Détection de problèmes de qualité dans les ontologies construites automatiquement à partir de textes

Toader Gherasim

THÈSE DE DOCTORAT

Détection de problèmes de qualité dans les ontologies construites automatiquement à partir de textes

Toader GHERASIM

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université de Nantes

École doctorale : Sciences et Technologies de l'Information et Mathématiques

Discipline : Informatique et applications, section CNU 27
Unité de recherche : Laboratoire d'informatique de Nantes-Atlantique (LINA)

Soutenue le 30 SEPTEMBRE 2013

Jury :

Présidente :	Sylvie DESPRES, Professeur, Université de Paris-XIII
Rapporteurs :	Chantal SOULE-DUPUY, Professeur, Université Toulouse 1
	Sylvie RANWEZ, Chargée de Recherche (HDR), LGI2P, École des mines d'Alès
Directrice de thèse :	Pascale KUNTZ, Professeur, Ecole Polytechnique de l'Université de Nantes
Co-directeur de thèse :	Giuseppe BERIO, Professeur, Université de Bretagne Sud
Encadrant de thèse :	Mounira HARZALLAH, Maître de conférences, Université de Nantes

Editée en LaTeX 2_ε.

A ma famille et à mes amis

Remerciements

Je tiens tout d'abord à remercier mes encadrants, Pascale Kuntz, Mounira Harzallah et Giuseppe Berio. Ils m'ont inspiré, guidé, écouté et supporté tout au long de ces quatre années. Leurs précieux conseils, leur expérience et leur disponibilité ont permis le bon déroulement de ma thèse.

Je remercie aussi tous les membres de l'équipe COD et du Département Informatique de Polytech'Nantes pour la qualité de leur accueil. Merci à tous ceux qui m'ont accompagné dans mon apprentissage de la recherche et de l'enseignement.

Je tiens à exprimer ma gratitude à Mme. Chantal Soule-Dupuy et à Mme. Sylvie Ranwez pour l'honneur qu'elles m'ont fait en acceptant d'être rapporteurs de ma thèse. Je remercie aussi Mme. Sylvie Desprès d'avoir accepté d'être examinatrice et d'avoir fait partie du comité de suivi de ma thèse.

Je tiens également à remercier tous les partenaires du projet ISTA3, qui a assuré le financement des deux premières années de ma thèse tout en me permettant de découvrir un autre aspect du monde de la recherche.

Enfin, merci à ma famille et à mes amis proches d'avoir été de mes plus fervents supporters tout au long de cette thèse.

Table des matières

Table des figures

Liste des tableaux

1 Introduction

Sommaire

1.1 De nouveaux besoins en ingénierie des ontologies

Depuis les travaux fondateurs de Gruber (1993) [57], les ontologies jouent un rôle majeur en Ingénierie des Connaissances, et leur essor, associé à celui du web sémantique, ne cesse de croître. Elles sont maintenant au coeur des systèmes de recherche d'information ou d'aide à la décision de multiples domaines (e.g. sciences de la vie [129], médical [2], juridique [23], apprentissage [77]). Initialement de tailles restreintes et construites entièrement « à la main », elles peuvent contenir aujourd'hui plusieurs milliers de concepts et de relations variées. Et, si dans la lignée des systèmes experts, leur construction a reposé, et repose encore bien souvent, sur l'expertise humaine, leur popularisation et le changement d'échelle nécessitent de plus en plus le recours massif à une forme d'automatisation qui limite la contribution des experts. L'accessibilité croissante de nombreuses ressources textuelles qui renferment des connaissances d'une grande richesse sur les concepts et les processus mis en oeuvre dans les applications ainsi que l'exploitation des connaissances du Web [89, 117] rendent cette aspiration de plus en plus crédible.

Des propositions à la fois méthodologiques et logicielles se sont multipliées ces dernières années pour construire automatiquement des ontologies à partir de corpus textuels [13, 33, 36, 85, 87, 109, 134]. Développées souvent dans des contextes applicatifs ces approches tentent par des techniques variées d'automatiser les tâches, ou plus souvent une partie des tâches, qui composent le processus de construction. Des résultats prometteurs ont été obtenus mais, devant la limite de certains résultats pour les applications réelles, des auteurs continuent de préconiser l'intégration d'une intervention manuelle dans les étapes de construction [120]. Cependant, l'intérêt pour une automatisation croissante n'est pas remis en question.

Cette évolution ravive les questionnements autour de l'évaluation de la qualité des ontologies. Ces questions ont été abordées dès les années 90 [53], et des propositions variées ont été proposées depuis (e.g. [37, 54, 64]). Idéalement, pour les nouvelles applications visées, on pourrait souhaiter que la construction automatique soit couplée avec une validation automatique mais nous sommes aujourd'hui encore loin de cet objectif. Et comme le mentionne Vrandecic dans le « Handbook of ontologies » [136] la définition de ce qu'est une « bonne ontologie » reste une question ouverte.

D'un point de vue opérationnel, l'expérience acquise en Ingénierie des ontologies par différents membres de l'équipe de recherche Connaissances & Décision du Laboratoire d'Informatique de Nantes Atlantique dans laquelle s'est effectuée cette thèse nous a convaincus que l'identification des erreurs était un premier pas pour une intégration contrôlée des ontologies construites automatiquement ou semi-automatiquement dans des systèmes décisionnels ; cette identification devant être idéalement suivie par une phase de réparation mais, là encore, l'objectif semble encore éloigné. Différents travaux ont porté ces dernières années sur l'analyse des erreurs [11, 19, 107, 108] mais les analyses proposées reposent sur des notions non standardisées, et de plus, peu d'analyses expérimentales ont été à notre connaissance menées pour proposer des estimations quantifiées.

Dans ce contexte, nous avons axé les travaux de cette thèse selon trois axes principaux : (1) une comparaison de différents outils logiciels disponibles pour la construction automatique basée à la fois sur des aspects techniques et sur des retours d'expérimentations, (2) la construction d'une typologie des problèmes de qualité reposant sur des définitions standardisables inspirées de la norme ISO 9126 définie en qualité logicielle, et (3) une analyse des problèmes de qualité identifiés dans un cadre expérimental que nous avons défini pour éprouver notre typologie et tenter d'avoir quelques indications quantitatives sur la fréquence des différents problèmes. Cette démarche est complétée en dernière partie par une proposition visant à identifier automatiquement une erreur fréquemment observée : la polysémie de l'étiquetage des concepts.

1.2 Une motivation liée à un contexte applicatif

Outre les besoins identifiés dans la littérature issus de l'évolution des usages des ontologies, notre problématique a été également motivée par des questions applicatives posées dans un projet FUI (Fonds Unique Interministériels) qui a contribué au financement des deux premières années de thèse (2009-2011). L'objectif général du projet ISTA 3 (Interopérabilité de 3ème génération pour les sous-traitants de l'aéronautique) initié par un des pôles de recherche fondateurs du Laboratoire international Virtuel pour l'Interopérabilité d'Entreprise (INTEROP-VLab) (www.interop-vlab.eu) est de rendre interopérables des applications informatiques des sous-traitants de l'aéronautique de rang 2 et 3 [1].

De façon générale, actuellement, les entreprises développent des relations de plus en plus nombreuses, variées, complexes et à haute valeur ajoutée avec des fournisseurs, des clients, des sous-traitants, etc. Ces relations concernent des métiers différents et sont assurées à travers des applications informatiques hétérogènes de conception de produit, de PLM (Product Life cycle Management), de SGDT (Gestion de Données Techniques), de ERP (Enterprise, Resource Planning), de vente, d'achat, etc. Dans ce contexte, l'interopérabilité de ces applications devient un enjeu majeur pour s'adapter rapidement et à moindre coût à des nouvelles coopérations. Dans le domaine de l'Interopérabilité des Systèmes et des Applications informatiques d'Entreprise (ISAE), trois approches ont été proposées afin d'établir l'interopérabilité des systèmes et applications hétérogènes : (1) une approche intégrée en utilisant un format standard, (2) une approche unifiée en utilisant un méta-modèle et (3) une approche fédérée permettant aux applications de s'adapter dynamiquement à l'aide d'ontologies. Dans la troisième approche, les ontologies permettent d'aborder le problème d'hétérogénéité sémantique des différentes applications sans leur imposer un vocabulaire ou un standard unique.

Le projet ISTA3 [3, 65] a intégré différentes recherches actuelles en interopérabilité d'entreprises : EM/BPM (Enterprise Modelling/Business Process Modelling), MDI (Model-Driven Interoperability), SOA (Services-Oriented Architecture), et les ontologies. C'est sur ce dernier aspect qu'a porté notre contribution. Plus précisément, une architecture hybride a été adoptée pour l'exploitation d'ontologies [137] ; elle est composée d'une ontologie générique de haut niveau commune aux différentes applications qui doivent interopérer et

1. c'est-à-dire des sous-traitants des sous-traitants des entreprises principales

d'ontologies spécifiques à chacune d'entre elle. Des relations de spécialisations ou de correspondances ont été identifiées entre les concepts de l'ontologie générique et les concepts des ontologies spécifiques. L'ontologie générique concerne le cycle de vie du produit de sa conception à sa fabrication. Les ontologies spécifiques portent sur le produit composite, sur sa conception et sa fabrication et sur les domaines de vente, de gestion de la production, etc.

Dans le projet, l'ontologie générique a été construite à la main en s'appuyant sur une partie de la norme STEP (STandard for the Exchange of Product model data) et en réutilisant des ontologies existantes. En revanche, la construction manuelle des ontologies spécifiques nécessitait une expertise métier forte dont l'acquisition s'est heurtée à la faible disponibilité des experts des entreprises partenaires. S'est donc « naturellement » posée la question d'un recours à une construction automatique ou semi-automatique à partir de corpus de textes techniques disponibles dans le champ industriel du projet. Et, dans un contexte opérationnel, nous avons été conduits à nous interroger sur la qualité des ontologies ainsi construites.

1.3 Contributions de la thèse

Avec comme objectif *in fine* l'amélioration de la qualité des ontologies construites automatiquement à partir de textes, nous aurions pu choisir de développer un nouveau processus de construction automatique. Cependant, deux raisons nous ont dissuadés de nous engager dans cette démarche. La première est liée aux contraintes du projet ISTA3 qui a financé les deux premières années de cette thèse. La durée restreinte (deux ans) de ce projet collectif ambitieux n'était pas adaptée à un cycle de développement complet débouchant sur des résultats sur les corpus industriels du projet. La seconde raison est liée à un manque de retours expérimentaux sur les systèmes existants. Avant de s'engager sur le développement d'un nouveau processus, il paraît raisonnable de s'appuyer sur un état des lieux des avantages et limites des outils actuels de construction automatique. Or, à notre connaissance, seules quelques comparaisons partielles ont été publiées dans la littérature.

Sans prétendre à l'exhaustivité nous avons donc choisi de commencer nos recherches par une telle comparaison selon deux axes complémentaires : les caractéristiques techniques et fonctionnelles des outils (entrées / sorties / prétraitements, disponibilité, paramétrage, etc.), et la qualité des résultats expérimentaux comparés à une ontologie faite « à la main » par un expert. Le cadre d'analyse a été balisé par le référentiel de tâches de Methontology [44] qui est certainement l'un des plus connus en Ingénierie des Connaissances. L'analyse des resultats expérimentaux a mis en évidence une qualité globale assez faible avec cependant une forte hétérogénéité, et nous a permis de dégager les outils plus performants.

Lors de cette première analyse expérimentale nous avons commencé à identifier différents problèmes de qualité sans toutefois en avoir une vision globale et structurée. Devant la fréquence élevée et / ou l'importance supposée de certains, nous avons décidé d'orienter la suite de nos travaux sur leur identification « systématique ». Un état de l'art sur les problèmes de qualité nous a convaincus de l'absence d'une approche standardisée. Nous avons ainsi proposé une typologie des problèmes de qualité selon deux dimensions complémentaires : (1) les erreurs (qui rendent l'ontologie inutilisable ou non conforme aux attentes) *versus* les

situations indésirables (qui affectent l'ontologie sans la rendre inutilisable), (2) les aspects logiques *versus* les aspects sociaux (qui concernent les expériences et les connaissances tacites des utilisateurs). La description des cas de la typologie s'appuie sur une formalisation rigoureuse concernant les aspects logiques, et une tentative de précision concernant les aspects sociaux qui sont beaucoup plus délicats à capter.

Cette typologie a été testée de façon expérimentale : nous nous sommes appuyés sur elle pour identifier les problèmes de qualité associés à deux ontologies construites à partir de corpus textuels distincts avec l'outil Text2Onto, qui est ressorti comme étant l'un des plus efficaces lors de notre comparaison préalable. Cette analyse a permis de recueillir quelques estimations statistiques – pour lesquelles nous sommes conscients des limites de leur significativité – de la fréquence des problèmes. Nous l'avons complétée par une analyse des facteurs explicatifs en reprenant la séquence des tâches de Methontology.

Toujours conscients des nécessités d'une automatisation, au moins partielle, la dernière partie de nos recherches a été consacrée à la question de l'identification automatique des problèmes de qualité. Après un rapide état de l'art des approches proposées dans la littérature pour chacun des problèmes de notre typologie, nous avons développé une nouvelle heuristique pour l'identification des étiquettes des concepts polysémiques, qui est un problème qui a été fréquemment rencontré dans nos expérimentations et pour lequel nous pensons qu'une première détection automatique peut être efficace.

L'étape suivante – la réparation automatique – n'est pas traitée dans ce manuscrit mais discutée en dernière partie.

1.4 L'organisation du manuscrit

La structure du manuscrit suit la chronologie de notre démarche décrite ci-dessus. Le document est organisé en 8 chapitres.

Le Chapitre 2 rappelle les définitions proposées pour définir la notion d'« ontologie » et décrit les différents éléments qui la composent. Nous détaillons plus particulièrement deux définitions – celles proposées par Stumme et al., 2003 [126] et Guarino et al., 2009 [60] – sur lesquelles nous nous appuyons dans la suite. Elles proposent deux regards complémentaires : l'une décrit l'ontologie sous son aspect structurel, et l'autre sous son aspect spécification formelle d'une conceptualisation.

Le Chapitre 3 est un état de l'art des approches et outils logiciels qui ont été proposées dans la littérature pour la construction automatique des ontologies à partir de textes. Sans être exhaustifs, nous avons tenté de balayer les principales approches en les associant à des outils opérationnels basés sur des algorithmes différents.

Le Chapitre 4 est consacré à la comparaison des différentes approches de construction automatique d'ontologies et des outils qui leur sont associés. Nous y définissons un cadre d'analyse qui permet une comparaison assez complète qui recouvre différents aspects : le degré de complétude et d'automatisation des approches, les caractéristiques techniques et

fonctionnelles des outils et la qualité des résultats obtenus dans le cadre d'une expérimentation.

Le Chapitre 5 commence par un état de l'art sur les problèmes qui affectent la qualité des ontologies. Cet état de l'art met en évidence la variété des problèmes – et même des terminologies employées dans la littérature –. Nous proposons ensuite une typologie des problèmes inspirée des travaux sur la qualité en génie logiciel et sur la qualité des modèles conceptuels.

Le Chapitre 6 tente tout d'abord d'associer la présence potentielle des problèmes de qualité de la typologie aux différentes tâches de Methontology. Puis, il présente une expérimentation sur deux ontologies – dont l'une relative aux données du projet ISTA3 – construites automatiquement avec l'outil Text2Onto.

Le Chapitre 7 liste pour chaque problème des heuristiques qui ont été proposées dans la littérature pour leur identification automatique. Puis, nous décrivons plus en détails une heuristique que nous avons développée pour l'identification de problème de polysémie dans l'étiquetage des concepts. Des retours d'expériences sur les ontologies construites au Chapitre 6 sont exposés.

Le Chapitre 8 tire les conclusions de notre démarche et trace des perspectives à court et moyen terme.

2

Les ontologies en ingénierie des connaissances

Sommaire

2.1 Introduction

Sans revenir à Aristote, depuis le XVI siècle le terme d'*ontologie* a été utilisé en philosophie pour designer la « théorie de l'être en tant qu'être [1] », une manière d'étudier et d'exprimer ce qui est « vrai » concernant les choses qui existent. Quelques siècles plus tard et après l'avènement des systèmes experts en intelligence artificielle, l'ingénierie des connaissances a renoué avec ces questions philosophiques pour tenter de modéliser les « connaissances » d'un domaine pour pouvoir les exploiter dans des systèmes informatiques pour différentes tâches (aide à la décision, recherche d'informations, etc.).

Ce chapitre a pour objectif de préciser la notion d'ontologie utilisée en ingénierie des connaissances et de rappeler les définitions sur lesquelles nous nous appuyons dans la suite. Dans la première section (Section 2.2) nous revenons sur les données, les informations et les connaissances qui sont au coeur de l'ingénierie des connaissances. Dans la Section 2.3 nous présentons différentes définitions qui ont été associées à la notion d'ontologie et nous détaillons les principaux éléments constitutifs d'une ontologie. Quelques éléments sur le langage OWL, utilisé pour représenter les différentes ontologies construites dans le cadre de cette thèse, sont introduits dans la Section 2.5. La Section 2.6, plus formelle, précise la structure d'une ontologie, définie formellement par Stumme et al. [126] et les notions de conceptualisation et de modèle intentionnel définies formellement par Guarino et al. [60]. Ces notions nous sont utiles au chapitre 5 pour construire une typologie des problèmes de qualité.

2.2 Des données aux connaissances

La définition d'une ontologie comme « moyen » opérationnel pour représenter et partager des connaissances repose sur la notion très discutée de « connaissance ».

L'informatique est souvent associée à la « science du traitement de l'information » [66], et le sous-domaine de l'ingénierie des connaissances à la « science de l'intégration des connaissances dans des systèmes informatiques pour résoudre des problèmes complexes nécessitant un haut degré d'expertise humaine » [43]. Dans des définitions plus précises, les termes *connaissance*, *information* et *donnée* sont souvent associés avec des sens variés. Le même contenu peut être considéré en tant que donnée, qu'information ou que connaissance en fonction du degré d'interprétation et d'intégration qu'un agent rationnel lui donne ; l'agent rationnel pouvant être un système informatique ou un être humain.

Il semble qu'il y ait un certain consensus pour considérer les *données* comme un ensemble de symboles dont la signification limitée est cadrée par le langage qui permet de les décrire. Tout système informatique prend en entrée, produit et manipule des données. L'*information* est l'interpretation qu'un agent donne, dans un contexte donné, à ces symboles.

Par exemple, « 20 » peut être considéré comme une donnée qui est composée de deux symboles, « 2 » et « 0 », d'un alphabet numérique. Mais, si cette donnée se trouve dans une base de données dans un champ interprété par le système comme étant l'âge d'une personne, alors cette donnée devient une information.

1. D'après l'Encyclopaedia Universalis.

La différence entre *information* et *connaissance* a été l'objet de nombreux débats qui dépassent le cadre de cette thèse [13, 66, 74]. Nous considérons ici que l'*information* devient *connaissance* quand elle est utilisée par un agent rationnel [13, 66, 74]. Celui-ci peut associer l'information à d'autres informations ou connaissances qu'il possède déjà. Les connaissances sont des informations qui peuvent servir comme support pour la prise de décisions ou comme fondement pour un raisonnement.

Les connaissances peuvent être explicites ou tacites [24, 98]. Les *connaissances tacites* sont les connaissances qui existent implicitement (comme résultat de l'expérience, par exemple) dans l'esprit humain et dans les organisations et qui ne sont pas exprimées de façon explicite.

Les *connaissances explicites* sont les connaissances décrites à l'aide d'un moyen de communication (par exemple un ensemble de symboles associé à des règles de composition) dont la transmission, le partage et la sauvegarde peuvent être rendues opérationnelles.

En ingénierie des connaissances, l'objectif est de formaliser et de rendre accessibles les *connaissances explicites* aux systèmes informatiques et aux êtres humains ; l'accessibilité des *connaissances tacites* aux systèmes informatiques étant elle une quête encore éloignée aujourd'hui.

2.3 Des connaissances aux ontologies

La définition proposée par Gruber [57] en 1993 reste la définition de référence de la notion d'ontologie dans le domaine de l'ingénierie des connaissances : « une ontologie est une spécification *explicite* d'une *conceptualisation* ». Différentes variations l'ont suivie. En 1997, Borst [14] a proposé une définition légèrement différente : « une ontologie est une spécification *formelle* d'une *conceptualisation partagée* ». L'année suivante, en 1998, Studer et al. [125] ont unifié les deux définitions : « une ontologie est une spécification *formelle*, *explicite* d'une *conceptualisation partagée* ». Comme d'autres auteurs, Guarino et al. [60] ont proposé une analyse approfondie de la signification de cette définition et ont argumenté l'importance de chacun des termes qui la composent.

D'après Gruber [57] une « *conceptualisation* » est « une vue abstraite et simplifiée du monde que l'on souhaite représenter dans un but donné ». Le qualificatif « *partagée* » associé à la notion de conceptualisation indique qu'elle est le résultat d'un compromis conduisant à un socle commun qui satisfait une communauté de personnes.

La spécification « *formelle* » d'une conceptualisation est la transposition de celle-ci dans un langage qui peut être interprété par les machines. L'expression « *spécification explicite* » indique que dans une ontologie « le type des concepts et les contraintes sur leurs utilisations sont explicitement définies » [125].

Au-delà des définitions générales, différents types d'ontologies ont été considérés selon les objectifs visés : ontologies de domaine, ontologies de tâches, ontologies terminologiques, etc. On peut distinguer aussi, selon le degré de complétude des modèles sous-jacents, les ontologies « légères » des ontologies « lourdes » [66, 131] qui peuvent intégrer des axiomes.

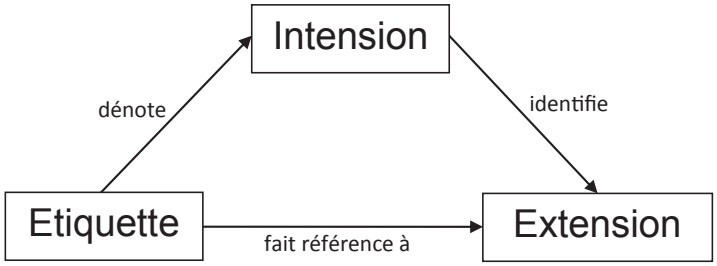

FIGURE 2.1: Triangle sémiotique.

2.4 Les composants de base d'une ontologie

Quelles que soient les définitions, les composants élémentaires d'une ontologie sont les concepts et les relations qui peuvent avoir des instances. Dans cette section introductive, nous complétons la liste de ces composants élémentaires par les axiomes. Mais nous ne les considérons plus dans la suite car dans le domaine de la construction automatique qui nous intéresse ici leur intégration est encore un problème largement ouvert.

2.4.1 Les concepts

Dans le dictionnaire, un concept est défini comme étant une « idée générale et abstraite que se fait l'esprit humain d'un objet de pensée concret ou abstrait, et qui lui permet de rattacher à ce même objet les diverses perceptions qu'il en a, et d'en organiser les connaissances ». Cette notion a été illustrée à l'aide d'un triangle sémiotique (figure 2.1) [25]. Chaque concept est défini par trois aspects :

1. son extension, ou sa définition référentielle : l'ensemble des êtres, objets ou faits auxquels le concept fait référence ;
2. son intension, ou sa définition différentielle : l'abstraction conceptuelle d'un ensemble des caractéristiques essentielles communes aux êtres, objets ou faits auxquels le concept fait référence et qui lui permettent de se différencier des autres concepts ;
3. un ou plusieurs termes, ou étiquettes (labels) : les expressions verbales qui permettent la désignation du concept dans un langage.

Les êtres, objets ou faits qui constituent l'extension d'un concept sont les instances du concept. L'extension d'un concept peut être vide, notamment dans le cas des concepts abstraits comme, par exemple, la « Vérité ». Deux concepts peuvent avoir la même extension mais des intensions différentes. Par exemple, un ensemble composé des deux jeunes $\{Jean, Paul\}$ peut être l'extension du concept « Etudiant » et du concept « Humain », qui ont des intensions différentes. La plupart des auteurs considèrent que l'intention d'un concept permet à elle seule de définir le sens du concept.

Plusieurs termes peuvent être l'étiquette d'un même concept (par exemple, « four », « fourneau », « fournil »). Un terme peut être l'étiquette de plusieurs concepts (par exemple,

le terme « kiwi » peut être l'étiquette du concept *« oiseau terrestre incapable de voler, endémique de la Nouvelle-Zélande et appartenant à l'ordre Apterygiformes »* mais aussi du concept *« fruit de plusieurs espèces de lianes du genre Actinidia »*).

Un concept peut être caractérisé par des propriétés valuées, appelées aussi des attributs. Par exemple, le concept « Etudiant » a les attributs « Date de naissance » et « Numéro d'étudiant ». Les attributs peuvent avoir des valeurs différentes pour chacune des instances associées au concept.

Certains auteurs distinguent les concepts « primitifs » des concepts « définis » [87]. Les concepts primitifs, « dont on postule l'existence » [87], sont décrits par des propriétés nécessaires (mais non suffisantes), alors que les concepts définis sont décrits par des propriétés nécessaires et suffisantes.

2.4.2 Les relations

Les relations, appelées aussi « propriétés d'objet » dans le cadre du langage OWL et des « rôles » dans le cadre des langages de la logique de description, décrivent des associations entre deux concepts [2]. Un ou plusieurs termes peuvent être associés à une relation en tant qu'étiquettes, tout comme un ensemble de règles spécifiant les propriétés logiques de la relation (la transitivité, la symétrie, la fonctionnalité, etc.).

La relation taxonomique (ou de subsomption) est un type particulier de relation entre les concepts. Elle est prédéfinie dans tous les langages pour la représentation des ontologies. C'est une relation transitive, asymétrique et réflexive. Elle définit un ordre partiel entre les concepts de l'ontologie et permet leur organisation dans une hiérarchie avec des héritages multiples.

La relation taxonomique (*« is a »*) est l'ossature des ontologies. Certaines se réduisent d'ailleurs à cette relation ; on parle alors de « taxonomie ». D'autres intègrent d'autres relations mais dont le rôle structurant reste modeste devant la relation taxonomique. Par exemple, sans rentrer ici dans le débat sur l'association de WordNet à une ontologie, une analyse a montré que la hiérarchie de subsomption (ici définie par les relations d'hyponymie et de hypéronymie) contenait près de 80% des relations (pour un ensemble de plus de 140000 noeuds) [18].

2.4.3 Les axiomes

Les axiomes sont des expressions logiques qui permettent de préciser la signification des concepts et des relations. Les axiomes sont aussi nommés des « connaissances inférentielles » parce qu'ils spécifient des règles de raisonnement qu'un moteur d'inférence peut utiliser pour déduire des nouvelles connaissances. Ils précisent la signification des concepts et des relations en définissant, par exemple, des restrictions sur les valeurs des attributs ou sur les arguments des relations.

Par exemple, les deux relations $Amis(Humain, Humain)$ et $Ennemis(Humain, Humain)$ ont un apport sémantique limité. Mais leurs significations peuvent être précisées à l'aide de trois axiomes : (1) *Les amis de mes amis sont mes amis*, (2) *Les ennemis de mes ennemis sont mes amis* et (3) *Les ennemis de mes amis sont mes ennemis*.

2. Certaines logiques de description permettent la définition de relations qui associent plus de deux concepts.

2.5 Représentation des ontologies. Le langage OWL

La définition de la notion d'ontologie proposée par Gruber précise que celle-ci est une spécification formelle sans détailler le formalisme qui doit être utilisé pour spécifier les connaissances. Depuis le début de années 1990, une multitude de formalismes (frames, logiques de description, graphes conceptuels) et de langages (RDF, OIL, DAML, OWL) ont été proposés.

En 2004 une première version du langage OWL (Web Ontology Language) [3] a été normalisée par W3C et proposée comme standard pour la représentation des connaissances sur Internet. Trois sous-langages ont été associés, classés par ordre d'expressivité croissante : OWL-Lite, OWL-DL et OWL-Full.

OWL-Lite a été conçu pour permettre la représentation de hiérarchies de concepts avec une expressivité limitée ; par exemple, il ne permet pas de définir des concepts comme l'union d'autres concepts, mais il permet de les définir comme l'intersection d'autres concepts, et les seules contraintes de cardinalité acceptées sont 0 et 1.

OWL-DL (OWL Description Logic) est un surensemble de OWL-LITE conçu pour offrir une expressivité maximale tout en garantissant la complétude du calcul (toutes les inférences sont assurées) et la décidabilité des algorithmes d'inférence. Par exemple, parmi les éléments interdits en OWL-Lite et acceptés en OWL-DL on peut citer les concepts définis comme l'union d'autres concepts, comme une énumération ou comme une négation ou encore les axiomes de disjonction entre concepts.

OWL-FULL offre, en plus d'une expressivité maximale, la liberté syntaxique de RDF (par exemple, le recouvrement de types : un concept peut aussi être une propriété ou un individu, et réciproquement) avec en contrepartie la perte de la garantie de complétude et de décidabilité des inférences.

Toute ontologie OWL-Lite conforme est une ontologie OWL-DL conforme et toute ontologie OWL-DL conforme est une ontologie OWL-FULL conforme.

Quelle que soit sa mouture, les éléments les plus importants proposés par le langage OWL sont les classes, les individus et les propriétés. Différents types d'axiomes peuvent être utilisés pour mieux formaliser leur sémantique : disjonction, équivalence, inverse, symétrie, réflexivité, transitivité, etc.

Les individus OWL sont des instances des classes. Les classes OWL correspondent à la notion de concept. Dans OWL-FULL une classe peut être une instance d'une autre classe (i.e. méta-classe) alors que dans les deux autres déclinaisons de OWL seuls les individus peuvent être instances d'une classe. Il y a 6 manières de déclarer une classe en OWL :

1. en précisant son nom (respectivement son adresse URI) ;
2. en énumérant ses individus ;
3. comme une restriction (contraintes sur les valeurs ou les cardinalités des propriétés) d'une autre classe ;
4. comme l'intersection d'autres classes (équivalent à la conjonction logique) ;
5. comme l'union d'autres classes (équivalent à la disjonction logique) ;
6. comme complément d'une autre classe (équivalent à la négation logique).

3. http://www.w3.org/TR/owl-features

Il existe deux types de propriétés en OWL :
- *Object property* (« propriétés d'objet », relations conceptuelles) pour définir les relations entre concepts : par exemple, la propriété *grand_parent(Homme, Enfant)* exprime une relation entre le concept *Homme* et le concept *Enfant*
- *Datatype property* (attributs) pour définir les relations entre un concept et un type de données : par exemple, la propriété *date_naissance* associe au concept *Homme* le type de données *Date*.

Une nouvelle version, OWL 2.0, a été proposée en 2009. Elle offre une plus grande expressivité (elle permet, par exemple, la disjonction des propriétés) et instaure trois nouveaux profils du langage censés faciliter l'utilisation et l'implémentation : OWL-EL garantit une complexité polynomiale des inférences ; OWL-QL facilite l'implémentation de l'ontologie dans les bases de données et OWL-RL facilite l'implémentation de l'ontologie dans un moteur de règles.

Comme OWL 1.0 et OWL 2.0 sont devenus des standards et que OWL 1.0 reste encore dominant, nous avons choisi d'utiliser dans cette thèse OWL 1.0 pour représenter les différentes ontologies.

2.6 Définitions formelles

Dans la littérature plusieurs définitions formelles de la notion d'ontologie ont été proposées [25, 38, 59, 60, 126]. Dans cette section, nous présentons deux de ces définitions sur lesquelles nous nous appuyons dans la suite de ce manuscrit.

La première définition, proposée en 2003 par Stumme et al. [126] (Section 2.6.1), considère l'ontologie comme une structure composée et précise ses composants élémentaires et les relations entre eux. La deuxième définition (Section 2.6.2), proposée par Guarino et ses collaborateurs et dont la dernière version a été publiée en 2009 [60], considère qu'une ontologie est avant tout une conceptualisation. Elle met en évidence les limitations inhérentes à la représentation d'une conceptualisation en tant que spécification formelle et à la reconstitution de la conceptualisation par l'interprétation de la spécification formelle. Plusieurs notions introduites dans le cadre de cette définition (modèles, modèles intentionnels, interprétation de l'ontologie, etc.) sont utilisées dans le Chapitre 5 de cette thèse.

2.6.1 Définition d'une ontologie en tant que structure composée

D'après Stumme et al. [126] une ontologie peut être définie du point de vue formel en précisant sa structure, ses composants et les types d'interactions possibles entre les composants.

Définition 2.6.1 (Ontologie)

Une ontologie est une structure $\mathcal{O} = \{\mathcal{C}, \mathcal{R}, \mathcal{A}, \mathcal{T}, \sigma_{\mathcal{R}}, \sigma_{\mathcal{A}}, \leq_{\mathcal{R}}, \leq_{\mathcal{C}}\}$ *avec :*
- $\mathcal{C}$ *- un ensemble de concepts ;*
- $\mathcal{R}$ *- un ensemble de relations ;*
- $\mathcal{A}$ *- un ensemble d'attributs ;*
- $\mathcal{T}$ *- un ensemble de types de données (entiers, booleens, dates, chaînes de caractères, etc.) ;*

- *tels que les éléments de* $\mathcal{C}$, $\mathcal{R}$, $\mathcal{A}$, $\mathcal{T}$ *sont disjoints ;*
- $\sigma_{\mathcal{R}}$ *- une fonction de* $\mathcal{R}$ *dans* $\mathcal{C} \times \mathcal{C}$ *appelée signature de relation et qui précise les concepts* $c_i, c_j \in \mathcal{C}$*, qui sont liés par une relation* $r \in \mathcal{R}$ *;*
- $\sigma_{\mathcal{A}}$ *- une fonction de* $\mathcal{A}$ *dans* $\mathcal{C} \times \mathcal{T}$ *appelée signature d'attribut et qui précise, pour chaque attribut* $a \in \mathcal{A}$*, les concepts qu'il décrit et son type de données ;*
- $\leq_{\mathcal{C}}$ *- un ordre partiel sur* $\mathcal{C}$ *appelé hiérarchie de concepts ou taxinomie ;*
- $\leq_{\mathcal{R}}$ *- un ordre partiel sur* $\mathcal{R}$ *appelé hiérarchie de relations, où* $r_1 \leq_{\mathcal{R}} r_2$ *implique* $\pi_i(\sigma_{\mathcal{R}}(r_1)) \leq_{\mathcal{C}} \pi_i(\sigma_{\mathcal{R}}(r_2))$ [4] *pour* $1 \leq i \leq 2$ *(le i-ème concept lié par la relation* r_1 *est égal à ou est une spécialisation du i-ème concept lié par la relation* r_2*).*

DÉFINITION 2.6.2 (Domaine et co-domaine)
Pour une relation $r \in \mathcal{R}$ *nous appelons* $\pi_1(\sigma_{\mathcal{R}}(r))$ *le domaine de r et* $\pi_2(\sigma_{\mathcal{R}}(r))$ *son co-domaine.*

En complément, une « ontologie lexicalisée » est une ontologie dans laquelle on associe une étiquette à chacun des composants et une « base de connaissances » (ou « ontologie peuplée ») est une ontologie contenant des instances de concepts, de relations et d'attributs. On peut également définir formellement un système axiomatique pour l'ontologie, mais nous l'omettons sciemment dans la suite parce que son utilité dépasse le cadre de cette thèse.

2.6.2 Définition de l'ontologie en tant que conceptualisation

Guarino et al., 2009 [60] ont défini formellement la notion d'ontologie en partant de sa définition informelle la plus largement acceptée : « une ontologie est une spécification *formelle*, *explicite* d'une conceptualisation *partagée* ». Ils ont d'abord défini formellement la notion de conceptualisation et ensuite précisé ce qu'est une spécification explicite et formelle d'une conceptualisation.

2.6.2.1 Qu'est-ce qu'une conceptualisation ?

Gruber [57] s'est inspiré des travaux de Genesereth et Nilsson [51] pour préciser de manière informelle qu'une conceptualisation est « une vue abstraite et simplifiée du monde que l'on souhaite représenter » et que toute conceptualisation est composée des « objets, concepts et autres entités qui sont censées exister dans un domaine d'intérêt et des relations qui existent entre ces éléments ».

Nous avons rappelé (Section 2.4.1) qu'un concept peut être défini en décrivant son intension et que son extension peut varier en fonction du contexte d'utilisation. Ce principe peut être appliqué aussi aux relations. L'extension d'un concept peut être précisée en énumérant toutes ses instances ; celle d'une relation en énumérant toutes les associations entre les instances qui composent les extensions des concepts impliqués dans la relation.

Comme une conceptualisation est composée des *« objets ... censés exister ... et des relations entre eux »*, Guarino et al. définissent la conceptualisation d'une situation concrète d'un domaine d'intérêt par énumération des objets qui existent dans le domaine, de leurs types (i.e. les concepts qui leur sont associés) et des relations entre eux (ces relations entre objets sont des instances des relations entre les concepts).

4. $\pi_i(n)$ représente le i-ème élément du n-uplet n

Définition 2.6.3 (**Structure relationnelle extensionnelle** ou conceptualisation définie de façon extensionnelle)

Une structure relationnelle extensionnelle est un tuple (D, R) *où*

- D *est un ensemble – nommé l'univers du discours – dont les éléments sont des objets existant dans le domaine d'intérêt ;*
- R *est un ensemble de relations sur* D *pour lesquelles on précise les extensions ; les concepts sont un cas particulier de relations dans cet ensemble (i.e. des relations unaires).*

Exemple 2.6.1 (inspiré de l'exemple proposé par Guarino et al., 2009 [60]) Supposons que le domaine d'intérêt soit constitué par les relations entre les employés qui travaillent à la conception d'une ontologie dans une entreprise ayant 100 employés (figure 2.2). Pour chaque employé on connaît seulement son identifiant : la lettre 'E' suivie du numéro de l'employé. Supposons également que les seuls objets de ce domaine sont les employés et que R contient trois relations unaires (*Employé*, *Manager* et *Chercheur*) et deux relations binaires (*coopère_avec* et *subordonné_à*). La structure relationnelle extensionnelle correspondante (D, R) est définie comme suit :

- D = *{ E001, …, E100 }* – l'univers du discours
- R = *{ Employé, Manager, Chercheur, coopère_avec, subordonné_à }* – l'ensemble de relations extensionnelles
- *Employé* = D
- *Manager = { …, E001, E015, … }*
- *Chercheur = { …, E005, E053, E099, … }*
- *coopère_avec = { …, (E005, E015), (E099, E005), … }*
- *subordonné_à = { …, (E005, E001), (E053, E015), (E099, E001), (E100, E015), … }*

Exemple 2.6.2 Considérons maintenant une évolution (D', R') de l'exemple 2.6.1 dans laquelle

- $D' = D$ et
- R' = *{ Employé, Manager, Chercheur, coopère_avec′, subordonné_à }* où
- *coopère_avec′ = coopère_avec* $\cup$ *{ (E100, E053) }* .

Bien que le domaine d'intérêt et la signification des relations soient restés les mêmes et que nous n'ayons ajouté qu'un seul nouvel élément à l'extension d'une des relations, il est évident que $(D, R) \neq (D', R')$. Le problème vient du fait que les relations sont définies à l'aide de leurs extensions qui reflètent un état particulier du domaine d'intérêt et ne parviennent pas à capter les significations des relations et les caractéristiques essentielles qui différencient les binômes (i.e. paires d'employés) qui font partie de l'extension de la relation de ceux qui n'en font pas partie. Admettons que, dans nos exemples, pour les relations *coopère_avec* et *coopère_avec′* les caractéristiques essentielles sont (1) l'existence d'un objectif commun aux deux employés impliqués dans la relation et (2) le fait que les deux employés déclarent agir pour atteindre cet objectif. La signification des deux relations peut être définie par une fonction qui, prenant en entrée une description détaillée de l'état du domaine d'intérêt, identifie tous les binômes qui coopèrent. Cette fonction est la définition intensionnelle de la relation *coopère_avec*.

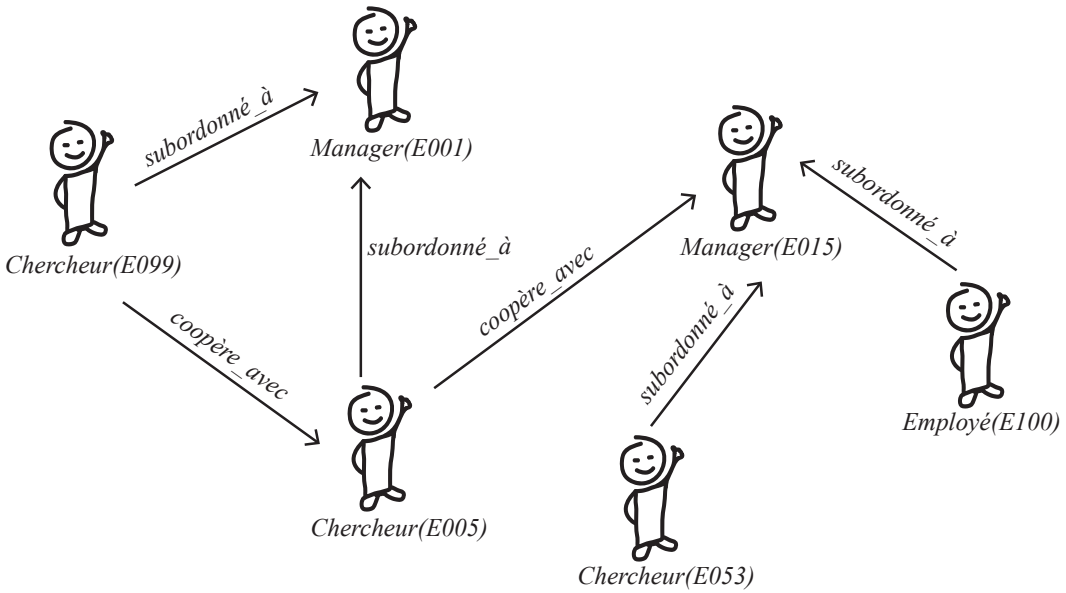

FIGURE 2.2: Une petite partie du domaine d'intérêt de l'exemple 2.6.1.

La formalisation d'une relation définie de façon intensionnelle repose sur les notions de « système » et de « monde » (ou « état du monde »). Un « système » correspond à la réalité du domaine d'intérêt perçue à un certain niveau de granularité et décrite à l'aide d'un ensemble de variables.

Dans notre exemple, le système est un groupe d'employés qui interagissent. Considérons que ce système soit observé à un niveau de granularité où les composants élémentaires sont les employés et où les seules variables observées sont celles qui indiquent si un employé a un certain objectif (appartenant à une liste prédéfinie d'objectifs) et s'il agit pour atteindre l'objectif. Si un seul objectif est prédéfini, alors l'état de chaque employé est décrit à l'aide de deux variables booléennes : (1) la première indique si l'employé considère l'objectif prédéfini comme son propre objectif et (2) la deuxième indique si l'employé agit pour atteindre l'objectif. Comme notre système contient 100 employés, son état est décrit par $100 \times 2 = 200$ variables. Toute combinaison possible pour les valeurs de ces variables constitue un état du monde.

DÉFINITION 2.6.4 (Monde)

Un état du monde *d'un système S est une combinaison unique des valeurs de toutes les variables qui caractérisent le système. Un* monde *est un ensemble totalement ordonné des états du monde, correspondant à l'évolution du système dans le temps. Si, pour simplifier, le temps n'est pas pris en compte alors les définitions d'un état du monde et du monde sont équivalentes.*

A partir de ces définitions, Guarino et al. précisent ce qu'est une relation conceptuelle (ou relation intensionnelle).

DÉFINITION 2.6.5 (Relation conceptuelle ou relation intensionnelle)

Soit S un système quelconque, D un ensemble d'éléments distincts de S, et W l'ensemble des états du monde possibles pour S. Le tuple (D, W), qui précise l'espace de variabilité de l'univers du discours D par rapport aux états possibles du S, est nommé espace du domaine pour S. Une relation conceptuelle ρ^n *d'arité n sur (D, W) est une fonction $\rho^n : W \rightarrow 2^{D^n}$ entre l'ensemble W et l'ensemble de toutes les relations extensionnelles d'arité n entre les éléments de D.*

La définition d'une conceptualisation peut donc être réécrite en remplaçant les relations extensionnelles par des relations conceptuelles.

Définition 2.6.6 (**Conceptualisation** ou structure relationnelle intensionnelle)

Une conceptualisation *(ou* structure relationnelle intensionnelle*) d'un système S est un triple $C = (D, W, \mathfrak{R})$ où D est l'univers du discours, W est l'ensemble de mondes possibles, et $\mathfrak{R}$ est un ensemble de relations conceptuelles sur l'espace du domaine (D, W) pour S.*

Exemple 2.6.3 En reprenant le système défini dans les exemples 2.6.1 et 2.6.2 nous pouvons maintenant définir une conceptualisation unique compatible avec les deux états du système :

- D = *{ E001, … , E100}* – l'univers du discours
- W = *{ w_1, w_2, … }* – l'ensemble des mondes possibles
- $\mathfrak{R}$ = *{ Employé'', Manager'', Chercheur'', coopère_avec'', subordonné_à'' }* – l'ensemble des relations conceptuelles

Nous considérons que les relations unaires (*Employé'', Manager'', Chercheur''*) ont la même extension dans toutes les mondes possibles :

- pour tous les mondes possibles *Employé* = D
- pour tous les mondes possibles *Manager = { …, E001, E015, … }*
- pour tous les mondes possibles *Chercheur = { …, E005, E053, E099, … }*
- *coopère_avec''*(w_1) = *{ …, (E005, E015), (E099, E005), … }*
- *coopère_avec''*(w_2) = *{ …, (E005, E015), (E099, E005), (E100, E053), … }*
- *coopère_avec''*(w_3) = …
- *subordonné_à''*(w_1) = *{ …, (E005, E001), (E053, E015), (E099, E001), (E100, E015), … }*
- *subordonné_à''*(w_2) = …

2.6.2.2 Qu'est-ce qu'une spécification formelle et explicite d'une conceptualisation ?

La spécification formelle d'une conceptualisation repose sur l'expression via un langage logique de l'ensemble de définitions intensionnelles des concepts et des relations. L'ensemble de formules logiques obtenu – appelé aussi « théorie logique »– constitue l'ontologie.

L'écriture d'un ensemble de formules logiques à l'aide d'un langage logique du premier ordre nécessite, en plus des opérateurs logiques, un alphabet qui constitue le vocabulaire de la théorie logique.

Pour une conceptualisation chaque élément du vocabulaire peut être associé soit aux éléments qui composent la définition extensionnelle de la conceptualisation (i.e. aux éléments de D et de R), soit aux éléments qui composent la définition intensionnelle de la conceptualisation (i.e. D et $\mathfrak{R}$). Dans le premier cas on parle d'un « modèle » de la théorie logique et dans le deuxième d'un « engagement ontologique ».

DÉFINITION 2.6.7 (**Modèle** ou structure extensionnelle de premier degré)

Soit L un langage logique du premier ordre, V le vocabulaire associé, et $\mathcal{S} = (D, R)$ *une structure relationnelle extensionnelle. Un* modèle*, nommé aussi* structure extensionnelle de premier degré*, pour L est un tuple* $M = (\mathcal{S}, I)$*, où I (fonction d'interprétation extensionnelle) est une fonction* $I : V \rightarrow D \cup R$ *qui associe chaque symbole du vocabulaire V à un élément de D ou à une relation extensionnelle de R.*

DÉFINITION 2.6.8 (**Engagement ontologique** ou structure intensionnelle de premier degré)

Soit L un langage logique du premier ordre, V le vocabulaire associé, et $C = (D, W, \mathfrak{R})$ *une conceptualisation. Un* engagement ontologique (ontological commitment)*, nommé aussi* structure intensionnelle de premier degré*, pour L est un tuple* $K = (C, \mathcal{I})$*, où* $\mathcal{I}$ *(fonction d'interprétation intensionnelle) est une fonction* $\mathcal{I} : V \rightarrow D \cup \mathfrak{R}$ *qui associe chaque symbole du vocabulaire V à un élément de D ou à une relation intensionnelle de* $\mathfrak{R}$*.*

EXEMPLE 2.6.4 Pour l'exemple 2.6.1 le vocabulaire V est composé par les noms des 5 relations : V = *{ Employé, Manager, Chercheur, coopère_avec, subordonné_à }* . Un modèle associe chacun des symboles du vocabulaire à un ensemble qui constitue l'extension de la relation ; par exemple, *Employé* $\rightarrow$ *{ I001, ..., I100 }*. Un engagement ontologique associe chacun des symboles du vocabulaire à une relation conceptuelle ; par exemple, *Employé* $\rightarrow$ *Employé''* .

Intuitivement, l'engagement ontologique peut être considéré comme une extension intensionnelle de la notion standard de modèle. En principe, pour un état du monde donné w, les éléments associés à un symbole dans le cadre d'un modèle M peuvent être obtenus en appliquant à la description de l'état du monde la fonction ρ qui définit la relation conceptuelle associée au même symbole par l'engagement ontologique correspondant. Il est évident que ces ensembles d'éléments (celui associé par le modèle M et celui obtenu à l'aide de la fonction ρ) peuvent être différents si, par exemple, l'état du monde w n'est pas compatible avec la définition donnée à la relation par la fonction ρ. On peut donc distinguer, parmi tous les modèles d'une théorie logique, ceux qui sont compatibles avec une conceptualisation – nommés « modèles intentionnels » de ceux qui ne sont pas compatibles.

DÉFINITION 2.6.9 (**Modèles intentionnels**)

Soit $C = (D, W, \mathfrak{R})$ *une conceptualisation, L un langage logique du premier ordre, V le vocabulaire associé, et* $K = (C, \mathcal{I})$ *un engagement ontologique. Un modèle* $M = (S, I)$*, avec* $S = (D, R)$*, est nommé* modèle intentionnel (intended model) *du langage L respectant l'engagement K si et seulement si les deux propriétés suivantes sont vérifiées :*

1. *Pour tous les symboles constants*[5] $t \in V$ *on a* $I(t) = \mathcal{I}(t)$
2. *Il existe un monde* $w \in W$ *tel que, pour chaque symbole prédicatif*[6] $v \in V$ *il existe une relation intensionnelle* $\rho \in \mathfrak{R}$ *satisfaisant* $\mathcal{I}(v) = \rho$ *et* $I(v) = \rho(w)$*.*

On note $I_K(L)$ *l'*ensemble des modèles intentionnels *du langage L qui respectent l'engagement ontologique K.*

La première condition exige que le modèle M et l'engagement ontologique K associent chaque symbole identifiant une constante au même élément de D. Dans l'exemple 2.6.1 il n'y

5. Un « symbole constant » est un symbole associé à un élément de D
6. Un « symbole prédicatif » est un symbole associé à un élément de R

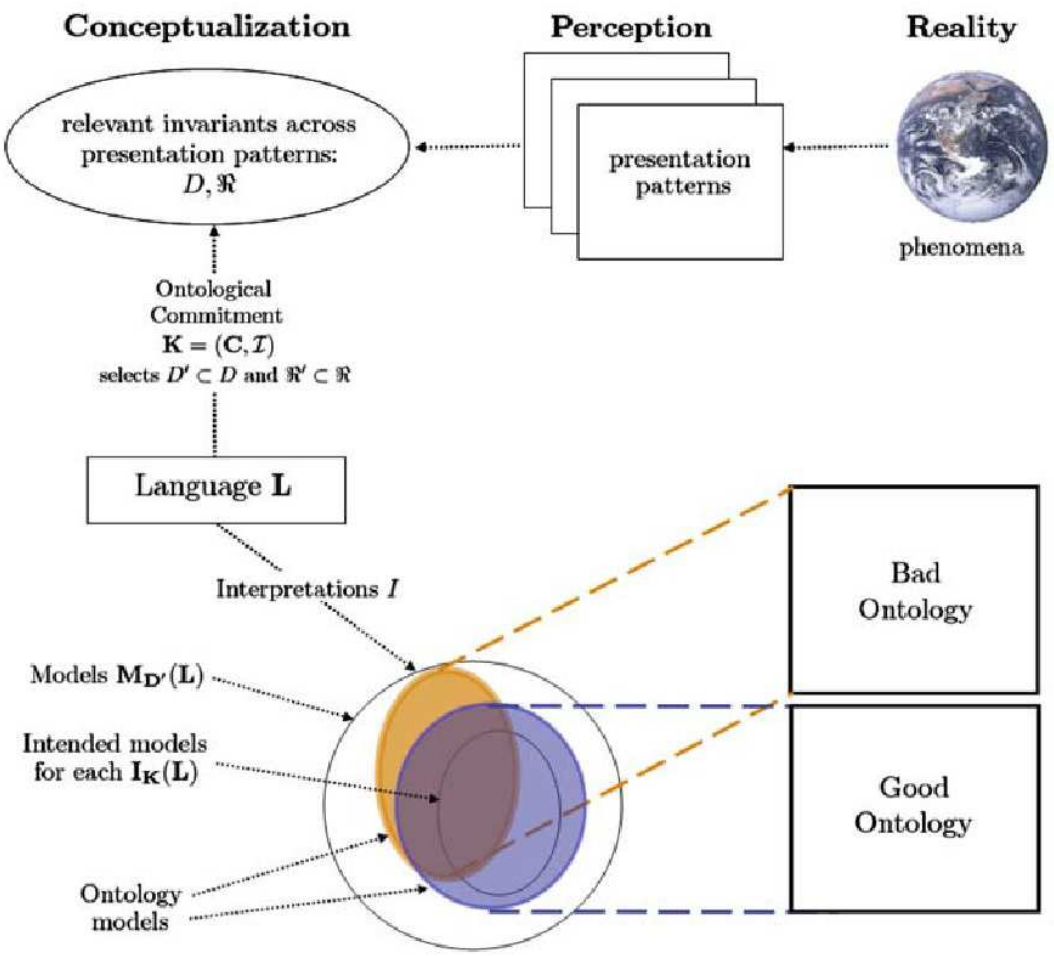

FIGURE 2.3: Illustration des relations entre une conceptualisation, le langage utilisé pour sa formalisation, les modèles intentionnels et l'ontologie (extrait de Guarino et al. (2009) [60]).

a aucun symbole identifiant une constante. La deuxième condition exige que l'extension associée à un symbole par le modèle M correspondant à l'état du monde w soit identique à celle obtenue en appliquant la fonction ρ (associée au symbole par l'engagement ontologique K) à la description de w. Par exemple, pour w_1, *I(Employé) = { I001, …, I100 } = Employé''*(w_1).

Avec ces précisions il est possible de définir l'ontologie comme une théorie logique conçue de sorte que tous ses modèles soient des modèles intentionnels.

DÉFINITION 2.6.10 (Ontologie)

Soit C une conceptualisation, L un langage logique du premier ordre de vocabulaire V, et un engagement ontologique K. Une ontologie O_K *est une théorie logique composée d'un ensemble de formules du L telles que l'ensemble de ses modèles approxime « au mieux » l'ensemble des modèles intentionnels du langage L selon K.*

La figure 2.3 illustre les relations entre les différentes notions introduites dans cette section : conceptualisation, engagement ontologique, modèles intentionnels et ontologie.

2.7 Conclusion

Dans ce chapitre, nous avons rappelé les définitions intuitives puis plus formelles de la notion d'ontologie telle qu'elle est utilisée en ingénierie des connaissances.

Comme celle-ci est un « moyen » opérationnel pour représenter et partager des connaissances, nous avons commencé par nous interroger sur les champs identifiés par les « données », les « informations » et les « connaissances », puis nous avons énuméré les composants de base d'une ontologie et brièvement présenté le langage OWL qui est maintenant le standard le plus utilisé pour représenter les différentes ontologies construites dans le cadre de cette thèse. La dernière partie de ce chapitre est consacrée aux définitions formelles qui renvoient à la fois aux propriétés structurelles et à la conceptualisation des connaissances. Une partie des notions et des notations associées à ces définitions (connaissances tacites et connaissances explicites, concepts primitif et concept défini, modèle intentionnel, interprétation, etc.) sont reprises dans la suite de cette thèse.

Dans le chapitre suivant, nous présentons les principes les plus importants des différentes méthodologies qui ont été proposées pour la construction des ontologies et nous analysons plus en détail les techniques qui ont été proposées pour l'apprentissage automatique des ontologies à partir de textes.

3

Construction d'ontologies : des approches manuelles à la construction automatique

Sommaire

3.1 Introduction

Initialement, la conception des ontologies était basée exclusivement sur l'expertise humaine, mais, durant cette dernière décennie, les besoins liés à la fois à leur diffusion et aux changements d'échelles des corpus disponibles ont conduit au développement d'approches de conception intégrant l'apprentissage automatique. Plusieurs termes sont utilisés dans la littérature pour identifier ces approches : construction automatique, construction semi-automatique ou construction basée sur de l'apprentissage. Dans le cadre de cette thèse, nous utilisons ces termes de façon interchangeable. Les approches qui utilisent l'apprentissage automatique se différencient en fonction des ressources qu'elles utilisent pour l'apprentissage [80] : documents textuels, bases de données, taxonomies, thésaurus, pages Internet, etc. Dans cette thèse, on se focalise sur les approches de construction automatique des ontologies à partir de textes.

Ce chapitre est structuré en deux parties. Dans la première partie, nous présentons plusieurs méthodologies qui ont été proposées pour guider et structurer le travail de construction manuelle d'une ontologie. Dans la deuxième partie, nous nous focalisons sur la construction automatique des ontologies à partir de textes. Nous insistons sur le rôle que l'apprentissage automatique peut avoir dans le processus de construction et sur les principales techniques d'apprentissage qui ont été proposées pour les différentes tâches du processus de construction. Nous présentons également différentes approches de construction automatique qui ont fait l'objet d'une comparaison plus approfondie dans le chapitre 4 de cette thèse.

3.2 Méthodologies de construction d'ontologies

En préambule, nous tentons de préciser les significations des termes « méthodologie », « méthode », « technique » et « tâche » qui sont souvent rencontrés dans les travaux sur la construction d'ontologies. Cette discussion s'appuie sur l'analyse publiée en 2001 par Gomez-Perez et al. [55], qui repose sur des définitions « standardisées » d'IEEE et que l'on retrouve en gestion de projets.

Dans ce cadre, la **méthodologie** définit la stratégie et le cadre général de la conception d'ontologies. Elle indique ce qui doit être fait (*quoi*), à quel moment (*quand*) et par qui (*qui*), mais n'indique pas précisément les aspects opérationnels de la mise en oeuvre. Une méthodologie est composée de techniques et de méthodes. La **méthode** est une procédure générale qui indique l'ordre dans lequel doivent être exécutés les différents processus qui la composent.

La **technique**, en tant que procédure utilisée pour accomplir un objectif, indique la manière dont une méthode doit être exécutée. Plusieurs techniques peuvent être associées à la même méthode.

Un **processus** est composé d'un ensemble d'activités définies chacune par un ensemble de tâches qui sont les unités élémentaires dans la gestion d'un projet. Chaque tâche représente un travail bien défini qui est assigné à une personne ou à un groupe de personnes.

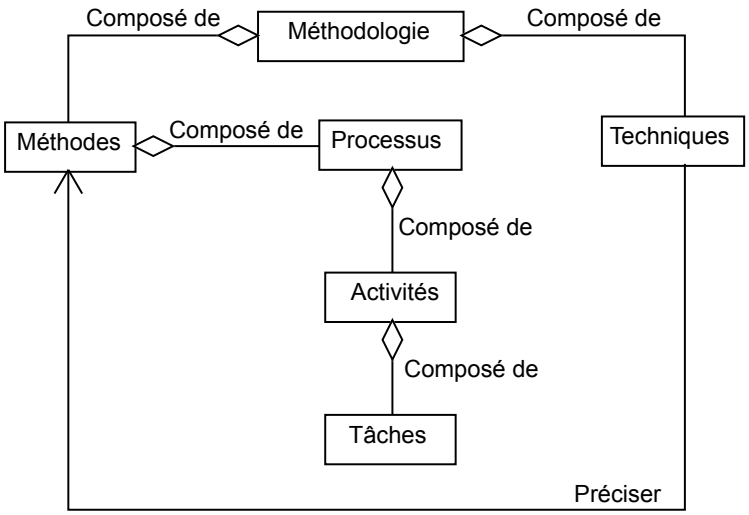

Figure 3.1: Représentation graphique (reprise de Gomez-Perez et al. [55]) des relations entre les termes méthodologie, méthode, activité et tâche.

3.2.1 Methontology

La construction manuelle d'ontologies est un processus long et complexe. Plusieurs méthodologies ont été définies pour structurer ce travail [44, 103, 128]. Dans cette section nous présentons en détail une des plus connues, nommée Methontology, et nous passons en revue les principes centraux de trois autres méthodologies plus récentes (On-To-Knowledge, DILIGENT et NeOn) en soulignant ce qu'elles ont apporté de novateur par rapport à Methontology.

D'après Suarez-Figueroa et al., 2008 [128] Methontology [28, 44, 55] est une méthodologie complète pour la construction d'une ontologie « indépendante [1] » sans modèle préalable.

Dans la construction d'une ontologie selon Methontology trois processus interdépendants interviennent (figure 3.2) : un processus de gestion, un processus de développement et un processus de support. Chaque processus est composé de plusieurs activités. Il y a trois activités de gestion du projet (planification, contrôle et assurance qualité), cinq activités de développement (spécification, conceptualisation, formalisation, implémentation et maintenance) et cinq activités de support (acquisition de connaissances, intégration, évaluation, documentation et configuration).

Methontology propose de commencer la construction de l'ontologie par l'activité de planification, durant laquelle on identifie les tâches qui doivent être exécutées, leur ordre, le temps et les ressources nécessaires à leur exécution.

Le processus de développement proprement dit peut commencer une fois la planification

1. Par ontologie indépendante (*single ontology*) on entend une ontologie qui ne dépend pas d'autres ontologies, par opposition aux réseaux d'ontologies (*networked ontologies*) qui peuvent être construites en suivant la méthodologie NeOn.

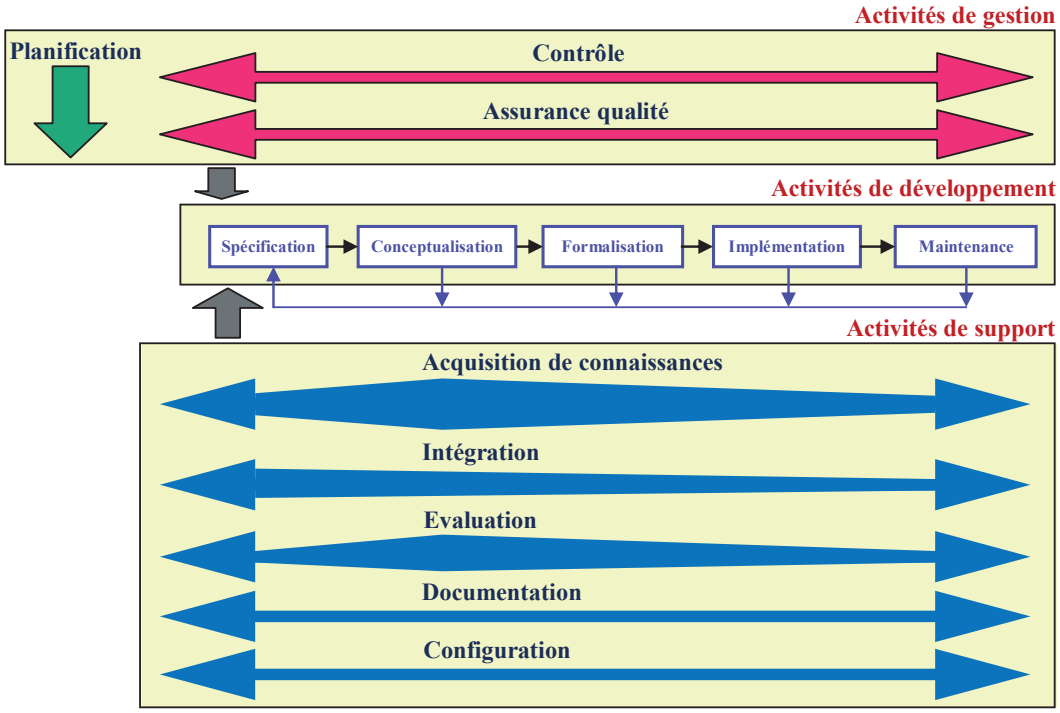

FIGURE 3.2: Les processus de Methontology et les activités qui les composent (image reprise de Gomez-Perez et al., 2001 [55]). Les flèches entre les activités de développement indiquent l'ordre d'exécution des activités et le fait qu'il est toujours possible de revenir à une étape précédente. Les activités de support sont représentées sur forme de flèches allongées : la longueur de la flèche indique la durée de l'activité et son épaisseur indique l'intensité de l'activité (cela permet d'indiquer, par exemple, que l'activité d'acquisition de connaissances connaît la plus forte intensité durant la conceptualisation).

finalisée. Les cinq activités du processus de développement sont exécutées en ordre séquentiel (figure 3.2), et il est possible de revenir en arrière à tout moment pour compléter ou corriger ce qui a été déjà construit. Les activités de contrôle et d'assurance qualité, tout comme les cinq activités de support, démarrent en même temps que le processus de développement et durent aussi longtemps que lui. Ces sept activités s'exécutent en parallèle.

Le processus de développement commence par l'activité de spécification durant laquelle on identifie l'objet de l'ontologie, ses utilisateurs, ses utilisations et le degré de formalisation requis. Il continue par l'étape la plus importante, la conceptualisation, durant laquelle les connaissances identifiées dans le cadre de l'activité d'acquisition de connaissances sont structurées pour obtenir un modèle conceptuel représenté de façon semi-formelle. Ce modèle est ensuite formalisé, implémenté dans un langage adapté et maintenu pour correspondre aux éventuelles évolutions des spécifications.

Pour les activités d'acquisition des connaissances, d'intégration et d'évaluation, l'effort le plus important est réalisé au début du processus de développement et notamment durant l'activité de conceptualisation (figure 3.2). Pour l'acquisition des connaissances Methontology recommande, sans entrer dans les détails, l'utilisation de différentes techniques comme

les réunions de brainstorming, les interviews d'experts ou l'analyse de textes. Dans l'activité d'intégration on identifie les parties des ontologies déjà existantes qui peuvent être réutilisées et intégrées dans la nouvelle ontologie. L'évaluation de l'ontologie, qui doit être réalisée tout au long du processus de développement, vise à identifier et à éliminer au plus tôt, depuis l'étape de conceptualisation, les problèmes d'incohérence, d'incomplétude ou de redondance.

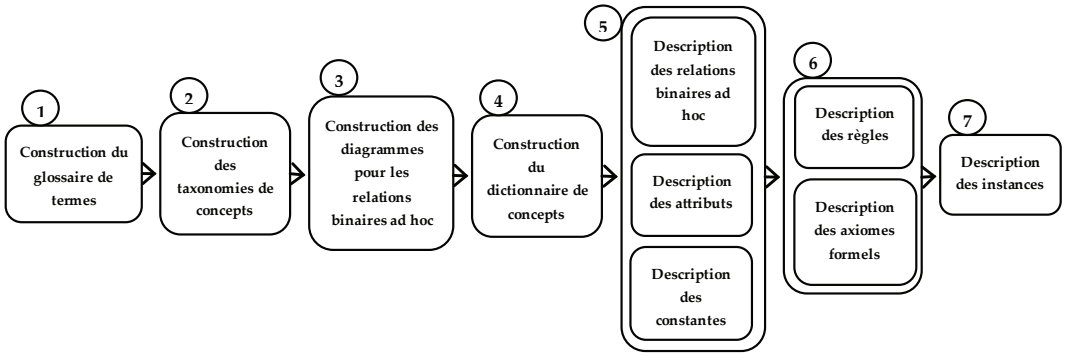

FIGURE 3.3: Les tâches qui composent l'activité de conceptualisation de Methontology (figure adaptée de Gomez-Perez et al., 2001 [55]).

Pour l'activité de conceptualisation les auteurs de Methontology ont décrit en détail la succession de sept tâches à réaliser (figure 3.3) :

1. La **construction d'un glossaire de termes**. Ce glossaire doit contenir tous les termes qui font référence à des connaissances qui doivent faire partie de l'ontologie. Ces termes peuvent être des noms ou des verbes identifiant des concepts, des instances, des attributs, des constantes ou des relations sémantiques.
2. La **construction des taxonomies de concepts**. Dans un premier temps les termes du glossaire qui sont synonymes et qui font référence à un même concept sont regroupés sous un seul intitulé qui sert d'étiquette pour le concept. Ensuite, les concepts « étroitement liés » sont regroupés par catégories. Les concepts sont considérés comme « étroitement liés » entre eux s'ils peuvent être tous structurés dans une taxonomie sans faire appel à des nouveaux concepts plus abstraits ou généraux qui ne sont pas directement obtenus à partir de la conceptualisation des termes présents dans le glossaire.
3. La **construction des diagrammes pour les relations binaires ad hoc**. Ces relations sont identifiées à partir de syntagmes verbaux. Le nom de la relation est identifié à l'aide du regroupement des verbes synonymes sous un seul intitulé. La relation porte sur les concepts identifiés dans l'étape précédente correspondant au sujet et au complément d'objet présents dans le syntagme verbal.
4. La **construction d'un dictionnaire de concepts**. Toutes les informations relatives aux concepts (leur sémantique, leurs attributs, leurs instances, etc.) sont regroupées dans le dictionnaire de concepts.
5. La **description des relations binaires ad hoc, des attributs et des constantes**. Pour chaque relation qui a été identifiée dans l'étape précédente il faut préciser son étiquette, les concepts qu'elle relie, sa cardinalité et s'il existe une relation inverse. Les attributs

peuvent concerner un concept (quand toutes les instances du concept ont la même valeur pour l'attribut) ou les instances d'un concept (quand chaque instance peut avoir sa propre valeur pour l'attribut). Dans la description détaillée des attributs il faut préciser leur type, leur nom, les concepts auxquels ils sont associés, leur domaine de valeurs, leur cardinalité et, dans le cas des attributs des concepts, leur valeur. Les constantes sont des valeurs qui ne changent pas dans le temps et qui peuvent être utilisées dans une ontologie comme limites pour le domaine de valeurs des attributs, par exemple. Dans cette étape il faut préciser pour chaque constante, si possible, son nom, son type, sa valeur, son unité de mesure et les attributs qui l'utilisent dans leur définition.

6. La **description des règles et des axiomes formels**. Il faut identifier et décrire formellement les règles et les axiomes de l'ontologie. Pour chaque règle et axiome, en plus de leur définition formelle, Methontology propose d'indiquer leur nom et les noms des concepts, des attributs et des relations qui y sont impliqués.
7. La **description des instances**. Pour chaque instance mentionnée dans le glossaire de termes il faut préciser son nom, les concepts qui lui sont associés et les valeurs de ses attributs.

Plusieurs ontologies ont été construites en suivant cette méthodologie (ex. [28, 55]) et aujourd'hui encore elle reste une référence bien qu'elle ait été proposée il y a plus de quinze ans. La plupart des auteurs considèrent que le mérite principal de cette méthodologie est d'avoir précisé le processus de transformation de connaissances informelles et implicites en connaissances formelles et explicites exploitables pour construire une ontologie.

3.2.2 On-To-Knowledge

Proposée peu de temps après Methontology, la méthodologie On-To-Knowledge [123] met au centre du processus de construction de l'ontologie les usages applicatifs. Les ontologies construites à l'aide de cette méthodologie sont donc spécialisées et dépendantes de l'application pour laquelle elles ont été conçues [128], alors que les ontologies construites en suivant Methontology sont généralement plus génériques et indépendantes d'une application spécifique donnée *a priori*.

Comme la figure 3.4 l'indique, On-To-Knowledge propose un processus de construction d'une ontologie en cinq étapes :

1. **L'étude de faisabilité**. Dans cette première étape, qui ne fait pas partie de la construction proprement dite de l'ontologie, il faut analyser tous les aspects de l'application dans laquelle l'ontologie sera utilisée. L'objectif est d'identifier la problématique, le domaine concerné et la manière dont l'ontologie peut contribuer à une solution.
2. **La phase « kickoff »**. Sur la base des résultats de l'étude de faisabilité, il faut constituer un document précisant les spécifications de l'ontologie à construire (son domaine et son objectif), les règles de conception à respecter (par exemple, les conventions de nommage), les sources d'informations disponibles (livres, interviews, etc.), les utilisateurs et les cas d'utilisation envisagés, les applications supportées et le questionnaire de compétences. Durant cette étape, il faut aussi rechercher des ontologies déjà développées qui correspondent aux spécifications.

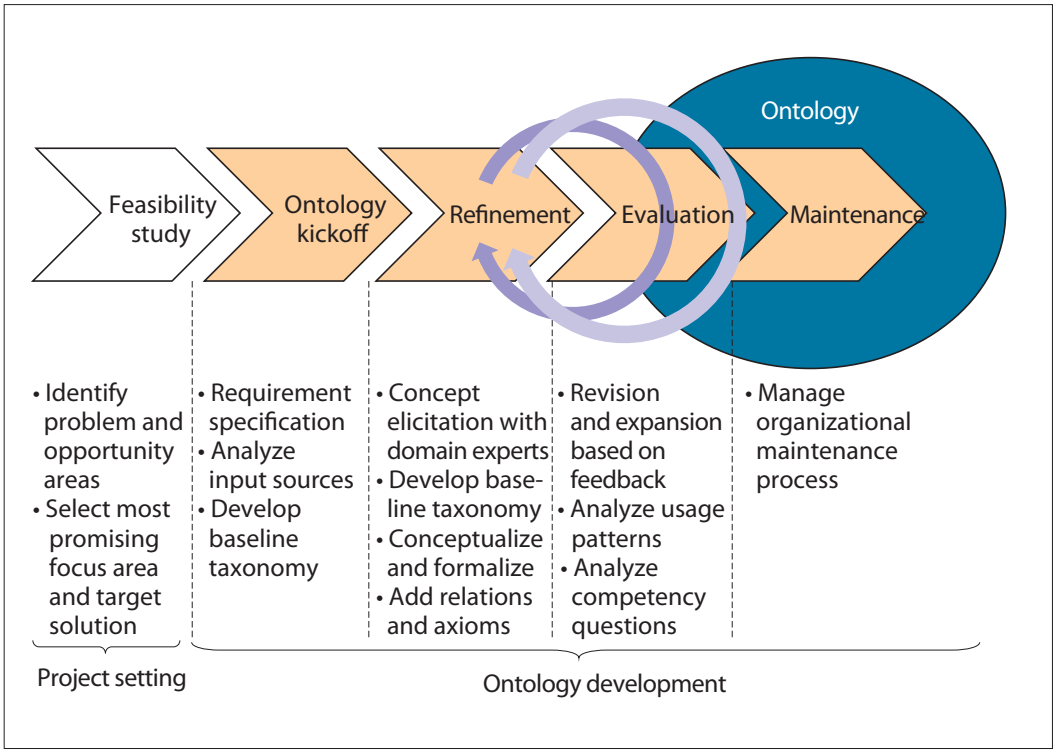

FIGURE 3.4: Les processus de la méthodologie On-To-Knowledge (image reprise de Staab et al., 2001 [123]).

3. **Le raffinement**. C'est dans cette étape que l'ontologie est effectivement construite. Dans un premier temps, les spécifications de l'ontologie sont enrichies et explicitées à l'aide des experts. Plus concrètement, il s'agit de préciser les concepts qui doivent faire partie de l'ontologie, d'organiser ces concepts dans une taxonomie, d'identifier les relations non taxonomiques et les axiomes qui doivent faire partie de l'ontologie. Dans un deuxième temps ces résultats intermédiaires sont formalisés pour obtenir l'ontologie.

4. **L'évaluation**. L'objectif de l'évaluation est d'établir si l'ontologie construite respecte les spécifications et si, lors de son utilisation dans le cadre de l'application, les résultats obtenus sont conformes aux attentes. Si nécessaire, il est possible de revenir dans l'étape de raffinement pour corriger ou compléter l'ontologie.

5. **La maintenance de l'ontologie**. Les auteurs d'On-To-Knowledge indiquent qu'il est nécessaire d'établir clairement, dès la fin du processus de construction, qui est responsable de la maintenance de l'ontologie et de quelle manière celle-ci doit être effectuée.

3.2.3 DILIGENT

L'idée novatrice de la méthodologie DILIGENT [103] est la construction collaborative d'une ontologie partagée par une communauté étendue d'utilisateurs [2].

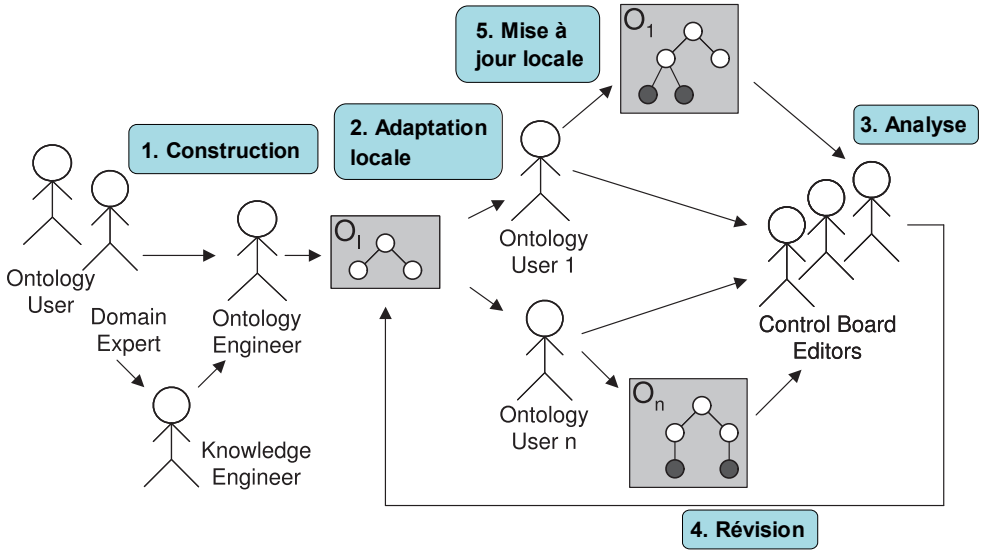

FIGURE 3.5: Les étapes du processus de construction proposé par DILIGENT (image reprise de [103]).

La problématique centrale est la maintenance de la trace des argumentations qui ont conduit à des changements dans l'ontologie. Le processus de construction proposé par DILIGENT se décompose en cinq étapes (figure 3.5) :

1. **Construction**. Une version initiale de l'ontologie est construite par un groupe restreint d'experts, d'utilisateurs et d'ingénieurs. Cette ontologie peut ne pas être complète. DILIGENT ne précise pas le processus de construction. Une fois construite, cette version initiale de l'ontologie est rendue publique.
2. **Adaptation locale**. Une fois que la version initiale de l'ontologie est disponible, les utilisateurs commencent à l'utiliser et à la modifier en fonction de leurs besoins. Toutes les modification apportées sont motivées et envoyées à un comité central de contrôle.
3. **Analyse**. Le comité central de contrôle analyse toutes les modifications qui ont été apportées à l'ontologie par les utilisateurs et précisent celles qui peuvent être acceptées. Ils précisent aussi ce qui doit faire partie de l'ontologie partagée, commune à tous les utilisateurs, et ce qui ne doit pas en faire partie.
4. **Révision**. L'ontologie partagée est modifiée sur la base de l'analyse réalisée dans l'étape précédente et la nouvelle version est rendue publique.
5. **Mise à jour locale**. Les utilisateurs utilisent la nouvelle version de l'ontologie.

2. Par communauté étendue d'utilisateurs nous entendons un group d'acteurs dispersés géographiquement et qui utilisent l'ontologie pour des applications différentes.

3.2.4 NeOn

La méthodologie NeOn [128] a été proposée en 2008 et, à notre connaissance, elle est la plus récente. Elle vise à répondre à un nouveau paradigme apparu dans le développement des ontologies :

- la construction d'ontologies basée sur la réutilisation, la restructuration, la modification et l'adaptation des différentes sources de connaissances déjà existantes (des ontologies ou des patrons pour la conception d'ontologies) ;
- la réutilisation dynamique [3] dans une ontologie des concepts définis dans d'autres ontologies disponibles en ligne. Ce phénomène a conduit à l'apparition des « réseaux d'ontologies » ; ces ontologies sont interconnectées et interdépendantes ;
- le rôle de plus en plus important de la collaboration et de l'argumentation durant la construction des ontologies. Des acteurs géographiquement éloignés, avec des cultures et des objectifs différents peuvent collaborer pour la construction d'une ontologie.

Cette situation contraste avec la manière dont les ontologies étaient développées quand Methontology a été proposée : un groupe restreint d'ingénieurs et d'experts identifie les connaissances qui doivent faire partie de l'ontologie, puis les structure et les formalise pour obtenir une ontologie qui est indépendante des autres.

La méthodologie NeOn a été conçue pour répondre à ce nouveau paradigme et à la diversité des scénarios possibles pour la construction d'une ontologie. C'est une méthodologie complexe qui couvre 59 types d'activités qui, en fonction du scénario de construction envisagé, peuvent faire partie du processus de construction de l'ontologie. La figure 3.6 présente une classification de ces activités.

Une présentation détaillée de la méthodologie NeOn dépasse le cadre de cette thèse. La figure 3.7 résume les points pour lesquels cette méthodologie est novatrice. On peut retenir notamment la proposition de règles très détaillées pour faciliter la création du document contenant les spécifications de l'ontologie à construire, pour la planification des différentes activités et pour la réutilisation de différents types de ressources dans le processus de développement.

3.3 Construction semi-automatique d'ontologies

Comme nous l'avons mis en évidence au début de la Section 3.2, le processus de construction d'une ontologie peut être analysé selon au moins deux aspects :

1. du point de vue des activités et des tâches qui le composent ou
2. par rapport aux techniques qui sont utilisées pour l'exécution des différentes tâches.

Dans cette section, nous analysons d'abord l'automatisation du processus de construction d'une ontologie par rapport aux activités et aux tâches qui le composent. Dans un deuxième temps, nous présentons les différentes techniques utilisées par les approches pour automatiser les tâches de l'activité de conceptualisation – l'activité centrale du processus de développement. Plus précisément, nous nous intéressons aux techniques pour l'extraction d'éléments

3. La définition de ces concepts n'est pas reprise de l'ontologie originale et intégrée de façon statique dans l'ontologie. L'ontologie peut contenir en revanche une référence vers l'adresse de l'ontologie dans laquelle le concept est défini.

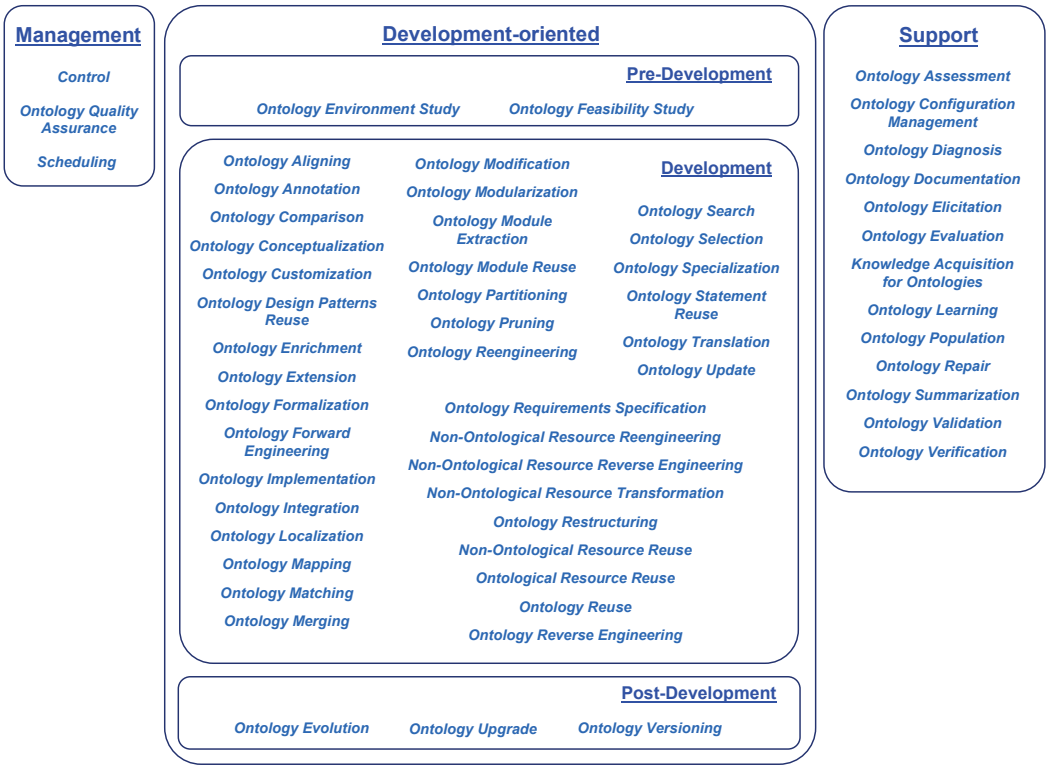

FIGURE 3.6: Classification des 59 activités qui peuvent faire partie du processus de développement d'une ontologie proposé par NeOn (image reprise de [127]).

significatifs à partir de textes et la composition de l'ontologie à partir de ces éléments. Enfin, nous présentons synthétiquement sept approches opérationnelles pour la construction automatique d'ontologies à partir de textes.

3.3.1 Le rôle de l'automatisation dans le processus de construction

La présentation de différentes méthodologies dans la section précédente a permis de souligner que le processus de construction d'une ontologie intègre un nombre très important d'activités et de tâches et ce, de l'analyse du problème et de la constitution des spécifications que doit respecter l'ontologie jusqu'à son évaluation et sa maintenance.

La construction automatique des ontologies est encore un domaine récent et, à notre connaissance, aucune des approches qui ont été proposées ne prend en compte l'automatisation de toutes les tâches de toutes les activités du processus de construction. Il existe néanmoins des approches (par exemple, Text2Onto [27]) qui proposent une construction partiellement automatisée, si on leur fournit en entrée les ressources contenant toutes les informations nécessaires. Nous reviendrons au chapitre 4 sur la qualité de leurs résultats.

Blomqvist, 2009 [13] a regroupé en six activités les 28 tâches et sous-tâches qui, dans son interprétation, doivent être exécutées pour la construction d'une ontologie. Elle a ensuite analysé le nombre d'approches de construction automatique qui proposent l'automatisation

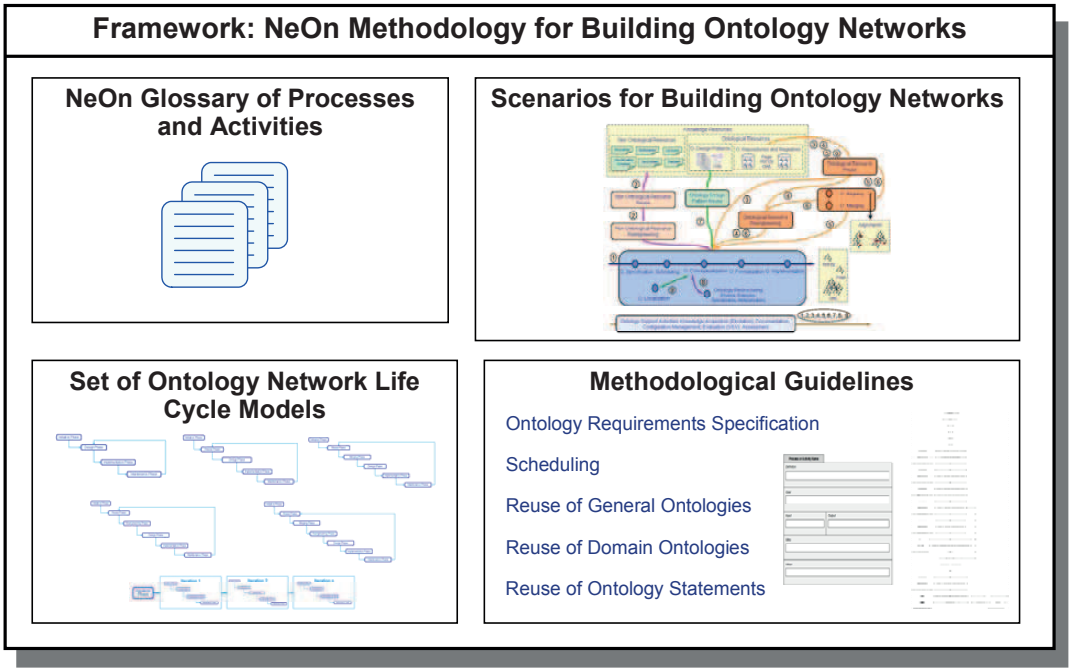

Figure 3.7: Vue globale des principaux points forts de la méthodologie NeOn (image reprise de [127]).

des différentes tâches (figure 3.8). Son analyse a montré que des approches automatiques existent pour 19 des 28 tâches, mais que ces approches permettent l'obtention de résultats considérés comme satisfaisants seulement pour 4 des 28 tâches. Elle a notamment mis en évidence qu'il n'existe pas d'approches automatiques pour les tâches qui concernent :

- la constitution des spécifications ;
- l'identification et l'évaluation des ressources qui sont utilisées pour l'apprentissage de l'ontologie ;
- l'extraction d'éléments complexes comme les axiomes et la hiérarchisation qui peuvent exister entre les relations sémantiques ;
- l'utilisation des connaissances de base sur l'objectif de l'ontologie : la résolution des conflits qui peuvent apparaître lors de la composition de l'ontologie ; la restructuration de l'ontologie ; l'ajustement du niveau d'abstraction ; le nettoyage et l'ajout de connaissances manquantes dans l'ontologie.

3.3.2 Approches pour l'extraction et la composition d'éléments significatifs à partir du texte

La principale difficulté dans la construction d'une ontologie est l'identification des connaissances qui doivent en faire partie. Les approches pour la construction automatique des ontologies à partir de textes se veulent une solution efficace pour ce problème. Ces approches sont basées sur l'hypothèse suivante : en identifiant et en assemblant des éléments significatifs du texte, il est possible d'extraire et de rendre formelles et explicites les connaissances

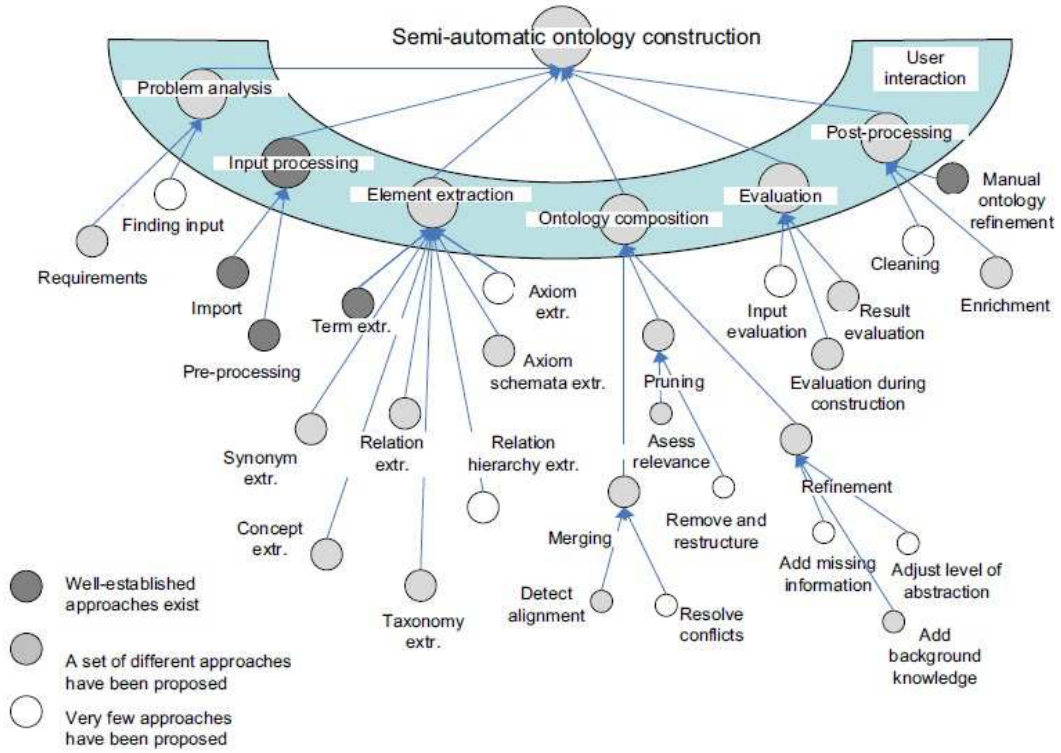

FIGURE 3.8: Les tâches qui composent le processus de construction automatique d'une ontologie. Mise en évidence des tâches pour lesquelles il n'existe aucune approche automatique (image reprise de [13]).

que le texte contient. Parmi les éléments significatifs qui peuvent être extraits à partir de ressources textuelles les plus connus sont les termes, les relations lexicales et syntaxiques et les contextes d'apparition de termes (dans une structure qui correspond à un patron, dans le voisinage d'autres termes, etc.). A partir de ces éléments, les approches automatiques essayent d'obtenir, en utilisant différentes techniques, les éléments constitutifs d'une ontologie, notamment les concepts, les relations de subsomption entre les concepts, les relations sémantiques, les instances et les axiomes.

Dans cette section, nous parcourons rapidement différentes techniques utilisées pour l'extraction d'éléments significatifs des textes et leur transformation en éléments constitutifs de l'ontologie.

3.3.2.1 Extraction de termes et formation de concepts

L'extraction des termes dans le cadre des approches pour la construction automatique des ontologies a pour objectif l'identification des termes qui sont présents dans le corpus d'entrée et qui font référence à des concepts, à des relations ou à des instances qui doivent faire partie de l'ontologie à construire. Un terme peut être simple ou composé de plusieurs mots.

Deux types d'approches ont été développées pour l'extraction de termes à partir de textes

[20] : les approches linguistiques et les approches statistiques. Ces approches peuvent évidement être utilisées conjointement (approches hybrides).

Les approches linguistiques sont basées sur l'analyse de la structure grammaticale des phrases qui composent le texte et sur l'identification du rôle syntactique (sujet, objet direct, etc.) et morpho-syntactique (nom, verbe, adjectif) de chaque mot ou groupement de mots. Cette analyse permet l'identification d'expressions nominales (« le composant chimique »), verbales (« s'enflamme au contact de l'oxygène ») ou adjectivales (« immédiatement disponible »). La stratégie la plus souvent utilisée dans les approches de construction d'ontologies considère que les expressions nominales font référence à des concepts ou à des instances.

Les approches statistiques sont basées sur l'analyse des fréquences d'apparition des termes dans le texte. Avant de déterminer ces fréquences, un « nettoyage » du texte est préalablement effectué : les mots vides (prépositions, conjonctions et certains verbes) sont supprimés, les mots composant le texte sont radicalisés (ils sont mis dans leur forme canonique – les verbes à l'infinitif, les noms au singulier, etc.). Pour choisir les termes qui vont être extraits, il est courant d'utiliser différentes mesures issues du domaine de la recherche d'informations, la plus connue étant certainement TF-IDF (Term Frequency-Inverse Document Frequency).

Pour extraire uniquement les termes qui font référence à des concepts vraiment intéressants pour l'ontologie certaines approches implémentent différents filtres. On peut citer par exemple ceux de TermExtractor [118]. Un de ses filtres considère que plus un terme est composé d'un nombre important de mots, plus il y a des chances qu'il fasse référence à des concepts très spécifiques de l'ontologie ; un autre compare la fréquence d'apparition du terme dans les textes pris en entrée à sa fréquence d'apparition dans des textes qui concernent un autre domaine.

La stratégie la plus simple utilisée pour construire des concepts à partir des termes extraits considère que chaque terme constitue un concept de l'ontologie. Elle a des limitations évidentes – les termes synonymes génèrent des concepts équivalents et les termes polysémiques des concepts mal définis. Pour y palier, certaines approches groupent les termes synonymes, considérant qu'ils font référence au même concept, et désambiguïsent les termes polysémiques.

3.3.2.2 Classification des termes et extraction de relations de subsomption

Dans la plupart des approches, l'identification des relations de subsomption entre les concepts est basée sur l'identification de relations lexicales ou contextuelles entre les termes qui désignent les concepts.

Deux types de techniques sont principalement utilisées pour identifier des relations entre les termes d'un texte [20, 95] : des techniques structurelles et des techniques contextuelles. En pratique, elles peuvent être combinées.

Dans les techniques structurelles, l'identification des relations est basée sur l'analyse de la structure des termes, i.e. :

- leur syntaxe (par exemple, le terme « ontologie de domaine » est une spécialisation du terme « ontologie » parce que ce dernier est inclus syntaxiquement dans le premier) ;
- leur morphologie (par exemple, le terme « cellule sanguine » est une variation du terme « cellule du sang ») ;

– le sens des mots qui composent le terme (les sens des termes et les relations entre les sens sont identifiés à l'aide de ressources comme WordNet).

Dans les techniques contextuelles l'identification des relations est basée sur l'analyse du contexte d'apparition des termes dans le texte. En fonction du type d'analyse proposé on peut distinguer deux sous-types complémentaires de techniques contextuelles : les techniques distributionnelles et les techniques à base de patrons.

Les techniques distributionnelles sont basées sur l'hypothèse distributionnelle de Harris selon laquelle il est possible d'induire la proximité sémantique de deux mots et les relations entre les concepts associés aux mots à partir de l'analyse syntactique des contextes d'apparition [63].

Les techniques à base de patrons font l'hypothèse selon laquelle l'existence d'une relation de subsomption entre deux concepts est prouvée par la présence dans le texte d'au moins une expression contenant des termes associés aux concepts et correspondant à certains patrons. Ces patrons peuvent combiner des éléments lexicaux, grammaticaux et paralinguistiques (par exemple, marques de ponctuation). Par exemple, le patron *NP tel que NP, NP, NP ... et NP* appliqué à l'expression « des fruits tels que les pommes et les oranges » permet de détecter les relations de subsomption « pomme *isa* fruit » et « orange *isa* fruit ».

Les patrons utilisés pour identifier les relations de subsomption peuvent être prédéfinis ou appris durant la construction de l'ontologie.

3.3.3 Approches de construction automatique

Dans cette section nous présentons synthétiquement sept approches de construction (semi) automatique associés à des outils logiciels opérationnels : Terminae, TextToOnto / Text2Onto, Sprat, Asium, OntoLearn, OntoGen et OntoLT.

3.3.3.1 Terminae

L'approche Terminae [6, 7] (figure 3.9) propose un outil logiciel qui assiste l'utilisateur dans la construction de l'ontologie en s'appuyant sur les résultats des outils de traitement automatique des langues (extracteur de termes, analyseur syntaxique, détecteur de synonymie, etc.). Lors de la conceptualisation, l'utilisateur identifie et structure manuellement les éléments constitutifs de l'ontologie (concepts, relations, instances) à partir des éléments linguistiques (termes, relations lexicales et syntaxiques) extraits automatiquement à partir du corpus.

L'utilisateur associe d'abord des « termino-concepts » [4] à chaque terme extrait à partir du corpus. Ensuite, il structure manuellement les termino-concepts en un « réseau termino-ontologique ». Dans la dernière étape, le réseau termino-ontologique est formalisé en une ontologie. Pendant la construction Terminae maintient une traçabilité entre les concepts de l'ontologie en cours de construction et le corpus (dans la figure 3.1 les lignes pointillées indiquent la traçabilité).

4. Les « termino-concepts » représentent les différents sens du terme.

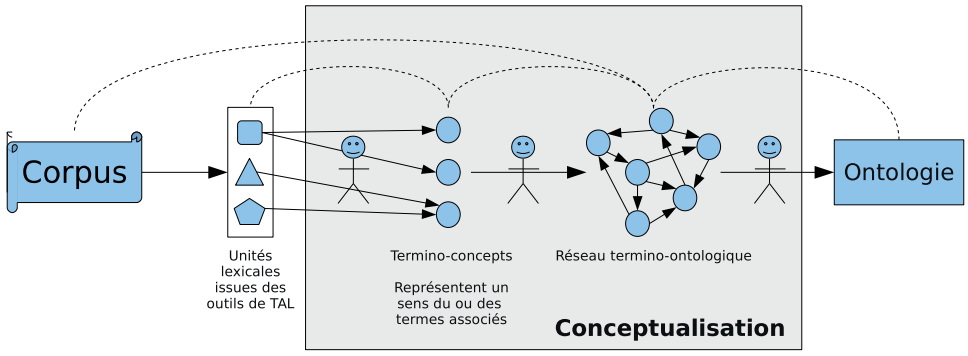

Figure 3.9: Les étapes du processus de construction d'une ontologie proposé par Terminae (image reprise de [87]).

3.3.3.2 Text-To-Onto et Text2Onto

L'approche Text2Onto [27] a repris et enrichi une approche précédente, nommée Text-To-Onto [79, 82]. L'outil logiciel associé à l'approche Text-To-Onto propose des modules pour l'extraction de concepts, d'instances, de relations taxonomiques et d'autres relations sémantiques. Chaque module permet de choisir parmi plusieurs algorithmes qui implémentent des techniques différentes. Les concepts sont identifiés soit à l'aide d'une technique statistique (les termes ayant une fréquence d'apparition dans le texte supérieure à un seuil sont considérés comme des concepts), soit à l'aide d'une technique linguistique grâce à un extracteur de termes intégré. Les instances sont identifiées à l'aide de techniques linguistiques et de patrons prédéfinis. L'utilisateur peut valider manuellement les concepts et les instances extraits. Deux algorithmes sont proposés pour l'identification des relations taxonomiques. Le premier utilise une technique distributionnelle : l'analyse de concepts formels (ACF). Le deuxième algorithme combine l'utilisation de patrons prédéfinis, de ressources externes (WordNet) et d'heuristiques exploitant la structure des termes. Les relations non taxonomiques sont extraites à l'aide de patrons prédéfinis ou par un algorithme classique d'extraction de règles d'association (dans ce dernier cas, le nom des relations doit être précisé manuellement).

L'outil associé à l'approche Text2Onto (figure 3.10) a proposé quelques nouveaux algorithmes et a repris une partie de ceux diponibles dans Text-To-Onto. Il apporte en plus la possibilité de combiner les résultats des différents algorithmes proposés pour la même tâche (par exemple, on peut utiliser en même temps tous les algorithmes pour l'extraction de relations taxonomiques) et une nouvelle architecture centrée sur l'évolution de l'ontologie construite en fonction de l'évolution du corpus. Text2Onto utilise l'architecture GATE [31] pour l'analyse des textes.

Pour l'extraction des concepts Text2Onto propose un seul algorithme, basé sur une technique linguistique. Il permet le calcul d'une mesure de confiance entre 0 et 1 pour les concepts extraits. Cette mesure combine différents scores statistiques, en fonction du choix de l'utilisateur : la TF-IDF, l'entropie, etc. Pour l'extraction des relations taxonomiques Text2Onto propose trois algorithmes : un premier basé sur des patrons d'extraction « à la Hearst », un deuxième basé sur WordNet et un troisième qui utilise une technique structurelle basée sur

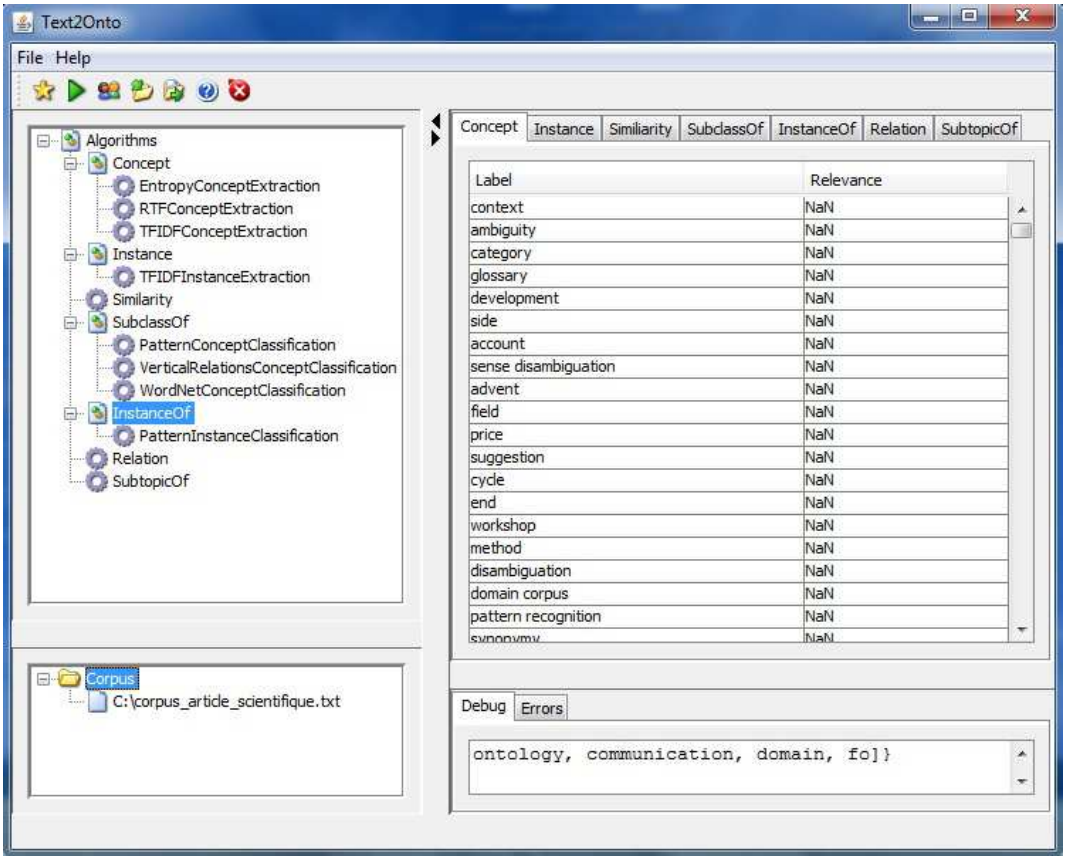

FIGURE 3.10: L'outil associé à l'approche Text2Onto.

l'inclusion syntaxique des termes. Text2Onto propose aussi un algorithme basé sur une technique linguistique pour l'extraction d'instances et un algorithme basé sur des patrons pour l'identification des relations *« instance-of »*.

Les éléments extraits par les différents algorithmes sont combinés automatiquement et le résultat peut être exporté sur la forme d'une ontologie OWL. L'utilisateur peut accepter ou rejeter les résultats obtenus par les différents algorithmes, mais il ne peut pas les modifier ou revenir aux parties du corpus dont ils sont issus.

3.3.3.3 Sprat

L'approche Sprat [85] utilise une série de patrons prédéfinis ; elle ajoute dans l'ontologie qui est en train d'être construite tous les concepts et les instances auxquels sont supposés faire référence les termes qui apparaissent dans des expressions textuelles correspondant à un patron prédéfini. En fonction du patron identifié, l'outil ajoute à l'ontologie, en même temps que le concept ou l'instance, la relation qui le relie au reste de l'ontologie.

Cette approche a l'avantage d'être automatisée. Mais son principal désavantage d'un point de vue opérationnel réside dans la difficulté de trouver un corpus contenant un grand nombre d'expressions qui correspondent aux patrons prédéfinis.

3.3.3.4 Asium

L'approche Asium [41, 42, 96, 97] fournit en sortie des hierarchies de concepts en utilisant une analyse syntaxique pour extraire les termes qui sont sujets ou compléments d'objet dans le texte. Ces termes qui constituent le premier niveau de concepts de la hiérarchie sont regroupés de façon non supervisée pour former la hiérarchie de concepts. Chaque classe de concepts, formée par le regroupement des concepts du niveau inférieur, constitue un nouveau concept. Ce regroupement dépend d'une dissimilarité qui est basée sur l'hypothèse que « les mots qui apparaissent ensemble après la même préposition et avec le même verbe représentent le même concept » [42]. La dissimilarité de type Jaccard prend en compte le nombre d'éléments en commun entre deux classes, la fréquence des éléments communs et la cardinalité des classes.

L'approche Asium permet aussi d'identifier automatiquement les concepts qui sont reliés par des relations sémantiques pour lesquelles l'utilisateur doit fournir préalablement des exemples [96]. Plus précisément, l'utilisateur doit choisir les relations sémantiques qu'il veut avoir dans l'ontologie et fournir des phrases exemples qui expriment ces relations. La syntaxe des phrases exemples est analysée et exprimée sur une forme abstraite à l'aide de « ASA » (« Abstraction of the Syntactic Analysis »), un algorithme spécifique à Asium. En complément, un outil d'apprentissage inductif, Propal, apprend des règles qui permettent de reconnaître dans le texte les phrases qui ressemblent aux exemples et qui sont censées exprimer la même relation sémantique mais entre d'autres concepts que ceux des exemples.

3.3.3.5 OntoLearn

Par rapport aux trois approches précédemment présentées, l'approche OntoLearn [94, 134] semble moins structurée et ne propose pas un outil intégrant toutes les fonctionnalités pour la construction automatique d'une ontologie. OntoLearn regroupe un ensemble d'outils indépendants pour l'extraction d'éléments d'ontologie à partir de textes (concepts, relations taxonomiques et autres relations sémantiques), et les résultats des outils doivent être assemblés manuellement pour obtenir l'ontologie.

L'outil pour l'extraction de concepts, TermExtrator, combine des techniques statistiques et linguistiques avec un grand nombre de paramétrages. Et, pour extraire seulement des termes spécifiques à un domaine, il peut comparer le corpus spécifique du domaine avec des corpus génériques. Un autre outil, GlossExtractor, permet d'identifier des définitions pour des termes en les cherchant sur Internet.

Pour l'extraction de relations taxonomiques OntoLearn propose deux solutions. La première commence par le regroupement des concepts en petites hiérarchies sur la base de l'inclusion lexicale. Chaque terme composé de ces hiérarchies est ensuite analysé de façon automatique (outil SSI) en associant à chaque terme un réseau sémantique construit sur la base de WordNet ; le réseau a comme noeuds les mots qui composent le terme associé. Les différents réseaux sémantiques sont ensuite combinés manuellement ; OntoLearn fait l'hypothèse que leur intersection met en évidence des relations taxonomiques qui permettent de regrouper les petites hiérarchies dans une seule hiérarchie globale. La principale limitation de cette solution est le manque d'automatisation de l'étape de combinaison des réseaux sémantiques.

La deuxième solution propose, à l'aide de l'outil WCL System, l'extraction de relations taxonomiques sur la base de l'analyse des définitions des termes identifiant les concepts.

L'analyse est réalisée à l'aide des patrons prédéfinis.

Pour l'extraction de relations non taxonomiques OntoLearn s'appuie, comme Asium, mais avec un algorithme d'induction différent, sur de l'apprentissage inductif de règles permettant d'identifier des concepts reliés par des relations prédéfinies.

3.3.3.6 OntoGen

L'approche OntoGen [46, 47] permet la construction semi-automatique d'un type particulier de petites ontologies à partir de collections de documents. Les ontologies construites sont des ontologies de thèmes (« topic ontologies »), où chaque concept de l'ontologie représente le thème d'une classes de documents, et non des concepts du domaine décrit dans les documents comme pour les autres approches. Les documents qui composent le corpus sont les instances des concepts de l'ontologie construite.

OntoGen est une approche interactive. Dans un premier temps l'outil analyse les documents, identifie les mots clés de chaque document et regroupe les documents à l'aide d'algorithmes de classification. Ensuite, l'utilisateur est guidé itérativement dans une démarche de conceptualisation descendante : les classes de documents sont proposées à l'utilisateur, qui choisit ceux qu'il veut avoir comme concepts de l'ontologie ; pour chaque classe choisie l'outil réalise une classification des documents associés et, à l'étape suivante, les nouvelles classes sont présentées à l'utilisateur en complément des classes initiales. L'outil permet de choisir la mesure utilisée pour l'identification des mots clés des documents (TF-IDF par défaut) et l'algorithme de classification (k-means, SVM, etc.).

3.3.3.7 OntoLT

OntoLT [21] se présente comme une extension pour l'éditeur d'ontologies Protégé. Il permet la définition manuelle de patrons reliant des éléments du texte (annotés linguistiquement) à des éléments de l'ontologie. OntoLT n'est pas une approche de construction automatique d'ontologies *stricto sensu*, mais plutôt un environnement qui permet à l'utilisateur de définir des règles pour extraire des éléments d'ontologie à partir d'un texte qui a été prétraité et annoté. Mais, comme OntoLT est souvent cité dans la littérature comme un outil pour l'apprentissage des ontologies, nous l'avons signalé dans cette section.

3.4 Conclusion

Après avoir rappelé les invariants des méthodologies proposées pour la construction des ontologies, nous avons présenté plus en détails le processus de conceptualisation de Methontology. Cette analyse nous a permis de montrer que, bien que chacune des méthodologies plus récentes ait apporté des éléments novateurs, les tâches définies dans Methontology semblent bien rester les tâches fondamentales pour la construction d'une ontologie.

Nous nous sommes ensuite centrés sur la construction automatique d'ontologies à partir de textes en discutant à la fois de l'apport mais aussi des limites actuelles de cette automatisation. Nous avons complété la discussion par une présentation de plusieurs approches opérationnelles de construction automatique.

L'analyse est poursuivie dans le chapitre suivant où nous proposons une comparaison expérimentale basée sur différentes dimensions complémentaires.

4

Comparaison d'outils logiciels de construction automatique

Sommaire

4.1 Introduction

Le chapitre précédent a mis en évidence la complexité du processus de construction d'une ontologie et a introduit l'intérêt de l'intégration de différents mécanismes d'apprentissage pour tenter d'automatiser différentes tâches.

Si, en principe, ces mécanismes implémentés dans les différents outils que nous avons présentés visent à simplifier et accélérer le processus de construction des ontologies, en pratique, les concepteurs d'une ontologie sont confrontés à trois problèmes majeurs.

Le premier problème concerne le choix de l'outil le mieux adapté. D'une part, ce choix doit prendre en compte les caractéristiques de l'ontologie qui doit être construite : ontologie légère ou lourde, peuplement de l'ontologie, etc. D'autre part, le choix doit tenir compte des caractéristiques des outils. Ce choix est rendu particulièrement difficile par la multiplication des approches et des outils, chacun avec ses points forts et ses faiblesses. Et, les performances des outils dépendent non seulement du type de techniques qu'ils implémentent mais aussi des caractéristiques des textes qui composent le corpus. Une analyse complète de ces aspects pourrait conduire à la création d'un système d'aide à la décision ou de recommandation qui pourrait assister les concepteurs d'ontologie dans le choix des outils. A notre connaissance, aucun système de ce type n'a été encore proposé.

Le deuxième problème concerne la complexité du processus d'évaluation permettant de vérifier que l'ontologie construite automatiquement respecte les critères de qualité requis (absence d'erreurs, conformité par rapport aux spécifications, etc.) et, le cas échéant, l'effort nécessaire pour l'améliorer.

Le troisième problème est lié au fait que, dans la plupart des approches, certaines tâches du processus de construction doivent être réalisées manuellement. D'après Blomqvist, 2009 [13], une question encore ouverte dans le domaine est celle de la combinaison des résultats obtenus automatiquement avec les tâches manuelles lors de la conception. Il est possible que, dans certaines situations, cette combinaison soit aussi difficile à réaliser qu'une construction complètement manuelle. Pour décider de la pertinence d'utilisation de la construction automatique dans une situation donnée, il faudrait quantifier l'effort nécessaire pour construire manuellement l'ontologie et le comparer avec l'effort nécessaire pour comprendre, corriger et intégrer des résultats intermédiaires obtenus automatiquement. A notre connaissance, aujourd'hui encore cette question reste aussi largement ouverte.

Dans ce chapitre, nous nous focalisons sur le premier problème. Le deuxième est abordé au chapitre 6. Et, le troisième n'est pas traité dans cette thèse ; nous y revenons dans la discussion finale.

Ce chapitre est structuré comme suit. Dans une première partie, nous présentons un ensemble de critères qui permettent d'analyser et de comprendre les limitations et les points forts des outils pour la construction automatique d'ontologies à partir de textes. Cette présentation inclut aussi une analyse comparative avec d'autres travaux antérieurs. Dans une deuxième partie, nous utilisons cet ensemble de critères pour analyser et comparer quatre approches : Text2Onto, OntoLearn, Asium et Sprat. Cette analyse se décline en trois phases : (1) association entre les tâches automatisées et les outils implémentés ; (2) analyse technique des outils ; (3) comparaison expérimentale des performances de chacun sur des corpus tests.

4.2 Critères pour l'analyse et la comparaison des approches et des outils pour la construction automatique d'ontologies

4.2.1 Etat de l'art

A notre connaissance, les travaux de Park et al., 2011 [102] sont parmi les rares travaux de la littérature à proposer un ensemble structuré (« framework ») de critères pour l'analyse comparative des outils pour la construction automatique des ontologies à partir de textes. Ils ont proposé trois types de critères : (1) critères concernant des aspects génériques, (2) critères concernant l'extraction et (3) critères concernant la qualité de l'ontologie construite.

1. Trois critères concernent des aspects génériques :
   - l'interface (langages supportés, possibilité de visualiser l'ontologie, facilité d'utilisation) ;
   - la disponibilité des outils (« availability ») ;
   - le temps nécessaire pour la prise en main de l'outil.
2. Sept critères concernent l'extraction :
   - le niveau d'extraction – l'outil permet seulement l'extraction de concepts ou il permet aussi l'extraction de relations ;
   - le degré d'automatisation – l'extraction est complètement automatique ou pas ;
   - la possibilité de sélectionner parmi plusieurs algorithmes ;
   - la possibilité de réutilisation des concepts définis dans d'autres ontologies (« referencability of other ontologies ») ;
   - la nécessité de prétraiter les textes – par exemple, l'application des annotations linguistiques ;
   - l'efficacité – le temps de calcul nécessaire pour l'extraction des concepts et des relations ;
   - la fiabilité – les résultats obtenus par les outils restent stables quand ils sont testés à plusieurs reprises dans des conditions identiques.
3. Huit critères, divisés en trois groupes, concernent la qualité de l'ontologie construite :
   - critères concernant la qualité syntactique
     - la validité syntactique (« lawfulness ») ;
     - la richesse syntactique – l'utilisation de l'expressivité du langage (l'utilisation de tous les operateurs du langage).
   - critères concernant la qualité sémantique
     - l'interprétabilité (« interpretability ») – utilisation des termes avec un sens correct dans l'ontologie ;
     - la consistance – unicité du sens des termes dans l'ontologie ;
     - la clarté – absence de termes polysémiques dans l'ontologie.
   - critères concernant la qualité pragmatique
     - la précision (« accuracy ») – validité du contenu de l'ontologie ;
     - la complétude (« comprehensiveness ») – l'ontologie contient un nombre réduit ou un nombre important de concepts ;
     - la pertinence – l'ontologie correspond aux spécifications.

Park et al., 2011 [102] ont utilisé ces critères pour comparer quatre outils : OntoLT, Terminae, Text2Onto et OntoBuilder [1].

Pour limiter la subjectivité dans l'évaluation de la qualité des ontologies construites par les différents outils, Park et al. ont demandé à quatre experts indépendants d'évaluer chaque ontologie.

4.2.2 Limitations des travaux existants

Si les travaux de Park et al. dressent les étapes principales d'une analyse comparative, on peut néanmoins identifier trois limitations.

La première limitation est dûe à leur hypothèse de travail : ils ont considéré le cas où la construction automatique d'une ontologie est réalisée à l'aide d'un seul outil, alors que dans la littérature plusieurs approches ont proposé l'utilisation successive de plusieurs outils, correspondant à des tâches différentes, combinée parfois avec des tâches manuelles (par exemple OntoLearn et Asium). L'analyse et la comparaison des approches et des outils devraient permettre de différencier les tâches qui sont automatisées de celles qui doivent être réalisées manuellement ou de celles qui sont ignorées.

La deuxième limitation concerne les critères liés aux aspects génériques et d'extraction des outils. Certaines définitions devraient être élargies pour prendre en compte des caractéristiques supplémentaires des outils (par exemple, le critère « degré d'extraction » doit concerner non seulement les concepts et les relations, mais aussi les instances ; en plus, il est préférable de distinguer les relations taxonomiques des autres relations sémantiques). De nouveaux critères pourraient être également ajoutés ; par exemple, l'appel à d'autres outils ou à des ressources externes durant le processus d'extraction. Des propositions d'extension sont proposées dans la section 4.2.3.2.

La troisième limitation concerne les critères de comparaison basés sur l'analyse de la qualité sémantique et pragmatique des ontologies construites par les outils. Certains critères semblent ambigus, difficilement applicables et leur pertinence peut être discutable. En effet, la définition proposée pour la consistance est ambigüe puisque usuellement la consistance d'une ontologie fait référence à la consistance logique. La définition du critère de « clarté » ne prend pas en compte le domaine de l'ontologie ; dans la plupart des cas un seul sens des termes polysémiques concerne le domaine de l'ontologie. Et, la définition de l'interprétabilité est également ambigüe et basée uniquement sur le jugement humain.

De plus, si les critères de précision, complétude et pertinence sont adaptés à l'évaluation de la qualité pragmatique, la méthode proposée pour leur application – basée uniquement sur le jugement humain – est discutable, en particulier pour les ontologies construites automatiquement qui peuvent être très grandes, qui ne contiennent pas de commentaires et qui peuvent contenir parfois des relations complexes.

4.2.3 Notre proposition

Pour pallier ces limitations, nous proposons une comparaison en trois étapes entre les différentes approches et leurs outils.

1. OntoBuilder permet la construction automatique d'ontologies à partir de formulaires HTML, alors que les trois autres outils prennent en entrée du texte non structuré.

4.2.3.1 Comparaison basée sur le degré de complétude et d'automatisation

La première étape porte sur le degré de complétude et d'automatisation des approches : identification des tâches ignorées par les approches et des tâches traitées ; mise en évidence de ce qui est automatisé et de ce qui ne l'est pas ; identification des outils et des techniques utilisées pour l'automatisation. Pour avoir une base commune pour cette comparaison, nous recourons aux tâches qui composent l'activité de conceptualisation de Methontology. Comme la plupart des approches se limitent à l'extraction des éléments centraux de l'ontologie (concepts, instances et relations), en ignorant les attributs, les axiomes et les règles, nous utilisons pour cette comparaison seulement les quatre premières tâches de l'activité de conceptualisation.

4.2.3.2 Comparaison basée sur les caractéristiques des outils

La deuxième étape concerne « l'analyse technique » des outils. Cette analyse couvre la plupart des critères génériques et d'extraction proposés par Park et al. Les critères que nous utilisons lors de cette analyse technique sont :

1. **disponibilité** (proposée également par Park et al. [102]) : nous distinguons trois cas – les outils indisponibles ; les outils disponibles avec possibilité d'installation sur une machine locale ; les outils disponibles qui peuvent être testés seulement en ligne, sans possibilité d'installation en local.
2. **entrées, sorties, prétraitement** (extension du critère « prétraitement » de Park et al. [102]) : le type de ressources que les outils prennent en entrée, la forme des résultats obtenus (par exemple, ontologie OWL, liste de termes ou de relations, etc.) et le travail nécessaire pour préparer les ressources pour que les outils puissent les utiliser.
3. **réutilisation et amélioration** (critère différent de celui de Park et al. [102] concernant la réutilisation des concepts définis dans d'autres ontologies) : on vérifie si l'outil peut réutiliser ses résultats. Par exemple, pour un outil qui produit en sortie une ontologie OWL, on vérifie s'il est capable de prendre en entrée une autre ontologie (construite antérieurement ou provenant d'une autre source) et de l'enrichir. Pour un outil qui extrait des termes, on vérifie s'il est capable de prendre en entrée une liste de termes et de l'enrichir.
4. **degré d'extraction** (extension du critère proposé par Park et al. [102]) : les degrés d'extraction utilisés sont les concepts, les instances, les relations taxonomiques et les autres relations sémantiques (et pas uniquement les concepts et les relations comme dans Park et al. [102]).
5. **degré d'automatisation** (même critère que celui proposé par Park et al. [102]) : on distingue les outils complètement automatiques des outils semi-automatiques qui nécessitent l'intervention manuelle des experts.
6. **paramétrage** (extension du critère de Park et al. [102] concernant la possibilité de sélectionner les algorithmes qui sont utilisés) : ce critère prend en compte les différentes possibilités de paramétrage des outils (la sélection des algorithmes est considérée comme un cas particulier de paramétrage).

7. **efficacité** (même critère que celui proposé par Park et al. [102]) : le temps nécessaire aux outils pour extraire les différents éléments.
8. **fiabilité** (même critère que celui proposé par Park et al. [102]) : on verifie si les résultats obtenus sont les mêmes lors d'exécutions répétées dans des conditions identiques.
9. **outils et ressources externes** (critère sans correspondance avec ceux de Park et al. [102]) : on verifie si les outils, en plus de corpus qu'ils prennent en entrée, utilisent durant le processus d'extraction d'autres ressources (comme, par exemple, WordNet ou des recherches sur Internet) ou des outils externes (par exemple, des outils de traitement de la langue).

4.2.3.3 Comparaison basée sur la qualité des résultats

La troisième étape concerne la comparaison des outils sur la base de la qualité des résultats obtenus dans le cadre de tests expérimentaux. Nous considérons qu'une comparaison plus précise que celle proposée par Park et al. est possible. Dans un premier temps, les résultats de chaque outil sont comparés à une ontologie construite manuellement. Ces comparaisons permettent une première évaluation quantitative par deux mesures classiques (la précision et le rappel) pour chaque outil. Nous complétons par deux mesures : la *spécificité des termes* extraits et la *richesse sémantique* des relations taxonomiques.

La *spécificité des termes* extraits est basée sur la considération suivante : il est plus probable que les termes complexes, composés de deux ou plusieurs mots, correspondent à des concepts plus spécifiques d'un domaine que les termes simples, composés d'un seul mot – par exemple, le terme *algorithme de désambiguïsation* correspond à un concept plus spécfique que le terme *algorithme*. Nous considérons qu'il est intéressant de considérer trois valeurs pour cette mesure : termes simples, termes composés de deux mots, et termes composés de trois mots ou plus. La *richesse sémantique* des relations taxonomiques est basée sur l'observation qu'une relation taxonomique peut relier deux concepts dont les « termes étiquettes » ont la particularité de pouvoir s'inclure lexicalement (le terme plus court fait partie lexicalement du terme plus long – par exemple, le terme *arbre* fait partie lexicalement du terme *arbre de concepts*). Intuitivement, une telle relation apporte moins d'informations qu'une relation taxonomique entre deux concepts dont les étiquettes n'ont pas cette propriété.

Pour pallier les limitations de la construction manuelle et prendre en compte le fait que les outils peuvent découvrir des concepts pertinents ne faisant pas partie de l'ontologie construite manuellement, nous avons introduit un processus d'ajustement. Les éléments extraits par les outils sont évalués directement par l'expert en suivant une adaptation de la méthode utilisée pour la validation de Text2Onto [27]. A chaque composant extrait par les outils, l'expert assigne une note entre 1 (composant invalide) et 4 (composant pleinement valide) en suivant l'échelle suivante :

– Pour les concepts et les instances :
    1. des termes qui ne sont pas l'étiquette ni d'un concept ni d'une instance
    2. des termes qui sont identifiés en tant qu'étiquette d'un concept alors qu'ils sont plutôt l'étiquette d'une instance, et le cas inverse
    3. des termes qui sont l'étiquette d'un concept ou d'une instance qui ne sont pas pertinents pour le domaine

4. des termes qui sont l'étiquette d'un concept ou d'une instance qui sont pertinents pour le domaine

– Pour les relations taxonomiques :
1. une relation incorrecte ou dont au moins un des termes qu'elle relie n'est pas l'étiquette d'un concept
2. une relation non taxonomique entre deux concepts
3. une relation taxonomique correcte mais pour laquelle au moins un des concepts impliqués est trop général ou trop spécifique par rapport au domaine de l'ontologie
4. une relation taxonomique correcte qui relie deux concepts pertinents pour le domaine

Les composants extraits par les outils, qui ont été évalués comme pleinement valides et qui ne font pas partie de l'ontologie construite manuellement sont ajoutés [2] aux composants de l'ontologie manuelle, obtenant ainsi une « ontologie d'union » (ou « ontologie étendue »).

Les résultats des outils sont comparés non seulement à l'ontologie manuelle, mais aussi à cette ontologie d'union, ce qui permet une meilleure compréhension des performances de chaque outil. Par conséquent, pour chaque outil on calcule deux valeurs pour la précision et le rappel : $Precision_m$ et $Rappel_m$ par rapport à l'ontologie construite manuellement O_m et $Precision_u$ et $Rappel_u$ par rapport à l'ontologie d'union O_u.

$$Precision_m = \frac{|EE \cap O_m|}{|EE|} \qquad Rappel_m = \frac{|EE \cap O_m|}{|O_m|}$$

$$Precision_u = \frac{|EE \cap O_u|}{|EE|} \qquad Rappel_u = \frac{|EE \cap O_u|}{|O_u|}$$

où EE est l'ensemble des composants extraits

4.2.3.4 Schéma d'analyse pour la comparaison des approches

La figure 4.1 présente une vue globale des relations entre les éléments utilisés dans la comparaison des approches et des outils. Les éléments ont été regroupés en trois groupes :

1. *Approche* : éléments permettant la caractérisation des approches (les tâches, les techniques utilisées pour l'automatisation, les outils et leur configuration) ;
2. *Sorties et statistiques* : éléments permettant la caractérisation des résultats obtenus dans les tests (l'ontologie construite automatiquement, avec ses composants, et les mesures quantifiant les performances des outils) ;
3. *Ligne de comparaison* : éléments servant de support pour la comparaison des approches (le référentiel de tâches de Methontology, l'ontologie construite manuellement et le résultat de l'analyse technique des outils).

2. Comme il s'agit seulement de concepts, d'instances et de relations taxonomiques, le problème qui peut apparaître lors de cette intégration concerne l'apparition de cycles de relations taxonomiques ; mais, comme les relations taxonomiques qui doivent être intégrées ont été validées en préalable par l'expert, il est peu probable que cette situation arrive.

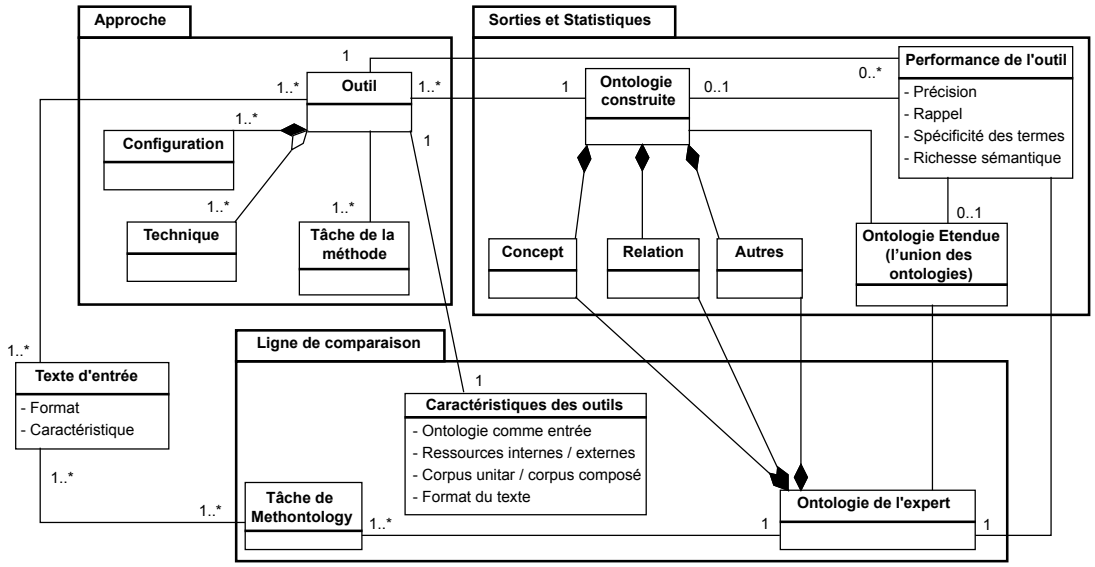

FIGURE 4.1: Diagramme UML résumant les relations entre les éléments utilisés pour notre démarche de comparaison des outils.

4.3 Analyse comparative de quatre approches

Dans cette section, nous présentons une analyse comparative détaillée de quatre approches de construction automatique des ontologies à partir de textes en suivant les principes introduits dans la section précédente.

Le choix de ces approches repose essentiellement sur quatre raisons :

1. elles automatisent au moins l'identification des concepts et des relations taxonomiques ; c'est pour cette raison que nous n'avons pas retenu Terminae où l'identification des concepts est semi-automatique et celle des relations taxonomiques est manuelle.
2. elles construisent des ontologies du domaine concerné par le corpus ; c'est pour cette raison que nous n'avons pas retenu OntoGen, qui construit une ontologie de thèmes.
3. elles sont génériques et peuvent être potentiellement appliquées à tous les domaines.
4. elles sont accompagnées d'outils logiciels qui les rendent opérationnelles.

Toutes les approches introduites dans la Section 3.3.3 coïncident avec les deux derniers critères, et seulement quatre satisfont aussi aux deux premiers : OntoLearn, Asium, Text2Onto et Sprat.

Dans la suite, nous analysons ces quatre approches selon les trois axes identifiés précédemment : le référentiel de tâches de Methontology, les aspects techniques et la qualité des résultats.

4.3.1 Comparaison par rapport au référentiel de tâches

Table 4.1: Correspondances et différences entre les tâches des quatre approches et le référentiel de tâches déduit de Methontology.

Tâches de Methontology	Tâches de OntoLearn		Tâches de Asium	Tâches de Text2Onto	Tâches de Sprat
Construction du glossaire de termes *Identification et définition des termes correspondant aux différents éléments de l'ontologie : concepts, attributs, instances et relations*	Extraction des termes – *L & S* Filtrage des termes – *à l'aide de filtres L & S* Validation des termes – *Man* Identification de définitions pour les termes *(constituant ainsi un glossaire)* – *à l'aide de recherches sur Internet – ES* *Termes = { concepts }*		Extraction des termes – *L* Validation des termes – *Man* *Termes = { concepts, instances }*	Extraction des termes – *L & S* *Termes = { concepts, instances }*	Extraction des termes – *L* *Termes = { concepts, instances }* **Exécution itérative** Choix d'un nouveau terme impliqué dans un patron *'is-a'* le reliant à un concepts déjà présent dans l'ontologie – *Pa*
Construction des taxonomies de concepts *Regroupement des termes qui correspondent au même concept* *Organisasion des concepts dans une ou plusieurs taxonomies*	Désambiguïsation sémantique de chaque terme complexe par l'intersection des réseaux sémantiques associées à chaque mot composant le terme : *algo. SSI (spécifique à OntoLearn) et WordNet – ES & St* Identification des relations taxonomiques entre les synsets associés aux mots composant les termes complexes	Extraction d'hyperonymes Identification des star patterns et classification des propositions – *Man* Construction de la Word-Class Lattice (WCL) – *PL* Alignement des propositions Identification des correspondances entre les propositions et WCL – *Pa*	Construction d'une taxonomie – *Identification du contexte de chaque terme et construction de la taxonomie par classification non supervisée (basée sur le contexte) des termes – Di*	Identification des relations *'is-a'* entre concepts *(à l'aide de WordNet, des patrons et des heuristiques (basées sur la structure des termes composés) – St & Pa & ES*	*(si le terme choisi -> concept)* Le concept est ajouté à l'ontologie – *à l'aide des règles d'intégration et de résolution de conflits (règles spécifiques à Sprat) – Pa*
Construction des diagrammes des relations binaires ad hoc *Regroupement des termes qui correspondent à la même relation* *Définition des relations en identifiant, pour chaque relation, les concepts concernés – par l'analyse des termes associés à chaque verbe (le sujet et le complément d'objet)*	Fournir des exemples de relations sémantiques du domaine – *Man* Apprentissage de règles pour l'identification des relations – *PL* Utilisation des règles sur les termes complexes pour identifier des relations entre ses composants – *Pa*		Identification des dépendances syntactiques indiquant une relation *– à l'aide de ASA, un formalisme spécifique à Asium* Choix de quelques exemples – *Man* Apprentissage de règles pour l'identification des relations – *à l'aide de l'apprentissage inductif basé sur le formalisme ASA – PL* Identification des concepts connectés par les relations apprises à partir de exemples – *automatiquement, à l'aide des règles – Pa*	Identification des relations *'subtopic-of'* entre les concepts – *(à l'aide de l'analyse statistique de la cooccurrence des concepts) – Di* Identification des relations génériques (définies par un verbe) – *(à l'aide de shallow parsing strategy et des informations sur la fréquence des termes) – Di*	Identification et intégration dans l'ontologie de relations entre le nouveau concept et des concepts déjà présents dans l'ontologie *(à l'aide de patrons prédéfinis) – Pa*
Construction du dictionnaire de concepts *(qui contient, pour chaque concept, ses attributs, ses instances et ses relations)*	—		—	Identification des relations *'instance-of' – Di*	*(si le terme choisi -> instance)* L'instance est ajoutée à l'ontologie – *à l'aide de règles spécifiques à Sprat – Pa*

La comparaison par rapport au référentiel de tâches de Methontology a un triple objectif :
- mettre en évidence les tâches et les sous-tâches de chaque approche qui correspondent aux tâches du référentiel. Cette analyse permet l'identification des étapes ignorées par certaines approches ;
- distinguer ce qui est automatisé de ce qui ne l'est pas ;
- mettre en évidence les outils et les techniques utilisés pour l'automatisation.

Cette comparaison est présentée à l'aide des tableaux 4.1 et 4.2.

Le tableau 4.1 montre tout d'abord que les quatre approches ignorent l'identification d'attributs pour les concepts. Et, il souligne le fait que OntoLearn et Asium ne permettent pas le peuplement de l'ontologie (l'identification des instances). Il est intéressant de noter que, dans le cas de Sprat, l'extraction de concepts, de relations taxonomiques, de relations ad-hoc et des instances ne constituent pas des étapes séparées et indépendantes, comme pour Methontology et les trois autres approches analysées. Sprat est basé sur un cycle itératif et, à chaque itération l'ontologie est enrichie avec un nouveau concept (avec sa relation taxonomique), une nouvelle relation *ad-hoc* ou une nouvelle instance.

Le tableau 4.2 montre que seuls Text2Onto et Sprat proposent des outils intégrés qui couvrent l'ensemble du processus de construction d'une ontologie. De plus, pour l'extraction des relations taxonomiques OntoLearn propose deux outils indépendants, SSI et WCL System qui implémentent chacun des techniques différentes.

Certains outils ont été développés spécifiquement pour la construction des ontologies : TermExtractor, GlossExtractor, SSI, WCL System, Asium, Text2Onto, NEBOnE et Sprat. Les autres outils mentionnés dans le tableau 4.2 ont été développés pour un autre objectif et ont trouvé une utilisation particulière dans le processus de construction de l'ontologie ; C4.5 et Propal, permettent l'apprentissage inductif de règles ; LinkParser et BioLG, permettent l'analyse de la syntaxe du texte ; JAPE permet l'analyse du texte à l'aide des expressions régulières (patrons) ; YATEA permet l'extraction de termes.

Comme Sprat, dans le cadre de l'approche avec le même nom, couvre l'ensemble des tâches du référentiel de tâches et que les résultats de TermRaider, JAPE et NEBOnE ne peuvent pas être intégrés manuellement avec ceux de Sprat, la suite de notre analyse se restreint aux trois outils TermRaider, JAPE et NEBOnE.

Les types de techniques (introduites dans la Section 3.3.2) utilisés par les quatre approches pour l'automatisation des différentes sous-tâches sont indiqués dans le tableau 4.1 à l'aide des acronymes suivants : S (approches statistiques), L (approches linguistiques), St (techniques structurelles), Di (techniques distributionnelles), MP (techniques à base de patrons où les patrons doivent être précisés manuellement), PL (techniques à base de patrons où les patrons sont appris pendant l'extraction), PA (techniques à base de patrons où les patrons sont connus en avance), ES (utilisation de ressources externes), Man (tâche qui doit être effectué manuellement).

Table 4.2: Correspondances entre outils et tâches pour chaque approche. Les outils testés sont en caractères gras.

<table>
<tr><th rowspan="2">Tâches de Methontology</th><th colspan="3">Tâches de OntoLearn</th><th colspan="2">Tâches de Asium</th><th colspan="2">Tâches de Text2Onto</th><th colspan="3">Tâches de Sprat</th></tr>
<tr><th colspan="2">Sous-tâche</th><th colspan="2">Outil</th><th>Sous-tâche</th><th>Outil</th><th>Sous-tâche</th><th>Outil</th><th>Sous-tâche</th><th colspan="2">Outil</th></tr>
<tr><td rowspan="3">Construction du glossaire de termes</td><td colspan="2">Extraction des termes
Filtrage des termes</td><td colspan="2">TermExtractor – (L & S)</td><td>Extraction des termes</td><td>YATEA – (L)</td><td>Extraction des concepts</td><td>Text2Onto module Concept – (L & S)</td><td>Extraction des termes</td><td rowspan="5">JAPE, NEBOnE – (L)</td><td>Term Raider – (L & S)</td><td rowspan="5">Sprat – (Pa)</td></tr>
<tr><td colspan="2">Validation des termes</td><td colspan="2">—</td><td rowspan="2">Validation des termes</td><td rowspan="2">—</td><td rowspan="2">Extraction des instances</td><td rowspan="2">Text2Onto module Instance – (L & S)</td><td rowspan="2">Choisir un terme</td><td rowspan="2">—</td></tr>
<tr><td colspan="2">Identification de définitions pour les termes</td><td colspan="2">GlossExtractor – (ES)</td></tr>
<tr><td rowspan="2">Construction des taxonomies de concepts</td><td>Désambiguïsation sémantique des termes complexes</td><td rowspan="2">Extraction d'hyper-onymes</td><td>SSI – (ES & St)</td><td rowspan="2">WCL System– (PL, Pa)</td><td rowspan="2">Construction d'une taxonomie</td><td rowspan="2">BioLG, ASIUM adapté pour Asium – (Di & Cl)</td><td rowspan="2">Identification des relations 'is-a'</td><td rowspan="2">Text2Onto module SubclassOf – (St & Pa & ES)</td><td rowspan="2">Ajouter le concept à l'ontologie</td><td rowspan="2">—</td></tr>
<tr><td>Identification de relations taxonomiques</td><td>—</td></tr>
<tr><td rowspan="3">Construction des diagrammes des relations binaires ad hoc</td><td colspan="2">Fournir des exemples de relations sémantiques du domaine</td><td colspan="2">—</td><td>Analyse ASA</td><td>LINK-PARSER</td><td rowspan="2">Identification des relations 'subtopic-of'</td><td rowspan="2">Text2Onto module SubtopicOf – (Di & Cl)</td><td rowspan="3">Identification et intégration des relations dans l'ontologie</td><td rowspan="3">—</td></tr>
<tr><td colspan="2">Apprentissage de règles</td><td colspan="2">C4.5 – (PL)</td><td>Choix de quelques exemples</td><td>—</td></tr>
<tr><td colspan="2">Identification de relations entre les composants des termes complexes</td><td colspan="2">—</td><td>Apprentissage de règles
Identification des relations</td><td>Propal – (PL)
—</td><td>Identification de relations génériques</td><td>Text2Onto module Relation – (Di & Cl)</td></tr>
<tr><td>Construction du dictionnaire de concepts</td><td colspan="2">—</td><td colspan="2">—</td><td>—</td><td>—</td><td>Identification des relations 'instance-of'</td><td>Text2Onto module InstanceOf – (Di & Cl)</td><td>Ajouter l'instance à l'ontologie</td><td>—</td></tr>
</table>

4.3.2 Comparaison technique

1. **disponibilité** :
   - OntoLearn : TermExtractor, GlossExtrator et SSI sont, *a priori*[3], disponibles en ligne sur le serveur de leur auteur ; ils peuvent être utilisés via une page web ; leur disponibilité dépend de celle du serveur où ils sont hébergés ;
   - Asium : l'outil Asium n'est pas disponible pour être testé ;
   - Text2Onto : l'outil Text2Onto est disponible et peut être installé en local ;
   - Sprat : les outils TermRaider et Sprat sont disponibles sur forme de services web ; leur disponibilité dépend de celle du serveur où ils sont hébergés ;
2. **entrées, sorties, prétraitement** :
   - OntoLearn : TermExtractor prend en entrée le corpus et son résultat est une liste de concepts ; GlossExtractor : liste de termes en entrée, liste de définitions en sortie ; WCL System : liste de définitions en entrée, liste de relations taxonomiques en sortie ;
   - Asium : Asium prend en entrée le corpus et offre en sortie une taxonomie de concepts ;
   - Text2Onto : l'outil Text2Onto prend en entrée des textes (documents TXT ou PDF) et produit en sortie une ontologie qui peut être sauvegardée dans plusieurs formats dont OWL ;
   - Sprat : l'outil Sprat prend en entrée le corpus et son résultat est une ontologie en OWL ;
3. **réutilisation et amélioration** :
   - OntoLearn : TermExtractor peut enrichir une liste de termes ;
   - Asium : pas de réutilisation ;
   - Text2Onto : l'outil Text2Onto peut faire évoluer, suite à une modification du corpus, une ontologie déjà construite ;
   - Sprat : l'outil Sprat peut prendre en entrée une ontologie et l'enrichir ;
4. **degré d'extraction** :
   - OntoLearn : extraction complètement automatique de concepts et de relations taxonomiques ; extraction partiellement manuelle de relations ad-hoc ; pas d'extraction d'instances ;
   - Asium : extraction complètement automatique de concepts et de relations taxonomiques ; extraction partiellement manuelle de relations ad-hoc ; pas d'extraction d'instances ;
   - Text2Onto : l'outil Text2Onto permet l'extraction complètement automatique de concepts, d'instances, de relations taxonomiques et de relations ad-hoc ;
   - Sprat : l'outil Sprat permet l'extraction complètement automatique de concepts, d'instances, de relations taxonomiques et de relations ad-hoc ;
5. **degré d'automatisation** :
   - OntoLearn : TermExtractor, SSI, GlossExtractor et WCL System sont automatiques ; l'identification des relations ad-hoc et l'intégration des résultats des différents outils nécessite l'implication de l'expert ;

3. ces outils ont été disponibles durant les deux premières années de ma thèse mais actuellement le serveur où ils sont hébergés est affecté par des problèmes de configuration

- Asium : l'identification de concepts et de la hiérarchie des concepts est automatique ; l'identification des relations ad-hoc nécessite l'implication de l'expert ;
- Text2Onto : outil complètement automatique ; l'utilisateur peut éliminer manuellement des résultats intermédiaires ;
- Sprat : outil complètement automatique, l'utilisateur ne peut pas intervenir durant le processus de construction ;

6. **paramétrage** :
   - OntoLearn : seul TermExtractor est configurable ; il permet de moduler les différents paramètres de ses filtres (par exemple, le nombre maximal de mots qui peuvent composer un terme) mais aussi de l'algorithme d'extraction (prise en charge de la position – dans le titre, par exemple – et du style du terme dans le texte – en gras ou en italique) ;
   - Asium : aucune configuration n'est possible ;
   - Text2Onto : l'outil Text2Onto permet le choix des algorithmes pour l'extraction des différents éléments et le choix de la stratégie utilisée pour combiner les résultats des différents algorithmes ;
   - Sprat : aucune configuration n'est possible ;
7. **efficacité** :
   - OntoLearn : l'exécution ayant lieu en mode différé, il nous est impossible d'estimer l'efficacité ; dans la plupart des cas les tâches sont exécutées en moins d'une heure ;
   - Asium : – impossible de tester – ;
   - Text2Onto : l'outil Text2Onto est efficace (temps d'exécution de moins de trois minutes dans nos tests) ;
   - Sprat : Sprat et TermRaider sont efficaces (temps d'exécution de moins d'une minute dans nos tests) ;
8. **fiabilité** :
   - OntoLearn : SSI, TermExtrator et WCL sont fiables ; les résultats de GlossExtractor dépendent de l'évolution de l'Internet ;
   - Asium : l'outil Asium est fiable ;
   - Text2Onto : l'outil Text2Onto est fiable ;
   - Sprat : Sprat et TermRaider sont fiables ;
9. **outils et ressources externes** :
   - OntoLearn : SSI utilise WordNet en tant que ressource auxiliaire et GlossExtractor utilise des recherches sur Internet ;
   - Asium : pas de ressources ou d'applications externes ;
   - Text2Onto : l'outil Text2Onto a besoin de la plateforme GATE et de l'analyseur syntactique TreeTagger pour l'analyse du texte ; il utilise WordNet et des recherches sur Internet comme des ressources auxiliaires ;
   - Sprat : pour utiliser Sprat et TermRaider il est nécessaire d'installer l'application NeonToolkit et un plugin spécifique ;

Comme Asium, le seul outil spécialisé proposé par l'approche homonyme, n'est pas disponible pour être testé, la dernière partie de l'analyse comparative des outils, présentée dans la section suivante, se restreint aux approches OntoLearn, Text2Onto et Sprat.

4.3.3 Comparaisons expérimentales

Pour comparer les approches en fonction de la qualité des résultats qu'elles permettent d'obtenir nous avons réalisé plusieurs tests et nous avons comparé les résultats obtenus en suivant la procédure présentée dans la première partie de ce chapitre (section 4.2.3.3).

4.3.3.1 La configuration utilisée

Pour tester les outils nous avons choisi un corpus de taille moyenne (4000 mots) dont le domaine – la construction des ontologies à partir de textes – nous est familier, ce qui nous a permis d'être nous mêmes les experts pour la validation manuelle des résultats des outils et pour la construction de l'ontologie manuelle. Le corpus est composé d'un seul document [4] – un article scientifique, écrit par Navigli et Velardi et intitulé « *Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites* » [91], dont on a éliminé toutes les parties qui n'étaient pas du texte brut (les images, les tableaux, les formules) ou qui n'étaient pas liées au domaine (les exemples et les références bibliographiques). Le texte ainsi obtenu est dense ; il contient un très grand nombre de termes faisant référence à des concepts pertinents pour le domaine par rapport au nombre de termes du texte.

En partant des connaissances exprimées dans ce corpus nous avons construit manuellement, en tant qu'experts, une ontologie légère, contenant seulement des concepts mentionnés dans le corpus, les relations taxonomiques entre ces concepts et les instances et les relations *ad-hoc* mentionnées dans le corpus (cette ontologie contient 417 concepts, 38 instances, 407 relations taxonomiques et sa taxonomie a une profondeur maximale de 5 ; l'ontologie est disponible dans l'Annexe B). La figure 4.2 présente les concepts centraux de cette ontologie.

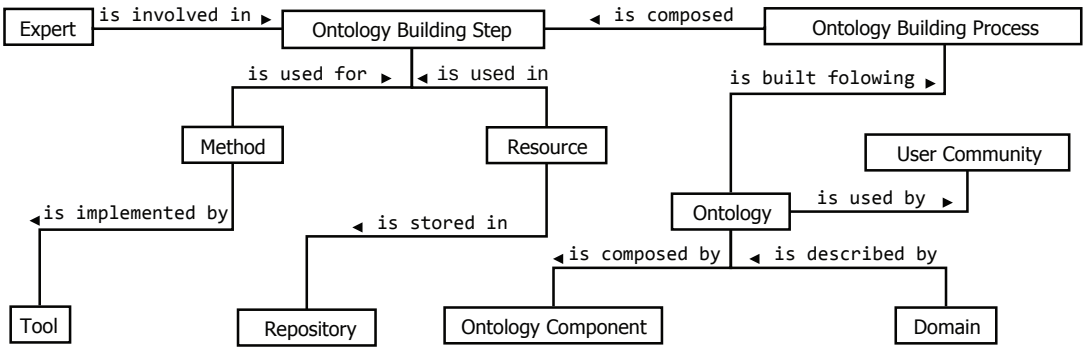

FIGURE 4.2: Les concepts centraux de l'ontologie construite manuellement.

4. Dans le cadre d'une analyse préalable [52] nous avons souligné que les résultats obtenus avec TermExtractor, un des outils associés à l'approche OntoLearn, sont différents suivant que le corpus traité contient plusieurs documents ou un seul document composé de l'union de ces documents. Ce comportement est du au filtrage des termes qui sont extraits par TermExtractor à l'aide d'un filtre nommé « *Domain Consensus* ». Si le seuil de ce filtre est fixé à zéro, alors les résultats obtenus avec TermExtractor sont indépendants du nombre de documents qui composent les corpus. Dans cette thèse, pour faciliter la construction de l'ontologie manuelle, nous avons limité notre analyse à un corpus contenant un seul document.

Nous avons testé les outils qui permettent l'extraction complètement automatisée de composants d'ontologie : TermExtractor, WCL System, Text2Onto et Sprat. Notons que pour obtenir des composants d'ontologie à partir des résultats de SSI il faut qu'un expert précise les règles qui doivent être utilisées pour la combinaison de réseaux sémantiques.

4.3.3.2 Analyse des resultats – concepts et instances

Trois outils permettent l'extraction automatisée de termes identifiant des concepts ou des instances : TermExtractor, Text2Onto et Sprat. Les tableaux 4.3, 4.4, 4.5 et 4.6 présentent les résultats de la validation de leurs résultats et de la comparaison avec l'ontologie construite manuellement (O_{mb}) et avec l'ontologie d'union (O_u).

Table 4.3: Concepts extraits automatiquement : Résultats de la validation de l'expert.

Evaluation[1]	*TermExtractor*[2]	*Text2Onto*	*Sprat*
Concepts extraits	253	444	9
Concepts évalués 1	22	30	3
Concepts évalués 2	4	2	3
Concepts évalués 3	30	138	0
Concepts évalués 4	197 (77%)	274 (61%)	3 (33%)

[1] voir section 4.2.3.3
[2] *TermExtractor* extrait seulement des termes faisant référence à des concepts

Table 4.4: Instances extraites automatiquement : Résultats de la validation de l'expert.

Evaluation	*Text2Onto*	*Sprat*
Instances extraites	94	0
Instances évaluées 1	13	0
Instances évaluées 2	10	0
Instances évaluées 3	51	0
Instances évaluées 4	20 (21%)	0 (0%)

On remarque que TermExtractor a la meilleure précision pour l'extraction de concepts, alors que Text2Onto a un meilleur rappel. Sprat a un très faible rappel. On remarque aussi que Text2Onto extrait plus de termes simples que TermExtractor.

4.3.3.3 Analyse des resultats – relations taxonomiques

Trois outils permettent l'extraction complètement automatique de relations taxonomiques : WCL System, Text2Onto et Sprat. Comme pour l'extraction des concepts, les résultats de ce dernier sont très limitées (tableau 4.7). De ce fait, la comparaison des performances des outils concerne ici seulement WCL System et Text2Onto.

TABLE 4.5: Concepts et instances extraits automatiquement : comparaison avec O_{mb} basée sur la *spécificité des termes*.

Nombre de mots	O_{mb}	*TermExtractor*	*Text2Onto*	*Sprat*
Tous	417	253	444	9
Un mot	81 (19%)	41 (17%)	311 (70%)	3 (33%)
Deux mots	220 (53%)	170 (66%)	116 (26%)	3 (33%)
Trois mots ou plus	116 (28%)	42 (17%)	17 (4%)	3 (33%)

TABLE 4.6: Concepts et instances extraits automatiquement : comparaison avec O_{mb} et O_u.

	O_{mb}	*TermExtractor*	*Text2Onto*	*Sprat*	O_u
Concepts extraits	–	253	444	9	–
Concepts valides	417	197	274	3	581
$\cap\ O_{mb}$	–	149	146	3	417
$Precision_m$	–	59%	33%	30%	–
$Precision_u$	–	78%	62%	30%	–
$Rappel_m$	–	36%	35%	0.7%	–
$Rappel_u$	69%	34%	47%	0.5%	–
Instances extraites	–	–	94	0	–
Instances valides	38	–	20	0	40
$\cap\ O_{mb}$	–	–	18	0	38
$Precision_m$	–	–	19%	0	–
$Precision_u$	–	–	21%	0	–
$Rappel_m$	–	–	47%	0	–
$Rappel_u$	95%	–	50%	0	–

Comme nous l'avons indiqué dans la section 4.3.2, WCL System ne peut être testé qu'en demandant à ses auteurs de l'exécuter sur notre jeu de données. Nous n'avons donc testé WCL System que sur un sous-ensemble représentatif des concepts de l'ontologie. De plus WCL System identifie des relations taxonomiques à partir de définitions de concepts alors que Text2Onto utilise des patrons, des heuristiques et des relations lexicales de WordNet.

Par conséquent, WCL System et Text2Onto ne peuvent pas être testés dans des conditions identiques (comme les extracteurs de termes qui prennent tous en entrée le même texte). La comparaison de leurs résultats doit donc tenir compte de ce problème.

Le tableau 4.8 présente une double comparaison, basée sur la précision et sur le rappel, entre les résultats de Text2Onto et les ontologies O_{mb} et O_u.

Nous avons demandé aux auteurs d'OntoLearn de tester WCL System sur un sous-ensemble de 36 termes sélectionnés parmi ceux extraits par TermExtractor et validés par

Table 4.7: Relations taxonomiques extraites automatiquement : Résultats de la validation de l'expert.

Evaluation	*Text2Onto*	*Sprat*
Relation extraites	362	5
Relation évaluées 1	78	2
Relation évaluées 2	69	3
Relation évaluées 3	104	0
Relation évaluées 4	111 (30%)	0 (0%)

Table 4.8: Relations taxonomiques extraites automatiquement : comparaison avec O_{mb} and O_u.

	O_{mb}	*Text2Onto*	O_u
Relations extraites	–	362	–
Relations valides	407	111	473
$\cap\ O_{mb}$	–	45	407
$Precision_m$	–	41%	–
$Precision_u$	–	31%	–
$Rappel_m$	–	10%	–
$Rappel_u$	86%	23%	–

l'expert [5]. Les auteurs d'OntoLearn ont utilisé comme entrée pour WCL System toutes les définitions que GlossExtractor a identifiées pour les 36 termes.

WCL System a identifié 278 relations taxonomiques impliquant, en plus des 36 termes donnés en entrée, 246 nouveaux termes présents dans les définitions extraites par GlossExtractor : 67 de ces 246 termes sont des étiquettes des concepts de O_u. Seules 25 des 278 relations taxonomiques ont été validées (notées 4) par l'expert. 38 concepts de O_u sont impliqués dans ces 25 relations.

Pour comparer les résultats de WCL System avec ceux de Text2Onto, nous nous limitons aux relations taxonomiques de WCL System, Text2Onto, O_{mb} et O_u qui relient uniquement des concepts qui sont communs aux quatre ensembles de concepts impliqués dans des relations taxonomiques validées par l'expert : les 38 concepts impliqués dans les 25 relations taxonomiques validées extraites par WCL System, les 150 concepts impliqués dans les 111 relations taxonomiques validées extraites par Text2Onto, les 417 concepts impliqués dans les 407 relations taxonomiques de O_{mb} et les 462 concepts impliqués dans les 473 relations taxonomiques de O_u. Ce sous-ensemble de concepts communs, CC, contient 21 concepts. La comparaison est résumée dans le tableau 4.9.

5. Dans le choix des 36 termes composant l'ensemble utilisé pour le test nous avons essayé de garder la même distribution de la *spécificité des termes* que celle de l'ensemble de 197 termes extraits par TermExtractor et validés par l'expert (voir les tableaux 4.3 et 4.5).

Nous avons adapté les mesures $Precision_m$, $Precision_u$, $Rappel_m$ et $Rappel_u$ définies dans la section 4.2.3.3 pour qu'elles prennent en compte uniquemment les relations reliant des concepts de CC :

$$Precision^*_m = \frac{|Select(EE : CC) \cap O_{mb}|}{|Select(EE : CC)|} \qquad Rappel^*_m = \frac{|Select(EE : CC) \cap O_{mb}|}{|Select(O_{mb} : CC)|}$$

$$Precision^*_u = \frac{|Select(EE : CC) \cap O_u|}{|Select(EE : CC)|} \qquad Rappel_u{}^* = \frac{|Select(EE : CC) \cap O_u|}{|Select(O_u : CC)|}$$

où EE = *l'ensemble des relations taxonomiques extraites par les outils,*
CC = *l'ensemble de 21 concepts communs, et*
$Select(EE : E_c)$ = toutes les relations $r_i \in EE$ qui relient uniquement des concepts $c_j, c_k \in E_c$

TABLE 4.9: Relations taxonomiques extraites automatiquement : comparaison basée sur un sous-ensemble de 21 concepts communs.

	O_{mb}	*WCL System*	*Text2Onto*	O_u
Relation extraites	–	11	7	–
Relations validées	6	9	5	13
$\cap$ O_{mb}	–	3	3	6
$\cap$ *WCL System*	3	–	3	9
$\cap$ *Text2Onto*	3	3	–	5
$Precision^*_m$	–	27%	43%	–
$Precision^*_u$	–	82%	71%	–
$Rappel^*_m$	–	50%	50%	–
$Rappel^*_u$	46%	69%	38%	–
Relations spécifiques[1]	16%	38%	8%	–

[1]Relations identifiées par un seul outil

Le tableau 4.9 montre une complémentarité entre les outils et O_{mb}. Chaque outil identifie des « relations spécifiques », qui n'ont pas été identifiées par l'expert. WCL System semble très performant parce qu'il est le seul à avoir identifié 38% des toutes les relations identifiées entre les 21 concepts.

Les résultats de WCL System et de Text2Onto sont complémentaires quand on analyse la *richesse sémantique* des relations identifiées. En fait, 76% des 111 relations validées extraites par Text2Onto sont des relations sémantiquement pauvres, alors que 76% des 25 relations validées extraites par WCL System sont des relations sémantiquement riches.

4.3.3.4 Analyse des resultats – autres aspects

Seuls deux outils permettent, au moins en théorie, l'extraction automatique de relations non taxonomiques : Text2Onto et Sprat. Malheureusement, dans tous nos tests Sprat n'a extrait aucune relation non taxonomique. Text2Onto a identifié 5 relations *« instance-of »*

qui associent des instances à des concepts ; 4 de ces relations ont été aussi identifiées par l'expert et font partie de O_{mb}. Text2Onto a identifié 15 relations non taxonomiques ; seules 4 de ces relations concernent le domaine de l'ontologie et aucune ne fait partie de O_{mb} (étant donne qu'elle contient uniquement des concepts organisés hiérarchiquement et des instances associées à des concepts).

Comme Sprat a obtenu des résultats très limités dans le cadre de cette expérimentation nous avons réalisé des tests supplémentaires. Plus précisément, comme Sprat peut prendre en entrée une ontologie et l'enrichir avec des nouveaux concepts, nous l'avons testé en lui donnant en entrée, en plus du corpus, une petite ontologie composée d'une sélection des concepts de O_{mb}. Nous avons répété ce test avec différentes ontologies en entrée pour vérifier si le contenu de l'ontologie donnée en entrée influence le processus d'extraction de concepts de Sprat. Dans tous ces tests, l'ontologie obtenue comme résultat a été l'union de l'ontologie donnée en entrée avec les résultats que Sprat obtient sur le corpus quand il ne prend pas une ontologie en entrée. De ce fait, nous pensons que l'ontologie donnée en entrée à Sprat influence peu son processus d'extraction.

Table 4.10: Résultats de Sprat quand nous avons augmenté graduellement la taille du corpus.

Evaluation	Corpus					
	4000*	9000*	15000*	19000*	27000*	8000*
Concepts extraits	9	14	20	27	49	27
Concepts évalués 1	3	3	4	4	6	5
Concepts évalués 2	3	3	3	3	3	1
Concepts évalués 3	0	4	7	10	9	1
Concepts évalués 4	3 (30%)	4 (29%)	6 (30%)	10 (37%)	31 (63%)	20 (74%)

*Le nombre de mots dans le corpus

Nous avons aussi testé Sprat en augmentant progressivement, en quatre étapes, la taille du corpus : de 4000 mots à 27000 mots [6]. A chaque étape nous avons ajouté au corpus un nouveau document [7] du même type que le premier – un article scientifique concernant le même domaine (la construction des ontologies à partir de textes) et ayant subi le même prétraitement que le premier (élimination de toutes les parties qui n'étaient pas du texte brut). Les résultats obtenus sont résumés dans le tableau 4.10. Le pourcentage des concepts notés 4 augmente avec la taille du corpus, mais l'ontologie construite reste très petite et peu structurée, ayant une profondeur maximale de 2. Aucune instance ou relation non taxonomique n'a

6. Les corpus plus larges incluent les corpus plus petits.
7. Les documents ajoutés au corpus :

étape 1 : Velardi et al., 2005, [132] : Evaluation of OntoLearn, a methodology for automatic learning of domain ontologies ;

étape 2 : Velardi et al., 2007, [134] : A Taxonomy Learning Method and its Application to Characterize a Scientific Web Community ;

étape 3 : Velardi et al., 2008, [133] : Mining the Web to Create Specialized Glossaries ;

étape 4 : Navigli et Velardi, 2005, [92] : Structural Semantic Interconnections : A Knowledge-Based Approach to Word Sense Disambiguation.

été identifiée.

Comme nous avons remarqué une variation très importante de la précision et du rappel de Sprat quand nous sommes passés du corpus de 19000 mots à celui de 27000 mots, nous avons aussi testé Sprat sur le corpus composé des 8000 mots qui ont été ajoutés au corpus de 19000 mots pour obtenir celui de 27000 mots. Sprat a obtenu des résultats remarquables sur ce corpus (une précision de 74%). Ces performances sont expliquées par le fait que le corpus de 8000 mots contient, par rapport aux autres corpus, une proportion plus importante d'expressions qui correspondent aux patrons prédéfinis de Sprat (des phrases contenant les expressions *« such as »* et *« kind of »*).

4.4 Conclusion

Dans ce chapitre, nous avons introduit une démarche pour analyser et comparer les approches et les outils pour la construction automatique des ontologies à partir de textes.

Cette démarche nous semble compléter les quelques méthodes proposées précédemment dans la littérature. Bien qu'encore très éloignée d'un système d'aide à la décision qui permettrait de guider l'utilisateur dans le choix d'une approche pour des usages spécifiques, notre démarche peut contribuer à la réflexion préalable au développement d'un tel système. Elle a été éprouvée expérimentalement pour comparer quatre approches et elle nous a permis de dégager leurs avantages et inconvénients. De cette comparaison expérimentale, il ressort que parmi les outils testés Text2Onto offre le meilleur équilibre entre la disponibilité, le degré d'automatisation et la qualité des résultats obtenus. Nous retenons donc cet outil dans la suite pour nos travaux expérimentaux sur l'analyse des erreurs.

5

L'évaluation de la qualité des ontologies

SOMMAIRE

5.1 Introduction

La forte variabilité des résultats obtenus avec les différents outils logiciels testés dans le chapitre 4 nous renvoie naturellement à la question de la validation de la qualité des ontologies construites. Bien que ce sujet fût abordé par Gomez-Perez en 1995 [53], peu de temps après les travaux fondateurs de Gruber, 1993 [57], et que, depuis, un nombre important de travaux y aient été consacrés [49, 58, 61, 76, 130, 139], les solutions proposées semblent ne pas répondre à tous les besoins et de nouvelles approches sont proposées régulièrement [5, 37, 130].

La difficulté de ce problème est dûe à la complexité inhérente de la notion d'ontologie, comme le montre la variété des définitions présentées dans le Chapitre 2, et à la diversité des applications, des contextes et des problématiques dans lesquels les ontologies sont utilisées.

Les approches pour l'évaluation des ontologies proposées dans la littérature se différencient par les critères utilisés pour définir la qualité d'une ontologie. Dans cette thèse, nous utilisons l'expression « modèle de qualité » pour nommer l'ensemble des définitions et des critères utilisés par une approche pour expliciter la notion de qualité.

Dans la première partie de ce chapitre, nous présentons un état de l'art sur la qualité des ontologies (Section 5.2). Nous présentons différents modèles de qualité proposés dans la littérature ainsi que les méthodes d'évaluation et les types de problèmes qui peuvent apparaître dans une ontologie et nuire à sa qualité. Dans une deuxième partie, nous proposons un nouveau cadre formel pour la description des problèmes de qualité et introduisons une nouvelle typologie de ces problèmes (Section 5.3).

5.2 Etat de l'art

Dans la littérature la question de la qualité des ontologies a été abordée selon deux plans : (1) celui de l'évaluation des ontologies (présenté dans la Section 5.2.1), et (2) celui de l'identification et de l'analyse des problèmes qui perturbent la qualité des ontologies (présenté dans la Section 5.2.2).

5.2.1 L'évaluation de la qualité des ontologies

L'évaluation de la qualité des ontologies est un problème complexe et les solutions proposées dans la littérature sont très diversifiées. Ces solutions ont été analysées et classées en fonction de plusieurs critères interdépendants : l'objectif de l'évaluation [37, 54, 64] ; le type d'utilisateur qui peut faire l'évaluation [64] ; le moment où l'évaluation peut être réalisée [64, 130] ; les aspects de l'ontologie qui sont évalués (le modèle de qualité utilisé) [15, 49, 136] ; la méthode d'évaluation [15, 49, 130] ; le caractère automatique ou manuel de l'évaluation [136] ; la nature quantitative ou qualitative de l'évaluation [37] ; etc.

Nous avons choisi de centrer cet état de l'art autour de deux de ces critères – les modèles de qualité et les méthodes d'évaluation – qui permettent une analyse très détaillée des solutions proposées pour l'évaluation des ontologies.

Pour l'usage des termes, bien que la plupart des auteurs utilisent simplement le terme « évaluation », certains auteurs ont introduit et utilisent des termes plus spécifiques pour identifier des types particuliers d'évaluation. C'est notamment le cas de Gomez-Perez et al.

[54] qui utilisent les termes « vérification de l'ontologie » et « validation de l'ontologie » pour identifier deux types complémentaires d'approches d'évaluation. La « vérification de l'ontologie » évalue si l'ontologie construite respecte des spécifications. La « validation de l'ontologie » vérifie la conformité de l'ontologie aux problèmes applicatifs auxquels elle doit répondre. Ces termes et cette différenciation entre les deux types d'évaluation ont été repris par plusieurs auteurs comme, par exemple, Obrst et al. [100] qui ont publié une étude sur les méthodes de validation des ontologies et Vrandecic [136] qui a analysé les méthodes de vérification des ontologies.

La suite de cette section est organisée comme suit. Dans un premier temps (Section 5.2.1.1), nous présentons les modèles de qualité proposés dans la littérature en insistant sur oQual, le modèle proposé par Gangemi et al. [49] qui, à notre connaissance, est le modèle de qualité le mieux structuré et le plus complet. Dans un deuxième temps (Section 5.2.1.2), nous présentons les méthodes d'évaluation des ontologies les plus usitées en les positionnant par rapport au modèle de qualité oQual.

5.2.1.1 Les modèles de qualité

L'objectif majeur de cette section est de présenter oQual, le modèle de qualité que nous considérons comme le mieux structuré et le plus complet. Il ne s'agit donc pas de présenter exhaustivement tous les modèles proposés dans la littérature. Cependant, pour argumenter le choix de oQual, nous proposons tout d'abord un survol synthétique de différentes approches.

5.2.1.1.1 Modèles de qualité inspirés d'autres domaines Certains auteurs se sont inspirés des modèles de qualité définis dans d'autres domaines (la linguistique [22], l'ingénierie logicielle [37, 45], les sciences de l'information [5], l'évaluation des niveaux d'acquisition des connaissances [5]) pour proposer des modèles de qualité adaptés aux ontologies.

Burton-Jones et al. [22] ont repris des définitions de la notion de qualité issues du domaine linguistique. Dans leur modèle, la qualité d'une ontologie est définie à l'aide de dix caractéristiques et se décline autour de quatre dimensions : (1) la qualité syntaxique (validité syntaxique, richesse de la syntaxe utilisée), (2) la qualité sémantique (intelligibilité, consistance, clarté), (3) la qualité pragmatique (complétude, précision, pertinence) et (4) la qualité sociale (autorité, historique). A chacune des dix caractéristiques les auteurs ont associé une mesure permettant une évaluation quantitative. Par exemple, la mesure de la validité syntaxique d'une ontologie est donnée par le rapport entre le nombre de règles de syntaxe non respectées par les formules logiques qui définissent l'ontologie et le nombre de ces formules logiques.

Duque-Ramos et al. [37] ont proposé en 2011 **OQuaRE** [1], un modèle de qualité basé sur le standard ISO 25000. Ce standard définit un processus complet d'évaluation d'un logiciel et complète le standard ISO 9126 qui caractérise la qualité pour les logiciels. Le modèle OQuaRE définit neuf dimensions (ou caractéristiques) de la qualité d'une ontologie et 44 sous-caractéristiques. Une de ces dimensions (la structure) est spécifique aux ontologies. Les huit autres dimensions sont inspirées du standard ISO 25000 : l'adéquation fonctionnelle, la fiabilité, l'opérabilité, la maintenabilité, la qualité d'utilisation ou l'ergonomie, la transférabilité, la compatibilité et l'efficacité de la performance. Duque-Ramos et al. ont aussi proposé

1. http://miuras.inf.um.es/evaluation/oquare/

un ensemble de 14 mesures (la profondeur de l'ontologie, le nombre moyen de relations par concept, etc.) et une association entre ces mesures et les 44 sous-caractéristiques permettant d'évaluer quantitativement chaque dimension de la qualité.

5.2.1.1.2 Modèles de qualité basés sur la dialectique Certains travaux s'inspirent de la dialectique philosophique d'analyse, de justification et de critique des arguments. C'est le cas du modèle de qualité **OntoClean** proposé par Guarino et Welty [61]. Ce modèle permet la vérification de la validité formelle des relations de subsomption (ou taxonomiques) entre les concepts d'une ontologie. Ce modèle repose sur quatre méta-propriétés qui caractérisent les concepts d'une ontologie (l'*identité*, la *rigidité*, l'*unité* et la *dépendance*). Pour être correctes les relations de subsomption d'une ontologie doivent respecter un ensemble de contraintes basées sur ces méta-propriétés. Par exemple, un concept ayant une propriété d'*identité* ne peut pas subsumer un concept qui n'a pas cette propriété. A notre connaissance, Guarino et Welty n'ont associé aucune mesure à leur modèle de qualité.

5.2.1.1.3 Modèles de qualité associés à des définitions de la notion d'ontologie Certains modèles de qualité sont construits autour d'une définition de la notion d'ontologie [49, 76, 130]. C'est le cas des modèles OntoQA et oQual.

OntoQA a été proposé par Tartir et al. [130]. Partant d'une définition formelle de la notion d'ontologie proposée en 2002 par Maedche et Zacharias [83], ces auteurs ont précisé la notion de *schéma (ou structure) de l'ontologie* (les concepts, leurs attributs et les relations entre concepts) et celle de *structure de la base de connaissances* (la structure définie par les instances de l'ontologie). Le modèle OntoQA considère que la qualité d'une ontologie dépend à la fois de la qualité de son schéma et de la manière dont ce schéma est effectivement utilisé pour représenter des connaissances du monde réel (les instances). La qualité du schéma d'une ontologie est évaluée par sa capacité à représenter des connaissances riches. Cette capacité dépend, à son tour, de la richesse des attributs, de la richesse de la structure taxonomique et de la richesse des relations. Le degré d'utilisation effective du schéma dépend de la présence de concepts qui n'ont pas d'instances, de la distribution d'instances dans l'ontologie et des relations qui existent entre les instances. Tartir et al. ont proposé huit mesures (le rapport entre le nombre d'attributs et le nombre de classes, le rapport entre le nombre de relations non taxonomiques et le nombre total de relations d'une ontologie, etc.) permettant une évaluation quantitative de chacune de ces caractéristiques. Notons qu'on retrouve ici des mesures ou des adaptations de mesures bien connues dans les travaux sur les « mesures sémantiques ».

5.2.1.1.4 Le modèle de qualité oQual Dans la plupart des modèles évoqués précédemment la formalisation est faible et, à notre connaissance, aucun modèle ne fournit une explication logique pour le choix des mesures et pour la manière dont ces mesures sont associées et permettent de quantifier les différents aspects de la qualité. Les explications fournies font souvent appel à des intuitions, à des similitudes avec d'autres domaines ou invoquent des questions pragmatiques. Le modèle de qualité **oQual** [49] se distingue de ces modèles par sa complétude, par sa construction logique et par son degré de formalisation. De plus, il a été construit sur la base d'une définition de la notion d'ontologie très proche de celle que nous

avons présentée au début de cette thèse (Section 2.6).

Le modèle oQual propose une modélisation du processus d'évaluation d'une ontologie. La qualité d'une ontologie est définie par rapport au cas d'utilisation envisagé. L'ontologie est considérée comme un objet sémiotique ayant plusieurs dimensions qui peuvent être mesurées. Nous omettons dans la suite les détails de cette modélisation et les définitions formelles que Gangemi et al. ont proposés car leur utilité dépasse le cadre de cette thèse. Nous présentons seulement les éléments qui permettent de décrire et quantifier la qualité d'une ontologie.

Pour caractériser la qualité d'une ontologie Gangemi et al. ont introduit et formalisé la notion de *« qood »* (« quality oriented ontology description ») – description orientée qualité d'une ontologie. Un « qood » exprime les exigences que l'ontologie doit satisfaire pour réussir le cas d'utilisation. Tous les ensembles de critères qui ont été utilisés dans la littérature pour évaluer la qualité d'une ontologie sont des « qoods ». D'après Gangemi et al., tous les « qoods » utilisés dans la littérature sont des combinaisons de neuf « qoods » élémentaires :

1. l'ergonomie cognitive : une ontologie peut être facilement comprise, manipulée et exploitée par ses utilisateurs ;
2. la transparence : une ontologie permet de comprendre, grâce à sa formalisation et à ses annotations, les choix conceptuels et les motivations de ses concepteurs ;
3. l'intégrité et l'efficacité computationnelle : une ontologie peut être analysée par un système d'inférence (elle est consistante et permet des inférences) et la complexité de cette opération ;
4. la méta-intégrité (« meta-level integrity ») : une ontologie respecte des critères de structuration taxonomique qui sont acceptés comme indicateurs de qualité ;
5. la flexibilité (« context-boundedness ») : une ontologie peut être facilement adaptée pour exprimer une nouvelle perspective sur les connaissances qu'elle modélise ;
6. la conformité par rapport à une expertise : une ontologie est conforme aux connaissances d'un ou des utilisateurs ;
7. la conformité par rapport aux règles d'extension, d'intégration et d'adaptation : une ontologie respecte des règles qui garantissent qu'elle peut être facilement comprise et manipulée dans le but de l'adapter et la réutiliser ;
8. l'adéquation organisationnelle (« organizational fitness ») : une ontologie peut être intégrée facilement et correspond aux besoins d'une organisation ;
9. l'accessibilité générique : une ontologie peut être facilement utilisée de manière effective dans une application.

Ces « qoods » élémentaires ne sont pas directement mesurables. Gangemi et al. ont recensé 39 aspects de l'ontologie qui peuvent être mesurés directement (par exemple, le nombre de concepts de l'ontologie, la profondeur du graphe correspondant à l'ontologie, la précision et le rappel par rapport à un modèle) et ont établi des corrélations (positives et négatives) entre ces aspects et les « qoods » élémentaires. Par exemple, pour l'« intégrité et l'efficacité computationnelle » ont été établies deux correlations positives (avec les aspects mesurables (1) "consistance logique" et (2) "ratio des disjonctions") et trois correlations négatives (avec les aspects mesurables (3) "degré d'enchevêtrement (*tangledness*) du graphe", (4) "nombre de restrictions" et (5) "nombre de cycles").

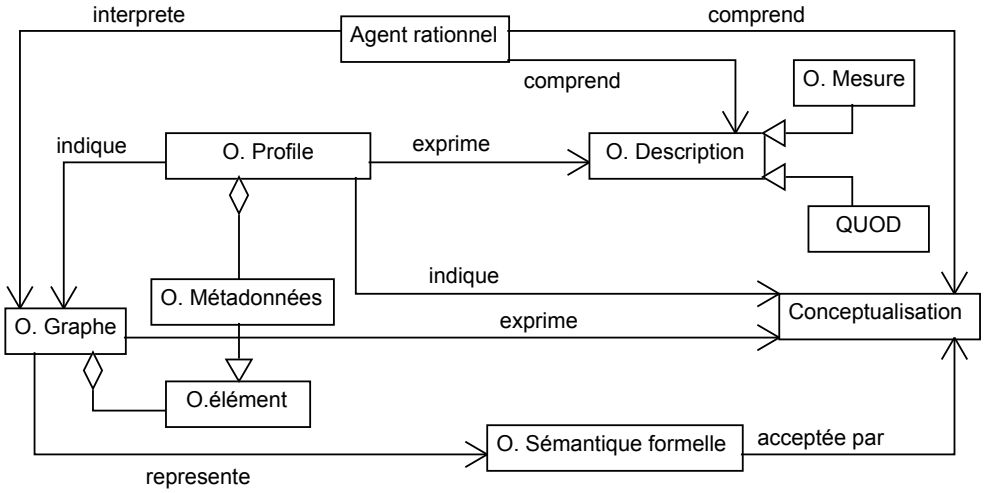

Figure 5.1: Diagramme UML (adaptée de Gangemi et al. [49]) illustrant les relations entre les éléments utilisés pour définir une ontologie en tant qu'objet informationnel.

Pour recenser les aspects mesurables d'une ontologie Gangemi et al. [49] se sont inspirés de travaux en sémiologie qui les ont conduits à définir l'ontologie comme un *objet sémiotique*, i.e. un *objet informationnel* (« *information object* ») et une *conceptualisation intentionnelle* (« *intended conceptualization* ») établis dans un *cadre* (« *communication setting* ») donné (figure 5.1). Un *objet informationnel* est une structure qui exprime une information, une signification ; il peut être interprété par un *agent rationnel*. Une *conceptualisation intentionnelle* est une information, une signification qu'on a l'intention d'exprimer à l'aide de l'*objet informationnel*. Le *cadre* dans lequel ces deux éléments sont établis est formé par le système d'encodage utilisé pour exprimer l'objet informationnel (le langage OWL), par le contexte d'utilisation et la tâche prévus pour l'*objet informationnel* et par le ou les *agents rationnels* qui vont interpréter l'ontologie.

En analysant les ontologies comme des objets sémiotiques, trois dimensions mesurables peuvent être mises en évidence [49] : (1) la dimension structurelle, (2) la dimension fonctionnelle et (3) la dimension d'utilisabilité. Selon la dimension structurelle, l'ontologie est perçue comme un *objet informationnel* et l'analyse concerne deux aspects de sa structure : l'aspect combinatoire (la structure formée par les concepts et les relations de l'ontologie modélisée par un graphe) et la sémantique formelle (la structure modélisée par un modèle logique). Selon la dimension fonctionnelle, l'ontologie est perçue comme un *langage* (un *objet informationnel* + une *conceptualisation intentionnelle*). L'analyse de l'ontologie vise à évaluer dans quelle mesure l'objet informationnel correspond à la conceptualisation intentionnelle i.e. dans quelle mesure l'ontologie répond à son objectif. Selon la dimension utilisabilité, l'analyse de l'ontologie porte sur le métalangage associé à celle-ci (l'ensemble des métadonnées associées à l'ontologie, par exemple les annotations, qui sont susceptibles d'aider un utilisateur à comprendre l'ontologie).

5.2.1.2 Les méthodes d'évaluation

Les méthodes d'évaluation visent à expliciter l'utilisation effective d'un modèle de qualité. Par exemple, si une mesure est associée à un modèle de qualité, la méthode d'évaluation indique la procedure opérationnelle de calcul de la mesure.

Dans cette section, on présente, pour chacune des trois dimensions de l'ontologie mises en évidence par le modèle oQual, les méthodes d'évaluation les plus importantes.

5.2.1.2.1 L'évaluation de la dimension structurelle Pour l'évaluation de la structure combinatoire des ontologies Gangemi et al. [49] ont proposé 23 mesures caractérisant les propriétés du graphe représentant la structure de l'ontologie : des mesures pour la profondeur et la largeur du graphe ; des mesures pour le degré d'enchevêtrement (« tangledness ») du graphe – le rapport entre le nombre des noeuds du graphe et le nombre de noeuds qui ont au moins deux pères dans la taxonomie (arbre associé à la relations « is a ») ; des mesures concernant le degré de dispersion des noeuds (« fan-outness ») du graphe ; des mesures concernant la « différence spécifique » du graphe – la proportion des noeuds qui ont le même père dans la taxonomie et qui ont en commun une relation non taxonomique ; des mesures pour la densité et pour la modularité du graphe.

Neuf mesures spécifiques ont été également proposées par Gangemi et al. [49] pour évaluer la sémantique formelle de l'ontologie : 8 concernent l'« adéquation logique » de l'ontologie (par exemple le rapport entre le nombre de concepts et le nombre de relations, le rapport entre le nombre de concepts et le nombre d'axiomes, la proportion de relations pour lesquelles une relation inverse existe, etc.) ; une mesure concerne l'« adéquation métalogique » de l'ontologie (le rapport entre le nombre de concepts consistants par rapport aux critères du modèle OntoClean et le nombre total de concepts de l'ontologie).

Le calcul effectif de ces mesures se déduit aisément du comptage préalable des cardinaux des différents ensembles utilisés dans leurs définitions (ensemble des noeuds, des concepts inconsistants, etc.).

5.2.1.2.2 L'évaluation de la dimension fonctionnelle La dimension fonctionnelle des ontologies est évaluée en mesurant l'écart entre l'ontologie (ou, plus précisément, entre ce que l'objet informationnel exprime) et la conceptualisation intentionnelle (ce que l'objet informationnel doit exprimer).

Gangemi et al. [49] s'appuient sur deux mesures classiques pour évaluer cet écart : la précision et le rappel. Ces mesures reposent sur les définitions introduites dans la Section 2.6 pour les notions de « l'ensemble des modèles correspondant à l'ontologie » et de « l'ensemble des modèles intentionnels » :

- la précision de l'ontologie est le rapport entre le nombre de modèles de l'ontologie qui sont des modèles intentionnels et le nombre total des modèles de l'ontologie ;
- le rappel (ou la couverture) de l'ontologie est le rapport entre le nombre de modèles de l'ontologie qui sont des modèles intentionnels et le nombre de modèles intentionnels possibles.

En pratique, il n'est pas envisageable de compter le nombre de modèles d'une ontologie ou le nombre de modèles intentionnels possibles. Pour mesurer l'écart entre l'ontologie et la conceptualisation intentionnelle, les méthodes d'évaluation s'appuient sur une expression

« concrète » qui approxime au mieux la conceptualisation intentionnelle. Brank et al. [15] ont identifié dans les méthodes d'évaluation quatre manières d'exprimer la conceptualisation intentionnelle : (1) le jugement humain ; (2) une structure qui exprime formellement la conceptualisation intentionnelle (par exemple une ontologie, un ensemble d'ontologies ou une base de données) ; (3) un jeu de données qui exprime de façon non structurée la conceptualisation intentionnelle ; (4) les spécifications ou les résultats attendus d'une application qui utilise l'ontologie.

Les méthodes d'évaluation qui utilisent le jugement humain comme expression de la conceptualisation intentionnelle sont nommées *méthodes boîte noire* (parce que la conceptualisation n'est pas explicitement exprimée), et dualement les autres méthodes sont nommées *méthodes boîte transparente* [49].

Plus précisément, dans le cas (1), les experts indiquent sur une échelle dans quelle mesure certains aspects de l'ontologie correspondent à leurs connaissances et à leur jugement [5, 76, 93, 135].

Dans le cas (2), l'ontologie est comparée à une référence (« golden standard ») – la structure qui exprime la conceptualisation intentionnelle. Certaines méthodes comparent l'ontologie et la référence au niveau du lexique [26, 81, 116], d'autres essayent d'aligner l'ontologie et la référence [16, 34, 49, 86, 142].

Dans le cas (3), la comparaison de l'ontologie avec le jeu de données [17, 32, 122] est réalisée en deux étapes : un prétraitement du jeu de données à l'aide d'algorithmes spécifiques (notamment des algorithmes de traitement du langage naturel) pour extraire des connaissances structurées suivie d'une comparaison similaire au cas (2) entre l'ontologie et les connaissances structurées extraites.

Dans le cas (4), l'ontologie est comparée avec les spécifications qui définissent les cas d'utilisation [49], ou les résultats obtenus par une application qui utilise l'ontologie sont comparés aux résultats qui étaient attendus [104, 140, 141].

5.2.1.2.3 L'évaluation de la dimension d'utilisabilité Pour évaluer cette facette des ontologies, Gangemi et al. [49] ont proposé quatre mesures focalisées exclusivement sur les annotations associées à l'ontologie et à ses composants : présence, quantité, complétude et fiabilité. A notre connaissance, mis à part Gangemi et al., peu d'auteurs se sont intéressés à l'évaluation de ces aspects.

Comme pour les mesures pour la dimension structurelle, la méthode utilisée pour évaluer concrètement cette dimension est le comptage des différentes annotations et l'application des formules proposées.

5.2.2 Problèmes affectant la qualité d'une ontologie

En sus d'une analyse globale de la qualité, il est important d'un point de vue opérationnel d'identifier explicitement la liste des problèmes qui nuisent à la qualité d'une ontologie. Ces problèmes sont souvent identifiés sous le terme d'*erreurs* mais certaines études publiées durant cette dernière décennie ont introduit des termes plus spécifiques : (1) les *erreurs de taxonomie* ou *erreurs structurelles* [10, 19, 39, 54, 55] ; (2) les *défaillances* ou les *anomalies de conception* (« *design anomalies* ») [10, 11] ; (3) les *anti-patrons* [19, 30, 106, 114] ; (4)

les *embûches* (« *pitfalls* ») ou *mauvaises habitudes* (« *worst practices* ») [106, 107] ; (5) les *défauts logiques* [19].

On pourrait ajouter à cette énumération (6) les *erreurs de syntaxe* [19] et (7) les erreurs de type *données liées* (« *linked data* ») [19]. Les *erreurs de syntaxe* concernent les violations des conventions du langage utilisé pour représenter l'ontologie. Nous ne les considérons pas dans cette thèse car, de façon générale, nous n'abordons pas les erreurs qui relèvent de l'ingénierie logicielle. Les erreurs de type *données liées* concernent les importations de concepts entre différentes ontologies – ce que permettent certains moteurs d'inférences lorsque ces dernières sont décrites en OWL. Cependant, à notre connaissance, ces importations sont peu utilisées en pratique.

5.2.2.1 Les erreurs de taxonomie

Depuis les travaux de Gomez-Perez et al. [54, 55] l'expression *erreurs de taxonomie* est utilisée pour identifier trois types d'erreurs qui peuvent affecter la structure taxonomique de l'ontologie : les inconsistances, les incomplétudes et les redondances.

Gomez-Perez et al. [55] ont décrit trois types d'inconsistances qui affectent la structure taxonomique des ontologies :

1. les erreurs de circularité (par exemple un concept qui est une spécialisation ou une généralisation de lui même) ;
2. les erreurs de partitionnement qui produisent des inconsistances logiques (par exemple un concept qui est la spécialisation de deux concepts qui sont disjoints) ;
3. les erreurs de sémantique (par exemple une relation taxonomique qui n'est pas consistante avec le sens de ses deux concepts).

Dans les deux premiers cas, l'inconsistance apparaît sur un plan logique, alors que dans le troisième cas l'inconsistance se manifeste sur le plan sémantique.

Les erreurs d'incomplétude concernent l'absence de certaines informations dans l'ontologie. Deux types d'incomplétudes ont été décrits :

1. l'absence des relations taxonomiques, qui rend la structure taxonomique incomplète ;
2. l'absence d'axiomes indiquant les éventuelles propriétés (exhaustivité, disjonction) des différents partitionnements des concepts.

Les redondances apparaissent quand la même information est présente plusieurs fois dans l'ontologie. Deux types de redondances ont été ainsi décrits :

1. la déclaration explicite d'une relation taxonomique qui peut être déduite, par inférence, à partir des autres relations taxonomiques présentes dans l'ontologie ;
2. l'existence des concepts ou des instances qui ont des définitions formelles identiques.

5.2.2.2 Les anomalies de conception

L'expression *anomalies de conception* (« *design anomalies* »), utilisée dans l'ingénierie logicielle, a été utilisée par Baumeister et al. [10, 11] pour identifier des situations qui, bien que exemptées de toute erreur logique, posent des problèmes pour la compréhension et la maintenance des ontologies. Cinq types d'*anomalies de conception* ont été décrits :

1. les *concepts isolés* (« *lazy concepts* ») qui sont des concepts de l'ontologie qui n'ont pas d'instances et qui ne sont impliqués dans aucune relation ou axiome ;
2. les *chaînes d'héritage* qui sont de longues chaînes de spécialisation où les concepts intermédiaires ont un seul père et un seul fils et ne sont pas impliqués dans d'autres relations ou axiomes ;
3. les *disjonctions lointaines* (« *lonely disjoint* ») qui sont des axiomes de disjonction superflus qui peuvent affecter le processus d'inférence ;
4. la *surspécialisation du domaine des propriétés* (« *over-specific property range* ») – par exemple, quand pour une propriété, « temperature », il est possible de choisir parmi 5 valeurs très spécifiques : *Domaine(temperature) = {very high ; high ; normal ; low ; very low}*, alors que l'application envisagée ne nécessite pas une telle précision mais seulement un choix parmi 3 valeurs : *Domaine'(temperature) = {high' ; normal ; low'}* ;
5. les *groupements répétitifs de propriétés* (« *property clumps* »), quand un group de deux ou plusieurs propriétés apparaît, en se répétant, dans la définition d'au moins deux concepts.

5.2.2.3 Les anti-patrons

Les notions de *patron de conception ontologique* (« *ontology design pattern* ») et d'*anti-patron* (« *anti-pattern* ») sont inspirées du génie logiciel. Les *patrons de conception ontologique* sont des formalisations des solutions de modélisation utilisées par les experts pour résoudre des problèmes récurrents de conception [50]. Les *anti-patrons* sont, par opposition, des formalisations de mauvais choix de modélisation qui sont connus *a priori* pour provoquer « *des erreurs, des comportements inattendus ou une lenteur à l'exécution* » [115].

Trois types d'anti-patrons ont été décrits :

1. les *anti-patrons logiques* – des expressions contenant des conflits détectables par les raisonneurs de la logique de description ;
2. les *anti-patrons cognitifs* – des expressions contenant des erreurs de modélisation dûes à une méconnaissance des axiomes logiques ;
3. les *conseils* (« *guidelines* ») – des expressions complexes qui sont correctes d'un point de vue logique et cognitif mais qui peuvent être exprimées de façon plus simple et précise.

Roussey et al. [115] ont décrit 4 anti-patrons logiques, 3 anti-patrons cognitifs et 4 « conseils ». Les définitions de ces 11 anti-patrons sont présentées dans la Section A.2.

Dans la littérature, une attention particulière a été accordée aux patrons décrivant des situations dont la présence dans une ontologie réduit de manière significative les performances des moteurs d'inférences [19, 75]. Bien que, dans la littérature, ces patrons n'aient pas été inclus dans les différentes listes d'anti-patrons, nous considérons qu'ils peuvent être considérés comme des cas particuliers d'anti-patrons de type "conseils".

5.2.2.4 Les embûches

La notion d'*embûche* (« *pitfall* ») a été introduite comme un complément à la notion de *patron de conception ontologique* et a été utilisée pour identifier les situations problématiques qui existent dans une ontologie et qui ne correspondent à aucun patron connu. Poveda et al. [106, 107] ont identifié expérimentalement et décrit de façon informelle 24 types d'embûches (ces descriptions sont présentées dans la Section A.1). Ils ont positionné ces embûches dans sept classes inspirées des trois dimensions mesurables des ontologies (Section 5.2.1.1) .

5.2.2.4.1 Positionnement par rapport aux dimensions de l'ontologie

Quatre classes correspondent à la dimension structurelle :

1. *décisions de conception* (« *modelling decisions* », *MD*) – des situations où les primitives du langage OWL ne sont pas utilisées de façon appropriée ;
2. *inférences fausses* (« *wrong inference* », *WI*) – des situations où, à cause d'axiomes imprécis, des connaissances fausses sont déduites par inférence ;
3. *inférences impossibles* (« *no inference* », *NI*) – des situations (absence de relations et d'axiomes) qui empêchent le processus d'inférence et ne permettent pas la découverte de connaissances nouvelles et désirables ;
4. *conception réaliste* (« *real world modeling* », *RWM*) – l'ontologie ne contient pas les informations de « bon sens » associées à ses éléments.

Une classe d'embûches correspond à la dimension fonctionnelle :

5. *satisfaction des spécifications* (« *requirement completeness* », *RC*) – l'ontologie ne correspond pas aux spécifications.

Deux classes d'embûches correspondent à la dimension d'utilisabilité :

6. *compréhensibilité de l'ontologie* (« *ontology understanding* », *OU*) – des situations qui rendent difficile la compréhension de l'ontologie, comme, par exemple, l'utilisation d'une étiquette polysème pour un concept ;
7. *clarté de l'ontologie* (« *ontology clarity* », *OC*) – des situations qui nuisent à la clarté de l'ontologie comme, par exemple, l'utilisation, dans le cadre d'une même ontologie, de plusieurs écritures des étiquettes des concepts.

A partir de cette description informelle, il est facile de déduire que les sept classes ne sont pas complètement disjointes et qu'il existe des *embûches* qui peuvent être positionnées dans plusieurs classes simultanément. Par exemple, le fait que l'ontologie ne précise pas qu'une relation soit l'inverse d'une autre relation correspond à la fois au cas « inférences impossibles » et au cas « compréhensibilité de l'ontologie ».

5.2.2.4.2 Positionnement par rapport à la typologie des erreurs de taxonomie

Poveda et al. [107] ont également essayé de positionner les embûches dans les trois classes d'erreurs de taxonomie décrites par Gomez-Perez et al. [55] et d'aligner leur classification avec celle de Gomez-Perez et al.

Mais, comme les erreurs de taxonomie portent uniquement sur la structure et le contenu de l'ontologie, quatre des embûches décrites par Poveda et al. concernant le contexte de

l'ontologie n'ont pu être positionnées dans une classe. Et, seules quatre des sept classes de la classification de Poveda et al. ont pu être alignées. Trois d'entre elles (inférences impossibles, satisfaction des spécifications et conception réaliste) couvrent des cas particuliers d'incomplétude et la quatrième (inférences fausses) correspond à des cas particuliers d'inconsistance. Les embûches des trois autres classes (compréhension de l'ontologie, clarté de l'ontologie et décisions de conception) peuvent se retrouver dans plusieurs classes de la typologie de Gomez-Perez et al. ou couvrent des situations qui n'ont pas pu être positionnées dans cette typologie.

5.2.2.5 Les défauts logiques

Les *défauts logiques* incluent les concepts insatisfiables (les concepts qui n'acceptent aucune instance) et les inconsistances logiques. Une partie des défauts logiques sont des erreurs taxonomiques (voir Section 5.2.2.1), mais pas tous (par exemple, le défaut logique « deux concepts de l'ontologie qui sont déclarés comme étant, en même temps, disjoints et équivalents » n'est pas une erreur taxonomique). Dans la littérature, ces défauts sont aussi identifiés sous le nom de *erreurs sémantiques* [19], terme utilisé avec un sens différent de celui de la Section 5.2.2.1 : ici il fait référence à des erreurs logiques, identifiables par les moteurs d'inférences, alors que dans la Section 5.2.2.1 il fait référence à des contradictions entre le contenu de l'ontologie et les connaissances des utilisateurs.

5.3 Vers une typologie des problèmes

L'état de l'art présenté dans la section précédente a mis en évidence l'hétérogénéité de la terminologie utilisée pour identifier les différents problèmes qui affectent la qualité des ontologies. Dans cette thèse, nous préférons utiliser un terme très générique – *« problème »* – pour identifier toutes les situations non optimales du point de vue de la qualité des ontologies. Ce terme couvre donc à la fois les erreurs, les anomalies, les embûches et les situations correspondant à des anti-patrons qui ont été décrites dans la littérature.

Dans la Section 5.3.1, nous proposons un cadre formel qui vise une description explicite des problèmes qui affectent les ontologies. Sur la base de ce cadre formel, nous introduisons ensuite dans la Section 5.3.2 une nouvelle typologie de ces problèmes qui a pour ambition d'offrir une vue unitaire et cohérente sur les problèmes de qualité. Elle est une première étape vers une démarche de vérification systématique des ontologies.

5.3.1 Cadre formel

Notre cadre formel recourt à plusieurs notions (conceptualisation, interprétation, modèles intentionnels, modèles de l'ontologie) définies formellement par Guarino et al. [60] et introduites dans la Section 2.6. Nous nous sommes aussi inspirés du modèle de qualité oQual [49], qui a été introduit dans la Section 5.2.1.1, et des travaux sur la qualité des modèles conceptuels (le modèle SEQUAL [72, 73]) et sur la qualité en génie logiciel (les standards ISO 9126 et ISO 25000).

Le modèle de qualité SEQUAL propose une déclinaison de la notion de qualité des modèles conceptuels sur la base des relations entre le modèle, le domaine de modélisation, l'objectif de la modélisation, le langage utilisé pour la modélisation et les interprétations que les acteurs techniques et sociaux donnent au modèle. L'idée qu'on a retenue est la séparation entre l'acteur technique et l'acteur social qui peuvent interpréter différemment le même modèle.

Les standards ISO 9126 et ISO 25000 définissent un modèle de qualité pour les logiciels. Ce modèle est composé de six groupes de caractéristiques qui définissent la qualité d'un logiciel :

1. capacité fonctionnelle : pertinence, exactitude, interopérabilité, sécurité, conformité aux spécifications fonctionnelles ;
2. fiabilité : maturité, tolérance aux pannes, facilité de récupération, conformité aux spécifications de fiabilité ;
3. utilisabilité (ou facilité d'utilisation) : facilité de compréhension, facilité d'apprentissage, facilité d'exploitation, pouvoir d'attraction, conformité aux spécifications d'utilisabilité ;
4. rendement / efficacité : comportement temporel, utilisation des ressources, conformité aux spécifications de rendement ;
5. maintenabilité : facilité d'analyse, facilité de modification, stabilité, testabilité, conformité aux spécifications de maintenabilité ;
6. portabilité : facilité d'adaptation, facilité d'installation, coexistence, interchangeabilité, conformité aux spécifications de portabilité.

L'idée qu'on a retenue du modèle de qualité oQual et des standards ISO 9126 et ISO 25000 est la séparation entre les aspects fonctionnels et les aspects non fonctionnels d'une ontologie, respectivement d'un logiciel.

5.3.1.1 Les dimensions du cadre formel

Le cadre formel que nous proposons est composé de deux dimensions complémentaires : (1) erreurs *vs* situations indésirables et (2) aspect logique *vs* aspect social de l'ontologie.

5.3.1.1.1 Erreurs et situations indésirables

Les erreurs sont les problèmes qui rendent l'ontologie inutilisable : soit l'ontologie ne peut pas être utilisée dans certains cas d'utilisation, soit les résultats obtenus ne sont pas conformes aux attentes.

Les situations indésirables sont des problèmes qui affectent l'ontologie sans la rendre inutilisable ; l'ontologie peut être utilisée dans l'ensemble des cas d'utilisation prévus et les résultats obtenus sont conformes aux attentes.

Si l'on fait référence aux standard ISO 9126 et ISO 25000, les erreurs touchent aux qualités « fonctionnelles » d'une ontologie, alors que les situations non désirables touchent aux qualités « non fonctionnelles » de celle-ci.

5.3.1.1.2 Aspect logique et aspect social

Les définitions d'une ontologie introduites au début de cette thèse (Chapitre 2) mettent en évidence une forme de dualité : à la fois une spécification formelle et explicite et en même temps le résultat de la mise en commun et du partage des connaissances de plusieurs personnes. Une ontologie peut être utilisée et interprétée en même temps par des machines (des acteurs techniques, dans le langage du modèle SEQUAL) et par des humains (des acteurs sociaux).

Les machines interprètent une ontologie sur la base de ses spécifications logiques et utilisent un moteur d'inférence pour faire des déductions à partir de ces spécifications. Les utilisateurs humains interprètent l'ontologie non seulement sur la base de ses spécifications logiques, des étiquettes associées aux concepts, des annotations (de l'ontologie et de ses éléments) mais aussi de leurs expériences, leurs vécus, leurs *connaissances tacites* (voir Section 2.2).

Dans cette section, on considère que les connaissances explicites sont exprimées sur la forme de modèles intentionnels, mais un ensemble d'exemples et de contre-exemples peut jouer un rôle similaire.

5.3.2 Proposition d'une typologie

Notre typologie s'appuie sur une déclinaison des deux aspects de la deuxième dimension (aspect logiques / aspect social) selon les deux aspects de la première (erreurs / situations indésirables).

Pour formaliser ce cadre et pour décrire les exemples de la partie logique, où cette formalisation est possible, nous utilisons le formalisme de la logique de description et les deux relations $\models$, $\vdash$ qui existent dans tout langage logique. Rappelons que la relation $\models$ est utilisée pour indiquer, d'une part, qu'une interprétation I est un modèle d'une théorie logique T (toutes les formules de T sont vraies dans I : pour chaque formule $\varphi \in T$, $I \models \varphi$), et, d'autre part, pour exprimer la conséquence logique (tout modèle d'une théorie logique T est aussi le modèle d'une formule : $T \models \varphi$). La relation $\vdash$ est utilisée pour indiquer le calcul logique : l'ensemble de règles utilisées pour prouver un théorème (toute formule) φ en partant d'une théorie T, s'écrit $T \vdash \varphi$.

5.3.2.1 Problèmes qui affectent l'aspect logique des ontologies

Les problèmes qui relèvent de la facette logique d'une ontologie sont soit des problèmes internes à l'ensemble de formules qui compose l'ontologie (contradictions internes, redondances, etc.), soit des problèmes dûs à des différences entre l'ontologie et ses modèles intentionnels. Dans les paragraphes suivants, on présente les différents problèmes logiques en précisant s'ils concernent le rapport de l'ontologie avec ses modèles intentionnels.

5.3.2.1.1 Erreurs logiques

On distingue 5 erreurs logiques :

L1. *Inconsistance logique*

Présence de contradictions logiques dans une ontologie pour laquelle aucun modèle ne peut exister (comme l'ensemble des modèles intentionnels n'est jamais vide, une ontologie sans modèle n'a aucun sens). Formellement, étant donnée une ontologie O et la relation $\models$ (conséquence logique) correspondant au langage logique L utilisé pour construire O, il n'existe aucune interprétation I de O telle que $I \models O$. Par exemple, si une ontologie contient les axiomes suivants $c_j \subseteq c_i$ (c_j *is-a* c_i), $c_i \cap c_j \subseteq \bot$ (c_i et c_j sont *disjoints*), $\{d\} \subseteq c_j$ (d est *instance-of* c_j), alors $\{d\} \subseteq c_i$ et $\{d\} \subseteq c_i \cap c_j$, il y a donc une contradiction logique dans la définition de cette ontologie ;

L2. *Ontologie inadaptée* par rapport aux modèles intentionnels

Situations où un fait qui est vrai dans l'ontologie est faux dans au moins un des modèles intentionnels de L. Formellement, il existe une formule φ telle que pour au moins un des modèles intentionnels de L, φ est fausse et $O \models \varphi$. Par exemple, si on a dans l'ontologie deux concepts c_i et c_j qui sont déclarés comme disjoints ($O \models c_i \cap c_j \subseteq \bot$) et que dans au moins un des modèles intentionnels il existe une instance d qui est commune à c_i et c_j ($\{d\} \subseteq c_i \cap c_j$), alors l'ontologie est inadaptée ;

L3. *Ontologie incomplète* par rapport aux modèles intentionnels

Ontologies qui n'affirment pas et ne permettent pas la déduction d'un fait qui est vrai dans tous les modèles intentionnels de O. Formellement, il existe une formule φ telle que pour chaque modèle intentionnel de L, φ est vraie et $O \not\models \varphi$. Par exemple, si dans tous les modèles intentionnels $c_k \cup c_j = c_i$, et l'ontologie O définit $c_j \subseteq c_i$ et $c_k \subseteq c_i$, alors il n'est pas possible de prouver ou conclure que $c_k \cup c_j = c_i$;

L4. *Inférences incorrectes* par rapport aux conséquences logiques

Situations où certaines conclusions sont obtenues par inférence, en utilisant un système de raisonnement considéré comme acceptable pour les applications envisagées pour l'ontologie, et où ces conclusions ne sont pas vraies dans les modèles intentionnels et ne doivent pouvoir être dérivées par aucun système d'inférence acceptable pour les applications envisagées pour l'ontologie. Formellement, une formule φ, qui est fausse dans les modèles intentionnels $O \not\models \varphi$, peut être déduite par inférence $O \vdash \varphi$ par un des systèmes d'inférence considérés comme acceptables ;

L5. *Inférences incomplètes* par rapport aux conséquences logiques

Situations où certaines conclusions ne peuvent pas être obtenues par inférence, en utilisant un système de raisonnement considéré comme acceptable pour les applications envisagées pour l'ontologie, bien que ces conclusions soient vraies dans les modèles intentionnels et doivent pouvoir être dérivées par tous les systèmes d'inférence acceptables pour les applications envisagées pour l'ontologie. Formellement, une formule φ, qui est vraie dans les modèles intentionnels $O \models \varphi$, ne peut pas être déduite par inférence $O \nvdash \varphi$ par les systèmes d'inférence considérés comme acceptables.

5.3.2.1.2 Situations indésirables logiques

On distingue 6 situations indésirables logiques répertoriées ci-dessous. Ces situations ont un impact négatif sur la qualité non fonctionnelle des ontologies. Les dimensions non fonctionnelles de la qualité (par exemple, la réutilisabilité, la maintenabilité, l'efficacité, etc.) sont définies dans les standards pour la qualité des logiciels ISO 9126 et ISO 25000.

L6. *Equivalence logique des éléments distincts de l'ontologie* (concepts / instances / relations)

Situations où deux éléments distincts de l'ontologie sont équivalents du point de vue logique. Par exemple, si c_i et c_j sont deux concepts dans O et $O \models c_i = c_j$;

L7. *Eléments d'ontologie symétriques, indifférenciables du point de vue logique*

Situations où il est impossible de prouver que deux éléments distincts de l'ontologie sont différents du point de vue logique ; plus précisément, quand il est impossible de prouver ces formules : ($O \models c_i = c_j$), ($O \models c_i \cap c_j \subseteq \bot$) et ($O \models \{d\} \subseteq c_i$ et $O \models \{d\} \subseteq c_j$). Ce cas (7) est couvert partiellement par le cas (3) présenté plus haut quand les modèles intentionnels fournissent des informations précises quant à l'équivalence ou la différence entre c_i et c_j ;

L8. *Eléments d'ontologie OU*

Elément c_i de l'ontologie défini comme la disjonction $c_k \cup c_j$, $c_i \neq c_k, c_j$ mais où il n'existe pas au moins un rôle non optionnel ou une propriété non optionnelle qui soit commune à c_k et à c_j et où c_k et c_j n'ont aucune instance commune. Dans le premier cas, une formalisation simple est de dire qu'il n'existe pas un rôle non optionnel r de sorte que $O \models (c_k \cup c_j) \subseteq \exists r.\top$; dans le deuxième cas, une formalisation encore plus simple est $O \models \{d\} \subseteq c_k$ et $O \models \{d\} \subseteq c_j$, où d est une constante qui ne fait pas partie de O. Le premier cas cible les éléments de l'ontologie potentiellement hétérogènes comme $Voiture \cup Personne$, qui n'ont probablement aucune équivalence dans les modèles intentionnels, ce qui peut conduire à une ontologie inadaptée conformément au cas 2. Le deuxième cas cible les éléments de l'ontologie potentiellement ambigus comme, par exemple, un rôle (ou une propriété) r équivalent du point de vue logique à la disjonction ($r_1 \cup r_2$), ($r_1 \cap r_2$) étant satisfiable ;

L9. *Eléments d'ontologie ET*

Elément c_i d'ontologie équivalent à une conjonction $c_k \cap c_j$, $c_i \neq c_k, c_j$ mais pour lequel il n'existe pas au moins un rôle ou une propriété commune pour c_k et c_j. Ce cas est important pour limiter autant que possible les éléments de l'ontologie potentiellement hétérogènes comme $Voiture \cap Personne$, qui peuvent conduire à des éléments d'ontologie non satisfiables ;

L10. *Insatisfiabilité*

Si certains des cas d'insatisfiabilité des éléments de l'ontologie (concepts, rôles, propriétés, etc.) sont couverts par le cas (2) car les modèles intentionnels ne peuvent pas contenir des concepts vides, l'insatisfiabilité en elle-même est une situation indésirable pour les éléments de l'ontologie : étant donné c_i un élément de l'ontologie, $O \models c_i \subseteq \bot$. Bien qu'il soit possible de définir ce qui ne doit pas être vrai (au lieu de définir ce qui doit être vrai), cette pratique n'est pas encouragée ;

L11. *Complexité élevée des inférences*

Lorsque un fait est exprimé dans l'ontologie d'une façon qui complique le raisonnement, alors qu'il existe des manières plus simple d'exprimer la même chose ;

L12. *Ontologie non minimale*

Ontologie contenant des informations qui ne sont pas nécessaires. Une information est considérée comme non nécessaire quand
- elle peut être dérivée des autres informations présentes. Par exemple, la redondance des relations taxonomiques : si on a dans une ontologie ces trois axiomes $c_i \subseteq c_j$, $c_j \subseteq c_k$, et $c_i \subseteq c_k$, le dernier axiome peut être déduit à partir des deux premiers ;
- elle ne fait pas partie des modèles intentionnels. Par exemple, un concept c_i qui fait partie de l'ontologie mais qui n'est pas défini par les modèles intentionnels.

5.3.2.2 Problèmes qui affectent l'aspect social des ontologies

Les problèmes qui relèvent de la facette sociale des ontologies sont en lien avec la perception (l'interprétation) que l'utilisateur donne à l'ontologie et avec l'utilisation que les utilisateurs humains font de l'ontologie. La perception et l'utilisation envisagées peuvent ne pas être exprimées de façon formelle. D'une certaine façon, une autre distinction entre la facette sociale et la facette logique d'une ontologie repose aussi sur la différence entre les connaissances tacites et les connaissances explicites.

5.3.2.2.1 Erreurs sociales

On distingue 4 erreurs sociales :

S1. *Contradiction sociale*

La perception (l'interprétation) que l'acteur social donne à l'ontologie ou à certains de ses éléments est en contradiction avec les axiomes de l'ontologie (analogie avec *ontologie inadaptée*) ;

S2. *Perception d'erreurs de conception*

L'acteur social indique des erreurs de conception comme, par exemple, le fait qu'une notion a été modélisée en tant qu'instance et pas en tant que concept (analogie avec *ontologie inadaptée*) ;

S3. *Absence de sens du point de vue social*

L'acteur social est incapable de trouver un sens à l'ontologie ou à certains de ses éléments comme, par exemple, dans le cas où des étiquettes « artificielles » (XYHG45) sont associées aux concepts (analogie avec *ontologie inadaptée*) ;

S4. *Incomplétude du point de vue social*

L'acteur social indique que des composants manquent dans l'ontologie (analogie avec *ontologie incomplète*).

5.3.2.2.2 Situations indésirables sociales

Les situations indésirables sociales sont, pour la plupart, liées aux difficultés rencontrées par l'acteur social quand il essaie de comprendre et d'utiliser l'ontologie. Comme dans le cas

des situations indésirables logiques, il est difficile de dresser une liste exhaustive. Cependant, on peut distinguer 7 situations indésirables parmi les plus communes.

S5. *Absence ou mauvaise qualité des explications textuelles*

Absence d'annotations ou annotations peu nombreuses ou d'une mauvaise qualité. L'absence des annotations complique le travail de ceux qui essaient de comprendre l'ontologie (pas d'analogie avec les problèmes logiques) ;

S6. *Eléments d'ontologie potentiellement équivalents*

L'acteur social peut percevoir comme équivalents deux éléments distincts de l'ontologie, comme, par exemple, des concepts qui ont des étiquettes synonymes (analogie avec *équivalence logique des éléments distincts de l'ontologie*) ;

S7. *Eléments d'ontologie indifférenciables du point de vue social*

Situations où l'acteur social ne peut pas trouver une différence entre deux éléments distincts de l'ontologie. Par exemple, deux concepts qui ont la même position dans la structure hiérarchique de l'ontologie et qui ont des étiquettes synonymes (analogie avec *éléments d'ontologie symétriques, indifférenciables du point de vue logique*) ;

S8. *Eléments d'ontologie avec des étiquettes polysémiques*

Eléments de l'ontologie qui ont des étiquettes polysèmes pouvant être interprétées comme l'union ou l'intersection des « sens » de leurs étiquettes, alors que ces sens peuvent être complètement disjoints (analogie avec *éléments d'ontologie ET* et *OU*) ;

S9. *Ontologie plate* ou absence de structuration

Ontologies qui contiennent très peu d'éléments structurants (notamment des relations taxonomiques) par rapport au nombre d'éléments non structurants (concepts et instances). Cette situation est particulièrement indésirable dans le cas des ontologies de grandes dimensions (analogie avec *complexité élevée des inférences*) ;

S10. *Ontologie formalisée à l'aide d'un formalisme non standardisé*

Quand l'ontologie est formalisée à l'aide de langages logiques ou de théories très spécifiques, non standardisées, les utilisateurs qui ne sont pas familiers avec ces langages doivent faire un effort significatif pour comprendre et utiliser l'ontologie (pas d'analogie avec les problèmes logiques) ;

S11. *Absence de versions certifiées de l'ontologie pour tous les langages standardisés*

Quand il n'existe pas de versions certifiées de l'ontologie dans les langages standardisés il est difficile de l'utiliser dans des contextes qui nécessitent une formalisation standardisée. Une transposition de l'ontologie du formalisme existant vers celui souhaité est *a priori* possible si ces deux formalismes sont standardisés, mais cette transposition peut introduire des erreurs et la validité de l'ontologie obtenue n'est pas certifiée (pas d'analogie avec les problèmes logiques) ;

S12. *Eléments inutiles dans l'ontologie*

Eléments de l'ontologie qui semblent inutiles, dans l'interprétation que l'acteur social lui donne (analogie avec *ontologie non minimale*).

5.4 Positionnement des problèmes de l'état de l'art dans le cadre formel

Grâce aux définitions précises du cadre formel présenté dans la section précédente il est possible de classer dans la même typologie les problèmes décrits dans la littérature. Le tableau 5.1 présente la classification des problèmes mentionnés dans l'état de l'art (Section 5.2.2). Les définitions de certains problèmes étant parfois larges ou ambiguës, ces derniers peuvent être associés à plusieurs classes de notre typologie.

TABLE 5.1: Positionnement des problèmes de l'état de l'art dans le cadre formel

Cadre formel				Problèmes décrits dans l'état de l'art
Aspect logique	**Erreurs**	1	Inconsistance logique	➢ inconsistances : "erreur de partitionnement – instances communes dans une décomposition disjointe" ➢ défaut logique : inconsistance
		2	Ontologie inadaptée	➢ inconsistances : "erreur de partitionnement – concepts communs dans une décomposition disjointe", "inconsistance sémantique" ➢ anti-patrons logiques : "OnlynessIsLoneliness", "UniversalExistence", "AndIsOR", "EquivalenceIsDifference" ➢ embûches : P5 (relation inverse erronée, *WI*), P14 (utilisation abusive de "allValuesFrom", *MD*), P15 (utilisation abusive de "not some"/"some not", *WI*), P18 (définition de domaine et / ou de codomaine trop spécifiques, *WI*), P19 (confusion entre ∩ et ∪, *WI*)
		3	Ontologie incomplète	➢ incompletude : "absence de relations taxonomiques", "absence d'axiomes de disjoinction / exhaustivité" ➢ embûches : P3 (utilisation de "is a" à la place de "subclass-of", *MD*), P9 (absence d'informations basiques, *RC & RWM*), P10 (absence de la disjonction, *RWM*), P11 (absence du domaine / codomaine dans la déclaration des propriétés, *NI & OU*), P12 (non déclaration de l'équivalence entre propriétés, *NI & OU*), P13 (non déclaration du fait que deux relations sont l'une l'inverse de l'autre, *NI & OU*), P16 (utilisation erronée des classes primitives et définies, *NI*)
		4	Inférences incorrectes	
		5	Inférences incomplètes	➢ embûches : P3 (utilisation de "is a" à la place de "subclass-of", *MD*), P24 (définitions recursives, *MD*)
	Situations indésirables	6	Equivalence logique des éléments distincts de l'ontologie	➢ inconsistance : "circularité" ➢ embûche : P6 (cycles dans la hiérarchie, *WI*) ➢ anti-patron cognitif : "SynonymeOfEquivalence"
		7	Eléments d'ontologie indifférenciables du point de vue logique	➢ embûche : P4 (éléments d'ontologie non connectés, *RC*) ➢ anomalies de conception : "concepts isolés" et "chaînes d'héritage"
		8	Eléments d'ontologie OU	➢ embûche : P7 (combiner des concepts pour former une classe, *MD & OU*)
		9	Eléments d'ontologie ET	➢ embûche : P7 (combiner des concepts pour former une classe, *MD & OU*)
		10	Insatisfiabilité	➢ inconsistance : "erreur de partitionnement – concept commun dans une décomposition disjointe" ➢ anti-patrons logiques : "OnlynessIsLoneliness", "UniversalExistence", "AndIsOR", "EquivalenceIsDifference" ➢ défaut logique : concept insatisfiable
		11	Complexité élevée des inférences	➢ anti-patrons conseils : les patrons décrivant des situations qui réduisent les performances des moteurs d'inférences
		12	Ontologie non minimale	➢ redondance : "redondance des relations taxonomiques" ➢ embûches : P3 (utilisation de "is a" à la place de "subclass-of", *MD*), P7 (combiner des concepts pour former une classe, *MD & OU*), P21 (classe par défaut, *MD*) ➢ anti-patrons cognitifs : "SomeMeansAtLeastOne" ➢ conseils : "Domain&CardinalityConstraints", "MinIsZero"

TABLE 5.1: Positionnement des problèmes de l'état de l'art dans le cadre formel

Cadre formel				Problèmes décrits dans l'état de l'art
Aspect social	**Erreurs**	1	Contradiction sociale	➢ inconsistance : "inconsistance sémantique" ➢ anti-patrons logiques : "AndIsOR" ➢ embûches : P1 (éléments polysémiques, *MD*), P5 (relation inverse erronée, *WI*), P14 (utilisation abusive de "allValuesFrom", *MD*), P15 (utilisation abusive de "not some"/"some not", *WI*), P19 (confusion entre ∩ et ∪, *WI*)
		2	Perception d'erreurs de conception	➢ embûches : P17 (hyperspécialisation de la hierarchie, *MD*), P18 (définition de domaine et / ou de codomaine trop spécifiques, *WI*), P23 (utilisation abusive des éléménts d'ontologie, *MD*) ➢ anti-patrons cognitifs : "SumOfSome" ➢ anomalies de conception : "disjonctions lointaines"
		3	Absence de sens du point de vue social	
		4	Incomplétude du point de vue social	➢ embûches : P12 (non déclaration de l'équivalence entre propriétés, *NI & OU*), P16 (utilisation erronée des classes primitives et définies, *NI*)
	Situations indésirables	5	Absence ou mauvaise qualité des explications textuelles	➢ embûches : P8 (absence des annotations, *OC & OU*)
		6	Eléments d'ontologie potentiellement équivalents	➢ embûches : P2 (synonymes comme classes, *MD & OU*)
		7	Eléments d'ontologie indifférenciables du point de vue social	
		8	Eléments d'ontologie avec des étiquettes polysémiques	➢ embûches : P1 (éléments polysémiques, *MD & OU*)
		9	Ontologie plate	
		10	Ontologie formalisée à l'aide d'un formalisme non standardisé	➢ embûches : P20 (interchanger les étiquettes avec les annotations, *OU*), P22 (utilisation de plusieurs critères de nommage dans l'ontologie, *OC*) ➢ conseils : "GroupAxioms", "DisjointnessOfComplement" et "Domain&CardinalityConstraints" ➢ anomalies de conception : "groupements répétitifs de propriétés"
		11	Absence de versions certifiées de l'ontologie	
		12	Eléments inutiles dans l'ontologie	➢ embûches : P21 (classe par défaut, *MD*)

Le Tableau 5.1 montre tout d'abord que notre typologie contient des classes de problèmes qui, à notre connaissance, n'ont pas été décrites dans la littérature [2]. Ces problèmes sont : *Absence de version certifiée de l'ontologie*, *Ontologie plate*, *Eléments d'ontologie indifférenciables*, *Absence de sens du point de vue social* et *Inférences incorrectes*. Dans la typologie,

2. Ces problèmes n'ont pas été décrits dans la littérature concernant l'évaluation et la qualité des ontologies, mais ils ont été décrits dans autres domaines.

il y a aussi des classes de problèmes pour lesquelles on trouve seulement des descriptions de cas très particuliers dans la littérature, et non une description de la classe de problèmes. C'est le cas de la classe *Ontologie formalisée à l'aide d'un formalisme non standard.*

Une analyse approfondie du Tableau 5.1 montre que les *anti-patrons logiques* sont des problèmes concernant, sans surprise, la dimension logique des ontologies et plus particulièrement l'erreur *ontologie inadaptée* et la situation indésirable d'*insatisfiabilité*. Les *anti-patrons cognitifs* couvrent des situations indésirables concernant la dimension logique. Les *conseils* concernent seulement des situations indésirables (liées tant à la dimension logique qu'à celle sociale).

La classe des problèmes que Gomez-Perez et al. ont nommé *inconsistances* est beaucoup plus large que l'*inconsistance logique* définie dans notre typologie. Un seul problème correspond à notre définition : l'existence d'instances communes dans une décomposition disjointe. Nous avons classé les autres problèmes comme suit : l'existence de concepts communs dans une décomposition disjointe est à la fois un problème d'*ontologie inadaptée* et un problème d'*insatisfiabilité* ; l'*inconsistance sémantique* est un problème d'*ontologie inadaptée* et une *contradiction sociale* ; la *circularité* est un problème d'*équivalence logique d'éléments distincts de l'ontologie*. Les problèmes que Gomez-Perez et al. ont nommé *incomplétudes* correspondent tous à la classe *ontologie incomplète* de notre typologie. Les *redondances* identifiées par Gomez-Perez et al. sont des problèmes d'*ontologie non minimale*.

Aucune des *anomalies de conception* présentées par Baumeister et al. ne correspond à une erreur logique.

Concernant les *embûches*, l'aspect le plus intéressant concerne les *inférences incomplètes*. Plus précisément, le fait d'introduire dans l'ontologie des relations *ad-hoc* comme *est-un, instance-de, etc.*, pour remplacer les relations « standard » (*subsumption, member-of, etc.*) ne doit pas être considéré comme un cas d'*ontologie incomplète* mais comme un cas *d'inférences incomplètes* parce qu'il est très peu probable que les relations *ad-hoc* soient définies exactement de la même manière que les relations « standard » bien que, *a priori*, cela soit possible. Lorsque les relations *ad-hoc* n'ont pas de définitions identiques aux relations « standard », un système d'inférence qui utilise l'ontologie ne peut pas obtenir tous les résultats qu'il aurait obtenus si l'ontologie avait été construite en utilisant les relations « standards ».

Toutes les *embûches* classées *conception réaliste* sont des erreurs logiques classées *ontologie incomplète*. Toutes les *embûches* qui font partie de la classe *clarté de l'ontologie* sont des situations indésirables sociales. Toutes les *embûches* classées *satisfaction des spécifications* sont des problèmes logiques dans notre typologie. Les *embûches* classées *inférences impossibles* sont des erreurs d'incomplétude (soit du point de vue logique, soit du point de vue social). La plupart (6/9 et 4/5) des *embûches* classées *décisions de conception* et *inférences fausses* correspondent à des *erreurs* dans notre typologie. Enfin, les *embûches* classées comme *compréhensibilité de l'ontologie* correspondent à dix classes différentes de problèmes de notre typologie (erreurs et situations indésirables concernant des aspects sociaux et logiques).

La plupart (16/20) des *embûches* associées à la dimension structurelle de l'ontologie correspondent à des *erreurs* dans notre typologie. Toutes (2/2) les *embûches* associées à la dimension fonctionnelle de l'ontologie correspondent à des problèmes logiques dans notre typologie.

5.5 Conclusion

La première partie de ce chapitre a été consacrée à un état de l'art sur les problèmes de qualité des ontologies. Cet état de l'art a mis en évidence une variabilité des problèmes rencontrés qui nous a conduit à introduire un nouveau cadre formel pour la description des problèmes, ainsi qu'une structuration sous la forme d'une typologie organisée selon deux dimensions complémentaires. Sans prétendre à l'exhaustivité, notre typologie a pour ambition d'offrir une vue unitaire et cohérente sur les problèmes de qualité. Et elle est une première étape vers une démarche de vérification systématique.

Dans le chapitre suivant, nous confrontons cette typologie aux différentes tâches d'un processus de construction, en s'appuyant sur Methontology. L'objectif est de mieux saisir la cause des différents problèmes et d'en estimer une certaine probabilité d'apparition (en restant ici très modeste sur la représentativité statistique de nos expérimentations).

6

Identification des problèmes de qualité pour des ontologies construites automatiquement

Sommaire

6.1 Introduction

La typologie des problèmes de qualité proposée dans le chapitre précédent ouvre la voie vers une vérification systématique des ontologies pour la détection d'éventuels problèmes de qualité. A notre connaissance, aucune méthode ou guide de « bonnes pratiques » n'ont été publiés dans la littérature pour cette problématique.

Dans ce chapitre, nous nous intéressons aux éléments qui peuvent servir de support à l'élaboration d'un guide de « bonnes pratiques » adapté au cas particulier important des ontologies construites automatiquement à partir de textes. En particulier, il s'agit d'identifier des problèmes dans l'ontologie, de comprendre les processus qui ont conduit à l'apparition de chacun des problèmes, d'analyser les liens entre les différents problèmes et d'estimer leur probabilité d'apparition.

L'analyse présentée dans les chapitres 3 et 4 nous a permis de mieux cerner les avantages mais aussi les limites des processus de construction automatique et de leurs implémentations logicielles. Dans ce chapitre, nous prolongeons l'analyse en faisant l'hypothèse qu'il est possible de déterminer pour chaque tâche de l'activité de conceptualisation les risques majeurs de résultats sous-optimaux. Cela nous conduit à associer aux différentes tâches les problèmes identifiés au chapitre précédent.

Ce chapitre est structuré en deux parties. Dans la première partie nous analysons, pour chaque étape de l'activité de conceptualisation d'une ontologie, la manière dont les implémentations imparfaites des différentes tâches impliquées dans une construction automatique conduisent à des résultats intermédiaires sous-optimaux qui s'associent à des problèmes de qualité dans l'ontologie construite. La deuxième partie est expérimentale.

Nous identifions les problèmes de qualité associés à deux ontologies construites automatiquement à partir de corpus différents avec l'outil Text2Onto dont le choix a été guidé par les résultats comparatifs du chapitre 4.

6.2 Compromis d'implémentation associés aux problèmes de qualité

Comme nous l'avons mentionné dans la section 3.3, les approches proposées dans la littérature ont automatisé le processus de construction d'une ontologie au prix de compromis dans l'implémentation et l'exécution de certaines tâches.

Ces compromis conduisent à des résultats intermédiaires sous-optimaux qui peuvent dégrader la qualité de l'ontologie obtenue en regard d'une construction manuelle.

6.2.1 Retour sur l'implémentation des tâches du processus de construction

Visant l'absence d'intervention humaine, la plupart des approches n'implémentent en fait qu'un sous-ensemble restreint des tâches de l'activité de conceptualisation et ignorent les autres activités du processus global de construction. Elles se limitent à l'extraction des concepts, des instances et des relations et à leur combinaison pour obtenir une ontologie. Elles ignorent bien souvent la description détaillée de ces éléments à l'aide d'attributs, d'axiomes et de règles formelles. Ainsi, elles n'implémentent pas les trois dernières tâches du processus

de Conceptualisation décrit par Methontology, et ne proposent qu'une opérationnalisation partielle des quatre premières.

Plus précisement, pour la première tâche (la construction du glossaire de termes) la plupart des approches ignorent la sous-tache d'identification de définitions en langage naturel pour les termes qui composent le glossaire. Pour la deuxième tâche (la construction de taxonomies de concepts), les sous-tâches les plus souvent ignorées sont l'agrégation de termes synonymes pour la construction des concepts et l'association d'une étiquette non ambiguë à chaque concept. Pour la troisième tâche (la construction de diagrammes pour les relations binaires ad-hoc), le regroupement des termes du glossaire qui correspondent à une même relation est rarement effectué. Pour la quatrième tâche (la construction du dictionnaire de concepts), les approches automatiques ignorent généralement l'identification des attributs des concepts.

Nous avons vu que même les techniques les plus performantes d'extraction automatique de composants de base pouvaient conduire à une précision et un rappel dégradé. Par conséquent, il est très probable que les différents résultats intermédiaires obtenus automatiquement soient incomplets ou qu'ils contiennent des éléments incorrects ou inutiles. Comme dans le cadre d'une construction automatique, ces résultats intermédiaires ne sont pas validés ni corrigés par un expert, ces sous-optimalités peuvent se propager et aboutir à des problèmes de qualité.

6.2.2 Origine des problèmes

Dans le tableau 6.1 nous résumons, pour chacune des tâches du processus de Conceptualisation de Methontology, les principaux risques d'obtention de résultats sous-optimaux et précisons, pour chaque risque, les problèmes de qualité qui sont générés (conséquences directe) ou dont l'apparition est favorisée (conséquences possibles). Le reste de cette section détaille le contenu du tableau.

Tâche 1 (Construction du glossaire de termes). Trois risques de résultats intermédiaires (les listes de termes associés à des instances, concepts, relations, etc.) sous-optimaux sont identifiés : (1) si des termes requis ne font pas partie du glossaire utilisé pour construire l'ontologie, alors celle-ci sera incomplète (des problèmes L3 et S4) ; (2) si des termes inutiles sont inclus dans le glossaire, alors ils peuvent générer des artéfacts inutiles (des problèmes S12) qui conduisent à une ontologie non minimale (problèmes L12) et il est aussi possible que ces artéfacts inutiles soient incompréhensibles (problèmes S3) ; (3) si le glossaire contient des associations incorrectes des termes avec des concepts alors qu'ils auraient du être associés à des instances, ou vice-versa, alors l'ontologie sera affectée par des erreurs de conception (problèmes S2).

Tâche 2 (Construction de taxonomies de concepts). Les résultats sous-optimaux sont dûs à des compromis dans la formation de concepts à partir des termes et à des imprécisions dans la découverte des relations taxonomiques. Methontology précise qu'avant d'identifier les relations taxonomiques il faut regrouper les termes qui font référence à un même concept et attribuer une étiquette à chaque concept. Pour automatiser le processus la plupart des approches (Text2Onto, OntoLearn, Sprat) ont renoncé à cette agrégation et considèrent que chaque terme correspond à un concept différent ; ce qui conduit à l'apparition de concepts indifférenciables et potentiellement équivalents (L7 et S6). Quand les approches regroupent

TABLE 6.1: Problèmes de qualité qui peuvent apparaître suite à des compromis dans l'implémentation des tâches de Methontology.

Tâches de Methontology		**Sous-optimalités des résultats de chaque tâche**	**Problèmes de qualité**	
			Conséquences directes	**Conséquences possibles**
1	Construction du glossaire de termes	absence de termes requis	L3, S4	–
		termes inutiles (par exemple, qui ne concernent pas le domaine)	L12, S12	S3
		association incorrecte de termes (par exemple, des termes associés à des concepts au lieu d'instances, ou vice-versa)	S2	–
2	Construction de taxonomies de concepts	agrégation incorrecte de termes sans aucun rapport	L12, S12	L3, S4
		absence d'agrégation des termes synonymes pour former un seul concept	L7, S6	–
		association à un concept d'une étiquette polysémique dans le domaine de l'ontologie	S8	L2, L9, L10, S2
		absence d'au moins une relation taxonomique requise	L3, S4	L7, S7
		absence de la plupart des relations taxonomiques requises	S9	–
		relations taxonomiques qui peuvent être inférées	L12, S12	L11
		relations taxonomiques incorrectes	L2, S1	L6
3	Construction de diagrammes pour les relations binaires ad-hoc	absence de relations ad-hoc requises	L3, S4	L7, S7
		relations ad-hoc non requises	L12, S12	L11
		relations ad-hoc incorrectes qui contredisent les modèles intentionnels (resp. les connaissances de l'utilisateur)	L2, S1	–
4	Construction d'un dictionnaire de concepts	absence de définitions en langage naturel pour concepts	S5	S3
		associations incorrecte d'instances avec des concepts	L2, S1	–
		absence d'attributs de concept nécessaires	L3, S4	L7, S7
		attributs de concepts incorrects	L2, S1	–
5	Description des relations binaires ad-hoc et des attributs	absence de description de relations ad-hoc	L3, S4	L7, S7
		description de relation ad-hoc contradisant les modèles intentionnels (resp. les connaissances de l'utilisateur)	L2, S1	–
		absence de description d'attribut	L3, S4	L7, S7
		description incorrecte d'attribut	L2, S1	–
6	Description des règles et des axiomes formels	absence d'axiomes nécessaires	L3, S4	L7, S7
		axiomes inutiles	L12, S12	L11
		axiomes en contradiction avec les modèles intentionnels (resp. les connaissances de l'utilisateur)	L2, S1	L1, L10
7	Description des instances	absence de description d'instance	L3, S4	L5, L7, S7
		description incorrecte d'instance	L2, S1	–

des termes pour former les concepts (comme ASIUM), les techniques utilisées ne garantissent pas la validité des regroupements. Une agrégation incorrecte conduit à l'apparition d'un concept mal défini et qui n'a pas sa place dans l'ontologie (L12, S12). Si, de plus, les termes maladroitement associés correspondent à des concepts indépendants qui doivent faire partie de l'ontologie, alors cette agrégation incorrecte génère en plus des problèmes d'incomplétude (L3, S4). Le plus souvent, dans les approches automatiques, les étiquettes associées aux concepts sont les termes à partir desquels les concepts ont été identifiés. Comme certaines approches (Text2Onto, Sprat) ne précisent pas le sens des termes utilisés comme étiquette, on retrouve dans l'ontologie construite des concepts ayant une étiquette polysémique dans le domaine de l'ontologie (S8). La présence d'un concept ayant une étiquette polysémique peut indiquer des erreurs de conception (S2) et suggérer que l'ontologie est inadaptée (L2) : un concept avec une étiquette polysémique qui fait partie de l'ontologie et des modèles intentionnels peut avoir une signification différente dans l'ontologie de celle des modèles intentionnels. En plus, une étiquette polysémique peut indiquer qu'il s'agit d'un concept défini comme la conjonction de deux significations différentes (L9) ou qui ne peut pas accepter d'instances (L10).

Les imprécisions dans la découverte de relations taxonomiques affectent aussi les résultats intermédiaires. Ainsi, si au moins une relation taxonomique requise n'est pas identifiée (4) alors l'ontologie sera incomplète (L3, S4) et il est possible que certains concepts restent indifférenciables (L7, S7). Si un nombre significatif de relations taxonomiques requises ne sont pas identifiées (5), alors l'ontologie sera plate (S9). Si des relations taxonomiques redondantes sont retenues (6) alors l'ontologie ne sera pas minimale (L12, S12) et la réalisation d'inférences peut être rendue plus complexe. Si des relations taxonomiques incorrectes sont retenues (7) alors l'ontologie va être inadaptée (L2), va contredire les connaissances des utilisateurs (S1) et l'apparition de fausses équivalences entre concepts est rendue possible (L6).

Tâche 3 (Construction de diagrammes pour les relations binaires ad-hoc). Les résultats intermédiaires sont sous-optimaux à cause des limitations des techniques d'extraction de relations. Si des relations requises ne sont pas identifiées alors l'ontologie sera incomplète (L3, S4) et les concepts qui auraient dû être impliqués dans ces relations vont rester indifférenciables (L7, S7). Si des relations non requises sont retenues alors l'ontologie ne sera pas minimale (L12, S12) et, éventuellement, la réalisation d'inférences sera plus complexe (L11). Si des relations qui contredisent les modèles intentionnels (respectivement les connaissances de l'utilisateur) sont retenues, alors l'ontologie sera inadaptée (L2) et l'utilisateur va percevoir des contradictions (S1).

Tâche 4 (Construction du dictionnaire de concepts). Methontology précise qu'il faut décrire chaque concept en lui associant ses relations, ses attributs et ses instances. En parallèle, dans l'activité de Documentation, il faut associer à chaque concept une définition en langage naturel. Sans ces définitions les concepts de l'ontologie construite automatiquement n'auront pas les annotations (S5) et certains concepts peuvent être ininterprétables pour l'utilisateur (S3). Si des attributs, des instances ou des relations [1] requises ne sont pas associés aux concepts alors l'ontologie sera incomplète (L3, S4) et il est possible que certains concepts

1. Pour éviter les répétitions, dans le tableau 6.1 nous avons retenu comme risque seulement l'absence d'attributs, parce que l'absence de relations et d'instances requises ont été déjà couvertes par les risque énumérées pour les tâches précédentes.

restent indifférenciables (S7, L7). Si les associations d'attributs et d'instances aux concepts sont incorrectes alors l'ontologie sera inadaptée (L2) et les connaissances des utilisateurs vont être contredites (S1).

Tâche 5 (Description des relations et des attributs). Les résultats intermédiaires sont sous-optimaux si une partie de ces descriptions est manquante ; ce qui va conduire à une ontologie incomplète (L3, S4) et qui peut contenir des concepts indifférenciables (S7, L7). Les descriptions peuvent en plus contredire les modèles intentionnels ou les connaissances de l'utilisateur (L2, S1).

Tâche 6 (Description des règles et des axiomes formels). Les résultats sont sous-optimaux s'ils ne contiennent pas tous les axiomes requis ce qui conduit à une ontologie incomplète (L3, S4) pouvant contenir des concepts indifférenciables (L7, S7). Les résultats sont aussi sous-optimaux s'ils contiennent des axiomes non requis ce qui conduit à une ontologie non minimale (L12, S12) et complexifie les inférences (L11) ; des axiomes non requis peuvent contredire les modèles intentionnels ou les connaissances de l'utilisateur (L2, S1) et peuvent conduire à des inconsistances dans l'ontologie (L1) ou à des concepts insatisfiables (L10).

Tâche 7 (Description des instances). Nous rappelons que pour cette dernière tâche il faut préciser, pour chaque instance, les concepts qui lui sont associés et les valeurs de ses attributs. En l'absence de cette description l'ontologie va être incomplète (L3, S4), certaines instances peuvent rester indifférenciables (L7, S7) et il est possible que certaines inférences ne soient pas réalisables (L5). Si la description est incorrecte alors l'ontologie sera inadaptée et contredira les connaissances de ses utilisateurs (L2, S1).

6.2.3 Discussion

La synthèse présentée dans le tableau 6.1 montre que certains problèmes de qualité peuvent apparaître suite à l'exécution sous-optimale de chacune des tâches. C'est notamment le cas des erreurs concernant l'incomplétude de l'ontologie (L3, S4) qui peuvent prendre leur origine dans les résultats sous-optimaux de chacune des 7 tâches de Methontology. On retrouve une situation similaire pour les erreurs concernant la conformité et l'adaptation de l'ontologie par rapport aux spécifications (L2, S1) qui peuvent apparaître dans 6 tâches, les situations où l'ontologie contient des éléments indifférenciables (L7, S7) – 5 tâches, et les problèmes de minimalité (L12, S12) – 4 tâches.

Certains autres problèmes (L1, S8, S9) sont associés à une seule tâche et à un type particulier de sous-optimalité des résultats intermédiaires.

L'identification des causes possibles dans le processus de construction de chaque type de problème devrait aider à faire de la prédiction. Ainsi, sachant que les approches ignorent les trois dernières tâches du processus de Conceptualisation, il est très probable d'aboutir à une ontologie incomplète ne contenant aucun problème d'inconsistance logique. De plus, à cause des faibles valeurs de la précision et du rappel de certaines approches proposées dans la littérature, il est très probable que les ontologies construites automatiquement contiennent des problèmes de type L2, S1, L12 et S12.

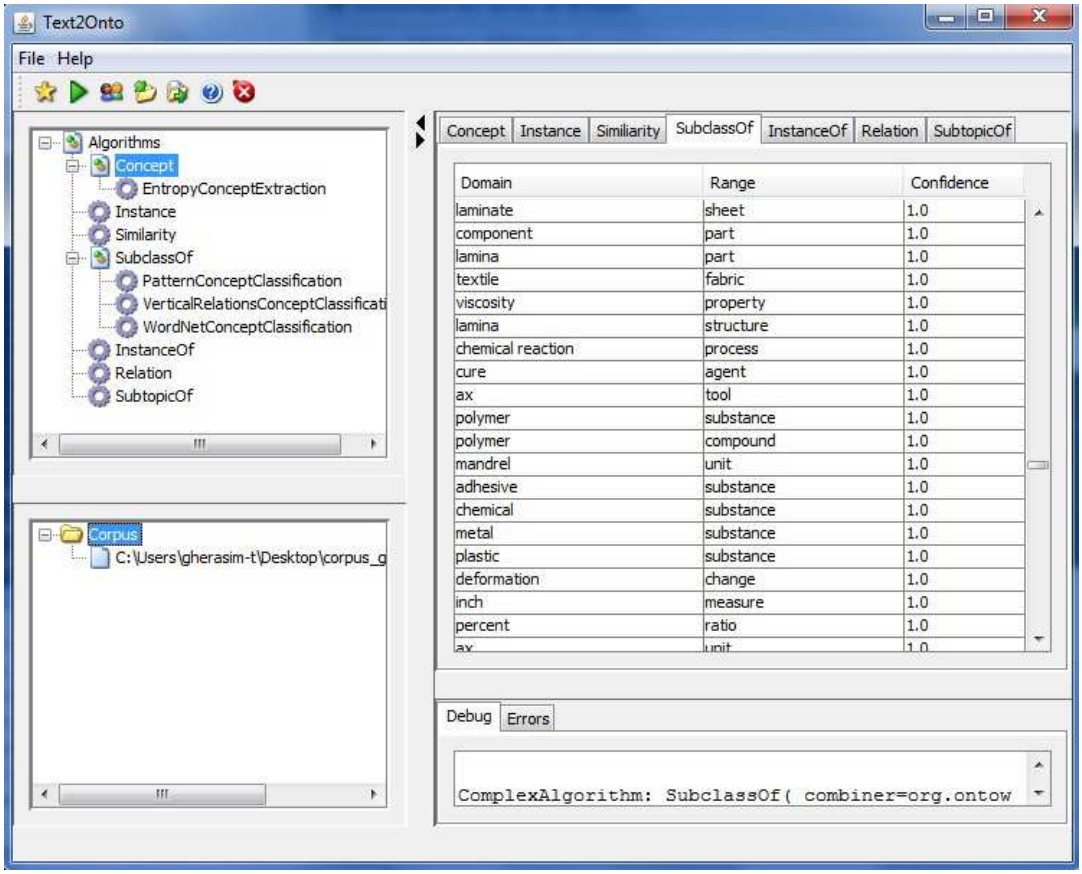

FIGURE 6.1: La configuration de Text2Onto utilisée pour la construction de O_1 et O_2.

6.3 Retours d'expériences du cadre applicatif

Pour vérifier la pertinence de l'analyse présentée dans la section précédente, nous avons identifié expérimentalement les problèmes présents dans deux ontologies construites automatiquement par Text2Onto.

6.3.1 Cadre expérimental

L'analyse approfondie des résultats de Text2Onto présentée dans la Section 4.3.3 montre que ses performances pour l'extraction de concepts et de relations taxonomiques sont nettement supérieures à ses performances pour l'extraction d'instances et de relations non taxonomiques. Par conséquent, pour la construction des deux ontologies que nous analysons ici, nous avons seulement utilisé les modules d'extraction de concepts et de relations taxonomiques de Text2Onto en activant tous les algorithmes proposés (figure 6.1).

La première ontologie (O_1) a été construite à partir du même corpus que celui utilisé lors de l'expérimentation présentée dans le chapitre 4. Rappelons que ce corpus contient 4500 mots et est une version simplifiée d'un article scientifique concernant les approches de construction automatique des ontologies. L'ontologie construite contient 444 concepts et

362 relations taxonomiques. Certains concepts « centraux » sont : *« ontology », « method », « resource », « domain », « tool », « repository », « ontology component », « domain », « expert », « ontology building step », « ontology building process ».*

La deuxième ontologie (O_2) a été construite à partir d'un document de 9500 mots qui concerne le domaine des matériaux composites et la fabrication d'objets en matériaux composites. Plus précisément, le document utilisé est un glossaire contenant les définitions de 376 termes spécifiques au domaine. L'ontologie construite contient 965 concepts et 408 relations taxonomiques. Certains concepts « centraux » sont : *« technique », « step », « compound », « fiber », « resin », « polymerization », « laminate », « substance », « form ».*

Tout en étant bien conscients que de nombreux auteurs préconisent le recours à des *scenarii* d'utilisation pour la conception d'ontologies de domaine [1], nous n'avons considéré ici aucun modèle intentionnel ni d'ensemble d'exemples et de contre-exemples pouvant les remplacer afin d'avoir des conditions expérimentales identiques pour l'analyse des deux ontologies. Pour la première ontologie, notre analyse s'est appuyée sur notre expertise du domaine. Pour la deuxième, notre connaissance du domaine s'est affinée au cours du projet ISTA3.

6.3.2 Problèmes identifiés

Le tableau 6.2 résume les résultats de notre analyse et présente les types et, lorsque c'est possible, la fréquence des problèmes que nous avons identifiés dans les deux ontologies construites automatiquement. L'espace limité du tableau ne permettant pas de préciser notre démarche, nous la décrivons dans le reste de cette section.

TABLE 6.2: Problèmes identifiés dans les deux ontologies construites automatiquement.

Types de problèmes	**Problèmes détectées dans les ontologies**	
	Ontologie O_1	**Ontologie O_2**
L1	Aucune inconsistance logique	Aucune inconsistance logique
L2	*Impossible à détecter*	*Impossible à détecter*
L3	*Impossible à détecter*	*Impossible à détecter*
L4	*Impossible à détecter*	*Impossible à détecter*
L5	*Impossible à détecter*	*Impossible à détecter*
L6	26 groupes de concepts équivalents	3 groupes de concepts équivalents
L7	28 groupes de concepts mutuellement indifférenciables	74 groupes de concepts mutuellement indifférenciables
L8	Aucun "élément d'ontologie OU"	Aucun "élément d'ontologie OU"
L9	Aucun "élément d'ontologie ET"	Aucun "élément d'ontologie ET"
L10	Aucun concept insatisfiable	Aucun concept insatisfiable
L11	Aucune situation susceptible de complexifier les inférences	Aucune situation susceptible de complexifier les inférences
L12	164 relations taxonomiques redondantes	1 relation taxonomique redondante
S1	147 relations taxonomiques qui contredisent les connaissances de l'évaluateur	15 relations taxonomiques qui contredisent les connaissances de l'évaluateur
S2	2 instances ont été identifiées comme étant des concepts	5 instances ont été identifiées comme étant des concepts
S3	13 concepts avec des étiquettes sans aucun sens	21 concepts avec des étiquettes sans aucun sens
S4	156 concepts non connectés	389 concepts non connectés
S5	L'ontologie ne contient aucune annotation	L'ontologie ne contient aucune annotation
S6	12 paires de concepts ayant des étiquettes synonymes qui peuvent être interprétées comme étant équivalentes	6 paires de concepts ayant des étiquettes synonymes qui peuvent être interprétées comme étant équivalentes
S7	Aucune paire de concepts indifférenciables	Aucune paire de concepts indifférenciables

TABLE 6.2: Problèmes identifiés dans les deux ontologies construites automatiquement.

Types de problèmes	**Problèmes détectées dans les ontologies**	
	Ontologie O_1	**Ontologie O_2**
S8	64 concepts ayant des étiquettes polysémiques	69 concepts ayant des étiquettes polysémiques
S9	Ontologie plate, affectée par un manque de structuration (profondeur moyenne des feuilles de 2.02)	Ontologie plate, affectée par un manque de structuration (profondeur moyenne des feuilles de 1.99)
S10	Aucun problème (l'ontologie est formalisée à l'aide du langage OWL)	Aucun problème (l'ontologie est formalisée à l'aide du langage OWL)
S11	L'ontologie n'est pas certifiée	L'ontologie n'est pas certifiée
S12	138 concepts non requis	28 concepts non requis

L1, L10. Pour détecter les inconsistances logiques et la présence de concepts insatisfiables, nous avons utilisé le moteur d'inférences PELLET intégré à l'éditeur d'ontologies Protégé [2].

L2, L3, L4, L5 De par notre protocole expérimental (aucun cas d'utilisation, modèle intentionnel ni ensemble d'exemples et contre-exemples), il nous est impossible de déterminer de manière formelle si les deux ontologies sont affectées par ces types de problèmes. Dans la Section 5.3.2.2, en introduisant les problèmes sociaux, nous avons souligné différentes analogies entre certains types problèmes sociaux et certains types de problèmes logiques, notamment entre {S1, S2, S3} et {L2} et entre {S4} et {L3} (la présence d'un problème du type S4 dans une ontologie peut indiquer la présence d'un problème du type L3). Comme nous avons identifié dans les deux ontologies des problèmes S1, S2, S3 et S4, nous pensons qu'il est très probable d'identifier des problèmes L2 et L3 dans ces ontologies si des cas d'utilisation et des modèles intentionnels étaient disponibles.

L6. Pour identifier les groupes de concepts logiquement equivalents, nous avons utilisé le moteur d'inférences PELLET en le laissant réaliser toutes les inférences possibles à partir de chaque ontologie construite par Text2Onto.

L7. Comme les deux ontologies ne contiennent que des concepts et des relations taxonomiques, deux concepts c_1 et c_2 appartenant à l'une de ces ontologies sont différenciables seulement s'il existe une différence entre les ensembles de relations taxonomiques L_1^p (les relations taxonomiques dans lesquelles c_1 a le rôle de père) et L_1^f (les relations taxonomiques dans lesquelles c_1 a le rôle de fils) associées au concept c_1 et les ensembles de relations taxonomiques L_2^p et L_2^f associé à c_2. Tous les concepts qui partagent les mêmes ensembles de relations taxonomiques L^p et L^f sont mutuellement indifférenciables. Nous avons identifié 28 groupes de concepts mutuellement indifférenciables dans O_1 et 74 groupes dans O_2.

2. http://protege.stanford.edu

L8, L9. Par définition, les concepts de type "OU" et "ET" sont des concepts définis à l'aide d'axiomes de conjonction et de disjonction entre concepts. Comme O_1 et O_2 ne contiennent aucun concept défini mais seulement des concepts primitifs, elles ne contiennent pas non plus des concepts de type "OU" ou "ET".

L11. O_1 et O_2 ne contiennent pas des situations correspondant aux patrons connus dans la littérature pour rendre les inférences complexes.

L12. Les relations taxonomiques sont les seuls éléments qui peuvent être redondants dans O_1 et O_2. Pour les identifier nous avons vérifié, pour chaque relation (c_1, c_2), dans le graphe correspondant à l'ontologie l'existence d'un chemin alternatif entre c_1 et c_2 après avoir éliminé (c_1, c_2).

S3. Nous avons considéré que l'étiquette d'un concept n'a pas de sens quand elle n'est pas un terme dans le corpus à partir duquel l'ontologie a été construite (par exemple, "term cannot", "purpose ontology", etc.)

S4. L'incomplétude de O_1 et O_2 peut être considérée comme une évidence, puisqu'elles ne contiennent que des concepts et des relations taxonomiques. Néanmoins, les ensembles de concepts et de relations taxonomiques extraites sont également incomplets. La présence de concepts non connectés est facilement identifiable.

S9. Le manque de structuration des deux ontologies est souligné par la proportion des concepts non connectés (35% dans O_1 et respectivement 40% dans O_2) et par la faible profondeur moyenne des feuilles (2.02 dans O_1 et respectivement 1.99 dans O_2).

S11. Comme les cas d'utilisation ne sont pas connus, on ne peut pas savoir si les ontologies sont adaptées. Néanmoins, on sait que toute ontologie construite automatiquement est, de par sa construction, non certifiée.

S12. Nous avons considéré comme concepts non requis seulement ceux pour lesquels cela est une évidence (par exemple, "anything, "cannot", etc.).

6.3.3 Discussion

Parmi les cinq types de problèmes quantifiables, les quatre les plus fréquents sont des erreurs affectant l'aspect social et trois sont communs à O_1 et O_2 : S4 (incomplétude du point de vue social), S12 (éléments inutiles dans l'ontologie) et S8 (étiquettes polysémiques). Une sixième erreur peut être ajoutée à cette liste – S9 (ontologie plate).

Deux des cinq types de problèmes les plus fréquents sont différents pour O_1 et O_2 : S1 (des relations taxonomiques considérées comme sémantiquement inconsistantes par l'expert) et L12 (des relations taxonomiques redondantes) pour O_1, et L7 (éléments d'ontologie indifférenciables du point de vue logique) et S3 (étiquettes sans sens) pour O_2.

Les différences les plus importantes entre O_1 et O_2 sont les problèmes de type L12 (O_1 : 164 ; O_2 : 1), S1 (O_1 : 147 ; O_2 : 15) et L7 (O_1 : 28 ; O_2 : 74).

La différence entre le nombre de relations redondantes (L12) identifiées dans O_1 par rapport au nombre de relations identifiées dans O_2 est expliquée en partie par la différence des densités des deux graphes : le nombre moyen de relations par concept est 0.8 pour O_1 et de 0.4 pour O_2. De ce fait, la probabilité d'avoir des relations "incorrectes sémantiquement" est structurellement plus grande dans O_1 que dans O_2 ; ce qui explique aussi la différence entre le nombre de situations S1 identifiées. La corrélation entre le nombre de problèmes

L12 et le nombre de problèmes S1 n'est pas surprenante étant donné que les deux ontologies ne contiennent pas d'instances ni de relations non taxonomiques.

Au-delà de ces différences, l'agrégation des différents problèmes concernant la structure des ontologies indique que les ontologies O_1 et O_2 partagent les mêmes propriétés générales : leur structure est hétérogène, combinant un large nombre de concepts non connectés avec des noyaux denses.

6.4 Conclusion

Dans la première partie de ce chapitre nous avons analysé, pour chaque étape du processus de construction d'une ontologie, les conséquences, en terme de problèmes de qualité, de la sous-optimalité des différentes tâches d'un processus de construction. Cette analyse a confirmé l'intuition issue de notre expérience : les principaux problèmes concernent l'incomplétude de l'ontologie (L3, S4) et sa conformité avec ses spécifications (L2, S1, L12, S12).

Dans la deuxième partie, nous avons adopté une démarche expérimentale en analysant les problèmes rencontrés pour deux ontologies – très différentes – construites automatiquement avec Text2Onto. Cette expérience a montré que, pour les ontologies testées, parmi les types de problèmes les plus fréquents on retrouve, en plus des problèmes concernant l'incomplétude et la conformité de l'ontologie, des problèmes du type S8 (étiquettes polysémiques).

Cette analyse nous conduit « naturellement » à la question de l'identification automatique des problèmes. Le chapitre suivant propose donc un petit état de l'art pour chaque problème de qualité recensé, et décrit une heuristique que nous avons développée pour un problème spécifique (problème d'étiquetage polysémique des concepts).

7

Vers la détection automatique des problèmes. Focus sur la détection des étiquettes polysémiques

SOMMAIRE

7.1 Introduction

Dans le chapitre précédent, nous avons mis en évidence des liens entre les problèmes de qualité dans les ontologies construites automatiquement et les compromis faits par les approches de construction automatique dans l'implémentation et l'exécution des différentes tâches du processus de construction. Bien que de nouvelles approches et des améliorations soient proposées régulièrement dans la littérature, les ontologies construites automatiquement restent affectées par de nombreux problèmes de qualité. Une vérification préalable suivie d'une correction reste donc nécessaire avant leur intégration dans un système décisionnel. Dans cette optique, différents chercheurs ont développé des outils qui peuvent détecter certains problèmes de qualité et, pour les plus avancés, identifier les causes du problème ou proposer à l'utilisateur des solutions pour l'amélioration de l'ontologie.

Dans ce chapitre, nous présentons tout d'abord un état des lieux des méthodes proposées pour la détection de chaque type de problèmes de qualité. Puis, nous développons une nouvelle approche pour la détection d'un problème qui est apparu fréquemment dans nos expérimentations : les concepts ayant une étiquette polysémique (problème S8).

7.2 Méthodes pour la détection des problèmes de qualité

Dans la typologie des problèmes de qualité introduite en Section 5.3, nous pouvons distinguer les problèmes dûs à des différences entre l'ontologie et ses modèles intentionnels ou son interprétation des problèmes internes à l'ensemble des formules qui compose l'ontologie. Pour la détection, cette distinction est importante : les problèmes du premier type peuvent être détectés en comparant l'ontologie avec des ressources externes alors que la détection des problèmes du deuxième type ne nécessite aucune ressource additionnelle. Nous appelons *problèmes extrinsèques* les problèmes du premier type et *problèmes intrinsèques* les problèmes du second type. Dans notre typologie, les problèmes intrinsèques renvoient tous à l'aspect logique des ontologies.

La détection des problèmes intrinsèques a été largement étudiée [19, 138]. En revanche, à notre connaissance, peu de travaux concernant la détection des problèmes extrinsèques ont été publiés dans la littérature.

7.2.1 La détection des problèmes affectant l'aspect logique des ontologies

Les problèmes L1, L6, L7, L8, L9, L10 et L11 sont des problèmes intrinsèques, et les problèmes L2, L3, L4 et L5 sont des problèmes extrinsèques. Et, une partie des problèmes L12 (les redondances) est intrinsèque, l'autre partie étant extrinsèque.

7.2.1.1 Les problèmes intrinsèques

7.2.1.1.1 Inconsistance logique (L1) et Insatisfiabilité (L10) Les inconsistances logiques et les concepts insatisfiables sont probablement les problèmes les plus étudiés dans la littérature et sont efficacement détectés par les raisonneurs logiques (ex. Pellet [121], HermiT [88], Racer [62], etc.). Une analyse détaillée des algorithmes utilisés par les moteurs d'inférences

dépasse le cadre de cette thèse. On peut juste noter que pour la détection des problèmes d'inconsistance les raisonneurs recherchent des contradictions entre les axiomes qui composent l'ontologie ou qui ont été inférés. Une des techniques utilisées pour la détection des concepts insatisfiables vérifie, pour chaque concept qui n'a pas d'instance, si l'ontologie reste consistante après l'ajout d'une instance au concept.

Bien qu'en théorie, le nombre de situations qui puisse conduire à des inconsistances dans une ontologie est infini [8], en pratique la plupart des inconsistances sont dûes à des situations qui peuvent être caractérisées par un nombre réduit de patrons [19, 138] (ex. un concept c_i qui a comme père simultanément un concept c_j et son complément $\neg c_j$). Ces patrons permettent d'identifier des problèmes d'inconsistance sans recours à un raisonneur.

Même pour des ontologies de taille réduite la compréhension des causes des problèmes d'inconsistance ou d'insatisfiabilité peut être un problème non trivial. Pour aider les utilisateurs, plusieurs outils permettent le calcul des justifications (des sous-ensembles minimaux d'axiomes de l'ontologie qui sont suffisants pour causer le problème) : RADON [1] [67], DION [2], SWOOP [3] [69].

7.2.1.1.2 Eléments de l'ontologie équivalents (L6), indifférenciables (L7) ou correspondant à un ET (L8) ou un OU (L9) Les éléments distincts de l'ontologie équivalents du point de vue logique sont détectés par les raisonneurs logiques [8]. Les éléments indifférenciables du point de vue logique peuvent être identifiés facilement à l'aide de la comparaison de paires d'éléments par des requêtes sur l'ontologie. Les éléments correspondant à une disjonction ou à une conjonction peuvent être identifiés à l'aide de requêtes sur les patrons qui leur correspondent.

7.2.1.1.3 Complexité élevée des inférences (L11) Plusieurs études récentes [19, 56, 75] ont mis en évidence un lien opaque et contre-intuitif entre les performances des différents moteurs d'inférences et le nombre d'axiomes d'une ontologie, leur taille et leur complexité. *A priori*, les performances des moteurs d'inférences les plus connus pour OWL-DL sont non-déterministes [19] et, comme ci-dessus (7.2.1.1.1), des patrons en nombre restreint ont été identifiés [75] ; ce qui à conduit au développement de l'outil Pellint permettant d'identifier des axiomes (ou des groupes d'axiomes) de l'ontologie qui sont susceptibles de rendre très complexes les processus d'inférences.

7.2.1.1.4 Ontologie non minimale (L12) L'identification des éléments redondants dans l'ontologie est une tâche simple pour laquelle plusieurs outils ont été proposés (ex. Odeval [29]). En revanche, l'identification des éléments inutiles de l'ontologie (inutiles parce qu'ils ne font pas partie des modèles intentionnels) est un problème non trivial qui, à notre connaissance, est encore ouvert.

1. RaDON : http://radon.ontoware.org/demo-codr.htm
2. DION : http://wasp.cs.vu.nl/sekt/dion/
3. SWOOP : http://www.mindswap.org/2004/SWOOP/

7.2.1.2 Les problèmes extrinsèques

La détection formelle des problèmes extrinsèques dans une ontologie requiert les modèles intentionnels de l'ontologie ou, le cas échéant, un ensemble d'exemples et de contre-exemples suffisamment large.

Dans ce cas, l'identification des problèmes d'ontologie inadaptée (L2) se ramène à la recherche de contradictions dans l'ensemble des formules obtenu par l'union de l'ontologie avec les modèles intentionnels ou les exemples et de la présence de contre-exemples parmi les formules qui peuvent être inférées à partir de l'ontologie. A chaque contradiction identifiée et à chaque contre-exemple inféré à partir de l'ontologie correspond un problème d'ontologie inadaptée.

On retrouve une argumentation similaire pour les problèmes L3, L4 et L5 : l'identification de problèmes d'ontologie incomplète (L3) se ramène à la recherche de contre-exemples qui, ajoutés à l'ontologie, ne produisent pas de contradictions et des formules présentes dans les modèles intentionnels ou dans les exemples qui ne peuvent pas être obtenues par inférence à partir de l'ontologie ; l'identification de problèmes d'inférences incorrectes (L4) se ramène à la recherche de contradictions entre les formules obtenues par inférence à partir d'une ontologie adaptée (ne contenant aucun problème d'ontologie inadaptée) et les exemples ou les modèles intentionnels et de la présence de contre-exemples parmi les formules obtenues par inférence à partir d'une ontologie adaptée (ne contenant aucun problème d'ontologie inadaptée) ; et l'identification de problèmes d'inférences incomplètes (L5) se ramène à la recherche de contre-exemples qui, ajoutés à l'ontologie et aux formules obtenues par inférence à partir d'une ontologie complète (ne contenant aucun problème d'ontologie incomplète), ne produisent pas de contradictions et de formules présentes dans les modèles intentionnels ou dans les exemples qui ne peuvent pas être obtenues par inférence à partir d'une ontologie complète.

Si les modèles intentionnels ou un ensemble d'exemples et de contre-exemples ne sont pas disponibles, une identification formelle de ces problèmes n'est pas possible. Cependant, des outils (par exemple, OOPS ! [108]) permettent l'identification de plusieurs cas particuliers de ces problèmes (par exemple, les embûches P10 - « absence de la disjonction » et P11 - « absence du domaine / codomaine dans la déclaration des propriétés »– voir table 5.1). Ils implémentent des heuristiques qui permettent d'identifier des situations susceptibles de correspondre à des problèmes de qualité ; par exemple, il n'est pas usuel que le domaine d'une relation ne soit pas précisé.

7.2.2 La détection des problèmes affectant l'aspect social des ontologies

Les problèmes qui affectent l'aspect social des ontologies sont tous des problèmes extrinsèques reposant sur l'interprétation que l'utilisateur donne à l'ontologie. Huit problèmes (S1 - S4, S6 - S8, S12) affectent des aspects locaux de l'ontologie et quatre problèmes (S5, S9, S10, S11) ont une perspective globale sur l'ontologie. A notre connaissance, la vérification systématique de la présence de problèmes affectant l'aspect social des ontologies est une question ouverte. Pour les problèmes locaux, certaines méthodes de détection basées sur des heuristiques et des patrons ont été proposées. Elles sont adaptées à des cas particuliers ; par exemple, l'utilisation d'une « classe par défaut » ou des « disjonctions lointaines ».

Plusieurs mesures ont été proposées pour évaluer la qualité des **annotations (S5)** [22]. Toutefois, elles ne permettent pas de mesurer l'impact des annotations dans le cadre du processus d'interprétation de l'ontologie par un utilisateur.

Les problèmes d'**ontologie plate (S9)** sont détectables grâce à différentes mesures combinatoires (par exemple, le ratio entre le nombre de concepts et celui des relations taxonomiques).

Les standards utilisés pour la **formalisation (S10)** des ontologies et les mécanismes de certification sont étroitement liés aux usages acceptés dans le cadre de la communauté. Actuellement, dans le domaine de l'ingénierie des connaissances, le langage OWL s'est imposé comme standard pour la formalisation des ontologies. Au-delà du langage utilisé pour la formalisation, on trouve plusieurs guides de bonnes pratiques concernant des problématiques très variées : comment nommer les concepts, comment structurer les annotations, comment simplifier l'ontologie en regroupant les attributs répétitifs, etc. Certains outils (par exemple OOPS ! [108]) permettent la vérification de la conformité de l'ontologie avec ces critères.

Bien que le problème de la **certification (S11)** de la qualité des ontologies ait été abordé dans la littérature, il reste aujourd'hui un problème ouvert. Il n'existe encore aucune autorité ou mécanisme de certification permettant de garantir la qualité d'une ontologie [68].

7.3 La détection des problèmes S8 – concepts ayant une étiquette polysémique

En vue d'une aide à la correction automatique, se pose préalablement le problème de l'ordre de la détection des problèmes. Par exemple, on peut s'interroger sur l'intérêt de détecter des relations taxonomiques redondantes tant qu'on n'a pas éliminé toutes les relations taxonomiques incorrectes, sachant que la présence de ces dernières peut conduire à des fausses détections de relations redondantes.

Intuitivement, il semble qu'il faille commencer par l'élimination des concepts et des relations taxonomiques inutiles, puis par la correction des relations taxonomiques incorrectes et l'ajout des concepts et des relations taxonomiques manquantes.

De plus, on sait que pour la détection et la correction automatique des problèmes d'ontologie non minimale, incomplète ou inadaptée, il faut disposer de modèles intentionnels ou d'un ensemble d'exemples et de contre-exemples et que, en pratique, ces ressources sont souvent indisponibles ; ce qui rend nécessaire une détection manuelle. On peut donc s'interroger si, d'un point de vue opérationnel, il n'est pas plus intéressant de commencer par la détection des problèmes qui peuvent être détectés automatiquement. En effet, même si des faux positifs sont détectés à cause des problèmes d'incomplétude ou des relations taxonomiques incorrectes, lors de l'analyse des résultats de la détection automatique ces faux positifs peuvent indiquer à l'utilisateur, de manière indirecte, la présence d'autres problèmes de qualité dans l'ontologie.

Nous faisons donc ici l'hypothèse qu'il est intéressant de pouvoir détecter automatiquement le plus grand nombre de problèmes même lorsque les problèmes d'ontologie inadaptée ou incomplète n'ont pas été préalablement corrigés.

Dans ce cadre, nous avons choisi de nous focaliser sur un problème fréquent dans nos expérimentations : la détection des concepts décrits par des étiquettes polysémiques (problème

S8).

Comme nous l'avons mentionné dans la Section 5.3.2.2.2, des analogies existent entre les problèmes sociaux S8 et les problèmes logiques L8 et L9 (éléments d'ontologie ET et OU) : les éléments ayant des étiquettes polysémiques peuvent être interprétés comme l'union ou comme l'intersection des différentes significations de l'étiquette.

7.3.1 Un rapide état de l'art

Le problème de la désambiguïsation d'un terme lors de sa découverte – avant de le transformer en concept – est un problème classique dans le domaine de l'extraction des connaissances à partir de textes et un nombre important d'approches ont été proposées [90]. Elles utilisent l'analyse du contexte d'apparition des termes dans le texte et, pour certaines, des ressources externes (e.g. WordNet, textes annotés) pour préciser le sens des termes.

D'après Navigli, 2009 [90], bien que l'objectif le plus important des recherches sur les méthodes de désambiguïsation soit leur adaptation en vue d'une utilisation dans des applications et que ces dernières soient souvent centrées sur des domaines très spécifiques, la désambiguïsation orientée domaine reste une problématique ouverte. La désambiguïsation orientée domaine regroupe les approches qui cherchent à préciser le sens le plus probable des termes dans un domaine et à améliorer la qualité de leurs résultats grâce à l'utilisation de connaissances spécifiques au domaine.

Parmi les travaux sur la désambiguïsation quelques uns ont porté précisément sur la détection et la désambiguïsation d'étiquettes de concepts polysémiques [40, 107, 110].

Poveda et al. [107] ont décrit une situation où un concept de l'ontologie ayant une étiquette polysémique est utilisé dans une ontologie pour représenter deux notions distinctes : par exemple, l'étiquette « Théâtre » désignant (1) la discipline artistique et (2) le lieu où les représentations sont exécutées. En fonction du contexte d'apparition de ce concept dans l'ontologie (i.e. ses éventuels concepts pères et les relations dans lesquelles il est impliqué), il est possible que cette situation corresponde à un problème S8 (étiquette polysémique) mais aussi à un problème L10 (concept insatisfiable) – si le concept « Théâtre » a deux pères disjoints « Discipline artistique » et « Lieu »– ou à un problème L8 (concept OU) – si le concept « Théâtre » est défini comme l'union de deux concepts « Théâtre romain » (un objectif touristique dans une ville) et « Théâtre d'improvisation » (un type particulier de la discipline artistique théâtre) qui n'ont en commun aucun rôle. L'outil *OOPS!* [108] ne peut pas, pour l'instant, détecter ce type de situations.

Fang et al. [40] se sont intéressés à la situation où un concept ayant une étiquette polysémique est utilisé dans l'ontologie pour représenter une seule notion. Ils ont proposé un algorithme qui détermine le sens le plus probable de l'étiquette en analysant les relations entre les synsets de WordNet associés aux mots qui composent l'étiquette et les contextes d'apparition du concept (1) dans l'ontologie (contexte défini comme l'ensemble de concepts qui se trouvent dans l'ontologie à une distance inférieure à un seuil du concept analysé) et (2) dans des documents annotés avec les concepts de l'ontologie (contexte défini comme l'ensemble de concepts avec lesquels ont été annotés les termes voisins – dans une fenêtre donnée – des termes annotés avec le concept analysé). Cette analyse associe le synset le plus probable à chaque mot qui compose l'étiquette et la signification de celle-ci est donnée par l'union de ces synsets.

Qasim et Khan [110] se sont intéressés au problème qui apparaît lors d'un alignement d'ontologies où chaque ontologie contient un concept ayant chacun la même étiquette polysémique mais une signification différente. Ils ont proposé une désambiguïsation de l'étiquette des concepts à l'aide de leur voisinage ontologique défini par les concepts pères / fils, les relations non taxonomiques dans lesquels les concepts sont impliqués, leurs attributs, leurs instances, etc.

Pour la détection et la désambiguïsation des étiquettes polysémiques les différentes approches s'appuient sur l'analyse du contexte d'apparition des concepts dans l'ontologie en faisant l'hypothèse que ces contextes ne contiennent aucun élément inutile ou incorrect. Pour les ontologies construites automatiquement, cette hypothèse est discutable en l'absence d'une validation manuelle préalable à la détection d'étiquettes polysémiques. Tout en étant bien conscients que la présence d'éléments inutiles ou incorrects dans l'ontologie peut conduire à la détection des faux positifs, nous considérons que la détection automatique d'étiquettes polysémiques basée sur l'analyse du contexte d'apparition des concepts reste intéressante.

7.3.2 Un nouvel algorithme pour la détection d'étiquettes polysémiques dans les ontologies construites par Text2Onto

L'algorithme que nous proposons cherche a détecter automatiquement les étiquettes potentiellement polysémiques grâce à l'analyse des significations attribuées aux étiquettes par un des algorithmes d'extraction de relations taxonomiques de Text2Onto, plus précisément celui basé sur WordNet – qu'on nomme ici *T2O-WordNet*.

Nous rappelons que, dans WordNet, chaque mot est associé à un ou plusieurs synsets et que des relations d'hyponymie / hyperonymie peuvent exister entre deux synsets (figure 7.1).

Pour identifier une relation taxonomique entre deux concepts, l'algorithme *T2O-WordNet* se base uniquement sur l'existence d'un lien de hyponymie / hyperonymie entre au moins deux synsets associés aux étiquettes des deux concepts. De ce fait, il est possible que l'étiquette d'un concept soit associée à plusieurs synsets différents soit parce qu'elle a été associée à des synsets différents dans des relations différentes, soit parce qu'une même relation peut être déduite à partir d'au moins deux synsets différents associés à la même étiquette.

Par exemple, lors de la construction de l'ontologie O_2 (Section 6.3) Text2Onto a extrait les termes « repair », « area » et « change » et les deux relations taxonomiques (1) « repair *is-a* area » et (2) « repair *is-a* change ». Dans WordNet, le mot « repair » est associé à trois synsets et, pour détecter chacune des deux relations taxonomiques l'algorithme *T2O-WordNet* a associé le mot « repair » à deux synsets différents : pour la relation (1) au synset « un endroit visité fréquemment » et pour la relation (2) au synset « l'acte de faire fonctionner quelque chose à nouveau ».

L'algorithme que nous proposons prend en entrée la liste des relations taxonomiques extraites par l'algorithme *T2O-WordNet* et produit en sortie une liste de concepts dont l'étiquette est supposée polysémique (Procédure 1). On considère qu'il est probable que l'étiquette d'un concept soit polysémique s'il n'existe aucun synset commun à toutes les listes de synsets avec lesquels l'étiquette a été associée lors de la détection de chaque relation taxonomique du concept.

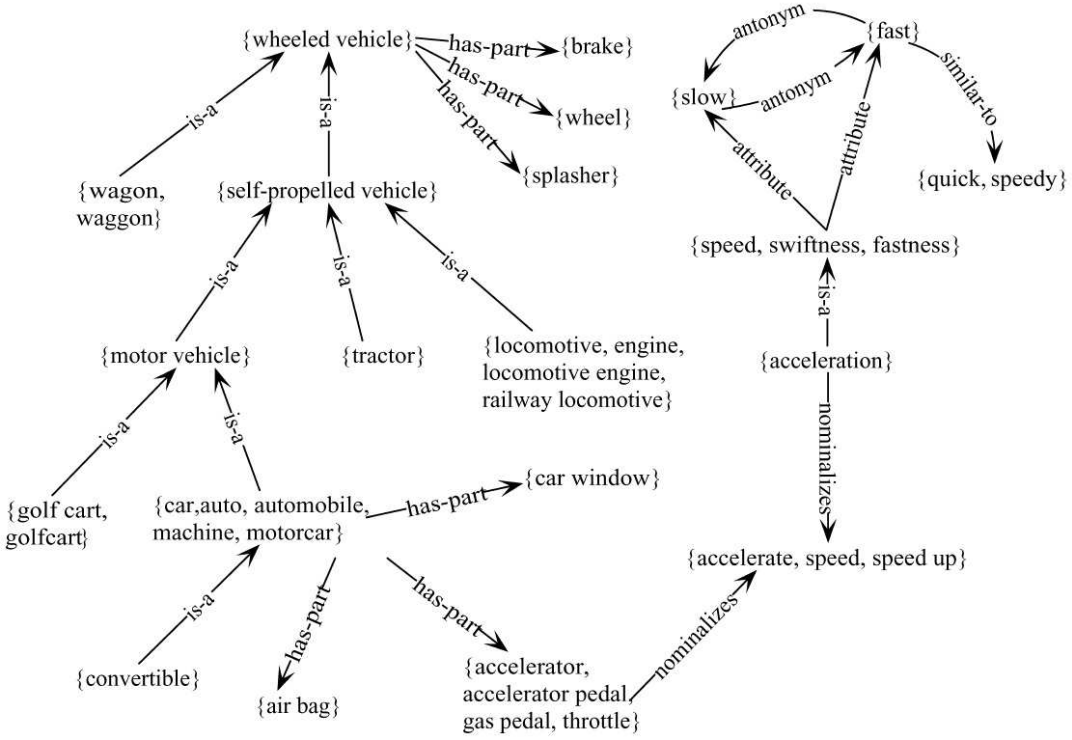

FIGURE 7.1: Illustration des relations sémantiques du synset { car_n^1, $auto_n^1$, $automobile_n^1$, $machine_n^6$, $motorcar_n^1$ } associé au mot *automobile* ; les indices n, v, a indiquent qu'il s'agit d'un nom, d'un verbe ou d'un adjectif ; les indices numériques indiquent le numéro du sens du terme avec lequel il est associé au synset ; par exemple, $machine_n^6$ indique que le terme *machine* est associé au synset avec son sixième sens en tant que nom, alors qu'il a dans WordNet cinq autres sens en tant que nom et deux sens en tant que verbe.

Procédure 1 DetecterEtiquettesPolysemiques(Wn, LR_w)

Entrée: Wn - WordNet, et LR_w - la liste des relations taxonomiques extraites par l'algorithme *T2O-WordNet*

Sortie: LC_p - liste des concepts ayant des étiquettes potentiellement polysémiques

1: $LC_p = \emptyset$
2: $LC^* = ExpliciterConcepts(Wn, LR_w)$
3: **for all** concept explicité c_i^* in LC^* **do**
4: **if** $|c_i^*.synsets\_communs| = 0$ **then**
5: $LC_p = LC_p \cup c^*$
return LC_p

L'algorithme est basé sur l'explicitation des concepts impliqués dans les relations extraites par l'algorithme *T2O-WordNet* (Procédure 1). La fonction *ExpliciterConcepts* (Procédure 2) construit une structure c^* (nommée « concept explicité ») pour chaque concept c impliqué dans une des relations de la liste LR_w prise en entrée. Cette structure regroupe :

– $c^*.etiquette$ – le terme qui est utilisé comme étiquette du concept ;

- $c^*.synsets$ – la liste des tous les synsets qui sont associés dans WordNet au terme $c^*.etiquette$;
- $c^*.relations$ – la liste des relations taxonomiques explicitées dans lesquelles le concept est impliqué en tant que père ou en tant que fils ; une relation taxonomique explicitée est une structure r^* construite par la fonction *ExpliciterRelation* (Procédure 3) et qui regroupe : $r^*.fils$ et $r^*.pere$ – les étiquettes du concept fils et du concept père ; $r^*.synsets\_fils$ et $r^*.synsets\_peres$ – deux listes contenant les synsets avec lesquels les étiquettes $r^*.fils$ et resp. $r^*.pere$ ont été associées lors de la détection de la relation taxonomique par l'algorithme *T2O-WordNet*.
- $c^*.synsets\_communs$ – la liste des synsets qui sont communs à toutes les listes $r_i^*.synsets\_fils$ (ou $r_i^*.synsets\_peres$) associées au concept par chacune des relations $r_i \in c^*.relations$; cette liste est calculée par la fonction $SynsetsCommuns$ (Procédure 4).

Exemple 7.3.1 Supposons que lors de la construction de l'ontologie O_2 l'algorithme *T2O-WordNet* a extrait seulement deux relations taxonomiques : r_1 = « repair *is-a* area » et r_2 = « repair *is-a* change ». Alors, dans la fonction *ExpliciterConcepts* (Procedure 2) :

- $LR_w = \{ r_1, r_2 \}$,
- $LC = \{ repair, area, change \}$, et
- $LC^* = \{ repair^*, area^*, change^* \}$ où
  - $repair^*.synsets = \{ S_{repair_n^1}, S_{repair_n^2}, S_{repair_n^3} \}$ [4]
  - $area^*.synsets = S_{area_n^1}, S_{area_n^2}, \ldots, S_{area_n^6} \}$ [5]
  - $change^*.synsets = \{ S_{change_n^1}, S_{change_n^2}, S_{change_n^3}, \ldots, S_{change_n^{10}} \}$ [6]
  - $repair^*.relations = \{ r_1^*, r_2^* \}$
  - $area^*.relations = \{ r_1^* \}$
  - $change^*.relations = \{ r_2^* \}$

Comme la détection de r_1 est basée seulement sur l'existence d'un lien d'hyperonymie entre les synsets $S_{repair_n^3}$ et $S_{area_n^1}$, et celle de r_2 seulement sur le lien entre $S_{repair_n^1}$ et $S_{change_n^3}$, on peut écrire :

- $r_1^*.synsets\_fils = \{ S_{repair_n^3} \}$
- $r_1^*.synsets\_peres = \{ S_{area_n^1} \}$
- $r_2^*.synsets\_fils = \{ S_{repair_n^1} \}$
- $r_2^*.synsets\_peres = \{ S_{change_n^3} \}$

En conséquence, les listes de synsets communs associées aux concepts sont :

- $repair^*.synsets\_communs = \emptyset$
- $area^*.synsets\_communs = \{ S_{area_n^1} \}$

4. Les synsets associés au terme « repair » :
$S_{repair_n^1} = \{repair_n^1, fix_n^3, fixing_n^1, fixture_n^4, mend_n^2, mending_n^2, reparation_n^3$ – the act of putting something in working order again $\}$,
$S_{repair_n^2} = \{repair_n^2$ – a formal way of referring to the condition of something $\}$,
$S_{repair_n^3} = \{haunt_n^1, hangout_n^1, resort_n^2, repair_n^3, stampingground_n^1$ – a frequently visited place $\}$.

5. $S_{area_n^1} = \{area_n^1, country_n^5$ – a particular geographical region of indefinite boundary (usually serving some special purpose or distinguished by its people or culture or geography) $\}$.

6. $S_{change_n^3} = \{change_n^3$ – the action of changing something $\}$.

– $change^*.synsets\_communs = \{ S_{change^3_n} \}$

Comme il n'existe aucun synset commun pour le concept $repair^*$ il est ajouté à LC_p, la liste des concepts dont l'étiquette est supposée polysémique.

Procédure 2 ExpliciterConcepts(Wn, LR_w)

Entrée: Wn - WordNet, et LR_w - la liste des relations taxonomiques extraites par l'algorithme *T2O-WordNet*

Sortie: LC^* - la liste de concepts explicités

$LC = \emptyset$
$LC^* = \emptyset$
for all relation r in LR_w **do**
    $LC = LC \cup r.pere$
    $LC = LC \cup r.fils$
for all concept c_i in LC **do**
    $c_i^*.etiquette = c_i$
    $c_i^*.relations = \emptyset$
    $c_i^*.synsets = Wn.getSynsets(c_i)$
    $LC^* = LC^* \cup c_i^*$
for all relation r_i in LR_w **do**
    $r_i^* = ExpliciterRelation(Wn, r_i)$
    for all c_j^* in LC^* **do**
        if $r_i^*.pere = c_j^*.etiquette \quad \| \quad r_i^*.fils = c_j^*.etiquette$ **then**
            $c_j^*.relations = c_j^*.relations \cup r^*$
for all concept explicité c_i^* in LC^* **do**
    $c_i^*.synsets\_communs = SynsetsCommuns(c_i^*)$
return LC^*

Procédure 3 ExpliciterRelation(Wn, r)

Entrée: Wn - WordNet, et r - une relation taxonomique extraite par l'algorithme *T2O-WordNet*

Sortie: r^* - relation taxonomique explicitée

$r^*.pere = r.pere$
$r^*.fils = r.fils$
$LS_p = Wn.getSynsets(r.pere)$
$LS_f = Wn.getSynsets(r.fils)$
for all synset S_p in LS_p **do**
    for all synset S_f in LS_f **do**
        if $Wn.isHypernym(S_p, S_f)$ **then**
            $r^*.synsets\_fils = r^*.synsets\_fils \cup S_f$
            $r^*.synsets\_peres = r^*.synsets\_peres \cup S_p$
return r^*

Procédure 4 SynsetsCommuns(c^*)

Entrée: concept explicité c^*

Sortie: LS_c - liste des synsets communs à toutes les relations explicitées du c^*

1: $LS_c = c^*.synsets$
2: **for all** relation r^* in $c^*.relations$ **do**
3: **if** $c^*.etiquette = r^*.pere$ **then**
4: $LS_c = LS_c \cap r^*.synsets\_pere$
5: **else**
6: $LS_c = LS_c \cap r^*.synsets\_fils$
return LS_c

7.3.3 Expérimentation : détection manuelle vs. détection automatique

Pour analyser expérimentalement les performances de l'algorithme, nous avons comparé ses résultats avec ceux obtenus manuellement suite à l'analyse présentée dans la Section 6.3.

Nous rappelons que, lors de la détection manuelle des étiquettes polysémiques, nous avons analysé l'ontologie telle qu'elle a été construite par Text2Onto sans avoir corrigé les erreurs qu'elle contient, et nous avons retenu toutes les étiquettes qui, à notre avis, pourraient être associées à au moins deux significations différentes. Parmi ces étiquettes certaines correspondent à des concepts qui ne devraient pas faire partie de l'ontologie parce qu'ils ne font pas partie *a priori* du domaine représenté ; par exemple, les étiquettes *« object »* et *« view »* pour l'ontologie du domaine « construction d'ontologies à partir de textes ». Certaines étiquettes peuvent être également interprétées comme polysémiques parce que les concepts correspondants sont impliqués dans une relation taxonomique incorrecte ou qui ne fait pas partie du domaine ; par exemple, l'étiquette *« relation »* a été considérée comme polysémique parce que le concept correspondant est impliqué dans la relation *« relation » is-a « object »* qui ne fait pas partie du domaine.

Table 7.1: Résultats obtenus avec l'algorithme de détection d'étiquettes polysémiques.

	Ontologie O_1	Ontologie O_2
Manuellement	64	69
Algorithme	51	63
Manuellement ∩ Algorithme	51	56
Précision algorithme	100%	88%
Rappel algorithme	80%	91%

La Table 7.1 résume les résultats obtenus qui indiquent un bon rappel (80% et respectivement 91%) et une bonne précision (100% et respectivement 88%) pour l'algorithme proposé.

Nous avons néanmoins analysé les raisons de l'absence de détection des étiquettes polysémiques par l'algorithme et de la détection d'étiquettes polysémiques non détectées manuellement. Toutes les étiquettes polysémiques détectées manuellement mais pas par l'algorithme correspondent à des concepts impliqués dans des relations taxonomiques autres que celles identifiées par l'algorithme *T2O-WordNet*. Toutes les étiquettes polysémiques détectées par

l'algorithme mais pas manuellement sont correctement détectées comme polysémiques. Tous les concepts ayant des étiquettes polysémiques détectés par l'algorithme mais pas manuellement sont correctement détectés comme polysémiques. L'absence de détection manuelle est certainement dûe à l'absence d'une visualisation intelligible de l'ontologie O_2 qui contient 965 concepts et 408 relations taxonomiques. Cette remarque ouvre la voie à l'intégration de supports visuels adaptés dans l'analyse des ontologies. Mais cette perspective dépasse le cadre de cette thèse.

7.3.4 Discussion : avantages et inconvénients de l'algorithme proposé

La présence d'étiquettes polysémiques est un problème (S8) extrinsèque dont la détection nécessite *a priori* une interprétation par un expert. Cependant, il est envisageable de détecter automatiquement tous les concepts ayant des étiquettes pour lesquelles une interprétation unique n'est pas possible à cause des relations taxonomiques impliquant le concept.

Bien que l'impossibilité d'associer une signification unique à une étiquette ne puisse pas garantir que, lors d'une interprétation, au moins deux significations différentes puissent être associées à l'étiquette, on peut estimer, intuitivement, que la plupart des étiquettes pour lesquelles le contexte d'apparition du concept dans l'ontologie indique qu'elles n'ont pas une signification unique seront identifiées comme polysémiques lors d'une interprétation manuelle.

Le principal avantage de la détection d'étiquettes polysémiques à l'aide de l'algorithme proposé est la contribution à l'automatisation d'un processus de détection manuelle fastidieux et susceptible d'être entravé d'erreurs. Un autre avantage est que l'algorithme peut être utilisé sur les ontologies construites par Text2Onto (ou un autre outil qui extrait les relations taxonomiques à partir de WordNet) indépendamment de la détection et de la correction d'autres problèmes. Néanmoins, nous sommes bien conscients que la qualité de ses résultats peut être amoindrie par la présence d'autres problèmes de qualité dans l'ontologie.

Une des limitations les plus importantes de l'algorithme est qu'il se base dans son analyse uniquement sur les relations taxonomiques extraites à partir de WordNet en ignorant les relations taxonomiques identifiées à l'aide d'autres techniques ainsi que les autres éléments qui peuvent éventuellement être présents dans l'ontologie (instances, relations non taxonomiques, attributs).

Un autre inconvénient est l'absence de prise en compte du domaine de l'ontologie dans le processus de détection. Mais, comme nous l'avons déjà rappelé (Section 7.3.1), l'intégration des connaissances définissant les limites d'un domaine dans un processus de désambiguïsation reste un problème ouvert.

Malgré ses limitations, on peut estimer que l'utilisation de l'algorithme proposé dans cette section apporte une accélération du processus de détection et de correction des problèmes de qualité affectant les ontologies construites automatiquement. Bien que, à cause de la présence des problèmes L2, L3 et L12, les résultats de l'algorithme peuvent contenir des faux positifs, ces derniers ne sont pas très gênants lors de l'analyse manuelle des résultats de la détection et peuvent constituer des indices indirects de la présence d'autres problèmes dans l'ontologie.

7.4 Conclusion

L'énumération pour chacun des problèmes de qualité des approches de détection automatique des problèmes existantes nous a permis de souligner la difficulté d'automatisation de la détection des problèmes extrinsèques. La plupart des approches existantes qui reposent sur des patrons ou des heuristiques spécifiques ne sont guère adaptées à une généricité des traitements.

Nous finissons ce chapitre par une proposition focalisée sur un problème apparu comme fréquent dans nos expérimentations. Notre objectif pour la détection des étiquettes polysémiques ne prétend évidemment pas éliminer l'indispensable expertise humaine mais notre proposition se positionne plutôt comme une preuve de concept pour montrer que l'automatisation permet de réduire l'effort humain demandé, et donc de conserver le temps de disponibilité restreint des experts à des tâches de forte valeur ajoutée.

8

Conclusion Générale et Perspectives

Les travaux développés dans cette thèse s'inscrivent dans le contexte de la démocratisation de l'utilisation des ontologies dans de nombreux domaines qui, couplée à l'explosion du nombre, de la taille et de la fréquence des modifications des documents qui doivent être étudiés pour leur construction, conduit à un besoin croissant d'automatisation des processus de construction et d'évolution. Ce besoin a stimulé le développement d'approches associées à des composants logiciels proposant différents degrés d'automatisation. Cependant, les retours d'expériences rapportées dans la littérature semblent rendre compte d'une variabilité importante des résultats associée à une qualité parfois défaillante.

Dans ce contexte, et avec l'objectif de contribuer à l'amélioration de la qualité des ontologies construites automatiquement, nous avons axé nos travaux selon trois axes principaux : (1) une comparaison fonctionnelle, technique et expérimentale des approches (avec les outils qui leur sont associés) pour la construction automatique d'ontologies à partir de textes, (2) la construction d'une typologie des problèmes de qualité qui affectent les ontologies, et (3) une première réflexion vers l'automatisation de la détection des problèmes de qualité.

Nous avons proposé un protocole de comparaison des approches qui comporte trois étapes complémentaires : (1) à l'aide du référentiel de tâches de Methontology les approches ont été comparées sur la base de leur degré de complétude et d'automatisation, et du type de techniques utilisées pour la construction ; (2) puis sur la base des caractéristiques techniques et fonctionnelles des outils associés aux approches, et (3) par une analyse expérimentale avec une comparaison avec une ontologie de référence construite manuellement. A notre connaissance, ce protocole est plus complet que ceux – peu nombreux – qui ont été proposés dans la littérature. Nous avons tenté de limiter la subjectivité de l'interprétation dans la comparaison par le recours à des mesures (précision, rappel par rapport à une ontologie de référence) mais nous sommes bien conscients des limites de cette démarche dans les cas réels d'applications où les retours d'utilisateurs doivent rester un élément prédominant de l'évaluation.

Nos expérimentations ont confirmé la forte dépendance des performances avec à la fois la nature des textes en entrée et les algorithmes de construction utilisés. L'explication des dépendances reste cependant une question souvent ouverte [87].

De plus, bien que souhaitant nous focaliser sur des approches automatiques, la première étape de notre comparaison a mis en évidence le fait que la plupart des approches développées n'implémentaient pas toutes les étapes prônées par Methontology et qu'elles nécessitaient, pour certaines étapes, un recours à des tâches manuelles. L'évaluation de l'effort de construction d'une ontologie par combinaison de tâches automatiques et manuelles est à notre connaissance une question également largement ouverte.

Pour mieux identifier les problèmes de qualité rencontrés, nous avons proposé une typologie des problèmes potentiels. La construction de cette typologie a été motivée par la variété des propositions que nous avons rencontrées dans notre état de l'art et l'absence d'une approche standardisée. Notre typologie combine deux dimensions intervenant dans l'interprétation des problèmes de qualité : les erreurs *versus* les situations indésirables et les aspects logiques *versus* les aspects sociaux. Sa construction nous a conduit à identifier 24 classes de problèmes qui recouvrent, en les complétant, les problèmes que nous avons pu recenser dans la littérature. Pour compléter l'analyse, nous avons confronté cette typologie aux différentes tâches de Methontology pour tenter d'identifier les facteurs explicatifs les plus probables. Et nous l'avons éprouvée expérimentalement pour l'identification des problèmes de qualité rencontrés lors d'un processus de construction automatique. Cette analyse expérimentale n'est que partielle puisque pour avoir un cadre de comparaison « sans biais » nous avons exclu les connaissances exogènes relatives aux tâches d'usage, aux domaines et aux utilisateurs. Elle nécessitera évidemment à l'avenir des prolongements dans un cadre applicatif que nous pouvons maîtriser. Cependant, nous pensons qu'elle a le mérite de mettre en évidence la prédominance de certains problèmes dont l'identification automatique et la réparation peuvent être envisagées, et de détecter certains « maillons faibles » sur lesquels des efforts devront porter à l'avenir.

Si au début de la thèse, nous avions espoir de contribuer plus amplement au développement d'un système opérationnel d'aide à l'identification automatique de problèmes de qualité, nous n'avons finalement abordé cette question de l'automatisation qu'en dernière partie de la thèse, l'identification et la catégorisation des problèmes étant apparues comme un préalable indispensable à la mise en place d'un tel système. Nous avons cependant commencé à recenser pour chaque problème de notre typologie quelques unes des méthodes d'identification automatique existantes ; cela nous a également permis de mettre en évidence des problèmes qui semblent encore ouverts. Et, nous avons proposé une heuristique pour un problème fréquent que nous avons identifié (la polysémie des étiquettes). Il s'agit juste ici d'une « preuve de concept » que nous avons testée expérimentalement dans le cadre de notre protocole.

Perspectives à court et moyen termes

Les questions ouvertes lors de notre analyse laissent entrevoir différentes pistes de recherche à plus ou moins long terme.

A court terme, nous souhaitons progresser sur la détection automatique des problèmes de qualité, et en particulier des problèmes L12 et S12 (concepts et relations taxonomiques hors domaine) et S8 (polysémie des étiquettes). Tout d'abord, nous envisageons d'explorer les possibilités offertes par WordNet Domains [1] [84] pour la détection des problèmes L12 et S12. En effet, cette extension de WordNet qui associe chaque synset à un ou plusieurs domaines d'une liste de plus de 200 domaines organisés hiérarchiquement semble très prometteuse. Elle a déjà montré son intérêt notamment pour des problèmes de la désambigüisation sémantique [4] et de détection de relations taxonomiques spécifiques [105]. En sus, nous prévoyons de nous appuyer sur WordNet Domains pour intégrer des connaissances de domaine dans l'heuristique de détection d'étiquettes polysémiques que nous avons développée. Pour faciliter leur utilisation effective ces heuristiques pourront être intégrées dans Protégé et NeOn Toolkit.

A moyen terme, nous visons à l'intégration de ces travaux dans un système d'aide à la validation de la qualité des ontologies construites. Dans cet objectif, différentes questions se posent. Nous avons souligné au chapitre 6 les liens entre les différentes phases du processus de construction et la fréquence des problèmes. Ce couplage devrait idéalement être intégré dans un processus de construction automatique mais à notre connaissance les systèmes actuels n'intègrent pas encore une phase d'évaluation efficace à chaque étape. En attendant, et devant la complexité prévisible de l'interopérabilité d'un système d'évaluation avec différents systèmes de construction – très hétérogènes comme on a pu le montrer au chapitre 4 – un système d'aide à l'évaluation autonome semble être un objectif plus raisonnable.

En parallèle, les besoins ne cessent d'évoluer. D'une part, l'intérêt pour la construction automatique d'ontologies qui servent de support à des systèmes de recherche d'information est croissant [35, 48, 111, 113]. Et d'autre part, si les ontologies ont longtemps été considérées – du moins d'un point de vue opérationnel – comme statiques, la question de leur évolution est maintenant au centre des préoccupations [101, 113, 119]. L'évolution des ontologies est définie respectivement par Maedche et al (2003) [78], et par Stojanovic (2004) [124] comme étant : *« la modification appropriée d'une ontologie et la propagation des changements dans les autres ontologies qui en dépendent. »* Une autre définition qui fait référence a été proposée respectivement par Klein, (2004) [71], et par Noy et al (2004) [99], qui dit que *« l'évolution d'une ontologie est la capacité de gérer les changements apportés lors de l'évolution en créant et en maintenant différentes versions d'une ontologie. Cette capacité consiste à identifier et à différencier les versions, à modifier les versions, à spécifier des relations qui rendent explicites les changements effectués entre les versions. »* Ce contexte fait donc évoluer la problématique de l'évaluation en renouvelant certains problèmes (e.g. ontologie incomplète (L3), ontologie non minimale (L12, S12)) et en en ajoutant de nouveaux

1. www.wndomains.fbk.eu

(e.g. problèmes liés à la gestion des versions de l'ontologie).

Notre état de l'art et nos expérimentations ont confirmé l'importance du rôle des utilisateurs experts dans le processus d'évaluation. Cette remarque module évidemment l'objectif « idéal » d'automatisation. Cependant, comme cela se développe de plus en plus dans les domaines connexes de la fouille de données [12] ou de la recherche d'information [9], il s'agit de développer des systèmes qui établissent un couplage entre les traitements automatiques et la part d'expertise humaine difficilement modélisable en l'état. Ce couplage peut être établi efficacement à travers des supports visuels interactifs : « *visual analytics is more than visualization and can rather be seen as an integrated approach combining visualization, human factors and data analysis* » [70]. Les travaux en visualisation de l'information appliqués au génie logiciel nous paraissent une bonne source d'inspiration [112] pour proposer des représentations graphiques interactives de grandes ontologies qui permettent d'explorer à la fois les relations structurelles (la combinatoire des relations) et sémantiques pour faciliter la détection des erreurs.

A

Les définitions des problèmes de qualité de l'état de l'art

Dans le tableau 5.1 nous avons positionné dans la typologie que nous avons proposée l'ensemble des problèmes que nous avons mentionnés dans l'état de l'art présenté dans la Section 5.2.2, dont notamment les 24 embûches décrites par Poveda et al. [107] et les 11 anti-patrons décrits par Roussey et al. [115]. Comme dans la Section 5.2.2 nous avons présenté seulement une description générale des différentes classes d'embuches et d'anti-patrons, sans définir chaque problème, nous présentons dans cette annexe les définitions de tous ces problèmes de qualité.

A.1 Les embûches

Comme nous l'avons précisé dans la Section 5.2.2.4, Poveda et al. [106, 107] ont identifié expérimentalement et décrit de façon informelle 24 types d'embûches (« *pitfalls* »). Nous les énumérons ici en rappelant leurs descriptions informelles.

P1. **Eléments polysémiques** : création dans l'ontologie d'une classe ayant une étiquette polysémique et qui est utilisée pour représenter à la fois au moins deux idées conceptuelles différentes. Par exemple, la classe « Theatre » pour représenter la discipline artistique et le lieu où les représentations sont exécutées.

P2. **Synonymes comme classes** : création dans l'ontologie de plusieurs classes déclarées comme équivalentes et dont les étiquettes sont synonymes. Par exemple, les classes « Waterfall » et « Cascade ».

P3. **Utilisation d'une relation** *ad-hoc* **« is » à la place de « subclass-of »** : la création et l'utilisation d'une relation *ad-hoc* « is » à la place des primitives définies par le langage OWL pour représenter les relations taxonomiques (« subclass-of ») dans l'ontologie.

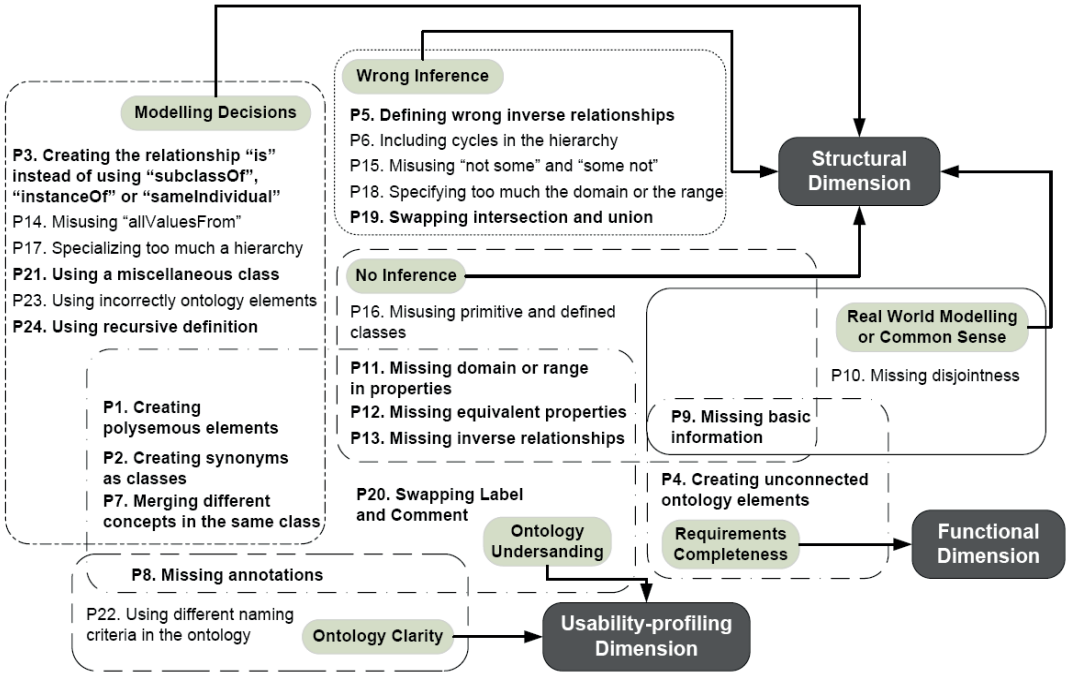

FIGURE A.1: Positionnement des 24 types d'embûches dans les sept classes inspirées des trois dimensions mesurables des ontologies (extrait de Poveda et al., 2010 [107]).

P4. **Eléments d'ontologie non connectés** : création d'éléments de l'ontologie (concepts, relations, attributs) isolés du reste. Par exemple, la création d'une relation « memberOfTeam » sans définir le concept « Team ».

P5. **Relation inverse erronée** : par exemple, si l'on considère un objet qui peut être vendu et acheté les relations « vendu par » et « achète par » ne doivent pas être déclarées comme étant l'une l'inverse de l'autre.

P6. **Cycles dans la hiérarchie** : par exemple, une classe « Professor » comme sous-classe de « Person », et la classe « Person » comme sous-classe de « Professor ».

P7. **Combiner des concepts pour former une classe** : création d'une classe dont l'étiquette fait référence à deux concepts différents. Par exemple, création d'une classe « StyleAndPeriod », ou » ProductOrService « .

P8. **Absence des annotations**

P9. **Absence d'informations basiques** : par exemple, ajouter à une ontologie une relation « follows » mais pas son inverse « precedes ».

P10. **Absence de la disjonction**

P11. **Absence du domaine / codomaine dans la déclaration des propriétés**

P12. **Non déclaration de l'équivalence entre propriétés**

P13. **Non déclaration du fait que deux relations sont l'une l'inverse de l'autre**

P14. **Utilisation abusive de "allValuesFrom"** : utilisation du quantificateur universel à la place du quantificateur existentiel ou pour ne pas permettre l'ajout de nouveaux

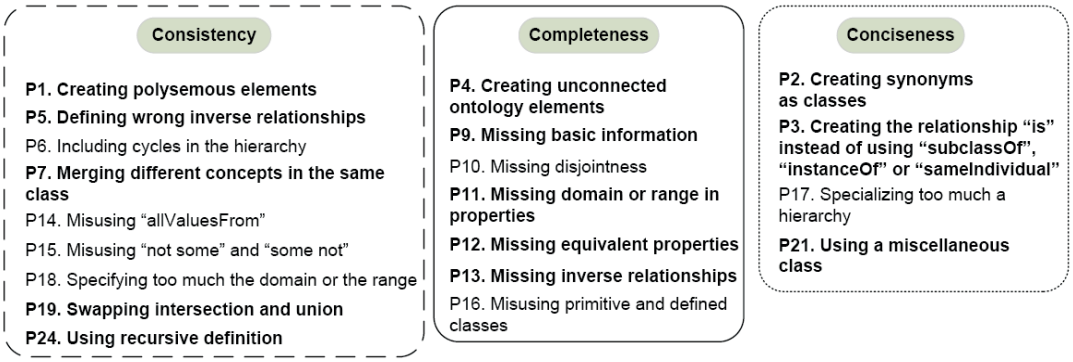

FIGURE A.2: Positionnement de 18 des 24 types d'embûches dans les trois classes d'erreurs de taxonomie (extrait de Poveda et al., 2010 [107]).

éléments au domaine d'une relation. Par exemple, cette définition « $Book \equiv \exists producedBy.Writer \cap \forall uses.Paper$ » ne permet pas d'ajouter « Ink » comme un élément utilisé pour écrire un livre.

P15. **Utilisation abusive de "not some"/"some not"** : par exemple, définir une pizza végétarienne comme une pizza qui contient des ingrédients qui ne sont pas de la viande et des ingrédients qui ne sont pas du poisson, au lieu de préciser que tout ingrédient n'est ni de la viande, ni du poisson.

P16. **Utilisation erronée des classes primitives et définies** : déclarer comme primitives (en précisant seulement des restrictions nécessaires mais pas suffisantes) les classes qui peuvent être définies complètement (à l'aide de des restrictions nécessaires et suffisantes).

P17. **Hyperspécialisation de la hiérarchie** : quand le concepteur de l'ontologie ajoute à la hiérarchie une ou plusieurs sous-classes qui sont trop spécialisées pour accepter des instances. Par exemple, la création des sous-classes « Madrid », « Barcelona », « Sevilla », etc. pour la classe « City ».

P18. **Définition de domaine et / ou de codomaine trop spécifiques** : par exemple, restreindre le domaine de la relation « isOfficialLanguage » à a classe « City », au lieu de permettre que la classe « Country », ou une classe plus générique comme « GeopoliticalObject », puisse avoir une langue officielle.

P19. **Confusion entre intersection et union** : utilisation de l'intersection au lieu de l'union lors de la définition du domaine d'une relation ou d'un attribut. Par exemple, définir le domaine de l'attribut « Name » comme l'intersection des classes « City » et « Drink » (uniquement les boissons qui sont des villes ont l'attribut « Name ») et non comme leur union (les boissons et les villes ont l'attribut « Name »).

P20. **Interchanger les étiquettes avec les annotations**

P21. **Classe par défaut** : création d'une sous-classe par défaut lors de la décomposition d'une classe pour contenir toutes les instances qui appartiennent à la classe racine mais qui n'appartiennent à aucune de ses sous-classes ; cette classe par défaut est nommée

le plus souvent « Other » ou « Miscellaneous ». Par exemple, la classe « HydrographicalResource », et les sous-classes « Stream », « Waterfall », etc., avec la sous-classe par défaut « OtherRiverElement ».

P22. **Utilisation de plusieurs critères de dénomination dans l'ontologie** : par exemple, si l'ontologie contient une classe dont l'étiquette commence avec une lettre majuscule, e.g. « Ingredient », et les étiquettes de ses sous-classes commencent avec une lettre en minuscule, e.g. « animalorigin », « drink », etc.

P23. **Utilisation abusive des éléments de l'ontologie** : par exemple, créer une relation « isEcological » entre une instance de la classe « Car » et les instances « Yes » ou « No », au lieu de créer l'attribut « isEcological » qui accepte des valeurs booléennes.

P24. **Définitions récursives** : un élément de l'ontologie est utilisé dans sa propre définition. Par exemple, préciser que le domaine de la relation « hasFork » est « l'ensemble des restaurant qui ont au moins une valeur pour la relation *hasFork* ».

Comme nous l'avons aussi précisé (Section 5.2.2.4) Poveda et al. [107] ont positionné ces 24 types d'embûches dans sept classes inspirées des trois dimensions mesurables des ontologies (Section 5.2.1.1) et dans les trois classes d'erreurs de taxonomie décrites par Gomez-Perez et al. (Section 5.2.2.1). Les figures A.1 et A.2 illustrent ces positionnements.

A.2 Les anti-patrons

Roussey et al. [115] ont décrit 4 anti-patrons logiques, 3 anti-patrons cognitifs et 4 « conseils ». Nous les énumérons ici en précisant leurs défintions.

A.2.1 Les anti-patrons logiques

Les anti-patrons logiques sont des expressions dans l'ontologie qui contiennent des conflits détectables par les raisonneurs de la logique de description.

L'anti-patron « AndIsOr »

$$c_1 \subseteq \exists r.c_2 \cap c_3 \ , \ disj(c_2, c_3)$$ [1]

Cet anti-patron est une erreur de modélisation courante due à l'ambigüité des conjonctions de coordination « ET » et « OU » qui ne correspondent pas forcément à la conjonction et à la disjonction logiques. Par exemple la phrase « j'aime les gâteaux avec du chocolat et des amandes » est ambiguë. Que contient réellement le gâteau ?

- du chocolat plus des amandes ?
  $Cake \subseteq \exists contain.Chocolate \cap \exists contain.Almond$
- du chocolat aux amandes ?
  $Cake \subseteq \exists contain.(Chocolate \cap Almond)$
- chocolat ou des amandes ?
  $Cake \subseteq \exists contain.(Chocolate \cup Almond)$

L'anti-patron « EquivalenceIsDifference »

$$c_1 \equiv c_2, disj(c_1, c_2)$$

Le concepteur de l'ontologie souhaite formaliser que les instances de c_1 ressemblent aux instances de c_2 mais qu'il y a néanmoins une différence. Par exemple, les instances de c_1 ont une information supplémentaire. Dans ce cas le concepteur aurait dû formaliser cette situation à l'aide d'une relation de subsomption entre c_1 et c_2 : $c_1 \subseteq c_2$.

L'anti-patron « OnlynessIsLoneliness »

$$c_1 \subseteq \forall r.c_2, c_1 \subseteq \forall r.c_3, disj(c_2, c_3)$$

Le concepteur de l'ontologie a créé une restriction universelle pour indiquer que les instances de c_1 peuvent être liées par la propriété r uniquement aux instances de c_2. Au cours du développement de l'ontologie, une nouvelle restriction universelle est ajoutée indiquant que les instances de c_1 peuvent être liées par r uniquement aux instances de c_3, c_2 et c_3 étant disjoints. Généralement, le concepteur a oublié au cours du développement l'existence du premier axiome dans la classe c_1 ou dans l'une de ses classes parentes.

1. $disj(c_2, c_3)$ ne veut pas dire que le concepteur de l'ontologie a précisé de manière explicite que c_1 et c_2 sont disjoints, mais que ces deux concepts sont détectés comme disjoints par un moteur d'inférences. Cette notation est utilisée pour remplacer $c_2 \cap c_3 \ \bot$

L'anti-patron « UniversalExistence »

$$c_1 \subseteq \forall r.c_2,\ c_1 \subseteq \exists r.c_3\ ,\ disj(c_2, c_3)$$

Le concepteur de l'ontologie ajoute une restriction existentielle (universelle) à une classe en oubliant qu'il existe déjà une restriction universelle (existentielle) dans cette classe ou l'une de ses classes parentes.

A.2.2 Les anti-patrons cognitifs

Les anti-patrons cognitifs sont des expressions contenant des erreurs de modélisation dûes à une méconnaissance des axiomes logiques.

L'anti-patron « SynonymeOfEquivalence »

$$c_1 \equiv c_2$$

Le concepteur de l'ontologie souhaite exprimer que deux classes c_1 et c_2 sont identiques. Mais, dans une même ontologie, cette relation d'équivalence n'est pas utile, surtout quand l'une des deux classes n'est pas utilisée dans les autres axiomes de l'ontologie.

L'anti-patron « SumOfSome »

$$c_1 \subseteq \exists r.c_2,\ c_1 \subseteq \exists r.c_3,\ disj(c_2, c_3)$$

Le concepteur de l'ontologie a ajouté une nouvelle restriction existentielle sans se rappeler qu'il a défini une autre restriction pour le même concept et le même rôle. Bien que parfois cela peut être correct (e.g. un enfant a au moins une mère et au moins un père), cela représente souvent une erreur de conception.

L'anti-patron « SomeMeansAtLeastOne »

$$c_1 \subseteq \exists r.c_2,\ c_1 \subseteq (\geq 1 r.T)$$

La restriction de cardinalité n'est pas nécessaire.

A.2.3 Les conseils

Les conseils sont des expressions complexes qui sont correctes d'un point de vue logique et cognitif mais qui peuvent être exprimées de façon plus simple et précise.

Le conseil « DisjointnessOfComplement »

$$c_1 \equiv not\ c_2$$

Le concepteur de l'ontologie veut exprimer que c_1 et c_2 ne partagent aucune instance commune. Pour ce faire, il serait plus approprié de définir que c_1 et c_2 sont disjoints au lieu de définir que c_1 est le complément de c_2. De plus, il est difficile de définir le complément d'une classe au cours de la construction d'une ontologie car le nombre de classes n'est pas encore figé. Par exemple, le concepteur pourra avoir besoin de créer une nouvelle classe c_3, disjointe de c_1 et de c_2.

Le conseil « Domain&CardinalityConstraints »

$$c_1 \subseteq \exists r.c_2,\ c_1 \subseteq (= 2r.T),$$

Le concepteur de l'ontologie n'a pas compris que « only » n'implique pas « some ». Il a oublié qu'une restriction existentielle contient une contrainte de cardinalité.

Le conseil « GroupAxioms »

$$c_1 \subseteq \forall r.c_2,\ c_1 \subseteq (= 2r.T),$$

Pour faciliter la compréhension de définitions complexes de classes il est préférable de regrouper toutes les restrictions d'une classe qui utilisent la même relation dans une seule restriction.

Le conseil « MinIsZero »

$$c_1 \subseteq (\geq 0r.T),$$

Le concepteur de l'ontologie veut exprimer que c_1 peut être le domaine de la relation r. Cette restriction n'est pas nécessaire.

B

Extrait [1] du code OWL décrivant l'ontologie construite manuellement O_{mb}

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax
      -ns#"
    xmlns:protege="http://protege.stanford.edu/plugins
      /owl/protege#"
    xmlns:xsp="http://www.owl-ontologies.com
      /2005/08/07/xsp.owl#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns="http://www.owl-ontologies.com/
      Ontology1289983605.owl#"
    xmlns:swrl="http://www.w3.org/2003/11/swrl#"
    xmlns:swrlb="http://www.w3.org/2003/11/swrlb#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://www.owl-ontologies.com/
      Ontology1289983605.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="Adjective_noun_phrase">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Noun_phrase"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Brother_node">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Node"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Taxonomic_relation_identification
      ">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Ontology_building_phase"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Unifying_conceptual_framework">
    <owl:equivalentClass>
      <owl:Class rdf:ID="Shared_understanding"/>
    </owl:equivalentClass>
  </owl:Class>
  <owl:Class rdf:ID="Similarity">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Property"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Object_class"/>
  <owl:Class rdf:ID="
      Terminological_confusion_elimination">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="
      Conceptual_confusion_elimination"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Generic_concepts_pruning"/>
  <owl:Class rdf:ID="Semantic_relation_type"/>
  <owl:Class rdf:ID="Axiom"/>
  <owl:Class rdf:ID="Automatic_graph_building">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Process"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Specificity_measure">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Measure"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Grammar"/>
  <owl:Class rdf:ID="Concept_adding">
    <rdfs:subClassOf rdf:resource="#Process"/>
  </owl:Class>
  <owl:Class rdf:ID="System"/>
  <owl:Class rdf:ID="Tagging_model">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Model"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Warehouse"/>
  <owl:Class rdf:ID="User_community">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Community"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Natural_language_definition">
    <owl:equivalentClass>
      <owl:Class rdf:ID="Textual_definition"/>
```

1. L'intégralité du code OWL décrivant l'ontologie construite manuellement est disponible sur www.tgherasim.com.

```
    </owl:equivalentClass>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Definition"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Complex_meaning">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Meaning"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Approach"/>
  <owl:Class rdf:ID="Systematic_definition">
    <rdfs:subClassOf rdf:resource="#Definition"/>
  </owl:Class>
  <owl:Class rdf:ID="Inductive_learner">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Learner"/>
    </rdfs:subClassOf>
    <owl:equivalentClass>
      <owl:Class rdf:ID="Inductive_learning_program"/>
    </owl:equivalentClass>
  </owl:Class>
  <owl:Class rdf:ID="Binary_feature_representation">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Binary_representation"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Terminological_string">
    <owl:equivalentClass>
      <owl:Class rdf:ID="Term"/>
    </owl:equivalentClass>
  </owl:Class>
  <owl:Class rdf:ID="Ontology_learning_architecture">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Architecture"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Semantic_relation">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Relation"/>
    </rdfs:subClassOf>
    <owl:equivalentClass>
      <owl:Class rdf:ID="Semantic_interrelationship"/>
    </owl:equivalentClass>
  </owl:Class>
  <owl:Class rdf:ID="Experiment"/>
  <owl:Class rdf:ID="Domain_label"/>
  <owl:Class rdf:ID="Sense_inventory">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Inventory"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Conceptual_relation">
    <rdfs:subClassOf rdf:resource="#Relation"/>
  </owl:Class>
  <owl:Class rdf:ID="Context_free_grammar">
    <rdfs:subClassOf rdf:resource="#Grammar"/>
  </owl:Class>
  <owl:Class rdf:ID="Semi-automatic_methodology"/>
  <owl:Class rdf:ID="Hyponymy_relation">
    <rdfs:subClassOf rdf:resource="#Semantic_relation
      "/>
  </owl:Class>
  <owl:Class rdf:ID="Stop_word_list"/>
  <owl:Class rdf:ID="Ontology_enrichment">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Ontology_building_phase
      "/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Shared_understanding">
    <owl:equivalentClass rdf:resource="#
      Unifying_conceptual_framework"/>
  </owl:Class>
  <owl:Class rdf:ID="Attribute_relation">
    <rdfs:subClassOf rdf:resource="#Semantic_relation
      "/>
  </owl:Class>
  <owl:Class rdf:ID="Set"/>
  <owl:Class rdf:about="#Semantic_interrelationship">
    <owl:equivalentClass rdf:resource="#
      Semantic_relation"/>
  </owl:Class>
  <owl:Class rdf:ID="Domain_community">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Group_of_people"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Semantic_disambiguation">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Ontology_building_phase
      "/>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Disambiguation"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Step"/>
  <owl:Class rdf:ID="Document_warehouse">
    <rdfs:subClassOf rdf:resource="#Warehouse"/>
  </owl:Class>
  <owl:Class rdf:ID="Hierarchical_fashion_arrangement
      ">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Ontology_building_phase
      "/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Community_representative"/>
  <owl:Class rdf:ID="Concept_component"/>
  <owl:Class rdf:ID="Classification_task">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Task"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Word_sense"/>
  <owl:Class rdf:ID="Relation_set">
    <rdfs:subClassOf rdf:resource="#Set"/>
  </owl:Class>
  <owl:Class rdf:ID="Example"/>
  <owl:Class rdf:ID="Concept_tagging">
    <rdfs:subClassOf rdf:resource="#Process"/>
  </owl:Class>
  <owl:Class rdf:ID="Gloss">
    <rdfs:subClassOf rdf:resource="#Definition"/>
    <owl:equivalentClass>
      <owl:Class rdf:ID="Textual_sense_definition"/>
    </owl:equivalentClass>
  </owl:Class>
  <owl:Class rdf:ID="Task_of_identifying_relations">
    <rdfs:subClassOf rdf:resource="#Process"/>
    <rdfs:subClassOf rdf:resource="#Task"/>
  </owl:Class>
  <owl:Class rdf:ID="Similarity_measure">
    <rdfs:subClassOf rdf:resource="#Measure"/>
  </owl:Class>
  <owl:Class rdf:ID="General_model">
    <rdfs:subClassOf rdf:resource="#Model"/>
  </owl:Class>
  <owl:Class rdf:ID="Graph_representation_creation"/>
  <owl:Class rdf:ID="Knowledge_repository">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Resource"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Object_classification"/>
  <owl:Class rdf:ID="Inductive_learning">
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Learning"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Probabilistic_learner">
    <rdfs:subClassOf>
      <owl:Class rdf:about="#Learner"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Syntactic-semantic_relation">
    <rdfs:subClassOf rdf:resource="#Relation"/>
  </owl:Class>
  <owl:Class rdf:ID="Domain_concept_forest"/>
  <owl:Class rdf:ID="
      Terminological_expression_capturing"/>

  ...

</rdf:RDF>
```

Bibliographie

[1] Abel, M., T. Bach, S. Dehors, R. Dieng-Kuntz, F. Gandon, P. Luong et C. Moulin. 2005, «Ontologies pour le web sémantique et pour le e-learning», dans *Actes des Journée thématique : Web Sémantique pour le E-learning, Plateforme AFIA*, p. 44–96. (Cité en page 92.)

[2] Aime, X., F. Dhombres, F. Fürst, P. Kuntz, F. Trichet et J. Charlet. 2012, «Rare diseases knowledge management : the contribution of proximity measurements in OntoOrpha and OMIM», dans *Studies in Health Technology and Informatics. Quality of Life through Quality of Information – Proceedings of the 24th European Medical Informatics Conference*, vol. 180, IOS Press, p. 88–92. (Cité en page 2.)

[3] Akif, J.-C., S. Cazajous, Y. Ducq, M. Grandin-Dubost, C. Lero et J.-P. Lorre. 2013, «Rapport commun de fin de projet, Livrable final du projet ISTA3 (Interopérabilité de 3ème génération pour la Sous-traitance dans l'Aéronautique)», Rapport technique, InterOP-VLab (http ://interop-vlab.eu/the-scientific-activities/research-projects-in-i-vlab/grand-sud-ouest-france-pole). (Cité en page 3.)

[4] Al-Harbi, O., S. Jusoh et N. Norwawi. 2012, «Handling ambiguity problems of natural language interface for question answering», *International Journal of Computer Science Issues (IJCSI)*, vol. 9(3), IJCSI Publications, p. 245–265. (Cité en page 113.)

[5] Almeida, M. 2009, «A proposal to evaluate ontology content», *Journal of Applied Ontology*, vol. 4(3-4), IOS Press, p. 245–265. (Cité en pages 63, 64 et 69.)

[6] Aussenac-Gilles, N., B. Biébow et S. Szulman. 2000, «Revisiting ontology design : a method based on corpus analysis», dans *Proceedings of the 12th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2000)*, *Lecture Notes in Computer Science*, vol. 1937, Springer, p. 172–188. (Cité en page 34.)

[7] Aussenac-Gilles, N., S. Despres et S. Szulman. 2008, «The TERMINAE Method and Platform for Ontology Engineering from Texts», dans *Proceedings of the 2008 Conference on Ontology Learning and Population : Bridging the Gap between Text and Knowledge*, IOS Press, p. 199–223. (Cité en page 34.)

[8] Baader, F., D. Calvanese, D. McGuinness, D. Nardi et P. Patel-Schneider, éd. 2003, *The description logic handbook : theory, implementation, and applications*, Cambridge University Press. (Cité en page 99.)

[9] Baeza-Yates, R. et B. Ribeiro-Neto. 2011, *Modern information retrieval : the concepts and technology behind search*, 2e éd., Addison-Wesley. (Cité en page 114.)

[10] Baumeister, J. et D. Seipel. 2005, «Smelly owls - Design anomalies in ontologies», dans *Proceedings of the 18th International Florida Artificial Intelligence Research Society Conference (FLAIRS 2005)*, AAAI Press, p. 215–220. (Cité en pages 69 et 70.)

[11] Baumeister, J. et D. Seipel. 2010, «Anomalies in ontologies with rules», *Web Semantics : Science, Services and Agents on the World Wide Web*, vol. 8(1), Elsevier, p. 55–68. (Cité en pages 2, 69 et 70.)

[12] Bertini, E. et D. Lalanne. 2009, «Surveying the complementary role of automatic data analysis and visualization in knowledge discovery», dans *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery : Integrating Automated Analysis with Interactive Exploration (VAKD 2009)*, ACM, p. 12–20. (Cité en page 114.)

[13] Blomqvist, E. 2009, *Semi-automatic Ontology Construction based on Patterns*, Thèse de doctorat, Institute of Technology of Linkoping University, Sweden. (Cité en pages 2, 9, 30, 32 et 42.)

[14] Borst, W. 1997, *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*, Thèse de doctorat, Dutch research school for Information and Knowledge Systems (SIKS), Universiteit Twente, Netherlands. (Cité en page 9.)

[15] Brank, J., M. Grobelnik et D. Mladenic. 2005, «A survey of ontology evaluation techniques», dans *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005) at the 7th International Multi-conference on Information Society 2005*, p. 166–170. (Cité en pages 63 et 69.)

[16] Brank, J., D. Mladenic et M. Grobelnik. 2006, «Gold standard based ontology evaluation using instance assignment», dans *Proceedings of the 4th International Workshop on Evaluation of Ontologies for the Web (EON 2006) at the 15th International World Wide Web Conference (WWW 2006)*, vol. 179, CEUR Workshop Proceedings, p. 54–61. (Cité en page 69.)

[17] Brewster, C., H. Alani, S. Dasmahapatra et Y. Wilks. 2004, «Data driven ontology evaluation», dans *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, European Language Resources Association, p. 641–644. (Cité en page 69.)

[18] Budanitsky, A. 1999, «Lexical semantic relatedness and its application in natural language processing», Rapport technique, University of Toronto, Canada. (Cité en page 11.)

[19] Buhmann, L., S. Danielczyk et J. Lehmann. 2011, «D3.4.1 Report on relevant automatically detectable modelling errors and problems», Rapport technique, LOD2 - Creating Knowledge out of Interlinked Data. (Cité en pages 2, 69, 70, 71, 73, 98 et 99.)

[20] Buitelaar, P., P. Cimiano et B. Magnini. 2005, *Ontology Learning from Text : Methods, Applications and Evaluation*, chap. Ontology learning from text : an overview, IOS Press, p. 3–12. (Cité en page 33.)

[21] Buitelaar, P., D. Olejnik et M. Sintek. 2004, «A Protege plug-in for ontology extraction from text based on linguistic analysis», dans *Proceedings of the 1st European Semantic Web Symposium (ESWS)*, *Lecture Notes in Computer Science*, vol. 3053, Springer, p. 31–44. (Cité en page 38.)

[22] Burton-Jones, A., V. Storey et V. Sugumaran. 2005, «A semiotic metrics suite for assessing the quality of ontologies», *Data & Knowledge Engineering*, vol. 55 (1), Elsevier, p. 84–102. (Cité en pages 64 et 101.)

[23] Casellas, N. 2011, *Legal Ontology Engineering – Methodologies, Modelling Trends, and the Ontology of Professional Judicial Knowledge*, *Law, Governance and Technology Series*, vol. 3, Springer. (Cité en page 2.)

[24] Charlet, J. 2002, *L'ingénierie des connaissances : développements, résultats et perspectives pour la gestion des connaissances médicales. Mémoire d'habilitation à diriger les recherches*, Université Pierre et Marie Curie, Paris, France. (Cité en page 9.)

[25] Cimiano, P. 2006, *Ontology Learning and Population from Text : Algorithms, Evaluation and Applications*, Springer. (Cité en pages 10 et 13.)

[26] Cimiano, P., A. Hotho et S. Staab. 2004, «Clustering concept hierarchies from text», dans *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC 2004)*, European Language Resources Association, p. 1721–1724. (Cité en page 69.)

[27] Cimiano, P. et J. Volker. 2005, «Text2Onto : a framework for ontology learning and data-driven change discovery», dans *Proceedings of the 10th International Conference on Natural Language Processing and Information Systems (NLDB 2005)*, *Lecture Notes in Computer Science*, vol. 3513, Springer, p. 227–238. (Cité en pages 30, 35 et 46.)

[28] Corcho, O., M. Fernandez-Lopez, A. Gomez-Perez et A. Lopez-Cima. 2005, «Building Legal Ontologies with METHONTOLOGY and WebODE», dans *Law and the Semantic Web*, *Lecture Notes in Computer Science*, vol. 3369, Springer, p. 142–157. (Cité en pages 23 et 26.)

[29] Corcho, O., A. Gomez-Perez, R. Gonzalez-Cabero et C. Suarez-Figueroa. 2004, «ODEval : a tool for evaluating RDF(S), DAML+OIL, and OWL concept taxonomies», dans *Proceedings of the 1st IFIP Conference on Artificial Intelligence Applications and Innovations (AIAI 2004)*, *Artificial Intelligence Applications and Innovations*, vol. 154, Springer, p. 369–382. (Cité en page 99.)

[30] Corcho, O., C. Roussey et L. V. Blazquez. 2009, «Catalogue of anti-patterns for formal ontology debugging», dans *Atelier Construction d'ontologies : vers un guide des bonnes pratiques, AFIA 2009*, p. 2–12. (Cité en page 69.)

[31] Cunningham, H., D. Maynard, K. Bontcheva et V. Tablan. 2002, «GATE : A framework and graphical development environment for robust NLP tools and applications», dans *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics*, ACM, p. 168–175. (Cité en page 35.)

[32] Daelemans, W. et M.-L. Reinberger. 2004, «Shallow text understanding for ontology content evaluation», *IEEE Intelligent Systems*, vol. 19, n° 4, p. 74–81. (Cité en page 69.)

[33] Deborah, L. J., R. Baskaran et A. Kannan. 2011, «Ontology construction using computational linguistics for e-learning», dans *Proceedings of the 2nd International Conference on Visual informatics : sustaining research and innovations*, *Lecture Notes in Computer Science*, vol. 7067, Springer, p. 50–63. (Cité en page 2.)

[34] Dellschaft, K. et S. Staab. 2008, «Strategies for the evaluation of ontology learning», dans *Proceedings of the 2008 Conference on Ontology Learning and Population : Bridging the Gap between Text and Knowledge*, *Frontiers in Artificial Intelligence and Applications*, vol. 167, IOS Press, p. 253–272. (Cité en page 69.)

[35] DeMaio, C., G. Fenza, V. Loia et S. Senatore. 2012, «Hierarchical web resources retrieval by exploiting fuzzy formal concept analysis», *International Journal of Information Processing and Management*, vol. 48(3), Elsevier, p. 399–418. (Cité en page 113.)

[36] Dixit, P., S. Sethi, A. Sharma et A. Dixit. 2012, «Design of an automatic ontology construction mechanism using semantic analysis of the documents», dans *Proceedings of the 4th International Conference on Computational Intelligence and Communication Networks (CICN 2012)*, IEEE, p. 611–616. (Cité en page 2.)

[37] Duque-Ramos, A., J. Fernandez-Breis, N. Aussenac-Gilles et R. Stevens. 2011, «OQuaRE : A SQuaRE-based Approach for Evaluating the Quality of Ontologies», *Journal of Research and Practice in Information Technology*, vol. 43(2), Australian Computer Society Inc., p. 159–173. (Cité en pages 2, 63 et 64.)

[38] Ehrig, M., P. Haase, N. Stojanovic et M. Hefke. 2005, «Similarity for ontologies - a comprehensive framework», dans *Proceedings of the 13th European Conference on Information Systems, Information Systems in a Rapidly Changing Economy (ECIS 2005)*, p. 1509–1518. (Cité en page 13.)

[39] Fahad, M. et M. Qadir. 2008, «A framework for ontology evaluation», dans *Supplementary Proceedings of the 16th International Conference on Conceptual Structures (ICCS 2008)*, vol. 354, CEUR Workshop Proceedings, p. 149–158. (Cité en page 69.)

[40] Fang, J., L. Guo et N. Yang. 2010, «Handling polysemy in description logic ontologies», dans *Proceedings of 7th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2010)*, vol. 4, IEEE, p. 1793–1797. (Cité en page 102.)

[41] Faure, D. et C. Nédellec. 1998, «A corpus-based conceptual clustering method for verb frames and ontology acquisition», dans *LREC workshop on Adapting lexical and*

corpus ressources to sublanguages and applications, édité par P. Velardi, European Language Resources Association, p. 5–12. (Cité en page 37.)

[42] Faure, D., C. Nédellec et C. Rouveirol. 1998, «Acquisition of semantic knowledge using machine learning methods : The system ASIUM», Rapport technique ICS-TR-88-16, Universite Paris Sud. (Cité en page 37.)

[43] Feigenbaum, E. et P. McCorduck. 1983, *The fifth generation - artificial intelligence and Japan's computer challenge to the world*, 1re éd., Addison-Wesley. (Cité en page 8.)

[44] Fernandez, M., A. Gomez-Perez et N. Juristo. 1997, «Methontology : From ontological art towards ontological engineering», dans *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, p. 33–40. (Cité en pages 4 et 23.)

[45] Fernandez-Breis, J., M. Egana-Aranguren et R. Stevens. 2009, «A quality evaluation framework for bio-ontologies», dans *Proceedings of the 2009 International Conference on Biomedical Ontology (ICBO 2009)*, University at Buffalo, NY : Nature Precedings, p. 136–139. (Cité en page 64.)

[46] Fortuna, B., M. Grobelnik et D. Mladenic. 2006, «Semi-automatic data driven ontology construction system», dans *Proceedings of the 9th International multiconference Information Society IS-2006*, p. 223–226. (Cité en page 38.)

[47] Fortuna, B., M. Grobelnik et D. Mladenic. 2007, «Ontogen : Semi-automatic ontology editor», dans *Proceedings of the 2007 Conference on Human interface (HCI)*, *Lecture Notes in Computer Science*, vol. 4558, Springer, p. 309–318. (Cité en page 38.)

[48] Gangemi, A. 2013, «A comparison of knowledge extraction tools for the semantic web», dans *Proceedings of the 10th European Semantic Web Conference (ESWC'2013)*, *Lecture Notes in Computer Science*, vol. 7882, Springer, p. 351–366. (Cité en page 113.)

[49] Gangemi, A., C. Catenacci, M. Ciaramita et J. Lehmann. 2006, «Modelling ontology evaluation and validation», dans *Proceedings of the 3rd European Semantic Web Conference (ESWC'2006)*, *Lecture Notes in Computer Science*, vol. 4011, Springer, p. 140–154. (Cité en pages 63, 64, 65, 67, 68, 69 et 73.)

[50] Gangemi, A. et V. Presutti. 2009, «Ontology design patterns», dans *Handbook on Ontologies*, édité par R. Studer et S. Staab, 2e éd., International Handbooks on Information Systems, Springer, p. 221–243. (Cité en page 71.)

[51] Genesereth, M. et N. Nilsson. 1987, *Logical Foundations of Artificial Intelligence*, Morgan Kaufmann Publishers. (Cité en page 14.)

[52] Gherasim, T., M. Harzallah, G. Berio et P. Kuntz. 2011, «Analyse comparative de methodologies et d'outils de construction automatique d'ontologies a partir de ressources textuelles», dans *Proceedings of EGC'2011*, p. 377–388. (Cité en page 54.)

[53] Gomez-Perez, A. 1995, «Some ideas and examples to evaluate ontologies», dans *Proceedings of the 11th Conference on Artificial Intelligence for Applications (CAIA 1995)*, IEEE, p. 299–305. (Cité en pages 2 et 63.)

[54] Gomez-Perez, A. 2004, «Ontology evaluation», dans *Handbook on Ontologies*, édité par S. Staab et R. Studer, 1re éd., International Handbooks on Information Systems, Springer, p. 251–274. (Cité en pages 2, 63, 64, 69 et 70.)

[55] Gomez-Perez, A., M. Fernandez-Lopez et O. Corcho. 2001, *Ontological Engineering : With Examples from the Areas of Knowledge Management, E-Commerce and the Semantic Web*, Advanced Information and Knowledge Processing, Springer. (Cité en pages 22, 23, 24, 25, 26, 69, 70 et 72.)

[56] Gonçalves, R. S., B. Parsia et U. Sattler. 2012, «Performance heterogeneity and approximate reasoning in description logic ontologies», dans *Proceedings of the 11th international conference on The Semantic Web (ISWC 2012)*, *Lecture Notes in Computer Science*, vol. 7649, Springer, p. 82–98. (Cité en page 99.)

[57] Gruber, T. 1993, «A translation approach to portable ontology specifications», *Knowledge Acquisition - Special issue : Current issues in knowledge modeling*, vol. 5(2), Academic Press Ltd., p. 199–220. (Cité en pages 2, 9, 14 et 63.)

[58] Gruninger, M. et M. Fox. 1995, «Methodology for the design and evaluation of ontologies», dans *Proceedings of the Workshop on Basic Ontological Issues in Knowledge Sharing (held in conjunction with IJCAI'95)*, p. 1–10. (Cité en page 63.)

[59] Guarino, N. 1998, «Formal ontology in information systems», dans *Proceedings of the 1st International Conference on Formal Ontologies in Information Systems (FOIS 1998)*, *Frontiers in Artificial Intelligence and Applications*, vol. 46, IOS Press, p. 3–15. (Cité en page 13.)

[60] Guarino, N., D. Oberle et S. Staab. 2009, «What is an ontology ?», dans *Handbook on Ontologies*, édité par R. Studer et S. Staab, 2e éd., International Handbooks on Information Systems, Springer, p. 1–17. (Cité en pages 5, 8, 9, 13, 14, 15, 19 et 73.)

[61] Guarino, N. et C. Welty. 2004, «An overview of Ontoclean», dans *Handbook on Ontologies*, édité par S. Staab et R. Studer, 1re éd., International Handbooks on Information Systems, Springer, p. 151–159. (Cité en pages 63 et 65.)

[62] Haarslev, V. et R. Möller. 2001, «RACER System Description», dans *Proceedings of the 1st International Joint Conference on Automated Reasoning (IJCAR 2001)*, *Lecture Notes in Computer Science*, vol. 2083, Springer, p. 701–706. (Cité en page 98.)

[63] Harris, Z. 1968, *Mathematical Structures of Language*, J. Wiley and Son. (Cité en page 34.)

[64] Hartmann, J., P. Spyns, A. Giboin, D. Maynard, R. Cuel, M. C. Suarez-Figueroa et Y. Sure. 2004, «Methods for ontology evaluation», Rapport technique, Knowledge Web Deliverable D1.2.3, v. 0. (Cité en pages 2 et 63.)

[65] Harzallah, M. 2012, «Conception des ontologies pour l'interopérabilité, Livrable L2.4 du projet ISTA3 (Interopérabilité de 3ème génération pour la sous-traitance dans l'Aéronautique)», Rapport technique, InterOP-VLab (http ://interop-vlab.eu/the-scientific-activities/research-projects-in-i-vlab/grand-sud-ouest-france-pole). (Cité en page 3.)

[66] Hernandez, N. 2005, *Ontologies de domaine pour la modélisation du contexte en recherche d'information*, Thèse de doctorat, Université de Toulouse, France. (Cité en pages 8 et 9.)

[67] Ji, Q., P. Haase, G. Qi, P. Hitzler et S. Stadtmüller. 2009, «RaDON – Repair and Diagnosis in Ontology Networks», dans *Proceedings of the 6th European Semantic Web Conference on The Semantic Web : Research and Applications (ESWC 2009)*, *Lecture Notes in Computer Science*, vol. 5554, Springer, p. 863–867. (Cité en page 99.)

[68] Kalfoglou, Y. 2010, *Cases on Semantic Interoperability for Information Systems Integration : Practices and Applications*, IGI Global. (Cité en page 101.)

[69] Kalyanpur, A., B. Parsia, E. Sirin et B. Cuenca-Grau. 2006, «Repairing unsatisfiable concepts in owl ontologies», dans *Proceedings of the 3rd European Conference on The Semantic Web : research and applications (ESWC'2006)*, *Lecture Notes in Computer Science*, vol. 4011, Springer, p. 170–184. (Cité en page 99.)

[70] Keim, D., F. Mansmann et J. Thomas. 2009, «Visual analytics : how much visualization and how much analytics ?», *ACM SIGKDD Explorations Newsletter*, vol. 11(2), p. 5–8. (Cité en page 114.)

[71] Klein, M. 2004, *Change Management for Distributed Ontologies*, Thèse de doctorat, University of Amsterdam, Netherlands. (Cité en page 113.)

[72] Krogstie, J. et H. Jorgensen. 2002, «Quality of interactive models», dans *Conceptual Modeling-ER 2002, Workshops of the 21st International Conference on Conceptual Modeling*, *Lecture Notes in Computer Science*, vol. 2784, Springer, p. 351–363. (Cité en page 73.)

[73] Krogstie, J., O. Lindland et G. Sindre. 1995, «Defining quality aspects for conceptual models», dans *Proceedings of the IFIP8.1 Working Conference on Information Systems Concepts : Towards a Consolidation of Views (ISCO3)*, p. 216–231. (Cité en page 73.)

[74] Lame, G. 2002, *Construction d'ontologie à partir de texte, une ontologie du droit dédiée à la recherche d'information sur le Web, Thèse de doctorat*, Thèse de doctorat, Ecole des Mines de Paris, France. (Cité en page 9.)

[75] Lin, H. et E. Sirin. 2008, «Pellint - A Performance Lint Tool for Pellet», dans *Proceedings of the 5th OWLED Workshop on OWL : Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC 2008)*, vol. 432, CEUR Workshop Proceedings, p. 1–4. (Cité en pages 71 et 99.)

[76] Lozano-Tello, A. et A. Gomez-Perez. 2004, «Ontometric : A method to choose the appropriate ontology», *Journal of Database Management*, vol. 15(2), IGI Global, p. 1–18. (Cité en pages 63, 65 et 69.)

[77] Lundqvist, K. O., K. Baker et S. Williams. 2011, «Ontology supported competency system», *International Journal of Knowledge and Learning*, vol. 7(3-4), Inderscience, p. 197–219. (Cité en page 2.)

[78] Maedche, A., B. Motik et L. Stojanovic. 2003, «Managing multiple and distributed ontologies on the Semantic Web», *International Journal on Very Large Data Bases*, vol. 12(4), Springer, p. 286–302. (Cité en page 113.)

[79] Maedche, A. et S. Staab. 2000, «Discovering conceptual relations from text», dans *Proceedings of the 14th European Conference on Artificial Intelligence (ECAI 2000)*, *Frontiers in Artificial Intelligence and Applications*, vol. 215, IOS Press, p. 321–325. (Cité en page 35.)

[80] Maedche, A. et S. Staab. 2001, «Ontology learning for the semantic web», *IEEE Intelligent Systems*, vol. 16(2), p. 72–79. (Cité en page 22.)

[81] Maedche, A. et S. Staab. 2002, «Measuring similarity between ontologies», dans *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2002)*, *Lecture Notes in Computer Science*, vol. 2473, Springer, p. 251–263. (Cité en page 69.)

[82] Maedche, A. et R. Volz. 2001, «The ontology extraction & maintenance framework Text-To-Onto», dans *Proceedings of the Workshop on Integrating Data Mining and Knowledge Management at the 2001 IEEE International Conference on Data Mining (ICDM 2001)*, p. 1–12. (Cité en page 35.)

[83] Maedche, A. et V. Zacharias. 2002, «Clustering ontology-based metadata in the Semantic Web», dans *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD 2002)*, *Lecture Notes in Computer Science*, vol. 2431, Springer, p. 348–360. (Cité en page 65.)

[84] Magnini, B. et G. Cavaglia. 2000, «Integrating subject field codes into WordNet», dans *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC 2000)*, European Language Resources Association, p. 1413–1418. (Cité en page 113.)

[85] Maynard, D., A. Funk et W. Peters. 2009, «SPRAT : a tool for automatic semantic pattern-based ontology population», dans *Proceedings of the International Conference for Digital Libraries and the Semantic Web*, p. 1–15. (Cité en pages 2 et 36.)

[86] Maynard, D., W. Peters et Y. Li. 2006, «Metrics for evaluation of ontology-based information extraction», dans *Proceedings of the 4th International Workshop on Evaluation of Ontologies for the Web (EON 2006)*, vol. 179, CEUR Workshop Proceedings, p. 65–72. (Cité en page 69.)

[87] Mondary, T. 2011, *Construction d'ontologies à partir de textes. L'apport de l'analyse de concepts formels*, Thèse de doctorat, Université Paris-Nord - Paris XIII, France. (Cité en pages 2, 11, 35 et 112.)

[88] Motik, B., R. Shearer et I. Horrocks. 2009, «Hypertableau reasoning for description logics», *Journal of Artificial Intelligence Research*, vol. 36(1), p. 165–228. (Cité en page 98.)

[89] Mustapha, N. B., H. B. Zghal, M. Aufaure et H. B. Ghezala. 2009, «Ontology learning from web : survey and framework based on semantic search», dans *Proceedings of 2nd International Conference on Web and Information Technologies (ICWIT 2009)*, p. 217–231. (Cité en page 2.)

[90] Navigli, R. 2009, «Word sense disambiguation : A survey», *ACM Computing Surveys (CSUR)*, vol. 41 (2), p. 1–69. (Cité en page 102.)

[91] Navigli, R. et P. Velardi. 2004, «Learning domain ontologies from document warehouses and dedicated web sites», *Computational Linguistics*, vol. 30(2), p. 151–179. (Cité en page 54.)

[92] Navigli, R. et P. Velardi. 2005, «Structural semantic interconnections : A knowledge-based approach to word sense disambiguation», *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, vol. 27(7), p. 1075–1086. (Cité en page 59.)

[93] Navigli, R., P. Velardi, A. Cucchiarelli et F. Neri. 2004, «Quantitative and qualitative evaluation of the OntoLearn ontology learning system», dans *Proceedings of the 20th International Conference on Computational Linguistics (COLING 2004)*, Association for Computational Linguistics, p. 1043–1050. (Cité en page 69.)

[94] Navigli, R., P. Velardi et A. Gangemi. 2003, «Ontology learning and its application to automated terminology translation», *IEEE Intelligent Systems*, vol. 18(1), p. 22–31. (Cité en page 37.)

[95] Nazarenko, A. et T. Hamon. 2002, «Structuration de terminologie : quels outils pour quelles pratiques ?», *Traitement automatique des langues. Structuration de terminologie*, vol. 43(1), p. 7–18. (Cité en page 33.)

[96] Nédellec, C. 2007, «Acquisition of relation extraction rules by machine learning», Rapport technique, Deliverable 6.4b for ALVIS (Superpeer semantic Search Engine) Project. (Cité en page 37.)

[97] Nédellec, C. et D. Faure. 1999, «Knowledge acquisition of predicate argument structures from technical texts using machine learning : The system ASIUM», dans *Proceedings of the 11th European Workshop on Knowledge Acquisition, Modeling and Management (EKAW 1999)*, *Lecture Notes in Computer Science*, vol. 1621, Springer, p. 329–334. (Cité en page 37.)

[98] Nonaka, I. et H. Takeuchi. 1995, *The knowledge-creating company : How japanese companies create the dynamics of innovation*, Oxford University Press. (Cité en page 9.)

[99] Noy, N., S. Kunnatur, M. Klein et M. Musen. 2004, «Tracking changes during ontology evolution», dans *Proceedings of the 3rd International Semantic Web Conference (ISWC 2004)*, *Lecture Notes in Computer Science*, vol. 3298, Springer, p. 259–273. (Cité en page 113.)

[100] Obrst, L., B. Ashpole, W. Ceusters, I. Mani, S. Ray et B. Smith. 2007, «The evaluation of ontologies : toward improved semantic interoperability», dans *SemanticWeb : Revolutionizing Knowledge Discovery in the Life Sciences*, édité par C. J. O. Baker et K.-H. Cheung, Springer, p. 139–158. (Cité en page 64.)

[101] Palma, R., F., Zablith, P. Haase et O. Corcho. 2012, «Ontology evolution», dans *Ontology Engineering in a Networked World*, édité par M. Suarez-Figueroa, A. Gomez-Perez, E. Motta et A. Gangemi, Springer, p. 235–255. (Cité en page 113.)

[102] Park, J., W. Cho et S. Rho. 2011, «Evaluating ontology extraction tools using a comprehensive evaluation framework», *Data & Knowledge Engineering*, vol. 69, Elsevier, p. 1043–1061. (Cité en pages 43, 44, 45 et 46.)

[103] Pinto, H. S., C. Tempich et S. Staab. 2004, «Diligent : Towards a fine-grained methodology for distributed, loosely-controlled and evolving engineering of ontologies», dans *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, *Frontiers in Artificial Intelligence and Applications*, vol. 110, IOS Press, p. 393–397. (Cité en pages 23 et 28.)

[104] Porzel, R. et R. Malaka. 2004, «A task-based approach for ontology evaluation», dans *Proceedings of the 1st Ontology Learning and Population Workshop (OLP 2004) at the 16th European Conference on Artificial Intelligence (ECAI 2004)*, p. 7–12. (Cité en page 69.)

[105] Potrich, A. et E. Pianta. 2008, «L-isa : Learning domain specific isa-relations from the web», dans *Proceedings of the 6th International Conference on Language Resources and Evaluation (LREC 2008)*, European Language Resources Association, p. 2368–2375. (Cité en page 113.)

[106] Poveda, M., M. Suarez-Figueroa et A. Gomez-Perez. 2009, «Common pitfalls in ontology development», dans *Proceedings of the Current topics in artificial intelligence (CAEPIA 2009), and 13th Conference on Spanish Association for Artificial Intelligence*, *Lecture Notes in Computer Science*, vol. 5988, Springer, p. 91–100. (Cité en pages 69, 70, 72 et 115.)

[107] Poveda, M., M. Suarez-Figueroa et A. Gomez-Perez. 2010, «A double classification of common pitfalls in ontologies», dans *Proceedings of the Workshop on Ontology Quality (OntoQual 2010) at the 17th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2010)*, p. 1–12. (Cité en pages 2, 70, 72, 102, 115, 116, 117 et 118.)

[108] Poveda-Villalon, M., M. Suarez-Figueroa et A. Gomez-Perez. 2012, «Validating ontologies with OOPS !», dans *Proceedings of the 18th international conference on Knowledge Engineering and Knowledge Management (EKAW 2012)*, *Lecture Notes in Computer Science*, vol. 7603, Springer, p. 267–281. (Cité en pages 2, 100, 101 et 102.)

[109] Punuru, J. et J. Chen. 2011, «Learning non-taxonomical semantic relations from domain texts», *Journal of Intelligent Information Systems - Integrating Artificial Intelligence and Database Technologies*, vol. 38 (1), n° 10844, p. 191–207. (Cité en page 2.)

[110] Qasim, I. et S. Khan. 2009, «Semantic mapping between global and source ontology using WordNet», dans *Proceedings of 2nd International Conference on Computer, Control and Communication*, IEEE, p. 1–5. (Cité en pages 102 et 103.)

[111] Reimer, U., E. Maier, S. Streit, T. Diggelmann et M. Hoffleisch. 2011, «Learning a lightweight ontology for semantic retrieval in patient-centered information systems», *International Journal of Knowledge Management*, vol. 7(3), p. 11–26. (Cité en page 113.)

[112] Reniers, D., L. Voinea, O. Ersoy et A. Telea. 2012, «The Solid* toolset for software visual analytics of program structure and metrics comprehension : From research prototype to product», *Journal on Science of Computer Programming, Available online 16 May 2012*, Elsevier. (Cité en page 114.)

[113] Rodriguez-Garcia, M., R. Valencia-Garcia et F. Garcia-Sanchez. 2012, «An ontology evolution-based framework for semantic information retrieval», dans *On the Move to Meaningful Internet Systems : OTM 2012 Workshops*, *Lecture Notes in Computer Science*, vol. 7567, Springer, p. 163–172. (Cité en page 113.)

[114] Roussey, C., O. Corcho et L. V. Blazquez. 2009, «A catalogue of owl ontology antipatterns», dans *Proceedings of the 5th International Conference on Knowledge Capture (K-CAP 2009)*, ACM, p. 205–206. (Cité en page 69.)

[115] Roussey, C., F. Scharffe, O. Corcho et O. Zamazal. 2010, «Une méthode de débogage d'ontologies owl basées sur la détection d'anti-patrons», dans *Actes de la 21e Conférence Ingénierie des Connaissances (IC2010)*, Presses des Mines, p. 43–54. (Cité en pages 71, 115 et 119.)

[116] Sabou, M., C. Wroe, C. Goble et H. Stuckenschmidt. 2005, «Learning domain ontologies for web service descriptions», dans *Proceedings of the 14th International Conference on World Wide Web (WWW'05)*, ACM, p. 190–198. (Cité en page 69.)

[117] Sanchez, D. 2007, *Domain ontology learning from the Web*, Thèse de doctorat, Technical University of Catalonia, Spain. (Cité en page 2.)

[118] Sclano, F. et P. Velardi. 2007, «TermExtractor : a web application to learn the common terminology of interest groups and research communities», dans *Proceedings of 7th Conference on Terminology and Artificial Intelligence (TIA 2007)*, édité par R. Dieng-Kuntz et C. Enguehard, Presses Universitaires de Grenoble, p. 1–10. (Cité en page 33.)

[119] Sellami, Z., V. Camps et N. Aussenac-Gilles. 2013, «DYNAMO-MAS : a multi-agent system for ontology evolution from text», *Journal on Data Semantics*, vol. 2(2-3), Springer, p. 145–161. (Cité en page 113.)

[120] Simperl, E. et C. Tempich. 2009, «Exploring the economical aspects of ontology engineering», dans *Handbook on Ontologies*, édité par R. Studer et S. Staab, 2e éd., International Handbooks on Information Systems, Springer, p. 445–462. (Cité en page 2.)

[121] Sirin, E., B. Parsia, B. Grau, A. Kalyanpur et Y. Katz. 2007, «Pellet : A practical owl-dl reasoner», *Web Semantics : Science, Services and Agents on the World Wide Web*, vol. 5 (2), Elsevier, p. 51–53. (Cité en page 98.)

[122] Spyns, P. 2005, «EvaLexon : assessing triples mined from texts», Rapport technique, Star Lab, Brussels, Belgium. (Cité en page 69.)

[123] Staab, S., H. P. Schnurr, R. Studer et Y. Sure. 2001, «Knowledge processes and ontologies», *IEEE Intelligent Systems*, vol. 16, no 1, p. 26–34. (Cité en pages 26 et 27.)

[124] Stojanovic, L. 2004, *Methods and Tools for Ontology Evolution*, Thèse de doctorat, University of Karlsruhe, Germany. (Cité en page 113.)

[125] Studer, R., V. Benjamins et D. Fensel. 1998, «Knowledge engineering : Principles and methods», *Data & Knowledge Engineering*, vol. 25 (1-2), Elsevier, p. 161–197. (Cité en page 9.)

[126] Stumme, G., M. Ehrig, S. Handschuh, A. Hotho, A. Maedche, B. Motik, D. Oberle, C. Schmitz, S. Staab, L. Stojanovic, N. Stojanovic, R. Studer, Y. Sure, R. Volz et V. Zacharias. 2003, «The Karlsruhe view on ontologies», Rapport technique, University of Karlsruhe, Institute AIFB, Germany. (Cité en pages 5, 8 et 13.)

[127] Suarez-Figueroa, M. 2010, *NeOn Methodology for Building Ontology Networks : Specification, Scheduling and Reuse*, Thèse de doctorat, Universidad Politecnica de Madrid, Spain. (Cité en pages 30 et 31.)

[128] Suarez-Figueroa, M., G. A. de Cea, C. Buil, K. Dellschaft, M. Fernandez-Lopez, A. Garcia, A. Gomez-Perez, G. Herrero, E. Montiel-Ponsoda, M. Sabou, B. Villazon-Terrazas et Z. Yufei. 2008, «NeOn Methodology for Building Contextualized Ontology Networks», Rapport technique, Deliverable D3.3.2 for NeOn Project, http ://www.neon-project.org/nw/Deliverables. (Cité en pages 23, 26 et 29.)

[129] Sy, M.-F., S. Ranwez, J. Montmain, A. Regnault, M. Crampes et V. Ranwez. 2012, «User centered and ontology based information retrieval systems for life sciences», *BMC Bioinformatics*, vol. 13 (Suppl 1), no S4, p. 1–12. (Cité en page 2.)

[130] Tartir, S., I. Arpinar et A. Sheth. 2010, «Ontological evaluation and validation», dans *Theory and Applications of Ontology : Computer Applications*, édité par R. Poli, M. Healy et A. Kameas, Springer, p. 115–130. (Cité en pages 63 et 65.)

[131] Uschold, M. et M. Gruninger. 2004, «Ontologies and semantics for seamless connectivity», *ACM SIGMOD Record*, vol. 33 (4), p. 58–64. (Cité en page 9.)

[132] Velardi, P., R. Navigli, A. Cucchiarelli et F. Neri. 2005, «Evaluation of OntoLearn, a methodology for automatic learning of domain ontologies», dans *Ontology Learning from Text : Methods, Applications and Evaluation*, vol. 123, édité par P. Buitelaar, P. Cimiano et B. Magnini, IOS Press, p. 92–106. (Cité en page 59.)

[133] Velardi, P., R. Navigli et P. D'Amadio. 2008, «Mining the web to create specialized glossaries», *IEEE Intelligent Systems*, vol. 23 (5), p. 18–25. (Cité en page 59.)

[134] Velardi, P., R. Navigli et M. Pétit. 2007, «A taxonomy learning method and its application to characterize a scientific web community», *IEEE Transactions on Knowledge and Data Engineering*, vol. 19(2), p. 180–191. (Cité en pages 2, 37 et 59.)

[135] Volker, J. et Y. Sure. 2006, «Data-driven change discovery - evaluation», Rapport technique, Deliverable D3.3.2 for SEKT Project, Institute AIFB, University of Karlsruhe, Germany. (Cité en page 69.)

[136] Vrandecic, D. 2009, «Ontology evaluation», dans *Handbook on Ontologies*, édité par R. Studer et S. Staab, 2e éd., International Handbooks on Information Systems, Springer, p. 293–314. (Cité en pages 2, 63 et 64.)

[137] Wache, H., T. Vogele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann et S. Hubner. 2001, «Ontology-based integration of information - a survey of existing approaches», dans *Proceedings of the IJCAI-01 Workshop on Ontologies and Information Sharing*, vol. 47, CEUR Workshop Proceedings, p. 108–117. (Cité en page 3.)

[138] Wang, H., M. Horridge, A. Rector, N. Drummond et J. Seidenberg. 2005, «Debugging owl-dl ontologies : A heuristic approach», dans *Proceedings of the 4th International Semantic Web Conference(ISWC-2005)*, *Lecture Notes in Computer Science*, vol. 3729, Springer, p. 745–757. (Cité en pages 98 et 99.)

[139] Welty, C. et N. Guarino. 2001, «Supporting ontological analysis of taxonomic relationships», *Data & Knowledge Engineering*, vol. 39, Elsevier, p. 51–74. (Cité en page 63.)

[140] Welty, C., R. Mahindru et J. Chu-Carroll. 2003, «Evaluating ontological analysis», dans *Proceedings of the Semantic Integration Workshop (SI 2003) at the 2nd International Semantic Web Conference (ISWC 2003)*, vol. 82, CEUR Workshop Proceedings, p. 107–114. (Cité en page 69.)

[141] Wolstencroft, K., R. McEntire, R. Stevens, L. Tabernero et A. Brass. 2005, «Constructing ontology-driven protein family databases», *Bioinformatics*, vol. 21(8), p. 1685–1692. (Cité en page 69.)

[142] Zavitsanos, E., G. Paliouras et G. Vouros. 2008, «A distributional approach to evaluating ontology learning methods using a gold standard», dans *Proceedings of the 3rd Ontology Learning and Population Workshop (OLP 2008) at the 18th European Conference on Artificial Intelligence (ECAI 2008)*, p. 1–6. (Cité en page 69.)

Liste de publications

Liste des travaux présentés par M. GHERASIM Toader David qui a déposé une demande d'autorisation de soutenance de thèse prévue le 30.09.2013 auprès de l'Ecole doctorale STIM (« Sciences et Technologies de l'Information et Mathematiques »)

1 - Chapitre d'ouvrage international (avec comité de lecture)
Gherasim, T., M. Harzallah, G. Berio et P. Kuntz. Methods and tools for automatic construction of ontologies from textual resources : A framework for comparison and its application. Dans Advances in Knowledge Discovery and Management, volume 471, pages 177-201, Springer, 2013.

2 - Publications en congres avec comite de lecture et actes
Gherasim, T., M. Harzallah, G. Berio et P. Kuntz. Analyse comparative de méthodologies et d'outils de construction automatique d'ontologies à partir de ressources textuelles. Dans Actes de la 11ème Conférence Francophone sur l'Extraction et la Gestion des Connaissances, pages 377-388, Cépaduès Edition, 2011 (article sélectionné parmi les 10 meilleurs).

Gherasim, T., G. Berio, M. Harzallah et P. Kuntz. Problems impacting the quality of automatically built ontologies. Dans Proceedings of the 8th Workshop on Knowledge Engineering and Software Engineering (KESE-2012), held in conjuncion with ECAI 2012, volume 949, pages 25-32, CEUR Workshop Proceedings, 2012.

Gherasim, T., G. Berio, M. Harzallah et P. Kuntz. Quality problem identification in automatically constructed ontologies. Dans Proceedings of the Seventh International Conference on Knowledge Capture (K-CAP 2013), Poster, pages 157-158, ACM, 2013.

3 - Ateliers nationaux
Gherasim, T., M. Harzallah, G. Berio et P. Kuntz. Construction automatique d'ontologies : comparaisons expérimentales à différentes échelles. Dans ExCo'Co Workshop, held in conjunction with IC'2011.

4 - Soumission
Article en soumission à une revue internationale
Gherasim, T., G. Berio, M. Harzallah et P. Kuntz. Identification of errors and unsuitable cases in automatically generated ontologies : from a state-of-the-art to an experimental analysis.

Détection de problèmes de qualité dans les ontologies construites automatiquement a partir de textes

Résumé : La démocratisation de l'utilisation des ontologies dans des domaines très variés a stimulé le développement d'approches proposant différents degrés d'automatisation du processus de construction d'une ontologie. Cependant, malgré le réel intérêt de ces approches, parfois les résultats obtenus peuvent être d'une faible qualité. L'objectif des travaux présentés dans cette thèse est de contribuer à l'amélioration de la qualité des ontologies construites automatiquement à partir de textes. Nos principales contributions sont : (1) une démarche pour la comparaison des approches, (2) une typologie des problèmes qui affectent la qualité les ontologies, et (3) une première réflexion sur l'automatisation de la détection des problèmes. Notre démarche de comparaison des approches comporte trois étapes complémentaires : (1) sur la base de leur degré de complétude et d'automatisation ; (2) puis sur la base de leurs caractéristiques techniques et fonctionnelles, et (3) expérimentalement par comparaison de leurs résultats avec une ontologie construite manuellement. La typologie proposée organise les problèmes de qualité selon deux dimensions : les erreurs versus les situations indésirables et les aspects logiques versus les aspects sociaux. Notre typologie contient 24 classes de problèmes qui recouvrent, en les complétant, les problèmes décrits dans la littérature. Pour la détection automatique nous avons recensé quelques unes des méthodes existantes pour chaque problème de notre typologie et nous avons mis en évidence les problèmes qui semblent encore ouverts. Et, nous avons proposé une heuristique pour un problème qui apparaît fréquemment dans nos expérimentations (étiquettes polysémiques).

Mots clés : ontologie, qualité de l'ontologie, problème de qualité, ontologie construite automatiquement, détection d'erreurs

Detection of quality problems in ontologies constructed automatically from texts

Abstract : The growing use of ontologies in a variety of application areas has stimulated the development of approaches proposing different degrees of automation of the ontology construction process. However, despite the real interest of these approaches, sometimes their results are of low quality. The aim of the work presented in this thesis is to contribute to the improvement of the quality of ontologies constructed automatically from texts. Our main contributions are : (1) a method for the comparison of the approaches, (2) a typology of problems that affect the quality of ontologies, and (3) a first reflection on automating the detection of quality problems. Our method for the comparison of approaches consists of three complementary steps : (1) on the basis of their degree of automation and completeness, (2) on the basis of their technical and functional characteristics, and (3) experimentally by comparing their results with a manually constructed ontology. The proposed typology organizes the quality problems according to two dimensions : errors versus unsuitable situations and logical aspects versus social aspects. Our typology contains 24 classes of problems that cover and complement the problems described in the literature. Concerning the automatic detection we have inventoried some of the existing methods for each problem in our typology and we have highlighted the problems for which the automatic detection remains an open issue. We have also proposed a heuristic for the detection of a quality problem that appears frequently in our experimentations (polysemic labels).

Keywords : ontology, ontology quality, quality problem, automatically built ontology, error detection