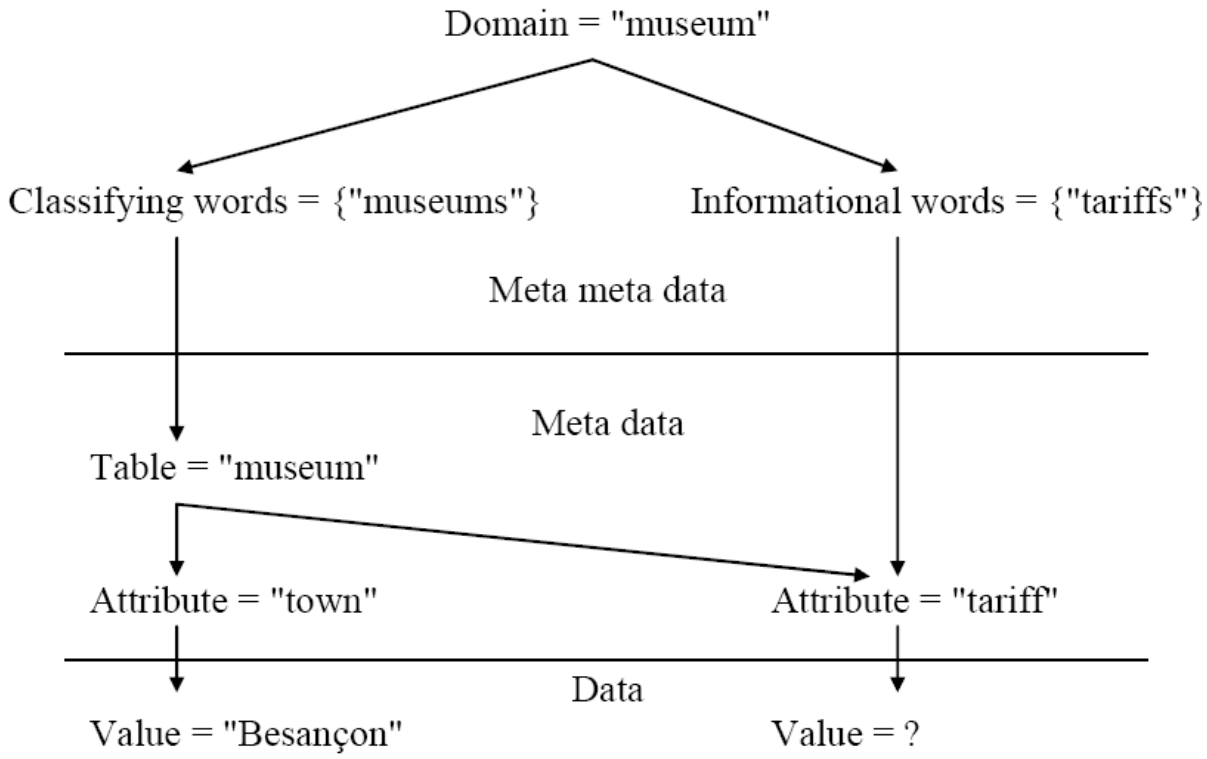


Figure 11. Détermination des valeurs et variables de MMS

A la fin de la détermination des valeurs et variables (Figure 11), les variables sont affectées à la clause SELECT, les valeurs utilisées comme conditions dans la clause WHERE, les mots correspondant aux tables sont rangés dans la clause FROM de SQL. La requête peut finalement s'écrire comme suit:

```
SELECT museum.tariff FROM museum WHERE museum.town = "Besançon"
```

2.2.11 Catégorisation hiérarchique

La catégorisation hiérarchique est une méthode utilisée par (Soumana, 2010) pour traduire des questions en requête SPARQL. La catégorisation est utilisée pour identifier les mots ayant un sens pour la réponse à la question. Le caractère hiérarchique s'intéresse à la relation sémantique entre les classes, les instances des classes et les propriétés. La méthode utilise le schéma de la base de données pour établir cette relation. Par exemple la question:

I am looking to go and see a film at the cinema in Walsall on Thursday the sixteenth of August and I would like it to be rated such that it contains moderate violence please.

peut être réduite à:

a film at the cinema in Walsall on Thursday the sixteenth of August that contains moderate violence

Les mots de la question sont catégorisés en *descripteurs* et en *marqueurs*. Les descripteurs sont les mots qui ont un sens dans la base de données. Ils correspondent aux classes et aux propriétés de la base de données. Les marqueurs correspondent aux mots linguistiques. Ils peuvent être des mots ou unités lexicales qui indiquent la quantification, la comparaison, la négation, des mots interrogatifs, des prépositions. Le schéma suivant décrit les étapes du traitement :

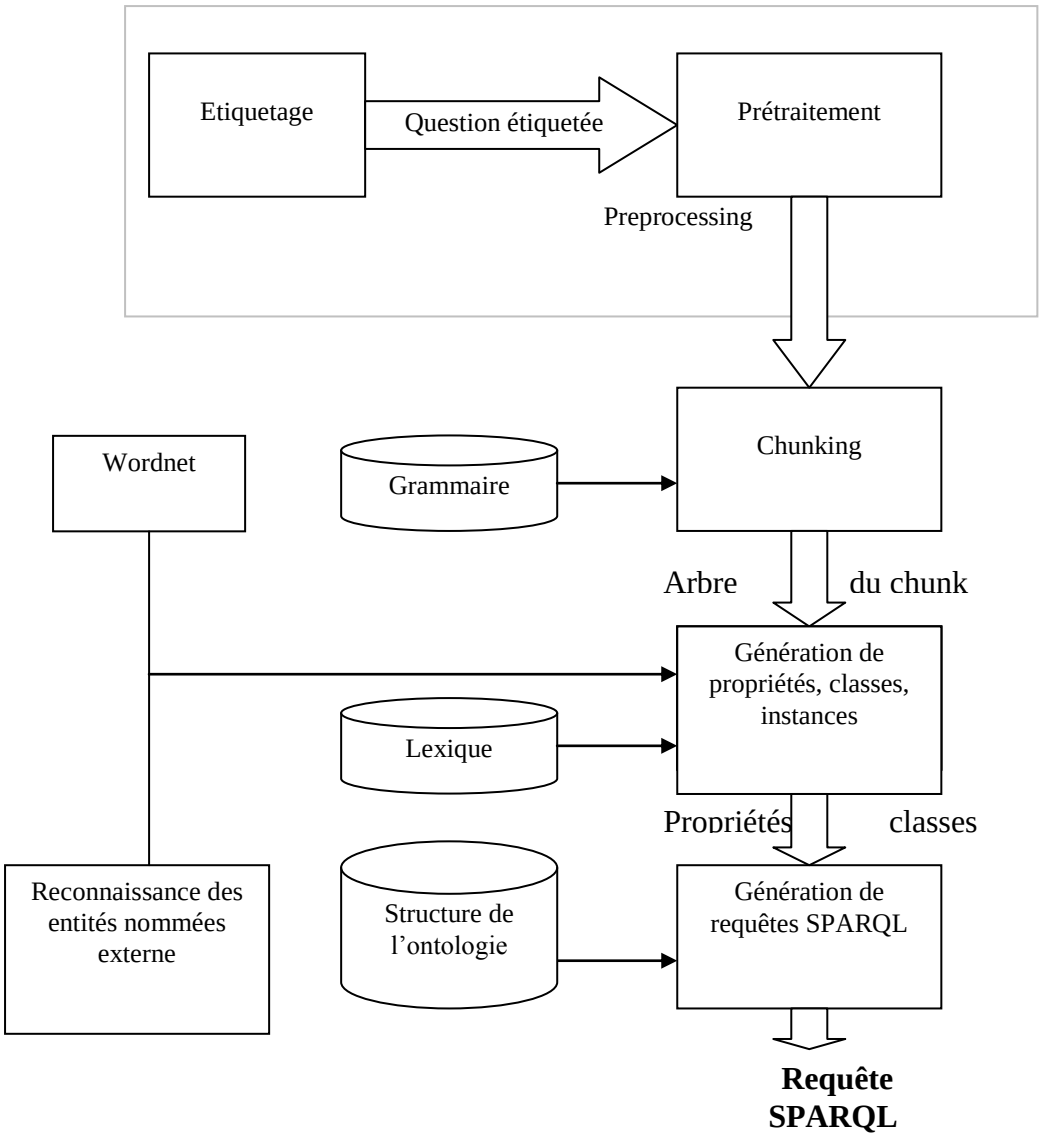


Figure 12. Architecture du système

Les principaux composants du système sont :

- Prétraitement : le prétraitement corrige les erreurs d'étiquetage. NLTK[9] est utilisé pour l'étiquetage et le chunking. Certaines désambiguïsations sont effectuées durant cette phase. Par exemple différencier le pronom relatif « who » du pronom interrogatif « who » qui peut être un focus à une question.

- Chunking : cette étape utilise une grammaire pour former des structures syntaxiques de base principalement de groupe nominal, prépositionnel ou verbal.

- Génération des propriétés, instances et classes : le système effectue la reconnaissance des classes, des instances de classes, des propriétés ou des valeurs de ces propriétés.

[9] http://www.nltk.org/

- Génération des requêtes SPARQL : à ce niveau la requête SPARQL est générée. L'ontologie du projet QALLME[10] est utilisé particulièrement sur l'ontologie Movie et Cinema (voir figure suivante).

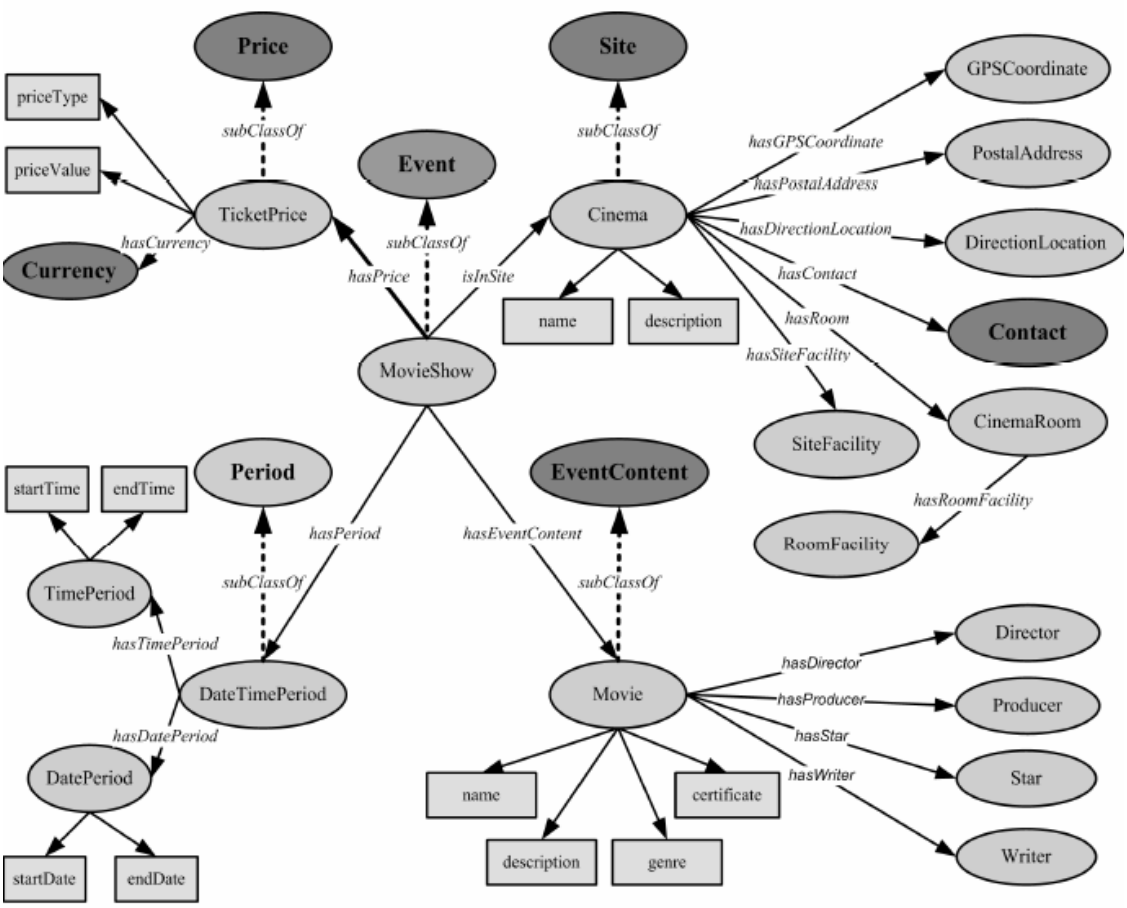


Figure 13. Movie & cinema ontology (OU et al. 2008)

Par exemple pour la question suivante :

What children's films are showing tomorrow afternoon at the Showcase Cinema in Walsall?

La phase de génération procède comme suit:

- Les classes, les instances, les propriétés et les valeurs des propriétés sont inventoriées comme suit :

[10] http://qallme.fbk.eu/

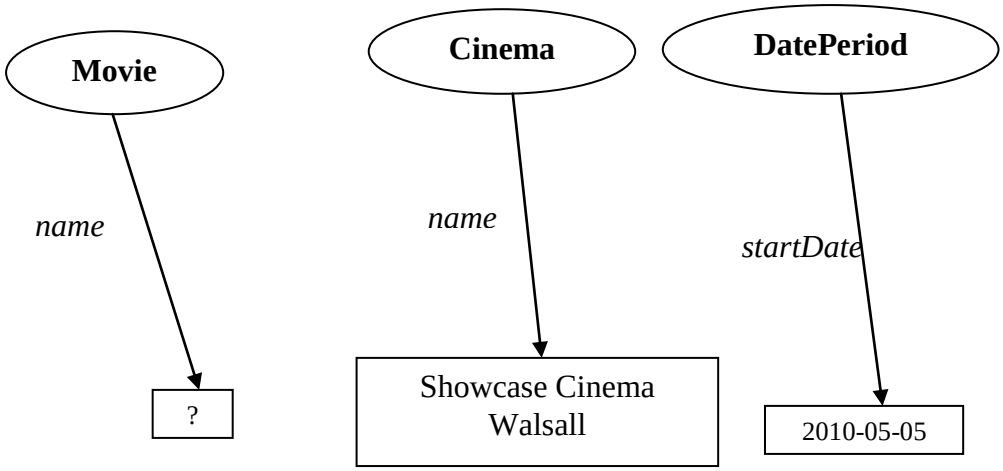


Figure 14. Inventaire des éléments de la question

Chaque valeur, instance de classe ou valeur de propriété est rattaché à la classe ou à la propriété. Le point d'interrogation est le focus de la question.

- La seconde étape consiste à générer un arbre à partir de cet inventaire en se basant sur l'ontologie. Avec cet inventaire, l'arbre suivant peut être obtenu.

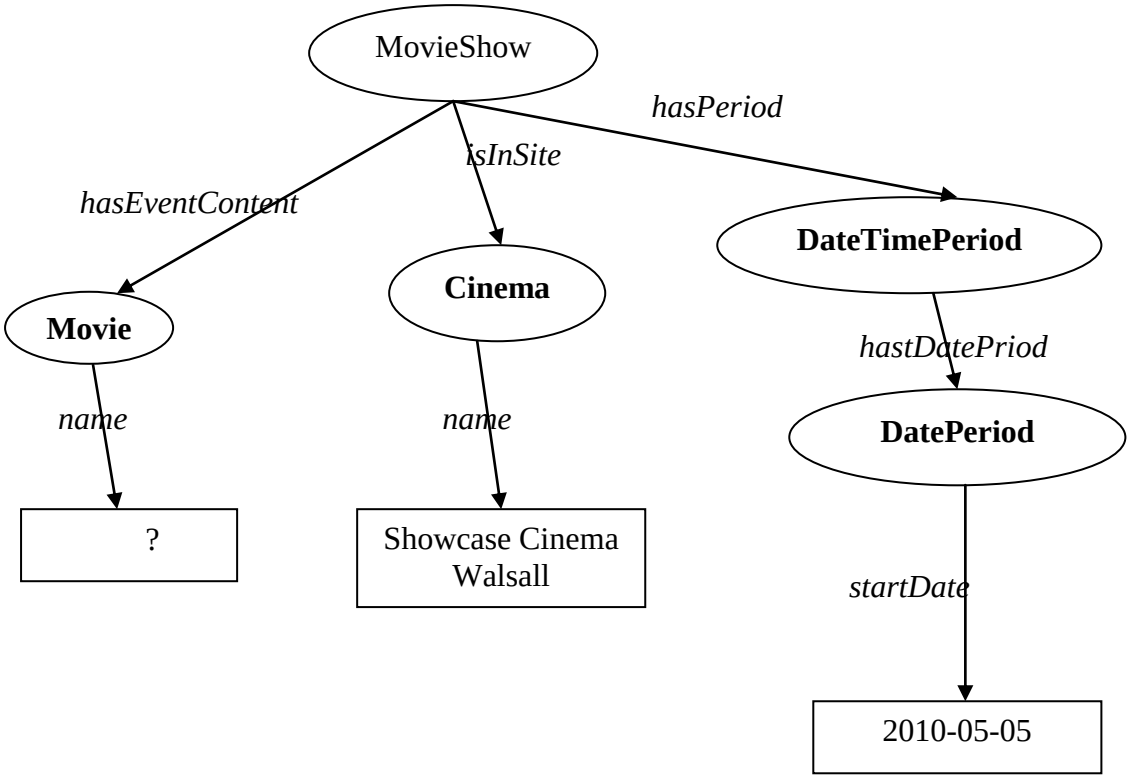


Figure 15. Formation de l'arbre

Le requête SPARQL peut-être générée comme suit :

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
```

```
PREFIX base:<http://qallme.itc.it/ontology/qallme-tourism.owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT distinct ?movieName
WHERE {
?Event rdf:type base:Event.
?Event  base:hasPeriod  ?DateTimePeriod.
?DateTimePeriod   base:hasTimePeriod  ?TimePeriod.
?TimePeriod   base:startDate ?TimePeriodstartDate.
?Event   base:hasEventContent  ?Movie.
?Movie  base:name  ?Moviename.
?Event   base:isInSite  ?Cinema.
?Cinema   base:name ?cinemaName.
FILTER( ?TimePeriodStartDate="2010-05-05"^^xsd:date).
FILTER(?cinemaName="Showcase Cinemas Walsall"^^xsd:string).}
```

(Soumana, 2010) reporte une précision de 91,42% sur 100 questions issues du corpus du projet QALLME.

2.3 Systèmes utilisant un jeu de données larges et hétérogènes

Les systèmes précédents présentent la particularité d'utiliser des jeux de données diverses pour leur évaluation. Ils apportent une approche méthodologique intéressante même s'ils sont basés sur des jeux de données restreints. A partir de 2011, les campagnes QALD[11] servent de cadre pour l'évaluation des Systèmes de Questions-Réponse sur des grands volumes de données comme Linked Data[12]. Contrairement aux systèmes précédents, les performances de ces systèmes sont faibles pour le moment. Ils ont une F-mesure de l'ordre de 0,16 à 0,58.

2.3.1 FREyA

FREya (Damljanovic et al., 2011) est un système interactif. L'interaction engagée avec l'utilisateur vise à :

[11] http://greententacle.techfak.uni-bielefeld.de/~cunger/qald/index.php?x=home&q=1

[12] http://linkeddata.org/

- améliorer le rappel en enrichissant le lexique du système avec le vocabulaire de l'utilisateur,
- améliorer la précision en résolvant l'ambigüité à travers le dialogue avec l'utilisateur. Les suggestions faites à l'utilisateur sont d'abord trouvées par inférence sur l'ontologie. Le système apprend à partir des choix de l'utilisateur.

FREyA est basé sur l'architecture suivante :

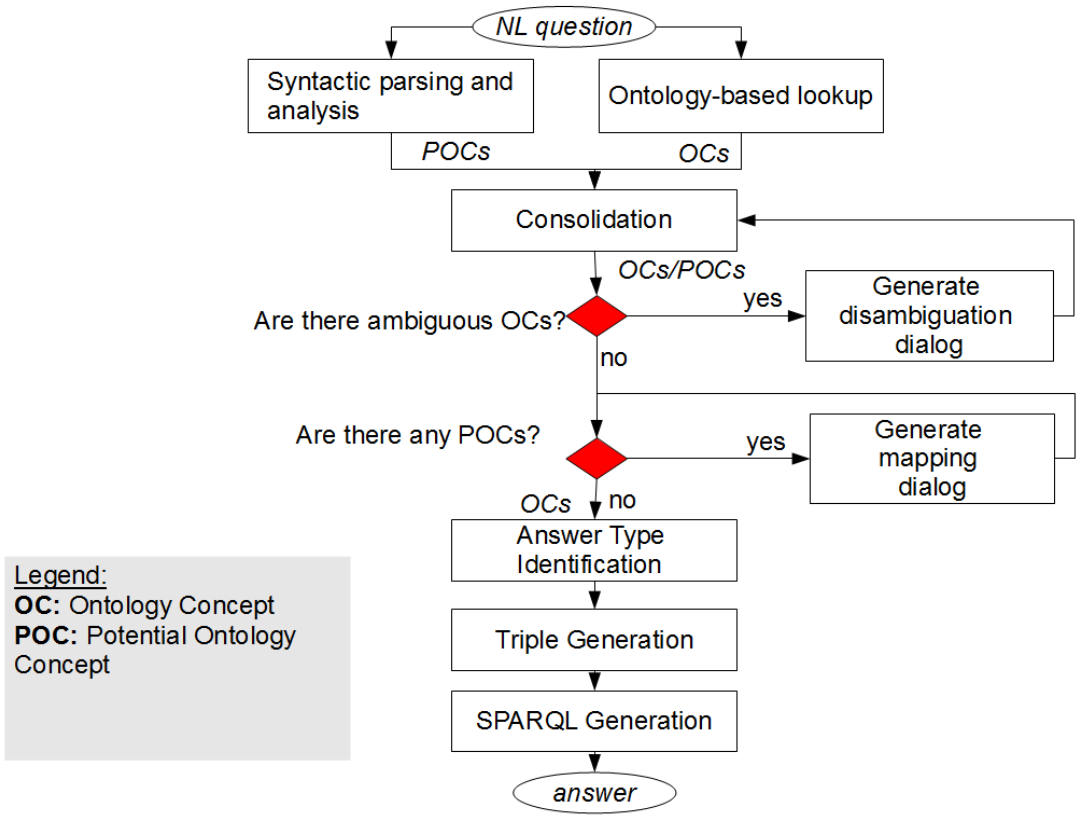


Figure 16. Architecture de FREyA

Le système effectue une analyse linguistique et une recherche dans l'ontologie. Le composant « Syntactic parsing and analysis » génère un arbre en utilisant l'analyseur syntaxique de Stanford parser et utilise ensuite plusieurs heuristiques pour déterminer les concepts potentiels de l'ontologie POCs (Potentiel Ontology Concepts). Les POCs font référence aux termes de la question qui peuvent être liés mais pas nécessairement à l'ontologie. Il peut s'agir des verbes, noms, ou des mots interrogatifs comme *qui, quand, où*.

Le composant « Ontology-based Lookup » identifie les termes de la question qui sont dans l'ontologie (OC : Ontological Concepts) sans considération du contexte ou de la grammaire. Les OCs représentent les classes, les instances de classe et les valeurs de propriété.

La phase de consolidation permet de relier les POCs qui n'ont pas pu être reconnus automatiquement. C'est à ce niveau que le dialogue est engagé avec l'utilisateur si nécessaire. Il existe deux types de dialogue :

- le dialogue de désambigüisation (disambiguation dialogs) invite l'utilisateur à résoudre les ambigüités identifiées dans la question.
- Le dialogue de sélection (mapping dialog) invite l'utilisateur à trouver le concept équivalent à un POC dans l'ontologie.

Après cette étape, le système génère les triplets RDF[13], puis la requête SPARQL.

2.3.2 CASIA

Le système CASIA (He et al., 2013) utilise une approche basée sur les triplets. Le système est composé de 3 parties principales :

- L'analyse de la question : étant donnée une question en langage naturel, le type de la question (de type oui/non, nombre, etc.) et les entités de la question sont déterminés premièrement. Pour déterminer le type de la question, des expressions régulières sont utilisées, par exemple :
  - "^ (are|did|is|was|does|were|do).*" : pour des questions de type oui/non
  - "^ how(many|tall|long).*" : pour des questions de « somme »
  - "^ (what|which|where|who).*" and "^ (show me|give me|list).*" : pour des question de type "SELECT".

  CASIA utilise Stanford NER[14] pour identifier les entités nommées, puis générer un arbre de dépendance incluant les entités nommées, des mots clés et des catégories grammaticales. Cet arbre est utilisé pour construire des triplets appelés query triple. Par exemple pour la question :

  Who are the parents of the wife of Juan Carlos I?

  Les query triples générés sont:

  <?who, are the parents of, wife>

[13] http://www.w3.org/RDF/

[14] http://nlp.stanford.edu/software/corenlp.shtml

<?wife, the wife of, Juan Carlos I >

- Mise en correspondance des ressources : cette étape consiste à traduire les query triples en triplets de l'ontologie. Un query triple peut correspondre à plusieurs triplets de l'ontologie :

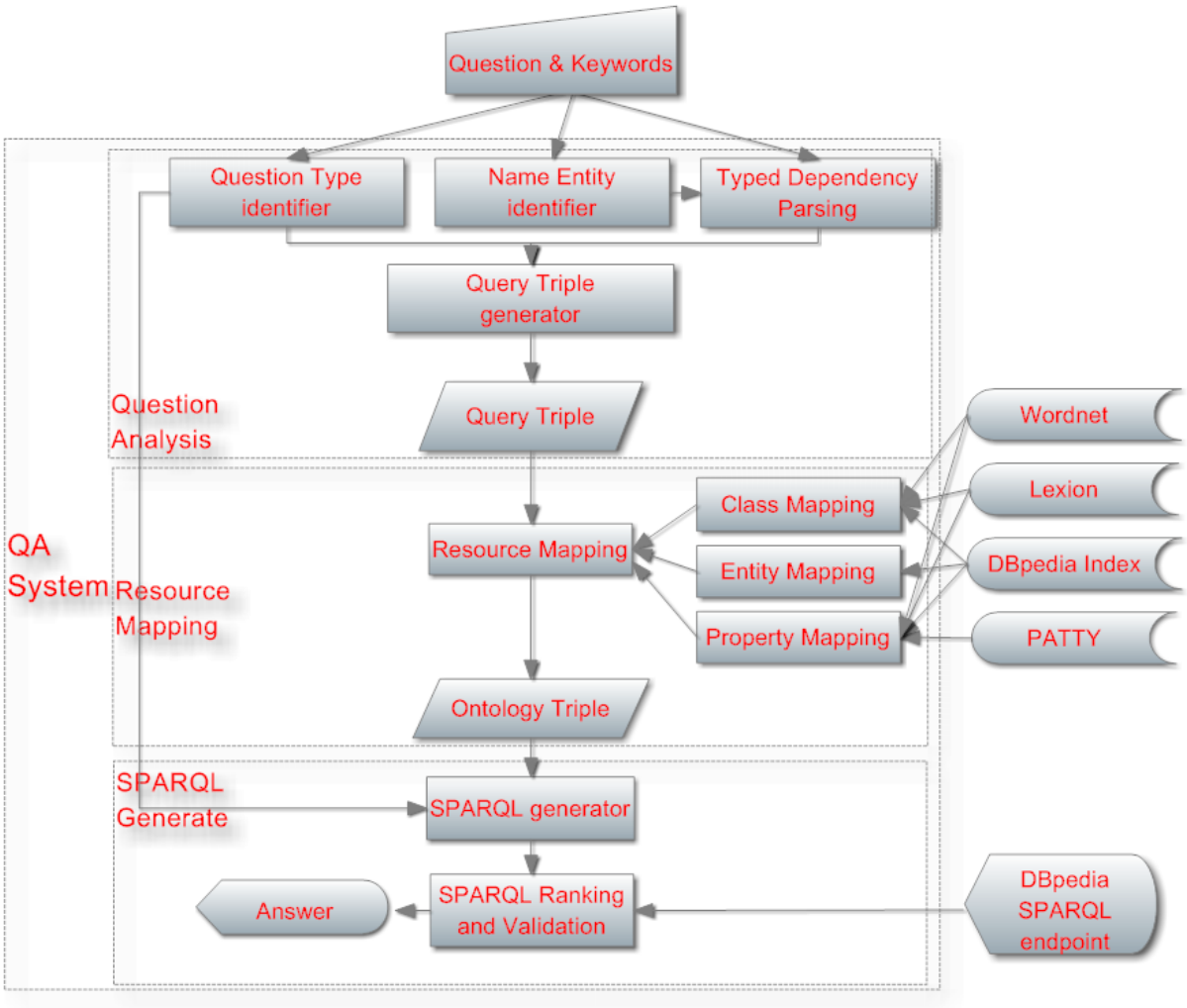


Figure 17. Architecture de CASIA

```
<?Person, rdf:type, dbc:Person>
<?Person, dbp:parent, ?wife>
<?wife, dbo:spouse, dbr:Juan Carlos I Of Spain>
```

- Génération de la requête SPARQL: selon le type de la question (booléen, nombre, etc.) et le résultat de la mise en correspondance des requêtes SPARQL candidates sont générés. Ces requêtes sont validées sur DBpedia. La requête avec le score le plus élevé sera retenue comme réponse finale.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dbp: <http://dbpedia.org/resource/>
SELECT DISTINCT ?URL
WHERE {
?URL rdf:type dbo:Person.
?URL dbo:parent ?wife.
?wife dbo:spouse dbp:Juan Carlos_I_of_Spain.}
```

2.4 Conclusion

Les interfaces en langage naturel développées depuis les années 70 connaissent de nos jours un regain d'intérêt. Avant la fin de la décennie 2000, la plupart des systèmes étaient développés pour des bases de données restreintes. La portabilité était le défi principal. La question de la portabilité n'a pas été définitivement résolue. L'utilisateur a généralement une notion du contenu de la base de données ou des types d'information qu'il peut en tirer. La base de données couvre généralement un domaine. Cette situation a favorisé le développement des interfaces qui ont un très bon résultat.

Les principales approches sont orientées schémas, patron (pattern), triplets, interaction avec l'utilisateur.

A partir de la décennie 2010, le volume de données générées devient important. Les utilisateurs potentiels qui veulent accéder aux données s'élargit. Ces utilisateurs n'ont pas besoin de connaitre la structure sous-jacente des données. De nouveaux défis s'ajoutent donc : celui de l'hétérogénéité de données et la capacité à bien comprendre les questions des utilisateurs.

CHAPITRE 3 : LA DIMENSION DOMAINE

Alors que la question de la portabilité des interfaces en langage naturel n'est pas définitivement résolue, l'avènement du Big Data[15] complique d'avantage l'accès à l'information. Maintenant il s'agit de faire en sorte que l'interface soit opérationnelle non pas seulement d'un domaine à un autre, mais qu'elle soit opérationnelle simultanément dans plusieurs domaines. Dans l'environnement Big Data, l'évolution des jeux de données est imprévisible. Un système d'interrogation dans ce contexte doit être en mesure de prendre en compte facilement les nouveaux changements, ce qui demande une architecture stable.

Les données sont stockées dans des bases de données non seulement pour les mémoriser mais aussi pour pouvoir faire des traitements sur ces données ultérieurement. Ce traitement peut être juste une simple consultation (afficher la donnée par exemple) ou des opérations complexes pour tirer une information. Les questions que les utilisateurs posent ont ce même but. Elles consistent soit à extraire une donnée brute, soit à afficher une donnée résultant de l'agrégation de plusieurs autres données. Pour faire ce traitement (ou ces séries d'opérations), il est important de connaitre les éléments qui participent à l'opération notamment l'opérateur et les opérandes. Notre approche globale consiste à bien identifier les opérateurs et les opérandes et la manière dont les opérations se composent en langage naturel.

Nous soutenons l'idée qu'une considération plus accrue pour la nature des données (leur propriété), des opérations applicables aux données (les traitements possibles avec les données) et de la manière dont ses opérations s'enchainent permet d'accroitre la robustesse des interfaces.

Nous distinguons deux types de domaines :

- un domaine abstrait: ce domaine se réfère à des notions indépendantes du domaine métier. C'est le cas par exemple des mesures qui interviennent dans plusieurs domaines (Soumana et al., 2012b).
- un domaine concret: ce domaine concret couvre le domaine métier.

Les deux domaines ne sont pas nécessairement disjoints. Donc il est possible d'avoir un chevauchement.

Par la suite, nous nous intéressons principalement au domaine abstrait du fait qu'il est limité, le domaine concret dépendant du métier.

[15] http://www-01.ibm.com/software/fr/data/bigdata/

3.1 Volume et hétérogénéité des données

Après le Web de documents, ces dernières années a vu le fleurissement des données provenant de divers domaines (Bizer et al., 2013) : le Web de données. Ces données décrivent des personnes, des entreprises, des livres, des publications, des filmes, des programmes de télévisions et radios, des gènes, des protéines, des médicaments et essais cliniques, des communautés en ligne, des données statistiques et scientifiques, des revues, des lieux, etc. Le volume de ces données croît rapidement. Les figures suivantes montrent la progression de l'interconnexion des données.

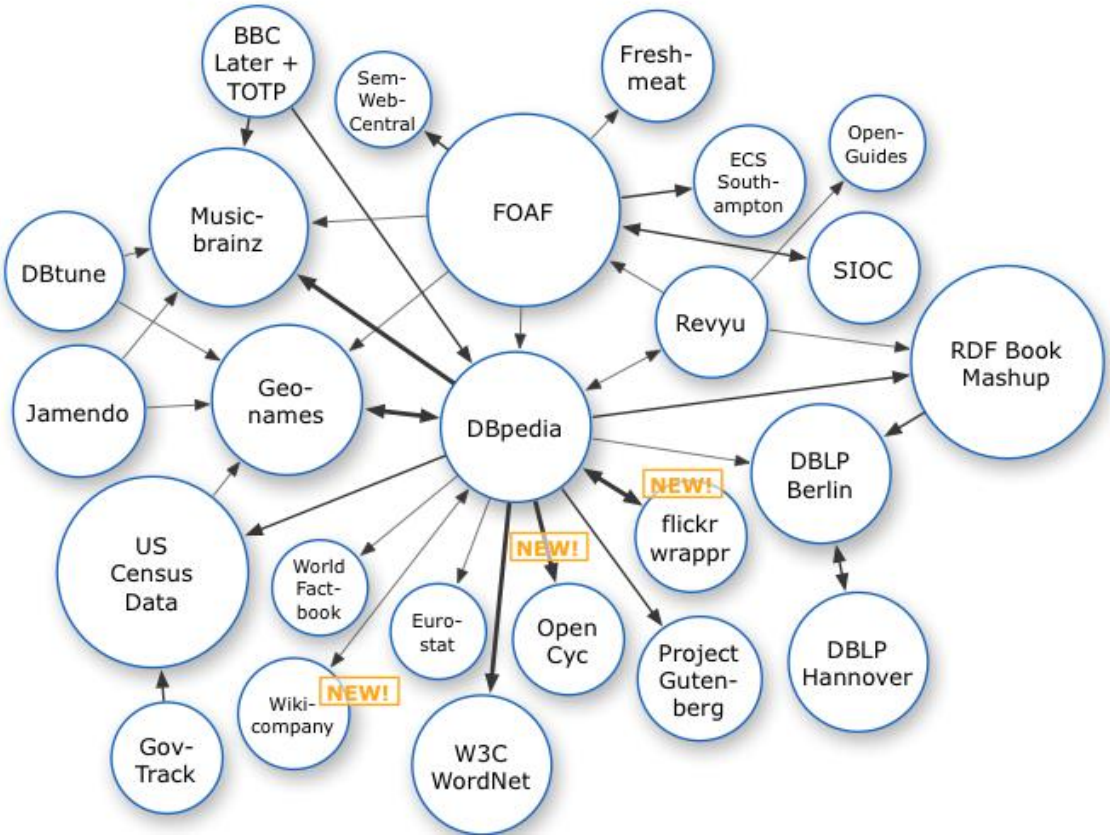


Figure 18. Interconnexion des données structurées (Octobre 2007)

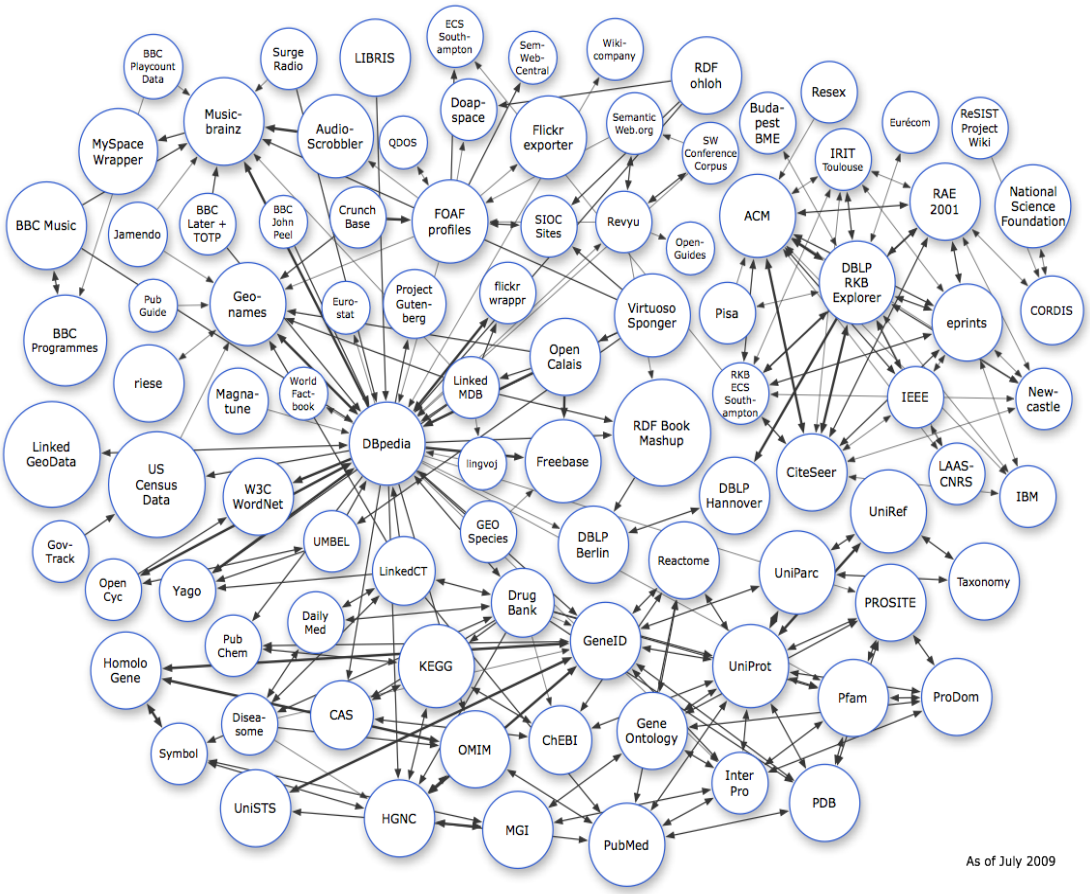
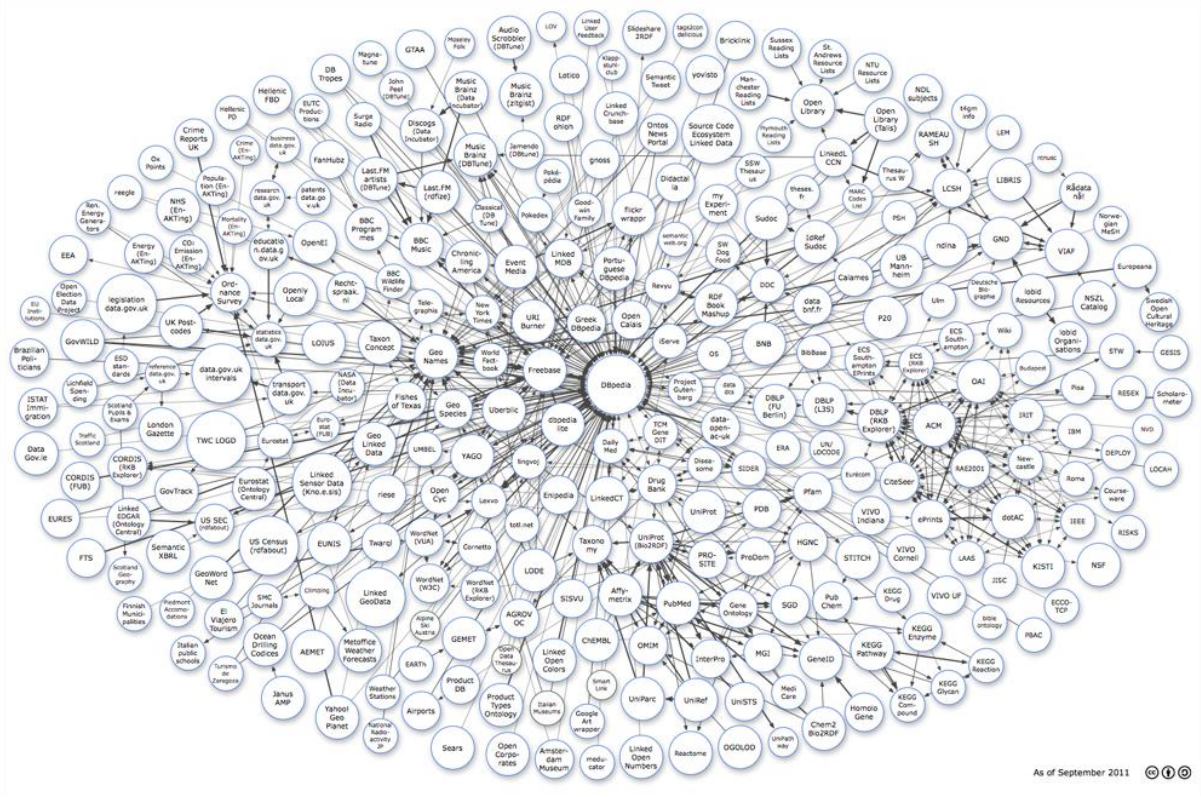


Figure 19. Interconnexion des données structurées (Juillet 2009)


DBpedia
Freebase
Geo Names
Uberblic
YAGO
LinkedCT
UniProt
PubMed
Gene Ontology
SGD
KEGG Pathway
KEGG Enzyme
KEGG Drug
KEGG Glycan
KEGG Reaction
KEGG Compound
GeneID
MGI
InterPro
OMIM
ChEMBL
AGROV OC
SISVU
Affy-metrix
Drug Bank
Eurostat (FUB)
Eurostat (Ontology Central)
US Census (rdfabout)
EUNIS
Twarql
Linked GeoData
Linked Sensor Data (Kno.e.sis)
Linked EDGAR (Ontology Central)
US SEC (rdfabout)
GovTrack
CORDIS (FUB)
CORDIS (RKB Explorer)
London Gazette
TWC LOGD
data.gov.uk intervals
transport data.gov.uk
statistics data.gov.uk
legislation data.gov.uk
UK Post-codes
Ord-nance Survey
Openly Local
LOIUS
Taxon Concept
Geo Species
Fishes of Texas
NASA (Data Incu-bator)
Geo Linked Data
UMBEL
Open Cyc
Lexvo
Wordnet (VUA)
Wordnet (W3C)
Wordnet (RKB Explorer)
Cornetto
lotd.net
LODE
Taxono my
UniProt (Bio2RDF)
Eurécom
CiteSeer
ACM
DBLP (RKB Explorer)
DBLP (L3S)
DBLP (FU Berlin)
SIDER
Disea-some
Daily Med
TCM Gene DIT
Data-open-ac-uk
Project Guten-berg
data dcs
Open Library
Open Library (Talis)
LCSH
LIBRIS
GND
VIAF
Sudoc
IdRef Sudoc
BNB
data bnf.fr
DDC
RDF Book Mashup
Revyu
Calames
P20
ndlna
UB Mann-heim
Deutsche Bio-graphie
Iobid Resources
Iobid Organi-sations
NSZL Catalog
Europeana
Swedish Open Cultural Heritage
Norwe-gian MeSH
Rådata nå!
LEM
RAMEAU SH
NDL subjects
NTU Resource Lists
St. Andrews Resource Lists
Sussex Reading Lists
Manchester Reading Lists
Plymouth Reading Lists
LinkedL CCN
Thesaurus W
MARC Codes List
PSH
Ulm
Wiki
ECS South-ampton
ECS South-ampton (Ephints)
ECS (RKB Explorer)
OAI
Budapest
Pisa
RESEX
Scholaro-meter
STW
GESIS
IRIT
IBM
NVD
DEPLOY
LOCAH
Roma
Course-ware
IEEE
RISKS
KISTI
NSF
JISC
ECCO-TCP
PBAC
bible ontology
VIVO UF
VIVO Cornell
LAAS
STITCH
HGNC
ProDom
PRO-SITE
Pfam
PDB
UniPath-way
UniSTS
UniRef
UniParc
Reactome
OGOLOD
Linked Open Numbers
Amster-dam Museum
medu-cator
Google Art wrapper
Italian Museums
Homolo Gene
Chem2 Bio2RDF
Medi Care
Pub Chem
VIVO Indiana
ePrints
dotAC
Newcastle
Ulrichs
Iconclass
Ordnance Survey
Geo-graphy
Linked Open Colors
Product Types Ontology
Product DB
Weather Stations
Airports
Sears
Open Corpo-rates
Yahoo! Geo Planet
Janus AMP
Ocean Drilling Codices
AEMET
Metoffice Weather Forecasts
GEMET
EARTh
Open Data Thesaurus
Smart Link
Linked Open Colors
Aspire Ski Austria
Cimixing
Linked GeoData
SMC Journals
El Viajero Tourism
Piedmont Accomo-dations
Finnish Munici-palities
Italian public schools
Turismo de Zaragoza
GeoWord Net
Scotland Geo-graphy
FTS
Semantic XBRL
EURES
Linked EDGAR (Ontology Central)
Scotland Pupils & Exams
Traffic Scotland
Data Gov.ie
ISTAT Immi-gration
Lichfield Spen-ding
Brazilian Poli-ticians
EU Institu-tions
GovWILD
ESD stan-dards
EEA
Open Election Data Project
Ren. Energy Genera-tors
reegle
Ox Points
Crime Reports UK
Hellenic FBD
Hellenic PD
DB Tropes
Magna-tune
GTAA
Audio Scrobbler (DBTune)
Moseley Folk
LOV
Linked User Feedback
Slideshare 2RDF
Bricklink
Semantic Tweet
yovisto
Lotico
Music Brainz (zitgist)
Music Brainz (Data Incubator)
Music Brainz (DBTune)
RDF onion
Linked Crunch-base
Ontos News Portal
Source Code Ecosystem Linked Data
Jamendo (DBTune)
gnoss
Poke-pédia
Didactal ia
SSW Thesaur us
my Experi-ment
Good-win Family
flickr wrappr
semantic web.org
SW Dog Food
Pokedex
Classical (DB Tune)
Last.FM (rdfize)
Last.FM artists (DBTune)
FanHubz
Surge Radio
Discogs (Data Incubator)
John Peel (DBTune)
EUTC Produc-tions
Hellenic PD
Crime (En-AKTing)
business data.gov.uk
Popula-tion (En-AKTing)
research data.gov.uk
patents data.go v.uk
BBC Program mes
BBC Music
BBC Wildlife Finder
BBC Tele-graphic
Chronic-ling America
Event Media
Linked MDB
Portu-guese DBpedia
New York Times
URI Burner
Greek DBpedia
Open Calais
World Fact-book
Geo Names
Freebase
DBpedia
iServe
OS
Ulan
Wiki

- la couche applicative (traitement) : c'est la partie où les logiciels sont développés sur la base de données.

Ces différentes couches sont représentées comme suit :

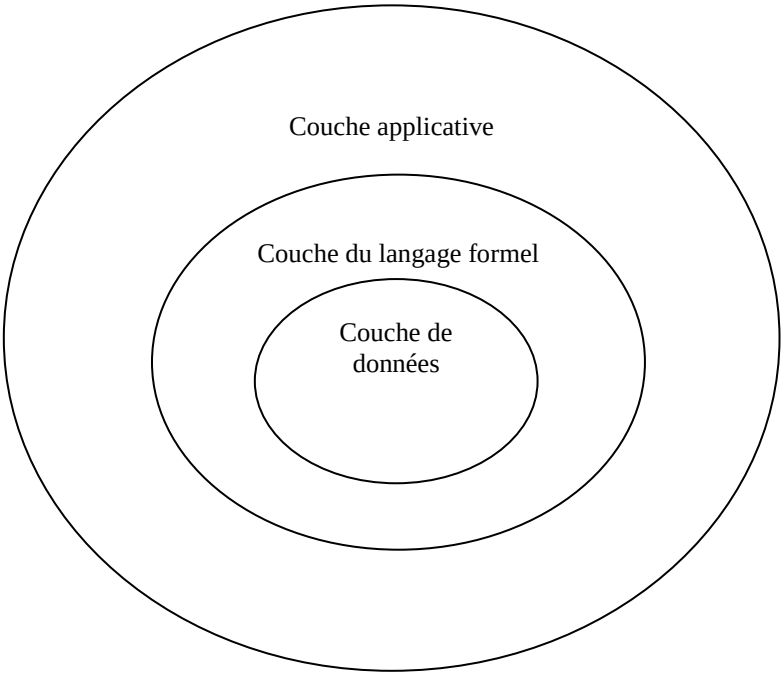


Figure 21. Différente partie d'une base de données

Pour un Système de Question-Réponse, seul le niveau applicatif change.

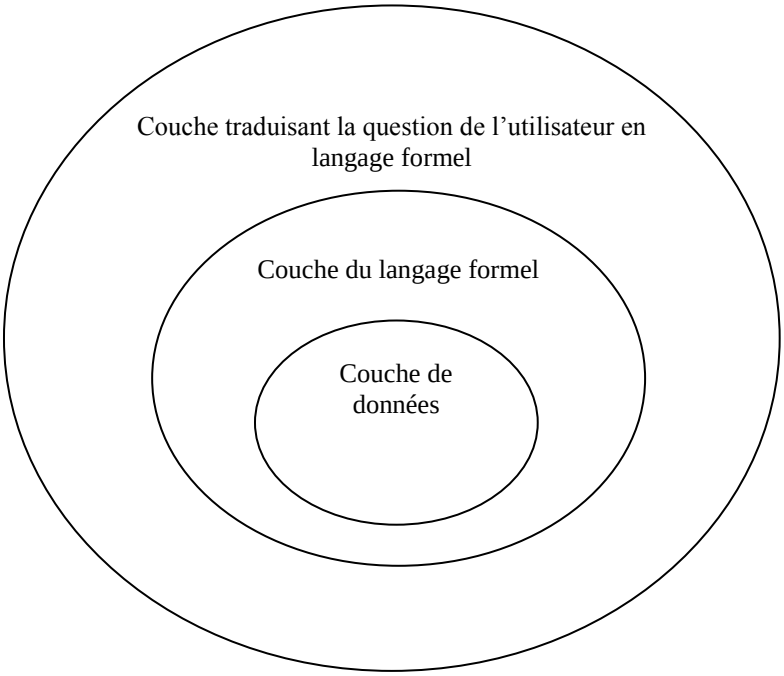


Figure 22. Système de Question-Réponse basé sur des données structurées

L'expressivité de la couche de données et du langage formel concourent chacun à faciliter le développement d'une interface en langage naturel. Par exemple plus la couche de données permet de dire plusieurs types de faits, plus il est facile de répondre à des questions concernant ces faits. Dans les paragraphes qui suivent, nous essayons de voir sommairement les différents modèles de données et ce qu'ils apportent pour une interface en langage naturel.

3.2.1 Expressivité des différents modèles de données

Le modèle de données a une influence sur la couverture des types de questions que le système peut supporter, la portabilité et la scalabilité (capacité de s'adapter à la montée en charge). Contrairement au langage naturel où il est possible de tout exprimer, les modèles de base de données ont une expressivité limitée mais non ambigüe. La flexibilité du schéma de données permet de s'adapter dans des environnements où les données et les sources changent constamment. Il existe plusieurs modèles de base de données : le modèle hiérarchique, le modèle réseaux, le modèle relationnel et le modèle objet. Récemment, de nouveaux besoins liés au web (volume considérable des données, données distribuées, montée en charge) ont permis l'émergence du modèle NoSQL et du NewSQL qui est une alternative à NoSQL pour les problèmes de sécurité et de cohérence de données (Rick, 2011). Ce dernier complète le modèle relationnel qui s'adapte mal aux problèmes des données Web. Les bases de données NoSQL sont premièrement utilisées par des groupes comme Google, Amazon, Facebook pour gérer des grands volumes de données. Dans cette section, nous nous intéressons principalement à l'expressivité des modèles de données. Pour une interface en langage naturel, l'expressivité du modèle de données est importante. Si le modèle de données ne permet pas de représenter un fait, il est difficile souvent de répondre à une question se rapportant à ce fait. Un modèle de données qui permet de représenter plus de faits, offre des moyens simples pour répondre à des questions se rapportant à ces faits. L'évolution des modèles de données est intéressante car elle est orientée vers la simplification des applications sur ces données ou une meilleure prise en compte de la nature des données. Ces aspects sont enrichissants également pour les interfaces en langage naturel pour les bases de données.

3.2.1.1 modèle hiérarchique

Le modèle hiérarchique utilise une structure arborescente pour stocker les données. En dehors de la racine de l'arbre chaque enregistrement est lié à un seul parent. C'est une structure de type 1-N. Un nœud a un seul parent et peut avoir plusieurs fils. Une relation de type 1-N est adaptée à des problèmes comme le système de fichiers où chaque répertoire peut contenir zéro ou plusieurs fichiers

ou répertoires mais chaque fichier ou répertoire a un seul parent. Le modèle hiérarchique utilise les concepts suivants :

- le *champ* : c'est la plus petite unité de données
- l'*article* (ou enregistrement) : c'est une collection de champs.
- l'*arbre* d'articles (ou d'enregistrement) : collection d'articles reliés par des associations père-enfants organisées sous forme d'une hiérarchie
- le *lien* : un lien se traduit par une relation père-fils. Cette relation assure la structure arborescente du modèle.

Du fait que le modèle hiérarchique prend en charge uniquement les relations de type 1-N, répondre à une question qui sort du cadre de cette relation n'est pas évidente. Le modèle hiérarchique le plus répandu de nos jours est sous le format XML.

3.2.1.2 modèle réseau

Le modèle réseau est une extension du modèle hiérarchique en supprimant la limitation du nombre de liens entre les nœuds du réseau. Les liens sont donc de type N-M. Une relation de type N-M permet de représenter des relations où les occurrences des entités participantes peuvent être plusieurs des deux cotés de la relation. Exemple la relation « inscrire » entre l'entité « cours » et « étudiant » est de type N-M. Un étudiant peut s'inscrire dans plusieurs cours et un cours peut avoir plusieurs étudiants. Le modèle réseau utilise le concept d'*agrégat*, d'*article* et de *lien*. Un agrégat est une liste de valeurs pour un même champ. Il est possible donc de stocker plusieurs valeurs dans le même champ. Il est possible par exemple de stocker tous les prénoms d'une personne dans le champ prénom. Ceci est susceptible d'augmenter le rappel d'un système d'interrogation. Les articles de mêmes types sont appelés des *ensembles*.

3.2.1.2 modèle relationnel

Le modèle relationnel a été formulé et proposé par (Codd, 1970). Il est encore actuellement le modèle le plus répandu. Les principaux concepts utilisés pour la couche de données sont :

- les relations qui sont aussi appelées des tables ;
- les attributs qui sont les caractéristiques des entités du modèle ; un attribut est représenté par une colonne sur la table ; le domaine d'un attribut est l'ensemble des valeurs qu'il peut prendre.

Le modèle relationnel dispose d'un fondement mathématique solide (l'algèbre relationnel) et simple qui a permis l'élaboration d'un langage de requêtes formelles SQL (Structured Query Language). Par

rapport aux modèles hiérarchiques et réseau, le modèle relationnel est plus orienté vers la cohérence des données et l'indépendance des applications des données. Un attribut a une seule valeur dans le modèle relationnel.

3.2.1.3 modèle objet

Le modèle objet vise à traiter des données complexes (données ayant moins de régularités) que le modèle relationnel ne peut gérer de manière optimale. Une caractéristique importante du modèle objet par rapport aux modèles précédents est que les traitements qui sont appliquées aux données ne sont pas dissociés des données. Pour un système de Question-Réponse, il est important de connaître les traitements applicables à la donnée car ces traitements peuvent intervenir dans une question. Une question ne consiste pas toujours à extraire une valeur qui est stockée dans la base de données. Il peut s'agir aussi d'effectuer des traitements sur la donnée extraite. L'intérêt d'une base de données, c'est aussi la possibilité d'effectuer des traitements sur ces données. Dans ces conditions, il est intéressant de savoir quels traitements sont applicables à la donnée. La séparation des données et des traitements comme dans le modèle relationnel ne facilite pas la reconnaissance des traitements applicables à la donnée. Le plus souvent le concepteur de l'interface en langage naturel se concentre sur le schéma des données qui est directement visible. Les principaux concepts utilisés par le modèle objets sont :

- l'objet (ou classe) : qui représente l'entité à modéliser
- les attributs : qui caractérisent l'objet
- les méthodes : les traitements applicables aux attributs.
- abstraction : permet de regrouper les objets selon des caractéristiques communes. Ce regroupement traduit la relation sémantique entre les objets tels que décrite par le monde réel alors que les liens entre les entités des modèles précédents sont plutôt guidés par le besoin de l'application. L'abstraction est un concept proche du langage naturel dans la mesure où elle décrit le monde réel.

3.2.1.4 modèle orienté graphe

Le modèle orienté graphe (du Web Sémantique) utilise la théorie des graphes pour stocker les données. Il est capable de stocker des données complexes (listes ou arbres). Le modèle hiérarchique et réseau sont également de modèle graphe. Les modèles de graphes les plus utilisés dans le web sont ceux du web sémantique basé sur RDF (Resource Description Framework). Les principaux concepts utilisés par RDF sont :

- le *triplet* : c'est un fait dans la base de données. Il est composé du sujet, du prédicat et de l'objet,
- le *sujet* : correspond à une classe de l'application,
- le *prédicat* : est une propriété d'une classe,
- l'*objet* : est soit une classe ou un littéral (valeur d'une propriété).

Les prédicats sont de deux types : prédicat de type objet (*object property*) et prédicat de type donnée (*datatype property*). Un prédicat de type objet relie deux objets (ou classes) de l'ontologie. Un prédicat de type donnée relie un objet à une donnée. Par exemple :

```
<François Hollande>  <préside> <France>

<François Hollande>  <date_Investiture> <15 Mai 2012>
```

Le premier triplet indique que *François Hollande* est président de la *France* (en supposant que *François Hollande* et *France* sont 2 instances de classes de l'application). Il est possible d'étendre le graphe en rattachant d'autres informations de la France comme la superficie, la population par exemple. Le prédicat du second triplet est de type donnée. Il indique la date d'investiture. Il n'est pas possible de rattacher une autre information sur la date.

RDFS est une extension de RDF qui donne plus de descriptions sur les classes et les propriétés. OWL[17] (Web Ontology Language) est basé sur RDF mais plus expressif. Il permet de tenir compte de la conjonction, la disjonction, négation, le complément, l'inclusion, la symétrie, la transitivité, l'équivalence et les classes ou propriétés disjointes. Les modèles précédents ne supportent pas ces caractéristiques (modèle relationnel, hiérarchique ou réseau). Le modèle objet supporte l'inclusion à travers l'héritage des classes. OWL est un langage de représentation de connaissances. Il est également plus expressif que les autres mais pourrait être amélioré (Frost et al., 2014) pour s'approcher plus du langage naturel.

3.2.1.5 modèles NoSQL

Les modèles NoSQL et NewSQL (Rick, 2011) ont été introduits récemment. Ils n'apportent pas une expressivité particulière. NoSQL relâche les contraintes d'intégrité du modèle relationnel pour des raisons de performance pour répondre à la montée en charge. Il est apparu dans l'environnement web avec les grands acteurs comme Google, Facebook ou Amazon. Une particularité du NoSQL pour les interfaces en langage naturel est l'absence de schéma prédéfini pour stocker les données. Le schéma

[17] http://www.w3.org/TR/2012/REC-owl2-primer-20121211/

des données est défini plus tard par l'application qui traite les données (Bruchez, 2013). Dans l'approche NoSQL, les données sont d'abord enregistrées et un schéma est défini par rapport à ces données. Dans l'approche SQL traditionnel, le schéma est d'abord défini et les données sont enregistrées après. La définition d'un schéma par avance suppose que le concepteur connaît le problème ce qui n'est pas le cas dans le Web où les données sont instables et leur évolution n'est pas parfaitement connue. La flexibilité du schéma est un problème pour une interface en langage naturel. Le lexique d'une interface en langage naturel est souvent défini manuellement à partir du schéma pour couvrir le plus de questions possibles. La variation du schéma peut faire en sorte que le lexique n'est pas adapté.
NewSQL est une alternative à NoSQL pour prendre en compte les fonctionnalités du modèle dans un environnement distribué de charge importante comme NoSQL. La cohérence et la sécurité des données sont prises en compte.

Les modèles de données ont évolués de la pure représentation de la logique du métier à des représentations qui tiennent compte de plus en plus du monde réel. Des connaissances supplémentaires sur les concepts tels qu'ils sont dans le monde réel sont ajoutées. La notion de prédicat introduite par les modèles basés sur RDF permet d'exprimer les faits d'une manière relativement proche du langage naturel. Elle couvre aussi les concepts utilisés dans les autres modèles. OWL a relativement un haut niveau de description. Une interface en langage naturel basée sur ce modèle à l'avantage de bénéficier de cette expressivité pour couvrir plus de questions que l'utilisateur est susceptible de poser. De ce fait le schéma lexical est basé sur RDF. Nous retenons aussi le fait de ne pas dissocier les données du traitement du modèle objet.

3.2.2 Expressivité du langage formel

Le langage d'interrogation formel comme SQL, SPARQL permet la médiation entre le langage naturel et les données. En général, le langage formel est lié au format des données. Mais pour des raisons d'expressivité, il peut être étendu. Il ne fournit donc pas tous les outils qui permettent de traduire directement la question de l'utilisateur. Par exemple SQL standard offre des opérateurs génériques comme l'opérateur de comparaison qui permet de comparer des nombres comme la population des villes mais n'offre pas directement des opérateurs qui permettent de dire que telle ville *est au nord* de telle ville par exemple. En langage naturel l'expression « *au nord* » est une relation simple qui peut exister entre deux entités d'une application mais dans le langage formel utilisé il n'est pas évident de trouver des opérateurs correspondants. Ceci est fréquent dans la plupart des applications de traduction automatique d'une langue à une autre. Le Système de Question-

Réponse doit s'attendre à gérer des situations pareilles. Des extensions sont souvent proposées pour palier ces insuffisances comme par exemple GeoSPARQL[18] pour le langage SPARQL. Ce langage permet de manipuler des entités géographiques. Le concepteur de l'interface en langage naturel doit avoir donc à l'esprit les limites du langage formel.

3.3 Type de questions traitées

Plusieurs classifications de questions ont été proposées (Li et Roth, 2002 ; Metzler et Croft, 2005) :

- abréviation,
- entité : une entité fait référence généralement aux entités nommées. Il est possible d'avoir une classification plus détaillée,
- description : la question peut être de type explication ou raisonnement.

Cette classification est issue des systèmes de Question-Réponse qui utilisent une base de données textuelle. La particularité des bases de données textuelles est qu'il n'est pas possible de faire des agrégations. Comme les agrégations sont possibles dans une base de données classique, les types de question que nous visons sont de type entité mais qui peuvent contenir des calculs sur les données.

Un langage d'interrogation comme SQL n'est pas destinée à raisonner sur les données. Il permet seulement de faire des calculs ou extraire des données. De ce fait le raisonnement ne peut pas être pris en charge directement. Mais il est possible de faire une autre couche qui fait ce travail de description à partir des résultats extraits.

3.4 Décomposition de domaine

L'objectif de la décomposition est de permettre au système de s'adapter à la diversité des domaines et au grand volume de données. La résolution du problème de portabilité qui se transforme de plus en plus en problème d'interopérabilité et de scalabilité (possibilité de s'adapter à la montée en charge des données) a été explorée de plusieurs manières. La plupart de ces approches distinguent un lexique dépendant du domaine et un autre lié au domaine. Pour PRECISE (Popescu et al. 2004) et AquaLog (Lopez et al., 2005), la question de la portabilité est traitée en générant automatiquement le lexique du domaine à partir des ressources sémantiques comme Wordnet. (Cimiano et al., 2007; Lopez et al., 2013) estiment qu'une construction automatique des ressources lexicales à partir des ressources sémantiques telle que Wordnet ne permet pas une couverture satisfaisante pour l'application. (Cimiano et al., 2008) proposent une construction manuelle du lexique faisant intervenir un utilisateur non nécessairement familier de l'informatique contrairement à TEAM (Grosz et al., 1987). Mais le lexique construit ne concerne que le lexique dépendant du domaine.

[18] http://www.opengeospatial.org/standards/geosparql

Plusieurs modèles ont été proposés pour améliorer l'interface ontologie-lexiques : LingInfo (Buitelaar et al., 2006), LexOnto (Cimiano et al., 2007), LexInfo (Cimiano et al., 2011), Lemon (Buitelaar et al., 2011) et Ontolex[19] qui est en cours de spécification au niveau de l'organisme W3C depuis 2012 et qui pourrait être standardisé ultérieurement. Une caractéristique de tous ces modèles est qu'ils s'intéressent à comment les éléments de l'ontologie sont lexicalisés dans le langage naturel. Dans le cadre d'un Système de Question-Réponse, ceci permet de lier les unités lexicales de la questions aux concepts de la base de données. Ontolex a l'ambition de définir un standard pour certaines applications du Traitement Automatique des Langues (dont les systèmes de Question-Réponse) dans le contexte du Web de données (Linked Data). Pour un Système de Question-Réponse, le niveau ontologie correspond à la couche de données (voire figure 22). La couche du langage formel et la manière dont la langue manipule les données ne sont pas couvertes.

(Rodolfo et al., 2011) soutient que, pour mettre en œuvre un processus de traduction réussie d'une question de l'utilisateur dans un langage de base de données, il est nécessaire de concevoir une architecture pour le processus de traduction en appliquant une autre technique de conception utilisée dans les systèmes complexes: une architecture basée sur les fonctionnalités couches, similaires au modèle OSI[20] (Open Systems Interconnection model). Le modèle OSI a permis le développement de l'Internet.

Cette notion de couche est intéressante pour une architecture stable face à la montée en charge du volume de données, la diversité des données et la volonté de donner accès à ces données à un plus grand nombre d'utilisateurs. Les approches utilisées jusqu'à la fin de la décennie 2000 pour concevoir des interfaces en langage naturel commencent à montrer leurs limites même si elles ont un intérêt méthodologique. Dans notre approche, nous adaptons une démarche qui concilie les deux aspects précédents : lexicalisation (comme pour les interfaces ontologie-lexique) et approche par couche. L'aspect couche de cette approche utilise la figure 22. Comme modèle de données, nous nous basons sur RDF[21] du fait de sa relative expressivité.

3.4.1 Hiérarchie des classes du lexique centrée sur les triplets

La construction du lexique se base sur les triplets. Nous construisons le lexique autour du triplet car il offre un moyen de désambigüisation. Le lexique autour du triplet constitue également des micro-

[19] http://www.w3.org/community/ontolex/wiki/Main_Page

[20] http://en.wikipedia.org/wiki/OSI_model

[21] http://www.w3.org/RDF/

lexiques susceptibles d'être portables. Ceci est important dans le cas où on travaille sur un sous ensemble des données. L'application pourra charger uniquement les micro-lexiques nécessaires (associés aux triplets concernés) au lieu de tout le lexique. Du fait que certains triplets se répètent dans plusieurs domaines, le lexique pourrait être quasi-stable à un certain moment indépendamment du nombre des jeux de données.

Les classes du lexique ne sont pas nécessairement les classes de l'ontologie. Cette dernière ne contient que les entités du domaine stockées dans la base de données. Dans une question il n'y a pas que les concepts du domaine. Il y a donc d'autres concepts qui ne sont pas nécessairement stockés dans la base de données. Les classes du lexique couvrent aussi bien les concepts de l'ontologie que ces concepts qui ne relèvent pas de l'ontologie.

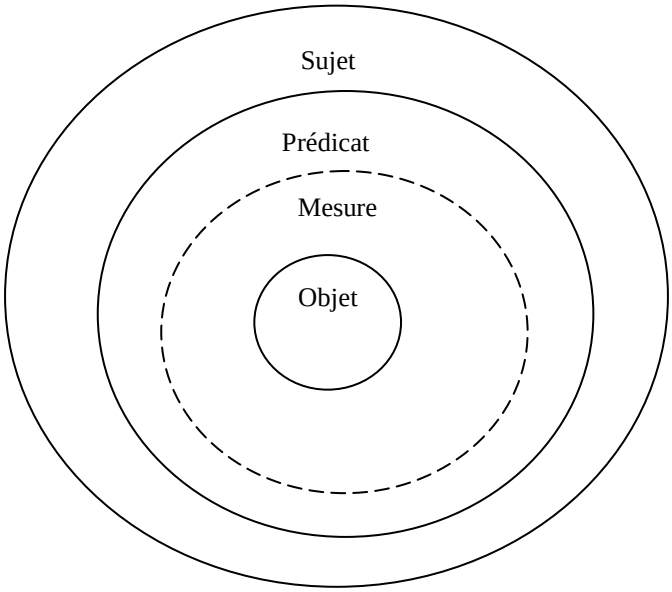


Figure 23. Hiérarchies des classes

La figure 23 montre le recouvrement du lexique. Chaque niveau est une classe. Le niveau objet est le niveau le plus bas. Il contient les types de bases : numérique, caractères, binaire, booléen et géométrique. Le sens d'un mot ou expression d'un niveau n'évoque que ces types sans aucune indication du domaine ou de la mesure. Pour les nombres par exemple, il s'agit des valeurs « un, deux, trois, etc. » ou bien des adjectifs comme « grand, petit, peu, etc. » qui indique un nombre sans une précision sur la dimension quantifiée. L'adjectif « long » n'appartient pas au niveau *objet* car son sens inclut déjà la dimension mesurée qui peut être la distance ou le temps. Le niveau mesure est optionnel. Il contient des mots ou des expressions qui se rapportent à une mesure. Le niveau objet est inclus dans le niveau mesure. A partir de la classe mesure, le lexique d'une classe peut-être classé en 2 parties :

- le lexique qui se réfère intrinsèque à la classe principale et
- le lexique se chevauchant entre les classes (incluses dans la classe principale à l'image de la figure 23).

Par exemple le mot « poids » se réfère exclusivement à la mesure alors le mot « lourd » se réfère à la fois à la mesure et certaines valeurs dans l'échelle de mesure (classe objet). Cette distinction est importante si le système doit calculer la valeur correspondante. Un exemple au niveau de la classe prédicat peut-être illustré avec le triplet suivant :

<zone, population, entier>

Ce triplet permet d'exprimer pour chaque zone sa population. Les questions suivantes peuvent être posées :

Quelle est la population de chaque région ? (1)

Quelles sont les zones peuplées ?(2)

Dans la première question « population » ne fait aucune restriction sur le nombre d'habitants. Le système doit afficher toute la population de chaque zone sans restriction. Donc « *population* » est propre au prédicat. Dans la seconde question, le système ne devrait pas afficher les zones ayant zéro habitant car le mot « *peuplées* » impose une restriction sur le nombre d'habitants. « Peuplées » chevauche la classe de prédicat et la classe objet qui définit l'échelle des valeurs du prédicat. Le prédicat doit prendre des valeurs dans cette échelle. Le lexique qui appartient intrinsèquement à une classe est appelé *lexique intraclasse*, celui qui chevauche les classes est appelé *lexique interclasse* (Soumana et al., 2013).

Dans le lexique interclasses, les valeurs peuvent être floues ou exactes sur des dimensions qui ne sont pas nécessairement des scalaires. La logique floue a déjà été utilisée dans certains systèmes (Cardey et al., 2001), (Cimiano et al., 2008) (Kaufmann et al., 2006). La logique floue est appliquée dans les interfaces en langage naturel sur des dimensions scalaires notamment les quantificateurs. Dans l'approche proposée, il s'agit de prendre en considération tous les mots et les expressions qui peuvent avoir un sens calculable selon le contexte et la base de données. Ceci permet d'élargir la correspondance entre les mots ou expressions du langage naturel et les concepts de la base de données. Dans une base de données la description des concepts est formelle. Une définition composée est faite à partir des primitives. La définition est tractable. En langage naturel, une définition composée (une série de conditions ou d'instructions en langage formel) peut être exprimée

en un seul mot ou expression sans qu'il y ait de relation détectable automatiquement par la machine. Les propriétés (par rapport à la base de données) nécessaires à la définition ne sont pas mentionnées dans l'expression ou la question.

L'ordre de constitution du lexique est le suivant : niveau objet, prédicat, mesure, puis sujet.

3.4.2 Le Lexique au niveau objet

Au lieu de générer le lexique automatiquement à partir des ressources comme Wordnet, nous examinons les différents types de données qui sont disponibles dans une base de données. Les propriétés de la donnée sont généralement liées à son type. La connaissance de ces propriétés permet d'interpréter les traitements qui sont applicables à la donnée. La figure 25 liste les différents types de bases

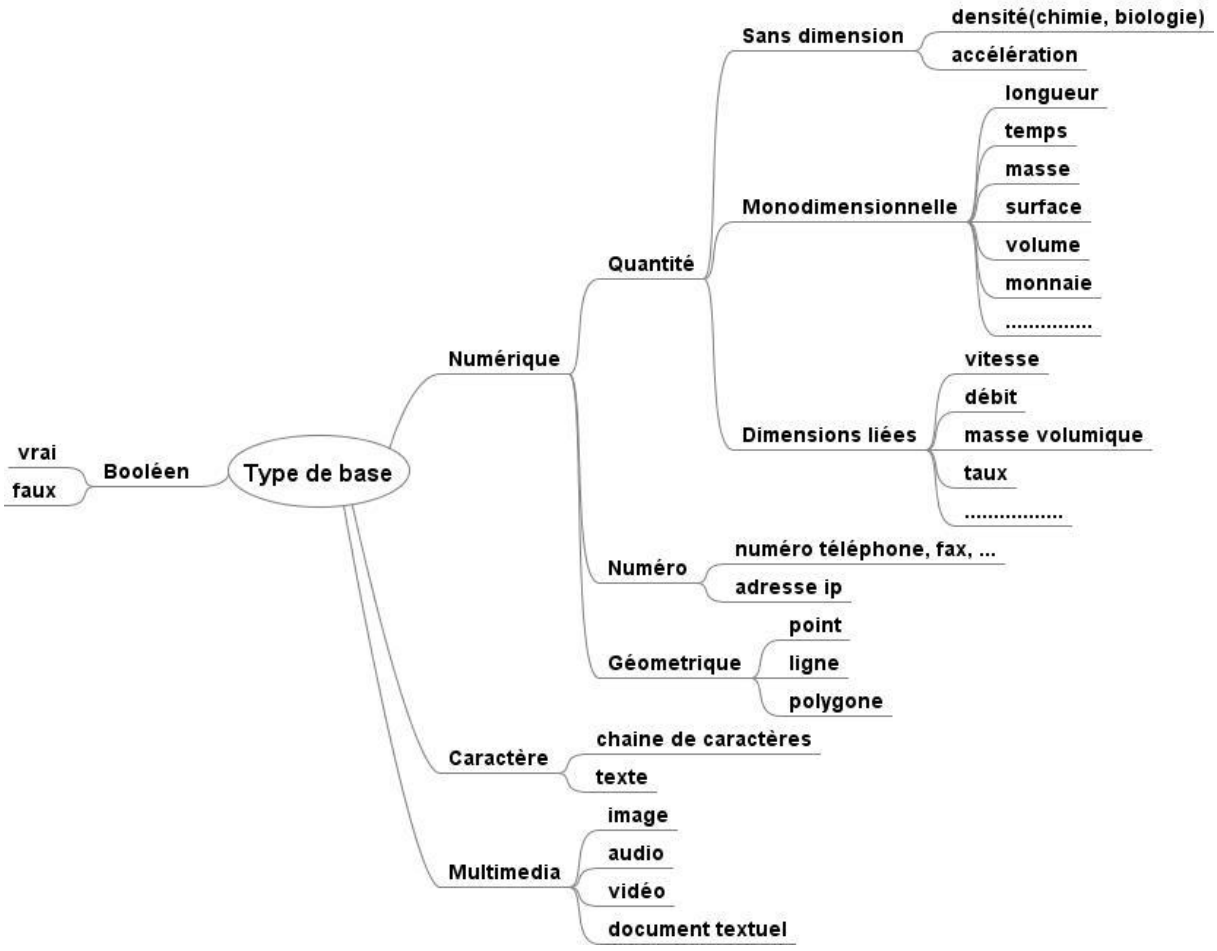


Figure 24. Répartition de types de base dans une base de données

Dans une base de données, il est possible de trouver les types suivants :

- numérique : ce type contient les données que l'homme interprète comme les nombres. Il s'agit principalement des données de mesures, des numéros ou des objets géométriques,

- caractères : ce sont les données textuelles,
- multimédia : les données multimédia concernent les images, l'audio, la vidéo et éventuellement des données textuelles. SQL ou SPARQL standard ne permettent pas de manipuler, d'interroger des données multimédia. Mais ces données peuvent être interrogées en utilisant des langages comme MPEG[22] Query Format,
- booléen : sont de type vrai ou faux.

L'interprétation des données est effectuée suivant leur type. Les paragraphes qui suivent montrent des exemples de construction du lexique pour les cardinaux (nombre exprimant la quantité sur la figure 25)

3.4.2 Exemple Lexique des cardinaux

A chaque niveau de chaque classe, nous distinguons les concepts suivants : variable, instance des variables, opérations et éventuellement des fonctions qui s'appliquent aux données. Ces concepts sont identifiés selon leur catégorie grammaticale.

3.4.2.1 Les variables

La référence à une valeur peut-être directe ou indirecte. Une référence directe évoque clairement la valeur.

Les cardinaux peuvent être classés en trois groupes selon la connaissance exacte de la valeur :

- une classe ayant des valeurs déterminées,
- une classe de valeurs approximatives,
- une classe des valeurs indéterminées.

La classe des valeurs déterminées regroupe les mots ayant des valeurs précises ou des valeurs dont la fourchette est bien connue comme 10 ou une dizaine. La classe approximative regroupe des mots ayant des valeurs floues comme grand, immense. La logique floue peut être utilisée pour déterminer de manière précise la valeur. Il est également intéressant dans les Systèmes de Question-Réponse de demander l'avis de l'utilisateur pour déterminer la valeur floue.

Les valeurs indéterminées font références à des valeurs qui sont intimement liées au contexte, à l'expérience de l'utilisateur. Il est difficile de donner une valeur consensuelle. Les valeurs indéterminées peuvent désigner en général les 2 extrêmes de l'échelle. Il est également intéressant d'interagir avec l'utilisateur pour avoir des données complémentaires. Les mots peuvent avoir également des modificateurs de la quantité qui marquent l'intensité. Le tableau suivant liste les instances avec les catégories grammaticales.

[22] http://mpeg.chiariglione.org/standards/mpeg-7/query-format

Tableau 3. Classification des variables des cardinaux

Nom	**Verbe**	**Adverbe**	**Determinant**
valeur	compter	Combien	quel*
nombre	dénombrer		
chiffre	nombrer		
donnée	évaluer		
	s'élever		
	calculer		
	estimer		
	énumérer		

3.4.2.2 Les instances

La référence à une valeur peut-être directe ou indirecte. Une référence directe évoque clairement la valeur.

Les cardinaux peuvent être classés en trois groupes selon la connaissance exacte de la valeur :

- une classe ayant des valeurs déterminées,
- une classe de valeurs approximatives,
- une classe des valeurs indéterminées.

La classe des valeurs déterminées regroupe les mots ayant des valeurs précises ou des valeurs dont la fourchette est bien connue comme 10 ou une dizaine. La classe approximative regroupe des mots ayant des valeurs floues comme grand, immense. La logique floue peut être utilisée pour déterminer de manière précise la valeur. Il est également intéressant dans les Systèmes de Question-Réponse de demander l'avis de l'utilisateur pour déterminer la valeur floue.

Les valeurs indéterminées font références à des valeurs qui sont intimement liées au contexte, à l'expérience de l'utilisateur. Il est difficile de donner une valeur consensuelle. Les valeurs indéterminées peuvent désigner en général les 2 extrêmes de l'échelle. Il est également intéressant d'interagir avec l'utilisateur pour avoir des données complémentaires. Les mots peuvent avoir également des modificateurs de la quantité qui marquent l'intensité. Le tableau suivant liste les instances avec les catégories grammaticales.

Tableau 4. Exemple de vocabulaire d'instanciation des cardinaux

EXACTITUDE	QUANTITE					
	Nom	adjectif	adverbe	déterminant	verbe	pronom
DETERMINEE	Mille million milliard couple paire singleton duo trio unité	unique seule singulier	uniquement seulement singulièrement	Nombre (0, 1, 2,3 etc.) aucun nul		rien personne
FORTE	multitude	énorme important fort considérable excessif supérieur démesuré imposant excellent meilleur	énormément fortement considérablement excessivement excellemment	tous	exceller	
MOYEN		moyen	assez de suffisamment			
FAIBLE		rare	rarement souvent peu			
NEUTRE		divers maint pluriel		déterminant marque du nombre : singulier ou pluriel		
INDETERMINEE		terrible étrange				

3.4.2.3 Les opérateurs

3.4.2.3.1es opérateurs de génération d'intervalle

Les opérateurs de génération servent à générer un ensemble de valeurs généralement sous forme d'intervalle. Nous nous basons sur les intervalles utilisés en mathématique. En langage naturel, les intervalles ne sont pas exprimés de manière aussi précise qu'en mathématique. L'appartenance des

bornes de l'intervalle n'est souvent pas explicite. L'appartenance des bornes peut être explicitement mentionnée par des modificateurs comme « inclus, exclus ». Par exemple du 5 au 10 *inclus*.

Tableau 5. Opérateurs de génération des intervalles

N°	Notation	Verbe	préposition	adjectif	Nom	adverbe	Symbole
1	[a ; b]		entre combiné avec « et » à au Exemple : 5 à10 ; 5 au 10			voire	- 5-10
5	[a ; + ∞ [		à partir de dès dès 5				
6	] a ; + ∞ [	dépasser	plus de supérieur à				
7	] – ∞ ; a]		jusqu'à				
8	] – ∞ ; a [		moins de inférieur à				
9	a ± Δ	avoisiner approcher frôler	autour de aux alentour de	proche de	voisin de	environ approxima-tivement	

Les opérateurs « à » et « au » peuvent être respectivement précédés de la préposition « de » et « du ». L'analyse de l'expression qui dénote l'intervalle s'étend éventuellement au groupe nominal qu'il détermine ou au verbe de la phrase pour s'assurer qu'il s'agit réellement d'un intervalle. Dans la phrase suivante :

Le Barça est battu 2-0 à Milan

Le Barça est battu 2 ***à*** *0 à Milan*

Les opérateurs ne génèrent pas un intervalle.

L'opération «-» peut-être également l'opérateur de soustraction dans certains contextes.

.

3.4.2.3.2 Les opérateurs de proportion

Les opérateurs de comparaison (=, >,<,<=, >=,≠) sont binaires. Le résultat de l'opération est de type booléen.

Tableau 6. Opérateurs de proportion

Nom	**verbe**	**adjectif**	**préposition**	**adverbe**	**symbole**
double triple quintuple centuple décuple multiple quadruple fraction	doubler tripler quintupler centupler décupler multiplier quadrupler fractionner	double triple quintuple centuple décuple multiple quadruple divisible	sur parmi des pour cent	doublement	% /

3.4.2.3.3 Les opérateurs de comparaison

Les opérateurs de comparaison (=, >,<,<=, >=,≠) sont binaires. Le résultat de l'opération est de type booléen.

Tableau 7. Opérateurs de comparaison

Opérateur	**Nom**	**Verbe**	**adjectif**	**préposition**	**adverbe**	**conjonction**
=	Égalité	égaler correspondre équivaloir égaliser équilibrer contrebalancer valoir	égale même identique pareil similaire équivalent	comme	exactement précisément	ainsi que

3.4.2.3.4 Les opérateurs logiques

Tableau 8. Opérateurs de logique

Opérateur	**Conjunction (coordination et subordination**	**Préposition**	**Adverbe**
Conjunction	Et aussi bien que	puis avec	
Disjunction	Ou Soit		autrement ou bien ou bien même
Negation		sauf excepté hormis à l'exception de à l'exclusion de abstraction faite de au lieu de	Pas

3.4.2.3.5 Opérations arithmétiques

Tableau 9. Opérations arithmétiques

Opération	**Verbe**	**nom**	**adjectif**	**adverbe**	**Déterminant**	**Symbole**
addition	additionner ajouter totaliser sommer	addition ajout totalité somme	total entier	en tout en tout et pour tout plus	tout	+
soustraction	soustraire ôter diminuer	soustraction diminution		moins		-
multiplication	Multiplier	multiplication fois				X
division	diviser subdiviser	division				÷, /

3.5 Conclusion

L'accroissement du volume de données structurées et la diversité des domaines a suscité de nouveaux défis dès la fin de la décennie 2000. Les approches utilisées pour l'interrogation des bases de données cloisonnées échouent à l'échelle du Web parce que les utilisateurs n'ont pas une compréhension apriori de tous les ensembles de données disponibles (Freitas et al., 2012).

Pour concevoir des systèmes robustes, nous avons orienté la démarche vers une lexicalisation qui prend en compte non seulement l'ontologie mais aussi les différentes couches du système. Pour cela nous proposons un *domaine abstrait (en général les mesures)* et *un domaine concret.* Le domaine concret couvre le domaine métier de l'application tandis que le domaine abstrait s'intéresse à la logique qui n'est pas liée nécessairement au domaine métier (comment les données sont manipulées hors du contexte d'une base de données). Alors que le domaine métier est largement pris en compte, le domaine abstrait ne fait généralement pas l'objet d'une large investigation. Il est important de couvrir ce domaine pour concevoir des systèmes robustes.

La constitution du lexique est effectuée à l'échelle des triplets. Pour gérer la variation du langage naturel et la manière dont le langage se réfère à ces domaines, nous avons défini deux types de lexique *: intraclasses* et *interclasse*. Le niveau *intraclasse* est une relation bijective entre un mot ou une unité lexicale du langage naturel vers un concept du domaine abstrait ou concret (classe, propriété, valeur, opérateur, etc.). Le niveau *interclasse* correspond au cas où une correspondance bijective n'est pas suffisante. Le mot ou l'unité lexicale en langage naturel correspond à une combinaison souvent complexe des classes, propriétés, valeurs, opérateurs qui ne sont pas tous nécessairement disponibles dans l'ontologie. Dans la majorité des systèmes, le lexique est constitué principalement du niveau *intraclasse.*

L'objectif est de réduire le fossé sémantique entre la question de l'utilisateur et la base de données. Cette approche demande cependant une construction manuelle du lexique. La représentation formelle du lexique n'a pas été abordée. Elle peut s'appuyer sur un langage de description plus expressif.

CHAPITRE 4 : LA MESURE DU TEMPS

4.1 Introduction

Dans ce chapitre nous abordons le traitement du temps qui représente un exemple de la prise en compte du domaine abstrait. L'objectif est à la fois de parvenir à un traitement plus flexible des expressions temporelles et d'avoir une méthode susceptible de s'adapter aux autres types de mesures. La prise en compte de la dimension temporelle est très importante dans une question. Dans un premier temps nous explorons le traitement du temps dans la littérature. Beaucoup de progrès ont été réalisés mais la recherche du traitement du temps est toujours d'actualité. Dans le cadre de l'analyse d'une question nous adoptons une approche compositionelle basée sur les propriétés des données temporelles et de la logique métier.

La mesure du temps intervient dans plusieurs applications (Système de Question Réponse, Résumé automatique, Extraction d'information). Le traitement de l'information temporelle est lié à la nature des objectifs poursuivis par l'application. L'information temporelle peut être traitée en plusieurs étapes (Kevers, 2011) :

- La première étape (ou reconnaissance) consiste à repérer et à délimiter à l'aide d'un marquage les expressions temporelles dans le texte.
- La seconde étape (ou normalisation) vise à interpréter cette expression selon un format normalisé pour lui associer une valeur temporelle selon un repère choisi (le calendrier par exemple).
- La troisième étape lie l'expression temporelle à un événement.
- La dernière étape s'intéresse à l'organisation chronologique des différents événements du texte.

Ces étapes ont été progressivement définies au cours des campagnes d'évaluations. Le traitement des expressions temporelles (telles que désigné par TIMEX) a été introduit pour la première fois lors des conférences MUC[23] (Message Understanding Conference), précisément au MUC-6 en 1995 en tant qu'une partie des entités nommées (Personne, Organisation, Lieu, Pourcentage, Monnaie) (Nadeau et Sekine, 2007). Les expressions temporelles identifiées dans les campagnes MUC sont limitées et fournissent peu d'information. Il s'agit principalement des dates et heures. A la fin des campagnes

[23] http://www-nlpir.nist.gov/related_projects/muc/

MUC, les activités de reconnaissance et de normalisation se sont donc poursuivies. Ces activités sont centrées surtout sur les formats d'annotation. On distingue le format standard timeML[24] (Ferro et al.2005), (Pustejovsky et al., 2003) et des formats ad-hoc.

4.2 TimeML un format émergeant

TimeML a été conçu dans le cadre du projet AQUAINT(ARDA's Advanced Question Answering for Intelligence). Ce projet vise à développer des Systèmes de Questions-Réponses pour le besoin des professionnels du renseignement ou de l'information sur des données de formats divers (texte, audio par exemple), de langues et de genres. Pour les données textuelles, ces systèmes sont basés sur des données non structurées c'est-à-dire que la réponse à la question de l'utilisateur doit être recherchée dans un texte brut et non dans une base de données (formelle). Dans ce cadre, TimeML a été élaboré comme langage d'annotation pour faciliter le raisonnement et l'inférence sur le temps. C'est le format dominant au cours de ces dernières années. Il a été modifié et standardisé par ISO en ISO-timeML[25] pour s'adapter à plusieurs langues au-delà de l'anglais. Des annotations en français ont été proposées (Bittar, 2008).

Initialement développé sur un corpus journalistique, il a été étendu à d'autres domaines comme la santé (Pustejovsky and Stubbs, 2012) pour analyser des dossiers médicaux personnels. Le projet THYME (Temporal Histories of Your Medical Events) s'inscrit dans cet objectif. Les modifications concernent par exemple les états des patients qui sont considérés comme des marqueurs temporels par l'annotation TimeML alors qu'ils devraient être considères comme des événements.

Au-delà des conférences MUC et des séries de workshops autour du TimeML, l'ACE[26](Automatic Content Extraction) s'est également intéressé au traitement de l'information temporelle. A partir de 2004, il lance la campagne d'évaluation TERN (Temporal Expression Recognition and Normalization). Les activités de TERN visent à identifier une expression temporelle dans un texte et à le normaliser selon le standard ISO-8601 en utilisant l'annotation TIMEX2. Les campagnes d'évaluation ACE n'ont pas malheureusement fixé des nouveaux défis pour les expressions temporelles et donc l'intérêt de participer à cette tâche particulière a diminué (Kolomiyets, 2009).

Récemment, TempEval (Temporal Evaluation), organisé au sein des conférences SemEval (Semantic Evaluation) à partir de 2007, a servi de cadre pour le traitement de l'information temporelle.

[24] http://timeml.org

[25] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=37331

[26] http://www.itl.nist.gov/iad/mig/tests/ace/2004/

TempEval utilise l'annotation TimeML. L'objectif est d'avancer la recherche notamment dans des gros volumes de données et dans un cadre multilingue. Les systèmes participants utilisent des approches statistiques, linguistiques ou hybrides. Pour les approches linguistiques figurent HeidelTime (Strötgen et al., 2013), NavyTime (Chambers, 2013), SUTime (Chang and Manning, 2013). Quant à ClearTK (Bethard, 2013), et TIPSem (Llorens et al., 2010), ils sont basés sur l'apprentissage automatique. Pour les systèmes ayant des approches hybrides figurent FSS-TimEx (Zavarella and Tanev, 2013) et ManTIME (Filannino et al., 2013). Les conclusions de TempEval 2013 (UzZaman et al., 2013) orientent le traitement de l'information temporelle dans le contexte d'une application spécifique. Le traitement de l'information temporelle orienté application pourrait avancer la recherche. Ces auteurs se demandent également si les relations établies par TimeML sont appropriées pour l'annotation linguistique.

Pour les systèmes de Question-Réponse basés sur des données structurées, la diversité des jeux de données ne facilite pas cette tâche orientée application. Le but de notre travail est également de développer une approche générique qui permet d'aborder la diversité des jeux des données.

4.3 Formats ad-hoc

Certains auteurs utilisent des formats qui répondent au réel besoin de leur application. Sans être exhaustif, on peut citer (Weiser, 2010) qui traite les expressions temporelles qui expriment des horaires. Il estime que les inférences qui sont possibles à partir de l'annotation timeML ne sont pas nécessaires pour son projet d'identification des horaires des sites touristiques et cela pourrait alourdir les résultats. Pour ces mêmes raisons (Heintzelman et al., 2013) n'utilisent pas TimeML pour analyser les verbatims des patients. Pour (Le Parc-Lacayrelle *et al.*, 2007) la défragmentation de l'annotation et l'attribution des valeurs ne sont pas adaptées à leur application d'information géographique. L'application a besoin à un certain moment (pour une comparaison par exemple) des valeurs précises ce qui n'est pas toujours le cas pour TimeML.

4.4 Caractéristique des données temporelles

Pour la plupart des travaux, les expressions temporelles ne sont pas explicitement définies de manière précise. C'est le manuel d'annotation qui détermine quel genre d'expression doit être pris en compte. La description du manuel s'intéresse aux expressions qui ont une portée applicative ou aux expressions pour lesquelles un traitement automatique est envisagé.

Dans le manuel TIDES (FERRO et al., 2005) qui a servi l'annotation du TimeML, pour être prise en

compte l'expression doit être un déclencheur lexical approprié (appropriate lexical trigger). Chaque déclencheur lexical est un mot ou une expression numérique dont le sens se rapporte à une unité de temps ou un concept comme « jour » ou « mois ». En outre pour être considéré comme un déclencheur, le référent doit pouvoir être orienté sur un calendrier, ou du moins orienté par rapport à un temps (passé, présent, futur). La table suivante est un exemple de déclencheur.

Tableau 10. Déclencheurs lexicaux du timeML

Part of Speech	Lexical Triggers	Non-Triggers
Noun	minute, afternoon, midnight, day, night, week, month, summer, season, quarter, year, decade, century, millennium, era, semester, [the] future, [the] past, time, period, point	instant, jiffy, episode, occasion, tenure, timetable, reign, light year, megawatt hour, lifetime, history
Proper name	(unique identifier for temporally-defined events:) Monday, January, New Year's Eve, Washington's Birthday, Solstice	
Specialized time patterns	8:00, 12/2/00, 1994, 1960s	
Adjective	recent, former, current, future, past, daily, monthly, biannual, semiannual, daytime, daylong, onetime, ago, preseason, short-term, long-term	early, ahead, next, subsequent, frequent, perpetual, later, contemporary, simultaneous, preceding, previous, existing, modern
Adverb[27]	currently, lately, hourly, daily, monthly, ago	earlier, immediately, instantly, forthwith, meanwhile, heretofore, previously, next, beforehand, following, later soon, sooner, shortly, eventually, occasionally, once, still, again, timely, whenever
Time noun/adverb	now, today, yesterday, tomorrow	
Number	3 (as in "He arrived at 3."), three, fifth (as in referring to "the fifth of June"), Sixties (as in referring to the decade "the Sixties"	

Les mots ou expressions qui font référence au temps et qui ne peuvent pas être situés sur le calendrier

[27] Un adverbe qui dérive d'un adjectif qui est un déclencheur.

ou un axe de temps (passé, présent, future) ne sont pas marquables (non markable). Les prépositions et les conjonctions de subordination qui ont une notion du temps font parties de cette catégorie.

Tableau 11. Non marquable

Part of Speech	Non-Triggers
Subordinating Conjunction	when, while, as, since, now [that], as long as, as soon as, sooner than, every time, any time
Preposition	at, on, in, for, over, throughout, during, before, after, since, until

Les noms propres qui désignent quelque chose d' autre qu'une entité temporelle ne sont pas pris en compte. Il s'agit des nom comme *Black September, Four Days In September*.

(Vicente-Díez et al. 2008) adopte une approche structurelle pour la typologie des expressions temporelles en anglais. La classification contient deux types de constituants :

- les unités temporelles
- les modificateurs

Cette classification est motivée par un point de vue linguistique. L'expression du temps est considérée comme une structure syntaxique qui dispose d'un noyau et des éléments facultatifs. Les unités temporelles correspondent au noyau. Les modificateurs représentent les éléments facultatifs.
Les unités temporelles sont composées de différents éléments :

- des unités de mesure temporelle (« hour », « minute », « week »),
- des unités déictiques (« today », « yesterday »),
- des unités nommées (« Monday », « 1998 », « 12/10/2007 »).

Les modificateurs selon leur position peuvent être classés en « pré-modificateur » avant le noyau ou en post-modificateur après le noyau. Les modificateurs peuvent cependant être classés selon le contenu sémantique. Par exemple les nombre ordinal ou les modificateurs de fréquence.

Des horaires complexes correspondants à l'ouverture ou la fermeture des sites touristiques ou des restaurants sont traités par (Weiser, 2010). La particularité des travaux de (Weiser, 2010) est qu'il

prend en compte les exceptions. Exemple : *Ouvert tous les jours de 11h à 18h. Fermé le soir sauf le vendredi et Ouvert du lundi au samedi de 11h à 17h30.*

4.5 Catégorisation des valeurs temporelles

L'interprétation temporelle a fait l'objet des nombreux travaux (Maurel et Mohri 1994), (Mani et Wilson, 2000), (Wilson et al., 2001), (Filatova et Hovy, 2001), (Muller et Tannier, 2004), (Battistelli et al., 2006), (Ahn et al., 2005, 2007), (Vicente-Díez et al., 2008), (Parent et al., 2008), (Martineau et al., 2009). Selon (Kevers, 2011), l'interprétation de la valeur de l'expression est souvent relativement sommaire. Il existe cependant des classifications étendues, établies par (Muller et Tannier, 2004) qui distinguent 11 catégories :

- les dates non absolues (« le 25 mars », « en juin », etc.),
- les dates absolues (« le 14 juillet 1789 »),
- les dates relatives au moment d'élocution (« il y a 2 ans », « l'année dernière »),
- les dates relatives au focus temporel (« trois jours plus tard »),
- les dates absolues de forme particulière (« au début des années 1980 »),
- les dates relatives de forme particulière (mois, saisons),
- les durées quelconques (« pendant 3 ans »),
- les durées contenant deux dates (« du 11 février au 27 octobre »),
- les durées absolues (« à partir du 14 juillet »),
- les durées relatives au moment de l'élocution (« depuis un an »),
- les durées relatives au focus temporel (« depuis »),
- les atomes de temps (« trois jours », « 4 ans », etc.).

(Kevers, 2011) élabore une spécification des expressions temporelles pour la reconnaissance et la normalisation. Les différentes catégories sont caractérisées par quatre critères binaires :

- Ponctuel ou Duratif (P ou D) ;
- Absolu ou Relatif (A ou R) ;

- Précis ou Flou (P ou F) ;
- Unique ou Répétitif (U ou R).

Le premier critère (Ponctuel ou Duratif) permet de distinguer les expressions dont la représentation temporelle correspond, à une certaine granularité donnée, à un point ou au contraire à un intervalle. L'expression ponctuelle désigne toujours une zone temporelle comme un tout indissociable et est exprimée par une référence unique à l'espace du temps. Une expression durative est obligatoirement exprimée à l'aide de deux bornes. Dans certains cas, l'une de celles-ci peut cependant ne pas être explicitement décrite. Des exemples des expressions ponctuels ou duratives sont :

P : « lundi 20 septembre 2010 »

P : « en 2010 »

P : « au vingtième siècle »

D : « du lundi au jeudi »

D : « jusqu'au mois de septembre »

Le second critère (Absolu ou Relatif) classe les expressions selon que leur localisation sur la ligne du temps est directe, indépendante de tout autre point dans l'espace du temps (localisation absolue) ou qu'elle nécessite l'utilisation d'un repère temporel (localisation relative). Une expression temporelle est donc absolue si elle contient en son sein tous les éléments nécessaires à son identification univoque dans un calendrier. La nature et le nombre de ces éléments varient en fonction de la granularité de l'expression. Pour être situées sans ambiguïté dans un calendrier, les expressions relatives nécessitent par contre la connaissance d'un point de repère. Celui-ci peut être le moment d'énonciation (référence déictique) ou un autre point temporel contextuel, explicitement renseigné ou non (référence anaphorique).

A : « le 20 septembre 2010 »

A : « 2005 »

R : « lundi »

R : « la veille »

Le troisième critères (Précis ou Flou) indique si la zone temporelle décrite par l'expression peut être délimitée de manière précise ou si, au contraire, sa délimitation manque de précision. Une expression est qualifiée de précise si on considère, quelle que soit sa granularité, qu'elle couvre l'entièreté de la zone qu'elle désigne, à l'exclusion de tout le reste.

P : « le jeudi 23 septembre 2010 » ;

P : « à 14h00 » ;

P : « le 20ème siècle ».

Une expression floue étend la couverture, de manière proche, mais indéterminée, autour de la zone désignée ou, au contraire, la restreint à une fraction de celle-ci. Le caractère flou peut donc être obtenu de plusieurs façons.

F : à l'intérieur d'une plage temporelle de granularité élevée, « durant le mois de décembre ») ;

F : aux alentours d'un repère temporel précis, « vers le 22 décembre 2009 » ;

F : spécification relative imprécise, « dans environ trois semaines ».

Enfin, le dernier critère (U Unique ou Répétitif) sépare les expressions donnant lieu à une représentation unique (U) ou répétée (R) sur la ligne du temps.

U : « mardi matin » ;

R : « chaque dimanche » ;

R : « durant trois vendredis ».

Ces quatre critères binaires sont croisés, ce qui produit une nomenclature de seize catégories. L'aspect *Répétitif* du dernier critère n'est pas pris en compte comme il est considéré moins important pour son application d' « accès sémantique aux bases de données documentaires ». Les combinaisons considérées sont donc au nombre de huit : PAPU, PAFU, PRPU, PRFU, DAPU, DAFU, DRPU et DRFU.

- PAPU : Référence Ponctuelle, Absolue, Précise et Unique. Cette combinaison correspond à un point dans l'axe calendaire. Selon la granularité, il peut s'agir d'heure, jour, semaine, week-end, mois année. L'expression contient alors les différents éléments qui permettent de préciser sans ambigüité le temps spécifié.

Exemple :

un mois : « mars 2007 » ;

une semaine ou un week-end : « le week-end des 12 et 13 juin 1999 » ;

un jour : « le 12 juillet 2007 » ;

une heure : « le 12 juillet 2007 après midi, à 16h00 »

La catégorie PAPU comprend également les zones temporelles désignées à l'aide d'une période nommée (nom de fête, de vacances, etc.), d'un nom de saison (météorologique) ou d'une autre période de l'année comme :

« Noël 2005 »,

« les grandes vacances 2006 »,

« l'été 2007 »,

« le carnaval 2008 »,

« le premier trimestre 2009 ».

- PAFU : Référence Ponctuelle, Absolue, Floue et Unique. Cette catégorie correspond à un point de l'axe calendaire mais qui est flou. C'est une expression PAPU floue. L'imprécision peut se situer également à divers niveau : heure, journée ou supérieur à une journée. Exemple :

« 22/12/2009 vers 18h24 » ;
« 22 décembre 2009 matin vers 10h24 »
« mardi 22 décembre 2009 durant l'après-midi » ;
« 22 décembre 2009, fin de matinée » ;
« vingt-deux décembre 2009 en fin d'après-midi ».
« aux environs de l'an 2000 » ;
« au début des années 30 » ;
« vers la fin de l'été deux-mille-quatre ».

- PRPU : Référence Ponctuelle, Relative, Précise et Unique

Les expressions qui rentrent dans la catégorie PRPU possèdent la particularité d'avoir besoin d'un point de référence afin de pouvoir être identifiées correctement. Cette caractéristique se matérialise sous la forme de dates sous-spécifiées ou de déplacements temporels. Exemple :
« décembre » ;
« jeudi » ;
« le jour de la Toussaint » ;
« [le matin,] [à 10 heures,] [le] jeudi quatorze décembre » ;
« le week-end du samedi seize décembre » ;
« la semaine du quatorze décembre ».
« à 13 heures » ;
« le soir » ;
« à l'aube, à 5h45 » ;
« la Toussaint précédente » ;
« la semaine dernière » ;
« l'année prochaine » ;
« l'heure d'après » ;
« ce soir » ;
« cette année là ».

- PRFU : Référence Ponctuelle, Relative, Floue et Unique

L'expression peut-être floue du fait de l'approximation de la valeur numérique ou du caractère indéfini du déterminant de l'expression temporelle.
« il y a [environ] un an » ;
« [à peu près] deux mois plus tôt »
« il y a quelques jours » ;
« plusieurs semaines après »
« quelques jours après jeudi » ;
« [environ] un an avant le 22 décembre 2009 » ;
« quasiment 4 jours après la Toussaint ».

- DAPU : Référence Durative, Absolue, Précise et Unique

La catégorie DAPU regroupe des zones temporelles exprimées sous la forme d'intervalles, et dont la ou les bornes sont exprimées sous la forme de points temporels bien identifiés. Cela signifie qu'il est possible de déterminer directement, et avec précision (en fonction d'une certaine granularité), l'emplacement dénoté par l'expression dans l'espace du temps. Exemple :
« 10/03/2005, entre 10h45 et 12h30 »,
« 10 mars 2005, depuis le matin »,
« Noël 2005, depuis 10h45 ».
« les années 2009 à 2010 »,
« du 8 mars 2005 au 10 mars 2005 »,
« du 8 au 10 mars 2005 » (mois et année factorisés),
« depuis décembre 2003 »,
« jusqu'au 10 mars 2005 ».

- DAFU : Référence Durative, Absolue, Floue et Unique

La catégorie DAFU correspond à une expression de type DAPU pour laquelle il existe une approximation de localisation. Les bornes sont donc constituées d'expressions répondant aux exigences de la catégorie PAPU. Cependant, au moins l'une d'elles reçoit une marque d'approximation, de partition ou une combinaison des deux.
« 10 mars 2005, depuis le début de matinée »,
« 10 mars 2005, depuis 10h45 jusqu'aux alentours de 12h30 ».
« depuis le 8 mars 2005 jusqu'aux environs du 10 mars 2005 »,

« à partir de début décembre 2009 »,
« jusqu'à fin 2010 ».
« durant la nuit du 9 au 10 mars 2005,
« au début de la nuit du 9 au 10 mars 2005 ».

- DRPU : Référence Durative, Relative, Précise et Unique

D'une manière générale, les expressions de type DRPU sont, du moins en surface assez similaires à celles rencontrées pour la catégorie DAPU. La différence entre ces deux catégories intervient dans le fait que les expressions qui constituent les bornes s'apparentent à la catégorie PRPU et non plus PAPU.
« [jeudi,] entre 10h45 et 12h30 »,
« [le 10 mars,] depuis 10h45 »,
« [le jeudi 10 mars,] jusqu'à soir ».
« du 8 au 10 mars »,
« jusqu'au 10 mars »,
« dès le 18 mai prochain »,
« jusqu'au 31 décembre de l'an dernier ».
« dans les trois jours »,
« depuis deux mois ».

- DRFU : Référence Durative, Relative, Floue et Unique

Enfin, la catégorie DRFU est le pendant approximatif des expressions de type DRPU. Les différents cas identifiés pour cette catégorie sont par conséquent transposables à celle-ci, en prenant soin d'y ajouter les marques d'imprécision et de partition nécessaires au caractère approximatif de DRFU.
« [le 10 mars,] depuis environ 10h45 »,
« [le jeudi 10 mars,] jusqu'en début de soirée »,
« [jeudi,] jusqu'aux alentours de 12h30 ».
« entre début janvier et fin février »,
« depuis début décembre »,
« jusqu'aux environs du 10 mars ».
« d'ici plus ou moins trois jours »,
« depuis à peu près deux mois »,
« d'ici deux à trois semaines ».

4.6 Schémas d'annotation TimeML

TimeML[28] utilise la syntaxe XML. Les différentes balises sont :

- la balise événement « EVENT »
- la balise temps « TIMEX3 »
- la balise « SIGNAL »
- des balises de relations : « TLINK », « SLINK » et « ALINK ». Ces balises établissent un lien entre la balise TIMEX3 et la balise EVENT. Chaque balise a des attributs dont l'identifiant.

4.6.1 La balise EVENT

Syntaxiquement un évènement est généralement un verbe ou un nom dans la phrase. L'événement peut également être un état. Il peut être de type suivant :

- *Reporting* : rend compte d'évènements décrivant ou rapportant l'action d'une personne ou d'une organisation : déclarer quelque chose, raconter un évènement ou informer sur un événement. Il correspond à des verbes comme say, report, tell, explain, state. Exemple :
  Punongbayan said that the 4,795-foot-high volcano was spewing gases up to 1,800 degrees.
  No injuries were reported over the weekend.
- Perception : Cette catégorie comprend les événements impliquant la perception physique d'un autre événement. Ces événements sont généralement exprimés par des verbes tels que: *see, watch, glimpse, behold, view, hear, listen, overhear.*

  Witnesses tell Birmingham police they saw a man running.
  "You can hear the thousands of small explosions down there", a witness said.
- Aspectual : l'aspect renseigne sur les différentes facettes de l'évènement. Il peut s'agir d'une initiation, réinitialisation, culmination, continuité, terminaison.

  The volcano began showing signs of activity in April for the first time in 600 years.
  All non-essential personnel should begin evacuating the sprawling base.

- I_action : c'est une action intentionnelle. Elle peut être marquée par des verbes comme : *attempt, try, scramble, investigate, investigation, look at, delve, postpone, defer, hinder, set back*

[28] http://www.timeml.org/site/publications/timeMLdocs/annguide_1.2.1.pdf

- I_state : similaire à I_action, il exprime un état intentionnel. Il peut être introduit par des verbes comme : *believe, think, suspect, imagine, doubt, feel, be conceivable, be sure*

- Occurrence : désigne tout autre type d'événement qui arrive dans le monde. Voici quelques exemples :

  The Defense Ministry said 16 planes have landed so far with protective equipment against biological and chemical warfare.

  Mordechai said all the gas masks from abroad would arrive soon and be distributed to the public, adding that additional distribution centers would be set up next week.

  Two moderate eruptions shortly before 3 p.m. Sunday appeared to signal a larger explosion.

4.6.2 La balise SIGNAL

Un signal est un élément textuel qui rend explicite la relation qui existe entre deux entités (Timex et événement, Timex et Timex, ou événement et d'événement). Il peut s'agire :

- des prépositions « on, during »
- des connecteurs temporels « when »
- des conjonctions de subordination « if »
- des caractères spéciaux comme « - », « / » qui expriment un intervalle. Exemple 4-6, Apr. 1999/Jul. 1999.Exemple d'annotation :

John taught <SIGNAL sid="s1"> on </SIGNAL> Monday.

All passengers died <SIGNAL sid="s1"> when </SIGNAL>

the plane crashed into the mountain.

They will investigate the role of the US <SIGNAL sid="s1">before, during and after</SIGNAL> the genocide.

4.6.3 La balise TIMEX3

TIMEX3 permet d'interpréter la valeur temporelle de l'expression. C'est le résultat de la phase de reconnaissance et de normalisation. Cette balise est le successeur de TIMEX2 qui est elle aussi basée sur TIMEX developpé lors des conférence MUCs (Message Understanding Conference). TIMEX2 et TIMEX3 sont similaires. TIMEX2 inclut les modificateurs de l'expression avec l'expression temporelle alors TIMEX3 sépare ces modificateurs en leur assignant une balise SIGNAL. TIMEX3 ne permet pas de balise imbriquée. La relation entre les expressions temporelles est gérée par d'autres liens. TIMEX2 autorise des balises imbriquées. TIMEX3 compte d'autres attributs

supplémentaires :

- type : qui indique la nature de l'expression. DATE, TIME, DURATION et SET sont les valeurs possibles pour cet attribut. La valeur DATE est attribuée pour les dates calendaires comme « mercredi 5 Janvier 20114 ». TIME est utilisé pour exprimer le temps horaire « ten minutes to three», «twenty after twelve ». DURATION indique une durée. La valeur est utilisée juste pour une durée explicite comme dans « Mr. Smith stayed 2 months in Boston ». La valeur SET désigne un ensemble de temps. Exemple «John swims every 2 days ».
- date de création du document.
- fonction temporelle : elle indique si un module est nécessaire pour déterminer la valeur normalisée de l'expression temporelle. Par exemple pour l'expression « late last night » la fonction temporelle peut avoir une valeur positive mais pour l'expression « twelve o'clock January 3, 1984 » la valeur positive n'est pas obligatoire car l'expression est proche de sa forme normalisée.
- point de début et point de fin : Ces attributs sont utilisés lorsqu'une durée est ancrée dans une autre expression de temps. Si un point de début pour une durée est donné dans le texte, l'autre point est calculé avec ce point. Ceci peut-être illustré par l'exemple suivant :

```
John begins teaching <TIMEX3 tid="t1" type="DURATION" value="P1W"
beginPoint="t2" endPoint="t3"> one week </TIMEX3> from <TIMEX3
tid="t2" type="DATE" value="XXXX-9-15"> September 15 </TIMEX3>

<TIMEX3 tid="t3" type="DATE" value="XXXX-9-22"
temporalFunction="TRUE" anchorTimeID="t1"/>
```

- quantité (*quant*) et fréquence (freq): Ces attributs sont utilisés lorsque l'expression temporelle est un ensemble SET. *Quant* est un littéral du texte qui quantifie l'expression . *freq* représente la fréquence. Exemple :

```
<TIMEX3 tid="t1" type="SET" value="P1W" freq="2X"> twice a week
</TIMEX3>

<TIMEX3 tid="t1" type="SET" value="P2D" quant="EVERY">
```

```
every 2 days
</TIMEX3>

<TIMEX3 tid="t1" type="SET" value="P1W" quant="EACH" freq="3d">
3 days each week
</TIMEX3>

<TIMEX3 tid="t1" type="SET" value="XXXX-10" quant="EVERY">
every October
</TIMEX>
```

Au-delà de l'attribut valeur (value), TIMEX3 partage avec TIMEX2 l'attribut *mod* qui indique les modificateurs de l'expression. Le modificateur est pris ici au sens de modificateur de quantité plutôt que modificateur au sens syntaxique comme défini par (Vicente-Díez et al. 2008). Les valeurs prises par l'attribut mode sont listées dans la table suivante dans la colonne « Token ».

Tableau 12. valeurs de l'attribut mode

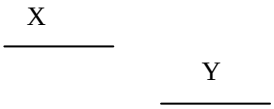
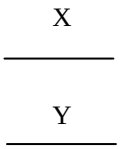
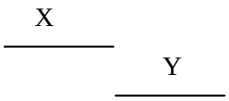
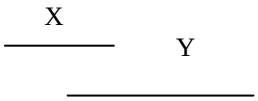
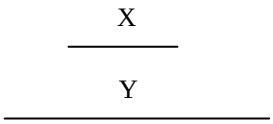
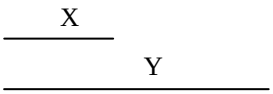
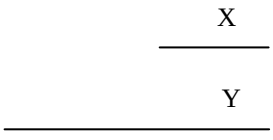
	Token	**Sample Expressions**
Point	BEFORE	more than (“more than a decade ago”)
	AFTER	less than (“less than a year ago”)
	ON_OR_BEFORE	no less than (“no less than a year ago”)
	ON_OR_AFTER	no more than (“no more than a year ago”)
Duration	LESS_THAN	less than (e.g., “less than 2 hours long”) nearly (e.g., "nearly four decades of experience")
	MORE_THAN	more than (e.g., "more than 5 minutes”)
	EQUAL_OR_LESS	no more than (e.g., “…will be open no more than 10 days”)
	EQUAL_OR_MORE	at least (e.g., “…will be open at least 10 days”)
Points and Durations	START	early (e.g., "the early 1960s") dawn (e.g., "the dawn of 2000") start (e.g., "the start of the quarter") beginning
	MID	middle (e.g., “the middle of the month”) mid- (e.g., “mid-February”)
	END	end late
	APPROX	about (e.g., “about three years ago”) around

4.6.4 Les liens

4.6.4.1 TLINK

Le lien TLINK (temporal link) permet d’établir une relation entre deux évènements ou un évènement et une expression temporelle. Les relations correspondent à la phase d’ordonnancement. Les relations sont basées sur celles de (Allen, 1984). Pour des événements ou de l’expression temporelle X et Y les 13 cas suivant sont possibles :

Tableau 13. Relations temporelles d'Allen

Relation	Symbole	Symbole inverse	illustration	Interprétation
X before Y	<	>		X se déroule avant Y
X equal Y	=	=		X est égale Y
X meets Y	M	mi[29]		X est avant Y mais il n'y a pas d'intervalle entre eux.
X overlaps Y	O	oi		X commence avant Y et ils se chevauchent.
X during Y	D	di		X se déroule complètement pendant Y
X starts Y	S	si		X et Y partage le même point de départ mais X termine avant Y
X ends Y	F	Fi		X et Y finissent ensemble mais X commence après Y.

4.6.4.2 SLINK

Un lien SLINK (subordination link) relie deux évènements. Il existe plusieurs types de liens SLINK:

[29] mi : le i signifie inverse

- modal : pour annoter des évènements introduisant une référence à un monde possible principalement des évènements de type « I_ACTION » et « I_STATE ».
- factif : lorsque certains verbes introduisent une présupposition sur la véracité de leurs arguments. Il s'agit des verbes comme *regret, forget ou manage.*

Mary regrets that she didn't marry John.

- contrefactif: contrairement à la relation précédente l'évènement produit une présupposition sur la non véracité de son argument. Il s'agit de certains verbes comme

forget (to), unable to (in past tense), prevent, cancel, avoid, decline, etc .

Mary was unable to marry John.

- évident: les liens d'évidence sont typiquement introduits par des rapports (évènement de type REPORTING) ou de perception (PERCEPTION)
- non évident : pour des situations de rapport ou de perception ayant une polarité négative.

John denied he bought only beer.

- conditionnel: une relation conditionnelle peut intervenir entre deux évènements quelconques et est généralement accompagnée par un signal comme « if ». L'évènement introducteur est celui de la clause introduite par « if ». L'événement subordonné est celui de la conséquence de la clause « if ».

If John brings the beer, Mary will bring the chips.

4.6.4.3 ALINK

Un lien ALINK (Aspectual Link) représente une relation entre un évènement *aspectuel* et son argument. Exemple de relation aspectuelle à considérer :

- initiation d'un évènement
- aboutissement d'un évènement
- arrêt d'un évènement
- la poursuite d'un évènement

4.7 Les méthodes d'extraction

Comme la plupart des applications en traitement automatiques des langues, l'extraction d'information temporelle utilise des approches symboliques, statistiques ou hybrides.

Les approches symboliques utilisent des règles pour traiter les différentes étapes.

4.7.1 Approches symboliques

Les approches symboliques ont la particularité d'être efficaces même dans un domaine où il n'y a pas de corpus d'entrainement. Les expressions régulières figurent en bonne place parmi les outils utilisés pour traiter les expressions temporelles. Les problèmes rencontrés avec les expressions sont (Kevers, 2011), (Weiser, 2010), (Zavarella et Tanev, 2013):

- détermination et interprétation du temps verbal,
- expressions avec point de référence constitué par un événement nommé : en général, seules les expressions qui sont explicitement temporelles sont prises en compte. Avec des expressions comme « deux heures *après l'accident* », pour bien situer l'heure en questions l'heure de l'accident doit être identifiée. Comme « accident » ne désigne pas explicitement une entité temporelle, le système n'arrive pas à bien interpréter la valeur précise.
- meilleur reconnaissance des intervalles
- gestion du point de référence pour les dates relatives
- portée et résolution des références lointaines
- prise en compte de l'aspect grammatical et lexical.
- lier l'expression temporelle à l'évènement
- développement de ressources conséquentes pour avoir une bonne couverture du domaine.

La plupart des problèmes sont en général liés au caractère local des expressions régulières. Une stratégie d'analyse locale ne permet pas une vue globale de la phrase. Ces problèmes peuvent cependant être traités en utilisant une analyse syntaxique (Kevers, 2011).

En plus des expressions régulières, HeidelTime (Strötgen et al., 2013) utilise les catégories grammaticales. Elles permettent de faire certaines désambigüisations notamment pour le verbe. Comme par exemple :

Il est arrivé ***lundi***

Il arrivera ***lundi***

La prise en compte du temps verbal permet de déterminer la date exacte du lundi en question soit le lundi déjà passé ou le lundi à venir.

La context-scanning strategy (CSS) (Desclés et al. 1997 ; Berri et al.1996; Ben Hazez et al., 2000) est une approche utilisé par (Vazov, 2001). Le CSS est basé sur l'hypothèse que la représentation sémantique n'est pas déterminée par une connaissance du monde mais peut être considérée comme les éléments de l'information linguistique locale. Le CSS a deux caractéristiques principales :

- il a été conçu pour effectuer une tache très spécifique : identifier et interpréter tous les éléments linguistiques de surface qui représentent une représentation sémantique particulière. Il ne prétend pas fournir une analyse complète des textes.

- Le CSS extrait un ensemble d'éléments interactifs intégrés dans les représentations sémantiques complexes. Contrairement aux approches basées sur la connaissance du monde, il en déduit ces représentations en utilisant exclusivement des données linguistiques figurant dans le texte.

La procédure d'identification repose sur une stratégie d'exploration contextuelle qui met en œuvre deux techniques complémentaires : recherche des patrons (expressions régulières) et *chart parsing* (Kay, 1980) qui est déclenchée en fonction des patrons repérés. Le *chart parsing* est un algorithme efficient pour traiter les ambigüités de la langue par rapport aux techniques de backtracking (chainage en arrière). Deux types de marqueurs sont exploités. Les marqueurs *autonomes* et les marqueurs *déclencheurs*.

Les marqueurs autonomes représentent des expressions temporelles autonomes. Ils sont regroupés en deux catégories disjointes :

- des chaines de caractères constants comme « par la suite », « le lendemain matin ».

- des éléments initiaux d'une chaine de caractères temporelle comme « quand » et mots qui les suivent (proposition introduite par ces mots initiaux généralement)

Les marqueurs déclencheurs sont à la fois un indicateur et un élément d'une large expression temporelle. Ils permettent de déterminer les limites de l'expression temporelle dans la phrase. Les marqueurs déclencheurs sont organisés en trois sous ensembles disjoints :

- des marqueurs qui bloquent l'expression à gauche. Il s'agit des marqueurs comme « durant » , « il y a ». Exemple : durant 2 ans, il y a 2 ans.

Ces marqueurs permettent de parcourir l'expression de gauche à droite comme l'expression est bloquée à gauche.

- Les marqueurs qui dénotent une expression temporelle et déclenchent un parcours gauche et droite par rapport au marqueur. C'est le cas des noms des mois (janvier, février), unités de

temps (minute, seconde) ou des saisons.

- Les marqueurs qui peuvent se positionner à un endroit quelconque de l'expression. Ils nécessitent un parcours gauche et droite. Ce sont des marqueurs comme *après*. Exemple :

Jeanne est arrivée trois minutes *après* Pierre.
Jeanne est arrivée trois minutes *après*.
Jeanne est arrivée *après* Pierre.
Jeanne est arrivée *après*.

Le système utilise les catégories grammaticales. Il traite d'abord les marqueurs autonomes avant d'exploiter les marqueurs déclencheurs.

4.7.2 Approches statistiques

Les algorithmes d'apprentissage utilisés sont généralement:

- CRF(Conditional Random Field) (Angeli et al., 2012 ; Adafre et de Rijke, 2005)
- SVM (Support Vector Machine) (Steven, 2013; Ahn et al., 2007),
- ME (Maximum Entropy) (Kolomiyets and Moens 2010;).

Les corpus généralement utilisés sont :

- TIMEBANK(Pustejovsky et al., 2003)
- Les corpus de TempEval
- French TimeBank (Bittar, 2010).
- et d'autres corpus spécialisés (Sun et Rumshisky, 2013).

Les descripteurs (features) utilisés sont généralement:

- Contexte des expressions
- Typographie (majuscule, minuscule)
- Catégories grammaticales
- Chunk
- Noyau des structures syntaxiques

4.8 Analyse temporelle d'une question pour des données structurées

Un des problèmes actuels des Systèmes de Question Réponse sur des données structurées est le manque de robustesse (Minock, 2008 ; Minock, 2010; Lopez et al., 2011; Yahya et al., 2013). Ce manque de robustesse se situe au niveau de la capacité du système à comprendre la question de l'utilisateur et de la capacité du système à identifier l'information recherchée même si la question est comprise.

4.8.1 Calculs liés à la nature des données

Une base de données est généralement statique du moins au moment où la réponse est recherchée. Mais la question de l'utilisateur n'est pas statique. Elle peut être formulée de plusieurs manières. En général, une base de données a deux fonctions : stocker les données et permettre de faire des traitements sur ces données plus tard. Le plus souvent, les questions difficiles sont des questions qui nécessitent des traitements. Des conditions souvent complexes doivent être calculées avant d'extraire la réponse. Dans cette thèse nous soutenons l'idée que la prise en compte de la manière dont ces traitements (ou calculs) s'expriment en langage naturel permet d'accroitre la robustesse des Systèmes de Question Réponse. La méthode d'identification de ces calculs peut s'appuyer sur les outils fournis par le langage formel de la base de données comme SQL ou SPARQL ou également sur la logique des données. Une prise en compte des fonctionnalités du langage formel est une bonne étape pour la robustesse du système. En général, la logique qui concerne des mesures existe déjà, par exemple pour le temps (Benthem, 1983; Allen,1984; Freska, 1992; Hayes, 1995). Mais la manière dont le langage naturel les exprime dans les questions a été peu abordée pour les Système de Question Réponses basés sur des données structurées dans un contexte hétérogène.

4.8.2 Compostions et expressions temporelles en contexte

Une expression temporelle en contexte est une expression qui à l'origine n'est pas explicitement temporelle. Par exemple :

Il est parti <u>deux jours après son enseignant</u>

Le mot «enseignant » ne dénote pas une expression temporelle comme « jour ». Mais du fait qu'il est nécessaire pour localiser la date de départ, c'est toute l'expression soulignée qui doit être prise en considération. Ceci est un peu différent des cas où l'expression pourrait avoir une connotation fortement temporelle comme :

Il est parti <u>deux jours après son anniversaire</u>

« anniversaire » a une connotation fortement temporelle. L'analyse du temps dans une question doit

nous permettre d'identifier le temps (si c'est le focus de la question) ou des contraintes temporelles qui pèsent sur la question. L'utilisateur se sert de ces contraintes pour désigner exactement ce dont il a besoin. Ces contraintes sont composées à la fois des expressions temporelles et d'autres outils qui appartiennent intrinsèquement à la langue ou au domaine. Ces outils peuvent être partagés entre le temps, la langue et le domaine.

A l'image d'une carte interactive comme Google maps, l'utilisateur à la possibilité d'obtenir une information avec une certaine granularité. Il peut par exemple observer tout le globe terrestre, un pays, une ville ou des choses plus petites encore. Pour cela, il peut effectuer des opérations comme *« agrandir la carte », « réduire la carte », « déplacer la carte dans une direction donnée »*. Il peut également se positionner directement sur un point précis en connaissant ses coordonnées. L'analyse temporelle suit ce schéma pour étudier les outils que le langage utilise et comment ces outils sont combinés pour exprimer exactement l'information voulue par l'utilisateur (dans un grand volume de données).

L'analyse temporelle d'une question pour des données structurées ne peut se limiter uniquement aux données purement temporelles. Il s'agit de prendre en compte tous les mots ou expressions qui peuvent avoir une notion de temps dans le contexte ou qui participent à l'interprétation de l'expression temporelle à l'image de (Vazov, 2001). Mais ce dernier n'effectue pas de normalisation qui est très importante dans notre contexte.

La manière dont ces expressions se combinent est aussi à prendre en considération. Par exemple, aucun des étiqueteurs temporels que nous avons eu l'occasion de tester (disponible en ligne, téléchargeable ou intégré dans un autre logiciel), n'identifie correctement l'expression « *5, 6 and 7 October 2013 »* pour le moment. Du fait de la coordination *and* dans cette expression, ces étiqueteurs temporels reconnaissent seulement *7 October 2013.* Les autres 2 dates (5 October 2013 et 6 October 2013) sont ignorées. Ce qui peut se traduire par une perte d'information.

4.9 Notre approche de traitement du temps pour l'interrogation des données structurées

Etant donnée la diversité des domaines auxquels le système peut être confronté, une approche générique permettant de s'adapter facilement est importante. Les entités temporelles explicites ne peuvent pas être traitées indépendamment des autres entités de l'application. Les objectifs poursuivis par l'analyse temporelle sont :

- Identifier et interpréter les entités temporelles explicites et leur logique telle qu'elle se manifeste en langage naturel.

- Identifier et interpréter la relation des entités temporelles explicites avec les autres entités de l'application qui forment une expression cohérente et complète.
- Réduire le développement de ressources liées à la méthode.
- Réutiliser la même méthode pour les autres mesures.

Pour atteindre ces objectifs, nous avons développé des ressources :

- Un étiqueteur : Il fournit les catégories grammaticales en mettant l'accent sur l'ambigüité grammaticale afin de faciliter le traitement des autres problèmes linguistiques ultérieurement. L'étiqueteur fournit également le temps verbal, pour permettre de situer les expressions lors de la normalisation. L'absence du temps verbal ou la difficulté à déterminer ce dernier a été à l'origine des interprétations erronées des expressions temporelles en particulier pour ceux qui ont mené une analyse locale avec les expressions régulières (Kevers, 2011; Weiser, 2010; Zavarella and Tanev, 2013).
- un analyseur syntaxique de surface. L'analyse syntaxique de surface fournit une représentation syntaxique de la phrase en se basant sur les dépendances des catégories grammaticales et en catégorisant également les unités lexicales qui permettent d'identifier facilement la composition de l'expression temporelle ou de la mesure dans d'autres situations.

Dans la littérature, l'information temporelle a été très rarement définie. En général, c'est à travers le manuel d'annotation que l'on comprend ce qu'est une expression temporelle ou non. Pour notre besoin d'analyse de l'information temporelle dans le cadre d'un système de Question-Réponse basé sur des données structurées, nous tentons néanmoins de donner une définition.

4.9.1 Définition de l'information temporelle

L'information temporelle est une expression qui dénote intégralement ou partiellement la notion du temps. Ces mots ou expressions peuvent être :

- des expressions temporelles explicites comme : jour, janvier, année
- des expressions temporelles contextuelles
- des expressions qui ont une dimension temporelle comme : contemporain, adolescent
- des marqueurs linguistiques ou les propositions qui indiquent le temps.

Le but de la définition est de déterminer l'information temporelle en incluant la temporalité en termes de composition avec les autres entités du domaine ou les outils de la langue. Mais comme nous travaillons sur une base de données, l'information temporelle doit avoir un sens au niveau de la base de données. Il ne s'agit pas donc de toutes les compositions ou subtilités de la langue. Par exemple si nous prenons la phrase:

Il est parti deux jours après son enseignant

Comme nous l'avons déjà vu, le mot « enseignant » n'est pas une entité temporelle explicite. Pour prendre en compte l'expression, la base de données doit avoir une propriété « date de départ » par exemple pour l'entité enseignant. La définition se limite en pratique à l'information que nous pouvons interpréter ou calculer au sens de la base de données.

Par comparaison à TimeML, les expressions nominales qui n'ont pas une notion temporelle explicite peuvent être considérées comme des évènements. Par conséquent, elles n'ont pas l'attribut VAL de TIMEX3 qui permet de trouver la valeur normalisée de l'expression.

4.9.2 Etapes prises en compte pour le traitement de la temporalité

Parmi les quatre étapes de traitement de l'information temporelle, nous nous intéressons particulièrement aux trois premiers :

- reconnaissance,
- normalisation ou interprétation.
- lier l'expression à l'événement.

La normalisation ne se fait pas au sens de la balise TIMEX3 de TimeML souvent très fragmentée (Le Parc-Lacayrelle et al., 2007). Comme les informations temporelles sont généralement stockées sous forme de date, il est préférable d'avoir directement la valeur finale de la date. Par exemple :*two weeks from June 7, 2003*[30] est normalisé comme suit:

```
<TIMEX3 tid="t6" type="DURATION" value="P2W" beginPoint="t61"
endPoint="t62"> two weeks </TIMEX3>
```

[30] http://www.timeml.org/site/publications/timeMLdocs/timeml_1.2.1.html#timex3

```
<SIGNAL sid="s1"> from </SIGNAL>

<TIMEX3 tid="t61" type="DATE" value="2003-06-07"> June 7, 2003 </TIMEX3>

<TIMEX3 tid="t62" type="DATE" value="2003-06-21" temporalFunction="true"
anchorTimeID="t6"/>
```

On pourrait juste avoir besoin de la valeur « **2003-06-21** » dans une requête formelle. Dans notre cas l'évènement peut-être soit une classe ou instance de classe soit un prédicat de l'ontologie.

Quant à l'ordonnancement, il se fait automatique dans la requête à travers les structures de comparaison.

4.9.3 Caractéristique des expressions temporelles

Comme la plupart des approches, TimeML, (Vicente-Díez et al. 2008), (Vazov, 2001), la tâche consiste d'abord à catégoriser les différents types de données qui interviennent dans l'information temporelle. Deux types de catégories sont proposées :

- les données de bases ou données élémentaires. Elles regroupent les unités lexicales qui ont une notion explicite du temps. Ces unités sont les mêmes dans la littérature. A ces unités, nous ajoutons également les attributs de la base de données qui peuvent avoir une valeur temporelle. Ce sont des attributs comme « date de naissance », « date de création ». Les instances ayant des attributs sont également à prendre en considération. Par exemple pour une classe « Personne », il est possible de faire référence à une instance de la classe « Personne » en utilisant des mots qui se définissent par un attribut temporel comme *adulte, octogénaire, adolescent*. La prise en compte du vocabulaire de la base de données dépend de l'application.

- Les opérateurs : les opérateurs incluent les prépositions, les conjonctions qui introduisent ou qui interviennent dans l'expression et les modificateurs de l'expression comme dans (Vicente-Díez et al. 2008). Les opérateurs peuvent être d'ordre morphologique également. La différence réside dans la prise en compte des modificateurs qui ont un sens dans la base de données et non pas seulement dans la logique temporelle. Par exemple dans l'expression « année *excédentaire* » si la base de données permet de calculer une *année excédentaire* selon la logique du métier, l'opérateur excédentaire sera pris en compte.

L'idée est de faire intervenir aussi la dimension métier dans l'interprétation des expressions temporelles. Ceci permet d'interpréter d'avantage des questions plus liées à la logique du métier qui peuvent se formuler différemment. La notion de modificateur est élargie à toutes les expressions environnantes susceptibles de modifier la valeur véhiculée par l'expression. Par exemple :

L'année ***dernière*** *;*

La seconde moitié de *l'année* ***dernière***

L'expression « *La seconde moitié de* » est un modificateur au même titre que « *dernière* ».

Un exemple de données de base et d'opérateurs est présenté dans les tables suivantes :

Tableau 14. Données de bases (ou données élémentaires)

Catégories grammaticales	**Données de bases**
Nom	temps, moment, époque, ère, quand, durée, millénaire, siècle, décennie, année, semestre, trimestre, mois, semaine, jour, heure, minute seconde, saison, présent, antiquité, aube, aurore, avenir, crépuscule, histoire, horaire, après-midi, veille, matin, midi, nuit, passé, ramadan, soir, soirée, délai, laps, lunaison, nuitée, quinquennat, fois, cycle, fréquence, périodicité, janvier, février, mars, avril, mai, juin, juillet, aout, septembre, octobre, novembre, décembre, hiver, printemps, été, automne.
adjectif	actuel, présent, passé, courant, automnal, hivernal, estival, nocturne, triennal, journalier, quotidien

Tableau 15. Opérateurs

Catégories grammaticales	**Opérateurs**
préposition	de, à, au, jusqu'à, jusqu'au, jusqu'en, entre, dans, depuis, à partir de, à compter de, après, avant, pendant, durant, tout au long de, par, vers, environ, plus, moins
adjectif	prochain, suivant, dernier, passé
Nom	moitié, triple, double, fois
Conjonction	quant, lorsque, avant que, après que, jusqu'à ce que, au moment où, en même temps, pendant que, et, où.
Déterminant	Déterminant cardinal, ordinal, défini, indéfini, démonstratif etc.

4.9.4 Catégorisation des expressions temporelles

L'objectif de la catégorisation est d'identifier les types de propriétés d'une donnée temporelle. Il repose sur le fait que les opérations appliquées à une donnée dépendent de ces propriétés. Une fois la propriété connue, il est possible d'identifier ses différents marqueurs. De ce fait, nous essayons d'abord d'identifier les propriétés des expressions temporelles. Nous ne prétendons pas à l'exhaustivité. Les propriétés suivantes peuvent être prises en compte :

- *ancrage* : permet de situer l'information temporelle dans l'axe du temps ou pas. Les valeurs possibles pour l'ancrage sont :
  - ✓ référentiel : l'expression est ancrée dans le temps
  - ✓ non référentiel : l'expression n'est pas ancrée dans le temps.
- *durée* : c'est l'espace de temps occupé par l'information qu'il soit ancré dans le temps ou pas. En pratique la durée n'est pas nulle même pour un point.
- *fréquence* : indique la répétitivité de l'information

- *inclusion* : indique dans une expression si une unité de temps peut être incluse dans une autre. Par exemple, le jour est inclus dans le mois. L'inclusion est une notion forte dans le temps et l'espace. La notion d'inclusion se réfère plutôt à des unités de temps indépendamment du contexte. Elle est utilisée pour une expression relative.

- *instanciation* : la possibilité de l'unité lexicale à être instanciée. Par exemple « jour » peut être instancié en lundi, mardi, mercredi etc. L'instanciation permet de résoudre les références anaphoriques. Par exemple l'expression « cette année 2006 ». Pris isolément « cette année » renvoie à l'année en cours. Par exemple 2014. En prenant en compte que 2006 est une instance d'année, il est possible de bien interpréter l'expression entière.

- *arithmétique* : possibilité d'effectuer des opérations arithmétiques comme l'addition, soustraction, la multiplication ou la division. Le traitement des opérations arithmétiques offre des perspectives intéressantes en traitement automatiques des langues. Mais la communauté a accordé peu d'attention à ce domaine (Narisawa et al., 2013).

- *ordonnancement* : possibilité d'ordonner les données temporelles.

- *opération logique* : possibilité de coordonner ou négativer des données.

- *approximation* : possible de prendre une valeur floue.

En fonction de la logique métier, d'autres propriétés peuvent apparaitre. Plusieurs de ces propriétés peuvent être exprimées sur une seule donnée de base ou dans une expression. L'interprétation de la donnée tient compte alors de la composition de ces propriétés sur la donnée. Pour chaque propriété, la langue dispose des opérateurs.
Exemple :

*Dans **la seconde moitié du 18 siècle, sauf pendant les années de guerre franco-anglaise**…(1)*[31]

*Le titre était en hausse dans **la seconde moitié de l'année avant 2014 sauf en Novembre**. (2)*

Pour l'interprétation de l'exemple (1), les opérateurs utilisés sont :

-moitié : opérateur arithmétique,

[31] Extrait de « Ce que nous devons de l'Afrique », Musée dauphinois, Grenoble, 2010

-seconde : opérateur d'ordre

-sauf : opérateur de négation (opérateur logique)

-pendant : opérateur d'ordonnancement

Pour chaque propriété des opérateurs sont utilisés comme marqueurs. Dans les paragraphes qui suivent nous présentons quelques exemples.

4.9.4.1 Opérateurs d'ancrage

Le tableau suivant montre des exemples d'ancrage selon les catégories grammaticales. La plupart des catégories sont neutres, c'est-à-dire qu' il est possible d'avoir l'ancrage référentiel ou non référentiel. Le tableau n'est pas exhaustif.

Tableau 16. Opérateurs d'ancrage

Catégorie	**Opérateur**	**Exemple**	**Type d'ancrage**
Déterminant démonstratif, défini ou ordinal	ce, cette, le, la les, premier, deuxième, dernier , etc	ce mois, le jour	référentiel
Déterminant indéfini	un, une, des	une année	non référentiel
Adjectif	prochain, suivant, dernier, passé	année suivante	référentiel

4.9.4.2 Opérateurs de durée

La durée peut être exprimée de plusieurs manières :

- en indiquant la distance,
- à partir d'un point de référence et de la distance. Si la distance est infinie, le point de référence porte implicitement la notion d'infinité par exemple *avant, après*. La borne peut être implicite comme « dans 3 ans » qui veut dire « 3 ans à partir du moment où on parle ».
- En indiquant les deux bornes.

Le tableau suivant présente quelques opérateurs suivant leur catégorie grammaticale.

Tableau 17. Opérateurs de la durée

Catégorie grammaticale	Structure	Type d'intervalle
Préposition	de …à… de …au… de…jusqu'à… de..jusqu'au… de… jusqu'en… entre…et…	Durée bornée
	avant, après	Permet d'indique une borne ou une distance.
	depuis…. dans….	Durée bornée par un point. Le point est lié au présent. Exemple « dans 3 ans ».
Adverbe	d'ici….	Durée bornée par un point lié au présent.
Locution verbale	Il y a …. il y a de cela….	Durée bornée par un point lié au présent.
Déterminant	nombre	Distance. Exemple 3 ans

Par exemple, en utilisant l'opérateur d'ancrage « premier » permet d'interpréter correctement l'expression « les 4 derniers mois avant 2008 ». Les étiqueteurs temporels que nous avons testés ne reconnaissent pas correctement l'expression « the last 4 months before 2008 ».

4.9.4.3 Les opérateurs de fréquence

La fréquence peut être marquée par les opérateurs suivants :

Tableau 18. Opérateurs de fréquence

Catégorie	**Opérateur**	**Exemple**
Déterminant	Tout, chaque	Chaque jour
Préposition	Par	par mois
D'ordre morphologique associé à une unité de temps	-suffixe	annuel mensuel trimestriel

4.9.4.4 Instanciation

Pour l'instanciation, il s'agit généralement des catégories nominales.

Tableau 19. Instances des unités temporelles

Classe	**Instanciation**
millénaire	instanciation ordinal : premier millénaire, deuxième millénaire
siècle	instanciation ordinal : premier siècle, 18^{e} siècle
décennie	le mot est suivi de la première année : décennie 80
année	instancier en nombre
semestre trimestre semaine	Numéroter : trimestre 1, trimestre 2
mois	janvier, février, mars, avril, mai, juin, juillet, aout, septembre, octobre, novembre, décembre
saison	hiver, printemps, été, automne
jour	lundi, mardi, mercredi, jeudi, vendredi.

4.9.4.5 l'inclusion

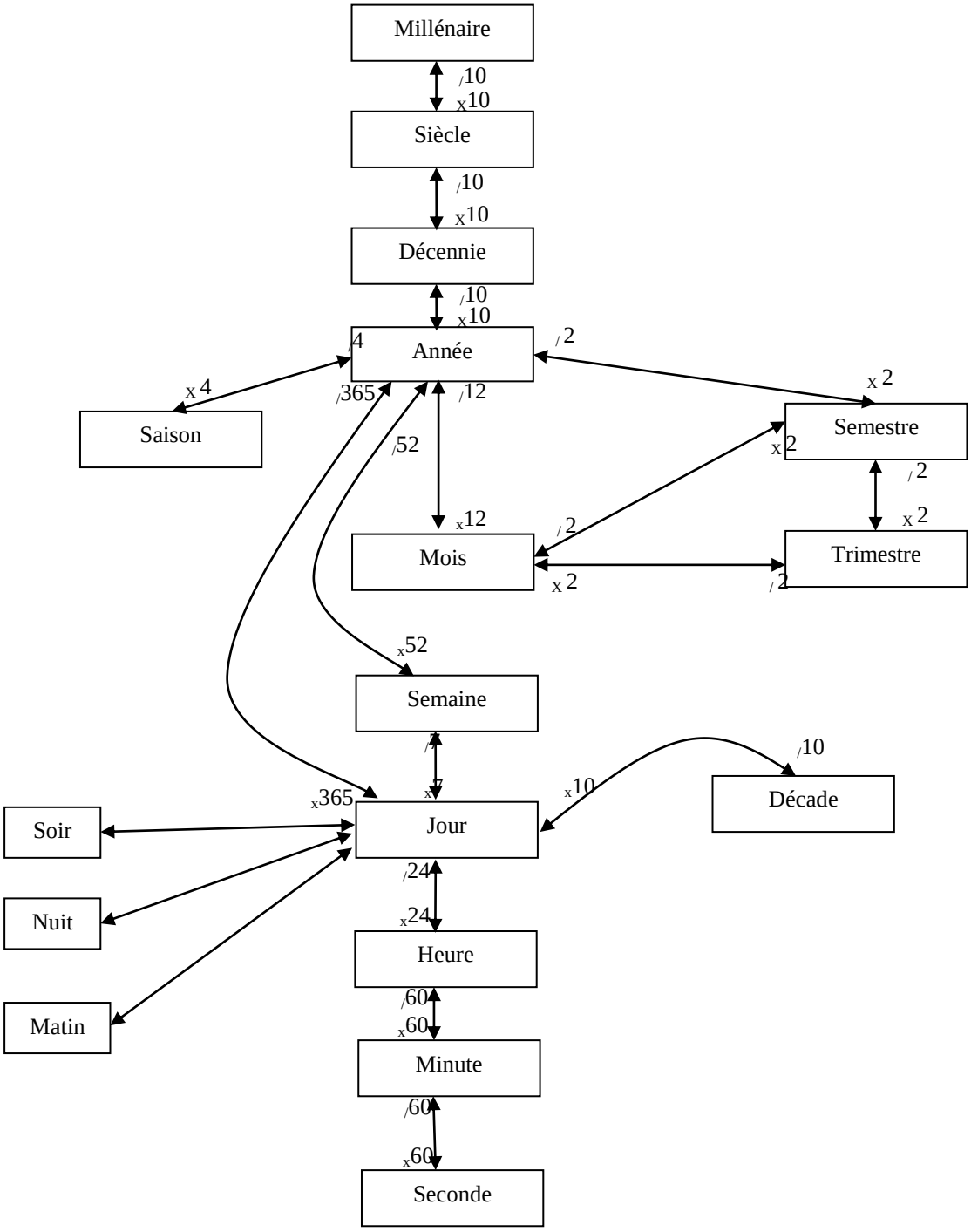


Figure 25. Granularité et relations des unités temporelles

Ce graphe présente les relations d'inclusion et les conversions possibles.

4.10 Conclusion

Le traitement de l'information temporelle est important dans la plupart des applications du Traitement Automatique des Langues. Plusieurs approches ont été abordées. TimeML a émergé comme un format d'annotation des expressions temporelles. Mais le fait que TimeML a été conçu sur des textes journalistiques (Pustejovsky and Stubbs, 2012) et vise principalement les expressions temporelles explicites n'a pas facilité son utilisation intégrale dans plusieurs domaines tel qu'il a été conçu. Actuellement, une application peut concerner plusieurs domaines. La gestion de la dimension temporelle est d'avantage compliquée.

Nous avons proposé un traitement de l'information temporelle (qui est susceptible d'être appliqué aussi aux autres types de mesure) qui est basé sur les propriétés de la donnée. La connaissance de la donnée permet d'appréhender les différentes manipulations (ou opérations) qui s'appliquent à la donnée. Ce traitement essaie également d'intégrer l'interprétions de la temporalité avec la logique métier.

Au niveau syntaxique, il s'appui sur un analyseur syntaxique de surface conçu de manière à faciliter l'identification des structures qui composent l'expression.

L'approche compositionnelle basée sur les propriétés des données temporelles et la logique métier permettent de gérer plus facilement la complexité des expressions temporelles en développant moins de ressources que dans (Kevers, 2011), (Weiser, 2010), (Zavarella et Tanev, 2013).

L'utilisation de l'analyse syntaxique de surface permet également de mieux gérer le temps verbal qui est généralement le point de référence lors de la normalisation. Par rapport à TimeML notre approche facilite l'analyse du temps combinée à la logique du domaine dans la mesure où la méthode repose à la fois sur la logique du temps et la logique métier et non seulement sur la logique temporelle.

Cette approche nécessite cependant de connaitre les propriétés de bases tant au niveau de la mesure que de la logique du domaine.

CHAPITRE 5 : EXPERIMENTATION

5.1 Introduction

Dans ce chapitre nous présentons l'architecture du système, les parties implémentées et des exemples. Dans un environnement hétérogène la flexibilité du système et la disponibilité des indices pour permettre une analyse linguistique sont des atouts. De ce fait, l'architecture du Système est orientée de manière à intégrer plus d'annotations aux niveaux lexical, syntaxique ou sémantique. Le schéma est susceptible d'être dynamique. A l'échelle du Web, les utilisateurs n'ont pas une connaissance de ces bases de données comme c'est le cas souvent à l'échelle d'une entreprise. Les questions des utilisateurs peuvent donc être imprévisibles.

La plupart des interfaces en langage naturel dans le cadre du Web de données (Linked Data) ont été faites pour la langue anglaise et souvent pour la langue allemande. Le multilinguisme n'est donc pas très développé en ce moment. Ceci est dû au fait que la compréhension de la question nécessite souvent de traiter certains problèmes qui sont généralement spécifiques à la langue qui ne peuvent pas être traités de manière générique. Nous avons essayé de nous intéresser au français d'abord.

5.2 Corpus

Les corpus utilisés sont : Europarl (Koehn, 2005), des corpus en ligne comme WebCorp Live[32]. Google est utilisé souvent comme corpus. Europarl est un corpus parallèle largement utilisé par la communauté du Traitement Automatique des Langues. La version française contient 32,550,260 mots. Europarl a été conçu à partir des verbatims du Parlement Européen.

5.3 Architecture du Système

L'architecture du système comprend quatre parties principales : étiquetage et désambiguïsation, générateur de l'arbre syntaxique, générateur de la forme logique, générateur de la requête formelle.

[32] http://www.webcorp.org.uk/live/

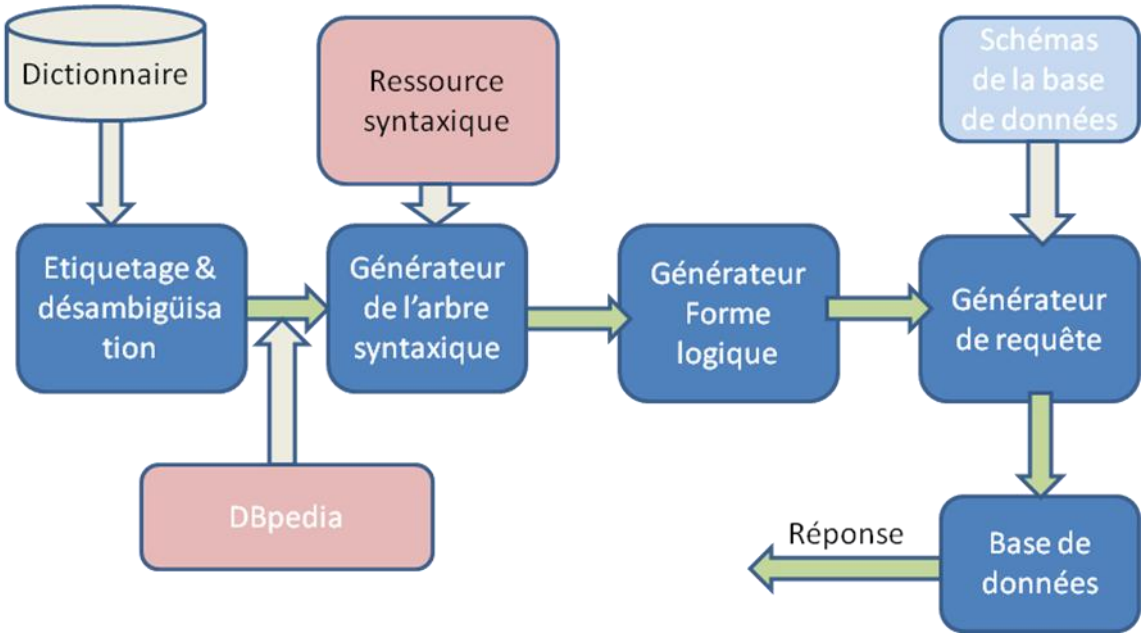


Figure 26. Architecture du système

5.2.1 Etiquetage et désambigüisation

Cette phase est appelé également *prétraitement*. L'objectif est de faciliter l'analyse linguistique (Soumana et al., 2013b). Le prétraitement vise à :

- servir de base à l'analyse syntaxique
- désambigüiser les catégories grammaticales
- bien identifier les anaphores,
- préserver la cohésion du groupe nominal,
- identifier les unités lexicales (mots, locutions, expressions figées)
- contrôler la phase d'étiquetage

La phase du prétraitement est nécessaire pour lever certaines ambigüités de la langue. Par exemple dans la phrase *« les entreprises devant rembourser la banque »*, le mot « devant » est étiqueté comme une préposition par la plupart des outils utilisés dans la littérature alors qu'il est un verbe. Ce qui change le sens de la phrase.

L'identification de l'anaphore n'est pas toujours triviale alors qu'il est nécessaire de l'identifier avant de pouvoir la traiter.

La cohésion du groupe nominal est importante. Un groupe nominal disloqué ne facilite pas l'interprétation de la question. Une unité lexicale est soit un seul mot, une locution, un nom composé

ou une expression. Il est important de conserver l'unité de ces expressions pour qu'elles ne soient pas disloquées également au niveau de la phase d'analyse syntaxique ce qui pourrait d'avantage compliquer l'interprétation en produisant des structures erronées.

La phase d'étiquetage est donc la première phase. De la qualité de cette phase dépend la qualité des autres étapes. En contrôlant cette étape, il est possible de prendre en amont des problèmes spécifiques à notre approche au lieu de se rendre compte plus tard que l'outil utilisé ne permet pas d'aller au delà de certaines limites ; tout ceci pouvant se traduire par des limitations techniques empêchant le déploiement de l'application à grande échelle. Ce déploiement à grande échelle est important pour notre application.

L'approche adoptée pour cette tâche consiste à établir des classes d'ambigüités et à les traiter en utilisant l'approche microsystémique (Cardey et Greenfield, 2003, 2005), (Cardey, 2013).

5.2.1.1 Vue générale du lexique de la langue

Le lexique est organisé en quatre ensembles : le lexique général, les nombres, les noms propres, le lexique spécialisé.

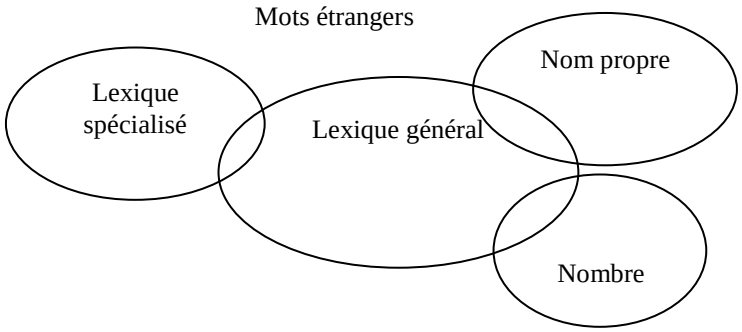


Figure 27. Vue générale du lexique

Le lexique général est l'ensemble des mots qu'un locuteur quelconque de la langue est susceptible de comprendre. Il comprend les mots simples, les mots composés, les locutions, les expressions (figées, moins figés, idiomatique, etc.). Le lexique général est fini en un instant donné et peut faire l'objet d'un dictionnaire. Les nombres servent à compter. La reconnaissance des nombres est aisée car ils sont bien formalisés. Certains nombres comme douzaine, centaine ou les chiffres (un à neuf) peuvent se retrouver dans un dictionnaire et sont donc considérés aussi comme élément du lexique général. Le nombre de noms propres est indéterminé. De ce fait, ils ne peuvent pas être recensés dans un dictionnaire de manière exhaustive. La délimitation précise des noms propres est souvent très délicate. Les noms propres partagent des éléments avec le lexique général (exemple : *Orange*, nom

de société, couleur ou fruit). Le lexique spécialisé est le lexique des domaines spécialisés comme la biologie, la chimie ou les mathématiques par exemple. La méthodologie doit prendre en compte les particularités du domaine.

5.2.1.2 Détermination des classes d'ambigüité et étiquetage

La méthode d'étiquetage consiste à établir des classes d'ambigüités (ambigüités de catégorie grammaticales) à partir d'un dictionnaire. Une classe d'ambiguïté *C* est par exemple :

$$C_{\text{déterminant, pronom}} = \{le,\ l',\ les,\ ce\} \quad (1)$$

$$C_{\text{déterminant, pronom, nom}} = \{tout,\ la\} \quad (2)$$

$$C_{\text{pronom, verbe}} = \{cela,\ lui,\ tu\} \quad (3)$$

L'exemple (1) donne des éléments qui peuvent être déterminant ou pronom. L'exemple (2) est l'ensemble des éléments auxquels l'étiquette *déterminant, pronom ou nom* peut être attribuée. Le mot « la » est un nom quand il désigne *une note de musique (sixième note de la gamme en do majeur).* L'exemple (3) correspond à l'ensemble des mots qui peuvent avoir l'attribut pronom ou verbe. Le mot « cela » peut être un pronom démonstratif ou le verbe « celer » à la troisième personne du singulier au passé simple de l'indicatif. Le mot « lui » peut être un pronom ou le participe passé du verbe « luire ». De même le mot « tu » est pronom ou participe passé du verbe taire. Chaque classe est traitée en utilisant l'approche microsystémique. Cette approche permet également de désambigüiser sans utiliser un dictionnaire en extension (Cardey et Greenfield, 2003). Le processus d'étiquetage commence par les catégories non ambiguës pour avoir un contexte local pour les mots ambigus. Le dictionnaire utilisé contient la catégorie grammaticale, le genre et nombre, le temps, la personne et le mode pour les verbes conjugués ainsi que le lemme. Les catégories du dictionnaire sont regroupées en 11 catégories auxquelles sont ajoutées trois autres catégories : la ponctuation, les symboles, les nombres et les mots inconnus. Des nouvelles classes d'ambigüités peuvent apparaître selon le domaine. Ces nouvelles classes peuvent être traitées par des modules liés au domaine en adoptant la même démarche.

5.2.1.3 Evaluation de l'étiqueteur

L'évaluation du système est faite sur le corpus (Koehn, 2005) en français. 16 959 phrases ont été extraites du corpus et étiquetées. Comme le corpus utilisé n'est pas annoté, l'évaluation est faite manuellement. 383 phrases totalisant 10 008 unités lexicales ont été évaluées. 9952 unités lexicales ont été étiquetées correctement. La précision du système est de 99,44% sur ce corpus.

En général, les erreurs les plus fréquentes se retrouvent au niveau des ambigüités qu'on peut qualifier de liées. L'ambigüité liée survient lorsque deux mots se suivent et l'étiquette de l'un dépend de l'étiquette de l'autre. Elle peut être illustrée par l'exemple suivant :

```
L'entreprise la concurrence    (4)

L'entreprise le concurrence    (5)

Le petit                       (6)
```

Pris isolement, les mots « la » et « concurrence » sont ambigus. Si les deux sont ensemble comme en (4), si le mot « la » est un article alors le mot « concurrence » est un nom. Si le mot « la » est un pronom alors le mot « concurrence » est un verbe. L'ambigüité liée diffère de l'ambigüité qu'on peut qualifié de libre comme en (6). Les mots « le » et « petit » sont ambigus pris isolément. Mais quelque soit l'étiquette possible {adjectif ou nom} pour le mot « petit », le mot « le » a toujours la même étiquette. Donc l'ambigüité entre « le » et « petit » peut être qualifiée de libre. Dans l'exemple 4, le mot « la » est étiqueté comme un article et le mot « concurrence » est étiqueté comme un nom. Ceci est dû au fait qu'au départ la priorité est accordée à la cohésion du groupe nominal. Les mots « la » et « concurrence » s'accorde en genre et nombre. La cohésion du groupe nominal est très importante pour un Système de Question-Réponse pour des bases de données lors de la phase d'interprétation. Des règles de correction appliquées ultérieurement peuvent lever l'ambigüité ; par contre l'exemple (5) est étiqueté correctement car le mot « le » ne s'accorde pas avec le mot « concurrence » dans un groupe nominal. L'expression « le concurrence » n'est donc pas un groupe nominal.

Exemple de phrases étiquetées :

Portons notre regard au-delà des dix, quinze ou vingt-cinq prochaines années et examinons ce qui est dans le meilleur intérêt de l'Europe : qu'est-ce qui contribuera à la prospérité et à la paix de cette famille que nous appelons " Europe " aujourd'hui ?

(Portons,porter,v+z1:p1p:y1p) (notre,,det+z1:ms:fs) (regard,,n+z1:ms) (au-delà,,prep+z) (des,du,prepdet+z1:mp:fp) (dix,,n+z1:ms:mp) (,,,pun) (quinze,,n+z1:ms:mp) (ou,,conjc+z1) (vingt-cinq,,det+dnum:mp:f) (prochaines,prochain,a+z1:fp) (années,année,n+z1:fp) (et,,conjc+z1) (examinons,examiner,v+z1:p1p:y1p) (ce,,pro+z1:3s:3p) (qui,,pro+z1:3s:3p) (est,être,v+z1:p3s) (dans,,prep+z1) (le,,det+z1:ms) (meilleur,,a+z1:ms) (intérêt,,n+z1:ms) (de,,prep+z1) (l',le,det+z1:ms:fs) (Europe,,np:ms:fs) (:,,pun) (que,,conjs+z1) (est-ce,,adv+z1) (qui,,pro+z1:3s:3p) (contribuera,contribuer,v+z1:f3s) (à,,prep+z1) (la,le,det+z1:fs)

(prospérité,,n+z1:fs) (et,,conjc+z1) (à,,prep+z1) (la paix,,gn+a1+z) (de,,prep+z1) (cette,ce,det+z1:fs) (famille,,n+z1:fs) (que,,pro+z1) (nous,,pro+z1:1p) (appelons,appeler,v+z1:p1p:y1p) (",,pun) (Europe,,np:ms:fs) (",,pun) (aujourd'hui,,adv+z1) (?,,pun)

Pour chaque unité, des informations concernant le lemme (si différent du mot), le genre, le nombre, la catégorie grammaticale et le temps pour les verbes conjugués sont disponibles.

5.2.2 Analyse syntaxique

Avant la phase d'analyse syntaxique, des outils de reconnaissance d'entités nommées sont utilisés. Actuellement DBpedia[33] (Auer et al., 2007) est utilisée pour repérer les entités nommées. La phase d'analyse ne consiste pas à faire une structure syntaxique complexe. Elle consiste juste à bien former les principaux syntagmes : nominal, verbal (le verbe et ses modificateurs), adjectival et adverbial. Les entités qui n'ont pas pu être bien repérées par DBpedia sont traitées au niveau syntaxique. Par exemple des mots « iPhone 5 » sont regroupés en une seule unité au lieu de les séparer comme le font certains analyseurs syntaxiques. Par exemple la phrases suivante : *Quel pays est trois fois plus grand que la Belgique et a plus de langues que le Kenya et le Canada ensemble?*

Après passage sur l'annotateur d'entités nommées nous obtenons la structure suivante :

[33] http://dbpedia.org/About

Quel pays est trois fois plus grand que la Belgique et a plus de langues que le Kenya et le Canada ensemble ?
S
GN
DET Quel;det+z1:ms;quel;
NN pays;n+z1:ms:mp;pays;
VER est;v+z1:p3s;être;
GN
DET trois;det+z1:mp:fp;trois;
NN fois;n+z1:fp;foi;
ADV plus;adv+advconjs+8;plus;;non_adjectif_modifier
ADJ grand;a+z1:ms;grand;
CJS que;conjs+z1;que;
GN
DET la;det+z1:fs;le;
NP Belgique ; np:ms:fs
;http://fr.dbpedia.org/resource/Belgique;DBpedia:Country,DBpedia:PopulatedPlace,DBpedia:Place,Schema:Place,Schema:Country,Freebase:/biology/breed_origin,Freebase:/biology,Freebase:/olympics/olympic_participating_country,Freebase:/olympics,Freebase:/military/military_combatant,Freebase:/military,Freebase:/sports/sports_team_location,Freebase:/sports,Freebase:/sports/sport_country,Freebase:/food/beer_country_region,Freebase:/food,Freebase:/media_common/netflix_genre,Freebase:/media_common,Freebase:/business/employer,Freebase:/business,Freebase:/book/book_subject,Freebase:/book,Freebase:/location/administrative_division,Freebase:/location,Freebase:/government/governmental_jurisdiction,Freebase:/government,Freebase:/royalty/kingdom,Freebase:/royalty,Freebase:/aviation/aircraft_owner,Freebase:/aviation,Freebase:/location/statistical_region,Freebase:/location/region,Freebase:/organization/organization_founder,Freebase:/organization,Freebase:/location/location,Freebase:/organization/organization_scope,Freebase:/organization/organization_member,Freebase:/government/government,Freebase:/business/business_location,Freebase:/location/country,Freebase:/location/dated_location,DBpedia:TopicalConcept
CJC et;conjc+z1;et;
VER a;v+z1:p3s;avoir;
ADV plus;adv+advconjs+8;plus;
PRP de;prep+z1;de;
GN
NN langues;n+z1:fp;langue;
PRO que;pro+z1;que;
GN
DET le;det+z1:ms;le;
NP
Kenya;np:ms:fs;Kenya;;http://fr.dbpedia.org/resource/Kenya;DBpedia:Country,DBpedia:PopulatedPlace,DBpedia:Place,Schema:Place,Schema:Country,Freebase:/food/beer_country_region,Freebase:/food,Freebase:/location/location,Freebase:/location,Freebase:/location/dated_location,Freebase:/sports/sport_country,Freebase:/sports,Freebase:/tv/tv_subject,Freebase:/tv,Freebase:/organization/organization_member,Freebase:/organization,Freebase:/film/film_location,Freebase:/film,Freebase:/sports/sports_team_location,Freebase:/book/book_subject,Freebase:/book,Freebase:/location/country,Freebase:/biology/breed_origin,Freebase:/biology,Freebase:/government/governmental_jurisdiction,Freebase:/government,Freebase:/location/statistical_region,Freebase:/periodicals/newspaper_circulation_area,Freebase:/periodicals,Freebase:/organization/organization_scope,Freebase:/business/board_member,Freebase:/business,Freebase:/business/business_location,Freebase:/olympics/olympic_participating_country,Freebase:/olympics,Freebase:/business/employer,DBpedia:TopicalConcept
CJC et;conjc+z1;et;
GN
DET le;det+z1:ms;le;
NP
Canada;np:ms:fs;Canada;;http://fr.dbpedia.org/resource/Canada;DBpedia:Country,DBpedia:PopulatedPlace,DBpedia:Place,Schema:Place,Schema:Country,Freebase:/symbols/coat_of_arms_bearer,Freebase:/symbols,Freebase:/award/award_discipline,Freebase:/award,Freebase:/film/film_location,Freebase:/film,Freebase:/royalty/kingdom,Freebase:/royalty,Freebase:/location/statistical_region,Freebase:/location,Freebase:/exhibitions/exhibition_sponsor,Freebase:/exhibitions,Freebase:/location/country,Freebase:/government/governmental_jurisdiction,Freebase:/government,Freebase:/location/administrative_division,Freebase:/location/location,Freebase:/olympics/olympic_participating_country,Freebase:/olympics,Freebase:/fictional_universe/fictional_setting,Freebase:/fictional_universe,Freebase:/cvg/computer_game_region,Freebase:/cvg,Freebase:/organization/organization_scope,Freebase:/organization,Freebase:/book/book_subject,Freebase:/book,Freebase:/organization/organization_member,Freebase:/biology/breed_origin,Freebase:/biology,Freebase:/business/employer,Freebase:/business,Freebase:/food/beer_country_region,Freebase:/food,Freebase:/film/film_subject,Freebase:/film/director,Freebase:/business/business_location,Freebase:/law/court_jurisdiction_area,Freebase:/law,Freebase:/military/military_combatant,Freebase:/military,Freebase:/periodicals/newspaper_circulation_area,Freebase:/periodicals,Freebase:/sports/sport_country,Freebase:/sports,Freebase:/location/dated_location,Freebase:/sports/sports_team_location,DBpedia:TopicalConcept
ADV ensemble;adv+padv+z1;ensemble;
PUN ?;pun;?;

S est la racine de l'arbre. Plusieurs informations sont récoltées sur les entités selon les différentes bases de données dans le nuage du Web de données.

5.2.3 Représentation intermédiaire et Génération de code

La génération de la forme logique s'inspire de (Soumana, 2010) paragraphe 2.2.11. Les modifications apportées concernent la prise en compte dans le graphe des traitements. Pour que le système soit robuste, il est important de pouvoir modéliser les traitements de au meme titre que le schema des dnnées. La représentation dans le graphe n'est pas obligatoire, mais elle offre une vue d'ensemble avec le schéma des données. Dans une question, les entités sont liées. Le fait de représenter la structure et les traitements dans un même graphe permet de mettre en évidence cette liaison ; il s'agit de la représentation des concepts et des opérations qui relèvent du domaine abstrait et de la mise en relation des sous graphes. Dans un graphe complexe, la projection d'une question est susceptible de concerner plusieurs sous ensembles du graphe qui n'ont pas nécessairement un lien direct.

6.3 Exemple

Pour illustrer les extensions faites à la méthode de (Soumana, 2010), nous prenons comme exemple la phrase suivante :

Quelle ville est quatre fois plus peuplée que Paris?

Si la question était *« Quelle est la ville la plus peuplée en France »* la majorité des systèmes sont en mesure de répondre car le langage formel comme SQL ou SPARQL fournissent des fonctions d'agrégation pour calculer automatiquement le résultat. Le système identifie donc les mots clés en langage naturel qui correspondent à la fonction d'agrégation. Un lexique du niveau *intraclasse* suffit puisque c'est une correspondance bijective. Mais comme le traitement est au delà d'une simple correspondance, la question peut être mal comprise. Si nous utilisons la base de données DBpedia[34], la question peut être représentée de la manière suivante figure 29. Les traits en gras représentent le traitement qui s'applique aux données.

[34] http://wiki.dbpedia.org/Ontology39?v=gkz

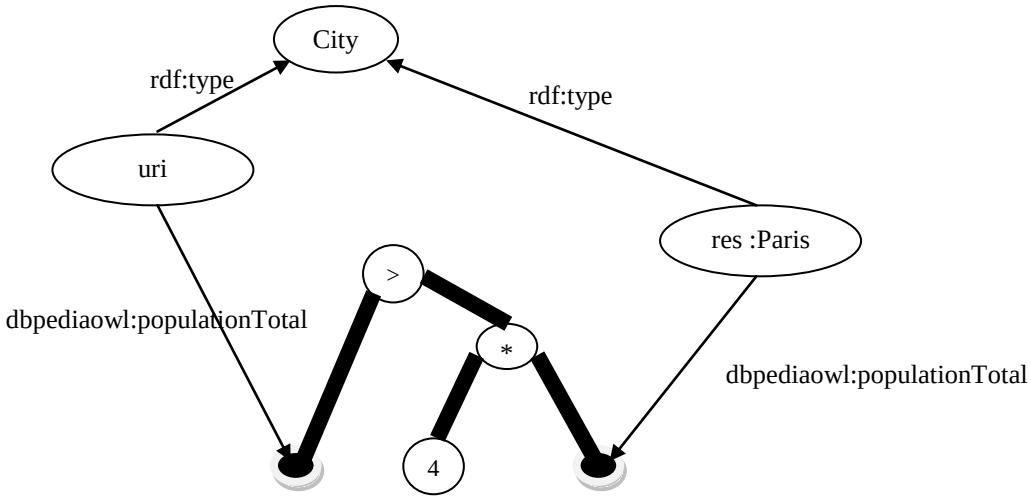


Figure 28. Représentation d'une question

La requête SPARQL correspondante peut être générée ainsi :

```
PREFIX res: <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT DISTINCT ?uri ?cityPopulation ?parisPopulation
WHERE {
      ?uri rdf:type dbpedia-owl:City.
      ?uri dbpedia-owl:populationTotal ?cityPopulation.
      res:Paris dbpedia-owl:populationTotal ?parisPopulation.
      FILTER(?cityPopulation>4*?parisPopulation)
}
```

En posant la même question à des moteurs comme Wolfram Alpha[35] ou Google, les résultats suivants sont obtenus (figure 30 et figure 31). Contrairement à Wolfram Alpha, nous ne visons pas des expressions mathématiques qui nécessitent une connaissance du domaine ou qui peuvent être interprétées seulement par un expert. Les traitements visés ici sont des traitements (ou des calculs) qui sont exprimés dans une syntaxe compréhensible par un locuteur de la langue et non seulement par un spécialiste du domaine. Si la requête SPARQL est exécutée sur les jeux de données de DBpedia[36], les résultats de la figure 32 sont obtenus.

[35] http://www.wolframalpha.com/

[36] http://dbpedia.org/sparql

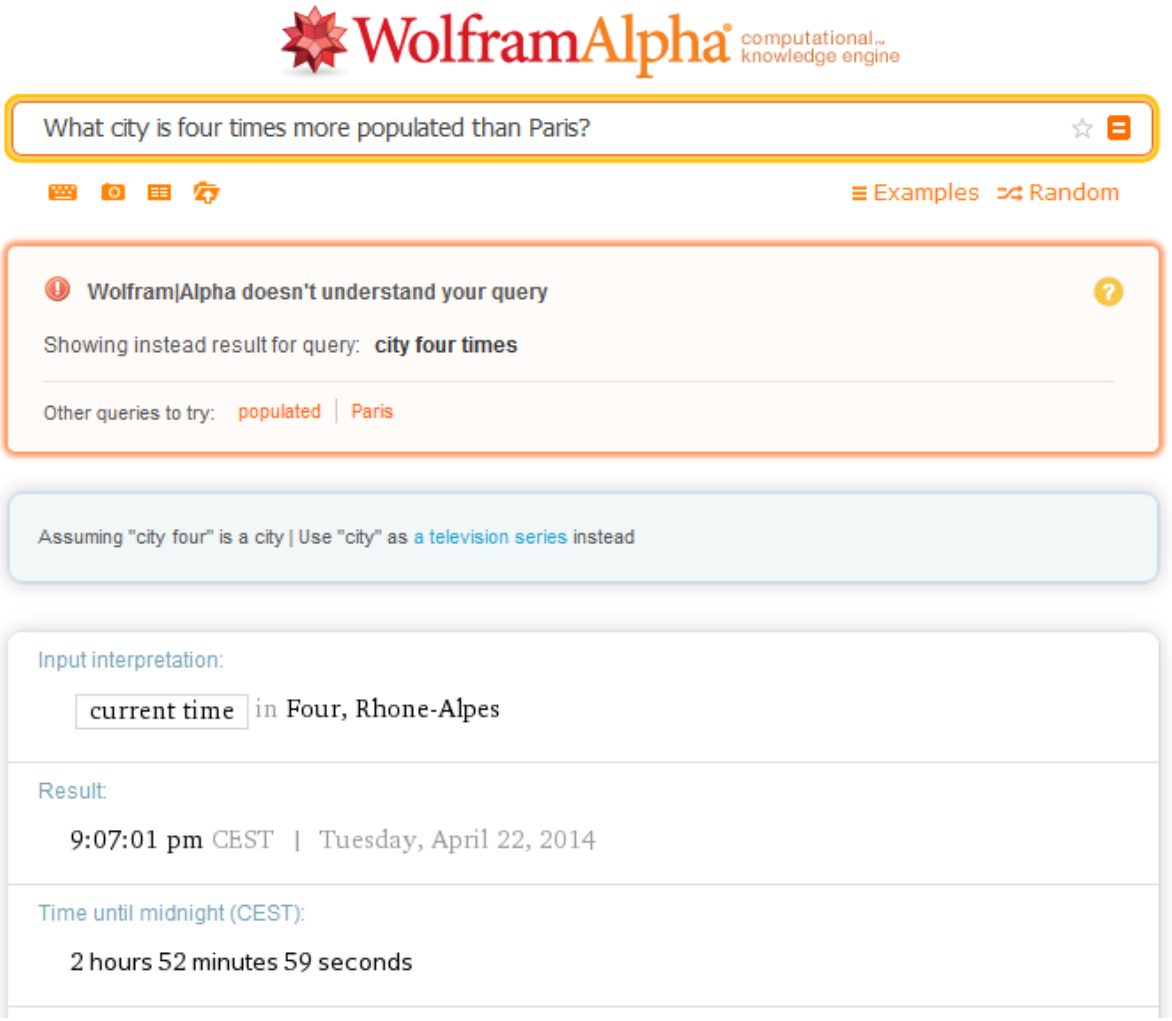
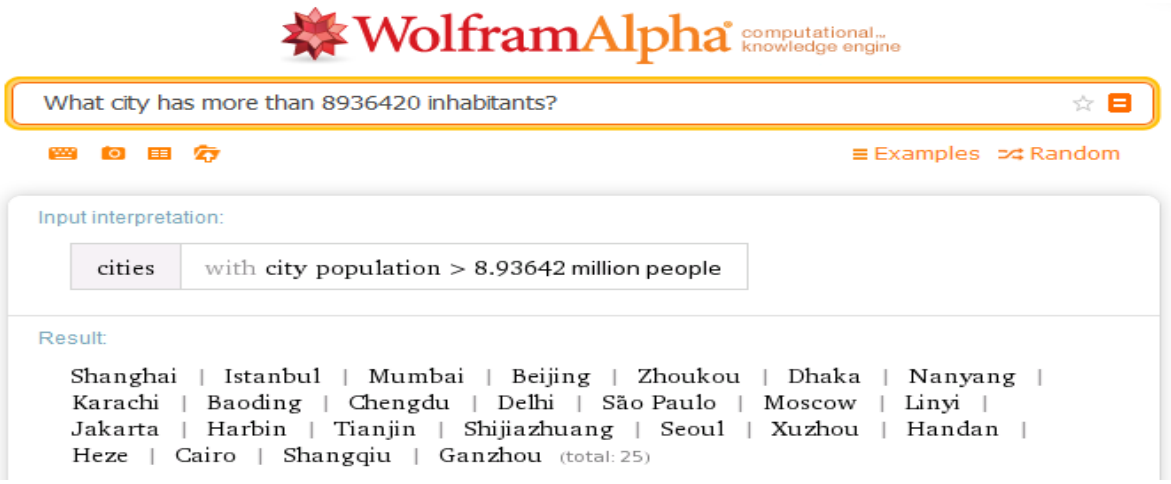


Figure 29. Exemple de réponse d'un moteur de recherche mathématique

Si nous considérons la population de Paris de 2 234 105 habitants et que nous posons la question d'une manière où il n'y a aucun traitement par exemple: « *Quelle ville a plus de 8936420 habitants ?*», Wolfram Alpha est capable de nous donner plusieurs réponses.



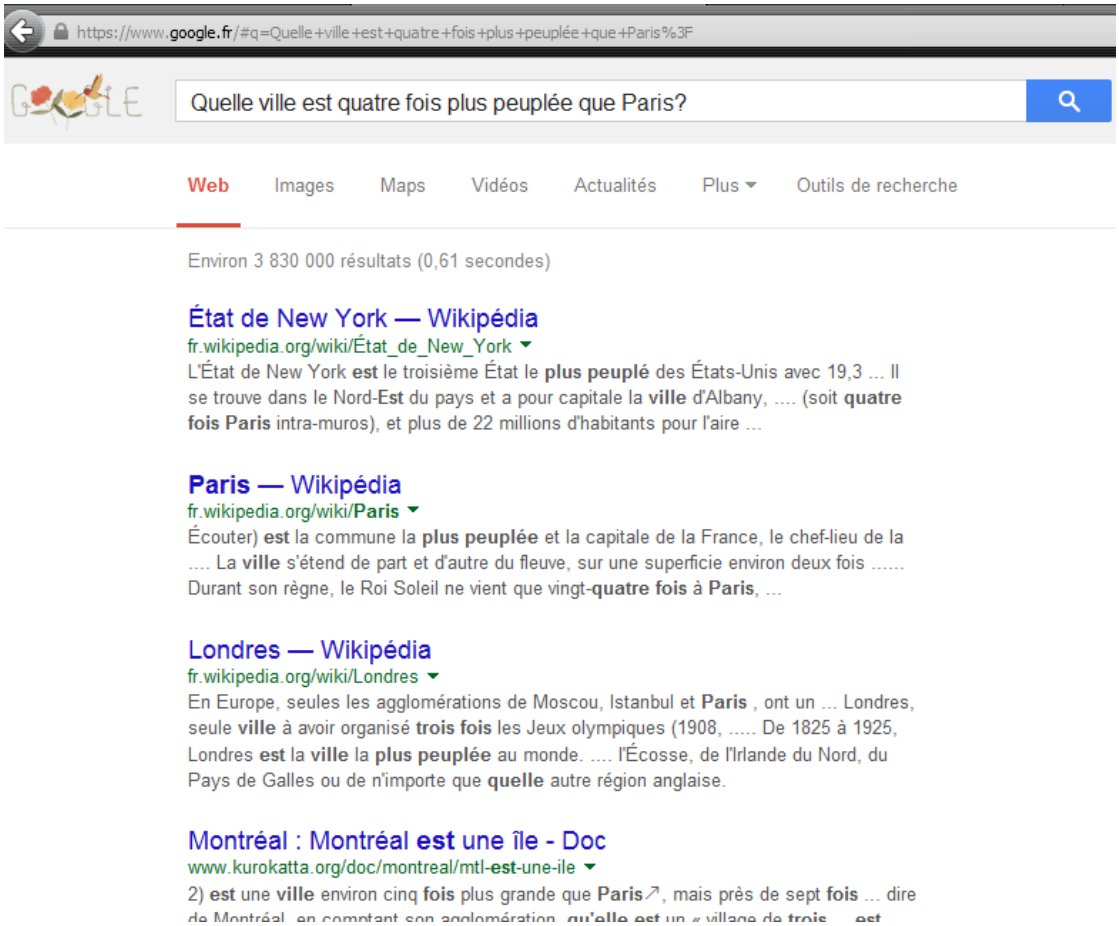


Figure 30. Exemple de réponse d'un moteur de recherche de mots-clés



uri	cityPopulation	parisPopulation
http://dbpedia.org/resource/Metro_Manila	21050000	2234105
http://dbpedia.org/resource/Baoding	11194379	2234105
http://dbpedia.org/resource/Fuyang	10142922	2234105
http://dbpedia.org/resource/Ganzhou	8969900	2234105
http://dbpedia.org/resource/Handan	9174679	2234105
http://dbpedia.org/resource/Lahore	11000000	2234105
http://dbpedia.org/resource/Linyi	10039400	2234105
http://dbpedia.org/resource/Nanyang,_Henan	10263006	2234105
http://dbpedia.org/resource/Shijiazhuang	10163788	2234105
http://dbpedia.org/resource/Suzhou	10465994	2234105
http://dbpedia.org/resource/Wenzhou	9122100	2234105
http://dbpedia.org/resource/Wuhan	10020000	2234105
http://dbpedia.org/resource/Zhoukou	8953172	2234105
http://dbpedia.org/resource/Chengdu	14047625	2234105
http://dbpedia.org/resource/Guangzhou	12700800	2234105
http://dbpedia.org/resource/Karachi	21200000	2234105
http://dbpedia.org/resource/Cairo	9120350	2234105
http://dbpedia.org/resource/Charlotte,_North_Carolina	7510872011	2234105
http://dbpedia.org/resource/Harbin	10635971	2234105
http://dbpedia.org/resource/Kinshasa	9046000	2234105
http://dbpedia.org/resource/Seoul	10581728	2234105
http://dbpedia.org/resource/Shenzhen	10357938	2234105
http://dbpedia.org/resource/Tokyo	13185502	2234105
http://dbpedia.org/resource/Mumbai	12478447	2234105

Figure 31. Exemple de réponse d'une base de données

Evidemment certaines réponses dépendent du fait que la donnée enregistrée dans la base de données peut être erronée. L'exemple illustre plus la prise en compte des traitements.

5.4 Conclusion

Ce chapitre illustre certaines caractéristiques du système. Les différentes parties du système ont été abordées notamment la phase de prétraitement et l'analyse syntaxique. La forme intermédiaire et la

génération du code ont été illustrées par un exemple. La prise en compte des traitements liés à la nature des données permet d'accroitre la robustesse du système.

CONCLUSION GENERALE ET PERSPECTIVES

Le développement des moyens de communication génère des volumes de données croissants. L'abondance des données s'est premièrement traduite dans les services informatiques par une inéquation des infrastructures de stockage au niveau des grands acteurs du Web comme Google, Facebook, Twitter ou Yahoo!. De nouveaux outils sont alors développés pour répondre non seulement à ce besoin de stockage mais aussi aux types de traitements qu'ils souhaitent faire de ces données. La disponibilité des supports de stockage à faible coût et des formats de données standards à favoriser le développement des bases de données dans divers domaines. Ces données sont de plus en plus interconnectées ; ceci implique également l'interconnexion des domaines puisque ces bases de données décrivent des domaines. Les applications informatiques tentent de tirer encore plus de profits de cette agréation de données.

En Traitement Automatiques des Langues, les applications qui dépendent de ces données (ou domaines) tentent de s'adapter également à ce phénomène. Les domaines qui étaient auparavant considérés comme des îles du fait qu'ils sont isolés sont en train de fusionner. Ceci change les méthodes de concevoir les applications qui utilisent ces données.

Dans cette thèse, nous nous sommes intéressé au cas des interfaces en langage naturel pour les bases de données, particulièrement à l'analyse de la question de l'utilisateur. Au delà de la diversité de domaines, il y a un intérêt croissant de la part des utilisateurs d'accéder à ces mines d'informations. Pour répondre aux besoins en information des utilisateurs, ces interfaces doivent être robustes. Plusieurs approches ont été proposées pour améliorer la robustesse des interfaces : approches orientées schéma, approches basées sur des langues contrôlées, approches basées sur des patrons (pattern), approches basées sur des triplets ou approches qui interagissent avec l'utilisateur. Ces approches présentent la particularité de se focaliser sur la base de données ou le langage formel d'interrogation pour comprendre la question de l'utilisateur. La manière dont ces données peuvent être utilisées hors du contexte d'une base de données n'a pas reçu une large investigation. Nous avons orienté notre travail vers la prise en compte de cette dimension. L'objectif n'est pas de résoudre les problèmes inhérents à la prise en compte de cette dimension mais de proposer une direction pour la recherche qui soit en mesure de concilier la diversité des données et la complexité linguistique.

La prise en compte de la manière dont les données peuvent être utilisées hors du contexte de la base de données se traduit par une décomposition du domaine en *domaine abstrait* et en *domaine concret*. Le domaine concret couvre la logique métier de la base de données.

Le domaine abstrait prend en charge l'usage des données hors du contexte d'une base de données en se basant sur leurs propriétés. Ces propriétés sont généralement décrites dans une certaine logique. Ces propriétés permettent d'identifier les opérations applicables aux données. Le plus souvent ces opérations interviennent dans les questions. Leur identification est donc nécessaire pour comprendre la question. La base de données et le langage formel ne suffisent pas à identifier ces opérations. Nous avons montré avec le temps (chapitre 4 en exemple) comment ces propriétés peuvent être prises en compte.

Au niveau de la base de données elle-même, l'identification des données enregistrées n'est pas aisée du fait que le langage naturel peut faire référence à ces données de plusieurs manières. Le fossé lexical entre le vocabulaire de l'utilisateur et l'ontologie est un frein au développement des interfaces en langage naturel. Plusieurs interfaces ontologie-lexique ont été proposées dont Ontolex[37] en cour de spécification au W3C (World Wide Web Consortium). Ontolex pourrait devenir un standard pour le Web de données pour des applications de traitement automatique des langues comme les Systèmes de Question-Réponse. Mais toutes ces interfaces ontologie-lexique se focalisent toujours sur l'ontologie (la base de données). Nous avons proposé une organisation du lexique à deux niveaux *: intraclasse* et *interclasse*. Le niveau *intraclasse* s'intéresse aux unités lexicales qui correspondent à des concepts de l'ontologie comme dans les interfaces ontologie-lexique. Le niveau interclasse s'intéresse plus aux unités lexicales définies par rapport à plusieurs classes du lexique pas nécessairement un concept enregistré dans la base de donnée.

Du fait de l'importance de l'analyse linguistique, nous avons développé l'infrastructure pour mener cette analyse. Il s'agit d'un étiqueteur et d'un analyseur syntaxique de surface.

Le prochain défi consistera à mener cette analyse linguistique.

[37] *www.w3.org/community/**ontolex**/*

Bibliographie :

Adafre S. F. et de Rijke M., (2005), Feature engineering and Post-Processing for temporal expression recognition using conditional random fields. In Proceedings ACL-2005 Workshop on Feature Engineering.

Ahn D., rantwijk J., et de Rijke M., (2007), A cascaded machine learning approach to interpreting temporal expressions. In Proceedings of NAACL HLT 2007, pages 420–427, Rochester, New York. Association for Computational Linguistics.

Ahn, D., Adafre, S. F. et de Rijke, M. (2005), Extracting temporal information from open domain text : A comparative exploration. Journal of Digital Information Management, 3(1):14–20.

Allen J., (1984), Towards a general theory of action and time. Artificial Intelligence, 23(2):123-154.

Androutsopoulos I., Ritchie G.D., and Thanisch P., (1995), Natural Language Interfaces to Databases - An Introduction, Natural Language Engineering, 1(1):29-81.

Angeli G., Manning C., Jurafsky D., (2012), Parsing Time: Learning to Interpret Time Expressions. North American Chapter of the Association for Computational Linguistics (NAACL) 2012.

Auer S., Bizer C., Lehmann J., Kobilarov G., Cyganiak R., Ives Z., (2007), DBpedia: A Nucleus for a Web of Open Data. In Aberer et al. (Eds.): The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11–15, 2007. Lecture Notes in Computer Science 4825 Springer 2007, ISBN 978–3-540–76297–3.

Battistelli D., Minel J. et Schwer S. (2006), Représentation des expressions calendaires dans les textes : vers une application à la lecture assistée de biographies. TAL, 47(3):11–37.

Ben Hazez S. and Minel J.-L., (2000), Designing tasks of identification of complex patterns used for text-filtering. In RIAO, pages 1558–1567.

Benthem V., J. F. A. K., (1983) , The Logic Of Time. Synthese Library vol. 156. D. Reidel

BERGMAN, M.K., (2001) White Paper: The Deep Web: Surfacing Hidden Value. Ann Arbor, MI: Scholarly Publishing Office, University of Michigan, University Library 7(1). DOI: http://dx.doi.org/10.3998/3336451.0007.104

Bernstein A., Kaufmann E., & Kaiser C., (2005b), Querying the Semantic Web with Ginseng: A Guided Input Natural Language Search Engine. 15th Workshop on Information Technology and Systems (WITS 2005), Las Vegas, Nevada, USA.

Bernstein A., Kaufmann E., Gohring, A., & Kiefer, C., (2005a), Querying ontologies: A controlled english interface for end-users, Proceedings. 4th International Semantic Web Conference (ISWC05), 112–126

Bernstein A., Kaufmann, E., Kaiser C., & Kiefer C., (2006), Ginseng: A Guided Input Natural Language Search Engine for Querying Ontologies. Jena User Conference Bristol, UK.

Berri J., Cartier E., Desclés J.-P., Jackiewicz A., and Minel J.-L., (1996), Filtrage automatique des texts, In Natural Language Processing and Industrial Applications, pages 22–35, Moncton, N.B., Canada.

Bittar A., (2008). Annotation des informations temporelles dans des textes en français. In Actes de RECITAL 2008, Avignon.

Bittar A., (2010), Building a TimeBank for French: A Reference Corpus Annotated According to the ISO-TimeML Standard, Université Paris Diderot, PhD Thesis.

Bizer C., Heath T., Berners-Lee T., Hausenblas M., Auer S., (2013), LDOW2013, Proceedings of the WWW2013 Workshop on Linked Data on the Web. CEUR-WS, Vol. 996, Rio de Janeiro, Brazil, May 2013.

Brown E., Epstein E., Murdock J . W., Fin T.H., (2013), Tools and Methods for Building Watson, IBM Re-search Report RC25356.

Bruchez R., (2013) Les bases de données NoSQL: Comprendre et mettre en œuvre. Paris : Eyrolles, 2013, 279 p.

Buitelaar P., Cimiano P., Mccrae J., Montiel-Ponsada E., Declerck T., (2011), Ontology Lexicalisation: The lemon Perspective, Ontology and Lexicon: new insights Workshop at 9th International Conference on Terminology and Artificial Intelligence (TIA 2011)

Buitelaar P., Sintek M., Kiesel M., (2006) A lexicon model for multilingual/multimedia Ontologies, In Proceedings of the 3rd European Semantic Web Conference (ESWC06), (2006)

Cardey S., (2013), Modelling Language, John Benjamins, Amsterdam/Philadelphia, ISBN 9789027249968, 204 p.

Cardey S., El Abed W., Greenfield P., (2001), Exploiting semantic methods for information filtering, in Actes du 3ème Colloque du Chapitre français de l'ISKO (International Society for Knowledge Organization), 5 et 6 juillet 2001, Université de Paris X : "Filtrage et résumé automatique de l'information sur les réseaux", pp. 219-225

Cardey S., Greenfield P., (2003), Disambiguating and Tagging Using Systemic Grammar, in Actes du 8th International Symposium on Social Communication, Santiago de Cuba, January 20-24, 2003, pp. 559-564.

Cardey S., Greenfield P., (2005), A Core Model of Systemic Linguistic Analysis, In Proceedings of the International Conference RANLP-2005 Recent Advances in Natural Language Processing, Borovets, Bulgaria, 21-23 September 2005, pp. 134-138.

Chambers N., (2013), NavyTime: Event and Time Ordering from Raw Text. In: SemEval'13: Proceedings the 7th International Workshop on Semantic Evaluation (together with *SEM 2013 and NAACL 2013) Atlanta, GA, USA, June 13-15, 2013.

Chang A.X., and Manning C.D., (2013), SUTime: Evaluation in TempEval-3. In Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)

Chao H.L., Chen C.H., Cardey S., (1999), Traitement automatique de la sémantique floue dans l'interrogation de base de données en langue naturel, In BULAG n°24, Université de Franche-Comté, Besançon, pp. 187-208.

Cimiano P., Buitelaar P., Mccrae J., Sintek M., (2011), LexInfo: A declarative model for the lexicon-ontology interface. J. Web Sem. 9(1): 29-51 (2011)

Cimiano P., Erdmann M., Ladwig G., (2007) Corpus-based pattern induction for a knowledge-based question answering approach, In Proceedings of the 1st IEEE International Conference on Semantic Computing (ICSC), 2007, pp. 671–678.

Cimiano P., Haase P., Heizmann J., Mantel M., Studer R., (2008), Towards portable natural language interfaces to knowledge bases - The case of the ORAKEL system, Data Knowl. Eng. 65(2): 325-354 (2008)

Cimiano P., Haase P., Heizmann J., Studer R., (2008), Towards Portable Natural Language Interfaces to Knowledge Bases: The Case of the ORAKEL System, Data Knowledge Engineering (DKE), 65(2), pp. 325-254, 2008.

Cimiano P., Haase P., Herold M., Mantel M., Buitelaar P., (2007), LexOnto: A Model for Ontology Lexicons for Ontology-based NLP, In Proceedings of the OntoLex07 Workshop held in conjunction with ISWC'07, 2007.

Codd E. F., (1970), A relational model of data for large shared data banks, ACM 13 (1970), no. 6, 377-387.

Collins M., (1999), Head-driven Statistical Models for Natural Language Parsing. PhD thesis, University of Pennsylvania.

Cunningham H., Maynard D., Bontcheva K., Tablan V., (2002), GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia

Damljanovic D., (2011), PhD Thesis. Natural Language Interfaces to Conceptual Models. The University of Sheffield. 2011

Damljanovic D., Agatonovic M., and Cunningham H., (2011), FREyA: an Interactive Way of Querying Linked Data using Natural Language .In Proceedings of 1st Workshop on Question Answering over Linked Data (QALD-1), Collocated with the 8th Extended Semantic Web Conference (ESWC 2011), May 30th, 2011, Heraklion, Greece, page 10-23, Heraklion, Greece, 2011.

Desclés, J., Cartier, E., Jackiewicz, A. Et Minel, J. (1997), Textual processing and contextual exploration method. In CONTEXT'97, pages 189–197, Rio de Janeiro.

El Abed W., (2001), Méta modèle sémantique et noyau informatique pour l'interrogation multilingue des bases de données en langue naturelle (théorie et application), Thèse de doctorat sous la direction de Sylviane Cardey, Centre de Recherche Lucien Tesnière, Université de Franche-Comté, Besançon

Ferro L., Gerber L., Mani I., Sundheim B. et Wilson G. (2005), TIDES 2005 standard for the annotation of temporal expressions. Rapport technique, MITRE.

Ferro L., Gerber L., Mani I., Sundheim, B., Wilson G., (2005), TIDES 2005 Standard for the

Annotation of Temporal Expressions. The MITRE Corporation.

Ferrucci D. A., Brown E.W., Chu-Carroll J., Fan J., Gondek D., Kalyanpur A., Lally A., Murdock J. W., Nyberg E., Prager J.M., Schlaefer N., and Welty C. A., (2010), Building watson: An overview of the DEEPQA project. AI Magazine, 31(3):59–79, 2010.

Filannino M., Brown G., Nenadic G., (2013), ManTIME: Temporal expression identification and normalization in the TempEval-3 challenge. In: Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). Atlanta, Georgia, USA: Association for Computational Linguistics

Filatova E., Et Hovy, E., (2001), Assigning Time-Stamps to Event-Clauses. In Proceedings of the 2001 ACL Workshop on Temporal and Spatial Information Processing, pages 1–8, Toulouse, France. Association for Computational Linguistics.

Freitas A., Curry E., Oliveira J. G., O'riain S., (2012), Querying Heterogeneous Datasets on the Linked Data Web: Challenges, Approaches, and Trends. IEEE Internet Computing 16(1): 24-33 (2012)

Freitas A., João G.O., O'Riain S., Curry E., Carlos Pereira da Silva J., (2011) "Treo: Combining Entity-Search, Spreading Activation and Semantic Relatedness for Querying Linked Data", In 1st Workshop on Question Answering over Linked Data (QALD-1), 2011

Freska C., (1992) ,Temporal reasoning based on semi-intervals. AI Journal 54 (1-2)199-227

Frost R. A., Agboola W., Matthews E., Donais J. A., (2014) An Event-Driven Approach for Querying Graph-Structured Data Using Natural Language. EDBT/ICDT Workshops 2014: 192-199

Grosz B.J., Appelt, D.E., Martin P.A., and Pereira, F.C.N., (1987), TEAM: An Experiment in the Design of Transportable Natural-Language Interfaces. Artifi-cial Intelligence, 32:173-243.

Hamel M.P. Et Marguerit D., (2013) Analyse des Big Data : Quels usages, quels défis ? , Commissariat général à la stratégie et à la prospective, 11/2013, n° 8.

Hayes P., (1995), A Catalog of Temporal Theories, Pat Hayes; Tech report UIUC-BI-AI-96-01, University of Illinois 1995

He S., Liu S., Liu S., Chen Y., Zhou G., Liu K., Zhao J., (2013), CASIA@QALD-3: A Question Answering System over Linked Data. In: Proceedings of the 4th Conference and Labs of the Evaluation Forum (CLEF2013), Valencia, Spain.

Heintzelman N.H., Taylor R.J., Simonsen L., Lustig R., Anderko D., Haythornthwaite J.A., Childs L.C., Bova G. S. (2013), Longitudinal analysis of pain in patients with metastatic prostate cancer using natural language processing of medical record text. JAMIA 20(5): 898-905 (2013).

Kalyanpur A., Boguraev B., Patwardhan S., Murdock J.W., Lally A., Welty C., Prager J., Coppola B., Fokoue A. 2012. Structured Data and Inference in DeepQA, IBM Journal of Research and Development 56(3/4), 10:1 - 10:14, IBM

Kaufmann E., Bernstein A., & Fischer L., (2007), NLP-Reduce: A" naive" but domain-independent natural language interface for querying ontologies, In proceedings of 4th European Semantic Web Conference (ESWC2007), Innsbruck, Austria.

Kaufmann E., Bernstein A., & Zumstein R., (2006), Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs. International Semantic Web Conference (ISWC2006), Athens, Geogia, USA.

Kevers L., (2011), Accès sémantique aux bases de données documentaires. Techniques symboliques de traitement automatique du langage pour l'indexation thématique et l'extraction d'informations temporelles, thèse de doctorat en Langues et lettres, Université catholique de Louvain, 31-01-2011

Koehn P. (2005). Europarl: A parallel corpus for statistical machine translation. Conference Proceedings: the Tenth Machine Translation Summit. Phuket, pp. 79-86.

Kolomiyets O., & Moens, M.F., (2009), Meeting TempEval-2: Shallow Approach for Temporal Tagging. In *Proceedings of the NAACL-HLT Workshop on Semantic Evaluations: Recent Advances and Future Directions*. Association for Computational Linguistics.

Kolomiyets, O. and Moens M.F., (2010), KUL: recognition and normalization of temporal expressions. In Proceedings of the 5th International Workshop on Semantic Evaluation, Sem-Eval '10, pages 325–328.

Le Parc-Lacayrelle, A., Gaio, M. et Sallaberry, C. (2007). La composante temps dans l'information géographique textuelle. Document numérique, 10(2007/2):129–148.

Li X. And Roth D., (2002) Learning Question Classifiers'. In: Proceedings of the 19th International Conference on Computational Linguistics.

Li Y., Yang H., Jagadish H.V., (2004) NaLIX: an interactive natural language interface for querying XML, Proceedings of the 2004 ACM SIGMOD international conference on Management of data, pp 900-902, Baltimore, Maryland.

Li Y., Yu C., Jadish H.V., (2005), Schema-Free XQuery, In VLDB.

Lin D., (1998), Dependency-based evaluation of MINIPAR, In Workshop on the Evaluation of Parsing Systems, 1998.

Llorens H., Saquete E., Navarro-Colorado B. (2010), TIPSem (English and Spanish): Evaluating CRFs and Semantic Roles in TempEval-2. In: Proceedings of the 5th internationalWorkshop on Semantic evaluation pp. 284-291. ACL (2010).

Lopez V., Motta E. (2004), Ontology-Driven Question Answering in AquaLog. NLDB 2004: 89-102

Lopez V., Unger C., Cimiano P., Motta E. (2013) Evaluating question answering over linked data, J. Web Sem. 21: 3-13 (2013)

Lopez V., Uren V., Motta E., Pasin M., (2010), AquaLog: An ontology-driven question answering system for organizational semantic intranets,Journal of web semantic, online : http://technologies.kmi.open.ac.uk/aqualog/aqualog_journal.pdf, 2010.

Lopez V., Uren V., Sabou M., Motta E., (2011), Is question answering fit for the semantic web?: a survey Semantic Web 2(2), 125--155, IOS Press, 2011

Lopez, V., Pasin, M., & Motta, E. (2005), AquaLog: An Ontology-portable Question Answering System for the Semantic Web. 2nd European Semantic Web Conference (ESWC 2005), Heraklion, Greece.

Mani I. Et Wilson G., (2000), Robust temporal processing of news. In Proceedings of the 38th Annual Meeting on Association for Computational Linguistics, pages 69–76, Hong Kong. Association for Computational Linguistics

Martin F. P., An algorithm for suffix stripping, Program, 14(3):130–137, 1980.

Martin K.,(1980), Algorithm Schemata and Data Structure in Syntactic Processing. Report CSL-80-

12, Xerox PARC, Palo Alto, California.

Maurel D., Et Mohri M., (1994), French temporal expressions : Recognition, parsing and real computation. In Reflections on the Future of text, pages 33–41, Waterloo, Ontario, Canada.

Metzler D. And Croft W.B., (2005) Analysis of Statistical Question Classification for Fact-based Questions, In Information Retrieval, 8(3), 481-504, 2005

Miller G. A., Beckwith R., Fellbaum C., Gross D., & Miller, K. J., (2004), Introduction to WordNet: An On-line Lexical Database. International Journal of Lexicography, 3(4), pp. 235-244

Minock M., (2010), C-Phrase: A System for Building Robust Natural Language Interfaces to Databases, Journal of Data and Knowledge Engineering (DKE), Elsevier, 69(3):290-302, 2010.

Minock M., Olofsson P., and Näslund A., (2008), Towards Building Robust Natural Language Interfaces to Databases, proceedings of the International Conference on Applications of Natural Language to Information Systems (NLDB), pages 187-198. June 2008

Muller P. et Tannier X., (2004), Annotating and measuring temporal relations in texts, In Proceedings of the 20th international conference on Computational Linguistics, Geneva, Switzerland. Association for Computational Linguistics.

Nadeau D., Sekine S., (2007), A survey of named entity recognition and classification, Journal of Linguisticae Investigationes 30:1 ; 2007

Narisawa K., Watanabe Y., Mizuno J., Okazaki N., Inui K., (2013), Is a 204 cm Man Tall or Small ? Acquisition of Numerical Common Sense from the Web. ACL (1) 2013: 382-391.

Negri M., Magnini B., and Kouylekov M.O., (2008), Detecting expected answer relations through textual entailment. In 9th International Conference on Intelligent Text Processing and Computational Linguistics, pages, Heidelberg, Germany, 2008. Springer, pp. 532-543

Ou S., Orasan C., Mekhaldi D. and Hasler L., (2008) Automatic Question Pattern Generation for Ontology-based Question Answering. In Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference (FLAIRS2008). Menlo Park, CA: AAAI Press., pp. 183-188.

Parent G., Gagnon M., Et Muller P., (2008), Annotation d’expressions temporelles et d’événements

en français. In Actes de la 15e Conférence sur le Traitement Automatique des Langues Naturelles, pages 39–48, Avignon, France.

Popescu A.M., Armanasu A., Etzioni O., Ko D., Yates A., (2004), Modern Natural Language Interfaces to Databases: Composing Statistical Parsing with Semantic Tractability, COLING 2004.

Pustejovsky J., and Hanks P., and Sauri R. and . See A., and Gaizauskas R., and Setzer A., and Radev D., and Sundheim B., and Day D., and Ferro L., and Lazo M., (2003), The TIMEBANK Corpus, In Proceedings of Corpus Linguistics 2003, 647-656, 2003

Pustejovsky J., Castaño J., Ingria R., Saurí R., Gaizauskas R., Setzer A., and Katz G., (2003), TimeML: Robust Specification of Event and Temporal Expressions in Text. In Proceedings of IWCS-5, Fifth International Workshop on Computational Semantics.

Pustejovsky J., Stubbs A., (2012), Natural Language Annotation for Machine Learning - a Guide to Corpus-Building for Applications. O'Reilly 2012, ISBN 978-1-449-30666-3, pp. I-XIV, 1-326

Rick C., (2011) Scalable SQL and NoSQL Data Stores.

Rodolfo A., Pazos R., Juan J. Gonzalez B., Antonio Aguirre Lam M., (2011) Semantic Model for Improving the Performance of Natural Language Interfaces to Databases. MICAI (1) 2011: 277-290.

Soumana I., (2010), A Natural Language Interface for SPARQL by means of Hierarchical Categorisation, in Natural Language Processing and Human Language Technology 2010, BULAG n°34, PUFC, ISSN 0758 6787, ISBN 978-2-84867-312-7, pp. 169-185

Soumana I., Cardey S., Greenfield P., (2012a), *Use of Natural Language Interfaces and Open Data in Local Infomediation*, in Proceedings of IEEE International Conference on Management and Service Science (MASS), Shanghai, China, 10-12 August 2012, 4 pages, CD-ROM.

Soumana I., Cardey S., Greenfield P., (2012b), *Generalized Named Entity Recognition and Measure Theory*, submitted to CHIST-ERA Conference 2012, July 5-6, 2012, Edinburgh, Scotland

Soumana I., Cardey S., Greenfield P., (2013), A Lexicon Design for Ontology-Based Question Answering, KEOD 2013: 5th International Conference on Knowledge Engineering and Ontology Development. 19-22 September 2013,Vilamoura, Portugal.

Soumana I., Cardey S., Greenfield P., (2013b), Etiquetage et Désambigüisation par Classification Basés sur l'Approche Microsystémique, in Colloque International en Traductologie et TAL, Oran, Algeria

Spivack N., The road to semantic search the twine.com story. http://www.novaspivack.com/uncategorized/the-road-to-semantic-search-the-twine-com-story, December 2009.

Steven B., (2013), ClearTK-TimeML: A minimalist approach to TempEval 2013, In Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013). Atlanta, Georgia, USA: Association for Computational Linguistics, pp. 10-14.

Strötgen J., Zell J., and Gertz M., (2013), HeidelTime: Tuning English and Developing Spanish Resources for TempEval-3. In: SemEval'13: Proceedings the 7th International Workshop on Semantic Evaluation (together with *SEM 2013 and NAACL 2013) Atlanta, GA, USA, June 13-15, 2013.

Uno T., (1997), Algorithms for Enumerating All Perfect, Maximun and Maximal Matching in Bipartite Graphs, International Symposium on Algorithm and Computation (ISAAC 1997), page 92-101.

UzZaman N., Llorens H., Derczynski L., Allen J., Verhagen M., Pustejovsky J., (2013), SemEval-2013 Task 1: TempEval-3: Evaluating Events, Time Expressions, and Temporal Relations, In Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval 2013)

Vazov, N., (2001), A system for extraction of temporal expressions from french texts based on syntactic and semantic constraints. In Proceedings of the workshop on Temporal and spatial information processing, volume 13, pages 1–8. Association for Computational Linguistics.

Vicente-Díez, M. T., Samy, D. Et Martínez, P. (2008). An empirical approach to a preliminary successful identification and resolution of temporal expressions in spanish news corpora. *In Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Wang C., Xiong M., Zhou Q., & Yu Y., (2007), PANTO: A Portable Natural Language Interface to Ontologies. European Semantic Web Conference (ESWC2007), Innsbruck, Austria.

Weiser S. (2010), Repérage et typage d'expressions temporelles pour l'annotation sémantique automatique de pages Web. Application au e-tourisme. Thèse de doctorat, Université Paris Ouest Nanterre La Défense, Paris, France.

Wilson G., Mani I., Sundheim B. Et Ferro L., (2001), A multilingual approach to annotating and extracting temporal information. In Proceedings of the workshop on Temporal and spatial information processing - Volume 13, pages 1–7. Association for Computational Linguistics.

Woods W., Kaplan R., Webber B., (1972), The Lunar Sciences Natural Language Information System: Final Report. Technical report, Bolt Beranek and Newman Inc., Cambridge, Massachusetts

Yahya M., Berberich K., Elbassuoni S., and Weikum G., (2013), Robust Question Answering over the Web of Linked Data CIKM 2013

Yahya M., Berberich K., Elbassuoni S., Ramanath M., Tresp V., Weikum G., (2012), Natural Language Questions for the Web of Data. In: Proc. of EMNLP'12(2012)

Zavarella V., Tanev H., (2013), FSS-TimEx for tempeval-3: Extracting temporal information from text. Second Joint Conference on Lexical and Computational Semantics (* SEM) 2, 58-63.