ROMIE, une approche d'alignement d'ontologies à base d'instances

Abdeltif Elbyed

▶ **To cite this version:**

Abdeltif Elbyed. ROMIE, une approche d'alignement d'ontologies à base d'instances. Autre [cs.OH]. Institut National des Télécommunications, 2009. Français. NNT : 2009TELE0014 . tel-00541874

HAL Id: tel-00541874

https://theses.hal.science/tel-00541874v1

Submitted on 1 Dec 2010

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Thèse de doctorat de l'INSTITUT NATIONAL DES TELECOMMUNICATIONS dans le cadre de l'école doctorale S&I en co-accréditation avec l'UNIVERSITE D'EVRY-VAL D'ESSONNE

Spécialité :
Informatique

Par
M Abdeltif ELBYED

Thèse présentée pour l'obtention du grade de Docteur de l'INSTITUT NATIONAL DES TELECOMMUNICATIONS

ROMIE, une approche d'alignement d'ontologies à base d'instances

Soutenue le 16 Octobre 2009 devant le jury composé de :

Rapporteurs :	
Mme. Chantal Reynaud	professeur à l'université Paris-Sud & INRIA Saclay
Mr. Jean Charlet	HDR, chargé de mission AP-HP, professeur associé à l'ECP INSERM
Examinateurs :	
Mr. Djamal Benslimane	Professeur à l'université Claude Bernard Lyon
Mme. Christine Froidevaux	Professeur à l'université Paris-Sud
Mr. Bruno Defude	Professeur à Télécom SudParis (Directeur de thèse)
Mme. Amel Bouzeghoub	Maître de conférences à Télécom SudParis (co-encadrant)
Mme. Fariza Tahi	Maître de conférences à l'université d'Evry-Val d'Essonne (co-encadrant)

Thèse n° 2009TELE0014

Toutes les lettres ne sauraient trouver les mots qu'il faut…

Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance…

Aussi c'est tout simplement que…

Je dédie cette thèse ...

A Mes parents,

A mes frères et sœurs,

A Ma femme Hind

Remerciements

Je tiens vivement à remercier toutes les personnes qui ont participé de près ou de loin à l'élaboration de ce travail, en particulier mes encadrantes Amel Bouzeghoub et Fariza Tahi.

Je tiens à exprimer ma gratitude envers mon directeur de thèse Bruno Defude pour la qualité de ses conseils, sa disponibilité ainsi que le degré de responsabilisation de son encadrement qui m'a permis de développer mon goût pour la recherche.

Je remercie également Guy Bernard pour avoir facilité mon inscription en thèse et pour son aide.

Je suis profondément reconnaissant à Amel Bouzeghoub pour le temps qu'elle a investi pour suivre et analyser les résultats de ce travail ainsi que pour les nombreuses discussions enrichissantes.

Je tiens à remercier Fariza Tahi pour m'avoir aidé et soutenu tout le long de cette thèse et pour ses conseils et son expertise dans le domaine biomédical.

Je tiens à remercier mes deux rapporteurs Mme Chantal Reynaud et Mr Jean Charlet pour l'intérêt qu'ils ont porté à ce travail en acceptant de le juger et pour le temps qu'ils ont consacré pour la lecture de ce mémoire. Je remercie également Mme Christine Froidevaux et Mr Djamal Benslimane de m'avoir honoré en acceptant d'examiner la thèse.

Je remercie les enseignants du département informatique de Télécom SudParis et les enseignants du laboratoire IBISC pour leur accueil, leur bonne humeur et leur convivialité, qui ont rendu plus légères les nombreuses heures de cours que j'ai dû assurer durant ma thèse.

Je salue également les membres du Laboratoire d'Informatique de Télécom SudParis et du laboratoire IBSC et en particulier les thésards et les stagiaires du département informatique.

Enfin, je remercie mes parents, ma femme et mon frère pour leur soutien et leur amour, depuis toujours.

Abstract

System interoperability is an important issue, widely recognized in information technology intensive organizations and in the research community of information systems. The wide adoption of the World Wide Web to access and distribute information further stresses the need for system interoperability. Initiative solutions like the Semantic Web facilitate the localization and the integration of the data in a more intelligent way via the use of ontologies. The Semantic Web offers a compelling vision, yet it raises a number of research challenges. One of the key challenges is to compare and map different ontologies, which evidently appears in integration tasks.

The main goal of the work is to introduce a method for finding semantic correspondences among ontologies with the intention to support interoperability of Information Systems. The approach brings together syntactic, linguistic, structural and semantic (based on instance information) matching methods in order to provide a semi-automatic mapping. The approach consists of two phases: semantic enrichment phase and mapping phase. The enrichment phase is based on the analysis of the information developed by the ontologies (like web resources, data, documents, etc.) and that are associated to the concepts in the ontologies. Our intuition is that this information as well as the relations that can exist between them is used in semantic enrichment between the concepts. At the end of enrichment phase, the ontology contains more semantic relations between its concepts that will be exploited in the second phase. The phase of mapping takes two enriched ontologies and calculates the similarity between the couples of concepts. A process of filtering enables us to automatically reduce the number of false relations. The validation of the correspondences is a direct interactive process (with an expert) or indirect (by measuring the satisfaction level of the user). The approach has been implemented in a prototype system called ROMIE (***Resource based Ontology Mapping within and Interactive and Extensible environment***). It was tested and evaluated in two applications: a biomedical application and technology enhanced learning (or e-learning) domain application.

Keywords: semantic web, mapping and alignment ontologies, biomedical domain, e-learning, multi agent system.

Résumé

L'interopérabilité sémantique est une question importante, largement identifiée dans les technologies d'organisation et de l'information et dans la communauté de recherche en systèmes d'information. L'adoption large du Web afin d'accéder à des informations distribuées nécessite l'interopérabilité des systèmes qui gèrent ces informations. Des solutions et réflexions comme le Web Sémantique facilitent la localisation et l'intégration des données d'une manière plus intelligente par l'intermédiaire des ontologies. Il offre une vision plus sémantique et compréhensible du web. Pourtant, il soulève un certain nombre de défis de recherche. Un des principaux défis est de comparer et aligner les différentes ontologies qui apparaissent dans des tâches d'intégration.

Le principal objectif de cette thèse est de proposer une approche d'alignement pour identifier les liens de correspondance entre des ontologies. Notre approche combine les techniques et les méthodes d'appariement linguistiques, syntaxiques, structurelles ou encore sémantiques (basées sur les instances). Elle se compose de deux phases principales : la phase d'enrichissement sémantique des ontologies à comparer et la phase d'alignement ou de *mapping*. La phase d'enrichissement est basée sur l'analyse des informations que les ontologies développent (des ressources web, des données, des documents, etc.) et qui sont associés aux concepts de l'ontologie. Notre intuition est que ces informations ainsi que les relations qui peuvent exister entre elles participent à l'enrichissement sémantique entre les concepts. A l'issue de la phase d'enrichissement, une ontologie contient plus de relations sémantiques entre les concepts qui seront exploitées dans la deuxième phase. La phase de *mapping* prend deux ontologies enrichies et calcule la similarité entre les couples de concepts. Un processus de filtrage nous permet de réduire automatiquement le nombre de fausses relations. La validation des correspondances est un processus interactif direct (avec un expert) ou indirect (en mesurant le degré de satisfaction de l'utilisateur). Notre approche a donné lieu à un système de *mapping* appelé ROMIE (***Resource based Ontology Mapping within an Interactive and Extensible environment***). Il a été expérimenté et évalué dans deux différentes applications : une application biomédicale et une application dans le domaine de l'apprentissage enrichi par les technologies (ou *e-learning*).

Mots clés : web sémantique, alignement et *mapping* d'ontologies, domaine biomédical, e-learning, systèmes multi-agent.

TABLE DES MATIERES

INTRODUCTION **17**

CHAPITRE I : ONTOLOGIES ET DOMAINES D'APPLICATION CIBLES **27**

I.1 LE WEB SÉMANTIQUE **27**

I.2 ONTOLOGIES **29**

I.2.1 ROLE DES ONTOLOGIES 29
I.2.2 LA NOTION D'ONTOLOGIE 30
I.2.3 L'ORIGINE DES ONTOLOGIES 31
I.2.4 LES CONSTITUANTS D'UNE ONTOLOGIE 32
I.2.5 LES LANGAGES D'ONTOLOGIES 36
I.2.6 LES OUTILS DE CONSTRUCTION D'ONTOLOGIES 40

I.3 DOMAINES D'APPLICATION CIBLES **42**

I.3.1 LE DOMAINE DE L'APPRENTISSAGE ENRICHI PAR LES TECHNOLOGIES 42
I.3.2 LE DOMAINE BIOMEDICAL 45

I.4 CONCLUSION **49**

CHAPITRE II : ETAT DE L'ART **51**

II.1 MISE EN CORRESPONDANCE D'ONTOLOGIES **51**

II.1.1 TERMINOLOGIE 51
II.1.2 LES APPROCHES DE L'INTEROPERABILITE SEMANTIQUE 52

II.2 LES PROBLEMATIQUES ET LES DIFFICULTES DE L'APPARIEMENT (*MATCHING*) **54**

II.2.1 LES TECHNIQUES DE L'APPARIEMENT (MATCHING) 55
II.2.2 CLASSIFICATION DES METHODES DE *MATCHING* 55
II.2.3 LES STRATEGIES DE COMBINAISON DES *MATCHERS* 58

II.3 CLASSIFICATION DES SYSTEMES DE *MAPPING* SUIVANT LE MODE D'INTEGRATION 59

II.3.1 LE *MAPPING* ENTRE UNE ONTOLOGIE GLOBALE ET DES ONTOLOGIES LOCALES 60

II.3.2 LE *MAPPING* D'ONTOLOGIES LOCALES 61

II.3.3 *MAPPING* D'ONTOLOGIES POUR LA FUSION ET L'ALIGNEMENT 63

II.4 CLASSIFICATION DES METHODES, OUTILS ET FRAMEWORK EXISTANTS 64

II.4.1 LES TECHNIQUES DE *MATCHING* SUPPORTEES 65

II.4.2 LANGAGES D'ONTOLOGIES ET DE *MAPPING* 66

II.4.3 IMPLEMENTATION ET EXPERIMENTATION 67

II.5 ANALYSE ET SYNTHESE 68

CHAPITRE III : OMIE – UNE APPROCHE POUR LE *MAPPING* D'ONTOLOGIES 71

III.1 LE SYSTEME OMIE 71

III.1.1 INTRODUCTION 71

III.1.2 LES CAS D'UTILISATIONS ET LES FONCTIONNALITES 72

III.2 LE PROCESSUS DE MAPPING DU SYSTEME OMIE 74

III.2.1 PRINCIPE GENERAL DE L'ALGORITHME DE *MAPPING* 74

III.2.2 LE PROCESSUS DE *MATCHING* 76

III.2.3 LE PROCESSUS DE FILTRAGE 89

III.2.4 PROCESSUS DE VALIDATION 92

III.3 IMPLEMENTATION DU SYSTEME OMIE AVEC LA TECHNOLOGIE AGENT 95

III.3.1 LES SYSTEMES MULTI-AGENTS (SMA) 96

III.3.2 L'ARCHITECTURE AGENT D'OMIE 97

III.3.3 COMPORTEMENT DES AGENTS 98

III.3.4 COORDINATION DES AGENTS 106

III.4 CONCLUSION ET CONTRIBUTIONS DU SYSTEME OMIE 107

CHAPITRE IV : SYSTEME ROMIE (OMIE A BASE DE RESSOURCES) 109

IV.1 LE SYSTEME ROMIE 110

IV.1.1 PRINCIPES DE ROMIE 110

IV.1.2 *MAPPING* D'ONTOLOGIES A BASE DE RESSOURCES (OU D'INSTANCES) 111

IV.1.3 ENRICHISSEMENT SEMANTIQUE D'ONTOLOGIE 114

IV.1.4 LE PROCESSUS DE *MAPPING* A BASE D'INSTANCES 115

IV.2 ROMIE APPLIQUE AU CONTEXTE EDUCATIF 122

IV.2.1 INTRODUCTION 122
IV.2.2 ENRICHISSEMENT SEMANTIQUE D'ONTOLOGIE 122

IV.3 ROMIE APPLIQUE AU CONTEXTE BIOMEDICAL 131

IV.3.1 LES RELATIONS ENTRE LES CONCEPTS D'ONTOLOGIE ET LES INSTANCES 133
IV.3.2 LES TYPES D'ANNOTATIONS DES INSTANCES BIOMEDICALES 134
IV.3.3 EXPLOITATION DES INSTANCES DANS LE PROCESSUS DE *MAPPING* 136

IV.4 CONCLUSION ET CONTRIBUTION DU SYSTEME ROMIE 146

CHAPITRE V : EXPERIMENTATION ET EVALUATION 148

V.1 PROTOTYPE 148

V.1.1 PLATE-FORME JADE 149
V.1.2 SERVEUR ONTOBROKER 149

V.2 EXPERIMENTATION ET EVALUATION DU SYSTEME 152

V.2.1 LES METRIQUES UTILISEES POUR L'EVALUATION 152
V.2.2 EVALUATION DU SYSTEME OMIE 153
V.2.3 EVALUATION DU SYSTEME ROMIE 158

V.3 DISCUSSION ET CONCLUSION 163

CONCLUSION ET PERSPECTIVES 167

BIBLIOGRAPHIE 173

Liste des figures

Figure 1: Les couches du standard du Web sémantique 28
Figure 2: Le schéma de RDF(S)....37
Figure 3. Exploration de l'ontologie Adult_Mouse_Anatomy par PROTEGE2000 41
Figure 4. L'exploration de l'ontologie MeSH par l'outil OboEdit 42
Figure 5. Le modèle SIMBAD et les caractéristiques des ressources éducatives 45
Figure 6. Liste des ontologies biomédicales sur le site OBO.....49
Figure 7. Approche intégrée....52
Figure 8. Approche fédérée....53
Figure 9. Approche unifiée 54
Figure 10. Classification des *matchers* 56
Figure 11. Composition séquentielle des *matchers*....58
Figure 12. Composition parallèle des *matchers*....59
Figure 13. Les fonctionnalités du système OMIE....74
Figure 14. L'architecture du système OMIE pour le *mapping* d'ontologies 75
Figure 15. Compositions des *matchers* basés sur la comparaison des chaînes de caractères.....84
Figure 16. Le processus interactif de validation 94
Figure 17. L'architecture multi-agents du système OMIE. 98
Figure 18. Automate d'états finis représentant le comportement de l'agent OA....100
Figure 19. Automate d'états finis représentant le comportement de l'agent MA....101
Figure 20. Automate d'états finis représentant le comportement de l'agent HGA 103
Figure 21. Automate d'états finis représentant le comportement de l'agent HFA....104
Figure 22. Relation entre les seuils et les *mappings* candidats....104
Figure 23. Automate d'états finis représentant le comportement de l'agent FVA....106
Figure 24. Diagramme UML de la communication des agents du système OMIE....107
Figure 25. Relations entre les concepts d'ontologie et les ressources....110
Figure 26. ROMIE comme extension d'OMIE....111
Figure 27. Un entrepôt d'instances (ou ressources) annoté par les termes des deux ontologies 112
Figure 28. Deux entrepôts d'instances (ressources) annoté par deux ontologies différentes....112
Figure 29. Propagation des relations sémantiques entre instances vers les concepts....114
Figure 30. Morphisme d'ontologie 116
Figure 31. Voisinage sémantique généré par une relation non symétrique....117
Figure 32. Voisinage sémantique généré par une relation symétrique....117
Figure 33. *Matcher* sémantique à base d'une relation non symétrique....118
Figure 34. Matcher sémantique à base d'une relation symétrique....119
Figure 35. Filtrage avec les relations croisées....121
Figure 36. Modèle de ressource SIMBAD....123
Figure 37. Exemples de relation sémantique inter-ressources. 129
Figure 38. Enrichissement sémantiques de l'ontologie dans le modèle Simbad....130
Figure 39. Liste des ontologies biomédicales sur le site OBO....132

Figure 40. Les types de relations exprimées dans un fichier d'annotation.....135
Figure 41. Cas n°1 - deux entrepôts d'instances disjoints annotés par deux ontologies 135
Figure 42. Cas n°2 - deux entrepôts d'instances en relation annotés par deux ontologies.....135
Figure 43. Cas n°3 - un entrepôt d'instances annoté par deux ontologies.....136
Figure 44. Deux entrepôts différents annotés par deus ontologies différentes.....137
Figure 45.. Deux entrepôts convergents annotés par deux ontologies différentes. 139
Figure 46. Un entrepôt annoté par deux ontologies différentes. 140
Figure 47. L'annotation des instances de FlyBase par plusieurs ontologies.....141
Figure 48. Le fichier d'annotation "Fly_pheno.rdf".....142
Figure 49. Liens sémantiques entre les instances de FlyBase.....143
Figure 50. Architecture logicielle 148
Figure 51. Présentation graphique des métriques d'évaluation 153
Figure 52. Edition d'ontologie par OMIE.....154
Figure 53. Interface de configuration du système OMIE.....155
Figure 54. Évaluation des méthodes de comparaison (matchers) utilisés par le système OMIE 157
Figure 55. Comparaison d'OMIE avec d'autres systèmes existants 158
Figure 56. Évaluation des *matchers* du système ROMIE.....162
Figure 57. Évaluation des filtres du système ROMIE.....162

Liste des tableaux et des algorithmes

Tableau 1. Les techniques de *matching* utilisées dans les systèmes existants.....65
Tableau 2. Classification des systèmes existants suivant le langage de l'ontologie et de *mapping*.....66
Tableau 3. Classification des approches de *mapping* suivant leurs implémentations et expérimentation.....67
Tableau 4. Les règles de matchers structuraux 85
Tableau 5. Les combinaisons possibles entre les métadonnées des ressources.....125
Tableau 6. Liste de toutes les combinaisons possibles.....126
Tableau 7. Les relations sémantiques entre deux ressources 127
Tableau 8. La liste des relations inter-concepts à partir des relations inter-instances 137
Tableau 9. Description de l'annotation du fichier fly-pheno.rdf 144

Algorithme 1. *matchers* structuraux: Top-Down et Bottom-up 86
Algorithme 2. *matcher* structurel indirect.....87
Algorithme 3. *matcher* sémantique Top-Down d'une relation non-symétrique 88
Algorithme 4. *matcher* sémantique d'une relation symétrique.....88
Algorithme 5. Le déroulement d'une requête d'utilisateur 93
Algorithme 6. Algorithme d'un *matcher* sémantique à base d'une relation non symétrique.....119
Algorithme 7. Algorithme du *matcher* sémantique à base d'une relation symétrique. 120
Algorithme 8. Algorithme d'un filtre à base d'une relation sémantique.....121
Algorithme 9. Exemple d'une règle d'inférence avec sa description utilisée par le *matcher* structurel 151

Introduction

Nous assistons ces dernières années à l'émergence de nouvelles applications ayant besoin de partager de l'information entre différents systèmes comme c'est le cas pour l'apprentissage enrichi par les technologies (Technology Enhanced Learning, TEL) et les applications biomédicales. L'enjeu est de développer des techniques facilitant l'interopérabilité sémantique entre ces systèmes d'informations, qui constituent généralement des sources de données autonomes et hétérogènes.

L'interopérabilité est une question importante, largement identifiée dans plusieurs domaines comme par exemple dans la communauté des systèmes d'information (SI). La dépendance et le partage d'information entre des organismes ont créé un besoin de coopération et de coordination qui facilite aussi bien l'échange et l'accès aux informations distantes qu'aux informations locales. La large adoption de l'internet (WWW : World Wide Web), pour accéder et distribuer l'information, engendre un besoin crucial de l'interopérabilité des systèmes.

Actuellement le Web contient plus de 4.2 milliards de pages, mais la grande majorité d'entre elles sont dans un format lisible et compréhensible uniquement pour l'homme. Par conséquent les machines ou plutôt les logiciels ne peuvent ni comprendre ni traiter cette information. De plus, une grande partie du Web reste jusqu'à présent inexploitée.

Pour pallier cette insuffisance du Web, les chercheurs ont créé la vision du Web sémantique [9], où les ontologies vont décrire la structure et la sémantique des données. L'idée est que les ontologies permettent à des utilisateurs d'organiser l'information en taxonomie des concepts, chacune avec leurs attributs, et décrivent des relations entre ces concepts. Quand des données sont présentées ou annotées par des ontologies, les logiciels peuvent mieux comprendre leurs sémantiques, ce qui facilite la localisation et l'intégration des données pour des objectifs divers.

Le terme d'*ontologie* dans la philosophie est *la science de ce qui est*, c'est-à-dire les natures et les structures d'objets, de propriétés, d'événements, de processus et de relations suivant le contexte palpable de la représentation. L'ontologie cherche une classification approfondie, dans le sens que tous les types d'entités sont inclus dans cette classification. Dans le domaine de l'informatique, une position plus pragmatique aux ontologies est adoptée, où l'ontologie est considérée comme *un accord sur une représentation de domaine*.

Du point de vue de la technologie, une ontologie est souvent considérée comme un système d'information, comme le reflète cette définition générale d'ontologie « une ontologie est une spécification ***formelle*** et ***explicite*** d'une ***conceptualisation partagée*** » [40], avec les significations suivantes :

- Le mot « ***Formelle*** » se rapporte au fait que l'ontologie devrait être compréhensible par une machine ;
- Le mot « ***Explicite*** » signifie que le type des concepts utilisés, et les contraintes sur leur utilisation sont explicitement définis ;
- Le mot « ***Conceptualisation*** » signifie qu'un modèle abstrait des phénomènes est identifié par des concepts appropriés à ces phénomènes ;
- Le mot « ***Partagée*** » reflète que l'ontologie devrait capturer la connaissance consensuelle admise par les communautés.

L'ontologie est un facteur clé qui facilite l'interopérabilité dans le web sémantique [1]. Les ontologies sont le noyau du Web sémantique parce qu'elles permettent aux applications de communiquer en utilisant des termes partagés. L'ontologie facilite donc la communication en fournissant des notions précises qui peuvent être employées pour composer et échanger des messages (questions, réponses, etc.).

Cependant, il n'existe pas d'ontologie universelle partagée, adoptée par tous les utilisateurs d'un domaine donné. Les problématiques et les tentatives d'amélioration de l'interopérabilité du système comptent donc sur la réconciliation des différentes ontologies utilisées dans un domaine par les différents systèmes. Cette réconciliation est souvent réalisée par l'intégration manuelle ou semi-automatisée des ontologies. Elle consiste à identifier les liens de correspondance entre les ontologies, on parle alors de ***mapping* d'ontologies**.

Problématique

Le web sémantique offre une vision uniforme et standardisée, mais il soulève également beaucoup de défis compliqués et difficiles. Il propose une normalisation des méthodes de description sémantique de

l'information sur le web, d'une part sur le formalisme uniforme XML, et d'autre part sur une organisation de la connaissance à l'aide des ontologies.

Dans cette perspective, il est nécessaire d'effectuer des tâches complexes telles que l'accès aux informations ou aux ressources distribuées et gérées par les entités distinctes et hétérogènes. Devant l'impossibilité de définir une centralisation mondiale des ontologies, les difficultés sont souvent liées au choix de la définition exacte des formalismes, ce qui soulève le problème de l'interopérabilité des applications.

Le problème principal de tous les travaux sur l'interopérabilité porte sur la comparaison et le m*apping* des différentes ontologies. Etant donnée la nature décentralisée du développement du Web, le nombre d'ontologies est très important. Nous pouvons trouver plusieurs ontologies qui décrivent soit des domaines semblables, mais avec l'utilisation de terminologies différentes, soit des domaines complémentaires. Pour intégrer les données des ontologies distinctes, nous devons connaître les correspondances sémantiques entre leurs éléments.

Dans ce cadre, les techniques de *mapping* doivent soulever d'autres défis comme : (i) la robustesse, car des erreurs mineures ne doivent pas avoir des conséquences importantes, et (ii) l'évolutivité, car elles doivent travailler en un temps raisonnable avec un nombre considérable de données distribuées présentes sur le Web et avec des ontologies qui peuvent contenir des centaines de concepts sémantiques, même lorsqu'elles concernent des domaines spécialisés.

Afin de réaliser l'intégration des ontologies, il est nécessaire d'impliquer aussi bien la syntaxe que la sémantique des ontologies. Pour résumer, le processus de *mapping* est un des éléments fondamentaux du processus d'intégration d'ontologies. Il permet d'analyser et de comparer des ontologies pour déterminer les correspondances entre leurs concepts et pour détecter des éventuels conflits. Le résultat du processus de *mapping* est un ensemble de liens de correspondances.

Les liens de correspondance peuvent être employés directement dans un composant de traduction, qui traduit les rapports qui sont formulés par les termes des différentes ontologies, ou encore indirectement par un processus de fusion qui peut employer ces liens pour détecter les points de fusion. Nous proposons dans cette thèse d'utiliser ces liens pour accéder à des ressources et les partager ; ces ressources étant annotées par les concepts qui sont liés par ces relations de correspondance.

Ainsi, l'interopérabilité entre applications dans les systèmes hétérogènes dépend principalement de la capacité d'identifier les *mappings* entre les ontologies qui les sous-tendent. Aujourd'hui, le *mapping* d'ontologies est encore - en grande partie - réalisé manuellement. Par conséquent, l'intégration sémantique et automatique (ou même semi-automatique) est désormais devenue une tâche capitale dans le déploiement d'une large variété d'applications de gestion de l'information.

Objectifs et identification des verrous scientifiques

Le but de notre travail est de proposer un système capable d'identifier les correspondances sémantiques entre les ontologies avec l'intention de soutenir l'interopérabilité des systèmes d'informations. Nous allons plus particulièrement nous intéresser au domaine biomédical et au domaine de l'apprentissage enrichi par les Technologies (en anglais Technologies Enhanced Learning, TEL) qui sont des domaines où ce besoin de partage est évident. L'objectif principal de cette thèse est décomposé en objectifs intermédiaires :

- **Objectif 1** : Résoudre le problème d'hétérogénéité du langage d'ontologie ;
- **Objectif 2 :** La couverture des ontologies est très rarement la même et n'est pas complète. D'où la nécessité d'offrir aux utilisateurs un système de *mapping* capable d'identifier les correspondances de ***tout*** ou ***partie*** des éléments de l'ontologie. En d'autres termes, être capable de faire de l'intégration totale ou partielle des ontologies ;
- **Objectif 3 :** Beaucoup de travaux portent sur l'appariement et un grand nombre de techniques ont vu le jour, mais ils sont concentrés sur le calcul de similarité entre les termes des concepts d'ontologies ou sur la façon de combiner ces techniques. En effet, les algorithmes existants parviennent dans le meilleur des cas à proposer 70% de *mappings* corrects et d'identifier 80% des correspondances existantes. De plus, ces valeurs changent selon la structure et la richesse sémantiques des ontologies à faire correspondre. Il y a donc encore un effort à faire pour avoir une fiabilité plus importante du *mapping* indépendamment de la conceptualisation des ontologies. Notre but est d'analyser et exploiter les ***ressources*** (ou les ***instances***) attachées à ces ontologies afin ***d'enrichir leur sémantique et améliorer les résultats du processus de mapping.***
- **Objectif 4** : Dans le cadre de l'automatisation du processus de *mapping,* nous souhaitons aider l'utilisateur en réduisant le nombre de faux résultats, en allant plus loin que le filtrage à base du seuil qui est la seule méthode adoptée par la plupart des travaux existants. Notre but

est d'exploiter tous les liens possibles (***liens hiérarchiques, sémantiques***) existants entre les concepts de chaque ontologie dans le ***processus de filtrage*** pour détecter certaines anomalies et contradictions parmi les résultats de *mapping* obtenus.

- **Objectif 5 :** La phase de validation et d'interaction avec l'utilisateur est une phase clé dans le processus de *mapping,* mais elle est très compliquée. Pour l'ensemble des systèmes existants, les interactions avec l'utilisateur portent directement sur la validation des liens de correspondance entre les concepts des ontologies, ce qui rend la phase de validation des *mappings* un processus lourd et moins sûr. Partant de ce constat, il faudrait trouver un autre moyen de valider les résultats, qui soit plus automatique.

Nous expliquons dans ce qui suit comment les objectifs ci-dessus ont été traités par notre étude.

Approche et innovation

Il n'y a aucune normalisation pour que les interprétations des informations d'un domaine lors de la création d'une ontologie soient les seules conceptualisations possibles dans le monde réel. Nous supposons que la richesse sémantique des concepts d'ontologie vient de l'interprétation des ressources et des informations annotées par ces concepts.

Par conséquent, nous proposons que le point de départ de la comparaison et la création des correspondances sémantiques entre des ontologies hétérogènes soit d'enrichir sémantiquement ces ontologies avant de lancer le processus de *mapping* à proprement parler. L'enrichissement sémantique facilite le *mapping* d'ontologies en rendant explicite la sémantique « cachée » des différents concepts d'ontologies. Le but fondamental est que, plus la sémantique est explicitement spécifiée, plus la comparaison des ontologies devient fiable et facile.

Les techniques d'enrichissement sémantique peuvent être basées sur différentes théories et utiliser des ensembles d'instances. Nous basons notre approche sur l'analyse de propagation, c.-à-d. la propagation des liens sémantiques existants entre les instances (ou ressources) et/ou générés à l'aide de l'analyse des contenus de ces ressources et de leurs caractéristiques, sur les concepts qui les annotent. L'idée sous-jacente est que le traitement de ces ressources (ou instances) employées dans un domaine donné confère aux concepts d'ontologies des significations partagées par d'autres domaines ou communautés, ce qui nous permet de réaliser l'interopérabilité entre ces domaines.

Dans bon nombre de cas, les ontologies existent sans aucune instance associée. Pour cette raison, nous proposons deux versions du système, le système de *mapping* d'ontologie en l'absence de ressources (appelé *OMIE*) et le système de *mapping* à base de ressources (appelé *ROMIE*).

Les principales contributions

Considérant les problématiques et les objectifs du *mapping* cités ci-dessus, l'approche proposée se compose d'une phase d'analyse descriptive, d'une phase de développement et d'une phase de test et d'évaluation. L'ensemble de ces phases incluent les étapes suivantes.

1. L'étude de l'existant dans le domaine du *mapping*. Cela nécessite une recherche sur les ***méthodes et les solutions existantes pour le mapping d'ontologie*** et une analyse du processus de *mapping*, ainsi que les propriétés caractérisant ce processus ;
2. L'étude ***des méthodes de*** calcul de similarités (***matching***) existantes applicables dans notre contexte pour résoudre ***les hétérogénéités terminologiques***. Ces méthodes sont soit basées sur des informations auxiliaires capables d'identifier les éventuels liens linguistiques entre deux termes, soit sur des fonctions mathématiques permettant le calcul de la distance syntaxique entre ces deux termes ;
3. Le développement de l'étape ***d'enrichissement sémantique, qui*** inclut des spécifications du composant (les instances ou les ressources) à employer pour l'enrichissement sémantique de l'ontologie et les différentes manières de les exploiter dans le processus de *mapping*.
4. Le développement ***des étapes de l'algorithme de mapping, qui*** inclut la description des méthodes de calcul de similarité basées sur : la comparaison terminologique, la structure des concepts d'ontologies et les liens spécifiques générés dans l'étape précédente (étape d'enrichissement) ;
5. L'étape d'application du ***prototype, qui*** inclut le développement et l'exécution d'un système de *mapping* basé sur l'algorithme de *mapping* de l'étape précédente et appliqué à deux domaines, à savoir ***les ontologies biomédicales et les ontologies éducatives*** ;
6. L'étape *d'évaluation, qui* inclut l'expérimentation et l'évaluation des différentes techniques de calcul de similarité employées par le système. En particulier***, l'évaluation de l'impact de l'enrichissement sémantique sur le processus de mapping***. L'étape d'évaluation contient aussi

une ***phase de comparaison des résultats obtenus par notre système avec ceux obtenus par d'autres systèmes existants*** dans l'état de l'art.

Compte tenu de ces étapes et des objectifs cités auparavant, les principales contributions de notre thèse peuvent être récapitulées dans les points suivants :

1. Problème d'hétérogénéité du langage d'ontologie : Nous pouvons pallier ce problème en traduisant toutes les ontologies dans un langage possédant l'expressivité de tous les autres (il s'agit dans notre cas de F-logic). Nous avons ainsi un moyen de réaliser un « pivot sémantique » favorisant l'interopérabilité;
2. Problème de la couverture des ontologies et de leur taille : Nous préconisons une mise en correspondance à la demande, c'est-à-dire uniquement lorsque l'utilisateur recherche une ressource annotée par un autre référentiel. Bien entendu, l'intégration totale, c'est-à-dire la mise en correspondance de TOUS les concepts est également possible ;
3. Problème de fiabilité des résultats de *mapping* indépendamment du contexte d'application. Nous exploitons les informations extraites des instances pour consolider les *mappings* calculés. C'est ce que nous appelons l'enrichissement sémantique de l'ontologie;
4. Problème de la précision des résultats : En plus de l'utilisation de filtres à base de seuils (comme c'est le cas dans la plupart des travaux existants) pour réduire le nombre de fausses correspondances, nous proposons d'utiliser des informations extraites des ontologies (relations hiérarchiques et/ou sémantiques) qui peuvent jouer un rôle important pour détecter les anomalies et les contradictions entre les *mappings* obtenus ;
5. Problème de l'automatisation du processus de *mapping* : La plupart des solutions de mapping existantes ne sont pas complètement automatisées. En effet, ces approches automatisent seulement une partie du processus. Nous proposons d'aller plus loin dans l'automatisation du *mapping* ;
6. Problème de réutilisation : La plupart des systèmes existants utilisent des outils spécifiques intégrés dans le système dans lequel l'intervention de l'utilisateur dans le processus de *mapping* est nécessaire. Nous proposons une solution indépendante du système et du domaine d'application de façon à favoriser sa réutilisation ;

7. Problème de l'interaction avec l'utilisateur : Il n'est pas facile de savoir dans les outils existants s'ils disposent d'un processus interactif et si oui comment cette interaction est réalisée. Nous considérons que l'interaction avec l'utilisateur est un point fondamental qu'il faut intégrer dans les outils de *mapping*. Dans notre approche, nous offrons à l'utilisateur la possibilité de donner son avis sur les propositions de *mapping* ;
8. Problème de la validation : Afin d'alléger le processus de validation fait à travers une interaction avec un expert sur tous les liens de correspondance trouvés par le système, nous proposons d'utiliser un processus de validation indirect. Puisque la découverte des correspondances se fait à la demande, suite à une requête de l'utilisateur, mesurer le degré de satisfaction de ce dernier revient à vérifier qu'il utilise bien la ressource renvoyée par la requête ;
9. Enfin, nous proposons un système basé sur une architecture multi-agent pour mettre en application toutes ces propositions et nous procédons à des tests de validation pour évaluer notre approche.

Structuration du document

De nos objectifs détaillés dans la section précédente, découle naturellement l'organisation de ce mémoire qui est structuré en cinq chapitres :

Dans le premier chapitre nous présentons le contexte scientifique de nos travaux en introduisant les concepts de base des ontologies et du web sémantique ; plusieurs langages et outils de représentation des ontologies sont présentés. Ce chapitre introduit ensuite les deux domaines d'application qui ont guidé ce travail, à savoir le domaine biomédical et le domaine de l'apprentissage enrichi par les technologies.

Le deuxième chapitre présente une étude structurée des techniques de mise en correspondance d'ontologies. Tout d'abord, nous commençons par la présentation des approches d'interopérabilité sémantique existantes, puis nous effectuons une classification des techniques d'appariement et des modes d'intégration des ontologies, avant de terminer par une synthèse et analyse des caractéristiques des systèmes existants pour le *mapping* d'ontologies.

L'étude de l'existant et le contexte de notre travail nous ont conduit à définir un objectif double : d'une part définir une nouvelle architecture de *mapping* pour pallier les manques et les limites des systèmes existants et d'autre part, exploiter les domaines d'application pour répondre à des besoins concrets.

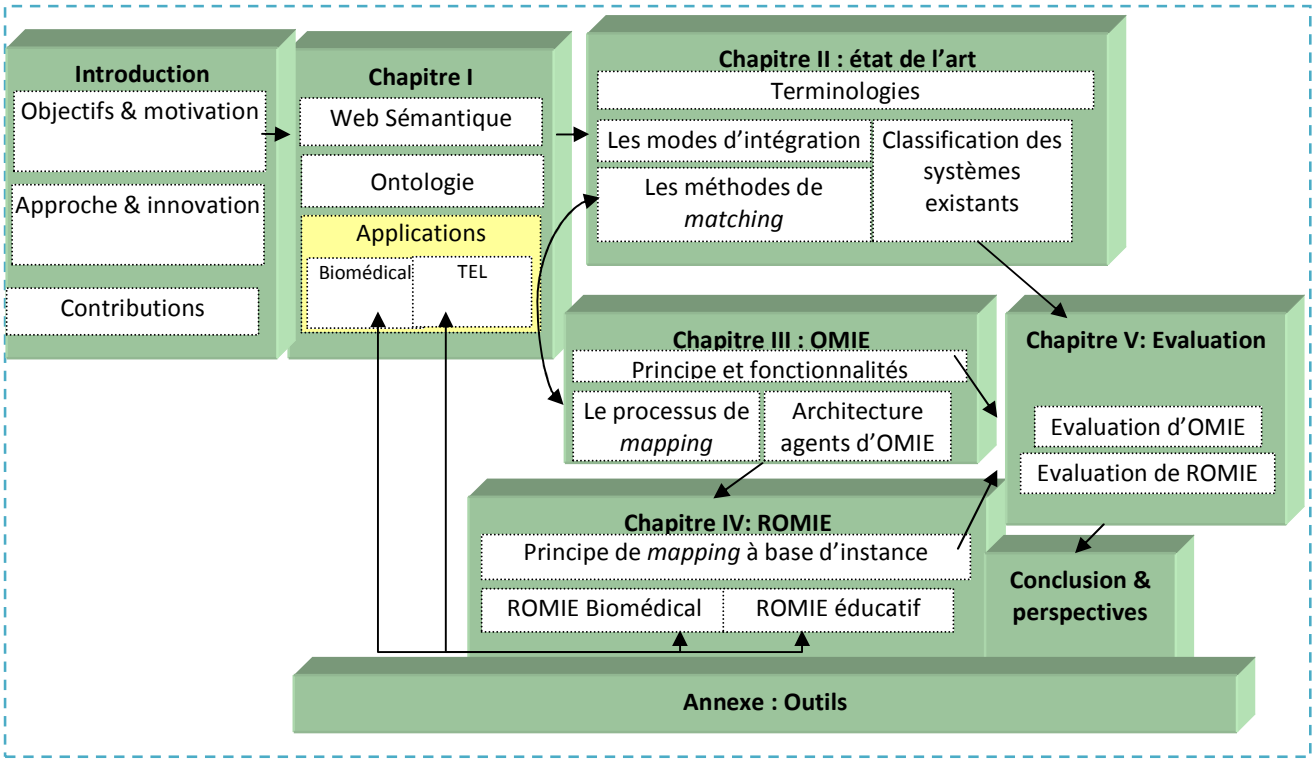
Le chapitre trois est consacré à la présentation de notre première contribution. Il s'agit du système OMIE qui propose une nouvelle architecture du système de *mapping* avec de nombreuses fonctionnalités. Il offre à l'utilisateur l'accès aux ressources (locales ou distantes) en utilisant les termes de son ontologie locale. Nous présentons également les mécanismes d'appariement (ou *matching*) ainsi que leurs combinaisons au sein d'OMIE. Pour plus de fiabilité des résultats de *mapping*, OMIE dispose de deux processus : (i) le processus de filtrage capable d'éliminer les réponses fausses d'une manière automatique et (ii) le processus de validation qui aide l'utilisateur à cibler le bon *mapping* et de profiter de son choix et sa satisfaction pour améliorer le processus de *mapping*. L'aspect distribué et coopératif de notre proposition nous a conduit à adapter la technologie des systèmes multi-agents pour la réalisation d'OMIE.

Nous présentons dans le chapitre quatre une autre contribution de notre thèse. Il s'agit d'exploiter les instances, les ressources et les données annotées par les ontologies à faire correspondre pour avoir plus de sémantique et par conséquent plus de fiabilité au niveau du processus de *mapping*. Cette extension du système OMIE par l'introduction des ressources pour le *mapping* est appelée ROMIE. Nous avons appliqué ROMIE aux deux types d'applications cibles. Nous montrons comment ROMIE exploite les ressources dans les deux cas et comment cette exploitation améliore le processus de *mapping*.

Finalement, Les expérimentations et les évaluations des deux systèmes OMIE et ROMIE sont présentées dans le chapitre cinq.

La conclusion de ce mémoire synthétise les principales contributions de cette thèse et donne quelques perspectives à ce travail. Les différents outils utilisés dans nos systèmes seront présentés dans l'annexe.

Le graphe ci-dessous illustre les différents chapitres de ce rapport ainsi que les liens entre eux.


Introduction
Objectifs & motivation
Approche & innovation
Contributions
Chapitre I
Web Sémantique
Ontologie
Applications
Biomédical
TEL
Chapitre II : état de l'art
Terminologies
Les modes d'intégration
Classification des systèmes existants
Les méthodes de *matching*
Chapitre III : OMIE
Principe et fonctionnalités
Le processus de *mapping*
Architecture agents d'OMIE
Chapitre V: Evaluation
Evaluation d'OMIE
Evaluation de ROMIE
Chapitre IV: ROMIE
Principe de *mapping* à base d'instance
ROMIE Biomédical
ROMIE éducatif
Conclusion & perspectives
Annexe : Outils

Chapitre I : Ontologies et domaines d'application cibles

Ce chapitre présente les concepts de base des ontologies et du Web sémantique.

I.1 Le Web sémantique

"...The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in co-operation."

Tim Berners-Lee, James Hendler and Ora Lassila

D'après *Tim Berners-Lee, James Hendler et Ora Lassila*, « le Web sémantique est une prolongation du Web actuel avec une signification bien définie des données et des ressources. Il permet aux ordinateurs et aux utilisateurs de coopérer et d'échanger l'information facilement ».

Le Web nous permet aujourd'hui d'accéder à des documents et à des services par l'intermédiaire d'Internet. L'interface d'accès aux services est représentée par des pages Web écrites dans du langage naturel, qui doit être compris par l'être humain. Le Web sémantique est une prolongation du Web actuel dans lequel l'information permet de donner une signification bien définie, afin que les ordinateurs et les personnes puissent travailler et coopérer entre eux efficacement. La vision du Web sémantique a été présentée la première fois par Tim Berners-Lee [9]. L'utilité du Web sémantique peut être vue au travers des exemples suivants : supposons que vous cherchiez à comparer les prix des canapés qui se fabriquent dans votre région (c.-à-d. le même code postal que le votre), ou que vous cherchiez les catalogues en ligne des différents fabricants de pièces de rechange d'une voiture de marque Peugeot 406. Les réponses à ces questions peuvent être disponibles sur le Web, mais pas sous une forme exploitable par la machine. Vous avez toujours besoin d'une personne capable de discerner et de vous donner la signification de ces réponses et de leur importance par rapport à vos besoins.

Le Web sémantique aborde ce problème de deux manières. Il permet d'une part aux communautés d'exposer leurs données de sorte qu'un programme n'ait pas à examiner le format, les images et les annonces d'une page Web pour identifier l'information appropriée. Il permet d'autre part à des personnes d'écrire (de produire) les dossiers qui expliquent - à une machine - les rapports entre différents ensembles de données. Par exemple, on peut faire « un lien sémantique » entre la colonne « code postal » d'une base de données et une forme avec un champ « postal» ou encore « CP» puisqu'ils signifient réellement la même chose. Ceci permet à des machines de suivre des liens et de faciliter l'intégration des données de plusieurs sources d'informations.

Le schéma de la Figure 1 montre les couches du standard du Web sémantique. Elle contient les éléments suivants :

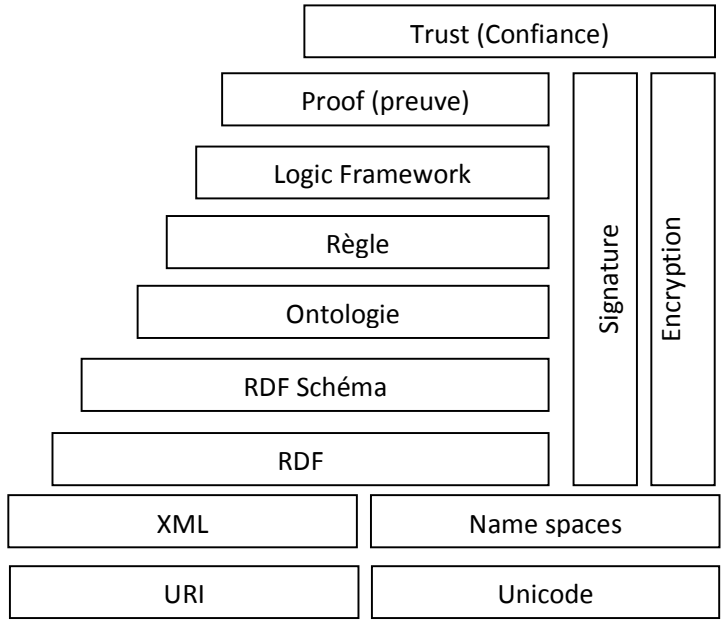


Figure 1: Les couches du standard du Web sémantique

- URI (*Uniform Resource Identifiers*) est un composant fondamental du Web actuel, qui fournit une identification unique des ressources ainsi que les relations entre ces ressources ;
- XML (*eXtensible Markup Language*) est un composant fondamental pour l'interopérabilité syntaxique ;
- RDF (Resource Description Framework) est un moyen d'encoder, d'échanger et de réutiliser des métadonnées structurées. C'est un idiome XML développé par le W3C et ayant fait l'objet d'une

Recommandation en 1999. RDF ne précise pas la sémantique des ressources décrites par les différentes communautés d'utilisateurs de métadonnées. À l'instar de XML, RDF est un langage extensible, un métalangage ; c'est un cadre « framework » de description des ressources applicable à n'importe quel domaine d'application ;

- RDFS est une prolongation de RDF. Un schéma RDF permet de décrire un vocabulaire et une sémantique des types de propriétés utilisées par une communauté d'utilisateurs ;
- La couche ontologie fournit plus de méta-information telles que la cardinalité des relations, leurs transitivité, etc. ;
- La couche logique permet l'écriture des règles ;
- La couche de la preuve exécute et évalue les règles utilisées ;
- La couche confiance fournit des mécanismes pour que les applications décident s'il faut faire confiance à la preuve indiquée ou pas ;
- Des signatures digitales sont employées pour détecter les éventuels changements de documents.

I.2 Ontologies

I.2.1 Rôle des ontologies

Nées des besoins de représentation des connaissances, les ontologies sont à l'heure actuelle au cœur des travaux menés dans le Web sémantique. Visant à établir des représentations à travers lesquelles les machines peuvent manipuler la sémantique des informations, la construction des ontologies demande à la fois une étude des connaissances humaines et la définition de langages de représentation, ainsi que la réalisation de systèmes pour les manipuler. Les ontologies participent donc pleinement aux dimensions scientifiques et techniques de l'Intelligence Artificielle (IA) : scientifiques comme étude des connaissances humaines et plus largement de l'esprit humain, ce qui rattache l'IA aux sciences humaines, et techniques comme création d'artefacts possédant certaines propriétés et capacités en vue d'un certain usage.

Au fur et à mesure des expérimentations, des méthodologies de construction d'ontologies et des outils de développement adéquats sont apparus. Emergeant des pratiques artisanales initiales, une véritable ingénierie se constitue autour des ontologies, ayant pour but leur construction mais plus largement leur gestion tout au

long d'un cycle de vie. Les ontologies apparaissent ainsi comme des composants logiciels s'insérant dans les systèmes d'information et leur apportant une dimension sémantique qui leur faisait défaut jusqu'ici.

Le champ d'application des ontologies ne cesse de s'élargir et couvre les systèmes conseillers (systèmes d'aide à la décision, systèmes d'enseignement assisté par ordinateur – *e-learning* –, etc.), les systèmes de résolution de problèmes et les systèmes de gestion de connaissances (par exemple dans le domaine du biomédical). Un des plus grands projets basé sur l'utilisation des ontologies consiste à ajouter au Web une véritable couche de connaissances permettant, dans un premier temps, la recherche d'information aussi bien au niveau syntaxique qu'au niveau sémantique.

L'enjeu de l'effort engagé est de rendre les machines suffisamment sophistiquées pour qu'elles puissent intégrer le sens des informations, qu'à l'heure actuelle, elles ne font que manipuler formellement. Mais en attendant que des ordinateurs « chargés » d'ontologies et de connaissances nous soulagent en partie du travail de plus en plus lourd de gestion des informations dont le flot a tendance à nous submerger, de nombreux problèmes théoriques et pratiques restent à résoudre.

I.2.2 La notion d'ontologie

« Ontologie » est un terme emprunté à la philosophie qui implique une branche de la philosophie qui traite la nature et l'organisation de la réalité. Dans le domaine de l'IA, de façon moins ambitieuse, on ne considère que *des* ontologies, relatives aux différents domaines de connaissances. En fait, plusieurs définitions d'ontologies sont données, mais celle qui caractérise l'essentiel d'une ontologie est fondée sur la définition relayée dans [41][40] :

« Une ontologie est une spécification **formelle** et **explicite** d'une c**onceptualisation partagée** » *Tom Gruber*, avec la signification des termes suivants :

- ***Formelle :*** réfère au fait qu'une ontologie doit être compréhensible par la machine, c'est-à-dire que cette dernière doit être capable d'interpréter la sémantique de l'information fournie ;
- ***Explicite :*** signifie que le type de concepts utilisés et les contraintes sur leur utilisation doivent être explicitement définis ;
- ***Conceptualisation :*** *se* réfère à un modèle abstrait de certains phénomènes dans le monde qui identifie les concepts appropriés de ce phénomène ;
- ***Partagée :*** indique que l'ontologie supporte la connaissance consensuelle, et elle n'est pas restreinte à certains individus mais est acceptée par un groupe.

Les taxonomies ou les thésaurus sont aussi des ontologies car elles se cantonnent à décrire des liens sémantiques du type "est-une-sorte de" et son inverse "est-représenté-par" ou, plus spécifiquement, "est-une-

sous-classe-de". Des ontologies plus complexes permettent la représentation de liens sémantiques plus spécifiques, par exemple, "est-localisé-dans". Mais surtout, les ontologies les plus abouties permettent également l'intégration de propriétés particulières, de règles d'utilisation et de contraintes.

I.2.3 L'origine des ontologies

L'Ingénierie des Connaissances (IC) est une branche de l'IA issue de l'étude des Systèmes Experts (SE). Si ces derniers n'avaient pour objet que la résolution automatique de problèmes, les Systèmes à Base de Connaissances (SBC), qui leur ont succédé sont censés permettre le stockage et la consultation de connaissances, le raisonnement automatique sur les connaissances stockées (sans préjugé sur le type de raisonnement à mener), la modification des connaissances stockées (ajout ou suppression de connaissances), et, avec le développement des réseaux, le partage de connaissances entre systèmes informatiques. De manière générale, il ne s'agit plus de faire manipuler en aveugle des connaissances à la machine, qui restitue à la fin la solution du problème, mais de permettre un dialogue, une coopération entre le système et l'utilisateur humain (systèmes d'aide à la décision, systèmes d'enseignement assisté par ordinateur, e-learning, recherche d'information sur le Web). Le système doit donc avoir accès non seulement aux termes utilisés par l'utilisateur, mais également à la sémantique que ce dernier associe aux différents termes, faute de quoi aucune communication efficace n'est possible. Plus précisément, les représentations symboliques utilisées dans les machines doivent avoir du sens aussi bien pour la machine que pour les utilisateurs, « avoir du sens » signifiant ici que l'on peut relier les informations représentées à d'autres informations.

Pour cela, la représentation des connaissances sous forme de règles logiques, utilisées dans les SE, ne suffit plus. Pour modéliser la richesse sémantique des connaissances, de nouveaux formalismes sont introduits, qui représentent les connaissances au niveau *conceptuel*, y compris la « structure cognitive » d'un domaine.

*« **Most KR formalisms differ from pure first-order logic in their structuring power, i.e. their ability to make evident the structure of a domain** »* [43].

Les langages à base de frames, les logiques de description et les graphes conceptuels sont des exemples de tels formalismes. Ces langages permettent de représenter les concepts sous-jacents à un domaine de connaissances, les relations qui les lient, et la sémantique de ces relations, indépendamment de l'usage que l'on souhaite faire de ces connaissances. Ainsi, une même base de connaissances peut être utilisée en consultation ou comme base de raisonnement.

Toutefois, il convient de souligner que la conceptualisation d'un domaine de connaissances ne peut se faire de manière non ambiguë que dans un contexte d'usage précis. Par exemple, un même terme peut désigner

deux concepts différents dans deux contextes d'usage différents [5]. On ne peut donc mener de façon totalement indépendante la représentation des connaissances d'un domaine et la modélisation des traitements que l'on souhaite leur appliquer.

En d'autres termes, modéliser des connaissances ne peut se faire que dans un domaine de connaissances donné, et pour un but donné, condition nécessaire à l'unicité de la sémantique associée aux termes du domaine.

Certains auteurs estiment cependant que les ontologies sont, par nature, destinées à être réutilisées [31], et s'attachent à construire des ontologies dont la sémantique est indépendante de tout objectif opérationnel. Au niveau ontologique, les primitives utilisées pour représenter les connaissances ne sont plus des mots du langage naturel, ou des primitives conceptuelles, et pas encore des prédicats logiques, mais des énoncés qui donnent le sens des connaissances, avec une interprétation contrainte.

Les ontologies sont donc des représentations de connaissances, contenant des termes et des énoncés qui spécifient la sémantique d'un domaine de connaissances donné dans un cadre opérationnel donné.

Le terme « ingénierie ontologique » a été proposé dans [66] pour désigner un nouveau champ de recherche ayant pour but la construction de systèmes informatiques tournés vers le contenu, et non plus vers les mécanismes de manipulation de l'information.

Avant de présenter les langages et les outils disponibles pour la manipulation d'ontologies, nous allons maintenant préciser quelles sont les primitives utilisées dans les ontologies.

I.2.4 Les constituants d'une ontologie

Les connaissances portent sur des objets auxquels on se réfère à travers des concepts. Un concept peut représenter un objet matériel, une notion, une idée [86]. Un concept peut être divisé en trois parties :

- un terme (ou plusieurs)
- une notion et
- un ensemble d'objets.

La notion, également appelée ***intension*** du concept, contient la sémantique du concept, exprimée en termes de propriétés et d'attributs, de règles et de contraintes.

L'ensemble d'objets, également appelé ***extension*** du concept, regroupe les objets manipulés à travers le concept ; ces objets sont appelés ***instances du concept***. Par exemple, le terme « table » renvoie à la fois à la notion de table comme objet de type « meuble » possédant un plateau et des pieds, et à l'ensemble des objets de ce type.

Il est à noter qu'un concept peut très bien avoir une extension vide. Il s'agit alors d'un concept générique, correspondant généralement à une notion abstraite (par exemple, la « vérité », prise dans le sens de

« ce qui est vrai » et non pas du « degré de vérité »). Deux concepts peuvent partager la même extension sans pour autant avoir la même intension. C'est le cas des concepts d'« étoile du matin » et d'« étoile du soir », qui désignent tous deux Vénus. De plus, des concepts partageant la même extension mais pas leur intension peuvent être désignés par le même terme. Ceci correspond à des points de vue différents sur un même objet. Par exemple, les chiens peuvent être considérés comme des animaux de compagnie, ou comme des ressources culinaires dans certaines cultures.

Le langage naturel contienne de nombreux termes désignant plusieurs concepts sémantiques différents (par exemple « table » pour un meuble et « table » pour un tableau de valeurs numériques), et de telles ambiguïtés ne sont pas gérables en machine, où on identifie généralement un concept à l'aide de ses termes. Néanmoins, la restriction à un domaine de connaissances permet généralement d'éviter les homonymies de concepts. Il apparaît par contre souhaitable de gérer les synonymies et de permettre la désignation d'un concept par plusieurs termes, pour assurer une plus grande souplesse d'utilisation de l'ontologie [39].

Un autre débat non tranché porte sur les propriétés contenues dans les intensions des concepts. En effet, certaines propriétés sont essentielles à la caractérisation d'un concept, dans le sens où la suppression de cette propriété entraîne la disparition du concept en tant que tel [12]. D'autres propriétés peuvent cependant être considérées pour caractériser le concept dans un contexte donné. Ces propriétés ne sont vraies que dans ce cadre et leur disparition ne modifie pas le concept. N. Guarino ne considère comme ontologique que les propriétés nécessaires [42]. G. Kassel admet dans les ontologies des propriétés incidentes, c'est-à-dire vraies seulement dans le cadre applicatif [47].

L'exemple du concept de « table » montre que cette notion ne peut se définir qu'en utilisant d'autres concepts comme « meuble », « plateau » et « pied ». De fait, les concepts manipulés dans un domaine de connaissances sont organisés au sein d'un réseau de concepts. L'ensemble des concepts y est *structuré hiérarchiquement* et les concepts sont liés par des propriétés conceptuelles. La propriété utilisée pour structurer la hiérarchie des concepts est la **« *subsomption (super-concept)»***, qui lie deux concepts : un concept C_1 subsume un concept C_2 si toute propriété sémantique de C_1 est aussi une propriété sémantique de C_2, c'est-à-dire si C_2 est plus spécifique que C_1. L'extension d'un concept est forcément plus réduite que celle d'un concept qui le subsume. Son intension est par contre plus riche.

Une liste des principales propriétés pouvant être associées à un concept est donnée ci-dessous. Les propriétés portant sur les extensions de deux concepts sont utiles dans le cas où on ne définit un concept que

par son intension, mais où l'on veut préciser qu'aucune instance commune aux deux concepts ne doit être créée ultérieurement sans respecter certaines propriétés.

Les propriétés portant sur deux concepts sont :

- **L'équivalence :** deux concepts sont équivalents s'ils ont même extension. Par exemple : « étoile du matin » et « étoile du soir » ;
- **La disjonction :** (on parle aussi d'incompatibilité) deux concepts sont disjoints si leurs extensions sont disjointes. Par exemple : « homme » et « femme » ;
- **La dépendance :** un concept C_1 est dépendant d'un concept C_2 si pour toute instance de C_1 il existe une instance de C_2 qui ne soit ni partie ni constituant de l'instance de C_2. Par exemple : « parent » est un concept dépendant de « enfant » (et vice-versa).

Ces propriétés ont été formalisées, afin d'être traduites dans des langages d'ontologie (voir plus loin).

Il est également possible de classer les propriétés que l'on utilise à l'aide d'autres critères. C. Welty propose ainsi de distinguer les propriétés indispensables, qui ne sont liées qu'au concept lui-même, comme la généricité, et les propriétés extrinsèques, qui font intervenir d'autres concepts dans leur définition [88].

Des modalités peuvent également être introduites au niveau des propriétés qui ne sont pas de nature modale, comme l'identité. Un concept pourrait alors porter possiblement (*i.e.* dans certains mondes) ou nécessairement (*i.e.* dans tous les mondes possibles) un critère d'identité.

En plus des propriétés, l'intension d'un concept peut contenir des ***attributs***. Par exemple, un « chien » possède quatre attributs « pattes » et un attribut « queue », entre autres. Un attribut peut être une instance de concept. Par exemple, un « président de la République française » a toujours le Palais de l'Elysée comme « résidence officielle ». D'autre part, on peut avoir besoin d'affecter des attributs (concepts ou instances) à une instance d'un concept. Par exemple, une « Ferrari », instance d'« automobile », porte un attribut « couleur rouge » instance du concept « couleur ».

Si certains liens conceptuels existant entre les concepts peuvent s'exprimer à l'aide de propriétés portées par les concepts, d'autres doivent être représentés à l'aide de relations autonomes. Une relation permet de lier des instances de concepts, ou des concepts génériques. Elles sont caractérisées par un terme (voire plusieurs) et une signature qui précise le nombre d'instances de concepts que la relation lie, leurs types et l'ordre des concepts, c'est–à–dire la façon dont la relation doit être lue. Par exemple, la relation « écrit » lie une instance du concept « personne » et une instance du concept « texte », dans cet ordre.

Tout comme les concepts, les ***relations*** peuvent être spécifiées par des propriétés dont une liste, non exhaustive, est donnée ci-après. Les relations sont organisées de manière hiérarchisée à l'aide de la propriété de

subsomption décrite précédemment. Les remarques faites au sujet de l'organisation des propriétés s'appliquent également dans ce cas.

Les propriétés fondamentales d'une relation sont :

- **Les propriétés algébriques :** symétrie, réflexivité, transitivité ;
- **La cardinalité :** nombre possible de relations de même type entre les mêmes concepts (ou instances de concept). Les relations portant une cardinalité représentent souvent des attributs. Par exemple, une « pièce » a au moins une « porte », un « humain » a entre zéro et deux « jambes ».

Les propriétés liant deux relations sont :

- **L'incompatibilité :** deux relations sont incompatibles si elles ne peuvent lier les mêmes instances de concepts. Par exemple, les relations « être rouge » et « être vert » sont incompatibles ;
- **L'inverse :** deux relations binaires sont inverses l'une de l'autre si, quand l'une lie deux instances I_1 et I_2, l'autre lie I_2 et I_1. Par exemple, les relations « a pour père » et « a pour enfant » sont inverses l'une de l'autre ;
- **L'exclusivité :** deux relations sont exclusives si, quand l'une lie des instances de concepts, l'autre ne lie pas ces instances, et vice-versa. L'exclusivité entraîne l'incompatibilité. Par exemple, l'appartenance et la non appartenance sont exclusives.

Les propriétés liant une relation et des concepts sont :

- **Le lien relationnel :** (propriété proposée par Kassel [47]). Il existe un lien relationnel entre une relation R et deux concepts C_1 et C_2 si, pour tout couple d'instances des concepts C_1 et C_2, il existe une relation de type R qui lie les deux instances de C_1 et C_2.
- **La restriction de relation :** (propriété proposée par Kassel [47]). Pour tout concept de type C_1, et toute relation de type R liant C_1, les autres concepts liés par la relation sont d'un type imposé. Par exemple, si la relation « mange » portant sur une « personne » et un « aliment » lie une instance de « végétarien », concept subsumé par « personne », l'instance de « aliment » est forcément instance de « végétaux ».

Ces propriétés associées aux ***concepts*** et ***relations*** complètent la sémantique différentielle de l'ontologie, au sens où elles contribuent à préciser les liens et différences entre les primitives cognitives du domaine de connaissances. Les aspects différentiels et référentiels sont cependant fortement imbriqués et la construction d'une ontologie est un processus complexe qui demande en particulier des choix de représentation délicats.

Nous présentons par la suite les langages et les outils de présentation et de création d'ontologies.

I.2.5 Les langages d'ontologies

Dans cette section, nous définissons quelques langages de représentation des ontologies les plus connus et les plus utilisés.

I.2.5.1 KIF

KIF [36] est un langage basé sur les prédicats du premier ordre avec des extensions pour représenter des définitions et des méta-connaissances, la logique du premier ordre étant un langage de bas niveau pour l'expression d'ontologies. Une extension du langage KIF, ONTOLINGUA, est utilisée dans le serveur d'édition d'ontologies, Ontolingua[1] du même nom.

I.2.5.2 RDF et RDF Schéma

Le W3C a adopté le langage RDF (Ressource Description Framework) comme un des formalismes standards de représentation de connaissances sur le Web[2]. Utilisant la syntaxe XML (Extended Markup Language[3]) qui constitue déjà un standard, le RDF permet de décrire des ressources Web en termes de ressources, propriétés et valeurs. Une ressource peut être une page Web (identifiée par son URI, United Ressource Identifier) ou une partie de page (identifiée par une balise). Les propriétés couvrent les notions d'attributs, relations ou aspects et servent à décrire une caractéristique d'une ressource en précisant sa valeur. Les valeurs peuvent être des ressources ou des littéraux. RDF dispose d'une sémantique formelle analogue à celle des graphes conceptuels, c'est-à-dire identique à celle d'un fragment de la logique du premier ordre [6].

Pour décrire n'importe quel type de connaissances à l'aide de ce formalisme, nous devons d'abord écrire en RDF le modèle sémantique à utiliser. Par exemple, pour décrire des connaissances en termes de concepts et de relations hiérarchisés, l'introduction des types « concepts » et « relations » et des propriétés de subsomption et d'instanciation est nécessaire. Un schéma de base incluant les primitives sémantiques généralement utilisées, a ainsi été ajouté au RDF et constitue ce qu'on appelle le RDF SCHEMA (RDF-S[4]). La Figure 2 montre les primitives

[1] Ontolingua, développé au Knowledge Systems Laboratory de l'université de Stanford. http://www.ksl.stanford.edu/software/ontolingua/

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

[3] http://www.w3.org/XML/

[4] http://www.w3.org/TR/rdf-schema/

de RDF(S). Les concepts et les relations sont déclarés dans un document RDF(S) comme instances de « Class » et de « Property ».

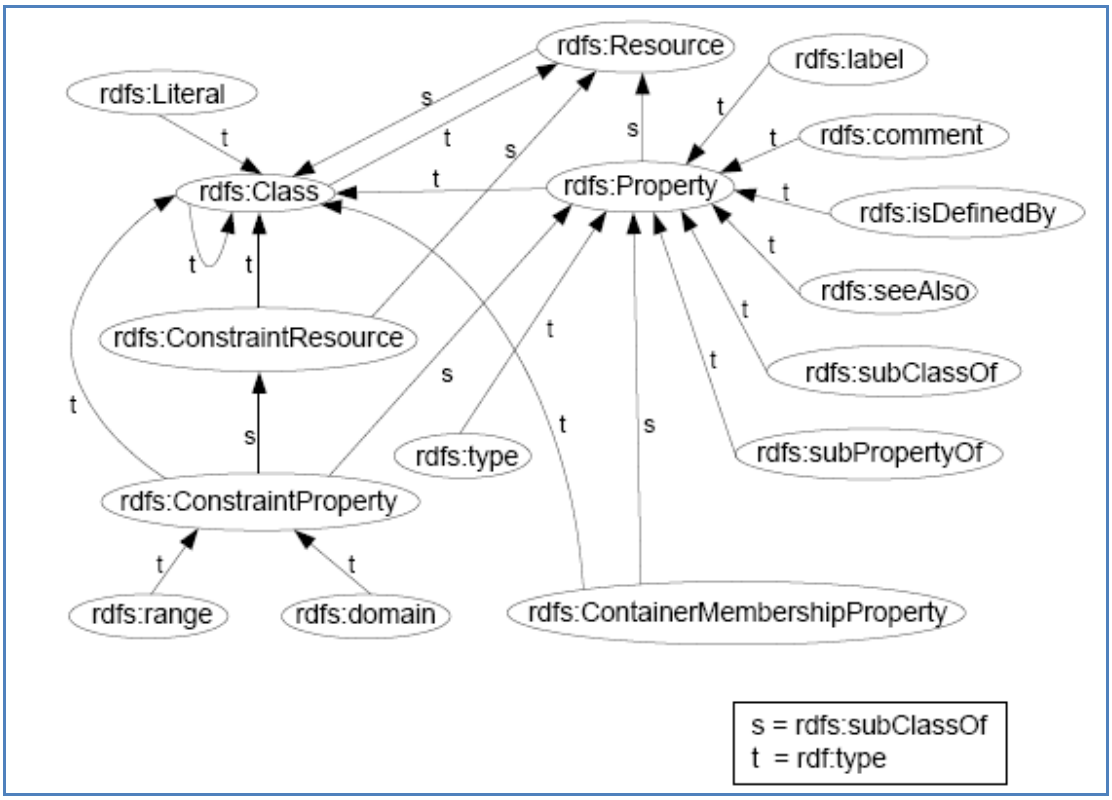


Figure 2: Le schéma de RDF(S)

Une fois ce schéma stocké sur le Web, les primitives qui y sont décrites peuvent être utilisées dans une page si on y inclut une référence à l'URI du schéma. Une application nécessitant l'accès à la sémantique de la page utilisera alors le schéma d'interprétation. Le RDF-S n'est cependant pas un langage opérationnel de représentation, au sens où il ne permet pas la représentation des axiomes et leur utilisation pour raisonner.

I.2.5.3 DAML + OIL

Dans l'optique d'une utilisation d'ontologies sur le Web, le langage RDF-S a été enrichi par l'apport du langage OIL (*Ontology Interchange Language*[1]) qui permet d'exprimer une sémantique à travers le modèle des frames tout en utilisant la syntaxe de RDF-S. OIL offre de nouvelles primitives permettant de définir des classes à l'aide de mécanismes ensemblistes issus des logiques de description (intersection de classes, union de classes,

[1] http://www.ontoknowledge.org/oil/

complémentaire d'une classe). Il permet également d'affiner les propriétés de RDF-S en contraignant la cardinalité ou en restreignant la portée [30].

Le langage OIL a été fusionné avec le langage DAML pour former le DAML+OIL. DAML (*Darpa Agent Markup Language*[1]) est conçu pour permettre l'expression d'ontologies dans une extension du langage RDF. Il offre les primitives usuelles d'une représentation à base de frames et utilise la syntaxe RDF [45]. L'intégration de OIL rend possible les inférences compatibles avec les logiques de description, essentiellement les calculs de liens de subsomption.

I.2.5.4 OWL

La combinaison de RDF/RDF-S et de DAML+OIL a permis l'émergence de OWL (*Web Ontology Language*), un langage standard de représentation de connaissances pour le Web,.

Développé par le groupe de travail sur le Web Sémantique du W3C, OWL peut être utilisé pour représenter explicitement les sens des termes des vocabulaires et les relations entre ces termes. OWL vise également à rendre les ressources sur le Web aisément accessibles aux processus automatisés [41], d'une part en les structurant d'une façon compréhensible et standardisée, et d'autre part en leur ajoutant des méta-informations. Pour cela, OWL a des moyens plus puissants pour exprimer la signification et la sémantique que XML, RDF, et RDF-S. De plus, OWL tient compte de l'aspect diffus des sources de connaissances et permet à l'information d'être recueillie à partir de sources distribuées, notamment en permettant la mise en relation des ontologies et l'importation des informations provenant explicitement d'autres ontologies.

OWL et les autres langages

XML [18] fournit une syntaxe pour des documents structurés, mais n'impose aucune contrainte sémantique à la signification des documents. RDF est un modèle de données pour représenter les objets et les relations entre eux, fournissant une sémantique simple pour ce modèle qui peut être représenté dans une syntaxe XML. RDF-Schéma est un langage de définition de vocabulaire pour la description de propriétés et de classes représentées par des ressources RDF. RDF-S permet de définir des graphes de triplets RDF, avec une sémantique de généralisation/hiérarchisation de ces propriétés et de ces classes.

OWL ajoute du vocabulaire pour la description des propriétés et des classes, des relations entre classes (par exemple *disjointness*), des cardinalités et des caractéristiques de propriétés (par exemple *symmetry*). OWL est développé comme une extension du vocabulaire de RDF et il est dérivé du langage d'ontologies DAML + OIL.

[1] http://www.daml.org/2001/03/daml+oil-index.htm

Sous langages de OWL

OWL a trois sous langages de plus en plus expressifs : OWL Lite, OWL DL, et OWL Full :

1. **OWL Lite :** Il supporte les utilisateurs ayant besoin principalement d'une hiérarchie de classification et des contraintes simples (un ensemble est limité à 0 ou 1 élément, par exemple). Il a une complexité formelle inférieure à celle d'OWL DL. OWL Lite supporte seulement un sous-ensemble de constructions du langage OWL.
2. **OWL DL :** D'après son nom, OWL DL utilise la logique de description DL [6]. Il a été défini pour les utilisateurs qui réclament une expressivité maximale tout en retenant la complétude informatique (toutes les conclusions sont garanties être calculables), et la possibilité de décision (les calculs finiront en un temps fini). Il inclut toutes les constructions du langage OWL, qui ne peuvent être utilisées que sous certaines restrictions.
3. **OWL Full :** Il a été défini pour les utilisateurs qui veulent une expressivité maximale et une liberté syntaxique de RDF mais sans les garanties informatiques. OWL Full permet à une ontologie d'augmenter la signification du vocabulaire prédéfini (RDF ou OWL). Il est peu probable que n'importe quel logiciel de raisonnement soit capable de supporter le raisonnement complet de chaque caractéristique d'OWL Full. Autrement dit, en utilisant OWL Full en comparaison avec OWL DL, le support de raisonnement est moins prévisible puisque l'implémentation complète d'OWL Full n'existe pas actuellement.

OWL Full et OWL DL maintiennent le même ensemble de constructions d'OWL. La différence se situe dans les restrictions sur l'utilisation de certaines de ses caractéristiques et sur l'utilisation des caractéristiques de RDF. OWL Lite permet le mélange libre d'OWL avec RDFS et, comme RDFS, n'impose pas une séparation stricte des classes, des propriétés, des individus, et des valeurs de données.

OWL Full peut être vu comme étant une extension de RDF, tandis que OWL Lite et OWL DL peuvent être vus comme des extensions d'une vue restreinte de RDF. Alors, les utilisateurs de RDF devraient se rendre compte qu'OWL Lite n'est pas simplement une extension de RDFS. OWL Lite met des contraintes sur l'utilisation du vocabulaire de RDF (par exemple, *disjointness* des classes, des propriétés, etc.). OWL Full est conçu pour la compatibilité maximale de RDF. Quant à opter pour OWL DL ou OWL Lite, il faut considérer si les avantages du OWL DL/Lite (par exemple, support de raisonnement) l'emportent par rapport aux restrictions de DL/Lite à l'utilisation des constructions de OWL et de RDF.

I.2.5.5 F-Logic

F-Logic [48] (La logique de Frame) est un langage de bases de données orienté objet qui combine l'expressivité des langages de bases de données déductives et la richesse de modélisation des modèles orientés objet.

Nous présentons ci-dessous quelques exemples de la création d'ontologie en F-Logic :

C_1 :: C_2 signifie que le concept C_1 est un sous-concept (direct ou indirect) de C_2 ;

I : C_1 signifie que I est une instance du concept C_2 ;

C_1 [P=>> C_2] signifie que les deux concepts C_1 et C_2e sont reliés par la propriété P ;

C[A->>V] signifie que le concept C dispose de l'attribut A ayant comme valeur V ;

I_1< :I_2 signifie que, l'instance I_1 fait partie de l'instance I_2 ;

etc.

I.2.6 Les outils de construction d'ontologies

De nombreux outils de construction d'ontologies utilisent des formalismes variés et offrent différentes fonctionnalités. Seuls les plus connus seront cités ici. Tous ces outils offrent des supports pour le processus de création d'ontologies, mais peu offrent une aide à la conceptualisation.

I.2.6.1 DOE

DOE (*Differential Ontologie Editor*) [84] [25] offre la possibilité de construire les hiérarchies de concepts et relations en utilisant les principes différentiels énoncés par B. Bachimont, puis en ajoutant les concepts référentiels. La sémantique des relations est ensuite précisée par des contraintes. Ce n'est qu'une fois l'ontologie ainsi structurée qu'elle est formalisée en utilisant la syntaxe XML.

I.2.6.2 PROTEGE 2000

Parmi les outils non liés à des formalismes de représentation, citons l'outil *PROTEGE2000* [72], [79]. PROTEGE2000 est une interface modulaire permettant l'édition, la visualisation, le contrôle (vérification des contraintes) d'ontologies, et la fusion semi-automatique d'ontologies à l'aide du plugin *Prompt* [70].

Le modèle de connaissances sous-jacent à PROTEGE-2000 est issu du modèle de frames et contient des classes (concepts), des slots (propriétés) et des facettes (valeurs des propriétés et contraintes), ainsi que des instances de classes et des propriétés. Il autorise la définition de méta-classes, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de bâtir une ontologie.

Il présente une interface graphique très agréable et simple à utiliser. Son utilisation dans le cadre de notre étude a été bénéfique pour pouvoir manipuler et/ou modifier les ontologies (de type rdf ou owl) ou encore pour créer des ontologies de test. La Figure 3 montre l'édition et les hiérarchies de classes de l'ontologie « adult_mouse_anatomy.owl ».

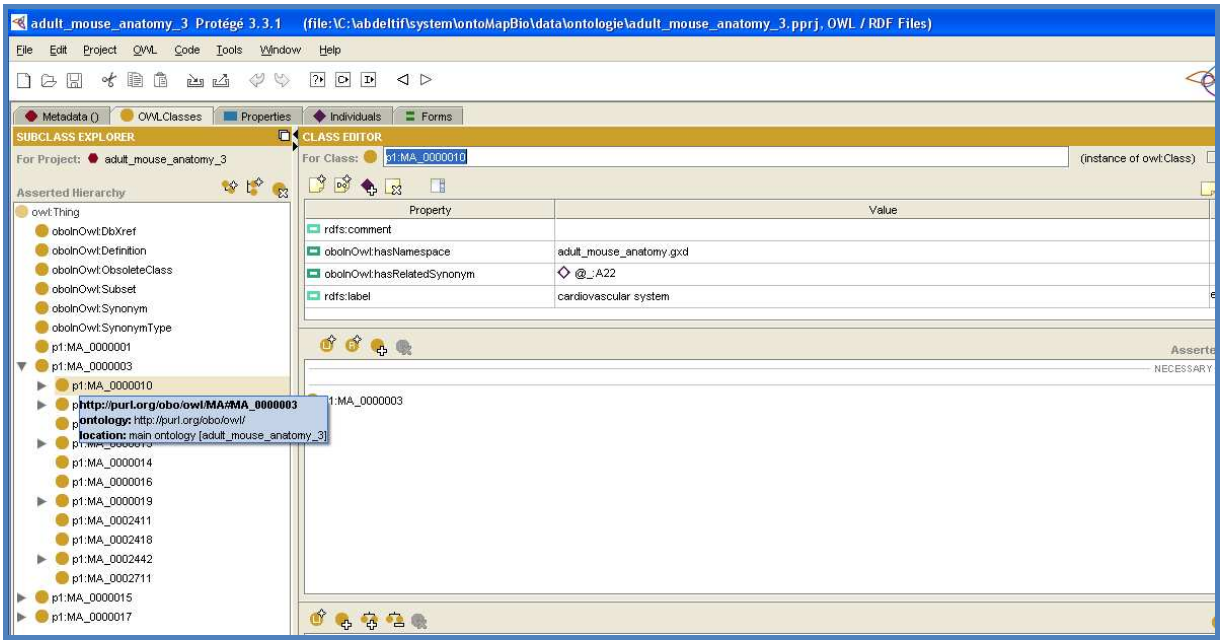


Figure 3. Exploration de l'ontologie Adult_Mouse_Anatomy par PROTEGE2000

I.2.6.3 OntoEdit

OntoEdit (*Ontology Editor)* [78] est également un environnement de construction d'ontologies indépendant de tout formalisme. Il permet l'édition des hiérarchies de concepts et de relations et l'expression d'axiomes algébriques portant sur les relations, et de propriétés telles que la généricité d'un concept. Des outils graphiques dédiés à la visualisation d'ontologies sont inclus dans l'environnement. OntoEdit permet d'éditer et de créer des ontologies de type 'obo'. Il est très utilisé dans le domaine biomédical.

OntoEdit intègre un serveur destiné à l'édition d'une ontologie par plusieurs utilisateurs. Un contrôle de la cohérence de l'ontologie est assuré à travers la gestion des ordres d'édition. Enfin, un plug-in nommé ONTOKICK offre la possibilité de générer les spécifications de l'ontologie par l'intermédiaire de questions de compétences. La Figure 4 illustre l'affichage et l'édition de l'ontologie obo « MeSH.obo » à l'aide d'OboEdit.

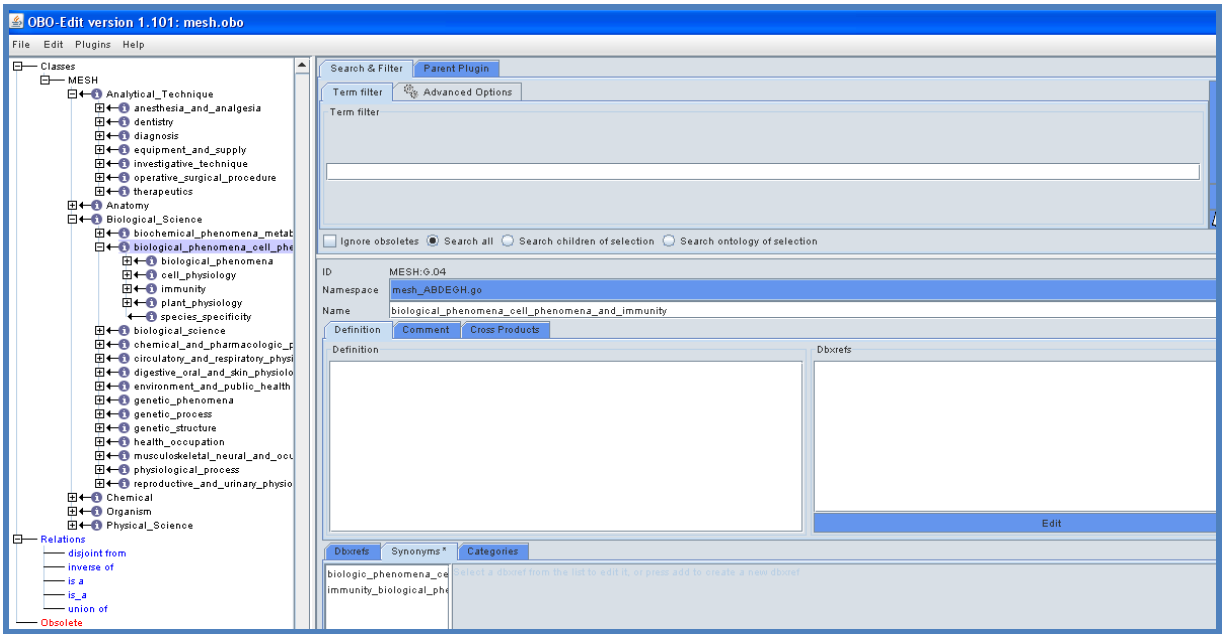


Figure 4. L'exploration de l'ontologie MeSH par l'outil OboEdit

I.3 Domaines d'application cibles

Nous avons développé et testé notre approche sur deux applications différentes : une dans le domaine de l'apprentissage enrichi par les technologies (TEL pour Technology Enhanced Learning ou plus simplement e-learning) avec le modèle SIMBAD et une autre dans le domaine biomédical dans le cadre du projet ANR SAPHIR et du projet POPS du pôle de compétitivité System@tic.

I.3.1 Le domaine de l'apprentissage enrichi par les technologies

I.3.1.1 Introduction

De nombreux systèmes d'apprentissage enrichis par les technologies existent, chacun d'entre eux étant fondé sur un domaine d'application particulier et possédant son propre vivier de ressources pédagogiques.

Afin de faciliter l'échange des ressources pédagogiques entre ces systèmes, il devient nécessaire de trouver des solutions permettant la coopération entre différents entrepôts de ressources. L'utilisation des ontologies de domaine est un moyen pour faciliter cet échange. En effet, un apprenant (ou un utilisateur) peut vouloir rechercher une ressource en dehors de son référentiel (les concepts du domaine qu'il a l'habitude d'utiliser et par conséquent qu'il maîtrise). Le problème est que l'intégration d'une nouvelle classification (une nouvelle ontologie) est coûteuse et demande un investissement très élevé. Il est ainsi nécessaire de proposer

des mécanismes qui permettent à l'apprenant d'accéder aux ressources des autres entrepôts d'une manière transparente et efficace.

I.3.1.2 Le modèle SIMBAD

SIMBAD est un TEL développé dans l'équipe et basé sur une ontologie de domaine. Afin de faciliter l'échange de ressources entre SIMBAD et tout autre TEL, il est nécessaire de trouver des solutions permettant la coopération entre ces différents entrepôts de ressources. Sachant qu'un utilisateur peut chercher des ressources en dehors de son ontologie de référence, le problème est que la compréhension d'une nouvelle ontologie est coûteuse et très compliquée. Il est ainsi nécessaire de proposer des mécanismes permettant à l'utilisateur d'accéder aux ressources des autres entrepôts d'une manière transparente, tout en utilisant les termes de son TEL préféré. Nous décrivons maintenant l'architecture du système SIMBAD. Une description plus détaillée de SIMBAD est donnée dans [17].

SIMBAD s'adresse à deux catégories d'utilisateurs : les auteurs de ressources et les apprenants. Les fonctions proposées dépendent du type d'utilisateur : (1) les auteurs ajoutent, recherchent et composent des ressources ; et (2) les apprenants suivent des formations. Ils identifient leurs besoins de formation en choisissant un cours précis. Par exemple, je veux suivre le cours de code BD21 dispensé dans mon université, en identifiant un ensemble de concepts à maîtriser. Par exemple, je recherche un cours me permettant de mieux maîtriser les bases de données relationnelles et les bases de données objet, ou encore en exprimant une requête plus libre. Par exemple, je cherche à consulter un des cours réalisés par Madame X.

Afin de réaliser ces fonctionnalités, SIMBAD s'appuie sur trois modèles : le modèle de domaine, le modèle de l'apprenant et le modèle de description de ressources.

Modèle de domaine : SIMBAD utilise une ontologie pour représenter l'ensemble des concepts du domaine de connaissances. Une ontologie permet d'organiser de manière hiérarchique les concepts (super-concept et sous-concept). Cette ontologie est enrichie par des relations rhétoriques (e.g., « est synonyme de », « enrichit »). Le but de ce modèle est de définir un référentiel commun, normalisé, partagé par tous les utilisateurs du système (apprenant, auteur, enseignant, administrateur). Ce modèle est relativement statique. Les ontologies permettent aussi de mettre en œuvre des mécanismes d'inférence que nous utilisons pour adapter les ressources pédagogiques au profil de l'apprenant.

Modèle de l'apprenant : il est composé de deux facettes. La première est dédiée aux préférences de l'utilisateur ; elle décrit des faits (nom, email, couleur préférée, langage, type de media préféré, type d'apprentissage). La seconde facette, appelée connaissances, décrit les concepts connus par l'apprenant, en

précisant le rôle maîtrisé par rapport à ce concept (par exemple, « introduction » à « SQL ») et le niveau de maîtrise (par exemple, « niveau élevé »). Il s'agit donc d'une vue (sous-ensemble) du modèle de domaine complétée d'évaluations (rôle et niveau de l'apprenant pour chaque concept). Le modèle de l'apprenant est dynamique : il s'accroît dynamiquement au fur et à mesure des acquisitions de l'apprenant.

Modèle de description de ressources : une ressource pédagogique est un composant qui doit être décrit (par un ensemble de métadonnées) lors de son ajout au système. Cette description permet de le retrouver, de le composer et de l'adapter. Un composant est associé à un ou plusieurs concepts du modèle de domaine. Dans notre modèle de description de ressources pédagogiques, nous distinguons deux types de métadonnées :

[1] les caractéristiques éducatives (auteur, langue, type de média) utilisant le standard LOM et

[2] la sémantique associée à la ressource. La description de cette sémantique est divisée en trois parties: les pré-requis, le contenu et la fonction d'acquisition (fonction permettant de faire évoluer le modèle de l'apprenant). Chacune de ces parties fait référence à des concepts du modèle de domaine. Une ressource « Interrogation-SQL », par exemple, a pour pré-requis le concept « Algèbre Relationnelle », pour contenu le concept « SQL » et pour fonction d'acquisition la modification du modèle de l'apprenant, en associant au concept « SQL » un niveau « élevé ». SIMBAD permet un accès à un entrepôt de ressources local.

Le pré-requis d'une ressource est décrit par le triplet (concept, rôle, niveau) où le concept est pris à partir du modèle de domaine, le rôle indique par quels aspects du concept cette ressource est concernée (par exemple : « introduction », « définition », « description », « application ») et le niveau précise le niveau de difficulté de la ressource (« low », « medium », « high »). En revanche, le contenu d'une ressource est décrit avec le couple (concept, rôle). La fonction d'acquisition indique quel triple (concept, rôle, niveau) sera ajouté au modèle de l'apprenant qui exprime l'état de satisfaction et de validation des connaissances de cet apprenant.

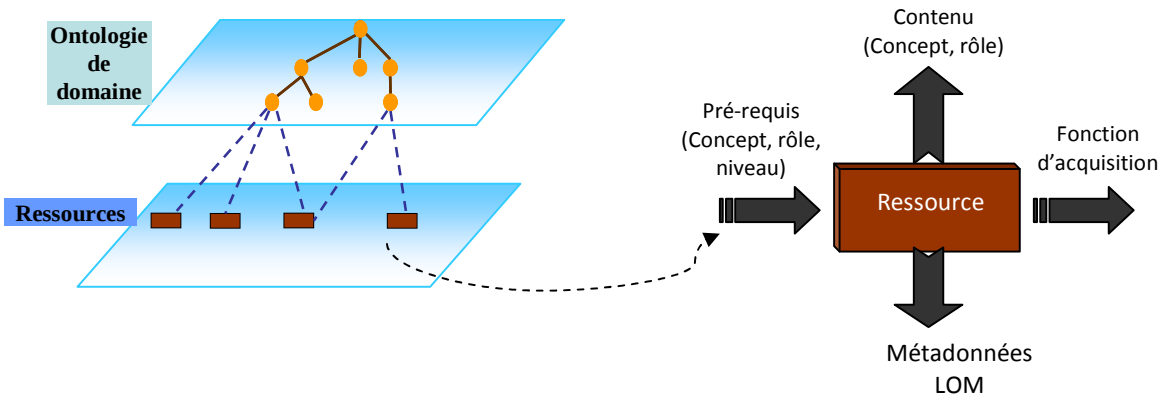


Figure 5. Le modèle SIMBAD et les caractéristiques des ressources éducatives

Une ressource pédagogique (ou éducative) peut être un ensemble : de pages Web, de fichier ou de programme (un simulateur par exemple). Nous supposons juste que c'est une unité accessible par l'intermédiaire d'un URI. Chaque concept de l'ontologie de domaine peut être lié à un ou plusieurs ressources avec un rôle spécifique.

Afin de permettre aux utilisateurs du système SIMBAD une ouverture vers d'autres entrepôts tels qu'ARIADNE [3] et EducaNext [26], nous proposons une approche de mapping d'ontologies permettant un accès facile à leurs ressources respectives.

I.3.2 Le domaine Biomédical

Les ontologies doivent fournir les concepts utilisés pour le marquage sémantique des données en vue du Web Sémantique. Le domaine biomédical dispose de standards terminologiques et thesaurus largement partagés par les communautés biomédicales, qui représentent un acquis important mais aussi une contrainte forte puisque qu'il n'est pas envisageable de les ignorer.

Dans notre étude, nous focalisons sur les ontologies dans la perspective du Web Sémantique. On met en lumière des besoins concrets du domaine biomédical -recherche, partage, et réutilisation de ressources - auxquels le Web Sémantique pourrait apporter des réponses. Cela permet de dégager certaines caractéristiques attendues et questions ouvertes sur les ontologies du Web sémantique biomédical. En particulier, la modularité doit permettre de combiner des ontologies spécifiques, ce qui nécessite d'avoir des solutions aux problèmes de correspondances des ontologies entre elles.

I.3.2.1 Le Web Sémantique biomédical

L'explosion du Web biomédical pose le problème de l'accès fiable aux informations pertinentes avec le minimum d'effort possible. Un moyen d'y parvenir est le marquage sémantique des ressources en utilisant des ontologies qui définissent les concepts et les relations avec une signification partagée et réutilisable pour différentes applications et différents usagers. Toutefois, la construction et la représentation des ontologies posent des problèmes difficiles.

Un groupe de travail du W3C a été chargé dans le cadre du Web Sémantique de concevoir un langage d'ontologies du Web « *Web Ontology Language* ». Le domaine biomédical est par ailleurs caractérisé par l'existence de nombreux standards terminologiques, thesaurus, et langages, partagés par les communautés biomédicales, qu'ils soient généralistes (par exemple, MeSH, UMLS) ou dédiés à un domaine de spécialité (par exemple, GeneOntology), ainsi que de riches banques de documents généralistes (par exemple, MEDLINE) ou plus spécifiques (par exemple, sur les maladies rares ORPHANET, RAREDISEASE).

Cette thèse vise à dégager certaines problématiques importantes qu'il faudra aborder pour l'interopérabilité du système biomédical.

I.3.2.2 Partage d'informations sur le Web sémantique biomédical

La recherche d'informations en médecine - par exemple - concerne de multiples utilisateurs du monde médical. Trouver rapidement sur le Web, avec le minimum de bruit possible, une information scientifique récente, a un intérêt non seulement pour le *chercheur* qui doit accéder à des bases hétérogènes et réitérer régulièrement les interrogations sur ces bases, pour les *patients* à la recherche d'informations, mais aussi dans la pratique médicale quotidienne où *médecins* et *industriels pharmaceutiques* sont amenés à rechercher de l'information. Une étude menée en 2000 aux Etats-Unis montrait que (1) plus d'utilisateurs utilisent le Web pour rechercher de l'information de santé que pour faire des achats, consulter des résultats sportifs ou des informations boursières, (2) les gens recherchent huit fois plus souvent de l'information sur une maladie que de l'information en nutrition ou fitness.

Les moteurs de recherche existants basés sur des mots-clés peuvent soit retourner des documents non pertinents pour cause d'homonymie ou d'un mauvais contexte par exemple, soit rater des documents à cause de l'utilisation de mots différents (synonymie), de mots plus spécifiques ou plus généraux (hypo-hyperonymie).

La solution classique des banques documentaires est d'indexer et rechercher les documents à l'aide de « descripteurs » d'un thesaurus plutôt que par mot-clé. C'est l'approche retenue pour l'indexation de la base

bibliographique MEDLINE® (*Medical Literature, Analysis, and Retrieval System Online*), basée sur le thesaurus MeSH[1].

Ce type de recherche et d'organisation a ses limites. En effet, certains concepts plus pointus peuvent être absents et la vérification de la cohérence est difficile. La définition d'ontologies avec les langages formels du Web sémantique devrait contribuer à une recherche plus « intelligente », à la classification et la vérification automatique de la cohérence des ontologies.

Plus particulièrement, le « partage d'informations » revêt dans le domaine biomédical de multiples aspects, notamment du fait de l'association étroite entre plusieurs activités différentes ou complémentaires. Le partage des données biomédicales, de plus en plus nombreuses et complexes compte tenu des progrès de la biologie et de l'imagerie, est donc indispensable pour assurer la délivrance et la continuité des soins. La richesse de ces informations, maintenant largement disponibles sous forme électronique, suppose des outils adaptés pour en assurer le partage au sein de différentes communautés d'utilisateurs. Les outils du Web sémantique, en facilitant le partage des référentiels sémantiques utilisés et l'interopérabilité des services proposés sur le Web, devraient donc contribuer de façon notable à un meilleur partage des données pour les applications biomédicales. Les ontologies pourront améliorer des applications existantes basées sur le Web et permettre de nouveaux usages du Web.

D'autre part, le Web sémantique pourra jouer un rôle déterminant dans la mise en place d'une *recherche biomédical* associant un plus grand nombre d'acteurs, en facilitant la constitution d'entrepôts de données ou d'entrepôts d'informations fédérés, articulés autour d'ontologies communes. L'enjeu est particulièrement important dans les domaines à caractère fortement pluridisciplinaire, comme par exemple en biologie. De tels domaines requièrent le partage à la fois de données et de connaissances, ce qui suppose « l'alignement » de plusieurs ontologies focalisées sur les liens sémantiques entre elles qui expriment les points de partage.

La construction d'ontologies pour un Web sémantique biomédical réactualise inévitablement un certain nombre de questions classiques mais récurrentes de représentation des connaissances en médecine : les « multiples points de vue ou multiples hiérarchies », la notion de contexte, la nature des liens is-a / kind-of, part-of, l'importance des relations rhétoriques, etc. Mais elle soulève aussi d'autres nouvelles questions plus

[1] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

spécifiques au Web sémantique, à savoir le *mapping* (ou la création des liens de correspondances) entre les ontologies.

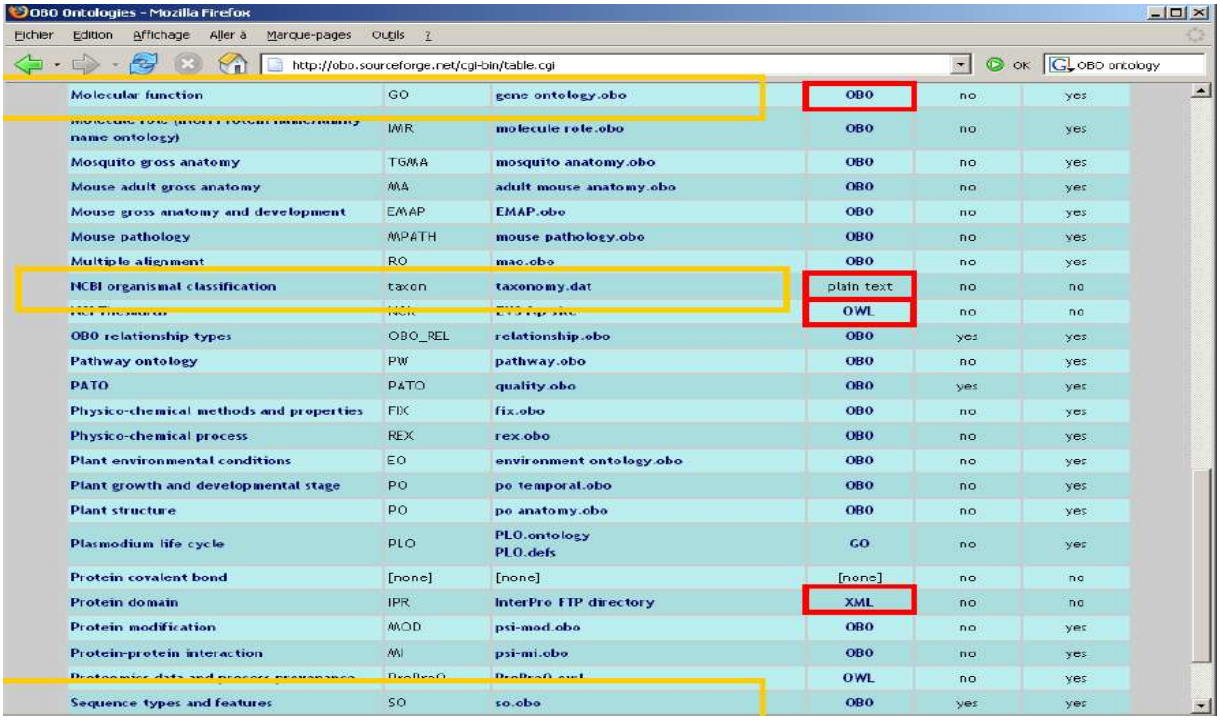
Le Web étant fondamentalement distribué et les connaissances biomédicales spécialisées, le Web sémantique biomédical contiendra plusieurs ontologies *modulaires*, produites indépendamment par divers utilisateurs, mais qui auront inévitablement besoin ***d'assembler, d'échanger et de partager*** les données présentées par ces multiples ontologies (par exemple des ontologies sur les gènes, les protéines, les organes, les maladies). Développées indépendamment et pour des usages différents, ces ontologies (i) utiliseront un vocabulaire et des langages de représentation différents et (ii) contiendront inévitablement des connaissances qui se recoupent. D'où l'utilité de réconcilier les ontologies biomédicales.

I.3.2.3 Les ontologies biomédicales

Un grand nombre d'ontologies biomédicales est accessible via le Web. Le site web OBO (*Open Biomedical Ontologies*) regroupe en un site unique les ontologies publiques et les vocabulaires contrôlés couvrant les différents domaines biomédicaux (Figure 6). La liste de ces ontologies disponibles ne cesse de s'accroître.

La plupart des ontologies disponibles sur le site OBO respectent les règles de syntaxe OBO [75]. Ce site contient aussi un moyen permettant la visualisation et l'édition de ces ontologies à l'aide de l'outil OBO-Edit [78]. Nous trouvons ainsi d'autres langages de représentation des ontologies comme : OWL, XML, GO, Fichier Plat, etc.

Les ontologies biomédicales sont également "orthogonales", c'est-à-dire qu'elles ne couvrent pas exactement le même sujet. Certaines sont spécifiques à un organisme particulier, d'autres sont génériques.



Molecular function	GO	gene ontology.obo	OBO	no	yes
name ontology)	MR	molecule role.obo	OBO	no	yes
Mosquito gross anatomy	TGMA	mosquito anatomy.obo	OBO	no	yes
Mouse adult gross anatomy	MA	adult mouse anatomy.obo	OBO	no	yes
Mouse gross anatomy and development	EMAP	EMAP.obo	OBO	no	yes
Mouse pathology	MPATH	mouse pathology.obo	OBO	no	yes
Multiple alignment	RO	mao.obo	OBO	no	yes
NCBI organismal classification	taxon	taxonomy.dat	plain text	no	no
			OWL	no	no
OBO relationship types	OBO_REL	relationship.obo	OBO	yes	yes
Pathway ontology	PW	pathway.obo	OBO	no	yes
PATO	PATO	quality.obo	OBO	yes	yes
Physico-chemical methods and properties	FIX	fix.obo	OBO	no	yes
Physico-chemical process	REX	rex.obo	OBO	no	yes
Plant environmental conditions	EO	environment ontology.obo	OBO	no	yes
Plant growth and developmental stage	PO	po temporal.obo	OBO	no	yes
Plant structure	PO	po anatomy.obo	OBO	no	yes
Plasmodium life cycle	PLO	PLO.ontology PLO.defs	GO	no	yes
Protein covalent bond	[none]	[none]	[none]	no	no
Protein domain	IPR	InterPro FTP directory	XML	no	no
Protein modification	MOD	psi-mod.obo	OBO	no	yes
Protein-protein interaction	MI	psi-mi.obo	OBO	no	yes
			OWL	no	yes
Sequence types and features	SO	so.obo	OBO	yes	yes

Figure 6. Liste des ontologies biomédicales sur le site OBO.

I.4 Conclusion

Nous avons décrit dans ce chapitre le cadre technique de notre travail. Nous avons présenté les concepts de base des ontologies et du web sémantique, qui sont la base de notre travail. Nous avons ensuite présenté les deux domaines d'application cibles sur lesquels nous avons illustré notre travail. Voici un résumé des principaux points que nous avons abordés :

- Le Web sémantique est établi sous forme de couches et l'ontologie est l'élément noyau du Web sémantique ;
- Une ontologie est une spécification explicite et formelle d'une conceptualisation partagée ;
- Des ontologies peuvent être classées selon leur niveau de formalisation ;
- La conception d'une ontologie est un processus itératif ;
- Il existe plusieurs langages de spécification d'ontologie avec des caractéristiques différentes. Parmi eux, DAML+OIL, RDF, OWL et F-Logic ;

- Pour permettre la création et la manipulation d'ontologies, une série d'outils sont nécessaires. Parmi eux, PROTEGE2000 et OboEdit sont sans doute les plus utilisés. Nous avons été amenés à les utiliser dans le cadre de nos deux domaines d'application.

Après ce panorama des technologies qui constituent le paysage technique de cette thèse, le chapitre suivant se focalise sur les travaux de recherche autour de l'alignement d'ontologies.

Chapitre II : Etat de l'art

Dans ce chapitre, nous examinons les outils, les systèmes et les travaux de recherche liés au processus de *mapping* d'ontologies. Nous commençons tout d'abord par la définition de quelques termes que nous employons tout au long de ce document puis nous définissons les différentes approches d'interopérabilité. Par la suite nous nous intéresserons à la problématique et à la classification des méthodes d'appariement ou *matching*. Une comparaison des outils et des systèmes existants qui traitent du *mapping* d'ontologies est présentée, basée sur les deux éléments considérés auparavant, c'est-à-dire les modes d'interopérabilité et les méthodes de *matching*.

II.1 Mise en correspondance d'ontologies

Dans cette section, nous présentons les différents modes d'intégration existants dans le domaine de l'alignement d'ontologies, mais tout d'abord, nous allons définir les terminologies les plus utilisées dans ce domaine et qui seront rencontrées tout au long de ce rapport.

II.1.1 Terminologie

Dans cette section, nous décrivons la signification des termes liés à la problématique du *mapping* d'ontologie à savoir : les *mappings*, le *matching*, le *matcher*, l'*alignement*, la *fusion* et l'*intégration* d'ontologies.

Correspondances ou Mappings : Les *mappings* sont des relations entre les éléments de deux représentations (ontologies, schémas de bases de données, etc.), indiquant une similarité relative selon une mesure donnée [50].

Appariement ou Matching : Le *matching* d'ontologies est le processus de définition d'un ensemble de fonctions permettant de spécifier des « *correspondances* » entre termes [90] [44].

Les méthodes de comparaison ou matchers : Un *matcher* est une fonction utilisée pour calculer la distance entre deux entités. Les *matchers* sont des fonctions qui peuvent être combinées dans le processus de *matching*.

Alignement d'ontologies: L'alignement d'ontologies est le processus d'établissement de liens de correspondances entre deux ontologies originales. Il est appliqué si les ontologies concernées deviennent homogènes entre elles et ceci tout en les gardant séparées (pas de fusion d'ontologies). Cette catégorie de *mapping* d'ontologies est faite habituellement quand les ontologies sources appartiennent à des domaines complémentaires. En revanche, certaines définitions comme [50] et [70] acceptent que les ontologies soient légèrement modifiées, pour que l'alignement aboutisse à un ensemble « pertinent et cohérent » (ce qui est rarement possible sans modification, pour deux ontologies développées indépendamment).

Fusion d'ontologies : La fusion d'ontologies est le processus de création d'une seule ontologie rassemblant les connaissances de deux ou plusieurs ontologies existantes et différentes qui décrivent le même sujet ou appartiennent au même domaine d'application. L'ontologie générée inclut les informations de toutes les ontologies sources, mais est plus ou moins inchangée.

Intégration d'ontologies : L'intégration d'ontologies est un processus de construction d'une nouvelle ontologie qui n'est pas forcément destinée à remplacer les autres (ces dernières peuvent continuer à être utilisées par ailleurs, à être mises à jour, à évoluer, etc.). Ces différentes ontologies peuvent être connexes.

II.1.2 Les approches de l'interopérabilité sémantique

Chaque entreprise peut développer sa (ses) propre(s) ontologie(s), correspondant aux concepts qui sous-tendent les données qu'elle manipule, soit en créant une nouvelle ontologie, soit en réutilisant des ontologies existantes. Pour parvenir à l'interopérabilité sémantique, il faut donc trouver un accord entre toutes ces ontologies ; on trouve dans la littérature trois types d'approches ([87], [70], [13]) : l'approche intégrée, l'approche fédérée et l'approche unifiée.

II.1.1.1 Approche Intégrée

Parfois nommée «médiation centralisée», «ontologie globale» ou «ontologie unique ou simple», elle consiste à s'accorder sur une seule ontologie (Figure 7). Cela implique un consensus sur le vocabulaire utilisé, sa sémantique, la granularité de l'ontologie, le point de vue, etc. Les sources

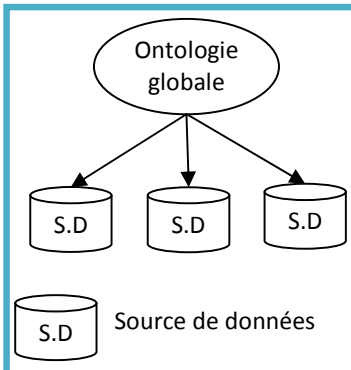


Figure 7. Approche intégrée

d'informations sont alors toutes reliées à cette ontologie globale. Cela revient souvent à fusionner les ontologies existantes en une seule.

Cette approche est naturelle lorsqu'on dispose d'une ontologie globale fournissant un vocabulaire partagé pour la spécification sémantique, ou lorsque plusieurs ontologies existent pour un même domaine, avec une granularité proche, et sont réalisées dans une même optique. Elle est à éviter quand l'une des ontologies qu'on veut intégrer (ou l'une des sources de données) contient des données hétérogènes et/ou évolue de manière indépendante : on doit alors s'attendre à devoir régulièrement modifier l'ontologie globale, et également les autres sources de données.

II.1.1.2 Approche Fédérée

Cette approche, également appelée «médiation décentralisée», «médiation distribuée» ou «ontologies multiples», consiste à considérer les ontologies comme étant des représentations approximatives du point de vue d'une communauté ou d'un individu [13], et à les faire correspondre l'une à l'autre (Figure 8). Donc chaque source d'information est d'écrite par sa propre ontologie et chaque ontologie est indépendante.

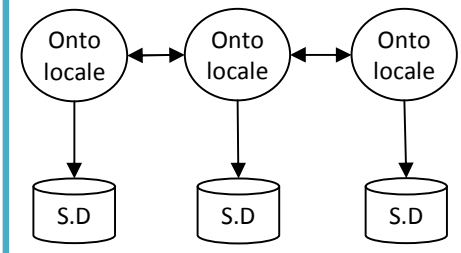


Figure 8. Approche fédérée

L'avantage de cette approche est que chaque ontologie source peut être définie sans prendre en considération les autres sources ou les autres ontologies. Elle permet une plus grande flexibilité et ceci en utilisant des ontologies évoluant de manière autonome et mises à jour fréquemment. Elle n'est pas bloquante si l'une d'elles vient à disparaître. On peut facilement supprimer une source (il suffit de supprimer les correspondances avec l'ontologie locale).

Mais le manque d'un vocabulaire commun conduit à une difficulté extrême pour comparer les différentes ontologies sources. Pour surmonter ce problème, un formalisme de représentation additionnel définissant le *mapping* inter-ontologies est fourni. Ce dernier identifie sémantiquement les termes correspondants des différentes ontologies sources. Ce *mapping* est difficile à définir à cause des nombreux problèmes d'hétérogénéité sémantique qui peuvent se produire. Parmi les principales difficultés, il y a les cas de synonymie et d'homonymie mais surtout l'ambiguïté due à un manque d'information [24]. Tous les systèmes de *mapping* existants jusqu'à présent dans la littérature nécessitent une grande intervention des experts du domaine. Nous nous intéressons dans le cadre de ce travail de thèse à cette problématique.

II.1.1.3 Approche Unifiée

Parfois nommée «hybride » ou «ontologie de plus haut niveau» («*upper level ontology*»), c'est un compromis entre les deux approches précédentes : elle consiste à établir des correspondances entre les ontologies locales (une ontologie par source), et à établir pour chacune d'elles des correspondances avec une seule ontologie de plus haut niveau (Figure 9).

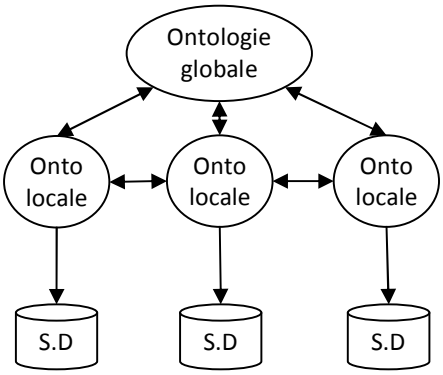


Figure 9. Approche unifiée

Comme pour l'approche précédente, sources et ontologies peuvent être développées de manière entièrement autonome, seules les correspondances sont à mettre à jour en cas d'évolution d'une ou de plusieurs ontologies.

Cette approche est surtout intéressante si les ontologies se conforment à un certain standard ; ainsi, les auteurs dans [87] proposent que toutes les ontologies locales soient décrites à l'aide d'un vocabulaire partagé (qui peut être une ontologie) comprenant les termes basiques du domaine : de nouveaux termes, plus complexes, pourront être crées à partir de la combinaison des premiers, à l'aide d'opérateurs spécifiques. Cette approche requiert cependant de commencer par créer un vocabulaire commun, ainsi que les règles de combinaison des termes. De plus, si les sources sont indépendantes, ce n'est pas le cas des ontologies, qui doivent utiliser le langage commun (il faut donc un consensus au préalable, ou construire soi-même les diverses ontologies correspondant aux différentes sources).

L'avantage de cette approche est que de nouvelles sources peuvent être ajoutées sans avoir besoin de modifier le vocabulaire partagé. Elle maintient encore l'acquisition et l'évolution des ontologies. De plus, même si les sources sont indépendantes, les ontologies doivent utiliser un langage commun (il faut donc un consensus au préalable, ou construire soi-même les diverses ontologies correspondantes aux différentes sources).

II.2 Les problématiques et les difficultés de l'appariement (*matching*)

Pour faire correspondre les différentes ontologies, deux étapes sont nécessaires : (1) s'abstraire de la différence entre les langages d'ontologies utilisés (par exemple en traduisant les ontologies dans un même formalisme de représentation), puis (2) chercher les concepts équivalents à apparier en tenant compte des différences de conceptualisation, de description de cette conceptualisation et de terminologie.

Nous passons assez vite la première étape par laquelle on trouvera plus de précision dans [50]. Dans nos applications, nous utilisons le serveur OntoBroker [77], qui est capable de traduire les différents formats d'ontologies (RDF, OWL) en format pivot « F-logic ».

II.2.1 Les techniques de l'appariement (matching)

Un grand nombre de *matchings* sont encore réalisés manuellement par des experts et cela ne devient plus possible tant le nombre et la taille des ontologies augmentent. L'utilisation d'un outil graphique semi-automatique (suggérant différentes relations de correspondance) permet de minimiser l'intervention humaine et d'accélérer le traitement.

Des travaux ont été menés sur ce front dans différents contextes : pour la traduction et l'intégration de schémas, dans le domaine de la représentation des connaissances, pour l'apprentissage automatique et la recherche d'information (« *Information retrieval* »), pour l'alignement et la fusion d'ontologies.

Parmi les premières recherches à ce sujet, [80] jugent que beaucoup de ces travaux ont été motivés par l'intégration de schémas de bases de données : l'appariement de schémas est la première étape vers l'intégration, permettant ainsi de récupérer les données dans la vue globale. Le *matching* entre schémas sert également pour traduire des données sources dans le format interne d'un entrepôt de données, pour traduire les messages lors des échanges du e-commerce, etc. C'est seulement depuis 1999-2001 que des travaux similaires commencent à émerger dans le domaine des ontologiess, notamment avec les publications [70] et [58].

La plupart des algorithmes ont fait l'objet d'une implantation spécifique et seuls Cupid [54], COMA [21] et Similarity Flooding [60], ont cherché à rendre leurs travaux davantage accessibles pour l'ensemble des domaines. On évoquera tout de même les apports d'autres propositions, car les techniques de *matching* dépendent beaucoup plus du type d'information qu'elles exploitent que du langage de représentation sur lequel elles se basent [80].

II.2.2 Classification des méthodes de *matching*

Les différentes méthodes de comparaison (*matchers*) utilisées dans le processus de *matching* sont organisées selon la classification ci-dessous (Figure 10) [28]. Les différents *matchers* sont classés suivant leur entrée, c'est-à-dire le type d'entités à comparer et la méthode utilisée pour calculer la similarité. Par exemple,

un des *matchers* syntaxiques compare deux entités (ou deux concepts) suivant l'égalité de leur noms (par exemple, les labels des concepts).

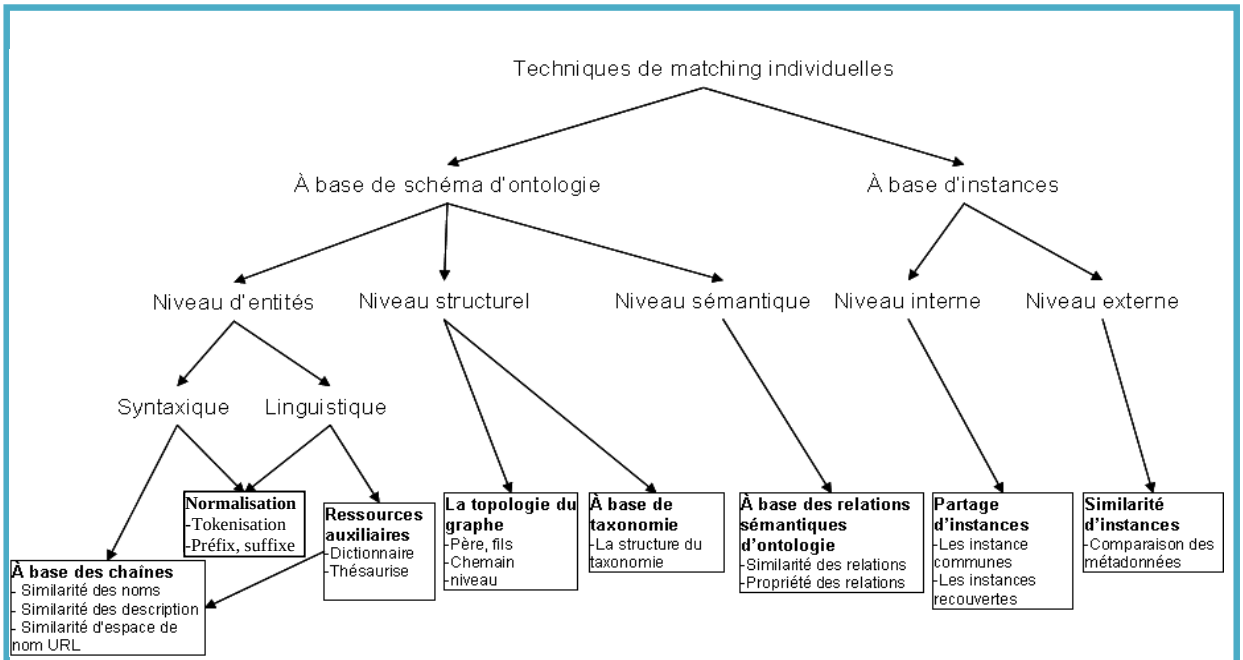


Figure 10. Classification des *matchers*

L'appariement entre deux entités ontologiques est soit basé sur le schéma d'ontologie ou basé sur les instances. Plusieurs niveaux, que nous explicitons ci-dessous, sont alors considérés :

II.2.2.1 *Matcher* au niveau entité

Le *matcher* au niveau entité compare les noms des entités en regardant le label ou l'identifiant d'un concept. Les noms peuvent être comparés, après une normalisation et parfois une concaténation avec le chemin depuis la racine. Les labels des concepts sont plus souvent renseignés que les autres types d'annotation, c'est donc une source d'information assez fiable. C'est souvent la première donnée observée lors du processus de *matching* ; sans cette première indication sur le concept représenté, aucun algorithme ne peut réaliser un *matching* convenable. Les descriptions et les commentaires peuvent servir également, après traitement par des algorithmes de Traitement Automatique des Langues (TAL).

Nous trouvons dans ce niveau deux approches de *matching* : l'approche syntaxique et l'approche lexicale, appelée aussi linguistique :

a. L'approche syntaxique effectue la correspondance à travers les mesures de dis-similarité des chaînes de caractères (par exemple, la distance de Hamming[1]) ;

b. L'approche lexicale ou linguistique effectue la correspondance à travers les relations lexicales (par exemple, synonymie, hyponymie, etc.). C'est généralement à partir des labels que l'on fait appel à des ressources auxiliaires tels que les dictionnaires de synonymes et d'hyponymes comme WordNet[2], mais aussi à des thésaurus et des ressources sémantiques ainsi qu'à des dictionnaires spécifiques aux domaines étudiés comme (UMLS[3]).

II.2.2.2 *Matcher* au niveau structurel et sémantique

Le *matcher* au niveau structurel et sémantique compare les structures internes des entités (par exemple, intervalle de valeur, cardinalité d'attributs, etc.). De nombreux travaux existants relatifs au problème de *matching* de schémas utilisent ce type de *matcher* pour lequel l'appariement entre les attributs de schémas se fait sur leur type de données.

Nous trouvons également dans ce niveau de *matching* un autre type d'appariement qui utilise la structure externe des entités, c'est-à-dire qu'il compare les entités au sein de leur topologie (par exemple, en comparant leurs pères, fils, etc.). Dans [54], les auteurs proposent une méthode qui consiste à parcourir l'ontologie des feuilles vers la racine (« bottom-up »). Cette méthode est plus lourde que l'approche inverse, mais peut permettre de comparer des schémas de structures très différentes et d'y trouver des candidats pour établir des correspondances.

Le *matcher* sémantique compare les interprétations (ou plus exactement les modèles) des entités. Il peut utiliser le voisinage ou les instances associées au concept pour définir son contexte et comprendre son interprétation.

[1] La distance de Hamming permet de calculer le nombre de positions dans lesquelles deux chaînes de caractères se différencient.

[2] WordNet est un dictionnaire en ligne de la langue anglaise, http://wordnet.princeton.edu/

[3] UMLS (Unified Medical Language System) est un metathesaurus medical. http://www.nlm.nih.gov/research/umls/

II.2.2.3 *Matcher* à base d'instances

Deux approches existent pour comparer les ontologies à partir des instances associées aux concepts d'ontologies :

- soit les deux ontologies à comparer référencent les mêmes instances et dans ce cas le *matcher* génère une similarité entre les concepts qui partagent les mêmes instances ;
- soit les deux ontologies à comparer ne référencent pas les mêmes instances et dans ce cas le *matcher* fait des recherches par mots-clés dans les instances (souvent des documents ou autres fichiers). La similarité est ensuite calculée entre les instances à l'aide de ces mots-clés. Les classes (concepts) liées à ces instances sont ensuite « appariées ».

II.2.3 Les stratégies de combinaison des *matchers*

Les différentes techniques citées auparavant peuvent ensuite être utilisées ensemble dans un algorithme « hybride » (deux ou plusieurs techniques dans un même algorithme) ou en un paramétrage d'algorithmes exécutés en parallèle (« composite »).

II.2.3.1 La combinaison séquentielle (hybride)

La méthode la plus simple pour combiner les *matchers* est l'utilisation séquentielle de ces *matchers* en choisissant un ordre d'exécution (Figure 11). Par exemple, nous choisissons de lancer le *matcher* à base de comparaison des labels avant de lancer le *matcher* structurel ou sémantique.

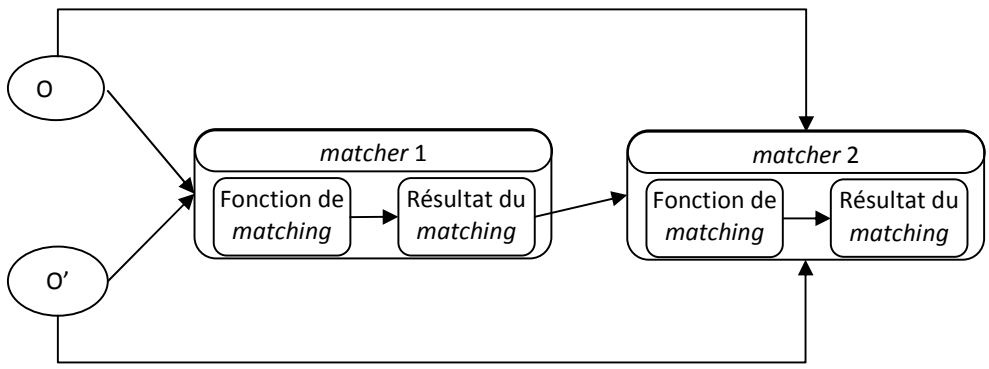


Figure 11. Composition séquentielle des *matchers*

II.2.3.2 La combinaison parallèle (composite)

Une autre manière de combiner les résultats de *matching* (c.-à-d. les valeurs de similarité) consiste tout d'abord à lancer parallèlement plusieurs *matchers,* puis par la suite à combiner leurs résultats de *matching* (Figure 12). Cette méthode est appelée « la composition parallèle » ou « *composite* des *matchers* ».

Il existe plusieurs mécanismes de combinaison des résultats (on parle aussi d'agrégation des résultats). Nous pouvons citer parmi eux : (i) une agrégation qui consiste à choisir un des résultats obtenus, par exemple, le fait de sélectionner la correspondance qui appartient à l'intersection des résultats de *matchings* (c.-à-d. qui est générée par tous les *matchers*) ; ou encore (ii) une agrégation qui sélectionne toutes les correspondances avec un degré de confiance très élevé.

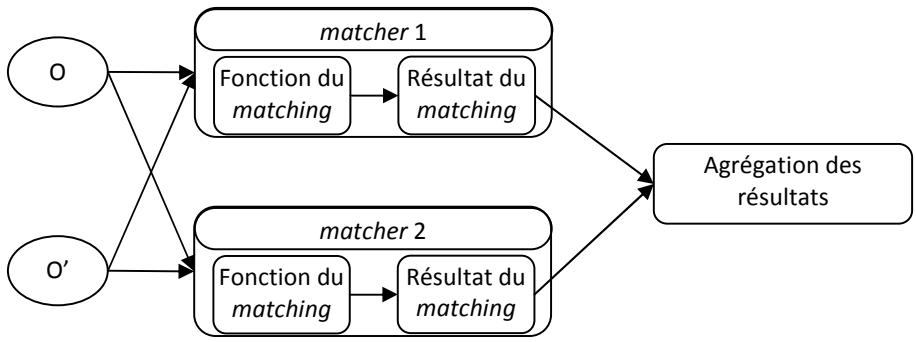


Figure 12. Composition parallèle des *matchers*

Remarque :

Notre approche combine les deux types de composition parallèle et séquentielle. En premier lieu, les *matchers* terminologiques sont lancés en parallèle puis les *matchers* structurels et sémantiques sont lancés également en parallèle.

II.3 Classification des systèmes de *mapping* suivant le mode d'intégration

Les systèmes hétérogènes distribués impliquent l'existence de plusieurs ontologies qui doivent être consultées et utilisées par ces différents systèmes. Ce caractère distribué du développement d'ontologies a mené à la création de différentes ontologies pour le même domaine d'application. Pour résoudre ces problèmes, il est nécessaire d'employer le *mapping* d'ontologies pour l'interopérabilité.

Nous classons les solutions proposées en trois grandes catégories :

1. le *mapping* entre une ontologie globale intégrée et des ontologies locales ;
2. le *mapping* entre des ontologies locales ;
3. le *mapping* pour la fusion et l'alignement d'ontologies.

La première catégorie de *mapping* d'ontologies supporte l'intégration des ontologies en décrivant les correspondances entre une ontologie globale intégrée et les ontologies locales. La deuxième catégorie permet l'interopérabilité pour les environnements fortement dynamiques et distribués comme les systèmes de médiation pour les données distribuées. Et enfin, la troisième catégorie de fusion ou d'alignement d'ontologies est employée dans le cadre du processus de réutilisation d'ontologies.

II.3.1 Le *mapping* entre une ontologie globale et des ontologies locales

Cette catégorie supporte les processus d'intégration d'ontologies. Ce genre de mapping indique comment les concepts dans l'ontologie globale et les ontologies locales sont mis en correspondance, comment ils peuvent être exprimés à l'aide d'une requête et comment ils sont typiquement modélisés en tant que vues ou requêtes.

II.3.1.1 Avantages et inconvénients

Les avantages de cette catégorie de *mapping* peuvent également être les inconvénients de la deuxième catégorie (*mapping* entre les ontologies locales) et inversement. En effet, il est plus facile de définir les correspondances dans ce cas de *mapping*, que dans les cas de *mapping* entre les ontologies locales, car une ontologie globale intégrée fournit un vocabulaire partagé et toutes les ontologies locales sont liées à cette ontologie globale. De plus, il peut être difficile de comparer les différentes ontologies locales car il n'existe pas de mappings directs entre les ontologies locales.

L'évolution et la mise à jour des ontologies sont très compliquées pour cette catégorie de *mapping*, parce que l'évolution des ontologies locales ainsi que l'ajout et/ou la suppression des ontologies pourraient facilement produire d'autres *mappings* vers l'ontologie globale. En définitive, il est pratiquement impossible d'utiliser ce type de *mapping* dans un environnement fortement dynamique.

II.3.1.2 Domaines d'application

Le *mapping* entre une ontologie globale et des ontologies locales est appliqué pour l'intégration d'ontologies dans des domaines tels que : le Web sémantique, la gestion des connaissances d'entreprise, et l'intégration des données ou d'informations.

Dans le Web sémantique, une ontologie globale intégrée extrait l'information à partir des ontologies locales et fournit une vue unifiée par laquelle les utilisateurs peuvent questionner les différentes ontologies locales.

En contrôlant des ontologies multiples pour la gestion de la connaissance d'entreprise, les différentes ontologies locales peuvent être combinées dans une ontologie globale intégrée.

Dans un système d'intégration de l'information, un schéma de médiation est construit pour répondre aux requêtes des utilisateurs. Dans ce cas, les *mappings* sont employés pour décrire des relations entre le schéma de médiation et les schémas locaux.

II.3.1.3 Systèmes et outils existants

Dans le domaine des bases de données, une ontologie globale intégrée ou un schéma de médiation sont créés sous forme de vues. Les mappings sont utilisés pour décrire les correspondances entre le schéma de médiation et les schémas locaux. Nous pouvons lister dans cette famille le système LSD [23] et le système de médiation MOMIS [8] et [7].

II.3.2 Le *mapping* d'ontologies locales

Cette catégorie facilite l'interopérabilité des environnements fortement dynamiques, ouverts et distribués, et peut être employée pour la médiation entre les données distribuées dans un tel environnement. Cette catégorie de *mapping* est plus adaptée à des applications du Web qu'à des applications de génération de *mappings* entre une ontologie globale intégrée et des ontologies locales.

II.3.2.1 Avantages et inconvénients

Ce *mapping* permet à des ontologies d'être mises dans un contexte permettant de garder le contenu local de chaque ontologie. Il peut fournir l'interopérabilité entre les ontologies locales quand celles-ci ne peuvent pas être intégrées ou fusionnées en raison de la contradiction mutuelle de leurs informations. Il est utile pour les environnements dynamiques, ouverts et distribués. Il évite également la complexité d'intégration des sources multiples.

Cette catégorie de *mapping* est plus simple à maintenir par rapport au mapping entre une ontologie intégrée et des ontologies locales, parce que les changements (l'ajout, la mise à jour, ou la suppression) des ontologies locales peuvent être faits localement sans tenir compte des autres *mappings*.

Finalement, la recherche des *mappings* entre ontologies locales est plus difficile que celle réalisée entre une ontologie intégrée et des ontologies locales en raison du manque de vocabulaire commun entre les différentes ontologies locales.

II.3.2.2 Les domaines d'application

Le *mapping* entre les ontologies locales est utilisé essentiellement dans le domaine du Web sémantique, en raison de sa nature décentralisée. Lorsqu'il n'y a aucune ontologie globale centrale, la coordination doit être faite en utilisant des ontologies, donc le *mapping* entre les ontologies locales devient nécessaire pour l'interopérabilité.

II.3.2.3 Les outils et les systèmes

A. **Le système GLUE** [24] **:**

Le système GLUE est un système semi-automatique de création de *mappings* d'ontologies utilisant des techniques d'apprentissage automatique. Il trouve le plus grand nombre de concepts semblables entre les deux ontologies et calcule la probabilité de la distribution commune des concepts en utilisant une approche multi-stratégies d'apprentissage pour la mesure de la similarité. GLUE comporte trois stratégies d'apprentissage, à savoir :

- *Content Learner* exploite les fréquences des mots dans le contenu des instances (concaténation des attributs d'une instance) et utilise le théorème de Naïve Bayes' ;
- *Name Learner* compare les noms des instances ;
- *Méta-Learner* combine les prévisions des deux stratégies précédentes et donne des poids à chaque stratégie suivant sa confiance d'être vrai ou faux.

B. **Le système MAFRA (Ontologie MAapping FRAmework)** [55] :

Pour les ontologies distribuées dans le Web sémantique, MAFRA fournit un processus de *mapping* distribué qui se compose de cinq modules horizontaux et de quatre modules verticaux.

Les cinq modules horizontaux sont les suivants :

1. *Ascenseur et normalisation* : Traite la langue et l'hétérogénéité lexicologique entre l'ontologie source et l'ontologie cible.
2. *Découverte de la similarité* : Découvre et établit des similarités entre les entités de l'ontologie source et les entités de l'ontologie cible.
3. *Pont sémantique* : Définit le *mapping* pour la transformation des instances de l'ontologie source à des instances de l'ontologie cible les plus semblables.

4. *Exécution* : Transforme des instances de l'ontologie source en des instances de l'ontologie cible selon les ponts sémantiques du troisième module.
5. *Post-processus* : Prend le résultat du module d'exécution et améliore la qualité des résultats de transformation.

Les quatre modules verticaux sont les suivants :

1. *Evolution : Maintient les ponts sémantiques en relation avec les changements au niveau des ontologies sources et cibles.*
2. *Bâtiment Coopératif De Consensus : Est responsable d'établir un accord sur les ponts sémantiques entre les deux ontologies dans le processus du mapping.*
3. *Domain Constraints and Background Knowledge (contraintes de domaine et connaissances de base) : Améliore la mesure de similarité et le pont sémantique en utilisant le dictionnaire WordNet ou des thesaurus spécifiques à un domaine donné.*
4. *Interface Graphique de l'utilisateur (GUI) : Intervention de l'être humain pour améliorer le mapping.*

II.3.3 *Mapping* d'ontologies pour la fusion et l'alignement

Cette catégorie de *mapping* permet de créer une ontologie globale à partir des ontologies locales à l'aide du processus de fusion d'ontologies (en anglais, ontology merging). Elle crée également des liens entre les ontologies locales pendant la procédure d'alignement d'ontologies. Les *mappings* n'existent pas entre l'ontologie fusionnée et les ontologies locales, mais existent entre les ontologies locales à fusionner et à aligner. Ces *mappings* permettent d'identifier les similarités et de résoudre les conflits entre les ontologies locales.

II.3.3.1 Avantages et inconvénients

Cette catégorie du *mapping* d'ontologies s'applique sur les ontologies qui couvrent le même domaine d'application. La recherche des *mappings* est une partie parmi d'autres, telles que la fusion ou l'alignement d'ontologies. Cette catégorie est intéressante pour des ontologies de taille importante.

II.3.3.2 Les domaines d'application

L'utilisation croissante des ontologies ainsi que le caractère distribué du développement des ontologies ont mené à un grand nombre d'ontologies qui ont le même domaine d'application. Ces différentes ontologies doivent être fusionnées ou alignées pour être réutilisées. Plusieurs applications tels que la recherche

d'informations, l'e-commerce, la médecine, etc., ont des ontologies à grande échelle qui devraient être utilisées dans le processus de fusion ou d'alignement d'ontologies.

II.3.3.3 Outils et systèmes

A. **Le système SMART** [71] :

SMART est un outil semi-automatique de fusion et d'alignement d'ontologies. Il créé une liste de similarités linguistiques initiale (synonyme, sous-chaîne partagée, suffixe commun et préfixe commun) basée sur la similarité de « classe-nom ». Ensuite il étudie les structures de la relation dans des concepts fusionnés, fait des suggestions aux utilisateurs pour identifier les conflits, et enfin fourni des solutions à ces conflits.

B. **PROMPT** [70]:

PROMPT est un outil semi-automatique de fusion et d'alignement d'ontologies. Il commence par les *matchers* linguistiques pour la comparaison initiale mais produit par la suite une liste de suggestions à l'utilisateur basée sur la connaissance linguistique et structurale, puis dirige l'utilisateur pour choisir les meilleurs *mapping*s. Prompt est un plugin de PROTEGE2000.

II.4 Classification des méthodes, outils et Framework existants

Parmi les approches et outils de *mapping* que nous avons étudiés, nous avons intégré les outils et les travaux utilisés pour le *matching* des schémas de bases de données. Certes, le problème n'est pas tout à fait identique, mais certains aspects des démarches proposées peuvent être réutilisés.

Nous comparons dans les prochaines sous-sections les méthodes de *mapping* d'ontologies et de *matching* de schémas selon trois critères :

1. Les techniques de *matching* supportées, afin d'étudier et de comparer l'impact des différentes méthodes de comparaison sur les résultats d'alignement ;
2. Les langages de représentation de l'ontologie et du *mapping*. L'étude de ce critère a pour objectif d'étudier d'une part, le problème d'hétérogénéité des langages de représentation d'ontologie et d'autre part, les langages de représentation des correspondances (*mapping*) ainsi que leurs réutilisations dans les processus d'intégration d'ontologies ;
3. L'implémentation et l'expérimentation. Ce critère est primordial pour évaluer la fiabilité des résultats de *mapping* obtenus.

II.4.1 Les techniques de *matching* supportées

Nous présentons et classons quelques approches de *mapping* existant dans la littérature dans le Tableau 1. Nous distinguons les deux critères de comparaison suivants :

1. Les techniques de *matching* employées par les outils de *mapping*, par exemple l'exploitation des termes, de la structure des ontologies ou encore les instances ;
2. le mode d'intégration, par exemple l'alignement, la fusion ou le *mapping*.

Tableau 1. Les techniques de *matching* utilisées dans les systèmes existants

Approche	Mode d'intégration	Technique de *matching*	Fonctionnement
Chimerae [59]	Fusion d'ontologies	Chimerae aide l'utilisateur à trouver le bon terme en lui proposant une liste de termes utilisés (et aide à résoudre les difficultés d'ordre terminologique)	Environnement basé sur le Web. Effectue la traduction au niveau langage depuis plusieurs formalismes. Chimerae utilise des heuristiques pour trouver les parties de l'ontologie à réorganiser.
OntoMorph [20]	Système de traduction et de transformation d'ontologies	OntoMorph utilise la plupart des techniques d'appariement disponibles.	OntoMorph emploie deux mécanismes de réécriture : syntaxique (pattern matching influencé par PLisp) et sémantique (basé sur PowerLoom, système de représentation de la connaissance permettant l'inférence). OntoMorph est très proche de Chimerae.
Prompt [70]	Fusion et alignement d'ontologies	Prompt commence par chercher les appariements possibles par similarité linguistique, mais se base surtout sur la structure et vérifie les actions de l'utilisateur pour détecter des éventuels conflits.	Proche de Chimerae et d'OntoMorph, Prompt est un module dans l'éditeur d'ontologies [Protégé]. Il permet des mises à jour automatiques, se rend compte de conflits et propose une aide pour les résoudre
FCA-Merge [82]	Fusion d'ontologies	Fait une comparaison basée sur les instances : FCA-Merge utilise des techniques linguistiques ; relie les termes clés des instances avec les concepts de l'ontologie (utilise un dictionnaire)	Il transforme les ontologies (parcours des feuilles vers la racine) en des contextes formels (techniques d'analyse de concepts formels) puis ajoute les termes extraits des documents (les instances).
Similarity Flooding [60]	*mapping* de sources génériques	Il est basé sur le calcul de la distance entre les termes et tient compte de la structure (l'appariement de deux concepts influence positivement leur voisinage). L'appariement est robuste aux cycles.	La source est convertie en un graphe. Pour chaque appariement réalisé, la similarité augmente pour les paires voisines. Une mesure juge de la qualité des appariements
Anchor-prompt [69]	*mapping* d'ontologies	Compare les labels, la typologie des attributs, la structure.	Nécessite l'introduction préalable d'un ensemble d'appariements pour différents concepts des deux ontologies.
COMA [21]	*matching* des schémas	Combine 13 algorithmes (implémentant les techniques pour l'appariement). Compare les labels, leurs similarités phonétiques, compare	L'utilisateur peut interagir pour sélectionner le mode de combinaison des *matchers*.

		la structure, la typologie des attributs, vérifie les synonymes. Utilise un historique des appariements effectués.	
GLUE [24]	*mapping* d'ontologies	Utilise les informations sur les instances (nom, taille, …) et sur la fréquence des mots contenus. Permet de prendre en compte le sens commun, les contraintes du domaine et la structure (prise en compte du voisinage).	Met en œuvre trois différentes stratégies d'apprentissage (adaptées au type d'information à acquérir) : une pour les noms, une pour les concepts et une autre pour combiner les deux approches (de manière probabiliste)
Cupid [54]	*matching* des schémas	Approche hybride ; compare les labels (thésaurus, etc.), la typologie des attributs, la structure (transforme le graphe en un arbre pour préserver le contexte défini par le chemin depuis la racine).	Parcourt les ontologies des feuilles vers la racine pour donner plus d'importance aux feuilles afin de pouvoir mettre en correspondance des schémas dont la structure intermédiaire varie mais peu.

II.4.2 Langages d'ontologies et de *mapping*

Le Tableau 2 énumère les langages de représentation d'ontologies et des *mappings* employés par les différentes approches étudiées. En analysant ce tableau nous pouvons constater qu'il y a trois types de représentation :

1. L'ontologie et le *mapping* sont représentés par le même langage. C'est le cas de MOMIS et OntoMap.
2. Le langage de *mapping* est différent du langage de l'ontologie. C'est le cas de MAFRA et RDFT, qui emploient tous les deux une méta-ontologie pour décrire les *mappings*.
3. Il n'y a pas un vrai langage de *mapping* : soit parce que l'objectif de l'outil est juste de découvrir des mesures de similarité entre les concepts des deux ontologies et non pas de réaliser un *mapping*, comme c'est le cas pour GLUE et S-Match ; soit parce qu'il s'agit d'un outil de fusion et par conséquent, nous avons une ontologie globale au lieu de liens de *mapping*, comme par exemple, PROMPT.

Tableau 2. Classification des systèmes existants suivant le langage de l'ontologie et de *mapping*.

Approche	**Langage d'ontologie**	**Langage du *mapping***	**Commentaires**
MAFRA [55]	RDFS	SBO : Semantic Bridge Ontology	SBO est une méta-ontologie. Il permet de présenter les *mappings* entre les concepts, les relations, et les attributs.
RDFT [76]	RDFS	RDFT	RDFT est une méta-ontologie qui décrit les types de *mapping*. Il permet seulement de présenter les *mappings* entre les concepts et entre les propriétés
Prompt [70]	Protégé-2000	N'existe pas : Prompt est un outil de fusion	Supporte le langage RDFS et OWL

GLUE [24]	Taxonomies	mesures de similarité	
S-Match [37]	DAGs	mesures de similarité	
OntoMap [49]	Similaire à OWL Lite	RDFS	Supporte le RDFS
InfoSleuth [35][68]	OKBC (Open Knowledge Base Connectivity)	N'existe pas	Il génère les *mappings* entre une ontologie et un schéma de donnée
OBSERVER [61]	Description logique	ERA: Extended Relational Algebra	Il génère les *mappings* entre une ontologie et un schéma de données
MOMIS [8]	$ODL_{/3}$	$ODL_{/3}$	Les bases de données relationnelles et semi-structurelles (e.g. XML) sont traduites par un adaptateur à l'ODL/3
ONION [64]	taxonomies	Règles	Les schémas des sources de données sont traduits en utilisant des adaptateurs (wrappers)

II.4.3 Implémentation et expérimentation

Le Tableau 3 énumère les types d'implémentations et les expériences qui ont été effectuées pour les différentes approches de *mapping* rapportées dans la littérature.

Comme nous pouvons constater sur ce tableau, la plupart des approches ont été seulement mises en application en tant que prototypes de recherche ou d'études conceptuelles. Pour la plupart de ces approches, nous nous sommes rendu compte qu'aucun développement ultérieur n'est prévu pour ces outils à l'exception du système PROMPT qui est toujours en développement (il existe à l'heure actuelle plusieurs versions) et qui a été adapté en tant que plug-in de Protégé2000.

La plupart des expériences citées dans la littérature sont basées sur des jeux de tests ; nous estimons que les vraies expériences avec des vraies ontologies à aligner ou à fusionner peuvent mieux valoriser et valider ces outils.

Tableau 3. Classification des approches de *mapping* suivant leurs implémentations et expérimentation

Approche	Implémentation	Expérimentation
MAFRA	Deux prototypes sont implémentés	Jeu de test
RDFT	Prototype	Application dans le projet GoldenBullet
PROMPT	Version 2.1.1 PROMPT	Évaluation en utilisant des ontologies de test dans le projet HPKB

GLUE/LSD/	Prototype	Jeu de test
S-Match	Etude conceptuelle	Jeu de test
OntoMap	Prototype	Jeu de test
RDFDiff	Prototype	Jeu de test
InfoSleuth	Prototype	Jeu de test
OBSERVER	Prototype	Prototype avec des ontologies bibliographiques réelles
MOMIS	Prototype	Jeu de test ARTEMIS (une partie de MOMIS) a été appliqué dans le domaine de l'administration publique italienne
ONION	Prototype	Jeu de test

Par ailleurs, il existe aujourd'hui de nombreuses méthodes automatiques permettant d'aligner des ontologies. Ces méthodes d'alignement sont basées sur des techniques très variées et obtiennent des performances très différentes en fonction des caractéristiques des ontologies à aligner. Dans ce contexte, il existe une campagne annuelle d'évaluation des outils d'alignement, appelée OAEI (*Ontology Alignement Evaluation Initiative*)[1], qui permet de comparer les résultats obtenus par les méthodes d'alignement participantes sur différents jeux d'ontologies. Cette campagne OAEI tente d'évaluer les algorithmes de mapping pour normaliser et améliorer le travail sur l'alignement d'ontologie. Parmi les principaux objectifs de cette initiative nous citons :

- L'évaluation des systèmes d'alignement et d'appariement ;
- La comparaison de la performance des techniques de mapping ;
- L'amélioration des techniques d'évaluation.

II.5 Analyse et synthèse

Il apparaît que certains problèmes évoqués plus haut n'ont pas trouvé encore de solutions fiables au problème de *mapping*. Partant des limitations de ces systèmes de *mapping* et en relation avec nos objectifs définis au début, nous proposons quelques éléments de réponse qui seront développés dans les prochaines sections.

1. Problème d'hétérogénéité du langage d'ontologie : Nous pouvons pallier ce problème en traduisant toutes les ontologies dans un langage possédant l'expressivité de tous les autres.

[1] http://oaei.ontologymatching.org

Nous avons alors trouvé un moyen de **réaliser un « pivot sémantique » favorisant l'interopérabilité.**

2. La couverture des ontologies est très rarement la même et n'est pas complète : **il ne faut donc pas chercher à mettre en correspondance TOUS les éléments.**

3. Beaucoup de travaux portent sur l'appariement, et un grand nombre de techniques ont vu le jour, mais le simple problème de l'équivalence concentre la majorité des efforts effectués : en effet, les algorithmes existants parviennent souvent à proposer 70% des relations de *mapping* correctes et d'identifier 80% parmi les correspondances existantes, et ces valeurs changent selon les ontologies à faire correspondre. **Il y a donc encore un effort à faire pour avoir une fiabilité plus importante du *mapping* indépendamment du contexte d'application, par exemple, l'exploitation des ressources ou des instances pour un *matching* sémantique.**

4. Le filtrage à base de seuil est la seule méthode adaptée par les travaux existants pour réduire le nombre des fausses correspondances ; pourtant, **l'existence des liens hiérarchiques et sémantiques entre les concepts de chaque ontologie peut jouer un rôle important pour détecter les anomalies et les contradictions entre les *mappings* obtenus.**

5. Le *mapping* d'ontologies est souvent un processus interactif (par exemple, PROMPT). Lorsque l'outil suggère le résultat du *mapping* à l'utilisateur, ce dernier peut confirmer ou infirmer ces propositions. Grâce à cette interaction, l'outil propose des suggestions plus précises. Malheureusement, dans plusieurs travaux de *mapping* comme MAFRA et MOMIS, il n'est pas clair de savoir si ces outils disposent d'un processus interactif ni comment cette interaction est réalisée. **L'interaction avec l'utilisateur est un point fondamental qu'il faudra intégrer dans les outils de *mapping*. Elle permet à l'utilisateur de donner son avis sur les propositions de *mapping*.**

6. Parmi les différentes approches de *mapping* cités auparavant, il n'y a que deux systèmes de *matching* (GLUE et S-Match) qui sont complètement automatisés, dans le sens où les similarités entre les concepts des deux ontologies à faire correspondre sont identifiées sans aucune intervention de l'utilisateur. Cependant, le *matching* d'ontologies est juste une étape

dans le processus de *mapping*. Par conséquent, ces approches automatisent seulement une partie de ce processus. **Il faudrait donc aller plus loin dans l'automatisation du *mapping*.**

7. Les systèmes d'intégration tels que MOMIS et ONION utilisent des outils spécifiques (respectivement, ARTEMIS et SKAT) pour la découverte de la similarité entre les concepts des ontologies. Ces outils sont typiquement intégrés dans le système dans lequel l'intervention de l'utilisateur dans le processus de *mapping* est nécessaire. **Il y a donc un grand besoin d'avoir des outils indépendants des systèmes qui les utilisent, de façon à favoriser leur réutilisation.**

Chapitre III : OMIE – Une approche pour le *mapping* d'ontologies

III.1 Le système OMIE

III.1.1 Introduction

Le problème actuel lié aux ontologies est qu'étant donné un même domaine ou des domaines connexes, il est possible que plusieurs ontologies soient disponibles, car elles sont développées simultanément par plusieurs communautés différentes. Le choix d'une ontologie particulière et/ou l'exploitation de plusieurs ontologies en même temps devient difficile. Le besoin de comparer les termes des ontologies, de passer de l'une à l'autre ou d'échanger les ressources et les instances entre des bases des ressources indexées par des ontologies devient donc nécessaire. L'objectif principal de notre système OMIE (*Ontology mapping within an Interactive and Extensible environment*) est de :

- Définir un système capable de réconcilier deux bases d'instances (ou deux entrepôts de ressources), chacune décrite par une ontologie. Ceci inclut la découverte des *mappings* entre ces deux ontologies.
- Réaliser un système automatique pour le *mapping* d'ontologies en réduisant l'implication directe de l'utilisateur dans le processus de validation. Dans les systèmes existants de *mapping*, l'utilisateur intervient directement pour valider ou refuser les *mappings* proposés. Avec le système OMIE, l'utilisateur cherche à accéder à des ressources en utilisant les termes de son ontologie locale, puis le système fait correspondre l'ontologie locale avec une ontologie distante et renvoie une liste de ressources (locales et distantes). La validation du *mapping* se fait en fonction de la satisfaction de l'utilisateur par rapport aux ressources renvoyées.

- Créer un système interactif capable d'utiliser et d'exploiter les connaissances des utilisateurs afin d'avoir un processus de validation des *mappings* plus fiable et évolutif.
- Développer un système extensible, incrémental et configurable : la modification de la configuration initiale du système par un utilisateur/expert est très souhaitable afin d'adapter le système aux particularités du domaine d'application. Par exemple, on utilisera entre autres une méthode de similarité à base d'UMLS pour le domaine biomédical.
- Différentes techniques de manipulation d'ontologies existent (fusion, intégration, alignement). Toutes ces techniques consistent à trouver les correspondances entre les concepts des différentes ontologies. Ce travail de génération de *mapping* statique peut très vite s'avérer coûteux, car il faut faire toutes les combinaisons possibles entre tous les concepts d'ontologies. Il peut même s'avérer inutile dans certains cas, car certains *mappings* risquent de ne jamais servir. D'où l'utilité de réaliser un système capable de produire des *mappings* « à la demande », c'est-à-dire seulement si c'est nécessaire. Par exemple, lorsqu'un utilisateur désire utiliser une ressource non disponible localement, le système cherche à faire correspondre le (ou les) concept(s) développant cette ressource.

La suite de ce chapitre est partagée en trois sections. Tout d'abord nous montrons les fonctionnalités et les cas d'utilisation du système OMIE. Ensuite nous expliquons l'algorithme de *mapping* utilisé pour détecter les correspondances et enfin nous présentons l'architecture du système OMIE qui se base sur une architecture multi-agents.

III.1.2 Les cas d'utilisations et les fonctionnalités

Pour bien comprendre les fonctionnalités et les besoins d'utilisation de notre système, nous commençons tout d'abord par un simple scénario d'utilisation du système OMIE dans le cadre biomédical.

« Thomas R. est un biologiste. Dans le cadre de ses recherches, il veut connaître les maladies (pathologies) du rein. Il ne possède pas cette information dans l'ontologie qu'il utilise habituellement et qui ne décrit que les organes. Il se connecte sur OMIE, s'identifie, le système reconnaît son profil et lui propose l'interface de travail des biologistes avec son ontologie habituelle. Thomas R. veut accéder à l'ontologie PATH[1] ou plus exactement au concept rein (ou équivalent) de PATH. OMIE lui propose alors le mode d' « intégration sélective » qui consiste à trouver les correspondances d'un concept d'une ontologie dans une autre ontologie.

[1] L'ontologie PATH est une ontologie qui regroupe et organise les pathologies.

Les résultats de ce *mapping* (mono-concept) sont affichés avec des degrés de similarité décroissants. Thomas choisi le premier lien et récupère ainsi les pathologies (le sous-arbre) liées au rein.

Dans l'équipe de Thomas R., d'autres chercheurs trouvent cela très intéressant et veulent également récupérer les pathologies de plusieurs autres organes. Dans ce cas, Thomas R. choisi le 2ème mode d'intégration « intégration totale » qui consiste à générer tous les *mappings* entre son ontologie locale et l'ontologie PATH. Les résultats obtenus par OMIE sont ensuite validés par Thomas R., qui connaît bien le domaine.

Afin d'améliorer les résultats obtenus, Pierre A., l'administrateur du système, décide d'utiliser un nouveau *matcher* publié par l'université de Hanovre. Pour cela, il dispose d'une interface lui permettant d'inhiber certains *matchers* ou d'en ajouter de nouveaux. Il intègre le nouveau *matcher* et lui associe un degré de confiance arbitraire. Ce degré sera revu à la baisse ou à la hausse (via un bouton dédié) après quelques tests. »

Nous décrivons maintenant les différentes fonctionnalités offertes par OMIE afin de répondre aux besoins d'utilisation d'un tel système de *mapping* d'ontologies. Elles peuvent être divisées en trois catégories principales (Figure 13) :

- Le mode requête permet de demander le *mapping* entre deux ontologies en choisissant une des deux approches proposées pour l'intégration. Il offre ainsi à l'utilisateur la possibilité de donner son avis sur les résultats fournis ;
- Le mode administration permet à l'administrateur du système OMIE d'ajuster les différents paramètres tels que : les seuils, les coefficients de confiance des *matchers*, etc. ;
- Le mode expert permet à l'expert du système d'analyser et de valider les résultats de *mapping*.

L'analyse de l'interaction homme-machine nous mène à concevoir des interfaces utilisateurs qui peuvent effectivement répondre au besoin des trois catégories d'utilisation, à savoir : (i) les utilisateurs finaux tels que les biologistes, les chercheurs et les médecins s'il s'agit d'une application biologique ou les professeurs et les étudiants s'il s'agit d'une application éducative, etc. ; les (ii) administrateurs et (iii) les experts du domaine. Ces interfaces vont simplifier l'accès rapide à toutes les fonctions de notre système. En effet, le système doit répondre à tous les besoins des utilisateurs, quelque soient leurs fonctions : répondre aux questions des utilisateurs finaux et extraire leurs retours (feedbacks), permettre à des administrateurs de modifier les paramètres des *matchers* et fournir à l'expert une interface qui facilite et accélère le processus de validation.

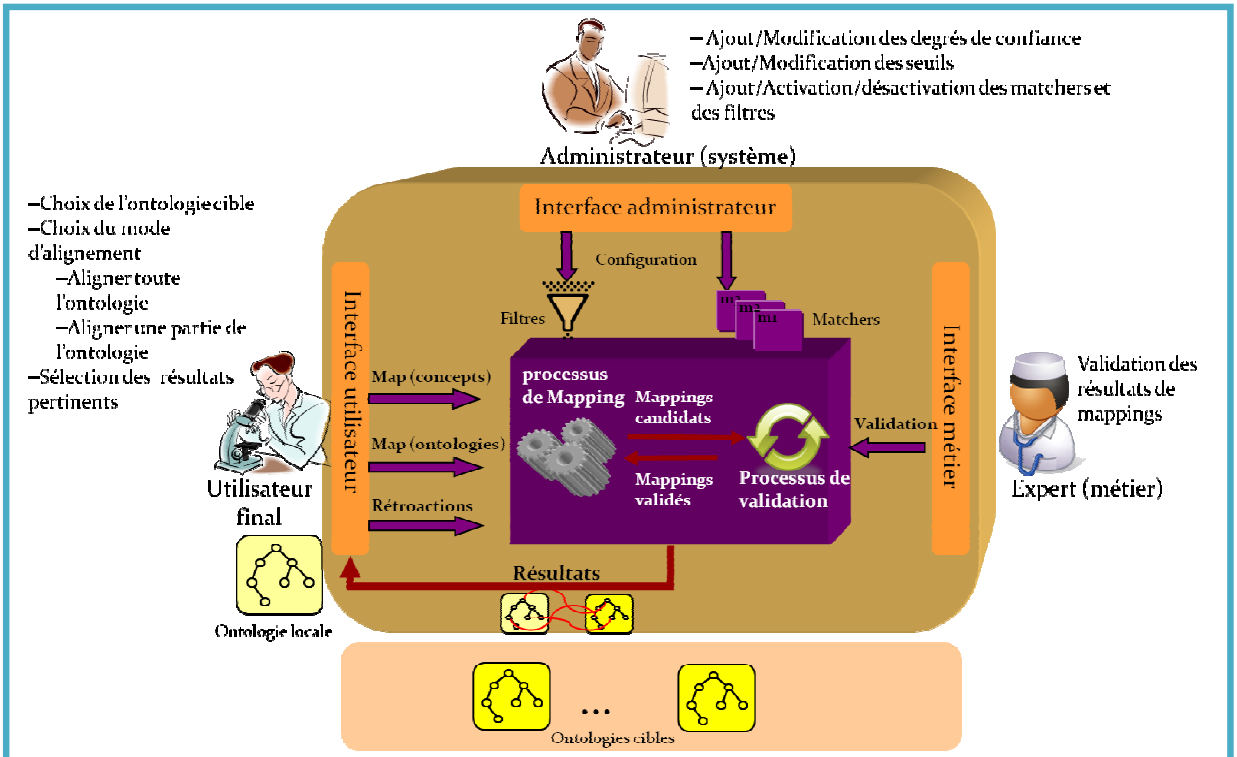


Figure 13. Les fonctionnalités du système OMIE

Après cette présentation des principales fonctionnalités d'OMIE du point de vue des utilisateurs, nous détaillons maintenant le processus de *mapping* du système OMIE.

III.2 Le processus de mapping du système OMIE

Cette section a pour but la définition du modèle conceptuel du *mapping* au sein du système OMIE. Nous commençons tout d'abord par une présentation du principe général de l'algorithme adopté. Cette phase consiste à montrer les différentes étapes de l'algorithme afin de produire les *mappings*. Les mécanismes et les méthodes du processus de *matching* sont abordés dans la section suivante.

III.2.1 Principe général de l'algorithme de *mapping*

Afin de produire des *mappings* entre deux ontologies, OMIE exécute un processus qui contient plusieurs étapes, chacune basée sur un ou plusieurs mécanismes de comparaison (voir Figure 14).

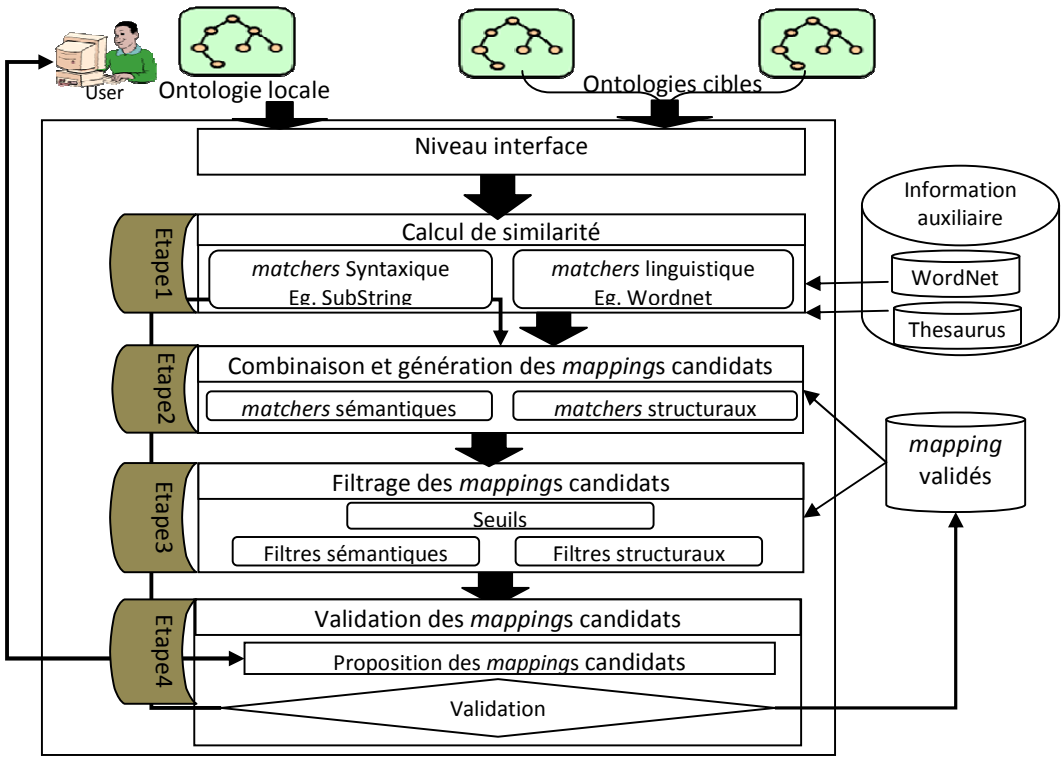


Figure 14. L'architecture du système OMIE pour le *mapping* d'ontologies

Etape 1 : Calcul de similarité: les *matchers* à base de comparaisons syntaxiques et linguistiques sont appliqués sur des couples de concepts afin de mesurer leur similarité terminologique. Nous employons dans notre cas plusieurs méthodes syntaxiques qui calculent par exemple la distance d'édition entre deux concepts, et/ou encore des méthodes linguistiques basées par exemple sur le dictionnaire WordNet. Chacune de ces méthodes renvoie une valeur de similarité. Par la suite nous donnons plus de détails concernant les *matchers* linguistiques et syntaxiques utilisés dans le système OMIE.

Etape 2 : Combinaison et génération des *mappings* candidats (ou des hypothèses de *mapping*) : les valeurs de similarité retournées dans l'étape précédente sont alors combinées en utilisant la formule suivante afin de produire une seule valeur de similarité SV entre chaque couple de concepts :

$$SV = \frac{\sum_i Conf\ \eta i \times SV\ \eta i}{\sum_i Conf\ \eta i}$$

Confηi et SVηi sont respectivement le niveau de confiance du *matcher* ηi et la valeur de similarité retournée par ce *matcher* ηi pour un couple de concepts donné.

Nous appelons cette liaison entre ces deux concepts une hypothèse de *mapping* ou encore un *mapping* candidat, et on la note « CM ». La génération d'une seule relation de correspondance entre deux concepts donne lieu par la suite à l'exécution d'autres *matchers*, tels que les *matchers* structuraux (topologiques) et sémantiques. Ils sont également appliqués pour améliorer les valeurs de similarité entre les *mappings* candidats et/ou encore pour produire de nouveaux *mappings* candidats. Ils sont basés sur des relations existantes entre les concepts d'ontologie et les *mappings* déjà établis et validés.

Etape 3 : Filtrage des *mappings* candidats : une fois l'ensemble des *mappings* candidats générés, nous employons des méthodes de filtrage afin d'éliminer les moins pertinents. Ces méthodes sont basées sur les relations structurelles et sémantiques entre les concepts et sur les *mappings* déjà validés. Un seuil est également utilisé pour écarter les hypothèses de *mapping* avec une valeur de similarité très basse.

Etape 4 : Le processus de validation : l'ensemble des hypothèses de *mapping* fournies après la phase de filtrage est ordonné suivant les valeurs de similarité. Ensuite l'utilisateur choisit l'hypothèse la plus pertinente pour sa requête. Ce retour (ou rétroaction) de l'utilisateur (appelé feedback) est sauvegardé pour une exploitation à postériori.

Nous décrivons dans les sections suivantes le principe et les règles de *matching* (étape 1 et étape 2), de filtrage (étape 3) et de processus de validation (étape 4).

III.2.2 Le processus de *matching*

L'objectif principal de la problématique de *mapping* d'ontologies consiste à identifier les correspondances entre les entités des ontologies à faire correspondre. On appelle cette phase de découverte des *mappings* : le processus de *matching*. Il s'agit de la combinaison d'un ensemble de méthodes de comparaison (des *matchers*) qui calculent le niveau de similarité entre deux entités.

Ces méthodes nous permettent de calculer la meilleure correspondance entre des couples d'entités. Elles peuvent maximiser la découverte du nombre de couples similaires et réduire le nombre de ceux qui sont dissimilaires. Il y a plusieurs façons de calculer la similarité (ou la dis-similarité) en utilisant des méthodes conçues dans le cadre de l'analyse des données, de l'ingénierie linguistique, des statistiques ou de la représentation des connaissances. Leur utilisation dépend des objets à comparer, de leur contexte et parfois de la sémantique externe de ces objets. Nous classifions par la suite ces différentes méthodes en deux grandes familles :

1. les *matchers* locaux, qui utilisent la comparaison des couples d'entité en tant que mots (des chaînes de caractères) pour résoudre les différents types d'hétérogénéité : linguistique, lexicale, syntaxique, etc. et

2. les *matchers* globaux, qui exploitent les relations hiérarchiques et/ou rhétoriques (appelées aussi relations sémantiques) de l'ontologie.

III.2.2.1 Matchers linguistiques et syntaxiques ou matchers terminologiques

Les *matchers* linguistiques et syntaxiques, appelés *matchers* terminologiques, consistent à comparer les termes. Les méthodes terminologiques comparent des chaînes de caractères. Elles peuvent être appliquées au nom, au label, ou au commentaire des entités (par exemple : les concepts ou les propriétés de l'ontologie), pour trouver ceux qui sont semblables. Par exemple, une chaîne de caractères S est la sous-chaîne d'une autre chaine T, s'il existe deux chaînes de caractères S_0 et S_{00} telles que $S_0+S+S_{00} = T$. Les deux chaînes S et T sont égales (S = T) si et seulement si S est sous-chaîne de T et T est sous-chaîne de S. Le nombre d'occurrences de S dans T est le nombre de couples distincts des sous chaînes S_0 et S_{00} tels que $S_0 + S + S_{00} = T$.

Il existe plusieurs façons d'évaluer la similarité entre deux entités, la plus courante consiste à définir une mesure de similarité. Nous présentons les définitions et les caractéristiques de quelques méthodes de mesure de similarité tels que : la similarité, la dis-similarité et la distance. La connaissance de ces terminologies est primordiale pour comprendre le fonctionnement des *matchers* de notre système.

Définition de la mesure de similarité : c'est une fonction définie par f : O×O→R (O ensemble des objets et R ensemble des réels). Elle retourne une valeur numérique qui exprime le degré de similarité entre deux objets. Cette fonction satisfait les propriétés suivantes :

- Positive : $\forall(x,y)\in O\times O\ f(x,y) \geq 0$;
- Maximale : $\forall(x,y,z)\in O\times O\times O\ f(x,x) \geq f(y,z)$;
- Symétrique : $\forall(x,y)\in O\times O\ f(x,y) = f(y,x)$.

Définition de la mesure de dis-similarité : cette mesure est l'opposée de la mesure de similarité. Elle est définie par f : O×O→R. Elle retourne une valeur numérique qui exprime le degré de dis-similarité (ou la divergence) entre deux objets. Cette fonction satisfait les propriétés suivantes :

- Positive : $\forall(x,y)\in O\times O\ f(x,y) \geq 0$;
- Minimale : $\forall x \in O\ f(x,x) = 0$;
- Symétrique: $\forall(x,y)\in O\times O\ f(x,y) = f(y,x)$.

Afin de calculer la dis-similarité entre les entités, nous utilisons la notion de distance entre les entités.

Définition de la distance ou encore de la métrique : c'est une fonction qui calcule la dis-similarité entre deux objets f : O×O→R. Elle satisfait la propriété suivante :

- Inégalité triangulaire : $\forall(x,y,z)\in O\times O\times O\ f(x,y) + f(y,z) \geq f(x,z)$;

Les méthodes de *matching* de type syntaxique sont des méthodes de similarité qui se basent sur la comparaison des chaînes de caractères. Ce genre de *matcher* tire profit de la structure de la chaîne de caractères, c'est-à-dire en tant que séquence de lettres. Il existe plusieurs méthodes de comparaison qui permettent de calculer la distance entre deux chaînes de caractères, suivant le type d'anomalie ou d'hétérogénéité que l'on cherche à résoudre.

Avant de comparer directement les chaînes de caractères qui ont une signification dans le langage naturel, un certain nombre de procédures de normalisation permettent d'améliorer les résultats de la comparaison. Nous listons par la suite quelques méthodes utilisées dans notre système pour la normalisation :

- La normalisation de la casse (majuscule/minuscule) : consiste à convertir chaque lettre de la chaine en minuscule (ou majuscule);
- La suppression de signes accentués : consiste à remplacer les caractères de type accentués (comme le « è »). Par exemple, nous remplaçons le mot Montréal par Montreal ;
- La normalisation des espaces : consiste à normaliser tous les espaces (comme : blanc, tabulation, retour de chariot, etc.) par un espace simple ;
- La normalisation des mots composés : permet de normaliser quelques liens entre les mots, comme remplacer les apostrophes et les tirets par un espace simple. Par exemple les mots : dataBase, data-base et data base;
- La suppression des chiffres : cette méthode doit être employée avec soin dans certains contextes. Par exemple il y a des noms chimiques contenant des chiffres ;
- L'élimination de la ponctuation : utile quand on compare des mots au lieu de comparer des phrases ;
- L'élimination des mots de liaison : consiste à éliminer les mots de liaison tels que : « à », « de », etc. Ceci est habituellement employé pour comparer les textes longs.

Pour mettre en correspondance deux chaînes de caractères, une normalisation est d'abord réalisée (comme c'est indiqué au-dessus) puis une intervention de plusieurs *matchers* est exécutée d'une manière parallèle. Chacun de ces *matchers* permet de résoudre un type spécifique d'hétérogénéité. Une valeur de similarité (SV) sera identifiée entre deux entités (mots) par chaque *matcher*. Cette valeur indique le degré de similarité (ou la distance) entre ces deux mots.

La granularité et la fiabilité de ces différents *matchers* ne sont pas égales ; par exemple, le *matcher* d'égalité totale est plus exact que celui basé sur la distance de hamming (voir les détails ci-après). Un degré de confiance (Cf) sera associé à chaque *matcher* par l'administrateur du système avant le lancement de celui-ci.

Nous déduisons alors que chaque *matcher* (noté ηi) donne une indication sur la similarité de deux concepts, mais aucun ne prévoit le *mapping* final. Il est de la forme <ηi, R, c, c', $Cf_{\eta i}$, $SV_{\eta i}$ >, avec :

- ηi est le nom du *matcher*. Il peut être un *matcher* linguistique, syntaxique, structural ou sémantique.
- R est la relation produite entre les deux concepts c et c'. Nous avons trois types de relations possibles : {contient : ⊇, inclut : ⊆, ou équivalent : ≡}.
- c et c' sont les deux concepts à comparer.
- $Cf_{\eta i}$: est le degré (ou le niveau) de confiance associé au *matcher* ηi.
- $SV_{\eta i}$: est la valeur de similarité identifiée par ηi entre c et c'.

Nous définissons ci-après les *matchers* utilisés dans OMIE dans la première étape de *matching*. Nous distinguons entre deux catégories :

1. Les *matchers* syntaxiques : calculent la similarité (ou la dis-similarité) entre deux chaînes de caractères en utilisant des fonctions et des méthodes de comparaison à base d'une suite de caractères.
2. Les *matchers* lexicaux ou linguistiques : utilisent en général des ressources auxiliaires pour comparer des mots. OMIE emploie deux sources d'informations auxiliaires à savoir le dictionnaire WordNet et le thesaurus UMLS (pour l'application biomédicale).

a) Les *matchers* syntaxiques

Définition du matcher à base d'égalité des chaînes de caractères :

La méthode d'égalité des chaînes de caractères est une mesure de similarité entre deux chaînes. Elle est définie par f : S × S → [0, 1]. Cette mesure retourne 1 si les deux chaînes sont égales et elle retourne 0 sinon. Formellement : f(x,x)=1 et f(x,y)=0 ∀ x#y.

Cette méthode, comme les autres méthodes de comparaison, est exécutée après une certaine normalisation syntaxique des deux chaînes de caractères à comparer. Malheureusement, cette mesure de similarité n'indique pas la valeur de différence entre les deux chaînes. On applique d'autres méthodes de comparaison telle que la distance de Hamming, qui est capable de donner un indice de similarité plus exact et plus significatif.

Définition du matcher à base de la distance de Hamming :

Le *matcher* Hamming est basé sur la distance de hamming. Celle-ci est définie par Richard Hamming et est utilisée en informatique, en traitement du signal et en télécommunications. Elle joue un rôle important dans la théorie algébrique des codes correcteurs et permet de quantifier la différence entre deux séquences de symboles.

La distance de Hamming est une mesure de dissimilarité entre deux suites de caractères. Elle compte le nombre de positions dans lesquelles les deux chaînes de caractères diffèrent. Elle est définie par $f : S \times S \rightarrow [0, 1]$ tel que :

$F(x,y)=(\Sigma\min(|x|,|y|)\ x[i]\#y[i]) + (||x|-|y||))/\max(|x|,|y|)$.

Définition du matcher à base de la distance subString :

Le *matcher* à base de la distance *subString* permet de calculer la mesure de dis-similarité entre deux chaînes de caractères. Cette mesure peut être utile dans le cas où on considère que deux chaînes de caractères sont semblables, ou si l'une des deux est une sous-chaîne de l'autre. Formellement, la distance SubString est une fonction définie par : $f : S \times S \rightarrow [0, 1]$ tel que :

$F(x,y)=1$ si $\forall(x,y)\in S*S\ \exists(a,b)\in S*S$ tel que $x=a+y+b$ ou $y=a+x+b$;

$F(x,y)=0$ sinon.

Nous raffinons cette distance afin de mesurer la proportion de la sous-chaîne commune entre les deux chaînes à comparer. Par conséquent, nous définissons la fonction qui calcule la distance de subString par la formule suivante :

$F(x,y)=2*|c|/|x|+|y|$ avec c la plus grande sous-chaîne commune entre les deux chaînes x et y.

Il est facile de voir que cette mesure est en effet une similarité entre la totalité des chaînes. On a pu également considérer une similitude entre des sous parties de ces chaînes. Cette définition peut être employée pour mesurer la similitude basée sur le plus grand préfixe commun, le plus grand suffixe commun, etc. Effectivement, la distance de N-gramme est utilisée pour ce genre de comparaison.

Définition du matcher à base de la distance N-gramme :

La distance n-gram calcule le rapport entre le nombre de n-grams communs au-dessus du nombre total de n-grams entre deux chaînes de caractères. Typiquement, soit ngram(s,n) l'ensemble de toutes les sous-chaînes de 's' de longueur 'n', la distance ngram entre deux chaînes s et t est définie par la fonction de dis-similarité suivante :

$\sigma=|ngram(s,n) \cap ngram(t,n)|/n*Min(|s|;|t|)$

Cette mesure de dis-similarité est efficace quand les caractères sont seulement absents dans l'une des deux chaînes comparées.

Définition du matcher à base de la distance de Levenshtein :

Le *matcher* Levenshtein est basé sur La distance de Levenshtein [53]. Cette distance mesure la similarité entre deux chaînes de caractères. Son nom provient de Vladimir Levenshtein qui l'a définie en 1965. Cette distance permet de savoir précisément le nombre minimal d'insertions, de suppressions et de substitutions des caractères requis pour transformer une chaîne en une autre. A chaque opération est assigné un coût. Si le nombre de différences entre les deux chaînes est grand alors le coût de transformation est important.

La distance de Levenshtein peut être considérée comme une généralisation de la distance de Hamming. On peut montrer en particulier que la distance de Hamming est un majorant de la distance de Levenshtein.

Par exemple, pour passer de "CHIEN" à "NICHE", il faut quatre opérations :

0. CHIEN ;
1. CHEN (par la suppression de I)
2. CHE (par la suppression de N)
3. ICHE (par l'ajout de I)
4. NICHE (par l'ajout de N)

Il y a peut-être une autre façon de faire, mais vous n'arriverez pas à moins de 4 opérations.

b) Les matchers linguistiques à base d'informations auxiliaires

Définition du matcher à base du dictionnaire WordNet [89]***:***

Les méthodes basées sur un langage se fondent sur des techniques de traitement du langage naturel (NLP) afin de trouver des associations entre les entités ou les classes. Ces méthodes exigent l'utilisation de ressources externes, par exemple les dictionnaires. Elles se basent essentiellement sur les liens sémantiques entre les termes en langage naturel, utilisant un lexique ou un dictionnaire externe. Par exemple, elles considèrent les synonymes comme des entités équivalentes et les hyponymes comme des entités qui se recouvrent (c.-à-d. l'une contient l'autre). Typiquement, le dictionnaire WordNet est employé pour trouver ce genre de relations lexicales entre les termes comparés comme les relations de synonymies, hyponymies, hyperonymies et d'autres.

WordNet est un projet sur la langue anglaise qui a été conçu par Georges Miller [62]. Il fournit des descriptions détaillées et précises des mots. Il arrive parfois que l'on rencontre plus de 20 sens pour un verbe (par exemple le mot « *give »* a 27 sens).

Dans WordNet, les différentes catégories syntaxiques sont étudiées séparément pour des raisons méthodologiques et techniques. Elles ne sont pas toutes étudiées : par exemple, on ne trouvera pas dans WordNet les propositions, les conjonctions et les pronoms. Les catégories étudiées séparément sont celles des noms, des verbes, des adjectifs et des adverbes mais les auteurs considèrent que les relations entre les catégories devront être étudiées et détaillées.

Wordnet structure chaque catégorie syntaxique dans un axe paradigmatique, selon une conception qui mêle psychologie et linguistique. L'unité minimale de cet axe est appelée « *synset* ». Ce dernier est un ensemble qui contient tous les sens des mots qui expriment la même notion. La version 1.5 de WordNet comprend 90462 *synset*s. Dans cette version, 75812 *synset*s comportent un label (c.-à-d. une définition ou un exemple), et 14650 *synset*s ne l'ont pas. WordNet considère la polysémie comme un phénomène discret : si un mot a plusieurs sens, les identifiants de ces derniers apparaissent dans différents *synset*s. Il structure les *synset*s entre eux principalement par une relation d'hyponymie. Ces relations sont généralement mono-hiérarchiques, c'est-à-dire qu'un *synset* a au maximum un père. On trouve aussi d'autres relations comme l'antonymie et les relations partie-tout (méronymie et holonymie).

Le projet EuroWordNet est une extension de WordNet basée sur les mêmes idées mais développé en plusieurs langues européennes, dont le français, et ce avec la participation de l'Université d'Avignon, le Rank Xerox *Research* Center de Grenoble et la société Bertin SA.

Définition du matcher à base des informations auxiliaires UMLS :

Le métathésaurus UMLS (Unified Medical Language System) [85] est un réseau sémantique constitué d'environ 2 125 395 concepts du domaine, qui sont définis à partir de 100 terminologies médicales existantes (en 2009). Un concept est un cluster de termes issus d'une ou plusieurs sources et considérés synonymes (2 100 000 libellés). Ces concepts sont reliés par 10 millions de relations héritées des terminologies sources. Un autre composant de l'UMLS, le *Semantic Network*, est un modèle de haut niveau du domaine biomédical, comprenant 134 types sémantiques (e.g. *Organism*, *Disease* or *Syndrome*) organisés en arbre. D'autres relations entre types sémantiques permettent de représenter des connaissances de haut niveau comme la relation *diagnoses* entre les deux types sémantiques *Sign* ou *Symptom* et *Disease* ou *Syndrome* exprimant la connaissance générale du domaine « les symptômes permettent de diagnostiquer des maladies ». Chaque concept du méta-thésaurus

possède un ou plusieurs types sémantiques. Par exemple, la définition issue de MeSH du concept frontal lobe (C0016733) est : *the anterior part of the cerebral hemisphere*.

Il possède les types sémantiques : *Body Part*, *Organ*, ou *Organ Component*. Il correspond au cluster de termes : *frontal lobe, frontal cortex, frontal lobe (brain), frontal region <2>, Lobus frontalis, Tissue of frontal lobe of brain*. Il est fils de *Cerebral lobe* et de *Cerebral Cortex* (la relation est issue non seulement de *MeSH*, mais aussi de *Neuronames Brain Hierarchy* [Bowden], terminologie de référence de la neuro-anatomie). Aucune de ces deux relations hiérarchiques n'est typée, d'où l'impossibilité de raisonner sur les méronomies et les taxonomies, la relation *part-of* avec l'un et *is-a/kind-of* avec l'autre fils n'étant pas explicitée. Cet exemple illustre aussi qu'un cluster de termes autorise une représentation des connaissances très souple. Par exemple, le terme *frontal region* existe avec deux occurrences différentes : *frontal region <1> (C0549224)* a pour type sémantique *Body Location or Region*, donc est une région alors que *frontal region <2>* est assimilé à *frontal lobe*. On note aussi que *frontal lobe, frontal cortex*, et *tissue of frontal lobe of brain* dénotent des objets de nature différente : *lobe* (une partie du cerveau), *cortex* (la surface du cerveau), *tissue* (la matière qui constitue le cerveau).

III.2.2.2 Combinaison des *matchers* et génération des hypothèses de *mapping*

Pour les *matchers* syntaxiques et linguistiques (lexicales), la comparaison des termes se fait d'une manière indépendante, c'est-à-dire on calcule la similarité entre deux termes en se basant juste sur la fonction ou la distance utilisée par le *matcher,* sans besoin de connaître le résultat fourni par les autres *matchers*. Par conséquent, nous utilisons la combinaison parallèle des *matchers* ; le résultat des différents *matchers* individuels (listés ci-dessus) sera combiné pour identifier un seul *mapping* candidat (appelé aussi hypothèse de *mapping*) entre chaque couple de concepts.

La Figure 15 schématise la composition parallèle des *matchers* syntaxiques et linguistiques au sein d'OMIE. Après une normalisation des deux chaînes de caractères S et T, tous les *matchers* comparent indépendamment cette normalisation et offrent des résultats (calcul de similarité) différents. Chacun de ces *mappings* candidats (notés MC_i ou HP_i) est un cinq-uplet : MC <R, c, c', $Conf_{Hp}$, SV_{Hp}> avec :

- R est la relation identifiée par les différents *matchers* individuels entre le couple de concepts c et c' ;
- $ConfHp = \sum_i Confi$ est le degré de confiance de l'hypothèse de *mapping* Hp et

$SVHp = \sum_i Conf\eta i \times SV\eta i \Big/ \sum_i Conf\eta i$ est la valeur de similarité produite par la combinaison des *matchers*.

Par exemple, étant donnés deux *matchers* η_1 et η_2 tel que η_1:<≡,c,d,2,0.5> et η2:<≡,c,d,1,0.7>, une hypothèse Hpi sera générée et définit par : Hpi : <≡,c,d,3,0.57>.

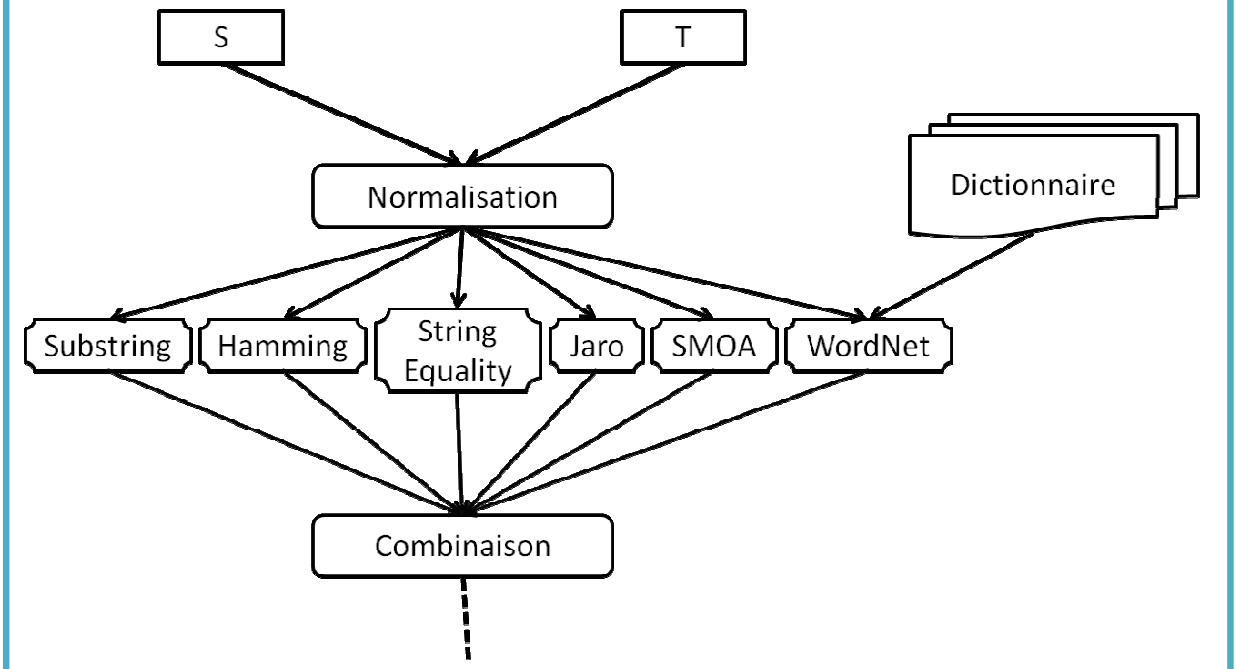


Figure 15. Compositions des *matchers* basés sur la comparaison des chaînes de caractères.

Nous notons l'ensemble des hypothèses de *mapping* produites par Ω :{Hp_i/i}. Ces hypothèses entre les couples de concepts vont être modifiées et améliorées en utilisant des règles et des méthodes de comparaison exploitant les relations ontologiques, c'est-à-dire les relations hiérarchiques et sémantiques (ou rhétoriques). Nous détaillons ci-après le principe de ces *matchers* structurels et sémantiques.

III.2.2.3 *Matchers* structurels et sémantiques

Après avoir obtenu quelques liens entre les concepts par la comparaison des chaînes de caractères, nous employons d'autres types de *matchers,* dans le but de trouver de nouvelles relations de similarité. Il s'agit des *matchers* structuraux et des *matchers* sémantiques dont l'utilisation permet d'augmenter la valeur de similarité

ou encore à produire de nouveaux *mappings*. Les exemples illustrés dans le tableau 4 montrent quelques règles de *matchers* structuraux et sémantiques en langage naturel :

Tableau 4. Les règles de matchers structuraux

Règles	Description
R1	Deux concepts sont similaires si leurs sur-concepts "père" sont similaires
R2	Deux concepts sont similaires si leurs sous-concepts "fils" sont similaires
R3	Deux concepts sont similaires si leurs "voisins" (c.-à-d. l'ensemble des concepts convergeant qui sont liés par une relation rhétorique) sont similaires

Chacune de ces règles donne une idée sur la similarité entre deux entités, mais typiquement aucune d'elles ne produit le *mapping* toute seule. Chaque règle appliquée nous fournit un poids de similarité entre deux entités comparées (concept ou relation). Un seuil est défini sur les valeurs de la similarité pour identifier la correspondance ou la non-correspondance.

L'idée principale des méthodes de comparaison structurelles (règles 1 et 2) et sémantiques (règle 3) est basée sur la notion d'ontologie morphisme qui se défini par : si les deux concepts c_1 et c_2 de l'ontologie locale « O » sont liés par une relation « R », leurs concepts correspondants c'_1 et c'_2 de l'ontologie cible « O' » doivent être liés par la même relation « R ». Cette relation « R » peut être soit (i) une relation structurelle, par exemple *is-a*, sous-concept ou sur-concept ; soit (ii) une relation sémantique, par exemple *synonym-of, has-part*, etc.

Concernant les *matchers* structuraux, ils comparent la structure (ou l'hiérarchie) ontologique des entités auxquels ils sont connexes. Mais l'application de ces méthodes, comme par exemple la création des relations entre les fils si leurs parents sont similaires, peut générer un nombre important de relations. Pour cette raison, nous utilisons les *matchers* structuraux pour créer des liens entre les groupes d'entités et/ou pour renforcer une relation existante plutôt que pour découvrir de nouveaux liens de correspondances entre les entités. Les résultats de ces *matchers* structurels sont combinés avec les autres méthodes linguistiques, syntaxiques et sémantiques. Il s'agit alors d'utiliser des relations de *mapping* déjà confirmées par le système. On note l'ensemble des *mappings* confirmés par Map <≡,c,d,conf,SV>.

Nous clarifions le principe des *matchers* structuraux et sémantiques à travers les algorithmes décrits ci-dessous. Nous définissons ci-après les deux *matchers* structuraux : « Top-Down » et « Bottom-up » avec :

- La variable nbr-Child (c) contient le nombre de concepts fils (les sous-concepts) du concept c.
- Les deux variables $Conf_{Top\text{-}Down}$ et $Conf_{Bottom\text{-}up}$ représentent respectivement le degré de confiance des *matchers* « Top-Down » et « Bottom-up »

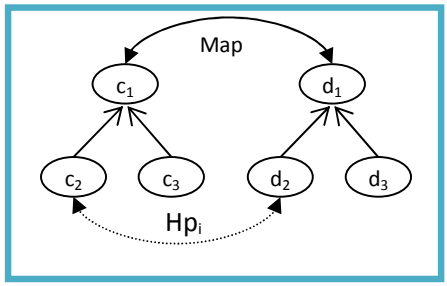
matcher Top-Down

SI ∃ Map :<≡,c_1,d_1,$Conf_1$,SV_1> et ∃ Hpi : <≡,c_2,d_2,$Conf_2$,SV_2>

Alors modifier les paramètres de Hpi par:

$SV_2 = SV_2 + (SV_1/Min(nbr\text{-}child(c_1),nbr\text{-}child(d_1)))$

$Conf_2 = Conf_2 + Conf_{Top\text{-}Down}$



matcher Bottom-up

SI ∃ Map : <≡,c_2,d_2,$Conf_2$,SV_2>

Alors :

Si ∃ Hpi : <c_1,d_1,$Conf_1$,SV_1> **alors** modifier Hpi par

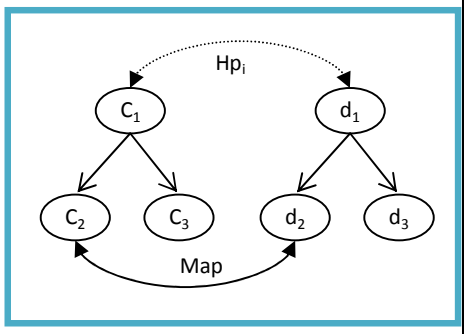
$SV_1 = SV_1 + (SV_2/Min\ [nbr\text{-}child(c_1),nbr\text{-}child(d_1)])$

$Conf_1 = Conf_1 + Conf_{Bottom\text{-}up}$

Sinon générer Hpi : <c_1,d_1,$Conf_1$,SV_1>

$SV_1 = SV_2/Min\ [nbr\text{-}child(c_1),nbr\text{-}child(d_1)]$

$Conf_1 = Conf_{Bottom\text{-}up}$



Algorithme 1. *matchers* structuraux: Top-Down et Bottom-up

La propriété transitive de la relation hiérarchique amène à définir des fils (respectivement des pères) directs et d'autres indirects. L'algorithme 1 montre une utilisation des fils (des pères) directs ; dans ce cas, on parle de l'exploitation de la structure d'ontologie au niveau un (1). Nous définissons le *matcher* top-down au niveau « j » de la même façon que le *matcher* direct top-down en introduisant le niveau « j » dans les deux variables SV et Conf tels que :

- $SV_2 = SV_2 + SV_1/j * Min[nbr\text{-}child\text{-}Level_j(c_1),nbr\text{-}child\text{-}Level_j(d_1)])$ avec $nbr\text{-}Child\text{-}Level_j$ (c) est le nombre des fils au niveau « j » du concept c.

- $Conf_2$=$Conf_2$+ $Conf_{Top\text{-}Down}$/j

Nous nommons le *matcher* structurel ou sémantique (η) appliqué au $j^{ème}$ niveau par $\eta_{/j}$. L'algorithme 2 donne un exemple d'application du *matcher* structurel « Top-Down » au $2^{ème}$ niveau.

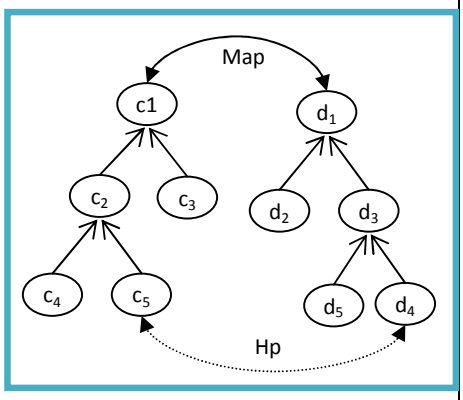
***matcher* Top-Down$_{/2}$**

SI ∃ Map :<≡,c_1,d_1,$Conf_1$,SV_1> et ∃ Hp : <≡,c_5,d_4,$Conf_2$,SV_2>

Alors modifier les paramètres de Hp par:

SV_2= SV_2 + (SV_1/2*Min[nbr-child-Level$_2$(c_1),nbr-child-Level$_2$(d_1)])= SV_2+SV1/2*Min[2,2]

$Conf_2$=$Conf_2$+ $Conf_{Top\text{-}Down}$/2



Algorithme 2. *matcher* structurel indirect

Une ontologie permet d'organiser de manière hiérarchique les concepts (super-concept et sous-concept) mais elle contient aussi un ensemble de relations rhétoriques dites sémantiques (par exemple, « est-synonyme-de », « partie-de », etc.). OMIE permet d'exploiter ces relations de la même façon que les relations hiérarchiques. Nous étudions les propriétés des relations rhétoriques telles que la symétrie et la transitivité et distinguons entre deux types d'exploitations possibles :

- Dans le cas où la relation est non-symétrique, nous aurons un *matcher* sémantique à base des pères et un autre à base des fils (voir Algorithme 3.) ;
- alors que si la relation est symétrique, le *matcher* sémantique sera défini à partir du voisinage sémantique de cette relation (voir Algorithme 4).

Les algorithmes 3 et 4 illustrent le principe des trois *matchers* sémantiques : (i) le *matcher* sémantique Top-Down, (ii) le *matcher* sémantique Bottom-up et (iii) le *matcher* sémantique de voisinage. Nous définissons les variables utilisées dans les algorithmes par :

- R est la relation sémantique définie dans les deux ontologies O et O' ;

- $|V\text{-Child}(c)|_R$= Ndr-OfSem-Child(c,R) est le nombre des concepts fils reliés au concept c par la relation non-symétrique R (c.-à-d., la partie 'Rang' de la relation R) ;
- $|V\text{-Father}(c)|_R$= Ndr-OfSem-Father(c,R) est le nombre des concepts pères reliés au concept c par la relation non-symétrique R (c.-à-d., la partie 'Domaine' de la relation R) ;
- $|V(c)|_R$= Ndr-OfSem-Concept(c,R) est le nombre des concepts reliés au concept c par la relation symétrique R ;
- $Conf_{Sem\text{-}R}$ représente le degré de confiance de la relation sémantique R.

***matcher* sémantique Top-Down**

SI ∃ Map :<≡,c_5,d_4,$Conf_1$,SV_1>

Alors

SI ∃ Hp :<≡,c_3,d_2,$Conf_2$,SV_2>

Alors

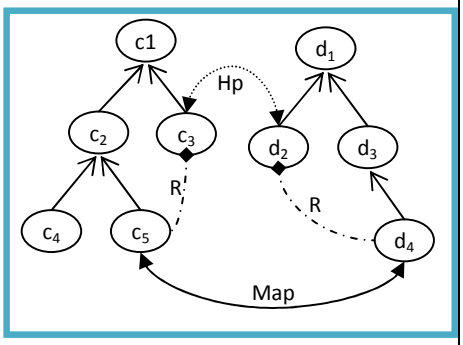
$SV_2=SV_2+(SV_1/Min[|V(c_5)|_R, |V(d_4)|_R])$;

$Conf_2=Conf_2+ Conf_{Sem\text{-}R}$;

Si non generate Hp<≡,c_3,d_2,$Conf_2$,SV_2>

$SV_2= SV_2 + (SV_1/Min[|V(c_5)|_R, |V(d_4)|_R])$;

$Conf_2= Conf_{Sem\text{-}R}$



la relation sémantique non-symétrique "R" (e.g. *part-Of*)

Algorithme 3. *matcher* sémantique Top-Down d'une relation non-symétrique

***matcher* sémantique**

SI ∃ Map :<≡,c_5,d_4,$Conf_1$,SV_1>

Alors

SI ∃ Hp :<≡,c_3,d_2,$Conf_2$,SV_2>

Alors

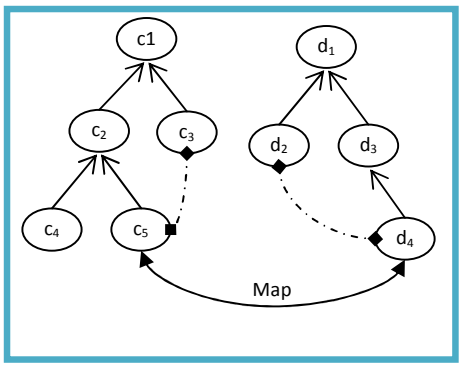
$SV_2=SV_2+(SV_1/Min[|V(c_5)|_R, |V(d_4)|_R])$;

$Conf_2=Conf_2+ Conf_{Sem\text{-}R}$;

Si non generate Hp<≡,c_3,d_2,$Conf_2$,SV_2>

$SV_2= SV_2 + (SV_1/Min[|V(c_5)|_R, |V(d_4)|_R])$;

$Conf_2= Conf_{Sem\text{-}R}$



la relation sémantique "R" symétrique (e.g. *synonymy*)

Algorithme 4. *matcher* sémantique d'une relation symétrique

III.2.3 Le processus de filtrage

Le besoin d'étudier et de détecter d'une manière automatique les relations du *mapping* non pertinentes nous a amené à enrichir notre processus de *mapping* par un processus de filtrage. Il s'agit de traiter l'ensemble des *mappings* candidats générés par le processus de *matching* (section précédente) avec des éléments filtrants (appelés 'Filtres') utilisant l'heuristique basée sur l'expérience des précédentes interactions et/ou sur le voisinage structurel et sémantique des ces éléments.

Chacun des filtres vise à réduire le nombre des fausses hypothèses de *mapping* en employant des règles de comparaison spécifiques capables de soulever les contradictions ou les anomalies. Ces règles peuvent exploiter la structure topologique ou encore les relations sémantiques liant les éléments de l'ontologie. Nous distinguons deux types de comparaison : (i) la comparaison des *mappings* candidats entre eux et (ii) la comparaison des *mappings* candidats avec des *mappings* confirmés dans les itérations précédentes.

Nous avons développé au sein du système OMIE plusieurs méthodes et règles pour filtrer les *mappings* candidats. Nous décrivons ci-dessous les filtres utilisés au sein d'OMIE classés suivant leur mode d'exploitation, c.-à-d :

1. des filtres exploitant les structures des entités à comparer, appelés les filtres structuraux ;
2. des filtres à base des relations sémantiques et rhétoriques de l'ontologie, appelés les filtres sémantiques ; et
3. des filtres à base de seuils.

III.2.3.1 Filtres structuraux

Nous avons développé plusieurs règles pour filtrer les *mappings* candidats fournis par la combinaison des différents *matchers*. Les différents filtres utilisés dans le système OMIE sont classifiés en trois familles. Dans cette section, nous expliquons l'une de ces trois familles, à savoir « les filtres structuraux ».

Il s'agit d'un ensemble de filtres qui exploitent entre autres les liens hiérarchiques (c.-à-d. structurels) pour identifier et éliminer les *mappings* candidats inadéquats, qui sont en général en contradiction avec la structure d'ontologie. L'idée principale de ces filtres est de respecter la même hiérarchie entre les concepts de l'ontologie source (O) et les concepts « images » de l'ontologie cible (O'). Nous montrons par la suite les schémas et l'algorithmique des filtres structuraux.

Le premier filtre (Filtre 1) illustre le principe du filtrage lors d'une contradiction entre *mappings* candidats (Hp_1 et Hp_2), qu'on appelle ainsi les hypothèses ou les *mappings* candidats croisés. Il s'agit de résoudre le problème de non respect de la relation hiérarchique (Fils →Père). Par exemple, le schéma suivant montre que le concept d_2 (respectivement d_1) est image du concept c_1 (respectivement d_2) par la relation du *mapping*. Comme c_2 est un sous-concept de c_1 (c.-à-d. il se trouve à un niveau inférieur et donc c_2 est plus spécifique que c_1), nous pouvons déduire que le concept d_1 (l'image de c_1) doit être plus spécifique que d_2 (l'image de c_1) ; et dans le cas contraire, on dit qu'il s'agit d'une contradiction entre deux *mappings* candidats croisés. Dans ce cas-ci, nous gardons le *mapping* candidat qui présente la plus grande valeur de similarité. On parle alors d'un croisement entre deux *mappings* candidats.

Filtre 1

SI

∃ $Hp_1 <\equiv, c_1, d_2, Conf_1, SV_1>$ &

∃ $Hp_2 <\equiv, c_2, d_1, Conf_2, SV_2>$

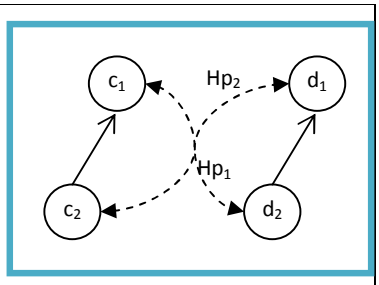
ALORS

SI $SV_1 > SV_2$

ALORS éliminer Hp_2

SINON

éliminer Hp_1



Le filtre 2 fonctionne de la même façon que le filtre 1 avec en plus l'utilisation des *mappings* générés précédemment. Dans certaines itérations, le système peut proposer un *mapping* candidat (Hp) qui sera en contradiction (ou croisement) avec un *mapping* déjà validé (Map). Dans ce cas, le choix est plus simple : on favorise le *mapping* confirmé et par conséquent la suppression de ce *mapping* candidat. On parle de croisement entre un *mapping* candidat et un *mapping* validé.

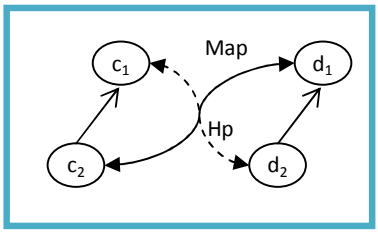
Filtre 2

SI

∃ $Hp <\equiv, c_1, d_2, Conf_1, SV_1>$ &

∃ $Map <\equiv, c_2, d_1, Conf_2, SV_2>$

ALORS éliminer Hp



Dans le filtre 3, le mapping validé sert également à supprimer une hypothèse de mapping qui n'est pas en adéquation avec ce mapping.

Filtre 3

SI

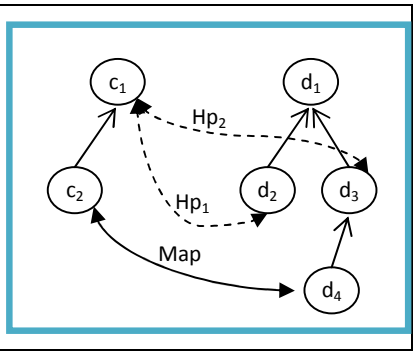
$\exists$ $Hp_1 <\subseteq, c_1, d_2, Conf_1, SV_1>$ &

$\exists$ $Hp_2 <\subseteq, c_1, d_3, Conf_2, SV_2>$) &

$\exists$ Map $<\subseteq, c_1, d_4, Conf, SV>$

ALORS

éliminer Hp_1



III.2.3.2 Les filtres sémantiques

Comme dans le cas des filtres structurels, nous exploitons les caractéristiques des relations sémantiques pour éliminer automatiquement certains liens de correspondance insatisfaisants. L'idée principale est de respecter les règles de l'ontologie morphisme. Par exemple, le filtre 4 illustre le principe du filtrage dans le cas ou nous avons deux *mapping* candidats qui sont contradictoires ou opposés. Nous définissons deux types de filtres sémantiques suivant les propriétés de la relation sémantique :

1. filtre à base d'une relation sémantique de type non-symétrique ; et
2. filtre à base d'une relation sémantique de type symétrique.

Filtre 4

SI

R est une relation non symétrique &

$\exists$ $Hp_1 <\subseteq, c_1, d_2, Conf_1, SV_1>$ &

$\exists$ $Hp_2 <\subseteq, c_2, d_1, Conf_2, SV_2>$

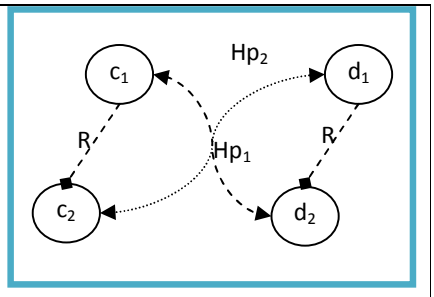
ALORS

SI $SV_1 > SV_2$ **ALORS**

éliminer Hp_2

SINON

éliminer Hp_1



III.2.3.3 Les filtres à base de seuil

Le filtre 5 illustre les filtres à base de seuils. Nous définissons un seuil (Th) sur les valeurs de similarité au-dessous desquelles les hypothèses ne sont pas considérées ; la valeur de Th peut être modifiée par l'administrateur du système.

Filtre 5

Pour chaque Hp <≡,c,d,Conf,SV>
SI SV<Th **ALORS** éliminer Hp

III.2.4 Processus de validation

La phase de validation est une phase clé dans le processus de *mapping*. Elle consiste à faire intervenir l'humain (qui peut être un simple utilisateur ou un expert du domaine) pour confirmer ou infirmer les liens de correspondance générés par le processus de génération et le processus de filtrage du *mapping*. Par conséquent, elle est primordiale car elle peut influencer soit positivement soit négativement les résultats engendrés par le système.

Le processus de validation est caractérisé par son aspect interactif. Il exploite les retours de l'utilisateur ainsi que sa satisfaction par rapport aux résultats proposés (par exemple, les ressources ou les documents demandés par l'utilisateur) et si nécessaire, il relance les deux précédents processus (de génération et de filtrage) avec de nouveaux arguments pour raffiner et/ou améliorer l'ancien résultat (Figure 16). Pour mieux comprendre le fonctionnement général du processus de validation, nous détaillons par la suite d'une part, cet aspect interactif et d'autre part, les différents retours et interactions possible de l'utilisateur avec notre système.

III.2.4.1 L'aspect interactif du processus de validation

Dans la littérature et dans un grand nombre de systèmes existants, les interactions avec l'utilisateur concernent directement la validation par l'expert du système, des liens de correspondance entre les termes des concepts des ontologies, ce qui rend cette phase de validation lourde et moins sûre. En revanche, notre objectif est d'introduire les expériences et les interactions de l'utilisateur final qui cherche, en général, à accéder à des ressources locales et distantes, autrement dit, faire impliquer les ressources (ou les instances) associées aux éléments d'ontologies dans les requêtes et les réponses d'utilisateur au lieu de valider directement les liens de correspondance entre les concepts associés à ces ressources. Nous parlons alors d'un *processus de validation indirecte*.

En effet, ce processus nous permet de réaliser un ensemble dynamique de liens de correspondance qui peuvent changer avec le temps grâce aux retours (*feedbacks*) des utilisateurs, via leurs interactions avec le système. Par conséquent, l'utilisateur accède à des ressources locales et distantes indépendamment des liens entre les termes de son ontologie locale et de l'ontologie distante.

Afin d'illustrer le principe de la validation indirecte et de l'interaction avec l'utilisateur, l'algorithme 5 résume le déroulement et les différentes étapes intermédiaires pour répondre à une requête utilisateur.

Déroulement d'une requête

Etape 1 : Requête utilisateur : je cherche les documents associés au concept 'c' de mon ontologie locale

Etape 2 : Processus de *matching* : génération d'un ensemble (•) de *mappings* candidats, MC:<≡,c,d_i,Conf,SV> avec les d_i concepts de l'ontologie distante

Etape 3 : Processus de filtrage : filtrage de l'ensemble • des MCs générés par le précédent processus, donc extraction d'un sous ensemble •'.

Etape 4 : Processus d'auto-validation : contient les étapes suivantes :

1. Suivant les caractéristiques de chaque MC de •', il confirme ou il infirme la validation automatique du MC. Chaque MC validé devient un *mapping* validé MV.
2. Tant qu'il y a des nouveaux MVs, aller à l'étape 2.
3. Il réorganise et ordonne l'ensemble des MCs et MVs par ordre décroissant de leur intérêt.
4. Suivant l'ordre établi auparavant, il extrait les ressources associées aux concepts d_i (dont le MC ou MV) ; par la suite, il envoie la liste de toutes ces ressources à l'utilisateur.

Etape 5 : Retour de l'utilisateur : il sélectionne un ou plusieurs ressources parmi la liste des ressources proposées par le processus de validation.

Etape 6 : Processus de validation : suivant la ou les ressource(s) sélectionnées par l'utilisateur :

1. Il identifie le MC ou le MV responsable de cette proposition pour les valider.
2. Tant qu'il y a des nouveaux MVs aller à l'étape 2.

Algorithme 5. Le déroulement d'une requête d'utilisateur

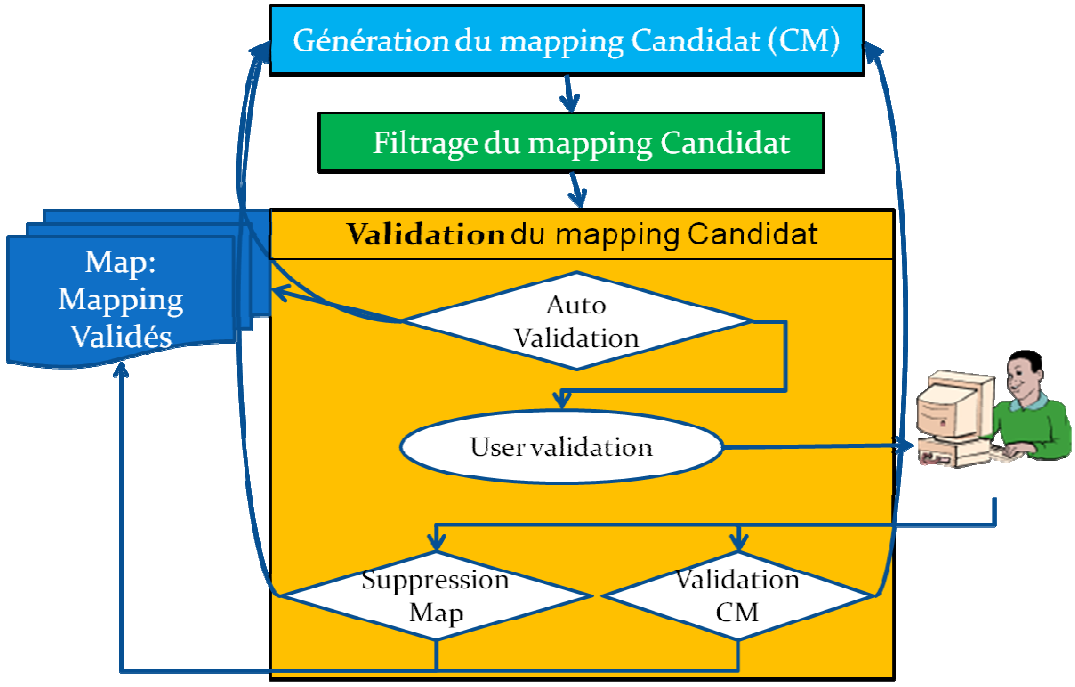


Figure 16. Le processus interactif de validation

III.2.4.2 Les retours (ou les rétroactions) d'utilisateur

Un aspect important du processus de validation est la rétroaction ou le retour utilisateur. L'utilisateur peut exprimer soit (i) directement son niveau de satisfaction par rapport aux liens de correspondance proposés par le système entre les concepts d'ontologies, soit (ii) indirectement en identifiant son degré de satisfaction par rapport aux documents (ressources) proposés par le système pour répondre à sa requête (dans le cas où il existe des ressources attachées aux concepts d'ontologie).

Dans le contexte pédagogique et avec le système SIMBAD, nous employons des informations sur l'utilisateur pour contrôler les retours et définir son degré de satisfaction. Ces informations sur l'utilisateur peuvent être ses compétences, son niveau de certitude, etc., obtenus à partir du modèle d'apprenant [17]. Avec ce modèle du système SIMBAD, un niveau de compétence est associé à chaque utilisateur selon son expérience. En outre, les résultats du *mapping* sont validés avec des niveaux de certitude différents. La combinaison de toutes ces informations permet d'une part, d'augmenter le niveau de la fiabilité du processus de validation et d'autre part d'améliorer les résultats du *mapping*.

Les *mappings* validés, appelé SVM (*Set of Validated mapping*), sont des 6-uplets Map: <rel, c, d, Conf, SV, N>, avec :

- c et d : couple de concepts des deux ontologies à comparer ;
- rel : relation de correspondance identifiée entre les deux concepts c et d ;
- Conf et SV : respectivement le degré de confiance et la valeur de similarité associés à cette relation de *mapping* ;
- N : nombre de fois où l'utilisateur a choisi et/ou a validé cette relation de *mapping*.

Lorsque le *mapping* produit est validé fréquemment par l'utilisateur, la valeur de « N » augmente tandis que le degré de confiance accordé aux autres correspondances (qui ne sont pas suffisamment validés) diminue.

III.3 Implémentation du système OMIE avec la technologie agent

Nous proposons d'utiliser la technologie des systèmes multi-agents (SMA) qui offre la possibilité à des agents spécialisés de s'exécuter de façon parallèle et concurrente. De plus, les agents utilisent leurs capacités d'apprentissage pour s'adapter et interagir avec les autres. Depuis quelques années, les systèmes multi-agents ont pris une place importante dans le domaine de l'informatique en général, et dans le domaine de l'intelligence artificielle et des systèmes distribués en particulier.

L'implémentation de notre système par la technologie des systèmes multi-agents est motivée par les raisons suivantes :

- Favoriser le parallélisme entre les méthodes de comparaison conquérantes. Par exemple, les *matchers* linguistiques et syntaxiques (s'ils sont combinés d'une manière parallèle) vont être exécutés indépendamment par des agents conquérants.
- Simplifier l'incrémentation et la modification du processus de *mapping*. Par exemple, l'ajout et/ou la suppression des *matchers* et des filtres se résume à l'ajout ou la suppression d'un agent spécifique.
- Faciliter l'échange des résultats et des connaissances entre les différents processus intermédiaires. Autrement dit, mettre en œuvre des agents pour le *matching*, des agents pour le filtrage et d'autres pour la validation.
- Améliorer le mode de stockage et la réutilisation des correspondances confirmées. Au lieu de mettre les résultats de *mapping* dans un fichier où son exploitation nécessite un traitement spécifique, nous présentons nos *mappings* sous forme de connaissances d'agents. Par exemple,

après l'alignement entre deux ontologies O_1 et O_2, nous enrichissons les connaissances des agents qui gèrent ces ontologies par des liens de correspondances identifiés entre ces dernières.

III.3.1 Les systèmes multi-agents (SMA)

Les SMA proviennent de la critique des approches séquentielles ou fonctionnelles et de l'intelligence artificielle classique. Sachant que l'intelligence d'un individu ne se forge que lorsqu'il est en contact avec d'autres individus de son espèce, cet individu que l'on appelle « *agent'* » a donc besoin d'interagir, de collaborer, de rentrer en conflit, de s'adapter et de communiquer avec son environnement pour pouvoir accomplir des buts beaucoup plus complexes. C'est de ce contexte qu'émane la notion de *SMA*.

Une autre motivation qui nous a poussé à implémenter notre système avec la technologie SMA est la capacité de s'adapter aux modifications et évolutions : changement des systèmes opératoires, changement de la configuration du système, mise à jour de l'ontologie et de l'entrepôt de ressources, ajout et suppression des fonctionnalités, etc. Les SMA du fait de leur nature distribuée, du raisonnement qui se fait localement, de l'ajout et la suppression d'agents au cours même du fonctionnement, facilitent énormément l'adaptabilité du système à toute évolution.

Nous adaptons pour le terme « ***agent »*** *la définition* suivante : Un agent est une entité virtuelle capable d'agir dans un environnement, de communiquer avec les autres agents, de s'engager sur des propositions et possédant des ressources propres. L'agent évolue dans un milieu qualifié de ***système multi-agents,*** que l'on peut définir comme un environnement composé d'agents, offrant des opérations permettant aux agents de percevoir, de manipuler, de produire des objets.

Les agents développés répondent à des besoins bien spécifiques d'une application, entraînant une diversité et une hétérogénéité au niveau des agents. Il apparaît donc un besoin naturel : celui de faire communiquer ces agents hétérogènes à travers un langage dit Langage de communication entre Agents. Un tel langage commun doit avoir une syntaxe non ambiguë pour que les agents puissent « **annoter et comprendre** » les messages de la même manière.

A l'heure actuelle, les deux langages les plus utilisés dans les SMAs sont **KQML** (*Knowledge Query and Manipulation Language*) [32] et **FIPA-ACL** (*Foundation for Intelligent Physical Agents*) [33]. Dans notre travail, nous employons le langage FIPA-ACL. Il est syntaxiquement similaire à KQML, mises à part certains noms de primitives réservées. FIPA-ACL dispose d'un ensemble de messages avec une sémantique associée, c'est-à-dire les conditions que doivent respecter l'expéditeur et les effets attendus sur le destinataire. Un message en FIPA-ACL peut être décrit comme suit :

```
(request
:sender X
:receiver Y
:content "<Class rdf:about='http://purl.org/obo/owl/MA#MA_02'></Class>"
:language OWL
:ontology http://purl.org/obo/owl/MA
)
```

III.3.2 L'architecture agent d'OMIE

L'architecture du processus de *mapping* du système OMIE se compose d'un ensemble d'agents, qui coopèrent entre eux afin de produire les *mappings* entre les ontologies à faire correspondre. Chaque agent a son propre comportement et communique avec son environnement (les agents et l'expert) en envoyant : des messages d'information, des requêtes ou des réponses. Notons que le degré de granularité n'est pas égal pour tous les agents utilisés par le système OMIE : certains d'entre eux jouent des rôles plus importants que d'autres. Le choix d'une architecture agent pour OMIE est assez naturel. Chaque *matcher* est représenté par un agent et nous avons introduit des agents pour chaque grande fonction d'OMIE (génération, filtrage, validation ainsi que médiation avec l'utilisateur).

La Figure 17 montre les cinq types d'agents utilisés par OMIE, à savoir : l'Agent d'Ontologie (OA), l'Agent *matchers* (MA), l'Agent de Génération des *mappings* candidats (des Hypothèses de *mapping*) (HGA), l'Agent de Filtrage d'Hypothèses (HFA) et enfin l'Agent de Feedback et Validation (FVA). Le système est complètement réparti, et tous les agents fonctionnent séparément. Chacun d'eux a un rôle spécifique, transmettant et/ou recevant des résultats sous forme de messages.

Les agents OA jouent l'intermédiaire entre l'utilisateur, les données et le système de *mapping*. En revanche, les autres agents communiquent et coopèrent entre eux pour réaliser les différentes tâches de *mapping* citées dans les sections précédentes.

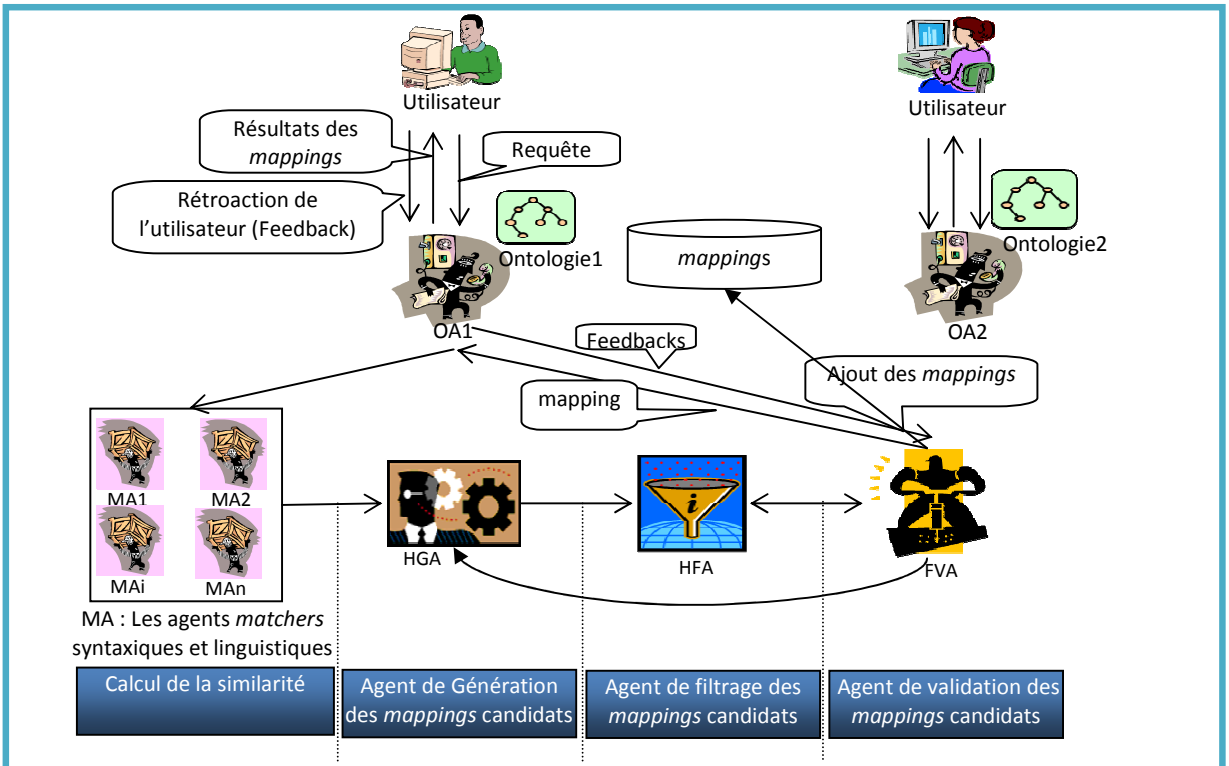


Figure 17. L'architecture multi-agents du système OMIE.

III.3.3 Comportement des agents

Nous décrivons dans cette section le comportement de chaque agent ainsi que ses différentes interactions et communications avec les autres agents au sein du processus de *mapping*.

III.3.3.1 Les agents ontologie : OA

Chaque agent ontologie (OA) est associé à une seule ontologie. Il joue le rôle d'une interface entre cette ontologie qualifiée de locale, la requête de l'utilisateur et les autres agents du système. L'agent OA communique d'une part avec les autres agents en langage FIPA-ACL et d'autre part avec l'ontologie locale par des requêtes en F-Logic. Ce type d'agent possède un autre rôle qui est de produire de nouvelles relations sémantiques entre les concepts locaux de son ontologie (voir la section IV.1.3 Enrichissement sémantique d'ontologie du chapitre ROMIE).

La Figure 18 illustre l'automate d'états finis d'un agent OA qui a quatre points d'entrée possibles :

1. Chaque agent OA communique avec l'utilisateur pour répondre à ses requêtes. Lorsque cet agent reçoit la demande de l'utilisateur (qui cherche à accéder à des ressources développées par certains concepts de son ontologie), il envoie les ressources (si elles existent) de l'entrepôt local répondant à la requête de l'utilisateur. Dans le cas de l'insatisfaction de l'utilisateur, OA communique avec les autres agents OA pour pouvoir accéder à des ressources distantes. Il extrait le (ou les) concept(s) de la requête et produit un voisinage de celui-ci (ou ceux-ci). Ce voisinage qui contient l'ensemble des concepts obtenus en utilisant les relations structurelles et sémantiques sera envoyé par la suite à tous les agents OA(s) pour chercher des éventuelles relations de *mapping* avec les ontologies distantes gérées par ces agents OA(s).
2. L'agent OA répond aussi aux besoins des autres agents, autrement dit, lorsqu'il reçoit une demande d'un autre agent (OA_2), il coopère avec les agents de *mapping* (MA, HGA, HFA et FVA) pour établir le *mapping* entre les concepts de la requête envoyée par OA_2 et son ontologie locale ; puis une extraction des ressources associées aux concepts mis en correspondance est envoyée à l'agent OA_2.
3. Chaque agent OA informe les autres agents OA, qui sont responsables des ressources distantes proposées à l'utilisateur, du choix de ce dernier.
4. La satisfaction de l'utilisateur pour certaines ressources distantes envoyées par les agents distants OA (à l'aide du *mapping*) joue un rôle primordial dans la phase de validation du *mapping*. C'est pour cette raison qu'OA informe l'agent FVA des rétroactions (feedbacks) de l'utilisateur.

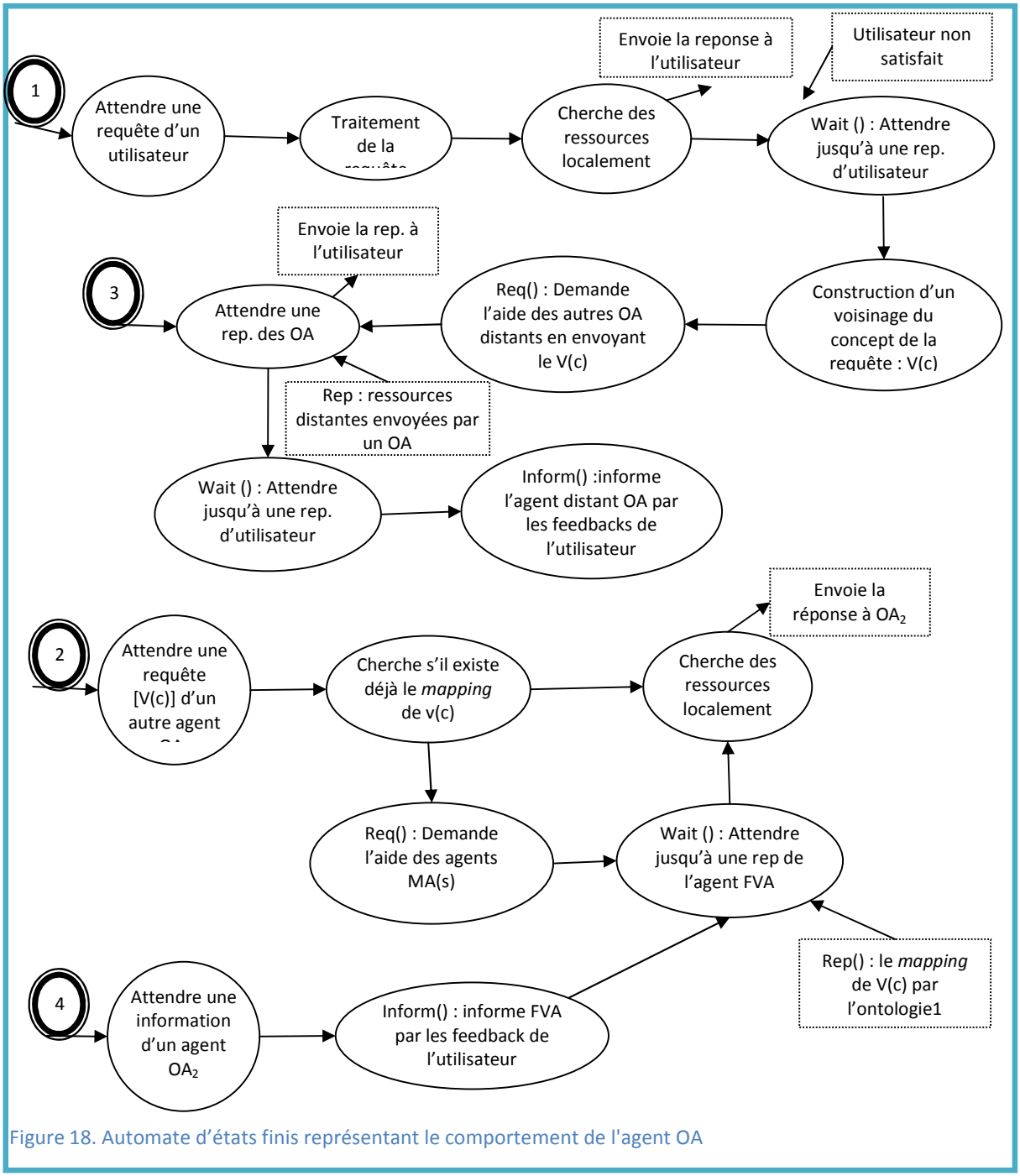


Figure 18. Automate d'états finis représentant le comportement de l'agent OA

Nous décrivons dans les prochaines sections les agents qui implémentent le processus de *mapping* à savoir : MA, HGA, HFA et FVA. Ils sont appelés les agents de *mapping*.

III.3.3.2 Les agents *matchers* : MA(s)

Lorsque l'agent OA reçoit les concepts (appelés concepts externes) envoyés par un autre OA, le groupe d'agents de *matching* MA coopère avec les autres agents de *mapping* (HGA, HFA et FVA) pour identifier le *mapping* entre ces concepts externes et des concepts internes (i.e. les concepts de l'ontologie locale) (Figure 19).

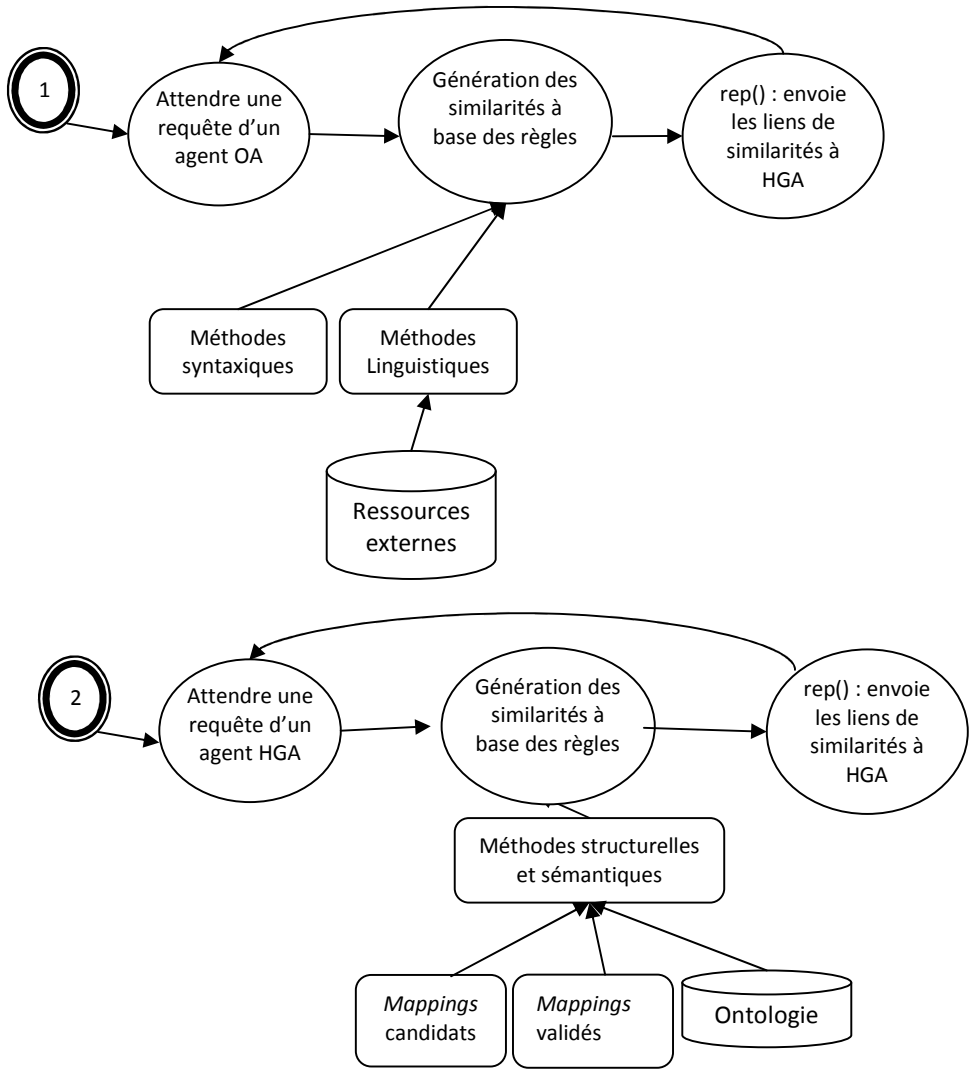


Figure 19. Automate d'états finis représentant le comportement de l'agent MA

Chacun des agents MA compare les différents types d'informations d'ontologie, tels que l'identifiant et le label du concept, les propriétés, les relations, etc. Chaque MA est caractérisé par sa méthode de *matching* pour identifier les similarités entre des couples d'entités. Nous distinguons deux catégories :

1. Des MAs qui répondent à la requête d'un agent OA. Dans cette première catégorie, nous trouvons deux types de *matchers* : (i) les *matchers* linguistiques et (ii) les *matchers* syntaxiques. Chacun d'eux détermine une valeur de similarité entre 0 et 1 pour un couple d'entités. Les *matchers* linguistiques utilisent des informations auxiliaires telles que les dictionnaires et les *matchers* syntaxiques utilisent les méthodes de comparaison de chaînes de caractères. Ces *matchers* sont exécutés en parallèle et de façon indépendante.

2. Des *matchers* structurels et sémantiques, qui sont sollicités par l'agent HGA pour définir et/ou améliorer les *mappings* candidats générés par la combinaison des résultats des *matchers* de la première catégorie. Afin d'aboutir à cet objectif, ils utilisent, d'une part la structure et les relations sémantiques de l'ontologie, et d'autre part les *mappings* générés précédemment entre les deux ontologies à faire correspondre.

L'ensemble des relations de similarité générées par les agents *matchers* est envoyé à HGA pour produire et/ou améliorer des hypothèses de *mapping*.

III.3.3.3 L'agent de génération du *mapping* : HGA

L'agent HGA reçoit les résultats des agents *matchers* MAs de la première catégorie (MAs syntaxiques et MAs linguistiques) sous forme de messages afin de produire un ensemble d'hypothèses de *mapping*. La Figure 20 montre les deux points d'entrée pour exécuter un comportement de l'agent HGA :

- D'une part, il coopère avec les agents MAs pour établir les *mappings* candidats. Puis, il demande l'aide des *matchers* structuraux et sémantiques pour trouver de nouvelles relations de similarité. L'application de n'importe quel *matcher* permet d'augmenter la valeur de similarité de l'ensemble des *mappings* candidats noté Ω. Ce dernier sera envoyé à l'agent HFA pour éliminer les *mappings* non pertinents.

- D'autre part, il communique avec l'agent FVA pour connaître l'évolution et le changement au sein de l'ensemble des *mappings*. Les *mappings* envoyés par FVA sont utilisés pour déduire de nouvelles hypothèses ou pour favoriser quelques hypothèses déjà émises. Finalement, l'ensemble Ω est envoyé à l'agent FHA.

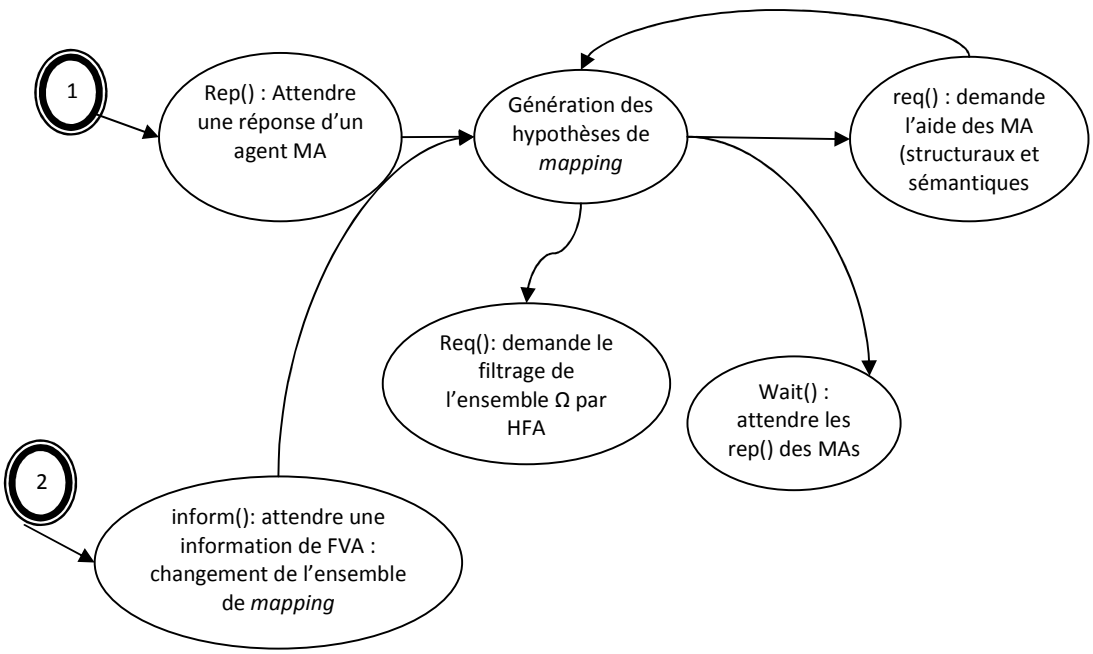


Figure 20. Automate d'états finis représentant le comportement de l'agent HGA

III.3.3.4 L'agent de filtrage de *mapping* : HFA

L'agent HFA filtre l'ensemble Ω des *mappings* candidats en utilisant les différentes méthodes de filtrage (structurelles, sémantiques, etc.) citées dans III.2.3 Le processus de filtrage. La Figure 21 montre aussi que HFA communique avec l'agent FVA pour connaître les changements au sein de l'ensemble des *mappings* validés. Le sous-ensemble des correspondances de Ω obtenu après le processus de filtrage est noté : Ω'.

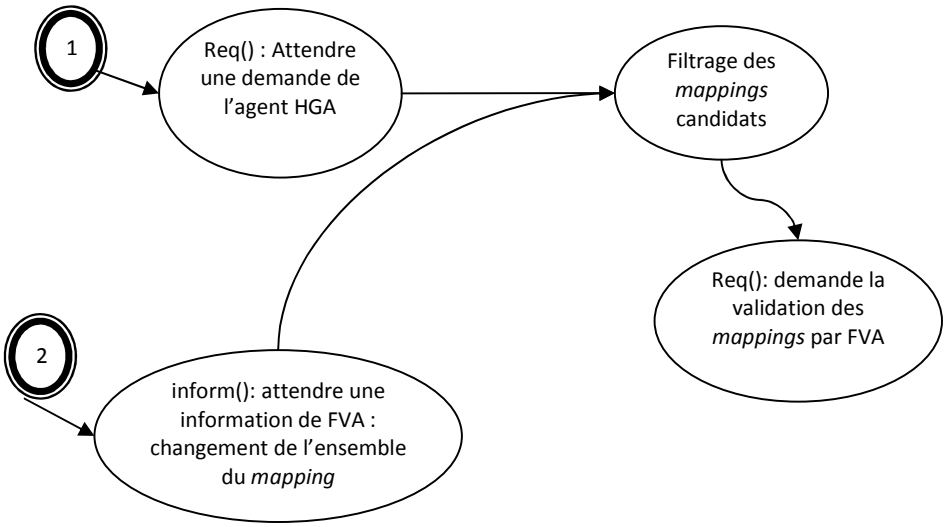


Figure 21. Automate d'états finis représentant le comportement de l'agent HFA

III.3.3.5 L'agent de feedback et de validation : FVA

Une fois la liste des *mappings* candidats fournie, une interaction s'installe entre l'agent FVA et l'utilisateur afin de proposer un ensemble de correspondances entre des couples de concepts filtrés et envoyés par l'agent HFA. En outre, L'utilisateur peut valider (c.-à-d. confirmer) ou invalider (c.-à-d. infirmer) une partie des *mappings* candidats proposés par cet agent. Avant cette phase de validation par l'utilisateur, l'agent FVA peut valider d'une manière automatique une partie des *mappings* candidats qu'il juge forcément corrects en se basant sur les valeurs de similarité et les confiances de ces *mappings* candidats. Pour résumer, nous avons deux types de seuils : un seuil minimal (noté $Seuil_{Min}$) qui joue le rôle d'une bande passante, utilisé par l'agent HFA comme un moyen de filtrage ; et un seuil maximal (noté $Seuil_{Max}$) utilisé par l'agent FVA pour les validations automatiques (Figure 22).

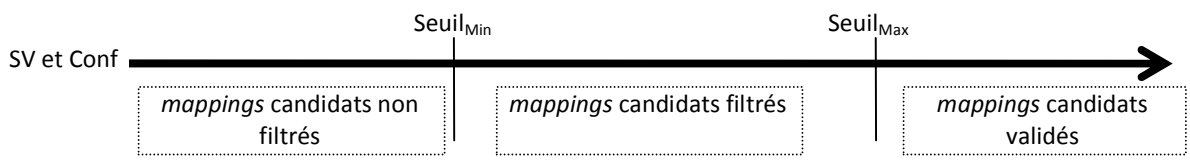


Figure 22. Relation entre les seuils et les *mappings* candidats

Dans le cas où l'utilisateur n'est pas satisfait par l'ensemble des correspondances générées (c.-à-d qu'aucun des *mappings* candidats reçus n'est validé par l'utilisateur), l'agent FVA coopère avec FHA pour

diminuer la valeur du seuil. Quand un *mapping* candidat est validé, il est stocké dans le SVM (*Set of Validated mapping*). Reste à noter que l'administrateur du système peut changer la valeur du seuil s'il juge qu'il y a assez ou peu de *mappings* candidats proposés.

Chaque interaction de validation est employée pour améliorer la qualité du processus d'appariement (par l'agent GHA) et le processus de filtrage (par l'agent FHA). La Figure 23 résume le principe de l'interaction entre l'agent FVA et l'utilisateur d'une part et de l'interaction de FVA avec les autres agents d'autre part. Nous avons alors deux états d'entrée possibles :

1. Le premier état permet à l'agent FVA de coopérer avec HFA pour recevoir la liste des *mappings* candidats générés et filtrés. Il exécute deux comportements d'une manière consécutive : (1) il valide automatiquement un sous-ensemble des *mappings* candidats et il informe les autres agents de ce changement sur l'ensemble SVM ; puis (2) il propose l'ensemble des *mappings* candidats et les *mappings* validés à l'utilisateur pour les éventuelles validations.
2. Le deuxième état consiste à coopérer avec l'utilisateur pour déduire ses interactions. Nous avons deux types de réponses différentes de l'utilisateur qui peut, soit confirmer et valider un ou plusieurs *mappings* candidats, soit invalider une correspondance déjà validée. Dans ces deux cas, l'agent FVA informe les autres agents de ces modifications.

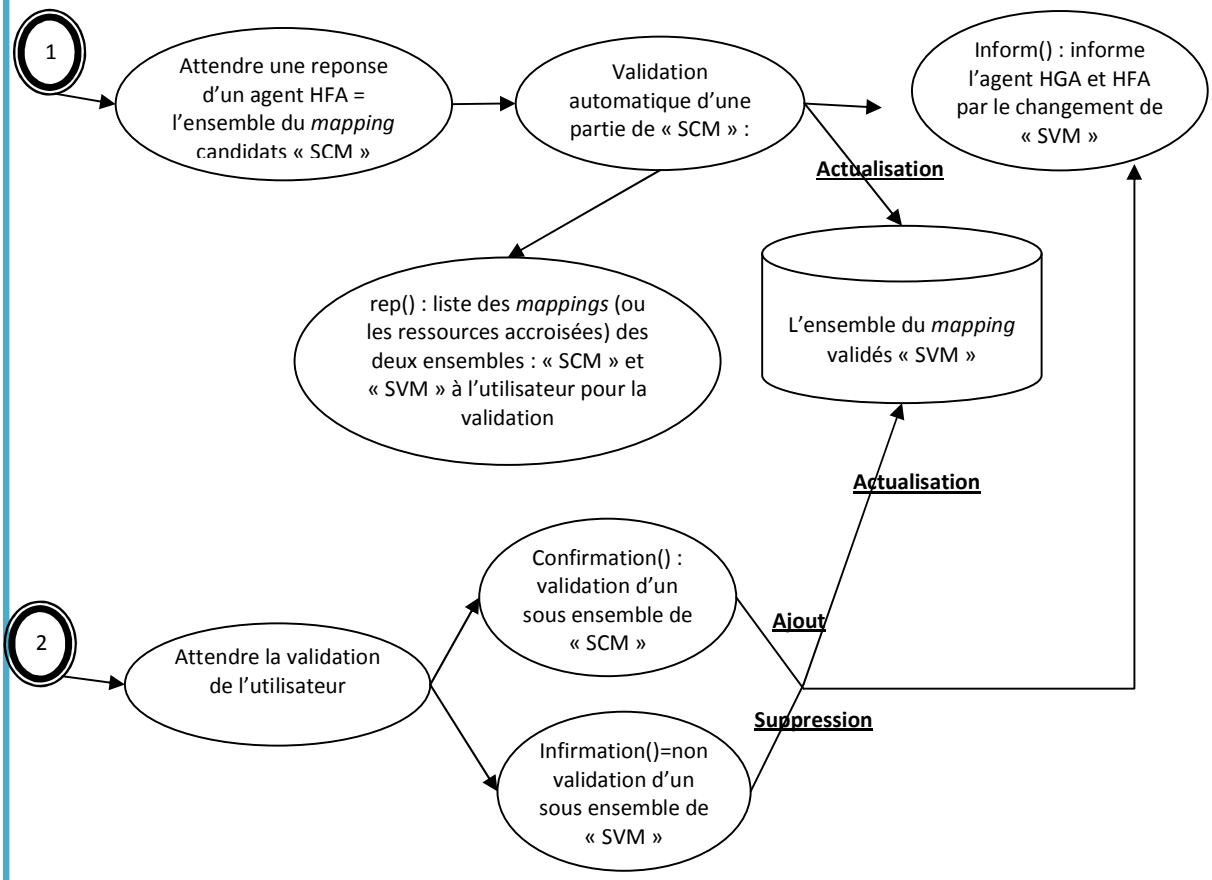


Figure 23. Automate d'états finis représentant le comportement de l'agent FVA

Après avoir présenté le comportement de chaque agent et ses communications avec son entourage (qui peut être un agent ou un utilisateur externe), nous montrons dans la prochaine section le comportement global de coopération et de coordination entre tous les agents et l'environnement extérieur pour réaliser le processus de l'alignement d'ontologies de notre système OMIE.

III.3.4 Coordination des agents

Les protocoles de communication ou d'interaction sont des règles de dialogue dans lesquelles les agents sont confinés. Pour illustrer le comportement général du système OMIE, la Figure 24 illustre un scénario de coopération entre les agents du système pour trouver le *mapping* et répondre à la requête de l'utilisateur.

La requête de l'utilisateur consiste à trouver les ressources associées au concept 'c' de l'ontologie O_1. Nous supposons dans ce diagramme qu'il n y a pas de ressources disponibles localement et que l'objectif est de

créer un *mapping* entre le voisinage du concept 'c' noté V(c) (c.-à-d. l'ensemble des concepts associés au concept c par une relation hiérarchique ou rhétorique –sémantique-) et les concepts de l'ontologie O_2.

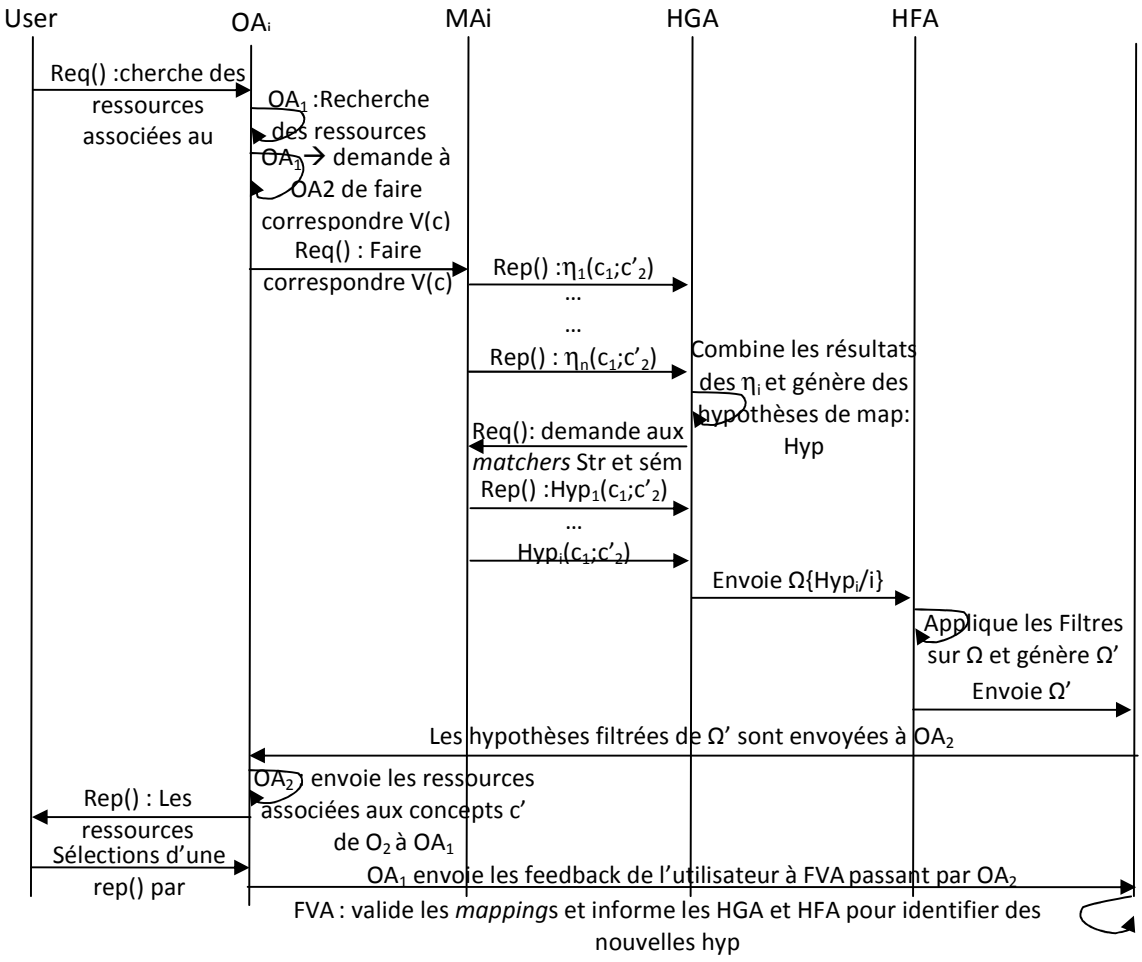


Figure 24. Diagramme UML de la communication des agents du système OMIE

III.4 Conclusion et contributions du système OMIE

Nous avons décrit dans ce chapitre le système OMIE. Il est basé sur une approche à quatre étapes : calcul de similarité, combinaison et génération de *mappings* candidats, filtrage, validation. OMIE est capable d'intégrer

facilement un grand nombre de *matchers* et de les combiner, ce qui facilite la génération des *mappings* candidats et donc augmente le rappel. Ceci ne se fait pas au détriment de la précision, puisque OMIE est également capable de filtrer efficacement les faux *mappings* grâce aux phases de filtrage et de validation qui améliorent fortement les résultats produits par le filtrage à base de seuil habituellement utilisé. En résumé, OMIE permet de remplir les objectifs initialement fixés :

- **Objectif 2 : Le mode partiel et total de *mapping*** : La couverture des ontologies est très rarement la même et n'est pas complète. D'où la nécessité d'offrir aux utilisateurs un système de *mapping* capable d'identifier les correspondances de ***tout*** ou ***partie*** des éléments de l'ontologie. C'est-à-dire l'intégration ou le ***mapping partiel*.**
- **Objectif 4** : Dans le cadre de l'automatisation du processus de *mapping* nous souhaitons aider l'utilisateur en réduisant le nombre de faux résultats, en faisant mieux que le filtrage à base de seuil qui est la méthode adoptée par la plupart des travaux existants. Pour cela, nous exploitons les ***liens hiérarchiques et sémantiques*** existants entre les concepts de chaque ontologie dans le ***processus de filtrage*** pour détecter certaines anomalies et contradictions parmi les résultats de *mapping* obtenus.
- **Objectif 5** : La phase de validation et d'interaction avec l'utilisateur est une phase clé dans le processus de mapping, mais elle est très compliquée. Dans la plupart des systèmes existants, les interactions avec l'utilisateur concernent directement la validation des liens de correspondance entre les concepts des ontologies, ce qui rend la phase de validation des mappings un processus lourd et moins sûr. Partant de ce constat, il faudrait trouver un autre moyen de valider les résultats qui soit plus automatique.

L'implantation d'OMIE sous forme d'un système multi-agents est naturelle et permet notamment de bénéficier des avantages de ces systèmes en termes d'extensibilité. La coopération entre les différentes étapes est également facilitée.

L'un des points fort du système OMIE est sa capacité à exploiter toutes les relations ontologiques notamment les relations sémantiques (c.-à-d. les liens entre les concepts autres que les liens hiérarchiques) dans le processus de *mapping*. Néanmoins, de nombreuses ontologies ne disposent pas d'assez de relations sémantiques. Dans le chapitre suivant, nous montrons une extension du système OMIE qui, par l'analyse des ressources et des instances annotées par les ontologies, raffine et améliore les résultats de correspondance du processus de *mapping* : il s'agit du système ROMIE.

Chapitre IV : Système ROMIE (OMIE à base de ressources)

Le processus de *matching* est basé sur la mesure de similarité entre les concepts des différentes ontologies. Cette étape importante adoptée par tous les algorithmes et les systèmes existants de *mapping* d'ontologies consiste à combiner plusieurs types de méthodes de comparaison, à savoir les méthodes syntaxiques, lexicales, linguistiques et autres. Ce genre de comparaison permet de mesurer la similarité entre les concepts du point de vue terminologique. Néanmoins, ces méthodes ne sont pas suffisantes pour résoudre tous les cas d'hétérogénéité, car elles comparent les entités indépendamment de leurs contextes d'utilisation.

C'est pour cette raison que l'on trouve d'autres méthodes de *matching* capables d'identifier les correspondances dans un contexte donné, en se basant sur la structure ou la hiérarchie des ontologies. Ce sont les méthodes dites structurelles, que l'on retrouve dans un grand nombre de systèmes existants. Dans la plupart du temps, l'information sur le voisinage structurel du concept porte sur ses pères et ses fils.

L'un des objectifs de notre approche est l'exploitation des relations ontologiques autres que les relations structurelles. On les qualifie de relations sémantiques, par exemple « located-in », « contraste », etc. Ces relations sont très importantes et permettent d'ajouter un niveau sémantique à la similarité. Malheureusement, les ontologies existantes possèdent rarement ce type de relations. Par conséquent, une étape importante de notre système est l'enrichissement de chacune des ontologies à faire correspondre par des relations sémantiques entre leurs concepts. Cette phase a lieu avant même l'exécution du processus de *mapping*. Les relations sémantiques entre les concepts de l'ontologie sont produites en utilisant les informations sur les ressources (ou les instances) reliées à ces concepts (Figure 25). L'extension du système OMIE par ce nouvel aspect de *matching* est nommée ROMIE (c.-à-d. OMIE à base des ressources ou d'instances).

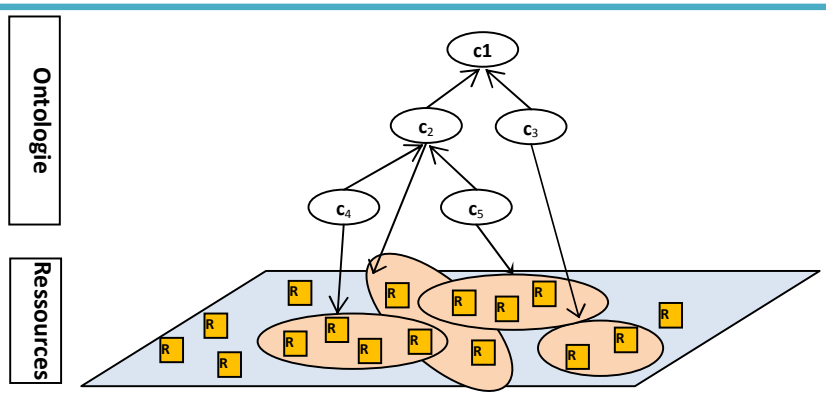


Figure 25. Relations entre les concepts d'ontologie et les ressources

IV.1 Le système ROMIE

IV.1.1 Principes de ROMIE

Le système ROMIE est une extension du système OMIE où l'on tient compte des instances (ressources) pour améliorer le processus de *mapping*. Contrairement aux approches de *mapping* qui se basent sur les termes des entités ou sur la structure de l'ontologie, notre approche à base d'instances se fonde principalement sur les informations réelles décrites par les ontologies. Le système ROMIE traite l'information contenue dans les instances de chaque élément (ou chaque concept) de l'ontologie. Il est capable d'identifier les liens de similarité entre les concepts des ontologies à faire correspondre en analysant leurs instances associées.

La Figure 26 décrit ROMIE comme extension d'OMIE. La principale extension porte sur l'étape préalable d'enrichissement de (des) l'ontologie(s) locale(s). Cela permet ensuite au processus de génération de mappings de produire plus de candidats et il faut également modifier les étapes de filtrage et de validation pour qu'elles soient capables de fonctionner efficacement.

L'architecture agent de ROMIE est semblable à celle d'OMIE, puisque l'étape d'enrichissement d'ontologies se fait en pré-traitement. Seuls les agents correspondant au filtrage et à la validation sont étendus pour pouvoir prendre en compte les relations générées lors de l'étape d'enrichissement.

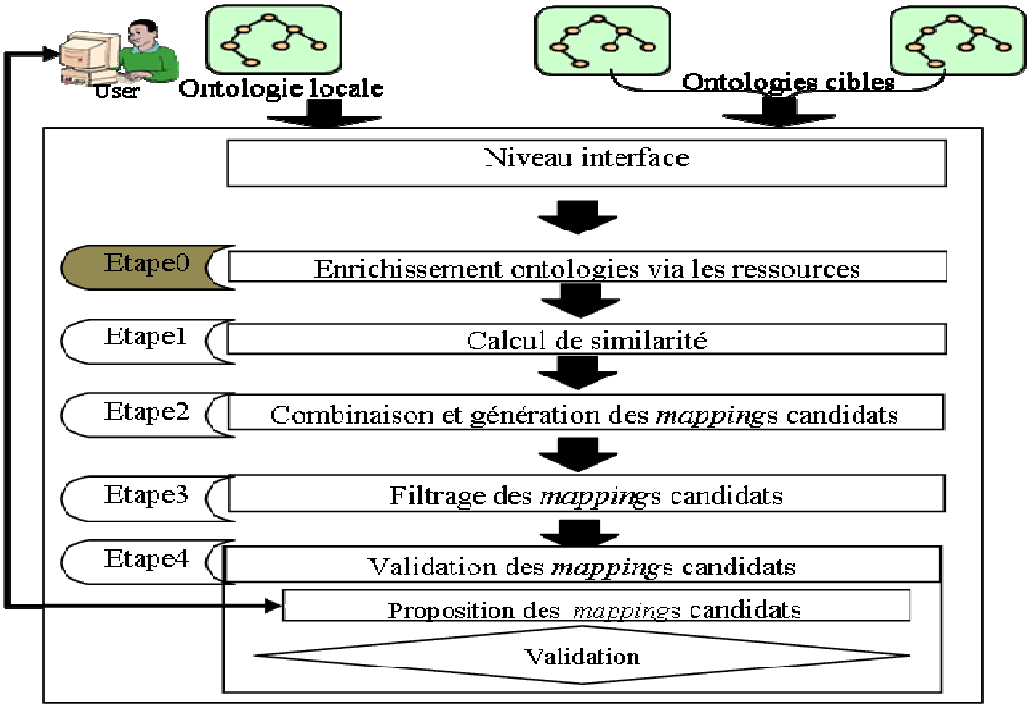


Figure 26. ROMIE comme extension d'OMIE

IV.1.2 *Mapping* d'ontologies à base de ressources (ou d'instances)

Nous distinguons deux types de méthodes de calcul de similarité entre les ontologies en utilisant les instances. Le premier type est appliqué dans le cas où un même entrepôt d'instances est partagé par les deux ontologies à faire correspondre (Figure 27) ; le deuxième est appliqué dans le cas où chaque ontologie dispose de son propre entrepôt d'instances (Figure 28). Nous décrivons dans ce qui suit l'exploitation des instances avec les méthodes de *matching* pour les deux cas suivants :

(i) des instances communes aux deux ontologies à faire correspondre et

(ii) des instances associées aux ontologies disjointes.

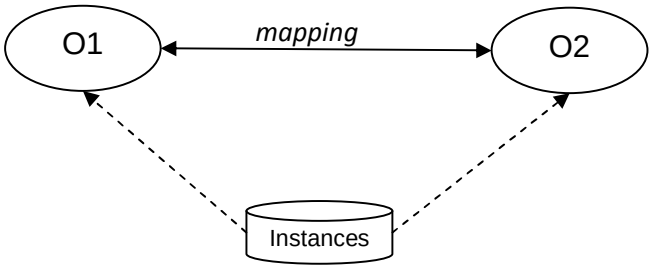


Figure 27. Un entrepôt d'instances (ou ressources) annoté par les termes des deux ontologies

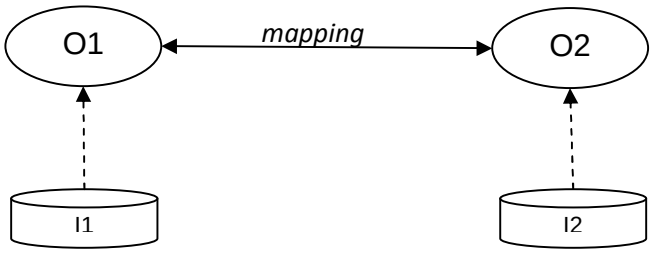


Figure 28. Deux entrepôts d'instances (ressources) annoté par deux ontologies différentes

Nous développons dans les prochaines sections le principe et l'intérêt de la comparaison à base d'instances.

IV.1.2.1Comparaison des instances communes

Nous décrivons ici le cadre d'utilisation des instances dans la procédure de *mapping*. Soient deux ontologies S (ontologie source) et T (ontologie cible) et un entrepôt d'instances EI.,Chaque instance d'EI est annotée par les termes de l'ontologie S et/ou de T. Nous voyons le *mapping* entre ces deux ontologies S et T comme un triplet (S, T, R), où R est une relation entre les concepts de S et les concepts de T. La relation R appartient à cet ensemble {≡, ⊂, ⊥}, respectivement : l'équivalence, l'inclusion et la différence. Dans d'autres contextes d'application, tel le contexte biomédical, nous pouvons définir d'autres types de relations, par exemple la relation « *part-of* ».

Le *mapping* d'ontologies à base d'instances consiste à créer des relations sémantiques entre les concepts de deux ontologies qui partagent un nombre important d'instances entre elles. C'est un aspect intuitif, qui se base sur le fait que la sémantique des concepts et de leurs relations est définie par l'intermédiaire de l'ensemble de leurs instances. L'idée de ce *matcher* à base d'instances est que plus le nombre d'instances communes entre deux concepts est élevé, plus ces deux derniers sont connexes ou équivalents.

Nous utilisons différents mécanismes pour calculer le rapport des instances communes entre deux concepts. L'une des méthodes la plus simple est de comparer l'intersection de l'ensemble des instances IA annotées par le concept A de l'ontologie S avec l'ensemble des instances IB annotées par le concept B de l'ontologie T. Nous considérons alors les cas suivants :

- Les deux concepts A et B sont similaires (A ≡ B) si IA ∩ IB =IA=IB.
- Le concept A est plus général que le concept B (c.-à-d, A ⊃ B) si IA ∩ IB =IB et IA≠∅.
- Les deux concepts A et B sont considérés comme dissimilaires (A ⊥ B) dans les autres cas, par exemple si le concept A partage une partie ou aucune de ses instances avec le concept B ou.

Pour raffiner la méthode de calcul du rapport entre les instances des concepts, on utilise la mesure de Jaccard appelée aussi la distance de Jaccard. Elle permet de calculer la distance entre deux ensembles en utilisant des probabilités. Formellement soit IA et IB deux ensembles, la distance de Jaccard entre IA et IB, notée σ (IA,IB) est :

- σ (IA, IB) = $\frac{P(IA \cap IB)}{P(IA \cup IB)}$

avec P(X) la probabilité qu'un élément aléatoire soit dans l'ensemble X. Cette mesure est normalisée et elle est égale à : (i) 0 si IA ∩ IB = ∅ et (ii) 1 si IA ∩ IB = IA=IB.

IV.1.2.2 Comparaison des instances séparées

Contrairement à la section précédente, les deux ontologies S et T annotent deux entrepôts d'instances distincts. La similarité entre les concepts des deux ontologies ne peut pas être calculée à l'aide des instances communes. Dans ce cas, il est préférable d'évaluer la distance entre ces concepts. Afin d'identifier des relations sémantiques autres que la correspondance naturelle entre les concepts de l'ontologie, nous employons les liens sémantiques entre les instances (voir la Figure 29). Cette méthode de création de relations sémantiques entre les concepts d'une même ontologie est appelée : l'enrichissement sémantique d'ontologie.

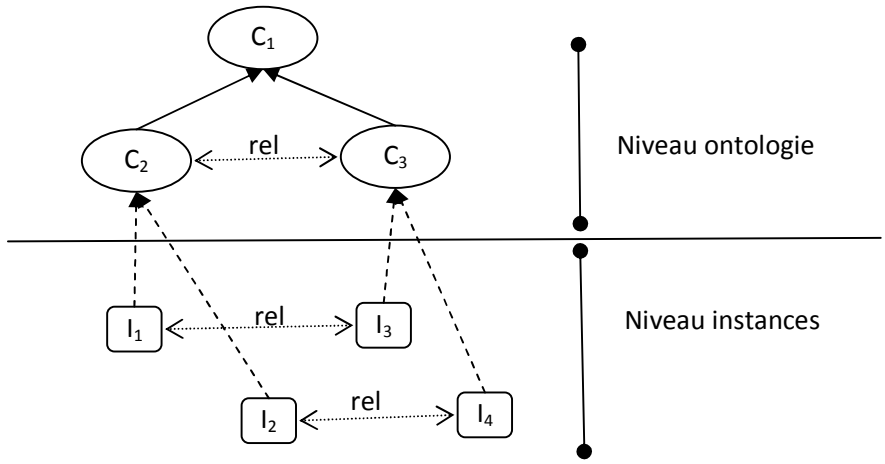


Figure 29. Propagation des relations sémantiques entre instances vers les concepts

IV.1.3 Enrichissement sémantique d'ontologie

L'une des contributions importantes du système ROMIE est la phase d'enrichissement d'ontologie. Cette étape joue un rôle crucial pour améliorer les résultats du *mapping* tout en donnant plus de sémantique aux termes des ontologies. Comme nous l'avons déjà présenté dans le chapitre précédent, nous employons une approche hybride pour le *mapping* d'ontologies qui associe à la fois des approches linguistiques, syntaxiques, structurelles et sémantiques.

Le *matcher* sémantique consiste à exploiter les différentes caractéristiques sémantiques de l'ontologie. Le problème est que le nombre et/ou la qualité de ces relations sémantiques existantes entre les concepts au sein d'une ontologie est/sont en général très faible(s). L'objectif principal de notre approche à base d'instances, dans cette phase d'enrichissement sémantique d'ontologie, est d'exploiter les informations sur les instances annotées par les concepts d'ontologie. Ces informations sont analysées pour inférer de nouvelles relations sémantiques entre les concepts d'ontologie. Cela a pour effet d'enrichir la sémantique de l'ontologie et par conséquent, d'améliorer les processus de *matching*.

La première étape dans ce processus est d'utiliser les relations entre les instances quand elles existent et/ou d'analyser leurs propriétés, afin de produire des relations entre elles. La deuxième étape propage ces relations au niveau ontologique entre les concepts. Par exemple, sur la Figure 29, nous propageons la relation qui relie I_1 avec I_3 aux deux concepts C_2 et C_3.

IV.1.4 Le processus de *mapping* à base d'instances

Dans cette section, nous étudions l'impact de l'étude des instances et des relations sémantiques engendrées sur le processus de *mapping*. Tout d'abord, nous définissons la notion d'ontologie morphisme qui nous aidera à définir et comprendre les différentes règles de *matching* et de filtrage. Ensuite, nous présentons les *matchers* et les filtres utilisés par le système ROMIE.

IV.1.4.1 Morphisme d'ontologies

Le principe du morphisme d'ontologies est de considérer qu'étant données deux ontologies, pour chacune des relations entre deux concepts d'une ontologie (qu'elles soient hiérarchiques ou sémantiques), il existe une relation équivalente entre les concepts « images » de l'autre ontologie, c.-à-d. les concepts reliés par la relation de *mapping*. Soient deux ontologies $O(c_1,c_2,\ldots,c_n)$ et $O'(c'_1,c'_2,\ldots,c'_n)$; si les deux concepts c_1 et c_2 sont liés par une relation sémantique telle que la relation « part-of », et s'il existe un *mapping* entre c_1 et c'_1 et entre c_2 et c'_2 alors c_1' et c2' sont reliés par la même relation sémantique. Ceci mène aux définitions suivantes :

Nous définissons une ontologie O par un 4-uplet O=(C, R, <, σ) tel que : C et R représentent respectivement l'ensemble des concepts et des relations de l'ontologie O ; la relation d'ordre partiel « < » est une relation hiérarchique entre les concepts de C ; La fonction $\sigma : R \rightarrow C\times C$ est une signature qui permet d'associer une relation sémantique de R à un couple de concepts de C. Nous définissons un morphisme d'ontologie comme ceci :

Etant données deux ontologies O=(C, R, <, σ) et O'=(C', R', <', σ'), un morphisme entre deux ontologies est défini par le couple de fonctions (F,G), tel que : F : C → C' est une fonction qui relie les éléments de C avec les éléments de C' et G : R → R' est une fonction qui relie les éléments de R avec les éléments de R'.

Nous décrivons par la suite les propriétés de ces deux fonctions. Soient deux concepts a et b de C et r une relation de R. Notons que F(x)=x' signifie que le concept x' est le concept correspondant (ou encore, le concept image) au concept x par la relation de *mapping* et G(r)=r' signifie que la relation r' est la relation équivalente à la relation r, en particulier r=r'. En se servant de la définition du morphisme d'ontologies, nous pouvons déduire les règles suivantes illustrées par la Figure 30 :

1. Règle1: Si $c_2 < c_1$ alors $F(c_2) < F(c_1)$, c'est-à-dire que si c_2 est un fils de c_1 dans l'ontologie O, alors $F(c_2)=c'_2$ est le fils de $F(c_1)=c'_1$ tel que c'_1 et c'_2 sont les concepts correspondant aux concepts c_1 et c_2 respectivement. Cette règle sera exploitée par le *matcher* et le filtre structurel.
2. Règle2 : Si $\sigma(r) = (c_2, c_3)$ alors $\sigma'(G(r))=(F(c_2),F(c_3))=(c'_2, c'_3)$, c'est-à-dire que l'existence d'une relation r entre les deux concepts c_2 et c_3 implique qu'il existe une relation r' équivalente à r (cas particulier,

r'= r) entre les deux concepts images c'_2 et c'_3. Cette règle sera exploitée par le *matcher* et le filtre sémantique.

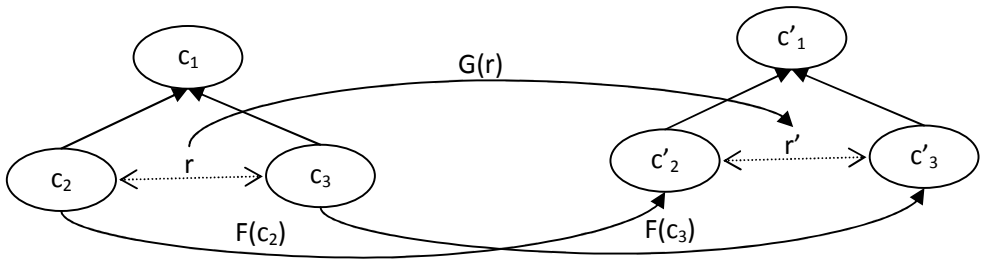


Figure 30. Morphisme d'ontologie

IV.1.4.2 *Matchers* à base de relations sémantiques générées

Les relations sémantiques produites entre les concepts dans l'étape d'enrichissement d'ontologie sont exploitées dans le processus de *mapping* par des *matchers* spécifiques, appelés aussi *matchers* sémantiques. Ces derniers n'exigent pas forcément que les concepts des deux ontologies partagent le même ensemble de ressources. Le principe de ces *matchers* est défini comme suit : deux concepts peuvent être similaires si leurs voisinages sémantiques le sont, sachant que le voisinage sémantique d'un concept est l'ensemble des concepts avec lesquels il est lié par une relation sémantique dans une même ontologie.

Grâce au morphisme d'ontologies et après l'application des *matchers* linguistiques, syntaxiques et structurels qui permettent d'obtenir quelques *mappings*, les *matchers* sémantiques produisent de nouvelles relations de *mapping*, afin d'extraire le maximum de relations de correspondance existantes entre les deux ontologies à faire correspondre, tout en utilisant les relations sémantiques produites dans l'étape d'enrichissement.

Le *matcher* sémantique (c.-à-d. le *matcher* à base de relations sémantiques d'ontologies existantes ou générées) fournit des hypothèses où les *mappings* candidats sont représentés par les 5-uplets sous la forme <rel,c,d,$Conf_{Sem'Matcher'}$, $SV_{Sem'Matcher'}$> avec :

- Rel : type de relation produit entre les deux concepts c et d ; il peut être soit une relation d'équivalence (≡), d'inclusion (⊆) ou de recouvrement (⊇).
- $Conf_{Sem'Matcher'}$: niveau de confiance associé au *matcher* sémantique ; la valeur de ce coefficient change suivant la granularité ou l'influence de ce *matcher* sur le processus de *mapping*.

- $SV_{Sem'Matcher'}$: valeur de similarité générée par le *matcher* sémantique entre les deux concepts c et d ; c'est une valeur numérique comprise entre 0 et 1.

Pour chaque concept nous calculons la cardinalité de son voisinage sémantique. Les deux figures (Figure 31 et Figure 32) montrent deux exemples possibles de définition du voisinage sémantique suivant les propriétés de la relation sémantique qui relie les concepts de l'ontologie :

1. Cas 1 : La relation sémantique qui définit ce voisinage est une relation non-symétrique (Figure 31). Nous distinguons dans ce cas deux sous-ensembles de voisinage différents : un ensemble de fils sémantiques et un autre de pères sémantiques. Nous associons à chaque ensemble un indice de cardinalité : l'indice « nbrOfSemChild » qui calcule la cardinalité de l'ensemble sémantique des fils et l'indice « nbrOfSemFather » qui calcule la cardinalité de l'ensemble sémantique des pères.
2. Cas 2 : La relation sémantique qui définit ce voisinage est une relation symétrique (Figure 32). Dans ce cas un seul ensemble de voisinage regroupe tous les concepts reliés.

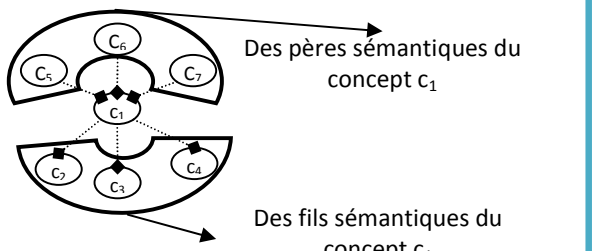


Figure 31. Voisinage sémantique généré par une relation non symétrique

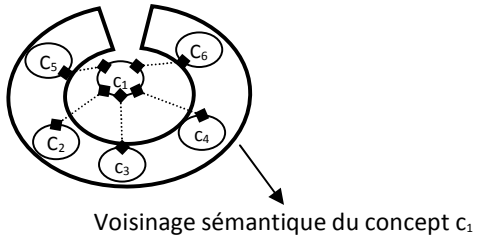


Figure 32. Voisinage sémantique généré par une relation symétrique

Nous détaillons à présent le principe des *matchers* à base de relations sémantiques. Celles-ci peuvent être présentes dans l'ontologie ou générées durant la phase d'enrichissement d'ontologie. Les deux sous sections suivantes illustrent le mécanisme de *matching* pour le calcul de la similarité entre les concepts à base de relations sémantiques.

a) Matcher sémantique à base d'une relation non-symétrique

La relation sémantique de type non-symétrique est semblable à la relation structurale hiérarchique, toutes les deux définissant pour chaque concept un ensemble de fils (ou sous-concepts) et un ensemble de pères (ou super-concept). Le *mapping* entre deux concepts implique l'existence d'un sous-ensemble de leur voisinage qui soit semblable. Selon ce type de voisinage, nous définissons deux types de *matchers* : (i) le *matcher*

sémantique basé sur les fils et qui repose sur le voisinage sémantique des fils d'une relation non-symétrique ; (ii) le *matcher* sémantique basé sur les pères et qui considère le voisinage sémantique père d'une relation non-symétrique

La Figure 33 et l'algorithme 6 détaillent le mode de fonctionnement du *matcher* sémantique à base d'une relation non symétrique. Nous notons rel(σn) la relation sémantique définie entre deux concepts locaux avec comme paramètre la mesure de Jaccard.

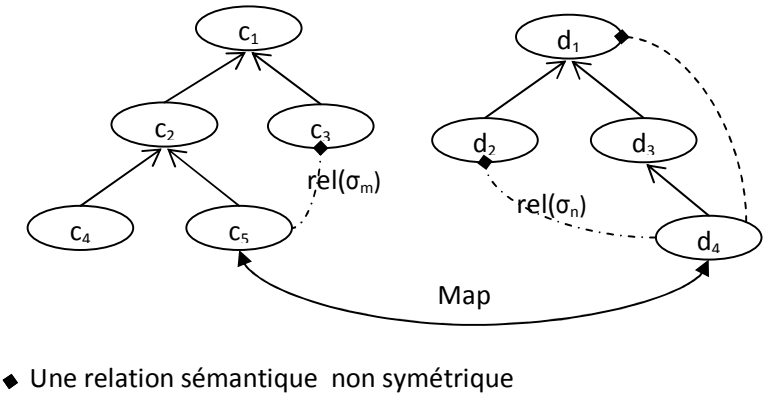


Figure 33. *Matcher* sémantique à base d'une relation non symétrique

L'algorithme 6 montre les démarches d'appariement des concepts des deux ontologies en utilisant une relation sémantique de caractéristique non symétrique. Ce *matcher* sémantique utilise les relations de *mappings* validées (par exemple, entre c_5 et $d_4$dans le schéma) pour produire de nouvelles relations de correspondance (c.-à-d. entre c_3 et d_2).

Etant données deux ontologies O et O' et les variables suivantes:
rel: relation sémantique non symétrique ;
rel(σ_m): relation sémantique entre c_5 et c_3 dans l'ontologie O ;
rel(σ_n): relation sémantique entre d_4 et d_2 dans l'ontologie O' ;
NdrOfSemChild(c_i,rel(σ_m)) : retourne le nombre de concepts fils reliés au concept c_j par la relation sémantique rel. E.g., NdrOfSemChild (d_4,rel(σ_n))=2 (c'est-à-dire d_2 et d_1)
SI ∃ Map<≡,c_5,d_4,$Conf_1$,SV_1>
ALORS
On définit les constantes suivantes par :
$\sigma = Min(\sigma_n, \sigma_m)$;

$MinOfSemChild(c_i, d_j)=Min(NdrOfSemChild(c_i, rel_k(\sigma_n)), NdrOfSemChild(d_j, rel_k(\sigma_m))$.

SI $\exists\ Hp<\equiv, c_3, d_2, Conf_2, SV_2>$

ALORS

Changer les paramètres de Hp par:

$SV_2= SV_2 + \sigma*(SV_1/ MinOfSemChild(c_5, d_4))$;

$Conf_2=Conf2+ \sigma*\ Conf_{Sem'Matcher'}$;

SINON

générer $Hp<\equiv, c_3, d_2, Conf_2, SV_2>$ avec

$SV_2= SV_2 + \sigma*(SV_1/ MinOfSemChild(c_5, d_4))$;

$Conf_2=Conf2+ \sigma*Conf_{Sem'Matcher'}$;

$Conf_2=Conf2+ \sigma*Conf_{Sem'Matcher'}$;

Algorithme 6. Algorithme d'un *matcher* sémantique à base d'une relation non symétrique

Une autre façon d'améliorer le processus de *matching* est de considérer non seulement les relations sémantiques directes mais également les relations indirectes. La caractéristique transitive de la relation sémantique nous permettra de définir des relations indirectes. Typiquement, si nous avons une relation sémantique entre c_1 et c_2 d'une part et entre c_2 et c_3 d'autre part, alors nous déduisons une relation sémantique indirecte entre c_1 et c_3.

b) Matcher sémantique à base d'une relation symétrique

Contrairement à une relation non-symétrique, une relation symétrique dispose d'un seul voisinage qui regroupe tous les concepts liés entre eux par cette relation. La Figure 34 et l'algorithme 7 détaillent le mode de fonctionnement du *matcher* sémantique à base d'une relation sémantique symétrique.

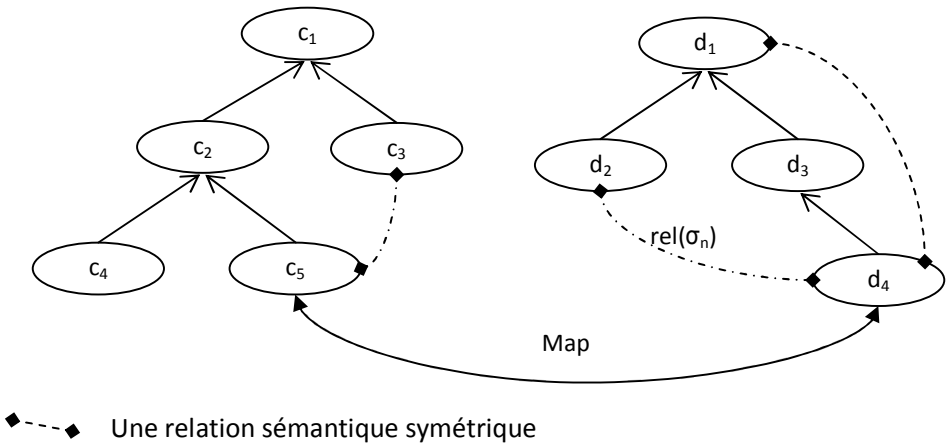


Figure 34. Matcher sémantique à base d'une relation symétrique

L'algorithme 7 présente le *matcher* sémantique à base d'une relation symétrique (voir l'exemple de la Figure 34). Ce *matcher* prend en considération entre autres la taille du voisinage sémantique engendré par la relation sémantique (c.-à-d, NdrOfSemConcept(c_i,rel(σ_m))) pour calculer une similarité proportionnelle avec le nombre de concepts de ce voisinage.

Etant données les variables suivantes :

rel : relation sémantique symétrique ;

rel(σ_m) : relation sémantique entre c_5 et c_3 dans l'ontologie O ;

rel(σ_n) : relation sémantique entre d_4 et d_2 dans l'ontologie O' ;

NdrOfSemConcept(c_i,rel(σ_m)) *retourne le nombre de concepts reliés au concept c_j par la relation* rel(σ_m) (par exemple, NdrOfSemConcept(d_4,rel_k(σ_n))=2)

SI ∃ Map<≡, c_5, d_4, $Conf_1$, SV_1>
Alors
  σ = Min(σ_n, σ_m);
  MinOfSemConcept(c_i, d_j)=Min(MinOfSemConcept(c_i, rel_k(σ_n), MinOfSemConcept(d_j, rel_k(σ_m))
  SI ∃ Hp<≡, c_3, d_2, $Conf_2$, SV_2>
  Alors
    SV_2= SV_2 + σ*(SV_1/ MinOfSemConcept(c_5, d_4));
      $Conf_2$=Conf2+ σ* $Conf_{SemMatcher}$;

Sinon *générer* Hp<≡, c_3, d_2, $Conf_2$, SV_2> avec

$SV_2 = SV_2 + \sigma * (SV_1 / MinOfSemConcept(c_5, d_4))$;

$Conf_2 = Conf2 + \sigma * Conf_{SemMatcher}$;

Algorithme 7. Algorithme du *matcher* sémantique à base d'une relation symétrique.

IV.1.4.3 Filtres à base de relations sémantiques générées

Une des caractéristiques capitales de ROMIE est sa capacité à réduire considérablement le nombre de faux *mappings* candidats, afin d'aider l'utilisateur pendant le processus de validation de *mapping*. Nous avons développé plusieurs méthodes pour filtrer les hypothèses de *mapping* produites. Ces méthodes sont basées sur les relations structurelles et sémantiques des ontologies. Comme pour les *matchers* sémantiques, nous exploitons les caractéristiques des relations sémantiques pour éliminer les correspondances insatisfaisantes automatiquement. L'idée principale est de respecter les règles de morphisme d'ontologie. Figure 35 et l'Algorithme 8 montrent le principe de filtrage lorsque nous avons deux hypothèses de *mapping* contradictoires (c.-à-d. les hypothèses croisées) : dans ce cas, nous gardons uniquement l'hypothèse la plus appropriée (c.-à-d. avec la plus grande valeur de similarité). Un autre type de filtrage possible est celui du filtre qui vérifie la compatibilité entre les hypothèses de *mapping* générées et les *mappings* déjà validés.

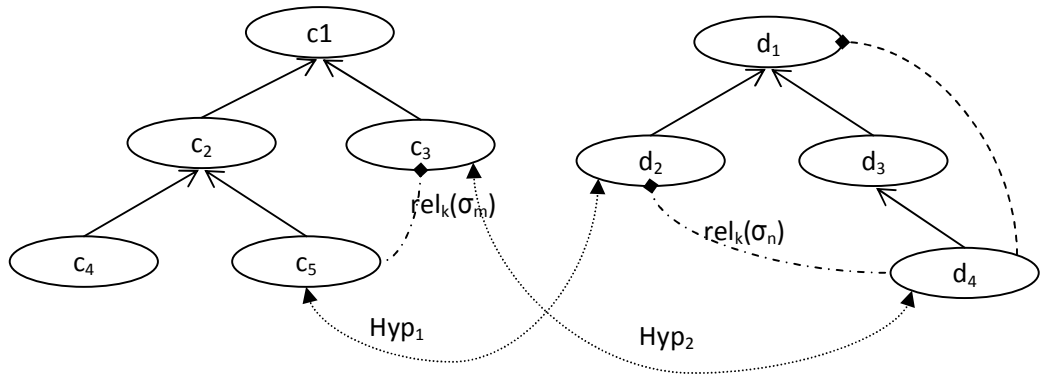


Figure 35. Filtrage avec les relations croisées

Hyp_1: hypothèse de *mapping* (*mapping* candidat).

rel_k: relation sémantique reliant le concept c_5 avec c_3 dans l'ontologie O_1 et d_4 avec d_2 et d_1 dans l'ontologie O_2.

Si

rel_k est une relation non symétrique et

```
      ∃ Hp1<⊆,c5,d2,Conf1,SV1> et
      ∃ Hp2<⊆,c3,d4,Conf2,SV2>
Alors
      Si SV1>SV2
      Alors éliminer Hp2
      Sinon éliminer Hp1
```

Algorithme 8. Algorithme d'un filtre à base d'une relation sémantique

Nous expliquons dans les sections qui suivent la méthode par laquelle nous avons exploité ces différents principes avec deux applications distinctes : une application éducative et une autre biomédicale.

IV.2 ROMIE appliqué au contexte éducatif

IV.2.1 Introduction

L'enrichissement automatique des ontologies dépend du domaine d'application car il faut étudier finement les relations entre instances et ontologies ainsi que leurs propriétés pour définir l'enrichissement. Ceci est très dépendant du domaine d'application et doit donc être refait à chaque nouvelle application. Cependant, un certain nombre de relations génériques peuvent être mises en évidence et réutilisées d'un domaine à l'autre. De manière plus générale, on peut capitaliser sur l'étude des relations et donc à terme ne plus avoir à en analyser de nouvelles.

ROMIE éducatif propose une solution dynamique de *mapping* d'ontologies, qui permet à chaque apprenant de construire sa requête avec les termes de son ontologie locale et d'accéder à l'ensemble des ressources disponibles sur les entrepôts pédagogiques distants. Dans notre approche, les ressources locales sont comparées afin d'enrichir les relations entre les concepts locaux, donnant plus de sémantique à nos concepts. Celle-ci est exploitée par la suite pour améliorer les résultats de génération de *mappings*.

Nous présentons dans les prochaines sous-sections, les étapes d'enrichissement sémantique des ontologies et par la suite l'exploitation des liens générés dans le processus de *matching* et le processus de filtrage.

IV.2.2 Enrichissement sémantique d'ontologie

Dans cette section, nous focalisons sur l'aspect utilisation des ressources pour l'enrichissement et le *mapping* sémantique que nous avons développé dans les sections précédentes. Pour produire de nouvelles relations entre les concepts locaux, nous commençons par la définition des relations entre les ressources. Nous

présentons les deux étapes d'enrichissement sémantique nécessaires à la procédure de *mapping* : génération de relations inter-ressources et génération de relations inter-concepts.

IV.2.2.1 Génération de relations inter-ressources

Rappelons que les ressources éducatives dans le modèle SIMBAD sont décrites par un ensemble de métadonnées. Nous distinguons entre deux types de métadonnées : le premier décrit des caractéristiques générales de la ressource (par exemple, auteur, titre, langue, médias) utilisant le standard LOM et le second décrit sa sémantique. Cette sémantique est structurée en trois parties : les ***pré-requis*** qui sont les entrées de la ressource, c'est-à-dire les connaissances exigées pour l'exécution de cette ressource, tandis que le ***contenu*** et la ***fonction d'acquisition*** sont les sorties de la ressource, c'est-à-dire ce qui est fourni par cette dernière (Figure 36) :

- Les ***pré-requis*** d'une ressource sont un ensemble de triplets (concept, rôle, niveau) où le concept est pris du modèle du domaine (c.-à-d. l'ontologie) et le rôle indique par quels aspects du concept cette ressource est concernée (par exemple, « introduction », « définition », « description », « application »).
- Le ***contenu*** d'une ressource est décrit avec un ensemble de couples (concept, rôle) .
- La ***fonction d'acquisition*** indique quel triplet (concept, rôle, niveau) sera ajouté au modèle d'apprenant si un état de validation est satisfaisant.

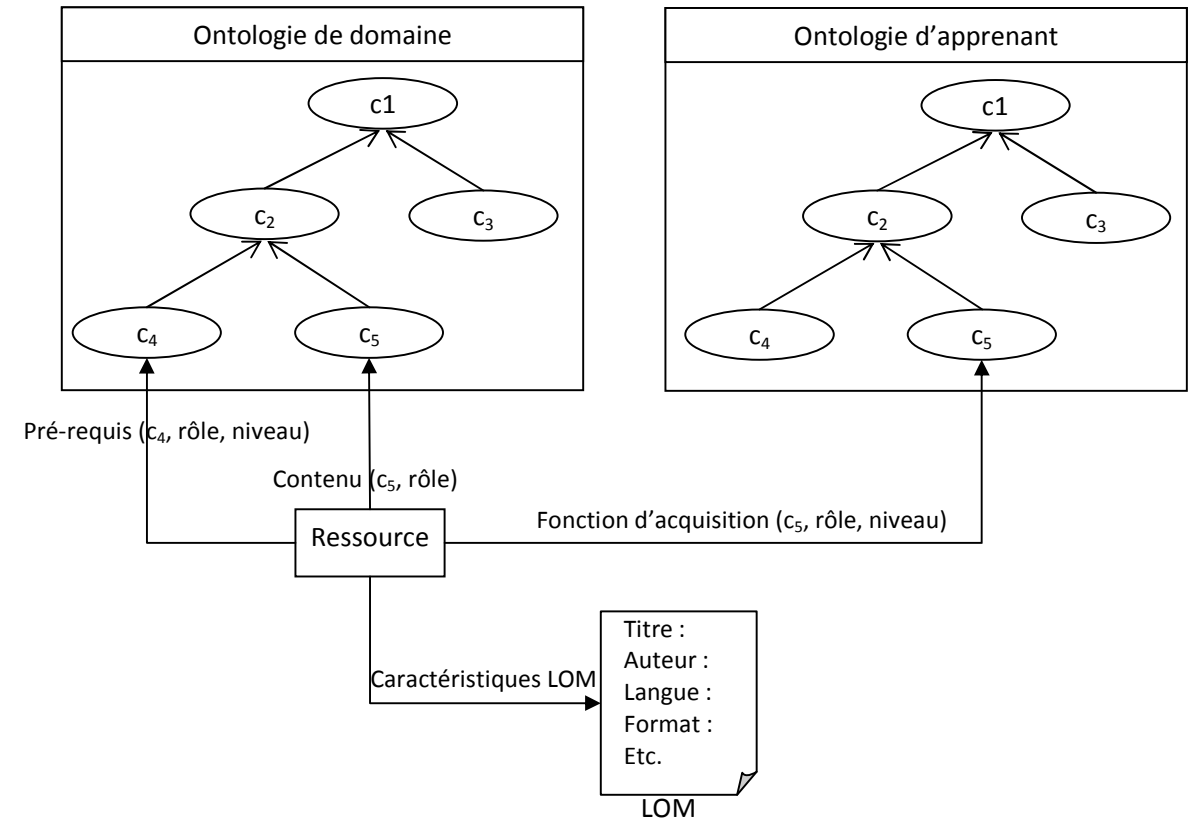


Figure 36. Modèle de ressource SIMBAD

La sémantique et la signification d'un concept sont parfois exprimées dans les ressources qui les annotent. Afin d'extraire la sémantique des ressources, nous avons étudié les métadonnées qui décrivent ces ressources.

L'analyse des propriétés des ressources nous permet de proposer et déduire d'une manière automatique un ensemble de relations sémantiques inter-ressources. Nous focalisons notre étude sur les caractéristiques de ces relations générées. Il s'agit d'une part, d'identifier la symétrie de la relation, afin d'appliquer le *matcher* et/ou le filtre sémantique approprié, et d'autre part d'identifier la transitivité de la relation qui joue un rôle important si on cherche à déduire des relations sémantiques indirectes entre les ressources. Nous expliquons dans les prochaines sections comment exploiter les caractéristiques des relations sémantiques dans les deux processus de *matching* et de filtrage.

Tout d'abord nous décrivons les phases de génération des relations sémantiques entre les ressources en se basant sur leurs métadonnées : le pré-requis et le contenu. Ces différentes relations inter-ressources sont

obtenues à l'aide des liens ensemblistes telles que les relations d'égalité (=), de divergence (#), d'inclusion ($\subset$) et d'intersection ($\cap$).

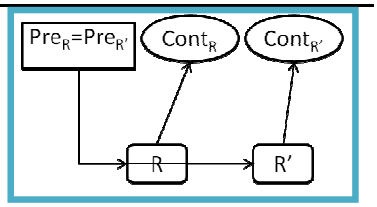
Il s'agit d'étudier les différentes combinaisons ensemblistes possibles. Nous utilisons un code binaire pour les combinaisons des différentes relations ensemblistes possibles. Voilà les cinq cas possibles lors d'une comparaison entre deux ensembles X et Y :

1. Si $X \cap Y = \varnothing$ → XY=00
2. Si $X \cap Y = X=Y$ → XY=11
3. Si $X \cap Y \# \varnothing$ et Si $X \subset Y$ →XY = 10
4. Si $X \cap Y \# \varnothing$ et Si $Y \subset X$ →XY = 01
5. Si $X \cap Y \# \varnothing$ et Si $Y \not\subset X$ →XY = 101

Soient deux ressources $R(Cont_R,Pre_R)$ et $R'(Cont_{R'},Pre_{R'})$ telles que : (i) Pre_R et $Pre_{R'}$ représentent respectivement l'ensemble des pré-requis de ces deux ressources R et R' ; (ii) $Cont_R$ et $Cont_{R'}$ représentent respectivement le contenu de ces deux ressources. Il existe quatre comparaisons possibles entre les deux caractéristiques des ressources : (i) Pre_R vs $Pre_{R'}$ et $Cont_R$ vs $Cont_{R'}$; (ii) Pre_R vs $Pre_{R'}$ et $Cont_R$ vs $Pre_{R'}$; (ii) Pre_R vs $Cont_{R''}$ et $Cont_R$ vs $Cont_{R'}$; (iv) Pre_R vs $Cont_{R'}$ et $Cont_R$ vs $Pre_{R'}$. Chaque comparaison engendre cinq possibilités. Alors on déduit que le nombre total de possibilités de combinaison est donc $5^4 = 625$ possibilités.

Afin de réduire ce grand nombre de possibilités, on suppose quelques contraintes telles que : $Cont_R$ et Pre_R sont des ensembles disjoints des concepts de l'ontologie, c'est-à-dire leur intersection est vide : $Cont_R \cap Pre_R = \varnothing$. Par conséquent : Pre_R vs $Cont_R$ = 00 et $Pre_{R'}$ vs $Cont_{R'}$ = 00. Suite à cette définition, nous déduisons les implications suivantes :

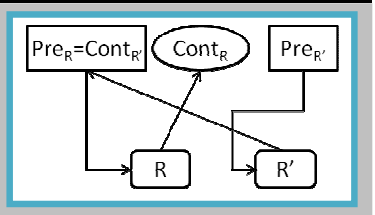
1. Si $Pre_R = Pre_{R'}$ c.-à-d. Pre_R vs $Pre_{R'}$ =11 alors
   a. Pre_R vs $Cont_{R'}$ = $Pre_{R'}$ vs $Cont_{R'}$ = 00 et
   b. $Cont_R$ vs $Pre_{R'}$=$Cont_R$ vs Pre_R = 00.



2. Si $Pre_R = Cont_{R'}$ c.-à-d. Pre_R vs $Cont_{R'}$ =11 alors

   a. $Cont_R$ vs $Cont_{R'}$ = $Cont_R$ vs Pre_R = 00 et

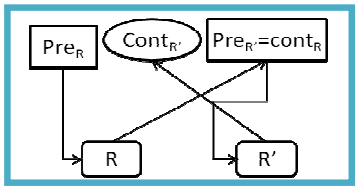
   b. Pre_R vs $Pre_{R'}$ = $Cont_{R'}$ vs $Pre_{R'}$ = 00.



3. Si $Cont_R = Pre_{R'}$ c.-à-d. $Pre_{R'}$ vs $Cont_R$ =11 alors

   a. Pre_R vs $Pre_{R'}$ = Pre_R vs $Cont_R$ = 00 et

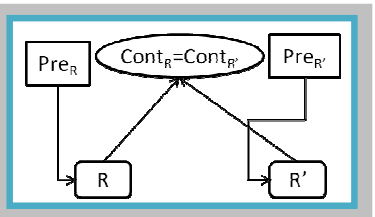
   b. $Cont_R$ vs $Cont_{R'}$ =$Pre_{R'}$ vs $Cont_{R'}$ = 00.



4. Si $Cont_R = Cont_{R'}$ c.-à-d. $Cont_R$ vs $Cont_{R'}$ =11 alors

   a. Pre_R vs $Cont_{R'}$ = Pre_R vs $Cont_R$ = 00 et

   b. $Pre_{R'}$ vs $Cont_R$ = $Pre_{R'}$ vs $Cont_{R'}$ = 00.

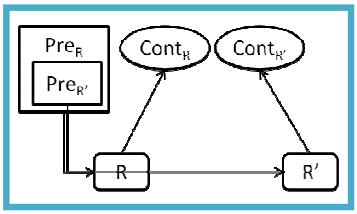


5. Si $Pre_R \subset Pre_{R'}$ c.-à-d. Pre_R vs $Pre_{R'}$ =10 alors

   a. $(Pre_R \cap Cont_{R'}) \subset (Pre_{R'} \cap Cont_{R'} = \emptyset) \rightarrow Pre_R$ vs $Cont_{R'}$ =00

6. (l'inverse de 5) Si $Pre_{R'} \subset Pre_R$ c.-à-d. Pre_R vs $Pre_{R'}$ =01 alors

   b. $(Pre_{R'} \cap Cont_R) \subset (Pre_R \cap Cont_R = \emptyset) \rightarrow Pre_{R'}$ vs $Cont_R$ =00



A partir de ces dépendances, nous obtenons les combinaisons possibles illustrées dans le tableau 5 :

Tableau 5. Les combinaisons possibles entre les métadonnées des ressources

N° de la règle	Pre_R vs $Pre_{R'}$	Pre_R vs $Cont_{R'}$	$Cont_R$ vs $Pre_{R'}$	$Cont_R$ vs $Cont_{R'}$
[1]	11	00	00	?
[2]	00	11	?	00
[3]	00	?	11	00
[4]	?	00	00	11
[5]	10	00	?	?
[6]	01	?	00	?

Le symbole « ? » signifie les différents cas possibles :{00, 01, 10, 11 ou 101}.

Par commodité, nous supposons par la suite que le contenu d'une ressource contient un seul concept d'ontologie. S'il y a plusieurs concepts dans le même contenu, on duplique cette ressource de telle sorte que chaque image est attachée à un seul concept. Par conséquent, on déduit que :

- $Cont_R$ vs $Cont_{R'}$ = (00 ou 11)
- $Cont_R$ vs $Pre_{R'}$ = (00, 10 ou 11)
- Pre_R vs $Cont_{R'}$ = (00, 01 ou 11)
- Pre_R vs $Pre_{R'}$ = (00, 01, 10, 11 ou 101)

Le Tableau 6 liste les différentes combinaisons possibles pour comparer deux ressources. Cette liste contient quatre-vingt dix relations. Les lignes sans numérotation indiquent une contradiction avec au moins l'un des liens de dépendance du Tableau 5. On obtient ainsi une liste plus réduite de vingt trois relations.

Tableau 6. Liste de toutes les combinaisons possibles.

N°	Pre_R $Pre_{R'}$	Pre_R $Cont_{R'}$	$Cont_R$ $Pre_{R'}$	$Cont_R$ $Cont_{R'}$	N°	Pre_R $Pre_{R'}$	Pre_R $Cont_{R'}$	$Cont_R$ $Pre_{R'}$	$Cont_R$ $Cont_{R'}$	N°	Pre_R $Pre_{R'}$	Pre_R $Cont_{R'}$	$Cont_R$ $Pre_{R'}$	$Cont_R$ $Cont_{R'}$
[1]	00	00	00	00		01	11	00	00		11	01	00	00
[2]	00	00	00	11		01	11	00	11		11	01	00	11
[3]	00	00	10	00		01	11	10	00		11	01	10	00
	00	00	10	11		01	11	10	11		11	01	10	11
[4]	00	00	11	00		01	11	11	00		11	01	11	00
	00	00	11	11		01	11	11	11		11	01	11	11
[5]	00	01	00	00	[14]	10	00	00	00		11	11	00	00
	00	01	00	11	[15]	10	00	00	11		11	11	00	11
[6]	00	01	10	00	[16]	10	00	10	00		11	11	10	00
	00	01	10	11		10	00	10	11		11	11	10	11
[7]	00	01	11	00		10	00	11	00		11	11	11	00
	00	01	11	11		10	00	11	11		11	11	11	11
[8]	00	11	00	00		10	01	00	00	[19]	101	00	00	00
	00	11	00	11		10	01	00	11	[20]	101	00	00	11
[9]	00	11	10	00		10	01	10	00	[21]	101	00	10	00
	00	11	10	11		10	01	10	11		101	00	10	11
[10]	00	11	11	00		10	01	11	00		101	00	11	00
	00	11	11	11		10	01	11	11		101	00	11	11
[11]	01	00	00	00		10	11	00	00	[22]	101	01	00	00
[12]	01	00	00	11		10	11	00	11		101	01	00	11
	01	00	10	00		10	11	10	00	[23]	101	01	10	00
	01	00	10	11		10	11	10	11		101	01	10	11
	01	00	11	00		10	11	11	00		101	01	11	00
	01	00	11	11		10	11	11	11		101	01	11	11
[13]	01	01	00	00	[17]	11	00	00	00		101	11	00	00
	01	01	00	11	[18]	11	00	00	11		101	11	00	11

	01	01	10	00		11	00	10	00		101	11	10	00
	01	01	10	11		11	00	10	11		101	11	10	11
	01	01	11	00		11	00	11	00		101	11	11	00
	01	01	11	11		11	00	11	11		101	11	11	11

Quelques relations listées dans ce tableau définissent la même relation sémantique. Par exemple, la relation 11 et la relation 14 sont vérifiées si le pré-requis d'une ressource est inclus dans l'autre. Seules les relations les plus significatives sont utilisées, elles sont citées dans le Tableau 7. Les propriétés associées à ces relations sont déduites à partir des propriétés des opérateurs de comparaison. Il s'agit des quatre opérateurs : l'égalité (=) ; l'intersection ($\cap$), l'inclusion ($\subset$) et la différence (#), tels que :

- L'égalité (=) est réflexive, symétrique et transitive ;
- L'intersection ($\cap$) est symétrique et non-transitive ;
- L'inclusion ($\subset$) est antiréflexive, antisymétrique et transitive ;
- La différence (#) est symétrique ;

Le Tableau 7 résume l'ensemble des relations sémantiques inter-ressources. Il est à noter qu'il y a six relations qui ne sont pas prises en charge dans notre étude (les relations : 2, 19, 20, 21, 22, 23 du Tableau 6) car elles n'ont pas de sens dans notre contexte.

Tableau 7. Les relations sémantiques entre deux ressources

	Règle de la ressource	Nom de la relation	propriétés de la relation	Lignes
1.	R et R' sont différents, s'il n'y a aucune relation entre leurs métadonnées	Mismatch	Symétrique	1
2.	R est équivalente à R', Si $Pre_R = Pre_{R'}$ et $Cont_R = Cont_{R'}$	Equivalence	Symétrique et Transitive	18
3.	R est faiblement substituable par R', Si $Pre_R \subset Pre_{R'}$	Substitution-faible	Antisymétrique et Transitive	11-14
4.	R est substituable par R', Si $Pre_R = Pre_{R'}$	Substitution-forte	Symétrique et Transitive	17
5.	R précède faiblement R', Si $Cont_R \subset Pre_{R'}$	Précédence-faible	Transitive	3-5

6.	R précède fortement R', Si $Cont_R = Pre_{R'}$	Précédence-forte	Antisymétrique et transitive	4-8
7.	R croise faiblement R', Si $Cont_R \subset Pre_{R'}$ et $Cont_{R'} \subset Pre_{R'}$	Croisement-faible	Symétrique	6
8.	R croise fortement R', Si $Cont_R = Pre_{R'}$ et $Cont_{R'} = Pre_{R'}$	Croisement-fort	Symétrique	7-9-10
9.	R est une partie de R', Si $(Cont_R , Pre_{R'}) \subset Pre_{R'}$	Part-of (partie de)	Antisymétrique et transitive	13-16
10.	R est plus spécifique que R' Si $Pre_R \subset Pre_{R'}$ et $Cont_R = Cont_{R'}$	Plus spécifique	Transitive	15
11.	R est plus général que R' Si $Pre_R \supset Pre_{R'}$ et $Cont_R = Cont_{R'}$	plus général	Transitive	12

Parmi les onze relations sémantiques produites entre deux ressources, nous montrons sur la Figure 37 un exemple de deux relations sémantiques à savoir la substitution forte et la précédence faible.

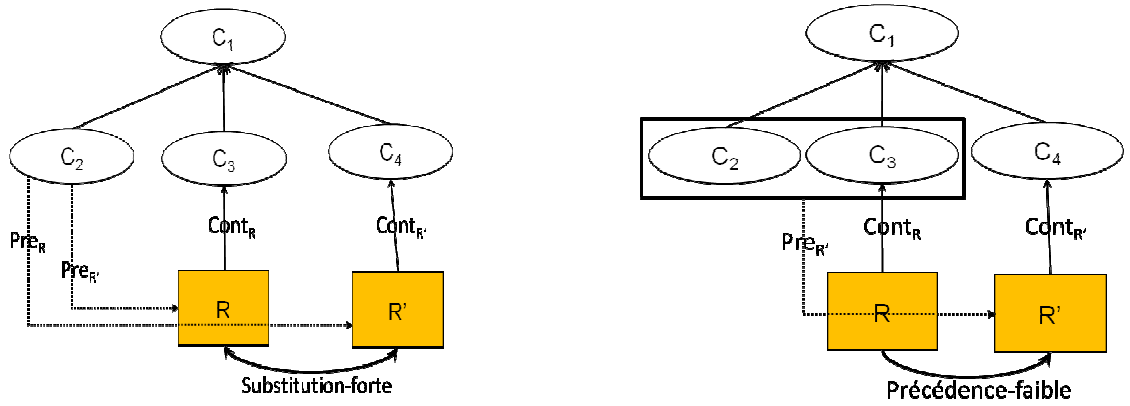


Figure 37. Exemples de relation sémantique inter-ressources.

IV.2.2.2 Relation sémantique inter-concepts

Pour chaque relation produite entre deux ressources, une relation équivalente entre les concepts associés sera créée. Cependant, cette propagation dépend du nombre de ressources associées à chaque

concept. Autrement dit, nous voulons considérer l'ensemble des ressources annotées par chaque concept plutôt qu'une ressource individuelle. Chaque ensemble de ressources développe le même concept mais avec des pré-requis probablement différents. Par conséquent, nous classons ces ressources dans plusieurs sous-ensembles regroupant les ressources équivalentes, c'est-à-dire celles qui ont un même contenu et un même pré-requis.

Soient deux concepts A et B et deux ensembles de ressources RA et RB associés respectivement à A et B. RA_i et RB_i sont les sous ensembles de ressources équivalentes qui développent respectivement le concept A et le concept B et qui appartiennent respectivement à RA et RB. Durant la phase de génération des relations inter-ressources (section précédente), les relations sémantiques entre les sous-ensembles RA_i et RB_i sont générées.

Le problème est de savoir comment propager cette connaissance vers les concepts correspondants pour avoir une nouvelle relation entre les deux concepts A et B. Notre proposition est d'associer un poids à chaque relation selon le degré de similarité entre les sous-ensembles RAi et RBi. Pour cela, nous avons employé la mesure de similarité de Jaccard définie précédemment. Elle est utilisée pour calculer la distance entre les deux ensembles RA et RB et prend la valeur « 0 » quand RA et RB sont disjoints, et la valeur « 1 » quand RA et RB sont identiques.

Nous associons un poids à la relation sémantique entre les deux concepts A et B, qui exprime le degré de la relation produite. Il est calculé à l'aide de la mesure de Jaccard entre les deux ensembles de ressources RAi et RBi associés respectivement à A et B (Figure 38). Formellement, la relation sémantique entre deux concepts A et B est décrite par le 3-uplet (A, B, σ_{rel}(RAi, RBi)) où :

- σ_{rel}(RAi, RBi) correspond à la valeur de la mesure de Jaccard entre les deux sous ensembles RAi et RBi, et
- rel est la relation sémantique qui relie chaque élément de l'ensemble RAi par au moins un élément de l'ensemble RBi.

Soit Я={rel_1,…rel_K} l'ensemble des relation sémantiques existantes entre les sous ensembles RA et RB (Figure 38.), notée $RA_k \leftarrow rel_i \rightarrow RB_l$. La mesure de Jaccard d'une relation rel_i entre deux ensembles RA et RB est définie par la formule suivante:

$$\forall rel_i \in Я\ \sigma_{reli}(RA,RB) = \frac{\left| RA \bigcap_{reli} RB \right|}{\left| RA \bigcup RB \right|} \quad \text{Avec } RA \bigcap_{reli} RB = \{a_j \cup b_k / \forall a_j \in A, \exists b_k \in B / a_j \leftarrow rel_i \rightarrow b_k\} = A_i \cup B_i$$

Normalisation :

1. $\sigma_{reli}(RA,RB) = 0$ si $RA \bigcap_{reli} RB = \varnothing$, c.-à-d. qu'aucune ressource de RA ne correspond à une ressource de RB par la relation rel_i et
2. $\sigma_{reli}(RA,RB) = 1$ si $RA \bigcap_{reli} RB = RA = RB$, c.-à-d. que chaque ressource de RA correspond à une ressource de RB par la relation rel_i.

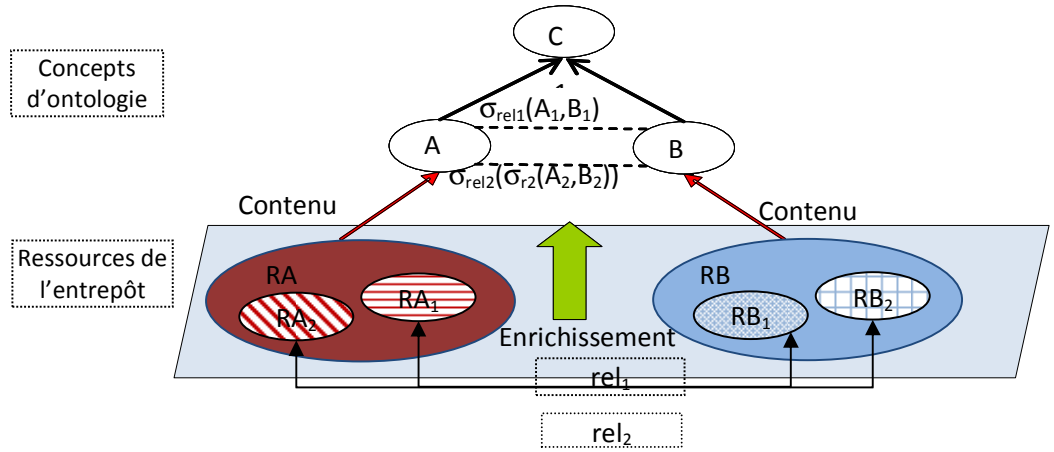


Figure 38. Enrichissement sémantiques de l'ontologie dans le modèle Simbad.

La Figure 38 illustre le principe d'enrichissement de l'ontologie à partir des relations existantes ou identifiées en examinant leurs métadonnées. Nous qualifions l'ensemble des liens crées lors de la phase d'enrichissement entre les concepts de la même ontologie par «liens (ou relations) sémantiques générés». Ces relations générées sont ensuite exploitées par les deux processus de matching et de filtrage pour identifier les correspondances.

IV.3 ROMIE appliqué au contexte biomédical

Les ontologies sont de plus en plus employées et deviennent importantes dans plusieurs secteurs, y compris le domaine biomédical. Elles sont adoptées comme une base pour l'interopérabilité entre les systèmes et pour l'intégration des données, en fournissant une terminologie commune au-dessus des données du domaine. En biomédical, plusieurs ontologies ont été développées afin de couvrir des domaines spécifiques, par exemple l'ontologie des gènes ou Gene Ontology (GO), l'ontologie des voies métaboliques ou Pathway ontology (PW), l'ontologie des maladies humaines (DOID), l'ontologie du modèle fondamental de l'anatomie (FMA),

l'ontologie de l'anatomie de la souris adulte (MA), l'ontologie de la pathologie de la souris (MPATH), etc. Un grand nombre de ces ontologies est accessible via le site Web OBO (*Open Biomedical Ontologies*) [75] (voir Figure 39). Ces ontologies sont, dans la plupart du temps, complémentaires ou se recouvrent. Par ailleurs, l'OBD (*Open Biomedical Database*) [74] englobe un ensemble d'entrepôts d'instances qui sont annotés par les ontologies d'OBO, tels que : OMIM data, ZFin, FlyBase, etc.

Nous constatons ces dernières années une augmentation du nombre de projets biomédicaux qui utilisent plusieurs ontologies développant des domaines de connaissances complémentaires. Le projet ANR SAPHIR (*Systems Approach for PHysiological Integration of Renal*) est une approche systématisée pour l'intégration physiologique des fonctions rénales, cardiaques et respiratoires. Afin d'employer les ontologies nécessaires d'une manière intégrée, des « ponts » doivent être créés entre ces différentes ontologies.

Le *mapping* d'ontologies dans le secteur biomédical est souvent établi manuellement par des experts, mais en raison de l'augmentation du nombre d'ontologies ainsi que de leurs tailles, on a besoin de créer des correspondances entre ces ontologies d'une manière automatique ou au moins semi-automatique. Plusieurs algorithmes et outils ont été proposés pour résoudre ce problème de *mapping*, mais en raison de la complexité de la tâche, aucune de ces solutions n'est complètement satisfaisante.

Aucun des travaux existants n'utilise les informations et/ou les données attachées aux ontologies lors du processus de *mapping*. En revanche, lorsque des instances ou des ressources sont disponibles, il y a une très bonne opportunité de les exploiter par le système de *mapping*. Par exemple, quand deux concepts partagent exactement le même ensemble d'instances, nous déduisons qu'il y a une forte probabilité que le lien de correspondance entre ces deux concepts représente un *mapping* correct.

Molecular function	GO	gene ontology.obo	OBO	no	yes
name ontology)	MR	molecule role.obo	OBO	no	yes
Mosquito gross anatomy	TGMA	mosquito anatomy.obo	OBO	no	yes
Mouse adult gross anatomy	MA	adult mouse anatomy.obo	OBO	no	yes
Mouse gross anatomy and development	EMAP	EMAP.obo	OBO	no	yes
Mouse pathology	MPATH	mouse pathology.obo	OBO	no	yes
Multiple alignment	RO	mao.obo	OBO	no	yes
NCBI organismal classification	taxon	taxonomy.dat	plain text	no	no
			OWL	no	no
OBO relationship types	OBO_REL	relationship.obo	OBO	yes	yes
Pathway ontology	PW	pathway.obo	OBO	no	yes
PATO	PATO	quality.obo	OBO	yes	yes
Physico-chemical methods and properties	FIX	fix.obo	OBO	no	yes
Physico-chemical process	REX	rex.obo	OBO	no	yes
Plant environmental conditions	EO	environment ontology.obo	OBO	no	yes
Plant growth and developmental stage	PO	po temporal.obo	OBO	no	yes
Plant structure	PO	po anatomy.obo	OBO	no	yes
Plasmodium life cycle	PLO	PLO.ontology PLO.defs	GO	no	yes
Protein covalent bond	[none]	[none]	[none]	no	no
Protein domain	IPR	InterPro FTP directory	XML	no	no
Protein modification	MOD	psi-mod.obo	OBO	no	yes
Protein-protein interaction	MI	psi-mi.obo	OBO	no	yes
			OWL	no	yes
Sequence types and features	SO	so.obo	OBO	yes	yes

Figure 39. Liste des ontologies biomédicales sur le site OBO

Notre analyse du domaine biomédical nous a permis de constater que les ontologies forment un graphe acyclique avec des nœuds qui représentent des concepts. Les arcs entre ces nœuds représentent l'une des deux relations : « is-a » ou « part-of ». Les concepts peuvent avoir de multiples instances, c.-à-d. les instances (ou les ressources) qui sont décrites où classifiées par le concept. De la même manière, une instance peut être associée à de multiples concepts. Par conséquent, les associations entre les instances et les concepts d'ontologie sont de cardinalité n:m.

Avant de présenter la façon dont le processus de génération de correspondances utilise les instances, il faut comprendre la nature et la particularité des instances biomédicales. Nous illustrons dans les deux prochaines sections les liens existants entre les concepts d'ontologies et les instances de l'entrepôt (c'est-à-dire, l'annotation des instances par les concepts) ainsi que les différents types d'annotation possibles. Finalement,

nous expliquons comment notre système ROMIE exploite toutes ces informations suivant le type d'annotation dans le processus de *mapping*.

Mais tout d'abord, une définition des relations entre les instances biomédicales est nécessaire.

IV.3.1 Les relations entre les concepts d'ontologie et les instances

Dans les ontologies, le terme « classe », également appelé « concept », est employé pour exprimer ce qui est général. En revanche, les instances représentent ce qui est particulier. Nous trouvons dans l'état de l'art d'autres dénominations du terme instance, comme individu ou ressource.

Notre tâche est de réaliser des liens entre les concepts (ou les classes) des ontologies. Cette tâche n'est pas facile. Les termes des ontologies biomédicales se réfèrent exclusivement aux classes, c'est-à-dire à ce qui est général. Nous ne pouvons pas définir la signification des labels (ou des URI) des classes sans prendre en considération leurs instances. Par exemple, il est impossible de connaître la vraie signification de la relation « part_of » reliant les concepts de l'ontologie, sans prendre en considération les instances correspondantes.

La relation d'annotation n'est pas une relation ontologique, car elle ne lie pas des classes de même nature, mais plutôt une classe avec un ensemble d'instances. Sur les annotations des bases d'instances qui sont disponibles, nous trouvons des relations au niveau instance, qui permettent de définir des liens entre les différentes instances. Par conséquent, nous pouvons distinguer trois types de relations binaires :

- <classe, classe> : est l'ensemble des relations ontologiques, telles que les relations hiérarchiques et rhétoriques entre les concepts de l'ontologie ; par exemple la relation « is_a » ou « part_of ».
- <classe, instance> : est l'ensemble des relations d'annotation permettant d'annoter un concept (classe) par un ensemble d'instances. ; par exemple la relation « instance_of ».
- <instance, instance> : est l'ensemble des relations inter instances (c.-à-d. les relations qu'on trouve au niveau instance) ; par exemple la relation « part_of* » ou encore « quality_of* » reliant les instances entre elles.

Les relations définies par le symbole « * » telle que la relation « part_of* » représentent les relations de niveau instance, c'est-à- dire entre deux instances. Il existe un nombre important de relations entre les instances biomédicales, et nous ne pouvons les citer et les définir toutes. Les relations que nous définissons ci-dessous devraient être abstraites et générales (indépendantes du contexte biomédical), afin qu'elles soient appliquées à d'autres domaines. Ci-après la définition de quelques relations que nous avons trouvées entre les instances des entrepôts biomédicaux :

Relation	Sa signification
i instance_of c	i est une instance de la classe c.
i_1 part_of* i_2	L'instance i_1 fait partie de l'instance i_2 (c.-à-d., i_1 inclue dans i_2)
i_1 quality_of* i_2	L'instance i_1 est de qualité i_2 (par exemple, la couleur des yeux est bleu)
i_1 Overlaps* i_2	L'instance i_1 contient l'instance i_2
Etc.	

Cette liste contient des relations de type <instance-instance> ainsi qu'une relation de type <instance-classe>. Ces relations sont nécessaires pour définir les relations qui sont notre cible principale dans notre étude, à savoir les relations de type <classe-classe>. Nous définissons par la suite les différentes façons d'annoter un entrepôt de données.

IV.3.2 Les types d'annotations des instances biomédicales

En étudiant les termes des concepts ontologiques, nous constatons que la vraie sémantique d'un concept est souvent mieux exprimée par les instances et leurs relations avec l'ontologie, plutôt que par des métadonnées comme le label ou l'URI du concept. Nous préconisons que les approches de *mapping* à base d'instances utilisent des associations existant entre les concepts d'ontologie et les instances. Cette association est nommée « annotation ». Par exemple, des objets de biologie moléculaire comme les protéines ou les gènes sont décrits ou annotés par les concepts de l'ontologie GO-MF (c.-à-d. la partie MF : *Molecular Function* de l'ontologie GO -Gene Ontology-).

Les annotations fournissent des liens entre les concepts de l'ontologie et les instances qui peuvent être des documents, des fichiers plats, des bases de données, etc. Une annotation d'une instance se compose de sa référence URI, d'une référence URI sur le concept de l'ontologie et des références sur les éventuelles relations vers les autres instances. La Figure 40 montre les deux types de relations que nous pouvons trouver dans un fichier d'annotation. Le premier type de relation est un lien entre les concepts de l'ontologie et les instances de l'entrepôt et le deuxième type exprime les relations existant entre les instances d'un entrepôt.

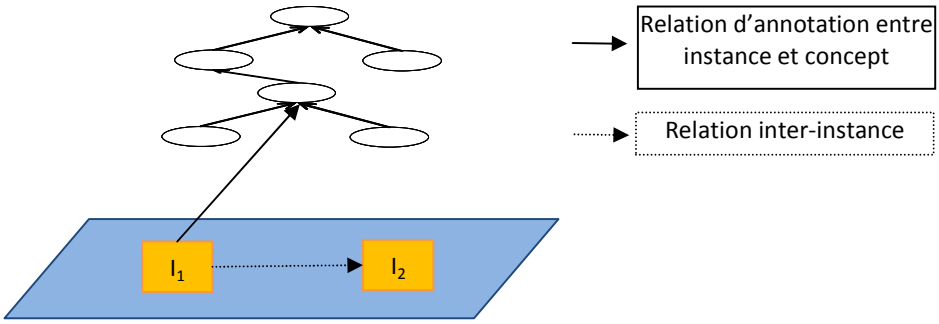


Figure 40. Les types de relations exprimées dans un fichier d'annotation.

Dans le domaine biomédical, nous trouvons plusieurs types d'annotations. L'étude de ces annotations nous a amené à définir trois sortes d'annotations :

1. Un entrepôt d'instances et des données annotées par une seule ontologie ; autrement dit, une ontologie spécifique qui développe une base d'instances (Figure 41).
2. Plusieurs ontologies qui développent plusieurs entrepôts d'instances différentes reliées par des relations sémantiques (Figure 42).
3. Un entrepôt d'instances annoté par plusieurs ontologies (Figure 43).

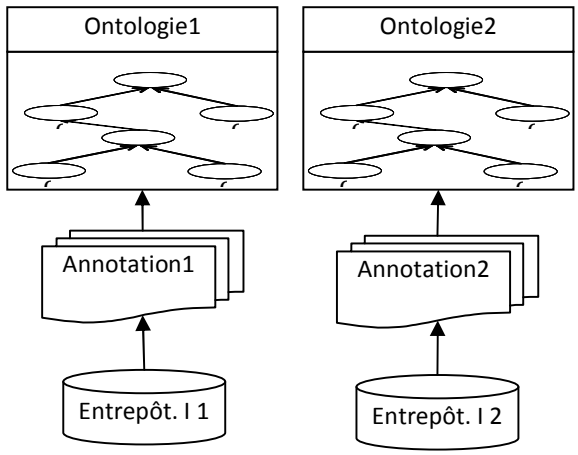


Figure 41. Cas n°1 - deux entrepôts d'instances disjoints annotés par deux ontologies

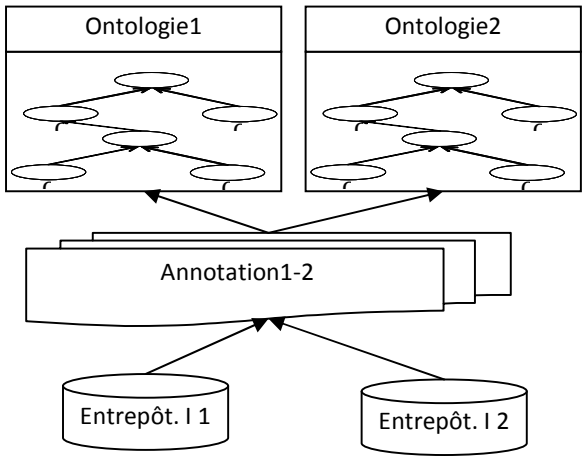


Figure 42. Cas n°2 - deux entrepôts d'instances en relation annotés par deux ontologies

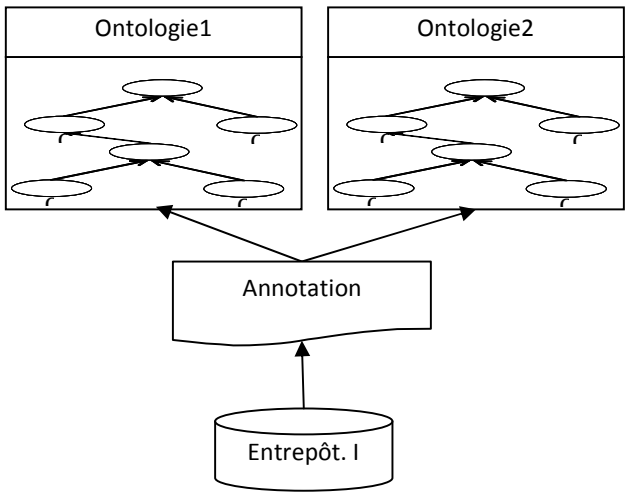


Figure 43. Cas n°3 - un entrepôt d'instances annoté par deux ontologies

IV.3.3 Exploitation des instances dans le processus de *mapping*

Nous définissons par la suite les différents modes de fonctionnement du processus de *mapping* suivant les trois types d'annotations cités ci-dessus. Nous concluons notre explication par un exemple réel d'ontologies et de leurs bases d'instances.

IV.3.3.1 Annotation des entrepôts d'instances disjointes par des ontologies disjointes

Le partage des instances entre des entrepôts disjoints avec leurs annotations par des ontologies différentes est semblable à notre étude présentée dans le contexte pédagogique. Il s'agit d'étudier et propager les relations existantes aux niveaux instances sur les concepts d'ontologies, avant même de commencer le processus de *mapping* (Figure 44).

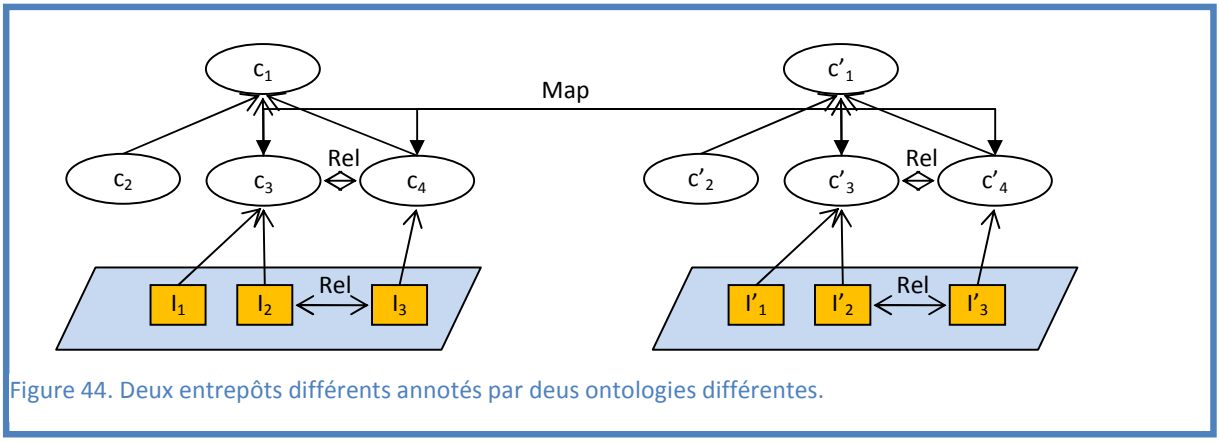


Figure 44. Deux entrepôts différents annotés par deus ontologies différentes.

L'enrichissement sémantique de l'ontologie consiste à créer des liens sémantiques entre les concepts en comparant leurs ressources (instances) associées. Les entrepôts de données biomédicales sont très riches de relations sémantiques entre les différentes instances.

Le Tableau 8 liste quelques relations engendrées entre les concepts en utilisant les liens existants entre les instances. Comme nous l'avons noté plus haut, il existe un nombre important de relations entre les instances biomédicales ; nous nous limitons ici aux relations fréquemment utilisées. Afin d'utiliser les relations inter-concepts dans les deux processus de *matching* et de filtrage, des propriétés sont à définir pour chacune de ces relations.

Tableau 8. La liste des relations inter-concepts à partir des relations inter-instances

Name	Définition	propriétés	Commentaire
i instance_of c	i est une instance du concept c		
c is_a c'	∀i instance_of c/i instance_of c'	[transitive] [réflexive] [antisymétrique]	
c part_of c'	∀i instance_of c ∃i' instance_of c'/ i part_of* i'	[transitive] [réflexive] [antisymétrique]	
c Has_part c'	∀i instance_of c ∃i' instance_of c'/ i has_part* i'	[transitive] [réflexive] [antisymétrique]	L'inverse de la relation part_of
c integral_part_of c'	Si c part_of c' et c' has_part c	[transitive] [réflexive] [antisymétrique]	
c located_in c'	∀i instance_of c ∃i' instance_of c'/ i located_in* i'	[transitive] [réflexive]	L'inverse est : c' location_of c
c contained_in c'	∀i instance_of c ∃i' instance_of c'/i located_in* i' et Not∃ i *overlaps* i'.	[transitive] [réflexive]	L'inverse est : c' contains_of c

Le nombre d'instances et de leurs relations joue un rôle important pour exprimer l'exactitude de la relation générée entre les concepts. Plus précisément, à l'aide de la mesure de Jaccard, un poids sera affecté aux différentes relations citées dans le tableau ci-dessus. Formellement, soit I (respectivement I') l'ensemble des instances i_j (respectivement i'_j) annotées par le concept c (respectivement c'). Nous définissons la relation « rel » entre les deux concepts « c » et « c' » à partir de la relation « rel* » entre les instances de I et I', avec le degré $\sigma_{rel}(I, I')$ défini par la mesure de Jaccard suivante :

$$\sigma_{rel}(I, I') = \frac{2 * \left| I \bigcap_{rel*} I' \right|}{\left| I \bigcup I' \right|} \text{ Avec } I \bigcap_{rel*} I' = \{ i_j \in I \ / \ \exists i'_k \in I' \ (i_j \ rel* \ i'_k) \}$$

IV.3.3.2 Annotation des entrepôts d'instances convergentes par des ontologies disjointes

Le processus de *mapping* à base d'instances – comme présenté dans les précédentes sections – est fondé sur deux phases successives : la première commence avant les méthodes de comparaison et de *mapping*, elle consiste à créer des nouvelles relations sémantiques au sein de chacune des deux ontologies à aligner ; ces relations seront exploitées dans la deuxième phase soit pour identifier des liens de correspondance, soit pour détecter et éliminer les *mappings* les moins pertinents.

Jusqu'à présent, les relations de correspondance permettent d'établir l'une des trois relations possibles, à savoir : l'inclusion ($\subset$), le recouvrement ($\supset$) ou l'équivalence ($\equiv$). En revanche, dans cette partie et avec ce type de lien entre les concepts d'ontologies et les instances de l'entrepôt, nous définissons une relation sémantique entre le couple de concepts à aligner.

L'une des particularités de l'annotation des instances biomédicales que nous avons pu soulever lors de notre étude est la possibilité d'avoir des liens sémantiques entre les instances de deux ou plusieurs entrepôts. Par conséquent, il ne s'agit pas d'une phase d'enrichissement au sein d'une même ontologie mais plutôt d'un lien de correspondance qui sera établi entre les concepts de deux ontologies différentes avec une relation de correspondance bien particulière. La Figure 45 montre la création d'un lien de correspondance (un *mapping* candidat) entre les deux concepts (c_4 et c'_2) avec la relation « Rel » qui relie deux sous-ensembles d'instances annotés par ces deux concepts.

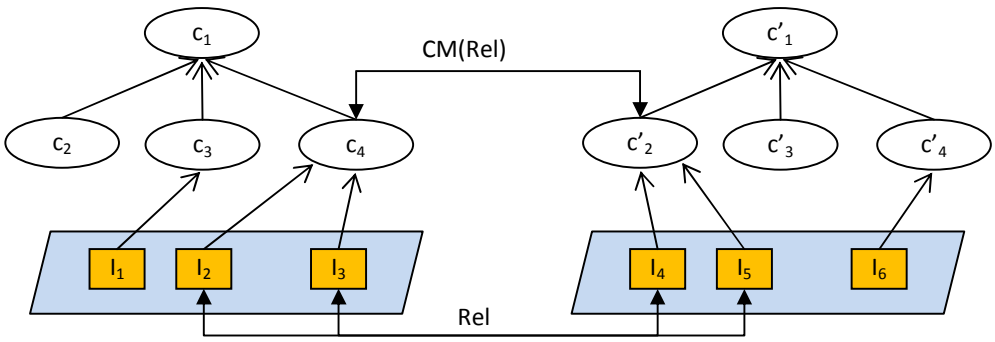


Figure 45.. Deux entrepôts convergents annotés par deux ontologies différentes.

Contrairement au précédent cas d'annotation (section précédente) qui consiste à réaliser une opération avant le processus de *mapping* , c'est-à-dire l'enrichissement de l'ontologie avant le processus de *mapping*, il s'agit dans le cas actuel d'un *matcher* sémantique qui permet de créer un *mapping* candidat, et son résultat sera combiné avec les résultats des autres *matchers*. Comme tous les *matchers* de notre système, ce *matcher* sémantique est caractérisé par le 5-uplet <R, c, c', Cf, SV > avec :

- R relation existant entre les instances annotées par le concept « c » et les instances annotées par le concept « c' » ;
- Cf degré de confiance associé à ce *matcher* par l'administrateur du système ;
- SV valeur de similarité, elle exprime la densité de la relation « R ». Nous utilisons la mesure de Jaccard pour calculer le pourcentage d'instances reliées par la relation « R » par rapport au nombre total d'instances des deux concepts « c » et « c' ». Formellement, étant donné I (respectivement I') l'ensemble des instances annotées par « c » (respectivement « c' »), SV est calculé par :

$$SV = \frac{2 * \left| I \bigcap_{rel*} I' \right|}{\left| I \bigcup I' \right|} \quad \text{Avec } I \bigcap_{rel*} I' = \{ i_j \in I \ / \ \exists i'_k \in I' \ (i_j \ rel* \ i'_k) \}$$

IV.3.3.3 Annotation d'un entrepôt d'instances par plusieurs ontologies

Cette partie décrit le comportement de notre processus d'appariement de deux ontologies qui partagent un entrepôt d'instances commun. Il s'agit d'une base d'instances annotées par les termes de plusieurs ontologies (Figure 46). Dans ce cas, la connaissance du domaine spécifique représenté par les associations (les annotations)

est utilisée pour déterminer sémantiquement les correspondances (*mappings*) significatifs. Notons que ce type d'annotation se retrouve dans d'autres domaines que le domaine biomédical.

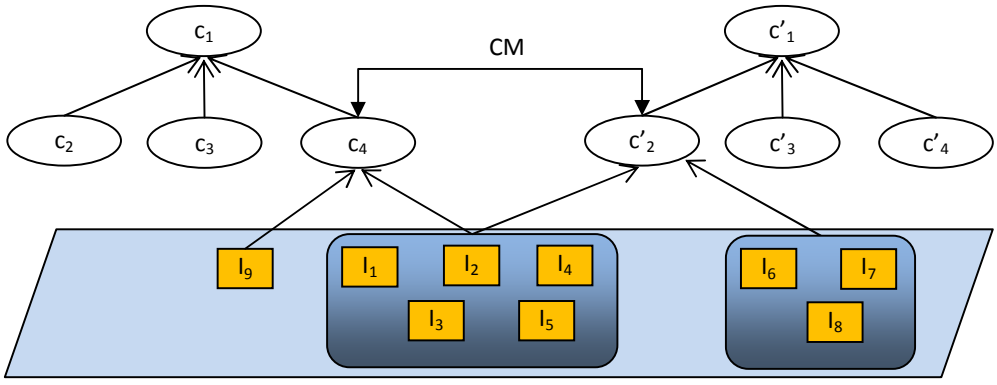


Figure 46. Un entrepôt annoté par deux ontologies différentes.

Afin d'aligner deux ontologies, nous avons besoin de méthodes d'appariement (*matchers*) pour déterminer la similarité entre les concepts de ces deux ontologies. L'idée principale de notre approche d'alignement à base d'instances est d'identifier et calculer la similarité entre les concepts à partir du nombre d'instances partagées, c.-à-d. le nombre d'instances communes associées aux deux concepts. Un avantage important de notre approche est l'indépendance des résultats d'appariement relativement aux noms des concepts et/ou aux autres métadonnées.

Dans notre étude, nous avons défini une méthode d'appariement (*matcher*) pour déterminer la similarité entre des couples de concepts en se basant sur le nombre d'instances communes entre eux. On nomme ce *matcher* « *matcher* d'instances communes » et on le note $M_{CommonInstance}$. Il est caractérisé par le 5-uplet <≡, c, c', Cf, SV > avec :

- ≡ : relation d'équivalence générée entre les deux concepts « c » et « c' »;
- Cf : degré de confiance associé au $M_{CommonInstance}$ par l'administrateur du système ;
- SV : valeur de similarité entre le couple de concepts. Elle exprime le rapport entre le nombre d'instances communes (c'est-à-dire les instances annotées à la fois par les deux concepts « c » et « c' ») et le nombre d'instances non communes (c'est-à-dire les instances annotées par « c » ou bien par « c' »). Afin de calculer ce rapport, nous utilisons la mesure de Jaccard. Formellement,

étant donné I (respectivement I') l'ensemble des instances annotées par « c » (respectivement « c' »), SV est calculé par :

$$SV = \frac{2 * \left| I \bigcap I' \right|}{\left| I \bigcup I' \right|}$$ tel que $I \bigcap I'$ retourne l'ensemble des instances communes entre les deux concepts « c » et « c' ».

IV.3.3.4 Exemple d'application

Nous montrons dans cette partie un exemple réel d'ontologies et de leurs bases d'instances. La particularité de cet exemple est la possibilité de voir et traiter les trois types d'annotations cités auparavant.

La Figure 47 montre l'annotation des deux bases d'instances : *FlyBase*[1] *et ZFIN*[2] par les cinq ontologies OBO suivantes : *Cell, Fly_anatomy, GO-BP, Pato et Quality*. La Figure 48 illustre un extrait du fichier d'annotation '*fly_pheno.rdf',* qui contient d'une part les liens d'instanciation des instances de *FlyBase* et *ZFin* par les concepts des cinq ontologies, et d'autre part les différentes relations entre les instances de ces deux entrepôts.

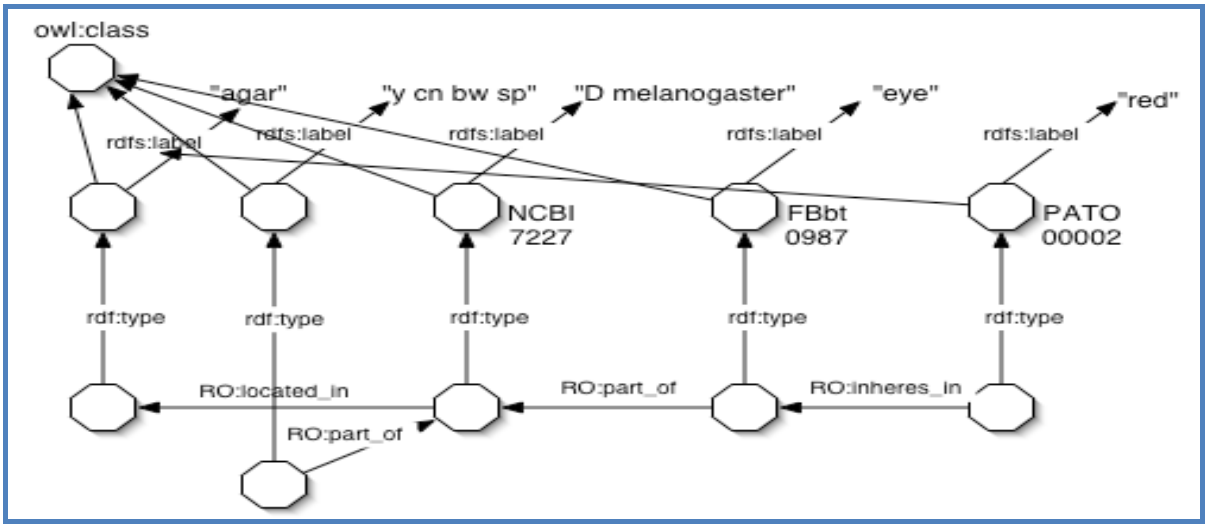


Figure 47. L'annotation des instances de FlyBase par plusieurs ontologies

[1] FlyBase est une base de données pour la génétique de drosophile et la biologie moléculaire, http://flybase.org/

[2] ZFIN est une base de données pour le modèle d'organisation du poisson, http://zfin.org/cgi-bin/webdriver?MIval=aa-ZDB_home.apg

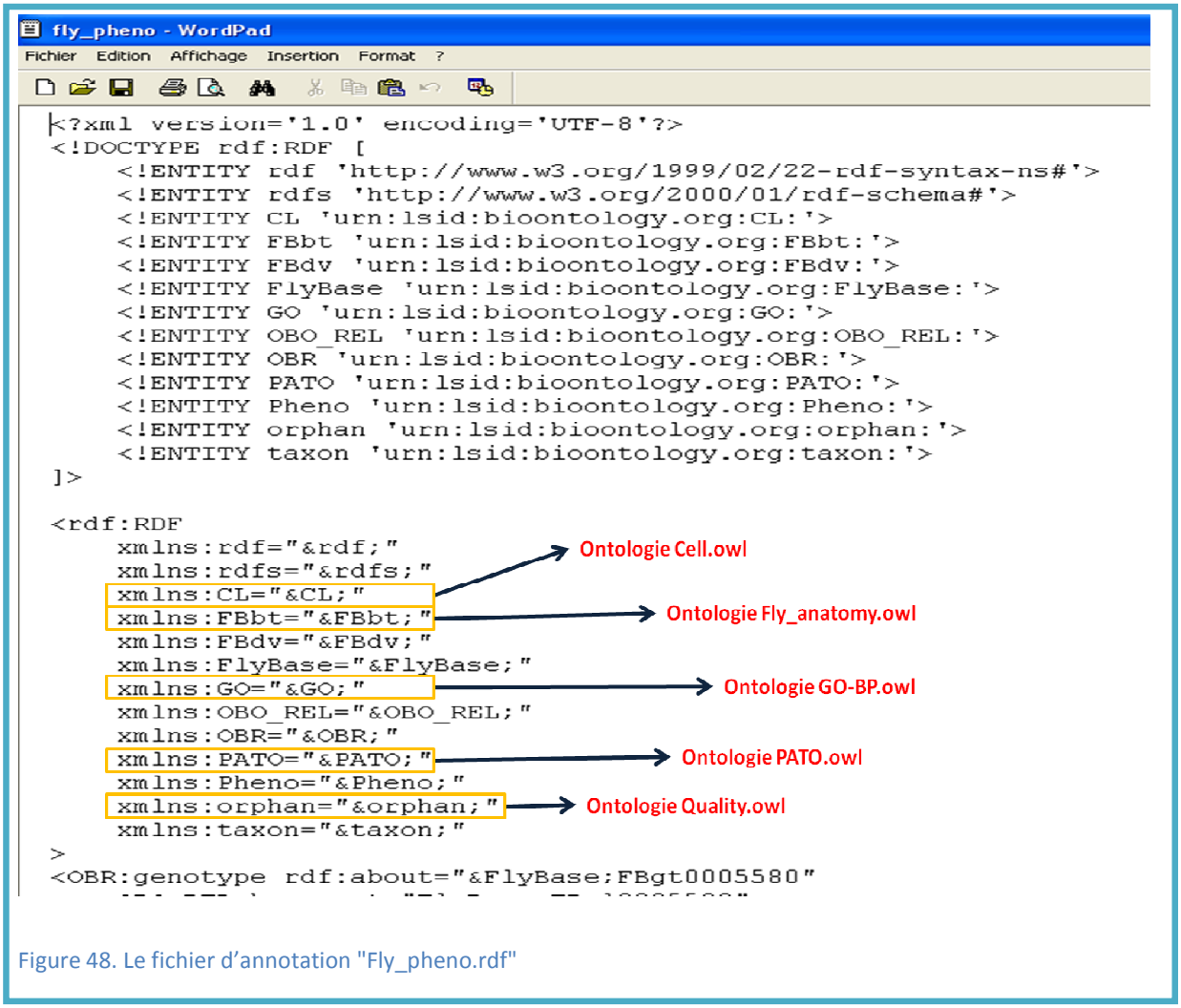


Figure 48. Le fichier d'annotation "Fly_pheno.rdf"

Le fichier d'annotation contient plusieurs informations sur les relations entre les instances et les relations d'annotation entre les concepts et leurs instances. Pour faciliter la présentation de notre exemple, la Figure 49 schématise les liens entre tous les éléments du fichier Fly_pheno.rdf.

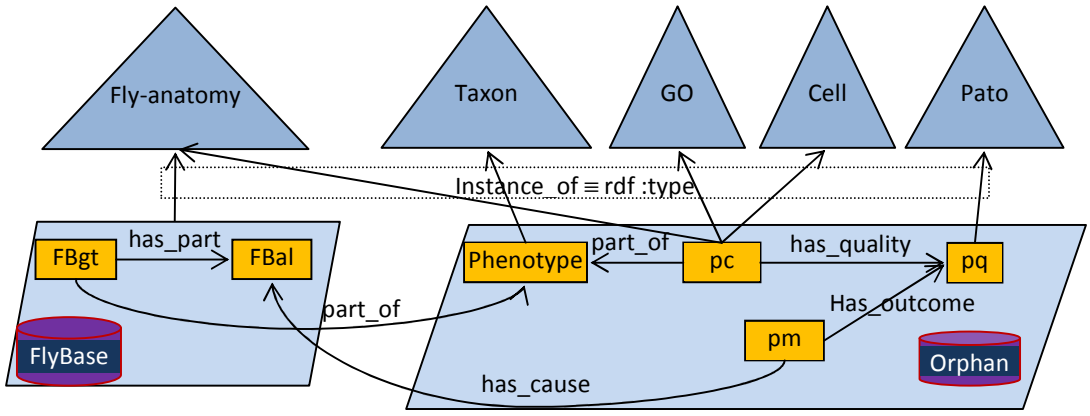


Figure 49. Liens sémantiques entre les instances de FlyBase

Le Tableau 9 illustre les trois cas qui viennent d'être présentés à travers un exemple. Nous reprenons le fichier d'annotation fly_pheno.rdf et nous regardons quelques fragments de ce fichier. Chaque ligne de ce tableau montre un cas d'exploitation possible parmi les trois cas définis dans les sections précédentes, à l'exception de la ligne 1 et de la ligne 3. Nous décrivons ci-après les différentes lignes de ce tableau :

• La ligne 1 : montre un fragment qui introduit les espaces de nom représentant les ontologies utilisées (Fly-anatomy, Taxon, GO, Cell et PATO) ainsi que les entrepôts d'instances (FlyBase et Pheno) exploités par ce fichier.

• Les lignes 3 et 4 représentent deux exemples de liens d'annotation : entre l'instance phenotype6 et le concept 7227 de l'ontologie Taxon et entre l'instance pq69 et le concept 643 de l'ontologie PATO.

Les autres lignes de ce tableau illustrent le principe des trois cas d'exploitations listés auparavant :

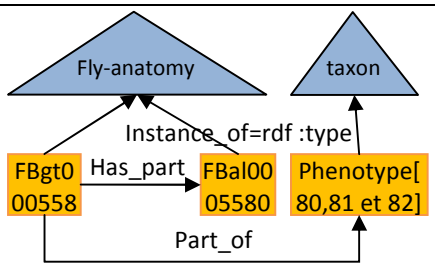
• La ligne 2 illustre les cas 1 et 2. Elle montre : (i) une relation sémantique (la relation « has_part ») entre deux ressources (FBgt005560 et FBal0005580) annotées par la même ontologie (Fly_anatomy) ; il s'agit ici du cas 1 ;ous pourrions en déduire une relation (par exemple has_part) entre les concepts de l'ontologie Fly_anatomy, à partir de la relation existant entre leurs instances. (ii) Une relation « part_of » entre une instance de Fly-anatomy (FBgt005580) et trois instances de l'ontologie taxon (phenotype80, phenotype81, phenotype82). Il s'agit du cas 2. Nous pourrions en déduire une relation entre Fly_anatomy et Taxon.

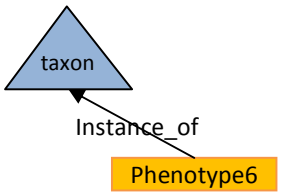
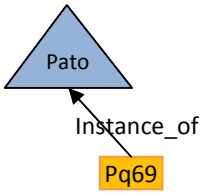
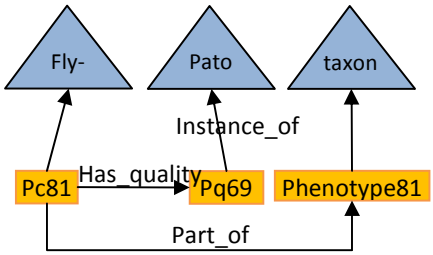
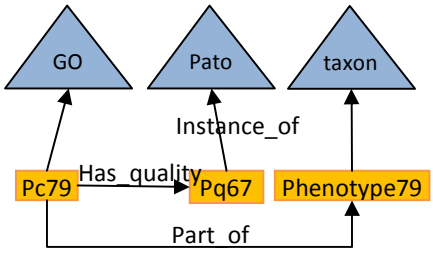
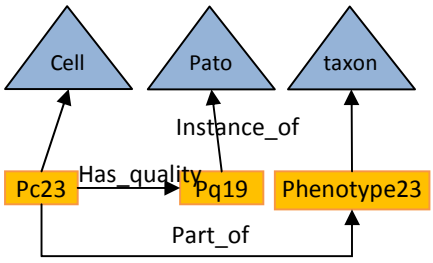
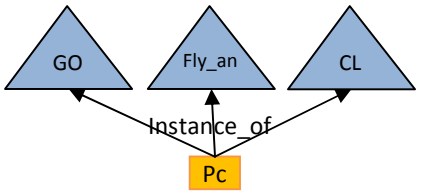
• Les lignes 5, 6 et 7 illustrent le cas 2 : il s'agit des deux ontologies qui annotent deux entrepôts de ressources différents, mais qui contiennent des relations sémantiques entre leurs ressources. Ce type de lien donne la possibilité de créer des relations sémantiques entre les concepts des deux ontologies. Par exemple :

- Entre, d'une part, Fly_anatomy et Taxon, et d'autre part, Fly_anatomy et Pato (Ligne5)
- Entre, d'une part, GO et Taxon, et d'autre part, GO et Pato (Ligne6)
- Entre, d'une part, Cell et Taxon et d'autre part, Cell et Pato (Ligne7)

• La ligne 8 illustre le cas3 : il s'agit de plusieurs ontologies qui annotent le même entrepôt d'instances. Dans notre cas, nous nous basons sur des relations intermédiaires entre les concepts des ontologies Fly_anatomy (respectivement, Cell et GO) avec les concepts des deux ontologies Taxon et Pato, pour déduire des relations entre les trois ontologies Fly_anatomy, Cell et GO.

Tableau 9. Description de l'annotation du fichier fly-pheno.rdf

N° Ligne	Fragment de balises RDF du fichier d'annotation fly_pheno.rdf	Signification	Cas d'exploitation
1	<rdf:RDF xmlns:rdf="&rdf;" xmlns:rdfs="&rdfs;" xmlns:**CL**="&CL;" xmlns:FBbt="&FBbt;" xmlns:FBdv="&FBdv;" xmlns:**FlyBase**="&FlyBase;" xmlns:**GO**="&GO;" xmlns:OBO_REL="&OBO_REL;" xmlns:OBR="&OBR;" xmlns:**PATO**="&PATO;" xmlns:Pheno="&Pheno;" xmlns:orphan="&orphan;" xmlns:**taxon**="&taxon;" >	Les espaces de noms des ontologies et des entrepôts d'instances.	
2	<OBR:genotype rdf:about="&FlyBase;FBgt0005580" OBO_REL:has_part="FlyBase:FBal0005580"> <OBO_REL:part_of rdf:resource="&orphan;phenotype80"/> <OBO_REL:part_of rdf:resource="&orphan;phenotype81"/> <OBO_REL:part_of rdf:resource="&orphan;phenotype82"/> <rdfs:label xml:lang="en"></rdfs:label> </OBR:genotype>		Cas 1, cas2

3	<rdf:Description rdf:about="&orphan;phenotype6"> <rdf:type rdf:resource="&taxon;7227"/> <rndfs:label xml:lang="en">phenotype6</rdfs:label> </rdf:Description>		
4	<rdf:Description rdf:about="&orphan;pq69"> <rdf:type rdf:resource="&PATO;0000643"/> </rdf:Description>		
5	<rdf:Description rdf:about="&orphan;pc81"> <rdf:type rdf:resource="&FBbt;00004729"/> <OBO_REL:has_quality rdf:resource="&orphan;pq69"/> <OBO_REL:part_of rdf:resource="&orphan;phenotype81"/> </rdf:Description>		Cas 2
6	<rdf:Description rdf:about="&orphan;pc79"> <rdf:type rdf:resource="&GO;0030154"/> <OBO_REL:has_quality rdf:resource="&orphan;pq67"/> <OBO_REL:part_of rdf:resource="&orphan;phenotype79"/> </rdf:Description>		Cas 2
7	<rdf:Description rdf:about="&orphan;pc23"> <rdf:type rdf:resource="&CL;0000100"/> <OBO_REL:has_quality rdf:resource="&orphan;pq19"/> <OBO_REL:part_of rdf:resource="&orphan;phenotype23"/> </rdf:Description>		Cas 2
8	À partir des lignes du cas 2 nous déduisons qu'il y a des instances qui sont annotées par les concepts des ontologies GO, CL et Fly_anatomy		Cas3

IV.4 Conclusion et contribution du système ROMIE

L'extension du système OMIE par l'utilisation des ressources pour l'alignement nous a permis les contributions suivantes :

- Nous proposons une nouvelle solution puissante d'alignement d'ontologies à base d'instances ou de ressources, utilisant les associations existantes entre les entrepôts d'instances et les ontologies. Nous décrivons plusieurs solutions de *matching* pour déterminer la similarité entre les concepts d'ontologies à base d'instances. De cette façon, la connaissance du domaine spécifique représentée par les associations (les annotations) peut être utilisée pour déterminer sémantiquement les *mappings* significatifs d'ontologies.
- Nous fournissons une solution flexible et extensible : plusieurs méthodes de comparaisons (c.-à-d. les *matchers*) des termes d'ontologies peuvent être combinées pour améliorer la qualité des résultats du *mapping*. Par exemple, la combinaison des appariements des *matchers* syntaxiques avec ceux des *matchers* à base d'instances.
- En particulier, nous présentons le mode d'exploitation des ressources dans le processus de *mapping* dans deux domaines d'étude, à savoir le domaine éducatif et le domaine biomédical.

Par rapport aux objectifs fixés pour notre étude, le système ROMIE donne des réponses aux objectifs 3 et 4. Plus précisément, il enrichit les ontologies à aligner par des liens sémantiques en se basant sur le traitement des ressources ou des instances (objectif 3) puis exploite ces relations produites dans le processus de l'appariement et le processus de filtrage (objectif 4).

Par la suite, nous présentons les résultats de nos expérimentations. Après une brève introduction sur les outils utilisés par notre prototype et une définition des métriques utilisées dans la phase d'évaluation, nous évaluons les résultats d'alignement en l'absence de ressources (c.-à-d. le système OMIE) puis en présence des ressources (c.-à-d. le système ROMIE). Dans ces deux cas, un exemple applicatif sera détaillé et une évaluation comparative sera présentée.

Chapitre V : Expérimentation et évaluation

V.1 Prototype

Afin de montrer l'intérêt de l'architecture et de la solution décrite dans les chapitres précédents, nous avons conçu un système avec deux versions : OMIE et ROMIE suivant le contexte d'utilisation. Ce système permet d'une part de valider l'architecture proposée et de montrer l'intérêt des différentes techniques et méthodes utilisées pour un *mapping* plus fiable. Il permet d'autre part aux utilisateurs de tester et de valider notre approche de *mapping* d'ontologies. La Figure 50 illustre l'architecture logicielle de notre prototype dont la réalisation a nécessité l'utilisation et la combinaison de plusieurs autres outils. L'architecture comprend principalement deux plateformes indépendantes que l'on a intégrées. Il s'agit de Jade et d'Ontobroker. Nous

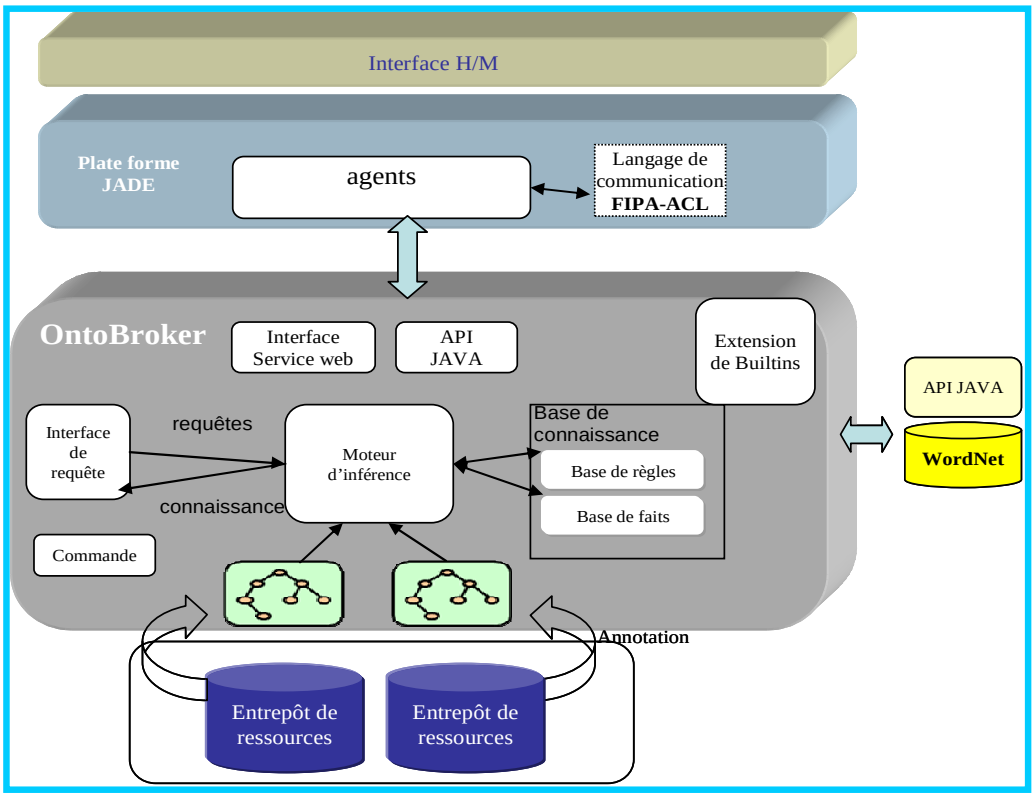


Figure 50. Architecture logicielle

décrivons dans ce qui suit le rôle de chacune de ces deux plateformes.

V.1.1 Plate-forme Jade

La plate-forme JADE (*Java Agent Development Framework*) [46] est utilisée pour l'implémentation et la gestion des agents. Les connaissances des agents sont exprimées sous forme de faits exploitables par le moteur d'inférence d'OntoBroker. Cette plate-forme d'agents s'appuie sur le langage Java et le langage de communication inter-agents FIPA-ACL [33]. Les agents développés communiquent entre eux pour échanger des résultats en FIPA-ACL, mais la plus grande partie de leur communication se fait vers Ontobroker. L'avantage de cette plateforme c'est qu'elle donne la possibilité d'ajouter à tout moment un agent de façon relativement simple. Par conséquent, rajouter un nouveau *matcher* ou une nouvelle ontologie distante revient à créer un nouvel agent (MA_i) ou (OA_i) et à décrire son comportement.

V.1.2 Serveur Ontobroker

Le serveur *OntoBroker* [77] est utilisé pour exploiter les ontologies. Il accepte plusieurs formats d'ontologies tels que RDF, OWL et F-logic, puis il transforme les différents formats d'entrée en un langage pivot « F-logic ». Les principaux composants d'OntoBroker utilisés par le prototype sont les suivants :

- Le moteur d'inférence : utilisé par les différents agents système, il est basé sur le principe du chaînage arrière[1]. Il permet la mise à jour de la base de connaissances de l'agent à partir des éléments des ontologies à aligner, des règles d'inférences et des '*builtins*'. Un exemple de règle utilisée par l'agent structurel est présenté dans l'algorithme 9 permettant de déduire une relation entre deux nœuds pères, à partir de la relation existante entre leurs fils.
- Les commandes internes ou *builtins* permettent d'améliorer considérablement l'expressivité et la polyvalence du système. Un *builtin* est un code Java externe qui définit une fonction ou un opérateur spécifique qui est utilisé ensuite par le moteur d'inférence comme une fonction interne du langage F-logic. Par exemple, les agents matchers définissent leurs méthodes de comparaison sous forme d'un *builtin* avec deux arguments en entrée (les deux éléments à comparer) et un argument en sortie (la valeur de similarité générée par la méthode de comparaison) puis le moteur d'inférence utilise ce *builtin* pour l'appliquer sur l'ensemble des éléments d'ontologies à aligner.

[1] Le principe du chaînage arrière consiste, à partir d'un but donné, à valider les prémisses de la règle qui ont permis d'aboutir à ce fait. Il effectue la même opération de manière récursive pour les prémisses trouvées, jusqu'à ce que tous les faits nécessaires pour atteindre le but soient validés.

- La Requête : est un composant qui nous permet d'écrire des requêtes exprimées en F-logic pour interroger les ontologies. Par exemple, la sélection de tous les concepts et leurs relations pour l'affichage de l'ontologie sur l'interface utilisateur.
- La Commande : est un composant que nous pouvons utiliser pour modifier l'ontologie (c-à-d. l'ajout, la modification ou la suppression d'un concept ou d'une relation). Les agents OAs utilisent des commandes pour ajouter les relations sémantiques d'enrichissement sur l'ontologie.
- L'API *WordNet* est utilisée pour la comparaison des concepts. Cette API est utilisée par l'agent MA à base de *WordNet* pour définir la fonction de son *builtin.*

Une évaluation complète des méthodes de comparaisons (c.-à-d, les *matchers*) utilisées par les deux systèmes OMIE et ROMIE a été réalisée dans les deux domaines d'applications présentés dans le chapitre II. Le domaine biomédical est décrit par deux ontologies différentes qui seront décrites un peu plus loin. Dans l'application liée à l'apprentissage enrichi par les technologies, nous utilisons deux entrepôts de ressources annotés par deux ontologies distinctes. Les objectifs que nous cherchons à montrer durant ce chapitre à travers nos expérimentations sont les suivants :

1. Définir les différentes métriques utilisées pour la validation des résultats de *mapping*.
2. Montrer la capacité d'adaptation d'OMIE et de ROMIE dans les deux contextes d'application proposés.
3. Evaluer et comparer l'influence et la précision des résultats des différents composants de notre approche de *mapping* (les *matchers* et les filtres) (partie OMIE).
4. Comparer les résultats de notre système à d'autres systèmes existants qui sont les plus utilisés pour l'alignement d'ontologies (partie OMIE). Compte tenu de l'absence de systèmes de *mapping* à base de ressources et pour rester conforme au jeu de test, nous avons comparé les résultats d'OMIE (c-à-d. sans utilisation des ressources) avec ceux des autres systèmes.
5. Etudier et évaluer l'impact d'utilisation des ressources et des instances sur l'amélioration des résultats de correspondance (partie ROMIE) .
6. Explorer l'amélioration portée par ROMIE par rapport à ce qui est produit par OMIE.

/ Le nom de la règle: FatherMatcher */*

1. RULE FatherMatcher:

/ Définition des variables de similarité */*

2. **FORALL NS1, X, NS2, Y, coef, SV**

/ Création de la similarité 'FatherMatch' entre les deux concepts 'X' et 'Y' (NS1 et NS2 représentent l'espace de nom de chaque ontologie) avec la valeur de similarité SV et la confiance 'conf'.*/*

3. similarity(FatherMatch,NS1#X,NS2#Y,coef,SV)

/ Signe qui sépare les conclusions et les prémisses*/*

4. <-

/ S'il existe un mapping entre un fils de X (FX) et un fils de Y (FY).*/*

5. EXISTS FX,FY,Cf,S,N1,N2,N Map(NS1#FX,NS2#FY,Cf,S) and

/ Définition de FX et FY en tant que fils directs respectivement de X et Y.*/*

6. DirectParent(NS1,X,FX) and DirectParent(NS2,Y,FY) and

/ Calcul du nombre de fils de chaque concept. N1 (respectivement N2) représente le nombre de fils de X (respectivement de Y).*/*

7. NbrChild(NS1#FX,N1) and NbrChild(NS2#FY,N2) and

/ On calcule le minimum N entre N1 et N2*/*

8. N is min(N1,N2) and

/ A partir de la configuration du système, cette ligne récupère le degré de confiance (CFMatch) et le seuil (ThMatch) associé à ce matcher*/.*

9. EXISTS $CF_{Matcher}$, $Th_{Matcher}$ MatcherConfig(FatherMatch,CFMatcher,Th)

/ Association de la valeur de CFMatch à la variable coef (la confiance de la similarité produite).*/*

10. coef is CFMatcher and

/ Prolongation de la valeur de similarité de mapping entre les deux fils FX et FY.*/*

11. SV is (S/N) and

/ La similarité ne peut être générée sauf si la valeur de similarité (SV) est plus grande que le seuil du matcher (ThMatcher).*/*

12. greaterorequal (SV, $Th_{Matcher}$) and

/ Vérification de l'appartenance des deux concepts X et Y à deux espaces de noms différents.*/*

13. not equal(NS1,NS2).

Algorithme 9. Exemple d'une règle d'inférence avec sa description utilisée par le *matcher* structurel

Le reste de ce chapitre est organisé comme suit. La première partie décrit l'application du système OMIE dans le contexte biomédical. La deuxième partie permet d'illustrer, à travers des expérimentations sur des ontologies pédagogiques, l'intérêt des ressources dans le processus de *mapping* pour améliorer les résultats de l'alignement d'ontologies (c.-à-d. le système ROMIE).

V.2 Expérimentation et évaluation du système

V.2.1 Les métriques utilisées pour l'évaluation

Afin de bien comprendre les résultats expérimentaux des deux systèmes OMIE et ROMIE présentées dans cette section, une définition des métriques d'évaluation est nécessaire.

Dans le domaine du *mapping* d'ontologies, les mesures de *Précision*, *Rappel*, *Fallout* et *Fmesure* [22] sont des métriques largement employées pour évaluer la qualité des alignements obtenus. L'OAEI (Ontology Alignement Evaluation Initiative) [73] retient ces mesures pour l'évaluation de la qualité de l'alignement.

L'objectif principal de ces mesures est l'automatisation du processus de comparaison des méthodes d'alignement ainsi que l'évaluation de la qualité des alignements produits. La première phase dans le processus d'évaluation de la qualité de l'alignement consiste à résoudre le problème manuellement. Le résultat obtenu manuellement est considéré comme le *mapping* de référence. La comparaison du résultat de l'alignement de référence avec celui obtenu par la méthode d'alignement produit trois ensembles : N_{found}, $N_{expected}$ *et* $N_{correct}$ (Figure 51). L'ensemble N_{found} représente les paires produites par le système. L'ensemble $N_{expected}$ désigne l'ensemble des couples appariés dans l'alignement de référence. L'ensemble $N_{correct}$ est l'intersection des deux ensembles N_{found} et $N_{expected}$. Il représente l'ensemble des paires appartenant à la fois à l'alignement obtenu et à l'alignement de référence. La *précision* est le rapport du nombre de paires pertinentes trouvées ($N_{correct}$) sur le nombre total de paires produites (N_{found}) :

- Précision = $|N_{correct}|/|N_{found}|$

Le *rappel* est le rapport du nombre de paires pertinentes trouvées ("$N_{correct}$") sur le nombre total de paires pertinentes ($N_{expected}$). Il spécifie ainsi la part des vraies correspondances trouvées. La fonction *rappel* est définie par :

- Rappel = $|N_{correct}|/|N_{expected}|$

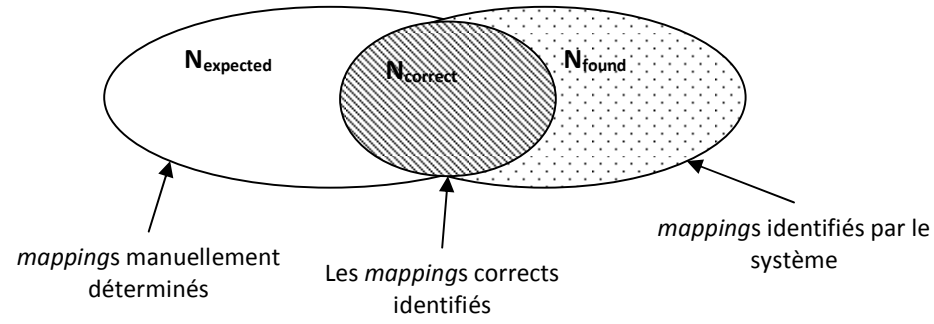


Figure 51. Présentation graphique des métriques d'évaluation

La métrique *Fallout* permet d'estimer le pourcentage d'erreurs obtenus au cours du processus d'alignement. Elle est définie par le rapport entre les correspondances non identifiées ($|N_{found}|$- $|N_{correct}|$) et le nombre total des paires trouvées ($|N_{found}|$). Cette métrique est définie par la formule suivante :

$$Fallout = \frac{|Nfound| - |Ncorrect|}{|Nfound|}$$

La métrique *Fmesure* est une moyenne harmonique. Elle combine la *précision* et le *rappel* et se définit par la formule suivante :

- $$Fmesure = \frac{2 * précision * rappel}{précision + rappel}$$

Étant donné que nous nous focalisons d'une part, sur la précision des résultats de *mapping*, c'est-à-dire connaître l'exactitude des correspondances fournies par le système, et d'autre part, sur le nombre des correspondances correctes non détectées par le système, nous utilisons ainsi principalement dans nos expérimentations les deux métriques précision et rappel.

V.2.2 Evaluation du système OMIE

V.2.2.1 Introduction

Nous avons choisi d'évaluer OMIE sur des ontologies du domaine biomédical disposant de plusieurs ontologies accessibles librement. La plupart de ces ontologies sont disponibles sous le format OBO, et nous les trouvons sur le site Web d'OBO [75]. Le système OMIE offre la possibilité de traduire le format OBO au format OWL à l'aide d'un transformateur XSLT. Mais l'un des problèmes des ontologies OBO est que les concepts sont

décrits par des identifiants (par exemple MA_00003, MA_00005, etc.), et le sens du concept est décrit par la propriété label (par exemple, le label du concept MA_00003 est : « système d'organe »). Ceci rend le *mapping* impossible puisque tous les concepts sont différents si nous comparons juste leurs identifiants (URI). En outre, et contrairement à l'éditeur d'ontologie Protégé [72] [70] et le système de *mapping* Prompt [70] qui éditent et font correspondre respectivement les concepts suivant leurs identifiants (URI), OMIE est capable d'éditer et de faire correspondre à la fois l'identifiant et le label du concept (Figure 52). De plus, il offre aux utilisateurs la possibilité de faire correspondre un ou plusieurs concepts de l'ontologie au lieu de la faire correspondre en entier.

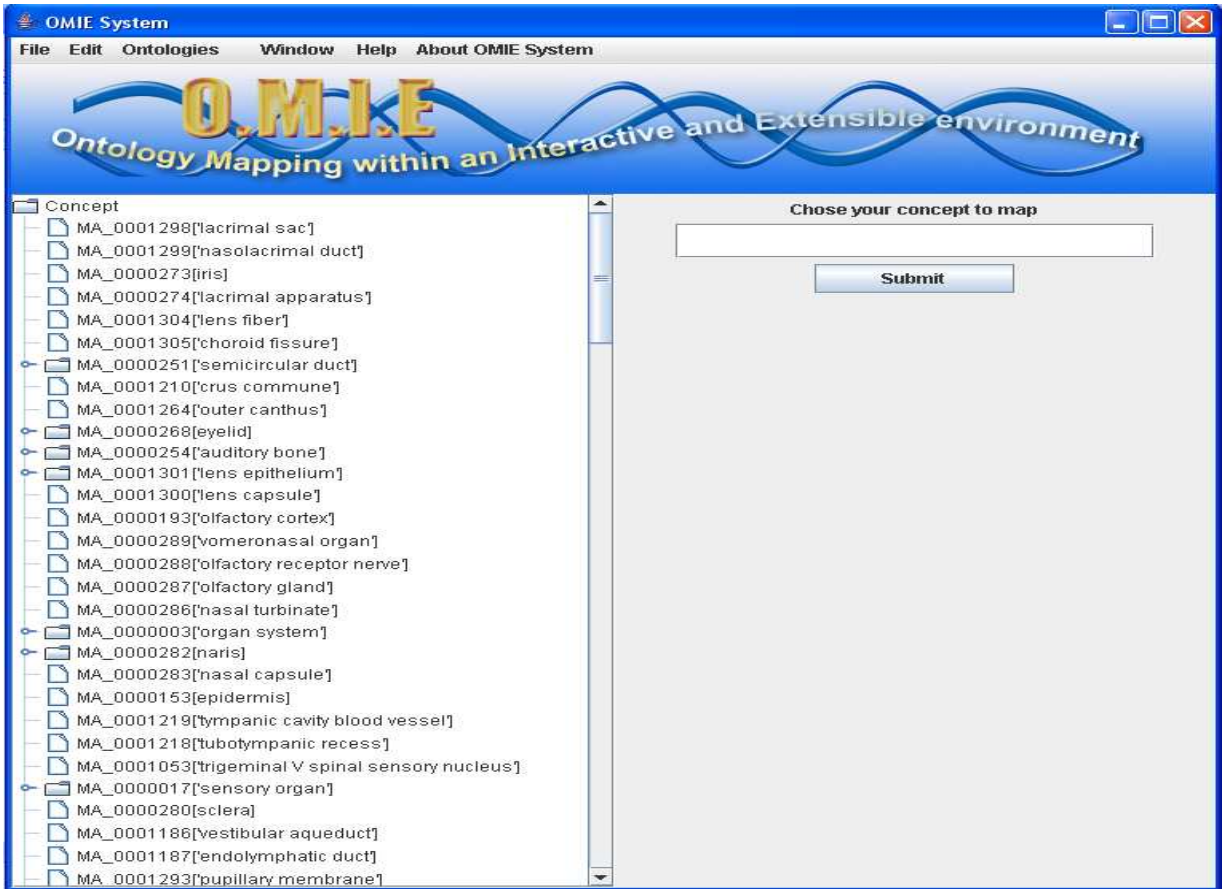


Figure 52. Edition d'ontologie par OMIE

OMIE offre à l'utilisateur la possibilité de choisir son ontologie locale et l'ontologie cible. Il lui permet également d'activer, désactiver, adapter et configurer les valeurs des différents *matchers* et filtres utilisés, tels que les poids de confiance ou les seuils (Figure 53).

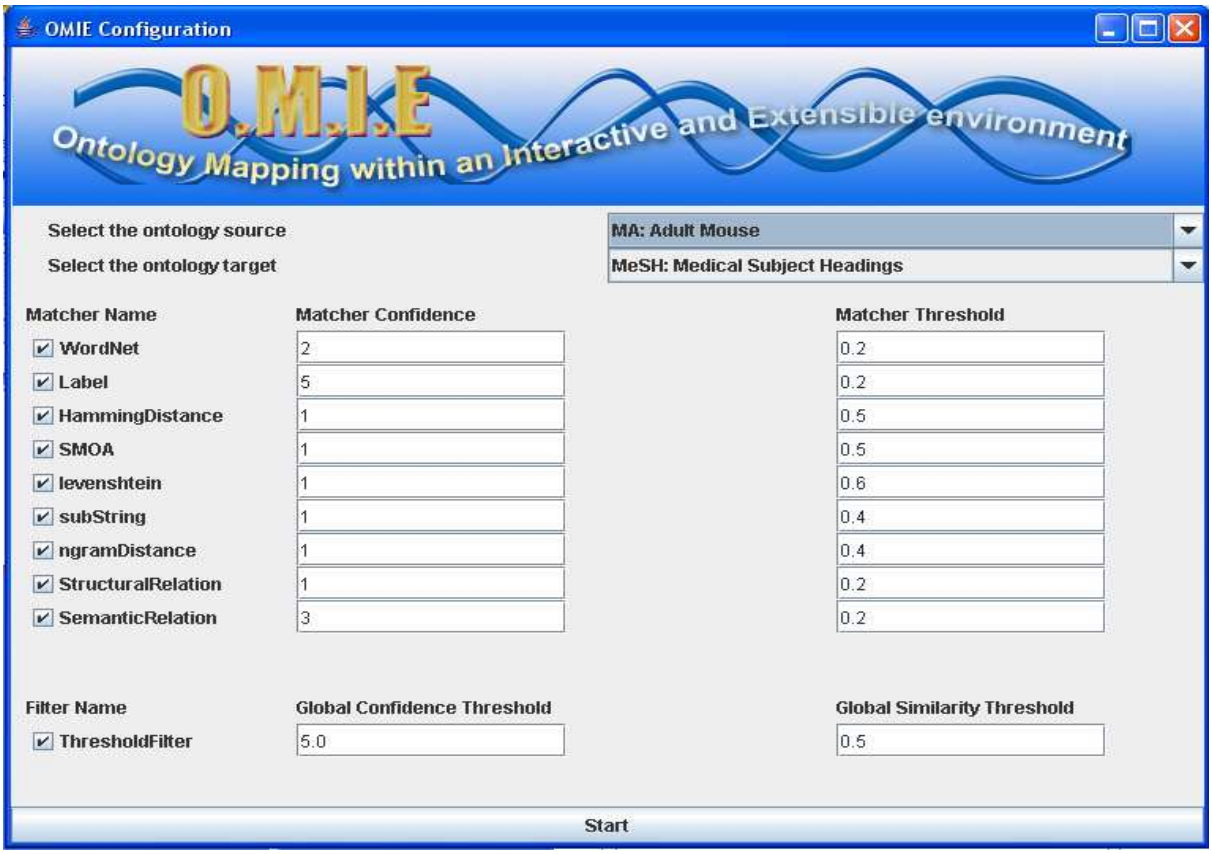


Figure 53. Interface de configuration du système OMIE

V.2.2.2 Scénarii d'expérimentations

Nous avons testé OMIE sur plusieurs ontologies OBO. Nous présentons dans cette section les résultats de *mapping* obtenus entre les deux ontologies MA (Mouse Anatomy) et MeSH (Medical Subject Headings [67]. Ces ontologies couvrent toutes les deux le même contexte (le contexte d'anatomie) mais elles sont développées séparément.

MeSH est un vocabulaire produit par la bibliothèque nationale américaine de médecine (NLM : *US-National Library of Medicine*). Elle est employée pour l'indexation et la recherche d'information des documents biomédicaux relatifs à la santé. Elle se compose d'un ensemble d'entités et de descripteurs structurés d'une manière hiérarchisée et elle contient plus de 1400 concepts et 60000 relations de synonymie.

En revanche, l'ontologie MA organise une structure hiérarchique de l'anatomie de la souris en utilisant les deux relations : « is-a » et « part-of ». Cette ontologie est employée pour décrire des données de l'anatomie de la souris d'une manière normalisée. Elle contient plus de 2400 concepts.

Pour notre expérimentation, nous nous sommes focalisés sur trois catégories développées par les deux ontologies :

1. Catégorie du nez, avec 15 concepts de l'ontologie MeSH et 18 concepts de l'ontologie MA qui développent l'anatomie du nez.

2. Catégorie de l'oreille, avec 39 concepts de l'ontologie MeSH et 77 concepts de l'ontologie MA qui développent l'anatomie de l'oreille.

3. Catégorie de l'œil, avec 45 concepts de l'ontologie MeSH et 112 concepts de l'ontologie MA qui développent l'anatomie de l'œil.

Dans notre expérimentation, nous avons évalué deux cas de test. Dans le premier cas, nous comparons les résultats obtenus par différents *matchers* pris individuellement, ainsi que leurs combinaisons. En outre, nous examinons l'impact des différents filtres pour améliorer le résultat de *mapping*. Dans le deuxième cas, nous comparons les résultats obtenus avec notre système OMIE avec d'autres systèmes existants dans la littérature. La première phase dans le processus d'évaluation de la qualité de l'alignement consiste à résoudre le problème manuellement. Dans notre cas, l'expert a pu découvrir, selon les trois catégories d'expérimentation, le nombre de liens de correspondance suivants :

- Neuf (9) relations de *mapping* entre les concepts de MA et de MESH développant l'anatomie du nez.

- Vingt-sept (27) relations de *mapping* entre les concepts de MA et de MESH développant l'anatomie de l'oreille.

- Vingt-sept (27) relations de *mapping* entre les concepts de MA et de MESH développant l'anatomie de l'œil;

V.2.2.3 Résultats d'expérimentation

La Figure 54 montre le rappel et la précision obtenus par OMIE tout au long du processus. On voit que les *matchers* seuls produisent un faible rappel et une faible précision (sauf les *matchers* sémantiques). Ces *matchers* sont complémentaires car leurs combinaisons donnent un bon rappel mais une précision faible (beaucoup de faux *mappings* générés). Le filtre à base de seuil choisit parmi les *mappings* candidats, ceux qui disposent d'une valeur de confiance et de similarité plus grande que le seuil prédéfini. Nous avons expérimenté notre système

avec un seuil de ***confiance égal à 5*** et comme ***similarité minimale la valeur de 0,5***. Malgré cela, le filtre à base de seuil améliore faiblement la précision. Sur le diagramme, les filtres structurels et sémantiques ont eu un effet beaucoup plus positif pour la détection de ce genre d'anomalies. En effet, ils éliminent certaines fausses propositions de *mapping* même si leurs valeurs de similarité sont importantes.

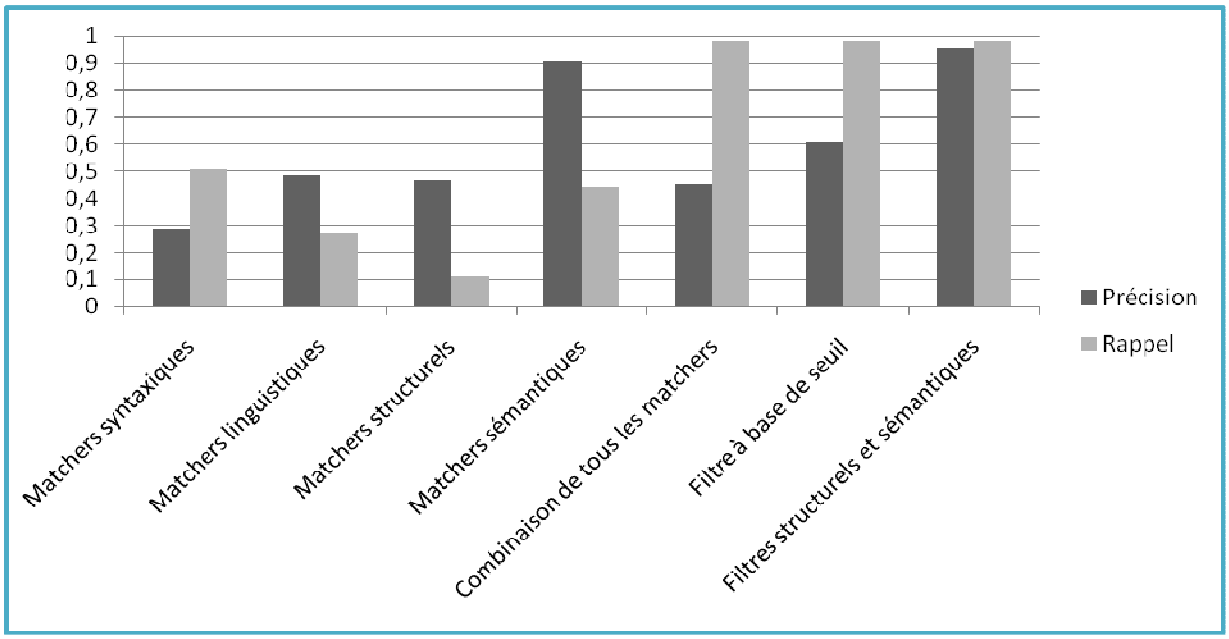


Figure 54. Évaluation des méthodes de comparaison (matchers) utilisés par le système OMIE

En conclusion, ces expérimentations valident bien l'approche proposée : OMIE augmente le rappel sans détériorer la précision. Les combinaisons de *matchers* de différents types sont efficaces pour trouver un maximum de bons *mappings,* mais en génèrent également beaucoup de faux *mappings*. Il faut donc un processus de filtrage efficace que ne réalisent pas les filtres à base de seuil. Par contre, le filtrage structurel et sémantique offre des résultats bien meilleurs.

Nous avons également comparé notre système OMIE avec trois outils de *mapping* existants : PROMPT, FOAM [27] et SAMBO [52], selon les mêmes métriques employées auparavant (c.-à-d. précision et rappel) (Figure 55).

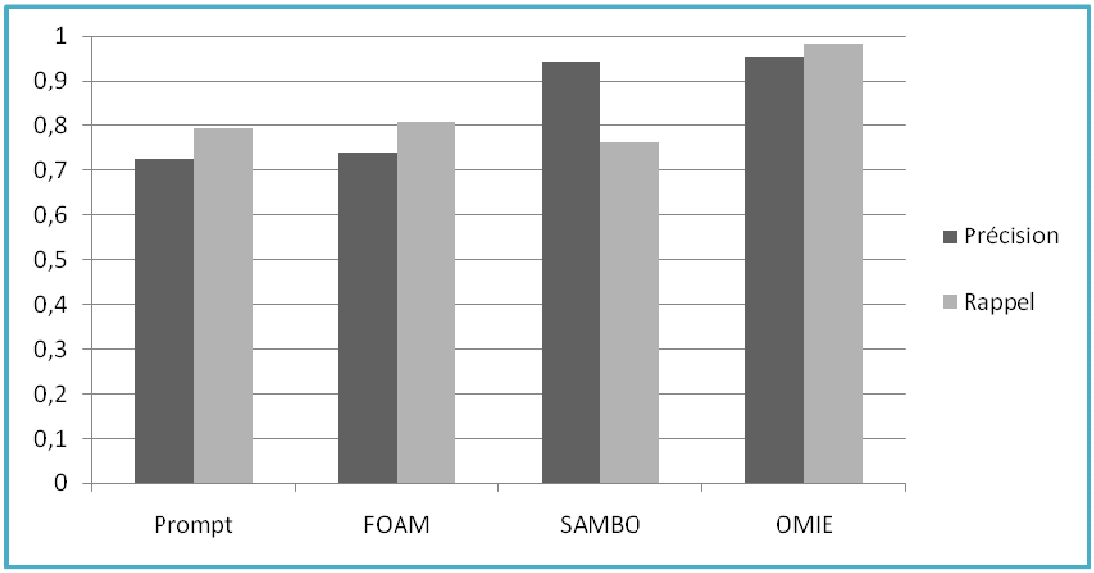


Figure 55. Comparaison d'OMIE avec d'autres systèmes existants

PROMPT et FOAM sont deux systèmes d'alignement et de fusion d'ontologies semi-automatiques. Ils proposent un ensemble de suggestions d'alignement que l'utilisateur peut accepter ou rejeter. Cependant, ces deux systèmes ne permettent aucune exploitation ou traitement de ces *mappings* confirmés. Par ailleurs, l'une des particularités d'OMIE est l'utilisation de cette liste de *mappings* admise pour améliorer les résultats des autres propositions ou pour déduire d'autres relations de correspondance. De plus, les trois systèmes comparent les concepts des deux ontologies MA et MeSH suivant leurs identifiants. Une transformation des deux ontologies (pour remplacer les URIs par les labels) a donc été obligatoire pour obtenir des liens de correspondance.

Nous constatons sur la Figure 55 que la précision d'OMIE est plus élevée que celles de PROMPT et de FOAM et semblable à celle de SAMBO. En revanche, la valeur de rappel d'OMIE est plus importante que les autres systèmes. Le résultat global de cette expérimentation est très bon. Par exemple, OMIE a identifié toutes les relations de *mapping* fournies manuellement par l'expert du domaine à part une seule relation parmi les vingt-sept relations produites entre les concepts développant l'anatomie de l'œil. Bien entendu, ces résultats demandent à être confirmés sur d'autres types d'ontologies et aussi sur des ontologies plus importantes en nombre de concepts. Il serait intéressant d'évaluer le comportement d'OMIE sur des ontologies présentant peu de ressemblances.

V.2.3 Evaluation du système ROMIE

Nous avons testé ROMIE sur deux ontologies du domaine pédagogique pour lesquelles nous disposons des ressources.

V.2.3.1 Fonctionnalités de ROMIE

Avant d'évaluer le système ROMIE, nous illustrons quelques fonctionnalités principales, notamment l'interopérabilité entre deux ontologies, l'une locale (notée : LocOnto), et l'autre distante (notée : DistOnto), et ceci à travers des scénarios d'utilisation. Nous décrivons ci-dessous les deux scénarios :

- Scénario 1 : un apprenant (ou un utilisateur) utilise l'ontologie LocOnto comme un support pour accéder aux ressources pédagogiques (par exemple, les cours et les exercices d'un module informatique), en formulant des requêtes qui portent sur les concepts de l'ontologie (par exemple, « quels sont les ressources qui portent sur le concept *base de donnée* »). Cependant, il arrive que l'utilisateur ne soit pas satisfait par la ressource fournie par LocOnto ou que la requête ne renvoie aucune ressource. Dans ce cas, le système ROMIE intervient pour rechercher la ressource la plus pertinente par rapport à la requête de l'apprenant sur des entrepôts distants représentés par l'ontologie DistOnto. ROMIE envoie une demande à DistOnto en demandant les ressources qui développent le même concept de la requête (c.-à-d. un concept de LocOnto).

- Scénario 2 : un auteur utilise LocOnto pour construire son cours à partir des ressources développant le même concept, qu'elles soient locales ou distantes.

Nous listons ci-après quelques exemples de requêtes d'un utilisateur qui cherche à accéder à des ressources pédagogiques :

- « Donnez-moi toutes les ressources disponibles qui développent le concept 'c' » ; ou encore
- « Donnez-moi toutes les ressources de type 'K' (en utilisant le standard LOM qui décrit les méta-données des ressources) qui développent le concept 'c'».

D'autres types de requêtes permettent de rechercher des concepts en exploitant plusieurs types de relations ou la topologie de l'ontologie. Par exemple :

- « Donnez-moi les parents (ou les fils) directs du concept 'c' » ;
- « Donnez-moi tous les ascendants (ou les descendants) du concept 'c' » ;
- « Donnez-moi tous les concepts liés au concept 'c' par la relation 'r' ».

Nous avons testé notre système ROMIE sur deux entrepôts éducatifs indexés par des ontologies du domaine éducatif, à savoir : Simbad [17] et ACM [1]. L'ontologie Simbad a été développée au sein de notre équipe ; elle annote les documents éducatifs (par exemple, les cours, les exercices, etc.). L'ontologie ACM/CCS est une classification du domaine de l'informatique. Elle contient neuf sous-domaines principaux organisés en sections.

Afin de faciliter la création d'un modèle SIMBAD, une application Web a été créé et permet à l'utilisateur à travers des interfaces graphiques de manipuler l'ontologie et ses ressources. Nous listons ci-dessous quelques fonctionnalités de cette application :

- Génération d'une ontologie en format F-logic à partir de son format initial (RDF ou OWL).
- Mise à jour de l'ontologie : ajout, modification et suppression des concepts et ces relations.
- Annotation d'une ressource par un ou plusieurs concepts de l'ontologie.
- Création des relations entre les ressources de même entrepôt.
- Création des propriétés des ressources. Il s'agit de créer des liens autres que les liens d'annotation entre la ressource et les concepts. Par exemple la relation de 'Pré-requis'.
- Création des ressources composées. Il s'agit de générer une ressource à partir d'autres ressources existantes.
- Etc.

V.2.3.2 Expérimentations

Dans nos tests, nous considérons une partie du modèle de l'informatique décrit dans Simbad, soit 30 concepts annotant 120 ressources (décrites par leurs métadonnées) en tant qu'ontologie locale (LocOnto) et une partie de l'ontologie ACM comme ontologie distante (DistOnto). La partie prise d'ACM décrit deux sections (la section logicielle et la section du système informatique) et annote 100 ressources.

Initialement et dans la première phase du *mapping*, les deux ontologies (LocOnto et DistOnto) ont été automatiquement enrichies grâce aux relations sémantiques entre les ressources des deux entrepôts. Par la suite, ROMIE propose deux possibilités à l'apprenant pour faire correspondre son ontologie locale avec l'ontologie distante :

- L'apprenant peut aligner directement ces deux ontologies. Il s'agit d'un « *mapping* total » et est noté TM.

- L'apprenant peut aligner son ontologie locale pas à pas (ou à la demande) d'une manière transparente. Dans ce cas, l'objectif de l'apprenant sera l'accès à une ou plusieurs ressources associées à un ou plusieurs concepts de l'ontologie LocOnto, donc le processus de *mapping* n'est en fait déclenché que si les ressources dans l'entrepôt local ne répondent pas à la demande de l'utilisateur. Ce type de *mapping* est appelé « *mapping* partiel » ou « *mapping* à la demande » et on le note PM.

Nous avons restreint nos expérimentations au scénario « *mapping* total ». En effet, le *mapping* partiel est très difficile à expérimenter pour les raisons suivantes :

- (i) L'expérimentation du *mapping* partiel (c.-à-d. le *mapping* par un sous-ensemble de concepts) est très délicate à réaliser. Un tel mode nécessite un nombre important d'utilisateurs et d'apprenants avec une grande interaction avec le système.
- (ii) En outre, la comparaison avec l'existant est impossible compte tenu de l'absence des systèmes de générations des *mappings* partiels, c'est-à-dire la génération des correspondances d'une partie de l'ontologie.
- (iii) Enfin, d'après la conception du système et quelques expérimentations que nous avons tenté de réaliser en mode partiel, nous avons constaté que nous obtenons au minimum des résultats semblables ou meilleurs que ceux du *mapping* global.

Nous avons réalisé plusieurs essais afin d'évaluer la performance du système ROMIE en produisant des *mappings* entre les ontologies ACM et Simbad.

- Dans la première étape d'évaluation (Figure 56), nous analysons l'impact des *matchers* sémantiques à base de ressources sur les résultats du *mapping*. Afin d'aboutir à cet objectif, nous avons réalisé trois expérimentations. Dans la première, seuls les *matchers* linguistiques et syntaxiques sont appliqués. Dans la deuxième expérimentation, des *matchers* structuraux ont été activés pour produire plus de *mappings* alors que dans la dernière expérimentation, nous avons introduit les *matchers* sémantiques. Chaque expérimentation est présentée séparément afin de montrer l'importance et l'impact de chacune des méthodes utilisées dans le système ROMIE pour améliorer les résultats de *mapping*.

- Dans la deuxième étape d'évaluation (Figure 57), nous analysons l'impact des filtres basés sur des ressources, afin de réduire le nombre de résultats de *mapping* erronés obtenus dans la première

étape par la combinaison de tous les *matchers*. Au début, nous avons appliqué seulement les filtres structuraux, puis nous avons utilisé secondairement les filtres sémantiques. Ces derniers utilisent des relations sémantiques produites pendant la phase d'enrichissement de l'ontologie.

Nous comparons et évaluons les résultats des différents tests obtenus par ROMIE, à travers les trois métriques suivantes : (i) le pourcentage de véritables *mappings* positifs, c'est-à-dire les vrais *mappings* identifiés par le système ; (ii) le pourcentage de véritables *mappings* négatifs, autrement dit, les vrais *mappings* non identifiés et enfin (iii) le pourcentage de faux *mapping* identifiés.

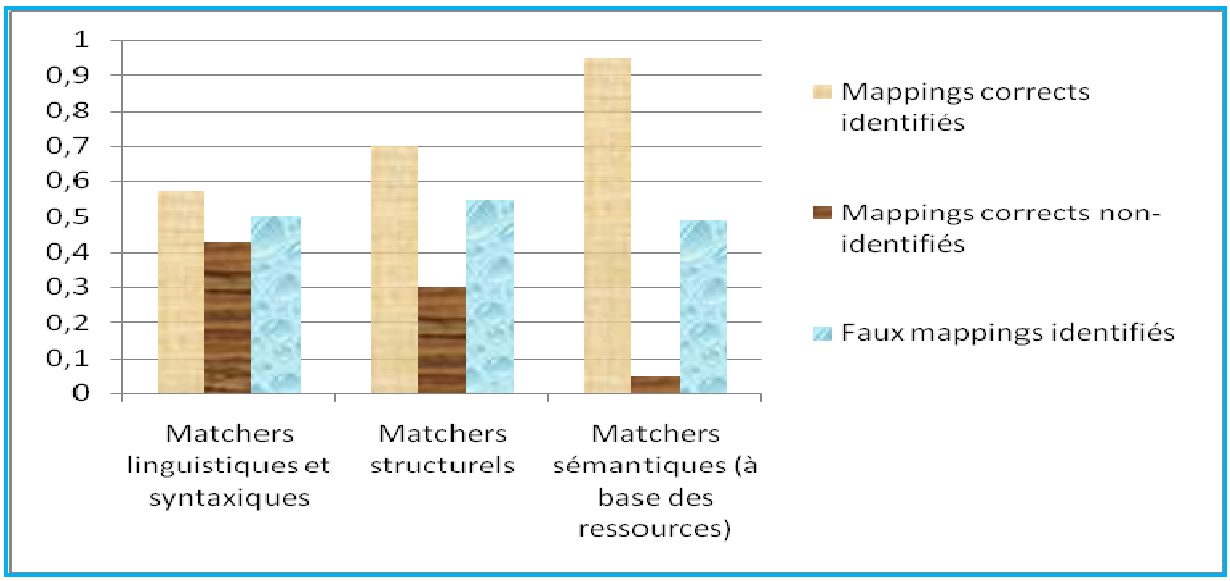


Figure 56. Évaluation des *matchers* du système ROMIE

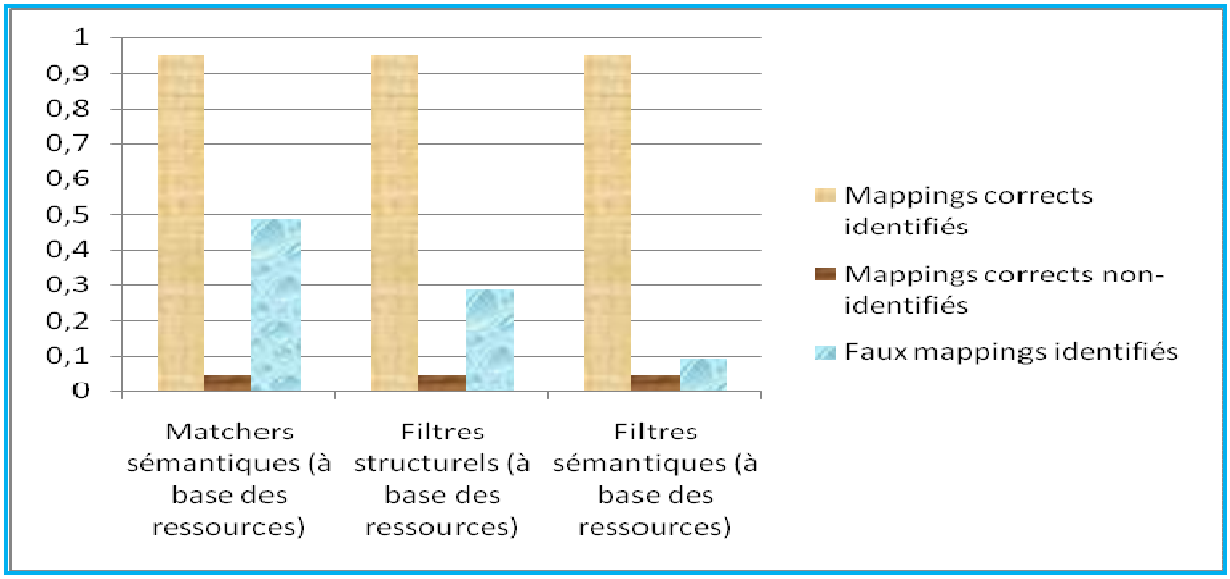


Figure 57. Évaluation des filtres du système ROMIE

Les résultats des *mappings* sont améliorés grâce à la succession et la combinaison des méthodes de *matching* et de filtrage. Nous constatons sur les deux figures précédentes que les *matchers* linguistiques et syntaxiques ont identifié plus de 50% des *mappings* corrects. Cependant, la moitié des résultats étaient faux. En outre, en appliquant les *matchers* structuraux, nous atteignons la valeur de 70% des vrais *mappings* identifiés mais avec plus de faux *mappings*. Le *matcher* sémantique à base des ressources a permis d'obtenir toutes les relations de *mapping* existant entre les deux ontologies (95%), avec un taux de résultats faux comparable aux *matchers* précédents. ROMIE améliore donc le rappel d'OMIE qui était déjà bon. Néanmoins, le nombre des faux *mappings* reste toujours très élevé, d'où l'utilité des filtres structuraux et sémantiques qui diminuent considérablement le nombre de faux *mappings*, passant de 50 à environ 10% sur tout le nombre de *mappings* obtenus. ROMIE arrive donc à améliorer le rappel d'OMIE sans détériorer la précision.

Le système ROMIE est une extension du système OMIE. Concrètement, ROMIE ajoute les points suivants par rapport à OMIE :

1. Le prétraitement : contrairement au système OMIE, avant le lancement du processus de *mapping*, ROMIE enrichie les ontologies à aligner par les relations sémantiques.
2. Le temps d'exécution : à l'exception du temps nécessaire pour la phase de prétraitement et qui varie entre 5 à 15 secondes suivant la taille de l'entrepôt de ressources, le temps nécessaire pour la génération des correspondances est similaire entre les deux systèmes.

V.3 Discussion et conclusion

Ce chapitre a été consacré à la mise en œuvre et la validation de notre approche. Nous avons commencé par la présentation de l'architecture logicielle de notre prototype, ainsi qu'une définition des métriques de comparaison les plus utilisées pour évaluer les résultats de *mapping*.

Ensuite nous avons fourni une évaluation expérimentale pour le *mapping* des ontologies réelles, à savoir : (i) les deux ontologies biomédicales MESH et MA (d'autres expérimentations effectuées sur les ontologies MPath, GO et Fly_anatomy ne sont pas présentées dans ce rapport, car les résultats obtenus sont semblables à ceux de MESH et MA), ou encore (ii) les ontologies éducatives à savoir SIMBAD et ACM. Pour ces dernières, des ressources sont attachées à leurs concepts.

Nous avons exploité les associations existantes entre les concepts de chaque ontologie ainsi que les associations déduites qui prennent en considération le sens et les caractéristiques des ressources. Nous avons fourni également des méthodes de comparaison (des *matchers*) à base des métadonnées d'ontologie (les termes des concepts et la structure hiérarchique d'ontologie) ainsi que des méthodes à base d'informations auxiliaires telles que le dictionnaire WordNet.

Cependant, le comportement de toutes ces méthodes de comparaison est défini par des règles d'inférence qui nous a permis d'automatiser, d'accélérer et d'améliorer la coordination inter-*matchers*. Nous avons défini ces règles en utilisant le moteur d'inférence F-logic d'OntoBroker pour accomplir cet objectif.

L'évaluation des tous les éléments mis en place pour le *mapping* ainsi que leurs combinaisons face aux systèmes existants est exprimée à l'aide des deux métriques : le rappel et la précision.

Au terme des objectifs listés dans le chapitre d'introduction, l'évaluation et l'expérimentation de notre système ont permis de valider les contributions suivantes :

- **Contribution 1 :** Le problème d'hétérogénéité du langage d'ontologie
  - Quelque soit le type d'ontologies ('rdf', 'owl' et ' F-logic'), les agents OAs les chargent sur le serveur OntoBroker et les utilisent par la suite comme une ontologie F-logic. Par conséquent, tous les autres agents utilisent des requêtes et des règles d'inférence en F-logic.
  - Dans des cas particuliers, quelques ontologies biomédicales ne sont disponibles qu'au format 'obo' (ce format n'est pas pris en charge par l'OntoBroker). Afin de pallier à cette insuffisance, nous avons développé un translateur XSLT pour transformer le format 'obo' en format 'owl'.

- **Contribution 2 :** Les modes partiel et total de *mapping*
    - OMIE et ROMIE sont capables de fonctionner en mode de *mapping* total (comme tous les autres systèmes) mais aussi en mode partiel où les *mappings* sont générés à la demande si besoin est (ce qui est plus efficace dans le cas où certaines parties d'ontologies ne sont en fait pas utilisées pour annoter).
- **Contribution 3** : Les méthodes de comparaison (les *matchers*)
    - Les *matchers* utilisés ainsi que leurs combinaisons ont permis d'identifier la majorité des liens de correspondance entre les ontologies à aligner. En outre, et avec le système ROMIE, la fiabilité des résultats de *mapping* reste indépendante de la richesse sémantique de l'ontologie. Autrement dit, la seule exigence du système ROMIE est l'existence de ressources qui nous permettent (le cas échéant) de pallier à l'insuffisance des relations sémantiques au sein des ontologies à aligner.
- **Contribution 4** : Les méthodes de filtrage
    - En revanche, la combinaison de plusieurs *matchers* engendre un nombre important de fausses relations de correspondance, comme c'est le cas dans plusieurs systèmes existants de *mapping*,. L'expérimentation nous a montré l'insuffisance des méthodes de filtrage à base de seuil (principalement utilisé dans la majorité des systèmes). Néanmoins, l'utilisation des liens hiérarchiques et sémantiques (qu'ils soient générés ou existants) entre les concepts de chaque ontologie améliore considérablement la phase de filtrage et par conséquent augmente la valeur de la précision des *mappings* produits.
- **Contribution 5** : Le processus de validation et d'interaction
    - Les résultats illustrés dans ce chapitre représentent les correspondances obtenues par le mode de *mapping* global en interaction avec un simple utilisateur (en l'occurrence nous même). Ainsi, nous n'avons confirmé que les *mappings* dont nous sommes sûrs. L'impact de nos validations (confirmation ou infirmation des *mappings*) était généralement positif sur la génération de nouvelles relations de *mapping* ou encore sur la réduction du nombre de mauvaises correspondances.
    - Enfin, de la même façon que le mode partiel de *mapping* et pour les mêmes raisons, nous n'avons pas pu approfondir nos tests pour les validations indirectes, autrement dit, faire introduire les ressources associées aux éléments d'ontologies dans les requêtes et les réponses,

puis extraire la validation des relations de *mapping* à partir de la satisfaction de l'utilisateur par la ressource qui lui a été proposée.

Conclusion et perspectives

Les travaux menés dans cette thèse se situent dans le domaine de l'ingénierie des connaissances et du Web sémantique. Notre objectif a été de tirer profit des travaux menés notamment dans le domaine de l'interopérabilité sémantique des connaissances, dans le but d'aligner des ontologies et également d'accéder à des ressources distantes de façon transparente.

Le résultat de notre travail est une méthode originale d'alignement d'ontologies dénommée ROMIE (Resource based Ontology Mapping within an Interactive and Extensible environment). L'originalité de notre méthode réside dans la combinaison des caractéristiques suivantes :

- Extensibilité : l'ajout, la suppression ainsi que la modification de la configuration initiale (les matchers et les seuils) du système lui permet de s'adapter aux particularités du domaine d'application.

- Adaptabilité : la découverte de l'alignement s'appuie principalement sur les relations entre les ressources ou contenus indexés sur les ontologies (instances, documents, etc...). Nul besoin d'avoir une ontologie riche à la base, il suffit d'avoir suffisamment de contenu indexé pour que la méthode puisse s'appliquer.

- Flexibilité: plusieurs méthodes d'appariement (syntaxique, linguistique, structurelle et sémantique) sont utilisées et combinées.

- Evolutivité : les mappings ne sont pas forcément générés en une seule fois, mais à la demande et de façon transparente pour l'utilisateur.

- Interactivité : le processus de validation des *mappings* utilise et exploite les connaissances des utilisateurs.

Son caractère extensible lui confère un côté relativement générique de par sa faible dépendance vis-à-vis de la sémantique du langage de représentation utilisé.

L'adaptabilité lui permet de s'appliquer à n'importe quelle ontologie (en OWL ou en OBO) de préférence riche en contenus (instances, ressources).

Son caractère flexible lui permet de produire des alignements plus riches en termes de sémantique que les méthodes d'alignement basées seulement sur des mesures de similarités. Il peut en effet accepter d'autres méthodes d'appariement.

Son caractère évolutif dans la découverte des mappings permet de calculer les correspondances uniquement au besoin. L'utilisateur désirant accéder à une ressource exprime sa requête avec les termes de son ontologie de référence. Si cette ressource ne se trouve pas dans ce référentiel, le système génère une correspondance avec les termes de l'ontologie distante pour répondre à la requête. Cette opération reste transparente pour l'utilisateur.

Enfin, son côté interactif permet de valider les mappings trouvés et de prendre en considération les retours des utilisateurs pour les améliorer.

En résumé, nos contributions nous permettent de répondre aux objectifs initialement fixés :

- **Objectif 1 : faciliter la prise en compte de l'hétérogénéité des ontologies**

Toutes les ontologies traitées sont représentées dans le formalisme F-logic, ce qui nous a amené à développer le cas échéant des traducteurs vers ce format (par exemple traducteur OBO–F-logic dans le cas des ontologies du domaine bio-médical).

- **Objectif 2 : *mapping* partiel ou total**

La couverture des ontologies est très rarement la même et n'est pas complète. D'où la nécessité d'offrir aux utilisateurs un système de *mapping* capable d'identifier les correspondances de ***tout*** ou ***partie*** des éléments de l'ontologie, c'est-à-dire l'intégration ou le ***mapping partiel*.** Nous proposons un processus de mapping à deux modes de fonctionnement : mapping sur deux ontologies complètes ou bien mapping à partir d'une requête de filtrage sur une ontologie source vers une ontologie cible.

- **Objectif 3 : améliorer le nombre de mappings trouvés**

Nous proposons plusieurs mécanismes pour améliorer le nombre des mappings trouvés. L'extensibilité d'OMIE nous permet d'ajouter à la demande de nouveaux *matchers* spécialisés qui vont être capable de prendre en

compte les spécificités de chaque ontologie. Ces *matchers* peuvent de plus être combinés pour être encore plus efficaces. ROMIE va encore plus loin en étant capable de prendre en compte l'information portée par les instances associées aux ontologies dans le processus de génération de *mappings*.

- **Objectif 4 : diminuer le nombre de faux mappings générés**

Dans le cadre de l'automatisation du processus de *mapping,* nous souhaitons aider l'utilisateur en réduisant le nombre de faux résultats, en faisant mieux que le filtrage à base de seuil qui est la méthode adoptée par la plupart des travaux existants. Pour cela, nous exploitons les ***liens hiérarchiques et sémantiques*** existants entre les concepts de chaque ontologie dans le ***processus de filtrage,*** pour détecter certaines anomalies et contradictions parmi les résultats de *mapping* obtenus. ROMIE va dans la même direction en exploitant également les liens produits à partir de l'étape d'enrichissement sémantique via les instances.

- **Objectif 5 : faciliter le rôle de l'expert dans l'étape de validation**

La phase de validation et d'interaction avec l'utilisateur est une phase clé dans le processus de *mapping,* mais est très complexe. Nous proposons de prendre en compte non seulement les retours explicites de l'utilisateur/expert sur les résultats produits par notre système, mais également les retours implicites lors d'un mapping partiel. En effet, lors d'un mapping partiel, l'utilisateur spécifie une requête et va implicitement valider le mapping réalisé s'il accède aux ressources renvoyées (cas de validation indirecte).

L'implantation d'OMIE sous forme d'un système multi-agents est naturelle et permet notamment de bénéficier des avantages de ces systèmes en termes d'extensibilité. La coopération entre les différentes étapes est également facilitée.

Les expérimentations menées sur OMIE montrent que le rappel est augmenté sans détérioration de la précision. Les combinaisons de *matchers* de différents types sont efficaces pour trouver un maximum de bons *mappings,* mais en génèrent également beaucoup de faux. Il faut donc un processus de filtrage efficace que ne réalisent pas les filtres à base de seuil. Par contre, le filtrage structurel et sémantique offre des résultats bien meilleurs.

Les expérimentations montrent que ROMIE améliore le rappel d'OMIE qui était déjà bon. Néanmoins, le nombre des faux *mappings* reste toujours très élevé, d'où l'utilité des filtres structuraux et sémantiques qui diminuent considérablement le nombre de faux *mappings*, passant de 50 à environ 10% sur tout le nombre de *mappings* obtenus. ROMIE arrive donc à améliorer le rappel d'OMIE sans détériorer la précision.

Futures directions

Les sections suivantes décrivent quelques futures directions possibles liées aux résultats présentés dans cette thèse.

Passage à l'échelle

Les expérimentations menées jusqu'à présent ont porté sur des ontologies de taille relativement modeste. Il est important d'évaluer nos propositions sur des ontologies de tailles plus réalistes (plusieurs centaines de concepts).

Intégration de nouveaux *matchers*

Le caractère extensible de notre approche fait que nous pouvons ajouter de nouvelles méthodes de calcul de similarité à tout moment. Nous avons commencé à regarder la possibilité d'intégrer une méthode basée sur la fouille de texte élaborée au sein du laboratoire IBISC. Cette méthode a l'avantage de pouvoir extraire des informations à partir du contenu des ressources. Ceci va contribuer à automatiser le processus d'enrichissement.

Faciliter le paramétrage du système

L'hétérogénéité et la nature distribuées de la modélisation des ontologies dans la pratique nous impose un paramétrage personnalisé et spécifique des paramètres du système. Le paramétrage signifie ici la modification et le réglage des différentes variables du système, tels que le degré de confiance associé à chaque *matcher* ou encore les seuils définis sur les valeurs de similarité calculées par ces *matchers*, selon le domaine d'application et/ou la particularité de la modélisation et l'organisation. Pour l'instant, tout le paramétrage doit être fait par l'expert, ce qui est assez difficile et très abstrait pour lui. D'ailleurs, nous avons pu observer au cours de la phase d'expérimentation que le choix des valeurs des paramètres était fortement lié au domaine d'application et aux résultats obtenus par les différents tests. Par exemple, si la valeur du seuil d'un *matcher* est « très petite », le temps d'exécution de ce *matcher* est « très élevé » ; en revanche, si cette valeur est « très grande », il y a un risque que des correspondances correctes ne seront plus identifiées par le système.

Plutôt que de demander à l'utilisateur/expert de définir les valeurs des différents paramètres, il serait plus facile et plus porteur de sens pour lui que le système utilise ses retours sur des résultats même partiels, pour déduire ces valeurs. Par exemple, l'utilisateur peut exprimer :

- sa satisfaction sur les résultats finaux du système (ce qui permet de valider le seuil appliqué sur les valeurs de similarité) ;
- sa préférence pour le type de relation créée par la correspondance entre le couple de concepts ;
- sa satisfaction pour les résultats d'un *matcher* (ce qui permet d'augmenter le degré de confiance associé à celui-ci).

Automatisation de l'analyse des ressources pour l'enrichissement sémantique

L'enrichissement sémantique des ontologies constitue l'un des avantages portés par notre proposition. Il est basé sur l'analyse des ressources et des documents annotés par ces ontologies. Les apports de notre approche ne peuvent être réalisés que s'il y a un certain nombre d'entrepôts de ressources « bien définis ». Autrement dit, il faut un effort préalable des experts du domaine pour décrire les différentes caractéristiques et les éventuelles relations de dépendance entre les ressources de ces entrepôts.

Les futures directions possibles devraient réduire l'impact de l'utilisateur et avoir un processus automatique ou au moins semi-automatique de création des propriétés des ressources. Nous estimons que les mécanismes et méthodes d'analyse et d'indexation des documents peuvent nous apporter des éléments de réponse à ce problème.

Pour conclure, l'approche proposée dans cette étude ne vise pas à améliorer les phases de construction d'ontologies ou d'annotation des entrepôts de ressources, mais elle suppose que les ontologies existent et sont reliées aux ressources. Cependant, ce n'est pas toujours le cas. En effet, dans le cadre biomédical par exemple, nous avons rencontré des difficultés à trouver un fichier bien formé d'annotation d'un entrepôt ou une base de données biomédicale. Par conséquent, un effort de pré-intégration est nécessaire pour appliquer le système ROMIE dans ce contexte.

Diffusion des outils réalisés

Nous comptons diffuser nos outils auprès de communautés plus larges. Cela nécessite d'une part de les intégrer dans une architecture ouverte et distribuée de type Web et d'autre part de passer d'un stade démonstrateur à un stade prototype. Ceci devrait nous permettre d'avoir des retours plus nombreux, sur des ontologies de domaines différents, de taille différentes, et avec des experts/utilisateurs différents.

Le mapping partiel devrait être évalué plus précisément en offrant ce type de service à une communauté de type e-learning (apprenant ou auteur) qui recherche plus des ressources annotées dans des ontologies diverses et distantes (approche requête) et non pas à faire correspondre deux ontologies.

Evaluer ROMIE selon les annotations fournies

ROMIE propose trois cas possibles d'annotations avant l'étape d'enrichissement :

- cas 1 : deux ontologies annotant des ressources distinctes sans relations entre les ressources ;
- cas 2 : deux ontologies annotant des ressources distinctes avec relations entre les ressources ;
- cas 3 : deux ontologies annotant les mêmes ressources.

Intuitivement, le cas 1 devrait donner de moins bons résultats que le cas 2 qui lui-même devrait donner de moins bons résultats que le cas 3. Il reste à montrer expérimentalement que cette intuition est bonne (notamment cas 2 versus cas 3). Il serait également intéressant de pouvoir estimer à priori le gain produit par un ensemble d'annotations donné ; cela permettrait de fournir un guide à l'utilisateur/expert en lui proposant de produire de nouvelles annotations le cas échéant.

Bibliographie

[1]. ACM. (2009). Association for Computing Machinery. http://www.acm.org/, consulté en février 2009.

[2]. Adobe-Flex. (2009). http://tryit.adobe.com/fr/flex/?sdid=CCKTR, consulté en janvier 2009.

[3]. ARIADNE. (2009). « Ariadne foundation for the European Knowledge Pool », disponible sur http://www.ariadne-eu.org. consulté en janvier 2009.

[4]. Baader, F., Calvanese, D., McGuinness, D., & Patel-Schneider, P. (2003). The Description Logic Handbook. Cambridge University Press.

[5]. BACHIMONT, B. (2000). *Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances, in CHARLET J., ZACKLAD M., KASSEL G. & BOURIGAULT D., eds., Ingénierie des connaissances : évolutions récentes et nouveaux défis* (éd. Eyrolles). Eyrols.

[6]. BAGET, J. (2004). Homomorphismes d'hypergraphes pour la subsomption en RDF/RDFS. In Actes de la 10e conférence Langages et Modèles à Objet (LMO'2004), pages 203–216.

[7]. Bergamaschi, S., Beneventano, D., Castano, S., & Vincini, M. (1998). Momis: An intelligent system for the integration of semistructured and structured data. Technical Report T3-R07, Università di Modena e Reggio Emilia, Modena (IT).

[8]. Bergamaschi, S., Castano, S., & Vincini, M. (1999). Semantic integration of semistructured and structured data sources. SIGMOD Record, 28(1). pp. 54–59.

[9]. Berners-Lee, T. (2001). The semantic web. *Scientific american* , pp. 284(5):35–43,.

[10]. BIEBOW, M., & SZULMAN, S. (1999). A method and a tool to build of a domain ontology, in Proceedings of the 11th European Knowledge Acquisition Workshop (EKAW'99), Springer.

[11]. Bodenreider, O., Hayamizu, T., Ringwald, D. C., & Zhang, a. S. (2005). Of Mice and Men: Aligning Mouse and Human Anatomies, In AMIA Annu Symp Proc. 61–65, .

[12]. Bouaud, J., Bachimont, B., Charlet, J., & Zweigenbaum, P. (1995). Methodological Principles for structuring an ontology, in Proceedings of IJCAI'95 Workshop : Basic Ontological Issues in Knowledge sharing.

[13]. Bouquet, P., Magni, B., & Serafini, L. (2003). A SATBased Algorithm for Context Matching. In : 4th International and Interdisciplinary Conference, CONTEXT 2003. June 23-25, 2003, Stanford, CA, USA. (Springer, Éd.) pp. 66-79.

[14]. Bouzeghoub Amel, Elbyed Abdeltif, Ontology mapping for web-based educational systems interoperability. IBIS : International Journal of Interoperability in Business Information Systems, 2006, n° 1, pp. 73-84

[15]. Bouzeghoub Amel, Elbyed Abdeltif, An ontology mapping algorithm to share learning resources. ICTTA '06 : 2nd IEEE International Conference on Information & Communication Technologies : from Theory to Applications, April 24-28, Damascus, Syria, IEEE, 2006, pp. 616-621

[16]. Bouzeghoub Amel, Elbyed Abdeltif, Tahi Fariza, OMIE : Ontology Mapping within an Interactive and Extensible Environment. Data Integration in the Life Sciences, Springer Berlin / Heidelberg, 2008, (Lecture Notes in Computer Science, 5109), pp. 161-168, ISBN 978-3-540-69827-2

[17]. Bouzeghoub, A., Defude, B., Ammour, S., & Duitama, J. (2004). A RDF Description Model for Manipulating Learning Objects, Proc. IEEE International Conference on Advanced Learning Technologies, Joensuu, Finland, August.

[18]. Bray, T., Paoli, J., & Sperberg-McQueen, C. (2000, October 6). Extensible Markup Language (XML) 1.0, Second Edition, World Wide Web Consortium.

[19]. Chabalier, J., Dameron, O., & Burgun, A. (2007). Integrating and querying disease and pathway ontologies: building an OWL model and using RDFS queries. Proceedings of the Bio-Ontologies Special Interest Group Workshop, Intelligent Systems for Molecular Biology (ISMB).

[20]. Chalupsky, H. (2000). OntoMorph: A translation system for symbolic knowledge. In Anthony G. Cohn, Fausto Giunchiglia, and Bart Selman, editors, KR 2000, Principles of Knowledge Representation and Reasoning Proceedings of the Seventh International Conference. Colorado, USA. pp. 471-482.

[21]. Do, H. H., & Rahm, E. (2002, Août). COMA - A System for Flexible Combination of Schema Matching Approaches. In : Proceedings of the 28th Intl. Conference on Very Large Databases (VLDB), Hongkong.

Bibliographie

[22]. Do, H., Melnik, S., & Rahm, E. (2002). Comparison of schema matching evaluations », Proceedings of the 2nd Int. Workshop on Web Databasess, German Informatics Society, Erfurt, Germany. pp. 221-237.

[23]. Doan, A. H., Domingos, P., & Halevy, A. (2001). Reconciling schemas of disparate data sources: A machine-learning approach. In Proceeding of SIGMOD.

[24]. Doan, A., Madhavan, J., & Domingos, P. (2002, May 7-11). Learning to Map between ontologies on the Semantic Web. In : the eleventh International World Wide Web conference (WWW2002), Honolulu, Hawaii, USA.

[25]. DOE. (2002). Differential Ontology Editor Home Page, http ://opales.ina.fr/public.

[26]. EducaNext. (2009). disponible sur http://www.educanext.org / consulté en Janvier 2009.

[27]. Ehrig, M., & Sure, Y. (2005). FOAM - Framework for Ontologie Alignment and 'Mapping' - Results of the Ontologie Alignment Evaluation Initiative. Integrating Ontologies.

[28]. Euzenat, J., & Shvaiko, P. (2007). *Ontology matching.* (Springer, Éd.) Berlin Heidelberg.

[29]. Farquhar, A., Fikes, R., & Rice, J. (1997). The Ontolingua Server: a tool for collaborative ontology construction. International journal of Human-Computer studies, 46(6):707–727.

[30]. Fensel, D., Horrocks, I., van Harmelen, F., & De, S. (2000). Oil in a nutshell. In Proceedings of European Knowledge Acquisition Workshop (EKAW'2000), volume 1937, pages 1–16. Springer-Verlag LNAI.

[31]. Fernandez, M., Gomez-Perez, A., & Juristo, N. (1997). From ontological art towards ontological engineering, in Proceedings of the Spring Symposium Series on Ontological Engineering (AAAI'97), AAAI Press.

[32]. Finin, T., Weber, J., Wiederhold, G., Genesereth, M., Fritzson, R., MacKay, D., et al. (1993). Draft specification of the KQML agent communication language. Technical report, DARPA Knowledge Shaaring Effort. ftp://ftp.cs.umbc.edu/pub/ARPA/kqml/papers/kqml.ps.

[33]. FIPA00037. (2002). FIPA ACL communicative act library specification. Techenical report, FIPA. http://www.fipa.org/specs/fipa00037.

[34]. FIPA00061. (2002). FIPA ACL message structure specification. http://www.fipa.org/specs/fipa00061.

[35]. Fowler, J., Nodine, M., Perry, B., & Bargmeyer, B. (1999). Agentbased semantic interoperability in infosleuth. SIGMOD Record, 28(1).

[36]. GINSBERG, M. L. (1994). Knowledge Interchange Format: the KIF of death. AI Magazine, 12(3):57–63.

[37]. Giunchiglia, F., Shvaiko, P., & Yat, M. (2004). S-match: an algorithm and an implementation of semantic matching. In Proceedings of ESWS'04, number 3053 in LNCS, Heraklion, Greece. pp. 61–75.

[38]. GO. (2009). Gene Ontology Consorcium, http://www.geneontology.org. consulté en Janvier 2009.

[39]. GOMEZ-PEREZ, A., FERNANDEZ, M., & DE VICENTE, A. J. (1996). Towards a Method to Conceptualize Domain Ontologies, in Proceedings of the European Conference on Artificial Intelligence, ECAI'96. pp. 41-52.

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

[41]. GRUBER, T., & OLSEN, G. (1994). An ontology for enginering mathematics, in J. DOYLE F. S.&TORANO P., eds., Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasonning,. Morgan-Kauffmann, .

[42]. GUARINO, N., & GIARETTA, P. (1995). *Ontologies and knowledge bases, towards a terminological clarification, in MARS N.* (I. P. Towards very large knowledge bases : knowledge building and knowledge sharing, Éd.)

[43]. GUARINO, N., CARRARA, C., & GIARETTA, P. (1994). An ontology of meta-level categories, in J. DOYLE F. S.&TORANO P., eds., Principles of Knowledge representation and Knowledge Reasonning, Morgan-Kauffmann, pages 270-280.

[44]. He, B., Chang, K., & Han, J. (2003, December). Automatic Complex Schema Matching across Web Query Interfaces: A Correlation Mining Approach. Technical Report UIUCDCS-R-2003-2388, Department of Computer Science, UIUC.

[45]. HENDLER, J., & MCGUINNESS, D. (2001). The Darpa Agent Markup Language. http://www.daml.org,.

[46]. Jade. (2009). Java Agent Development Framework, http://jade.tilab.com/, consulté en Janvier .

[47]. KASSEL, G. (2002). OntoSpec : une méthode de spécification semi-informelle d'ontologies, in Actes des journées francophones d'Ingénierie des Connaissances (IC'2002). pp. 75-87.

[48]. Kifer, M., Lausen, G., & Wu, J. (1995). Logical foundations of object-oriented and frame-based languages. Journal of the ACM, 42(4):741-843, 1995.

[49]. Kiryakov, A., Simov, K. I., & Dimitrov, M. (2001). Ontomap: The upper-ontology portal. In Proceedings of "Formal Ontology in Information Systems", Ogunquit, Maine.

[50]. Klein, M., Gomez-Pérez, A., Gruninger, M., & Stuckenschmidt, H. (2001, August 4-5). Combining and relating ontologies : an analysis of problems and solutions. Workshop on Ontologies and Information Sharing (IJCAI-01).

[51]. Lambrix, P., & Tan, H. (2006). SAMBO - A system for aligning and merging biomedical ontologies. J. Web Sem. 4(3). pp. 196-206.

[52]. Lambrix, P., Tan, H., & Liu, Q. (2008). SAMBO and SAMBOdtf Results for the Ontology Alignment Evaluation Initiative.

[53]. Levenshtein. (1966). Levenshtein, Binary codes capable of correcting deletions, insertions, and reversals. Cybernetics and Control Theory.

[54]. Madhavan, J., Bernstein, P. A., & Rahm, E. (2001, September 11-14). Generic Schema Matching with Cupid. In : Proceedings of Very Large Databases Journal (VLDB01). pp. 49-58.

[55]. Maedche, A., Motik, B., Silva, N., & Volz, R. (2002). MAFRA - A MApping FRAmework for Distributed Ontologies. (EKAW 2002). pp. 235-250.

[56]. May, P. E. (2006). ZIB Structure Prediction Pipeline. *Composing a Complex Biological Workflow through Web Services. , vol. 4128* (pp. 1148--1158.).

[57]. McGuinness, D. L., & van Harmelen, F. (2004, February 10). OWL Web Ontology Language Overview, W3C Recommendation.

[58]. McGuinness, D. L., Fikes, R., Rice, J., & Wilder, S. (2000). An Environment for Merging and Testing Large Ontologies.(KR 2000). pp. 483-493.

[59]. McGuinness, D. L., Fikes, R., Rice, J., & Wilder, S. (2000). The Chimaera Ontology Environment. (AAAI/IAAI 2000). pp. 1123-1124.

[60]. Melnik, S., Garcia-Molina, H., & Rahm, E. (2002, 26 February - 1 March). Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. In : Proceedings of the 18th International Conference on Data Engineering, San Jose, CA. pp. 117-128.

[61]. Mena, E., Illarramendi, A., Kashyap, V., & Sheth, A. P. (2000). OBSERVER: An approach for query processing in global information systems based on interoperation across pre-existing ontologies. Distributed and Parallel Databases. pp. 223–271.

[62]. Miller, G. (1995). WordNet: A lexical database for english. communication of the ACM. pp. 39-41.

[63]. Miller, T. B.-L. (2002, Octobre). The semantic web lifts off. . *In ERCIM News N0. 51* .

[64]. Mitra, P., & Wiederhold, G. (2001). An algebra for semantic interoperability of information sources. In IEEE International Conference on Bioinformatics and Biomedical Egineering. pp. 174–182.

[65]. Mitra, P., Wiederhold, G., & Ker, M. L. (March 2000). A graph-oriented model for articulation of ontology interdependencies. In Proceedings of Conference on Extending Database Technology (EDBT 2000), Konstanz, Germany.

[66]. MIZOGUCHI, R., & IKEDA, M. (1997). Towards ontolgy engineering, in Proceedings of the Joint Pacific Asian Conference on Expert Systems.

[67]. NLM. (2009). Medical Subject Headings, www.nlm.nih.gov/mesh, consulté en janvier.

[68]. Nodine, H., Fowler, J., Ksiezyk, T., Perry, B., Taylor, M., & Unruh, A. (2000). ,. Active information gathering in infosleuth. International Journal of Cooperative Information Systems, 9(1-2):3–28.

[69]. Noy, N. F., & Musen, M. A. (2001, août). Anchor-PROMPT: Using Non-Local Context for Semantic Matching. In : Proceedings of workshop on Ontologies and Information Sharing at the International Joint Conference on Artificial Intelligence (IJCAI), Seattle, WA.

[70]. Noy, N. F., & Musen, M. A. (2000). Prompt: algorithm and tool for automated ontology merging and alignment. In Proceeding of Seventeenth National Conference on Artificial Intelligence AAAI.

[71]. Noy, N. F., & Musen, M. A. (1999). Smart: Automated support for ontology merging and alignment. Technical Report SMI-1999-0813, Stanford Medical Informatics.

[72]. NOY, N., FERGERSON, R. W., & MUSEN, M. A. (2000). The knowledge model of Protégé2000 : combining interoperability and flexibility, in Proceedings of the International Conference on Knowledge Engineering and Knowledge Management (EKAW'00).

[73]. OAEI. (2009). http://oaei.ontologymatching.org/, consulté en .

[74]. OBD. (2009). Open Biomedical Database, http://www.fruitfly.org/~cjm/obd/, consulté en Janvier 2009.

[75]. OBO. (2009). Open Biomedical Ontology, http://obo.sourceforge.net. consulté en Janvier 2009.

[76]. Omelayenko, B. (2002). RDFT: A mapping meta-ontology for business integration. In Proceedings of the Workshop on Knowledge Transformation for the Semantic Web (KTSW 2002) at the 15-th European Conference on Artificial Intelligence, Lyon, France. pp. 76–83.

[77]. OntoBroker. (2009). OntoBroker user guide, http://www.ontoprise.de/, consulté en Janvier .

[78]. ONTOEDIT. (2004). Ontology Editor Home Page, http ://www.ontoprise.de/com/.

[79]. PRO. (2002). PROTEGE2000, Protege2000 Ontology Editor Home Page, http://protege.stanford.edu/.

[80]. Rahm, E., & Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. VLDB Journal: Very Large Data Bases. *10* (4), pp. 334-350.

[81]. Smith, T. W. (1981). Identification of Common Molecular Subsequences. . *147* (195--197).

[82]. Stumme, G., & Maedche, A. (2001). Fca-merge: Bottom-up merging of ontologies. In 7th Intl. Conf. on Artificial Intelligence (IJCAI '01), Seattle, WA, USA. pp. 225–230.

[83]. Thomas, S. A. (2007). SAPHIR - a multi-scale, multi-resolution modeling environment targeting blood pressure regulation and fluid homeostasis. Conf Proc IEEE Eng Med Biol Soc. 1:6648-51.

[84]. TRONCY, R., & ISSAC, A. (2002). DOE: Une mise en oeuvre d'une méthode de structuration différentielle pour les ontologies, in Actes des journées francophones d'Ingénierie des Connaissances (IC'2002), pages 63-74.

[85]. UMLS. (2009). UMLS:Unified Medical Language System. http://www.nlm.nih.gov/research/umls/, consulté en janvier 2009.

[86]. USCHOLD, M., & KING, M. (1995). Towards a methodology for building ontologies, in Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI'95.

[87]. Wache, H., Visser, U., & Scholz, T. (2002). Ontology Construction - An Iterative and Dynamic Task. In: Florida Artificial Intelligence Research Society Conference (FLAIRS), Pensacola, FL, USA. pp. 445-449.

[88]. WELTY, C., & GUARINO, N. (2001). Supporting ontological analysis of taxonomic relationships, Data et Knowledge Engineering (39). pp. 51-74.

[89]. WordNet. (2009). WordNet a lexical database for the English language, http://wordnet.princeton.edu/, consulté en Janvier.

[90]. Xu, L., & Embley, D. (2003, October 20). Using Domain Ontologies to Discover Direct and Indirect Matches for Schema Elements. In : Second International Semantic Web Conference (ISWC- 03).