République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mentouri Constantine
Faculté des Sciences de l'Ingénieur
Département Informatique

N° d'ordre :
Série :

Thèse

pour obtenir le diplôme de
doctorat en sciences

Spécialité
Informatique

Titre

Développement des ontologies multi-points de vue: une approche basée sur la logique de descriptions

Présentée par :

Mounir HEMAM

Soutenue le 09 /01/2012

Devant le Jury :

N. ZAROUR	Professeur à l'université de Constantine	Président du Jury
Z. BOUFAIDA	Professeur à l'université de Constantine	Directrice de thèse
A. MOKHTARI-AISSANI	Professeur à l'USTHB - Alger	Examinatrice
F. BELALA	M.C. à l'université de Constantine	Examinatrice
F. BENCHEIKHA	M.C. à l'université de Skikda	Examinatrice

Dédicace

A la mémoire de mes chers parents!

Remerciements

"La connaissance est en fin de compte fondée sur la reconnaissance".
Ludwig Wittgenstein

Enfin, l'heure est venue de rédiger ces fameux remerciements, but ultime de chaque doctorant puisqu'il s'agit bel et bien de l'événement qui marque la fin de la thèse.

En premier lieu, je tiens à exprimer ma profonde reconnaissance au Professeur Mme Z. BOUFAIDA qui m'a formé à la recherche et a assuré avec beaucoup de gentillesse et d'attention l'encadrement de ma thèse de doctorat. Nombre des travaux réalisés ici sont le fruit de ses conseils et de sa patience. J'ai pu à plusieurs reprises apprécier et profiter de sa haute compétence scientifique et pédagogique que tout le monde lui connaît. Qu'elle trouve dans ces lignes si courtes, la sincère expression de ma gratitude et le témoignage de ma reconnaissance.

Je tiens à remercier profondément le Professeur N. ZAROUR pour l'honneur qu'il me fait en acceptant la présidence de ce jury.

C'est avec un grand plaisir que je compte le Professeur Mme MOKHTARI AISSANI Aicha de l'USTHB parmi les membres de ce jury. Qu'elle trouve l'expression de ma sincère reconnaissance pour l'attention avec laquelle elle a examiné mes travaux, et pour être venu de si loin pour participer à ce jury.

Je suis également tout à fait honoré de la présence à ce jury du docteur Mme F. Belala, Maitre de Conférences à l'université de Mentouri de Constantine et je la remercie vivement pour l'intérêt qu'elle a accordé à mon travail et d'avoir accepté de participer à ce jury.

J'adresse mes sincères remerciements au docteur F. BENCHEIKHA, Maitre de Conférences à l'université de Skikda, pour m'avoir fait l'honneur d'étudier mes travaux de thèse et de participer à ce jury en qualité d'examinateur.

Mes derniers mais non moins profonds remerciements iront à ma famille et plus particulièrement à mon épouse Meriem, pour leur soutien constant à travers ces longues années.

Résumé

Une ontologie est la représentation conceptuelle d'un domaine, permettant aux acteurs (humains et logiciels) de partager des connaissances. Elle contient un vocabulaire formalisé regroupant pour une discipline donné, l'ensemble des concepts, et de leur relations. Les définitions associées à chaque concept sont issues d'un consensus entre les différents acteurs et futurs utilisateurs de l'ontologie.

Puisqu'il existe généralement plusieurs façons d'appréhender les connaissances d'un domaine, la représentation des ontologies n'est donc pas une tâche facile. Ceci est dû principalement, à la difficulté de trouver des définitions consensuelles des concepts d'un domaine satisfaisant les définitions propres à chaque utilisateur, qui traduisent son point de vue sur le domaine. La difficulté de représenter des ontologies est liée principalement à l'existence de plusieurs utilisateurs qui peuvent s'intéresser au même domaine mais avec des points de vue différents.

Dans cette thèse, nous nous intéressons au problème de développement d'une ontologie dans une organisation hétérogène en prenant en compte différents points de vue et terminologies des communautés au sein de cette organisation. Une telle ontologie, que nous appelons ontologie multi-points de vue, confère à un même univers de discours plusieurs descriptions partielles telles que chacune soit relative à un point de vue. De plus, les différentes descriptions partielles partagent à un niveau global des éléments ontologiques et des passerelles sémantiques constituant un consensus entre les différents points de vue.

Pour fournir des éléments de réponse à cette problématique nous définissons un modèle de connaissances multi-points de vue fondé sur les notions de point de vue et d'ontologie. Cette dernière représente les connaissances du domaine partagées par plusieurs acteurs et le point de vue représente les connaissances du domaine qui sont pertinentes et visibles selon la perception d'un seul acteur. Ce modèle de connaissances multi-points de vue est utilisé pour la formalisation de l'ontologie multi-points de vue, en logique de descriptions. L'ontologie est transcrite dans le langage Vp-OWL, que nous définissons comme une extension du langage OWL, par l'ajout de nouveaux constructeurs dont la sémantique est donnée par une méta-ontologie. Enfin, une méthode, nommée Vp-MethOnto**,** permet le développement des ontologies multi-points de vue.

Mots-clés: Ontologie, points de vue, représentation des connaissances, logique de descriptions, Web sémantique

Abstract

An ontology is a conceptual representation of a domain, allowing the actors (human and software) to share knowledge. It contains a formal vocabulary grouping for a given discipline, the set of concepts and their relationships. The definitions of concepts are derived from a consensus between different actors and future users of the ontology

Since there is usually several ways of apprehending knowledge of a domain, the representation of ontologies is therefore not an easy task. This is due primarily to the difficulty of finding consensus definitions of concepts in a domain satisfying the definition of each user, which reflect his viewpoint on the domain. The difficulty of representing ontologies is mainly related to the existence of several users who can be interested in the same domain but with different viewpoints.

In this thesis, we are interested in the problem of developing an ontology in a heterogeneous organization, by taking into account different viewpoints and different terminologies of communities in the organization. Such ontology, that we call multi-viewpoint ontology, confers to the same universe of discourse, several partial descriptions, where each one is relative to a particular viewpoint. In addition, these partial descriptions share at global level, ontological elements and semantic bridges constituent a consensus between the various viewpoints.

In order to provide response elements to this problem we define a multi-viewpoints knowledge model based on viewpoint and ontology notions. Ontology represents domain knowledge shared by several actors and the viewpoint represents domain knowledge that are relevant and visible according to the perception of a single actor. The multi-viewpoints knowledge model is used to formalise the multi-viewpoints ontology in description logics. The ontology is transcribed in the Vp-OWL language, which we define as an extension of OWL, by adding new constructors where the semantics is given by a meta-ontology. Finally, a method called Vp-MethOnto, allows the development of multi-viewpoints ontologies.

Keywords: Ontology, Viewpoint, Knowledge Representation, Description Logic, Semantic Web.

ملخص

الأنطولوجيا هي تمثيل المفاهيمي لميدان معين، مما يسمح للجهات الفاعلة (البشرية والبرامج) لتقاسم المعرفة. انها توفر وسيلة للتعبير عن المفاهيم باستخدام لغة تمثيل رسمية. تعريفات المفاهيم هي نتيجة لتوافق في الآراء بين مختلف الجهات المعنية و مستخدمي الأنطولوجيا.

لأن هناك عادة العديد من الطرق لالتقاط المعارف في ميدان ما، لذا فإن تمثيل الانتولوجيا ليست مهمة سهلة. هذا يرجع أساسا إلى صعوبة العثور على تعريفات لمفاهيم توافق الآراء و التي تلبي احتياجات كل مستخدم، و تعكس وجهات نظره على الميدان. ترتبط أساسا صعوبة تمثيل الانتولوجيا لوجود العديد من مستخدميها قد يكونون مهتمين بنفس الميدان، ولكن مع وجهات نظر مختلفة.

في هذه الرسالة، نحن نركز على مشكلة تصميم الأنطولوجيا داخل منظمة غير متجانسة، مع الأخذ بعين الاعتبار وجهات نظر ومصطلحات مختلفة للمجتمعات داخل تلك المنظمة. هذه الأنطولوجيا، التي نسميها الأنطولوجيا متعددة جهات النظر، تمنح لنفس الميدان العديد من الأوصاف الجزئية بحيث يكون كل على وجهة نظر. بالإضافة إلى ذلك ، كل الأوصاف الجزئية يتقاسمون على مستوى عام مفاهيم ومصطلحات وكذلك جسور معينيّة ما يشكل توافق في الآراء بين مختلف وجهات النظر.

لتقديم بعض الاجابات على هذه المشكلة نعرف نموذج المعارف متعدد وجهات النظر بناء على مفهومي وجهة نظر والأنطولوجيا. الأنطولوجيا تمثل تقاسم المعرفة من قبل العديد من الجهات الفاعلة وجهة نظر تمثل المعرفة التي هي ذات صلة والمرئية من خلال تصور وجود فاعل واحد ا. يستخدم هذا النموذج لإضفاء الطابع الرسمي على الأنطولوجيا متعددة جهات النظر، وذلك باستخدام المنطق الوصفى. تتم كتابة الأنطولوجيا في لغة Vp-OWL، والتي تعتبر امتدادا للللغة OWL عن طريق إضافة منشئات جديدة. أخيرا، يقترح طريقة، المسماة Vp-MethOnto، لبناء الأنطولوجيا متعددة جهات النظر.

الكلمات المفتاحية :
الأنطولوجيا ، وجهات النظر ، تمثيل المعرفة ، المنطق الوصفى ، الويب الدلالي

Table des matières

Chapitre III– Outils et technologies du Web sémantique

Chapitre IV– Vp-MethOnto : Une méthode pour le développement des ontologies multi-points de vue

Liste des tableaux

Liste des figures

Introduction générale

« Les commencements ont des charmes inexprimables »
Moliére

1 Contexte de la recherche

À l'heure actuelle, les sources d'informations et de connaissances dans les entreprises (ou organisations) deviennent de plus en plus importantes et elles jouent aussi un rôle crucial pour le développement de ces entreprises. La collaboration et le partage du travail entre les différents acteurs, au sein des entreprises, nécessite un partage de connaissances et ce besoin peut devenir encore plus crucial. Pour réussir, les entreprises doivent bien gérer ces sources de connaissances et supporter l'utilisation effective des connaissances [Luong, 2007].

Dans les dernières années, le développement impressionnant de l'Internet a renforcé l'apparition d'une énorme quantité d'informations disponibles sur le Web. La richesse et la croissance exponentielle de ce volume d'informations promet d'être une mine d'or pour les entreprises. Cependant, cette croissance d'informations donnera lieu aussi à de vrais obstacles si les informations ne sont pas bien structurées et bien représentées. Cela montre les limitations du Web actuel du point de vue de la recherche, l'organisation, l'accès et la maintenance de ces informations… selon les exigences de l'utilisateur.

La naissance de la nouvelle génération du Web (le Web sémantique) est un des efforts pour pallier à ces limitations. L'idée du Web sémantique est de rendre le contenu des ressources diffusées sur le Web accessible aux programmes informatiques de façon à ce que ceux-ci soient mieux à même d'aider les utilisateurs humains. Il s'agit de décrire ces ressources selon une représentation formelle, c'est-à-dire en lien avec une sémantique clairement définie et qui soit conçue pour une interprétation par des programmes.

La base de l'infrastructure du Web sémantique s'appuie sur l'explicitation de la conceptualisation d'un domaine, partagée par une communauté et représentée dans une ontologie du domaine concerné. Les ontologies sont devenues très populaires, en occupant une place de choix dans le domaine de l'ingénierie des connaissances et ont prouvé leur efficacité pour la représentation des connaissances.

De manière générale, une ontologie contient un vocabulaire formalisé regroupant pour une discipline donnée, l'ensemble des concepts, et de leur relations. Les définitions associées à chaque concept sont issues d'un consensus entre les différents acteurs et futurs utilisateurs de

l'ontologie. OWL (Web Ontology Language) est le langage standard pour la représentation des ontologies sur le Web et constitue en ce sens la principale technologie sur laquelle repose le Web sémantique. OWL est issu à la fois des langages du Web et de formalismes de représentation des connaissances, tels que les logiques de descriptions.

2 Problématique

Représenter consiste à faire une abstraction d'une réalité riche en information. Cette abstraction dépend de l'acteur qui la réalise, de son domaine d'intérêt et de ses connaissances préalables. Ainsi, lorsque plusieurs observateurs (acteurs) travaillent sur un même univers de connaissance, ils observent des éléments et relations différents. Dans des domaines complexes et multi-disciplinaires, co-existent souvent différentes façons d'appréhender les éléments de connaissances, c'est-à-dire différents points de vue (ou perceptions) selon lesquels ces connaissances peuvent être représentées. Ainsi, le même domaine peut avoir plusieurs représentations alternatives, où chacune d'entre elles est décrite selon une perception ou un point de vue particulier. Chaque points de vue constitue donc une façon propre à une discipline de représenter, d'organiser et d'utiliser ces connaissances.

Généralement, les organisations vivent dans un environnement dynamique et hétérogène. De ce fait, la construction et l'exploitation des ontologies, au sein d'une organisation, n'est donc pas toujours une tâche facile. Ceci est dû principalement, à la difficulté de trouver des définitions consensuelles des concepts d'un domaine satisfaisant les définitions propres à chaque utilisateur, qui traduisent son point de vue sur le domaine.

La difficulté de construction des ontologies est liée principalement à l'existence de plusieurs communautés d'utilisateurs qui peuvent s'intéresser au même domaine mais avec des points de vue différents. Ces communautés, évoluant dans un environnement pluridisciplinaire, coexistent et collaborent entre elles. Chaque communauté a ses intérêts propres et perçoit différemment les entités conceptuelles du même univers de connaissances à représenter.

La plupart des méthodes et méthodologies de constructions des ontologies négligent cette variété de perceptions et elles offrent des outils et des directives pour créer un modèle unique du monde, conçu pour une seule vision du monde. L'approche par point de vue, à laquelle nous nous intéressons, s'oppose à cette approche mono-point de vue et permet donc de modéliser une même réalité selon des points de vue différents. Un point de vue dans son sens large, est la perception qu'une personne ou un groupe de personnes en fait d'un monde observé.

Dans cette thèse, nous nous intéressons au problème de développement d'une ontologie dans une organisation hétérogène en prenant en compte différents points de vue et terminologies des communautés au sein de cette organisation. Une telle ontologie, que nous appelons *ontologie multi-points de vue*, confère à un même univers de discours plusieurs descriptions partielles telles que chacune soit relative à un point de vue. De plus, les différentes descriptions partielles partagent à un niveau global des éléments ontologiques consensuels et des passerelles. Ces dernières établissent les communications entre les points de vue et représentent ainsi la collaboration interdisciplinaires.

3 Contributions

L'objectif visé par cette thèse est de proposer une approche de développement des ontologies qui tiennent compte des différents points de vue des utilisateurs. L'élément clé de notre approche est de permettre la description de ce type d'ontologies, sans éliminer l'hétérogénéité mais en faisant cohabiter l'hétérogénéité (au niveau local) et le consensus (au niveau global). Ainsi, pour atteindre l'objectif de la thèse qui est de permettre le développement des ontologies multi-points de vue dans le cadre du Web sémantique, notre démarche est la suivante :

- Au niveau conceptuel, nous proposons un modèle de représentation des connaissances en prenant en compte la notion de point de vue : *modèle de connaissances multi-points de vue*. Le modèle proposé est fondé sur les notions de *point de vue* et d'*ontologie* et permet la description, selon différents points de vue, de plusieurs familles d'individus appelées *concepts globaux*.
  - Un concept global est un concept qui est vu et partagé par l'ensemble des points de vue avec certaine propriétés communes.
  - Chaque point de vue détermine d'une part, un ensemble de propriétés du concept global significatives "visibles" pour le point de vue et, d'autre part, une structuration de la connaissance du concept global dans une hiérarchie de *concepts locaux*, induite par la relation de spécialisation.
  - Les différents points de vue peuvent être liés par des *passerelles* : une passerelle indique une relation d'inclusion, d'égalité ou d'exclusion entre des concepts locaux de différents points de vue ; les passerelles établissent une communication entre les divers points de vue.
  - Une instance selon notre modèle de connaissances multi-points de vue représente un individu particulier d'un concept global, qui peut être rattaché, dans chaque point de

vue, à son concept local d'appartenance le plus spécialisé. Un individu possède donc une description de base (i.e. description globale) et peut être décrit partiellement selon un ou plusieurs points de vue.

- Au niveau formel, le modèle de connaissances multi-points de vue est appliqué pour formaliser l'ontologie multi-points de vue en logique de descriptions. Pour cela, nous utilisons un sous-langage de la logique de descriptions de type $\mathcal{SHOQ}(\mathcal{D})$ pour exprimer les différentes notions inhérentes aux points de vue telles que les concepts globaux et locaux, les passerelles, les estampilles …. Les estampilles sont des notions de modélisation utilisés afin de différencier les représentations multiples d'une même réalité [Balley et al., 2004]. Nous adaptons le mécanisme d'estampillage utilisé dans [Benslimane et al., 2006] pour permettre la multi représentation des concepts dans le formalisme de la logique de descriptions. Dans notre approche, une estampille (i.e. label) permet de reconnaître pour chaque élément ontologique (i.e. concept, rôle, individu) le point de vue auquel il appartient. Enfin, une définition formelle de la syntaxe et de la sémantique, en logique de descriptions, des différentes notions introduites pour le support des points de vue est présentée.
- Au niveau opérationnel et sur la base du formalisme de logique de description développé pour la formalisation de l'ontologie multi-points de vue, un langage d'ontologies multi-points de vue, nommé **Vp-OWL**, est construit. Ce dernier, permet de représenter des ontologies multi-points de vue par un langage qui est une extension du langage d'ontologie OWL. A cet effet, de nouveaux constructeurs sont introduits pour prendre en compte l'aspect multi-points de vue. Par ailleurs, une méta-ontologie (i.e. une ontologie de représentation) est définie pour permettre la description de la sémantique des différents constructeurs du langage Vp-OWL selon la syntaxe RDF/XML de OWL. L'une des caractéristiques clés des ontologies multi-points de vue représentées selon notre langage Vp-OWL est qu'elles peuvent être exploitées par des raisonneurs, tel que le système RACER, offrants des services d'inférences associés aux logiques de descriptions. Le système RACER, que nous avons utilisé, permet de lire un document au format Vp-OWL (i.e. une ontologie multi-points de vue) puis fournir des services d'inférences pour aider à construire et à maintenir l'ontologie multi-points de vue par la détection des inconsistances, des redondances, des dépendances cachées, et des erreurs de classification.
- Enfin, une méthode pour le développement des ontologies multi-points de vue, nommée **Vp-MethOnto**, est proposée permettant d'aboutir à une ontologie multi-points de vue

opérationnelle représentée par le langage Vp-OWL. Pour ce faire, quatre principales étapes sont suivies à savoir: (i) spécification des besoins, (ii) conceptualisation, (iii) formalisation et (iv) codification.

4 Organisation de la thèse

Cette thèse est organisée en cinq chapitres :

- Le **Chapitre 1** propose un état de l'art de l'ingénierie ontologique. il présente tout d'abord la notion d'ontologie, leur différents composants, types et usages. Les différents modèles de connaissances utilisés en ingénierie ontologique sont ensuite exposés, particulièrement le modèle des logiques de descriptions. Les différentes phases du cycle de vie des ontologies sont ensuite décrites, ainsi que les méthode et méthodologies associées.
- Le **Chapitre 2** est consacré à l'étude du concept de point de vue. Après une réflexion sur les différentes interprétations que l'on peut accorder à ce terme, nous nous attachons à donner un aperçu de quelques travaux proposant une formalisation et une représentation multi-points de vue des connaissances. Nous abordons aussi une notion proche de celle de points de vue, qui est celle de contexte, telle qu'elle est considérée en représentation des connaissances.
- Le **Chapitre 3** décrit la vision du Web sémantique (i.e. l'architecture, les composants principaux, les langages de représentation, etc.). A la fin, il aborde l'utilisation des technologies du Web sémantique en vue de matérialiser une mémoire d'entreprise en un Web sémantique d'entreprise (WSE).
- Le **Chapitre 4** constitue le cœur de la contribution, à savoir le développement d'ontologies multi-points de vue. Il présente et décrit en détaille les étapes constituant la méthode Vp-MethOnto en mettant l'accent sur les objectifs de chacune de ses étapes ainsi que les actions et outils qui doivent être mener et utiliser. Pour illustrer notre approche, une étude de cas est présentée pour le domaine d'immobilier.
- Le **Chapitre 5** présente, dans un premier temps, l'implémentation du langage Vp-OWL. Ensuite, il présente les différentes étapes à suivre pour la codification de l'ontologie multi-points de vue dans le langage Vp-OWL.

Chapitre 1

La représentation des connaissances et l'ingénierie ontologique

« *Il y a une science qui étudie l'être en tant qu'être et les attributs qui lui appartiennent essentiellement* »
Aristote

1 Introduction

De nos jours, la modélisation formelle joue un rôle central pour les problèmes, pour lesquels on dispose uniquement des connaissances de nature linguistique. Le domaine de l'intelligence artificielle a été introduit à cet effet. Son objectif primordial est la représentation des connaissances en utilisant un langage formel.

Une branche de l'intelligence artificielle, appelée *l'ingénierie des connaissances* s'intéresse, notamment, en l'étude des systèmes experts. Ces derniers avaient comme objectif principal la résolution automatique de problèmes. Les systèmes à base de connaissances ont été développés par la suite pour permettre le stockage, la consultation et le raisonnement automatique sur les connaissances stockées.

Nées des besoins de représentation des connaissances, *l'ingénierie ontologique* est, à l'heure actuelle au cœur des travaux menés dans le domaine de l'ingénierie des connaissances. Visant à établir des représentations et des modèles pour permettre aux systèmes informatiques de manipuler la partie sémantique des informations. L'ingénierie ontologique est un domaine pluridisciplinaire puisque la construction d'ontologies – l'objet de travail principal de l'ingénierie ontologique – demande à la fois une étude des connaissances humaines et la définition de langage de représentation, ainsi que la réalisation des systèmes pour les manipuler.

Dans ce chapitre, nous présentons un état de l'art sur les ontologies. Dans les premières sections de ce chapitre nous donnons les différentes définitions, les différents composants, types et usages des ontologies. Ensuite, les modèles de représentation de connaissance utilisés en ingénierie ontologique sont présentés dans la section 6, le modèle basé sur la logique de descriptions, sur lequel se base notre travail, y étant largement détaillé. Enfin, nous décrivons les différentes phases du cycle de vie de construction des ontologies, ainsi que les méthodes et les méthodologies associées.

2 Donnée, information et connaissance

Il n'existe pas de définition précise de *donnée* ni de *connaissance*. Selon [Chabalier, 2004], les deux termes « donnée » et « connaissance » se définissent généralement dans le domaine d'informatique au travers des définitions de «base de données» et de «base de connaissances». En effet, une base de données permet de structurer et de stocker des données brutes qui peuvent exprimer des résultats d'observations. Alors qu'une base de connaissances est capable de stocker des données mais également de fournir les mécanismes capables de raisonner sur ces données. A la différence d'une base de données classique, une base de connaissances permet, par les mécanismes de raisonnement, de déduire de nouvelles connaissances.

Les connaissances sont ainsi capables de spécifier la manière dont les informations peuvent être manipulées. La figure 1.1 est une adaptation de l'exemple présenté dans [Chabalier, 2004] traitant des différences entre les notions de « donnée », « information », et « connaissance ». En effet, la structuration d'une donnée brute permet de la transformer en information, et replacer cette information dans un contexte interprétable constitue une connaissance.

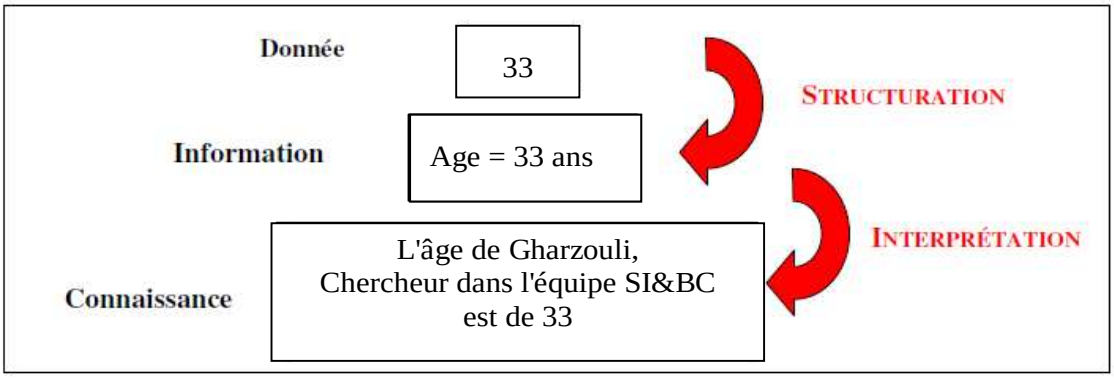


Figure 1.1- Différence entre donnée, information et connaissance

Pour représenter des connaissances, une certaine ***modélisation*** doit être effectuée, notamment celles des objets et des types d'objets impliqués dans ces connaissances, et leurs inter-relations. Cette collection d'objets, de types d'objets et de leurs relations est parfois appelée une ***conceptualisation***.

Selon le dictionnaire de l'informatique, la conceptualisation représente la collection des ***objets***, de ***concepts*** et des autres ***entités*** qui sont supposés exister dans un certain domaine d'intérêt, et les ***relations*** qui les relient. Une conceptualisation est une ***vue abstraite***,

simplifiée du monde que l'on veut représenter. Par exemple, on peut conceptualiser une famille par un ensemble de noms, de sexes et de relations entre les membres de la famille. Le choix d'une conceptualisation est la première étape de la représentation de connaissances. Chaque base de connaissances, système à base de connaissances, ou agent modélisé au niveau connaissance est, explicitement ou implicitement, relatif à une certaine conceptualisation. Une représentation formelle de cette conceptualisation s'appelle une ***ontologie*** .

3 Notion d'ontologie

Historiquement, le terme Ontologie a tout d'abord été défini en Philosophie comme une branche de la Métaphysique qui s'intéresse à l'existence, à l'être en tant qu'être et aux catégories fondamentales de l'existant. Plus tard, le concept d'ontologie est apparue en pleine lumière dans le domaine de l'intelligence artificielle, afin de résoudre les problèmes de modélisation des connaissances et, plus précisément, en ingénierie des connaissances. Ceci a engendré de nombreuses définitions que nous allons résumer dans les paragraphes suivants en présentant les points de vue en sciences cognitives et en informatique.

3.1 Ontologies dans la perspective des sciences cognitives

Chez les Grecs anciens, le terme **Ontologie** (avec un O majuscule) signifie : « *étude de l'être en tant qu'être, indépendamment de ses déterminations particulières* » laissant sous-entendre un discours indépendant d'un point de vue quelconque. Il importe donc de faire une distinction entre l' « observateur » des choses et les choses « observées ». En philosophie, il existe trois hypothèses sur l'appréhension de la connaissance : l'objectivisme, le subjectivisme et le constructivisme [Dietz, 2006].

L'*objectivisme* défend l'hypothèse selon laquelle le monde existe par lui-même, de façon indépendante de l'observateur. L'observateur croit en la vérité d'une réalité objective. Le *subjectivisme* défend l'hypothèse que la réalité n'existe pas en dehors du sujet qui l'observe et qu'à la limite, chaque observateur a sa propre image de la réalité. Quant au point de vue *constructiviste*, il admet l'hypothèse qu'il n'y a pas d'objectivité absolue de la réalité, que la réalité existe aussi par l'interprétation de celle-ci. En contrepartie, il admet aussi que les éléments interprétés font partie d'une réalité que l'on pourrait qualifier de semi-objective. Pour Dietz, l'ontologie s'interprète selon un point de vue constructiviste. En effet, l'ontologie est une composante particulière de la réalité qui sert de base de communication au discours

sur une partie de la réalité (objectivisme). En même temps, l'ontologie est le résultat d'une construction issue de l'interprétation d'un agent intelligent (subjectivisme).

3.2 Ontologies dans la perspective informatique

La notion d'ontologie a été abordée pour la première fois par John McCarthy dans le domaine de l'intelligence artificielle (IA). Il affirmait déjà en 1980 que les concepteurs des systèmes intelligents fondés sur la logique devraient d'abord énumérer tout ce qui existe [Valery, 2004]. Cette approche présentée par John McCarthy n'est pas la seule puisque par la suite plusieurs définitions ont été proposées par d'autres auteurs du domaine :

- En intelligence artificielle, la définition communément admise d'une ontologie a été donnée par T. Gruber où il décrit une ontologie comme une spécification explicite d'une conceptualisation. T. Gruber défend ce point de vue comme suit : " *An ontology is a specification of a conceptualization. That is, an ontology is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agent* " [Gruber, 1995].
- Après, afin de compléter le sens philosophique originel, N. Guarino a introduit la notion d'ontologie formelle, qui est définie en tant que modélisation conceptuelle, ou une représentation de cette modélisation. " *Une ontologie est un accord sur une conceptualisation partagée et éventuellement partielle* " [Guarino et Giaretta, 1995].
- De même, M. Uschold définit une ontologie comme une description formelle d'entités et de leurs propriétés, relations, contraintes et comportements. De plus, les auteurs ont introduit, dans [Uschold et Grüninger, 1996], la notion de l'ontologie explicite "*An explicit ontology may take a variety of forms, but necessarily it will include a vocabulary of terms and some specification of their meaning*".
- C. Roche a donné une définition générique et simple "*Une ontologie est une conceptualisation d'un domaine à laquelle sont associés un ou plusieurs vocabulaires de termes. Les concepts se structurent en un système et participent à la signification des termes. Une ontologie est définie pour un objectif donné et exprime un point de vue partagé par un groupe de personne. Une ontologie s'exprime dans un langage (représentation) qui repose sur une théorie (sémantique) qui garantit des propriétés de l'ontologie en termes de consensus, cohérence, réutilisation et partage*" [Roche, 2005].

- Pour Oberle, les ontologies, en tant que systèmes de représentation, peuvent être considérées au même titre que des systèmes de modélisation conceptuelle, tels que le modèle *entité-relation* de Chen (1976) ou le modèle *UML* de Rumbaugh, Jacobson et Booch (1999). Cependant, l'ontologie se distingue de ces systèmes de représentation par les points suivants [Oberle, 2006]:
  - l'idée primaire des ontologies est d'établir un certain niveau de consensus autour d'un vocabulaire afin de faciliter l'échange d'informations entre applications ;
  - les ontologies sont formalisées au moyen d'un système de représentation logique dont la sémantique est spécifiée de façon non ambigüe ;
  - l'ontologie est exprimée au moyen d'un langage de représentation qui, à l'état d'exécution, sert de représentation à des applications de recherche et de raisonnement.

Dans la figure 1.2, nous présentons une synthèse des principales définitions d'une ontologie, du point de vue informatique, schématisée dans le langage graphique de *Modélisation par Objets Typés* (MOT), dont on trouve une description détaillée dans [Paquette 2007, Héon, 2010].

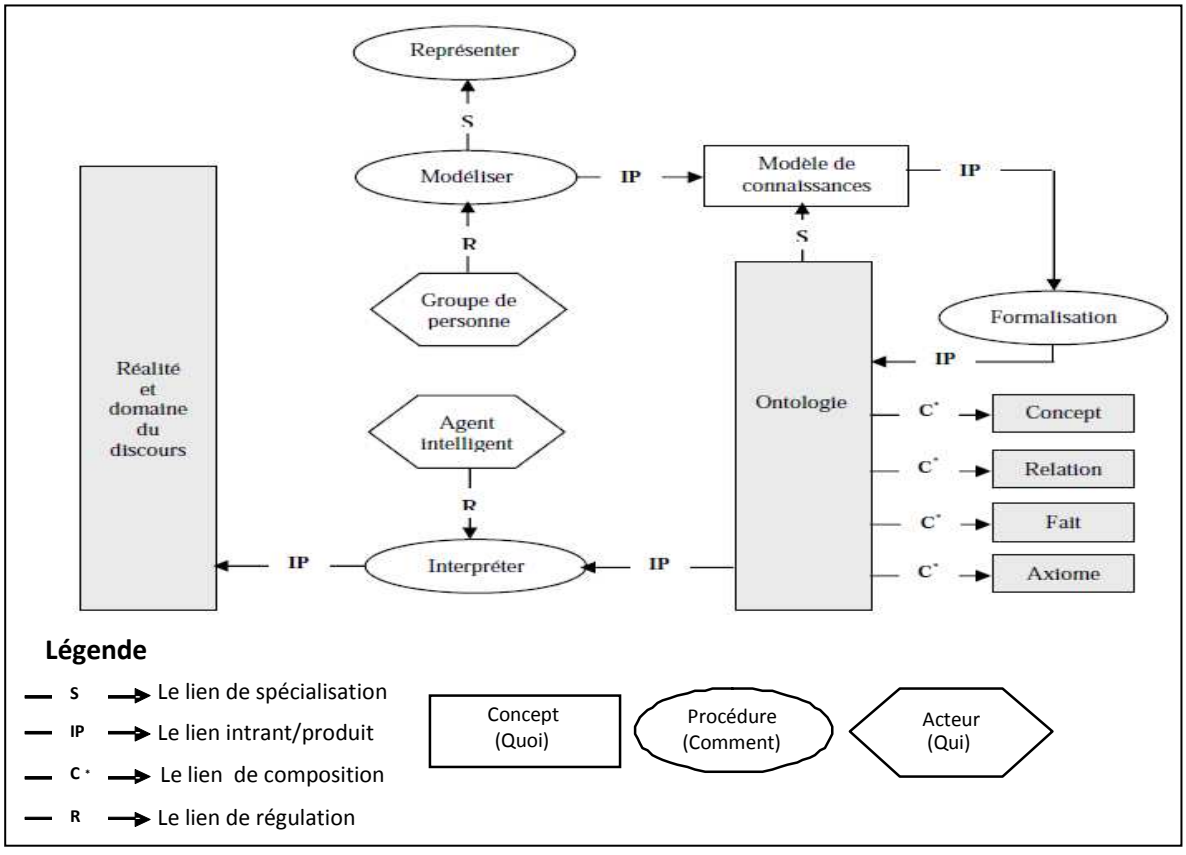


Figure 1.2- Définition d'une ontologie représentée en MOT

Ainsi, une ontologie peut est considérée comme étant un artéfact dont la structure permet de représenter des concepts, des relations, des axiomes et des faits. Le *concept* est la représentation d'une entité abstraite de la réalité à décrire. À un niveau d'abstraction factuelle, on retrouve l'*instance* ou l'*individu* dont le représenté est en lien avec un objet concret de la réalité observée. L'action de créer un fait à partir d'un concept se nomme : *instancier*. Finalement, l'*axiome*, qui définit le concept, est une assertion (c'est-à dire un énoncé qui est supposé vrai dans le domaine représenté) à propos des instances en relation avec les concepts de l'ontologie.

De manière conceptuelle, une ontologie est la description d'une réalité et la formalisation d'un modèle conceptuel, qui, selon Gruber, correspond à une *conceptualisation*. L'ontologie est donc le résultat d'une double activité de modélisation et de formalisation. La *modélisation* est un processus de représentation de la réalité dont la résultante est un modèle conceptuel. La *formalisation* est un processus « de mise en forme » de ce modèle conceptuel afin que l'artéfact résultant (l'ontologie) soit interprétable, de façon cohérente et non ambigüe, par un agent informatique.

Dans ce qui suit, nous détaillons, d'une manière non exhaustive, les éléments qui constituent l'ontologie.

4 Eléments constitutifs de l'ontologie

Comme nous l'avons dit précédemment, les ontologies rassemblent les connaissances propres à un domaine donné. En représentation des connaissances, ces ontologies existent sous la forme de concepts et de relations, et permettent d'en fixer la sémantique selon un degré de formalisme variable. Nous détaillons ci-dessous les différents composants de l'ontologie considérée en tant qu'objet informatique.

4.1 Les concepts

Un concept est la description abstraite d'un ou plusieurs objets semblables. Un concept correspond à ce qui a une existence propre, matérielle ou immatérielle, dans le domaine étudié du monde réel. L'ensemble des propriétés d'un concept s'appelle sa *compréhension* ou son *intension*, et l'ensemble des objets ou êtres qu'il englobe, son *extension*. L'extension du concept est représentée par un ensemble d'objets qui sont appelés *instances du concept*. Les instances représentent des éléments spécifiques d'un concept. Chaque instance est caractérisée par son appartenance à un (ou éventuellement plusieurs) concept(s) et par des valeurs de propriétés.

Selon le langage de représentation d'ontologie, des équivalences conceptuelles peuvent en plus être introduites en définissant des concepts comme étant équivalents à des expressions construites à partir d'autres concepts. De tels concepts sont dits concepts définis contrairement aux autres concepts dont l'intention ne peut être comprise que par une connaissance extérieure au modèle. Ces derniers concepts sont appelés concepts primitifs. Ces expressions de concepts permettent de distinguer deux familles d'ontologies [Pierra, 2003]:

- les ontologies canoniques qui ne comportent que des concepts primitifs, et
- les ontologies non canoniques qui possèdent aussi des concepts définis.

Un concept est caractérisé par un ensemble de propriétés. Dans [Furst , 2002], l'auteur s'inspirait des propriétés proposées par N. Guarino. Nous donnons les plus intéressantes:

- Un concept est *générique* s'il n'admet pas d'extension. La vérité, par exemple, n'a pas d'extension.
- Un concept porte une propriété d'*identité* si cette propriété permet de différencier deux instances de ce concept.
- Un concept est *rigide* s'il ne peut pas être une instance d'autres concepts. Par exemple, l'être vivant est un concept rigide, mais un " être humain " n'est pas un concept rigide, car l'humain est une instance du concept " être vivant ".
- Un concept est *anti-rigide* s'il peut être une instance pour d'autres concepts. Comme le cas de l'être humain dans l'exemple précédent.

4.2 Les relations entre concepts

Les concepts (respectivement, les instances) peuvent être reliés entre eux par des relations au sein d'une ontologie. Une relation est définie comme une notion de lien entre des entités, exprimée souvent par un terme ou par un symbole littéral ou autre. Généralement, les liens sont classés en deux catégories : des liens *hiérarchiques* et des liens *sémantiques*.

Une relation *hiérarchique* lie un élément supérieur, dit l'hyperonyme, et un élément inférieur, dit l'élément hyponyme, ayant les mêmes propriétés que le premier élément avec au moins une en plus. Dans certains cas, le couple (*Hyperonymie, Hyponymie*) s'interprète par (*Type*, *Sous-Type*). L'hyperonyme englobe l'hyponyme. On pourra alors écrire « HYPONYME est une sorte de HYPERONYME ». *Exemple* : Le chat est une sorte d'animal, donc, " animal " est l'hyperonyme de "chat".

Une relation *sémantique* permet de lier des instances de concepts, ou des concepts génériques. Elles sont caractérisées par un terme (voire plusieurs) et une signature qui précise le nombre d'instances de concepts que la relation lie, leurs types et l'ordre des concepts, c'est–à–dire la façon dont la relation doit être lue. Par exemple, la relation « écrit » lie une instance du concept «`Personne`» et une instance du concept « `Manuscrit` », dans cet ordre.

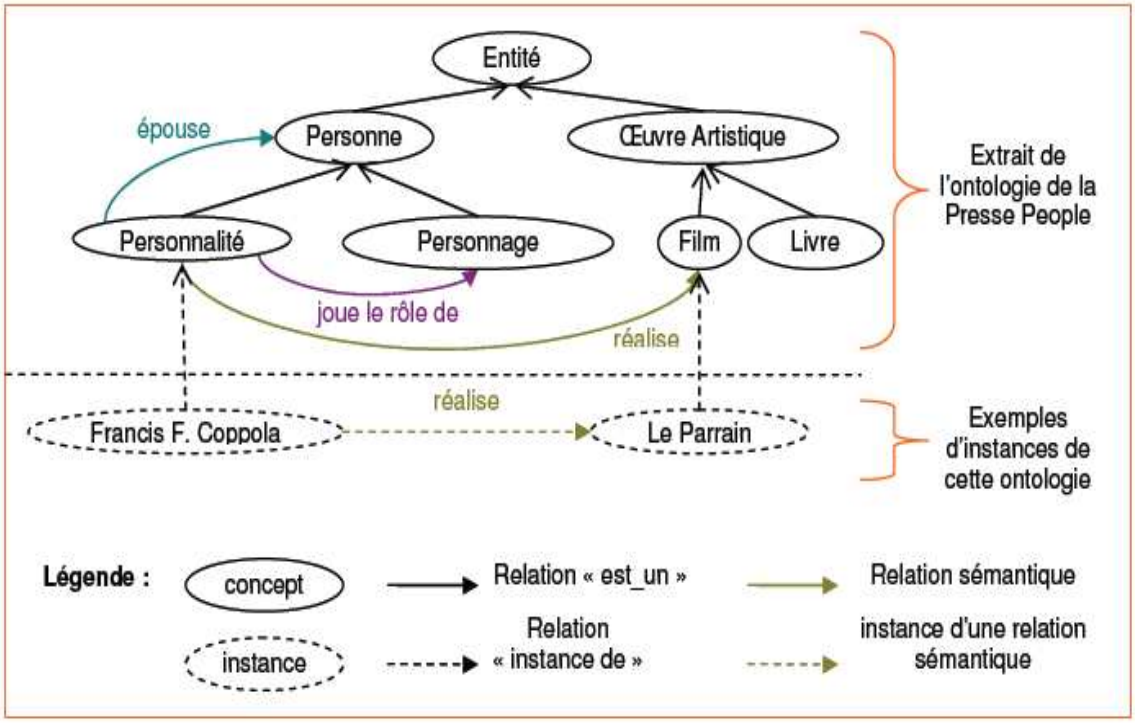


Figure 1.3 - Extrait d'une ontologie dans le domaine de la presse «People» [Amardeilh, 2007]

La figure 1.3 présente un extrait d'une ontologie décrivant les connaissances dans le domaine de la presse «People» où nous avons des concepts tels que Personne, Personnage, Personnalité Œuvre artistique, Film, Livre qui sont classés en ordre hiérarchique. Dans cet exemple « Francis F. Coppola » est une instance du concept «Personnalité» et une relation sémantique «réalise» est instanciée entre cette instance et celle du concept « Film », i.e. « Le Parrain». L'ontologie contiendra donc les informations Personnalité(Francis F. Coppola), Film(Le Parrain) et réalise(Francis F. Coppola, Le Parrain).

Il est à noter qu'on peut distinguer différents niveaux d'ontologies selon le but pour lequel elles sont conçues. La classification de [Niles et Pease, 2001, Gangemi et al., 2003] repose sur deux critères : le sujet et la structure d'une conceptualisation. En ce qui concerne le sujet de la conceptualisation, les auteurs distinguent :

- Les **ontologies de domaine** : les plus connues, elles expriment des conceptualisations spécifiques à un domaine, elles sont réutilisables pour des applications sur ce domaine.

- Les **ontologies d'application** : elles contiennent des connaissances du domaine nécessaires à une application donnée ; elles sont spécifiques et non réutilisables.

- Les **ontologies génériques** : appelées aussi ontologies de haut niveau, elles expriment des conceptualisations très générales tels que le temps, l'espace, l'état, le processus, les composants, elles sont valables dans différents domaines; les concepts figurant dans une ontologie du domaine sont subsumés par les concepts d'une ontologie générique, la frontière entre les deux étant floue.

- Les **ontologies de représentation** ou **méta-ontologies** : indiquent des formalismes de représentation de la connaissance ; les ontologies génériques ou du domaine peuvent être écrites en utilisant des primitives d'une telle ontologie.

Dans ce mémoire, le mot ontologie sera utilisé pour désigner une ontologie de domaine, ce qui n'exclut pas que cette ontologie de domaine soit elle-même fondée sur une ontologie de représentation.

Dans la section suivante nous parlons de l'usage et de l'utilité des ontologies dans les différents domaines.

5 Usages des ontologies

Les ontologies, au début, ont connu une large utilisation dans le domaine de l'intelligence artificielle. Aujourd'hui les ontologies sont largement utilisées pour différents buts (traitement de langage naturel, gestion de la connaissance, e-commerce, le Web sémantique, etc.) dans différentes communautés (i.e. ingénierie de connaissance, bases de données et génie logiciel).

Notons que la communauté de bases de données aussi bien que la communauté de conception orientée objet construisent également des modèles de domaine en utilisant les concepts, les relations, les propriétés, etc., mais la plupart du temps les deux communautés imposent des contraintes moins sémantiques que celles imposées dans les ontologies lourdes.

Dans le commerce électronique, les ontologies sont habituellement utilisées pour représenter les produits et les services qui sont offerts dans les systèmes du e-commerce et qui sont donnés aux utilisateurs dans les catalogues pour consultation.

La vision du Web sémantique [Berners-Lee et al., 2001] est d'ajouter la sémantique compréhensible par la machine (méta-information) au World Wide Web en utilisant une

ontologie pour définir et organiser cet espace de méta-information. Le Web sémantique vise à réaliser l'intégration de toutes les sources d'informations sur le Web, permettant la réutilisation des données à travers les applications et rendant la recherche intelligente sur Internet possible.

Comme synthèse, nous pouvons dire que les ontologies possèdent un grand nombre d'applications et d'usages tels que:

- les systèmes de recherche d'informations
- les inférences, pour découvrir les incohérences sur les connaissances ou les données.
- les systèmes à base de connaissances dédiés à la résolution de problèmes.
- l'aide au diagnostic: des maladies, des problèmes techniques etc.
- l'interopérabilité des systèmes d'information.
- La coopération des logiciels
- les systèmes de traitement automatique du langage naturel.

Pour définir des ontologies, un modèle d'ontologies est nécessaire. Un modèle d'ontologies est un formalisme permettant de représenter des ontologies. Ceci fera l'objet des prochaines sections.

6 Ontologies et modèles de connaissance

La modélisation des connaissances consiste à représenter un ensemble de données ou connaissances sous une forme adaptée pour qu'un opérateur humain et/ou machine, puisse les interpréter et les manipuler. Une représentation est définie selon un modèle qui fournit les règles syntaxiques de modélisation, appelées la syntaxe. Le modèle peut être muni d'une sémantique, logique par exemple, pour définir clairement le sens de ce qui est modélisé. Le modèle est dit formel si les modélisations basées sur ce modèle peuvent être interprétées syntaxiquement et sémantiquement sans ambiguïté.

Les modèles de représentation de connaissance utilisés en ingénierie ontologique peuvent être regroupés selon les paradigmes conceptuels qu'ils réifient. Sont ainsi distingués :

- les modèles à base de Frames ;
- le modèle des Réseaux Sémantiques;
- les modèles des Logiques de Description.

Chacun de ces modèles de représentation est implémenté dans un ou plusieurs langages implémentant une partie ou la totalité du modèle, en particulier des langages adaptés au Web

et utilisant la syntaxe XML (Cf**.** Chapitre. 3). Les langages implémentant ces modèles sont souvent opérationnels, c'est-à-dire qu'ils offrent des mécanismes de raisonnement, et servent alors à représenter des ontologies déjà opérationnelles. D'autres ne font que permettre la spécification déclarative de connaissances. Le modèle des logiques de descriptions étant celui que nous utilisons dans nos travaux, nous en détaillerons la présentation.

6.1 Le modèle des Frames

Le modèle des Frames est un classique de l'Intelligence Artificielle et a été initialement proposé comme langage de représentation d'ontologies par T. GRUBER [Gruber, 1995]. Le principe de ce modèle est de décomposer les connaissances en *classes* (ou frames) qui représentent les concepts du domaine. À un frame est rattaché un certain nombre d'*attributs* (slots), chaque attribut pouvant prendre ses valeurs parmi un ensemble de *facettes* (facets). Une autre façon de présenter ces attributs est de les considérer comme des *relations* binaires entre classes dont le premier argument est appelé *domaine* (domain) et le deuxième *portée* (range) (*cf*. Figure 1.4).

Des *instances* des classes, correspondant à l'extension de chaque concept, peuvent être ajoutées, ainsi que des *fonctions* qui sont des types particuliers de relations liant un ensemble de classes à une valeur calculée à partir des valeurs des attributs des classes. La spécification de propriétés conceptuelles des attributs (ou relations) recourt à des formules de la logique du premier ordre. La sémantique de la subsomption est purement extensionnelle : une frame F1 est plus spécifique qu'une frame F2 si toute instance de F1 est instance de F2. F-Logic est l'exemple le plus connu de langage opérationnel à base de frames [Kifer et al., 1995].

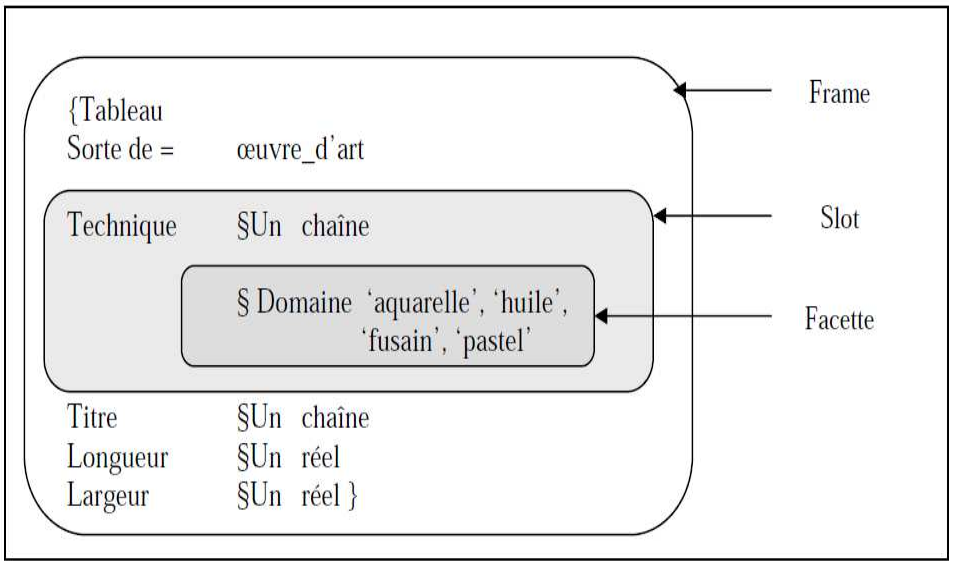


Figure 1.4- Structure Frame, Slot et Facette

6.2 Représentation graphique des connaissances : Les réseaux sémantiques

Un réseau sémantique est une structure de graphe qui encode les connaissances ainsi que leurs propriétés. Les nœuds du graphe représentent des objets (concepts, situations, événements, etc) et les arcs expriment des relations entre ces objets. Ces relations peuvent être des liens " sorte - de " exprimant la relation d'inclusion ou des liens " est-un " représentant la relation d'appartenance. Par exemple, on peut dire que le busard est un rapace qui est une sorte d'oiseau (*Cf.* Figure 5).

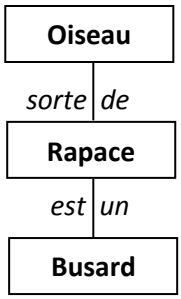


Figure 1.5- Exemple de réseau sémantique utilisant les relations "est un" et " sorte-de "

En fait, une ontologie est considérée comme un réseau sémantique. Elle regroupe un ensemble de concepts décrivant complètement un domaine. Ces concepts sont liés les uns aux autres par des relations, d'une part, taxonomiques (hiérarchisation des concepts) et, d'autre part, sémantiques.

Parmi les modèles classiques en IA de réseaux sémantiques, on trouve les graphes conceptuels, développés essentiellement par Sowa [Sowa, 2000], qui se veulent un langage intermédiaire entre des formalismes destinés à l'ordinateur (typiquement, les langages de programmation) et les langues naturelles.

Une ontologie formelle, sous forme de graphes conceptuels, fait appel à deux types de structures : des *hiérarchies de types (supports)* et des *graphes conceptuels* proprement dits. La **hiérarchie de types** est un treillis qui représente une relation d'ordre partiel entre différents types de concepts (*Cf.* Figure 1.6). En règle générale, la relation d'ordre exprimée dans une hiérarchie de types est la relation de spécialisation/généralisation. Outre les hiérarchies de types, le formalisme des graphes conceptuels fait appel à une seconde structure de données : les **graphes conceptuels**. D'un point de vue mathématique, *les graphes conceptuels* sont des graphes bipartites orientés dont les deux types de nœuds sont appelés *concepts* et *relations (cf.* Figure 1.7*).*

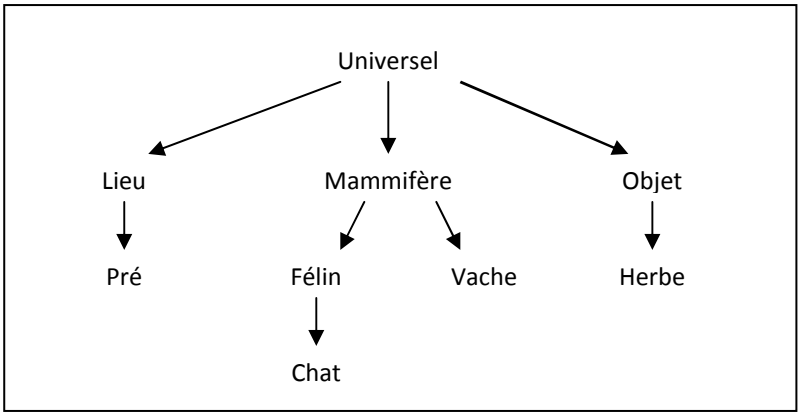


Figure 1.6- Exemple de hiérarchie de types de concepts

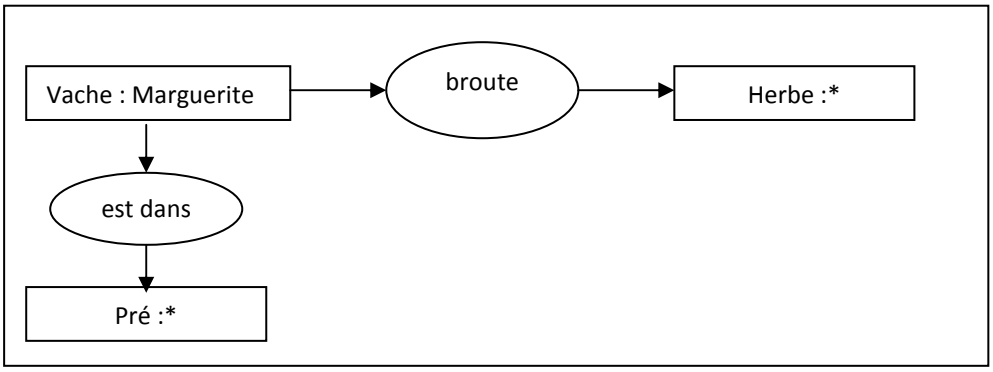


Figure 1.7- Exemple de graphe conceptuel

L'intérêt principal des réseaux sémantiques est double: d'une part, ils fournissent une représentation graphique directement interprétable par l'humain ; d'autre part, ils permettent à la machine de disposer d'un formalisme tout aussi directement exploitable. Néanmoins, bien qu'adaptés à la représentation de définitions ou d'événements, c'est-à-dire la représentation du quoi, ses réseaux sémantiques sont dotés d'une sémantique peu claire et non véritablement fixée, ce qui conduit les utilisateurs à commettre certaines mauvaises actions : **(i)** considérer la même entité, tantôt comme un individu, tantôt comme une classe, **(ii)** confondre propriétés descriptives et propriétés typiques, **(iii)** confondre la *définition* d'une classe donnée par des conditions nécessaires et suffisantes d'appartenance à la classe (notion de classe *définie*), la *description* d'une classe où les conditions d'appartenance sont seulement nécessaires (notion de classe *primitive*). Ces problèmes ont été clairement posés et des solutions ont été recherchées. C'est là une des origines des travaux qui ont mené à la définition du formalisme des *logiques de descriptions*, qui traite de manière satisfaisante les questions de sémantique et

d'inférences, en introduisant notamment la relation de subsomption et le processus de classification.

6.3 Concepts et rôles pour la représentation des connaissances : les logiques de descriptions

Les logiques de descriptions (LDs) [Baader et al., 2007] sont une famille de langages de représentations de connaissances. Ces logiques ont été introduites dans les années 80 dans le but de se donner un langage permettant de définir une terminologie d'une façon supposée plus "naturelle" qu'en logique du premier ordre. Elles sont aussi moins expressives que la logique du premier ordre puisque la plupart de ces logiques forment un sous-ensemble décidable de la logique du premier ordre. Maintenant, il existe une grande variété de logiques de description dont l'expressivité et l'efficacité de raisonnement dépend de l'utilisation ou non de constructeurs spécifiques.

Les ontologies en logique de description modélisent les connaissances selon deux niveaux de description : le *niveau terminologique* (TBox) et le *niveau factuel* (ABox) (*cf.* Table 1.1). Le niveau terminologique définit des concepts et des rôles représentant respectivement des ensembles d'entités du monde et des relations entre ces entités. Le niveau factuel, quant à lui, énonce des faits sur des individus représentant les entités elles-mêmes.

Table 1.1- Exemple de T-box et de A-box en LDs

Représentation en DLs	Signification
La terminologie du domaine (T-Box)	
Femme := Personne ⊓ (¬Homme)	Un Femme est une personne qui n'est pas un Homme
Mère := Femme ⊓ (∃ avoir-enfant.Persone)	Une Mère est une Femme reliée à personne par la relation avoir-enfant
Les assertions (A-Box)	
Femme (Khenchela) Mère (Kahina) Avoir-enfant (Kahina, Khenchela)	Khenchela est une Femme Kahina est une Mère Kahina a un enfant du nom de Khenchela

Niveau terminologique. Les concepts et rôles sont définis soit en les nommant (concepts ou rôles atomiques), soit en les construisant à l'aide d'autres concepts ou rôles et de constructeurs spécifiques. Les connaissances sur ces concepts et rôles sont décrites à l'aide

d'axiomes indiquant la subsomption[1] (de concepts ou de rôles) ou la transitivité d'un rôle. La Table 1.2 donne une liste de constructeurs existant en logiques de description. Elle n'est pas exhaustive.

Table 1.2– Syntaxe et sémantique des constructeurs de logiques de descriptions.

Termes atomiques	Syntaxe	Sémantique	Famille
individu	a	$a^I \in \Delta^I$	
concept atomique	A	$A^I \subseteq \Delta^I$	
rôle atomique	R	$R^I \subseteq \Delta^I \times \Delta^I$	$(\mathcal{AL})$
Constructeur de concepts	**Syntaxe**	**Sémantique**	**nom**
concept universel	$\top$	$\top^I = \Delta^I$	$(\mathcal{AL})$
concept vide	$\bot$	$\bot^I = \emptyset$	$(\mathcal{AL})$
conjonction	$C \sqcap D$	$(C \sqcap D)^I = C^I \cap D^I$	$(\mathcal{AL})$
disjonction	$C \sqcup D$	$(C \sqcup D)^I = C^I \cup D^I$	$\mathcal{U}$
négation	$\neg C$	$\Delta^I \setminus C^I$	$\mathcal{C}$
restriction existentielle	$\exists R.C$	$\{x \mid \exists y.\langle x,y\rangle \in R^I \wedge y \in C^I\}$	$(\mathcal{AL})$
restriction de valeur	$\forall R.C$	$\{x \mid \forall y.\langle x,y\rangle \in R^I \Rightarrow y \in C^I\}$	$(\mathcal{AL})$
restrictions	$\leq nR$	$\{x \mid \sharp\{y.\langle x,y\rangle \in R^I\} \leq n\}$	$\mathcal{N}$
numériques	$\geq nR$	$\{x \mid \sharp\{y.\langle x,y\rangle \in R^I\} \geq n\}$	$\mathcal{N}$
nominaux	$\{a_1,\ldots,a_n\}$	$\{a_1^I,\ldots,a_n^I\}$	$\mathcal{O}$
Constructeur de rôles	**Syntaxe**	**Sémantique**	**nom**
conjonction de rôles	$R \sqcap S$	$(R \sqcap S)^I = R^I \cap S^I$	$.^{\sqcap}$
disjonction de rôles	$R \sqcup S$	$(R \sqcup S)^I = R^I \cup S^I$	$.^{\sqcup}$
complément de rôle	$\neg R$	$(\Delta^I \times \Delta^I) \setminus R^I$	$.^{\neg}$
clôture transitive	R^+	transitive closure of R^I	$.^+$
rôle inverse	R^-	$\{\langle y,x\rangle \mid \langle x,y\rangle \in R^I\}$	$\mathcal{I}$
composition de rôles	$R \circ S$	$\{\langle x,z\rangle \mid \exists y.\langle x,y\rangle \in R^I \wedge \langle y,z\rangle \in S^I\}$	$.^{\circ}$
axiomes de TBox	**Syntaxe**	**Contraintes d'interprétation**	**nom**
subsomption	$C \sqsubseteq D$	$C^I \subseteq D^I$	$(\mathcal{AL})$
inclusion de rôles	$R \sqsubseteq S$	$R^I \subseteq S^I$	$\mathcal{H}$
transitivité de rôle	$\mathrm{Trans}(R)$	$R^I = (R^+)^I$	$\mathcal{S}$
axiomes de ABox	**Syntaxe**	**Contraintes d'interprétation**	**nom**
appartenance à un concept	$C(a)$	$a^I \in C^I$	$(\mathcal{AL})$
appartenance à un rôle	$R(a_1,a_2)$	$\langle a_1^I, a_2^I\rangle \in R^I$	$(\mathcal{AL})$
idendité	$a_1 = a_2$	$a_1^I = a_2^I$	?

Niveau factuel. Ce niveau n'introduit que des noms d'individus et des propriétés sur ces individus, en exploitant les termes du niveau terminologique. Les axiomes de la ABox sont appelés assertions. Il existe généralement deux types d'assertions en LD (*Cf.* Table 1.3**)**.

[1] La relation de *subsomption* est une relation de généralisation-spécialisation qui permet d'organiser concepts et rôles.

Une assertion de la forme $C(a)$, où C est un nom de concept et a un individu, indique que a est une instance de C, c'est-à-dire qu'il représente un élément de la classe C. Par ailleurs, une assertion de la forme $r(a, b)$, permet d'indiquer que l'individu a est en relation avec l'individu b par la propriété (ou le rôle) r.

Table 1.3 – Les assertions de concepts de rôles en logiques de descriptions

Assertion	Syntaxe	Sémantique
inclusion d'assertion de concept (d'individu)	$C(a)$	$C(a)^{\mathcal{I}} = a^{\mathcal{I}} \in C^{\mathcal{I}}$
inclusion d'assertion de rôle	$r(a,b)$	$r(a,b)^{\mathcal{I}} = (a^{\mathcal{I}}, b^{\mathcal{I}}) \in r^{\mathcal{I}}$
égalité d'individus	$a = b$	$(a{=}b)^{\mathcal{I}} = (a^{\mathcal{I}}{=}b^{\mathcal{I}})$
inégalité d'individus	$a \neq b$	$(a{\neq}b)^{\mathcal{I}} = (a^{\mathcal{I}}{\neq}b^{\mathcal{I}})$

C est un nom de concept, r est un nom de rôle, et a ainsi que b identifient des individus.

6.3.1 Interprétation et modèle

La sémantique d'un langage de représentation des connaissances définit la manière dont les symboles doivent être interprétés et comment les formules contraignent ces interprétations. Elle se caractérise par :

– une notion d'interprétation associant aux symboles d'une signature des éléments d'un certain "univers" ou domaine d'interprétation,

– et des contraintes définissant la satisfaction d'une formule.

Définition 1 (Interprétation). Une sémantique est associée aux concepts, aux rôles et aux individus par l'intermédiaire d'une interprétation $I = (\Delta^I, .^I)$ composée d'un ensemble non vide Δ^I, appelé domaine de l'interprétation, et d'une fonction d'interprétation $.^I$ qui fait correspondre à un concept un sous-ensemble de Δ^I, à un rôle un sous-ensemble de $\Delta^I \times \Delta^I$, et à un individu un élément de Δ^I et à une assertion de rôle un élément de $\Delta^I \times \Delta^I$, de telle sorte que les expressions en colonne « Sémantique » des Tables 1.2 et 1.3 soient satisfaites.

Définition 2 (Modèle). Une interprétation I est dite un modèle pour une T-Box $\mathcal{T}$ si et seulement si I satisfait toutes les expressions de $\mathcal{T}$. Pour une A-Box $\mathcal{A}$, I est un modèle si et seulement si I satisfait toutes les assertions de $\mathcal{A}$. I est un modèle pour $\mathcal{K} = (\mathcal{T}, \mathcal{A})$ si et seulement si I est un modèle pour $\mathcal{T}$ et $\mathcal{A}$ à la fois.

6.3.2 Expressivité des logiques de descriptions

Les langages de la famille des logiques de descriptions se distinguent les uns des autres par l'ensemble des constructeurs qu'ils offrent.

L'une des premières logiques de description est le langage $\mathcal{FL}^-$ -pour *Frame Logic*- [Brachman et Levesque, 1984]. Le langage $\mathcal{FL}^-$ offre les constructeurs de conjonction ($\sqcap$), de quantificateur universel ($\forall$), de quantificateur existentiel non typé ($\exists r$, c-à-d $\exists r.\top$), et la restriction du co-domaine d'un rôle ($r|C$). Ce constructeur $r|C$ peut être rapproché du constructeur existentiel typé ($\exists r.C$), mais avec deux différences : il s'applique aux rôles et n'a aucune contrainte d'existence ; $(r|C)^I = \{(x, y) \in \Delta^I \times \Delta^I | (x, y) \in r^I \wedge y \in C^I\}$.

Le langage $\mathcal{FL}$ est une extension de $\mathcal{FL}-$ où le quantifieur existentiel est typé ($\exists r.C$). Le langage AL = {$\top, \bot$,C1 $\sqcap$ C2, $\forall$r.C, $\exists$r, $\neg$A} étend le langage $\mathcal{FL}-$ en y ajoutant la négation, sur des concepts primitifs uniquement. Ce langage peut-être considéré comme la logique de base des autres logiques de descriptions. En effet, les logiques de descriptions sont des combinaisons des différents éléments de la Table 1.2. Par exemple si on ajoute la négation complète (étendue aux concepts définis), repérée par la lettre C, à la logique $\mathcal{AL}$ on obtient la logique $\mathcal{ALC}$. On peut remarquer que le langage $\mathcal{ALC}$ équivaut à $\mathcal{ALUE}$, puisque l'union ($\mathcal{U}$) et la quantification existentielle typée ($\mathcal{E}$) peuvent s'exprimer par la négation complète et inversement : $C1 \sqcup C2$ correspond à $\neg(\neg C1 \sqcap \neg C2)$ et $\exists r.C$ correspond à $\neg\forall r.\neg C$ [Baader et al., 2003].

6.3.3 Classification et instanciation

Les logiques de descriptions permettent de faire l'hypothèse d'un *monde ouvert*, où les connaissances d'un domaine peuvent à tout moment être complétées. C'est-à-dire que contrairement à l'hypothèse d'un *monde fermé*[2] où ce qui n'est pas actuellement connu est considéré comme *faux* (aussi appelé *negation as failure*), ici ce qui n'est pas connu reste incertain. Ainsi, la classification et l'instanciation des connaissances sont à la base des raisonnements en logiques de descriptions pour répondre à ce type de modélisation [Baader., 2009].

[2] L'expression *monde fermé* fut introduite par [Reiter, 1980] dans le domaine des bases de données; par opposition l'expression *monde ouvert* a vu le jour.

Définition 1 (Classification et Instanciation). La classification est un processus qui consiste à placer un concept ou un rôle dans leurs hiérarchies respectives, on parle de classification de concepts et de classification de rôles. L'instanciation[3] est un processus qui consiste à déterminer les concepts (notamment le ou les plus spécifiques) dont un individu donné peut être une instance.

Les trois principales opérations d'inférence pour la classification et l'instanciation sont le test de subsomption, le test de satisfiabilité de concept et le test d'instanciation:

Définition 2 (Test de subsomption). Pour une base de connaissances $\mathcal{K} = (\mathcal{T}\text{-}Box, \mathcal{A}\text{-}Box)$, un concept C_1 (respectivement un rôle r_1) subsume un concept C_2 (respectivement un rôle r_2) par rapport à $\mathcal{K}$, si et seulement si pour tout modèle I de $\mathcal{K}$, $C_2^I \subseteq C_1^I$ (respectivement $r_2^I \subseteq r_1^I$).

Définition 3 (Test de satisfiabilité). Pour $\mathcal{K} = (\mathcal{T}\text{-}Box, \mathcal{A}\text{-}Box)$, un concept C est satisfiable si et seulement si il existe au moins un modèle I de $\mathcal{K}$ tel que $C^I \neq \emptyset$, c-à-d C admet des instances.

Définition 4 (Test d'instanciation). Pour $\mathcal{K} = (\mathcal{T}\text{-}Box, \mathcal{A}\text{-}Box)$, une assertion de concept a est instance d'un concept C (notamment pour C le plus spécifique), notée $\mathcal{K} \models C(a)$, si et seulement si pour tout modèle I de K, $a^I \in C^I$. On dit que C(a) est satisfiable par I.

6.3.4 Raisonneurs en logiques de descriptions

Les raisonneurs[4] en logiques de descriptions profitent des dépendances entre les tests de subsomption et de satisfiabilité. Deux familles d'algorithmes de raisonnement pour les logiques de descriptions cohabitent.

La première famille, issue du système KL-ONE [Brachman et Schmolze, 1985], utilise des algorithmes de type *Normalisation/Comparaison* (abrégé par NC), qui sont basés sur la subsomption [Napoli, 97]. Ces algorithmes ont l'avantage d'être simples à mettre en œuvre, mais sont efficaces pour des logiques de descriptions simples comme $\mathcal{ALC}$.

La seconde famille utilise des algorithmes basés sur la satisfiabilité, avec l'utilisation d'algorithmes appelés *tableaux* [Horrocks et Sattler, 2007]. Les raisonneurs actuels font partie de cette seconde famille. Ces algorithmes peuvent s'appliquer sur des logiques de descriptions plus expressives, comme celles basées sur SH. Des systèmes comme KRIS

[3] On parle parfois de classification d'instance.

[4] Un raisonneur est une application logicielle qui peut inférer des conséquences logiques issues d'un ensemble de connaissances établies.

[Baader et Hollunder, 1991], FaCT - *Fast Classification of Terminologies* - [Horrocks, 1998], ou d'autres plus récents comme FaCT++ (version en C++ de FaCT), Pellet [Sirin *et al.*, 2004] et Racer - *Renamed ABox and Concept Expression Reasoner* - [Haarslev et Müller, 2001] se basent sur de tels algorithmes, tout en possédant des optimisations sophistiquées. FaCT++, Pellet et Racer sont orientés pour la manipulation de documents OWL DL, et sont capables de raisonner sur la T-Box et la A-Box (uniquement la T-Box pour FaCT++).

6.4 Discussions

Un formalisme de représentation d'ontologie peut être jugé selon deux dimensions principales, qui sont fortement liées entre elles: la représentation de connaissances et les mécanismes de raisonnement.

A la différence des graphes conceptuels et des langages des frames, les logiques de descriptions sont caractérisées par les points suivants:

- Les LDs sont bien adaptées à la représentation d'ontologies complexes car elles disposent d'une sémantique claire, un haut pouvoir de description de concepts complexes, par exemple *grâce aux restrictions de valeurs, la négation et la conjonction (disjonction) de concepts*. D'autre part, elles procurent des mécanismes d'inférences efficaces telle que la subsomption et la satisfiabilité des concepts.
- Les LDs ont l'avantage de faire apparaître explicitement les notions d'identité (individus), de structure (rôles), de classes (concepts), de sous-typage (subsomption) et d'expressivité (choix des constructeurs de concepts).
- Les LDs sont le type de formalisme retenu par le projet *Web Sémantique* pour représenter les ontologies **[**Horrocks et al., 2005**]**. En effet le langage d'encodage et d'échange d'ontologies pour le Web Sémantique noté *OWL* (*Cf***.** Chapitre 3) est basé sur la sémantique des logiques de descriptions et en particulier sur le langage *SHIQ* **[**Horrocks et al., 2003**]**.

Ces différentes caractéristiques des logiques de descriptions nous ont amené à adopter ce formalisme de représentation des connaissances dans le cadre de notre travail. Mais il existe un autre facteur qui a également contribué à faire pencher la balance en sa faveur: la communauté des logiques de descriptions. En effet, cette communauté de chercheurs a publié de nombreuses recherches dans des domaines extrêmement variés et a développé des outils robustes.

7 Fondements de l'ingénierie ontologique

Après ce survol sur les formalismes de représentation des connaissances ontologiques, cette section s'intéressera à la construction d'ontologies formelles. Dans cette section, les principales approches pour construire une ontologie seront présentés. Ensuite, la place qu'occupe le processus de construction des ontologies dans le cycle de vie de l'ontologie sera illustrée. Finalement nous présenterons une modeste revue des principales méthodologies mise en œuvre lors de la construction d'une ontologie.

7.1 Les différentes approche pour la conception d'ontologies

En général, il existe trois principales approches pour la conception d'une ontologie de domaine : inspirationnelle, synthétique et collaborative [Keita, 2007]. La table 1.4 ci-dessous décrit ces trois approches.

Table 1.4 - Les trois principales approche pour la conception d'ontologie

Approches de conception d'ontologie	Base de conception
Inspirationnelle	Point de vue individuel sur le domaine.
Synthétique	Ensemble d'ontologies existantes, chacune fournit une caractérisation partielle du domaine.
Collaborative	Multiple points de vue sur le domaine.

Dans l'approche *inspirationnelle*, un concepteur seul prend les décisions de rassembler les conditions de l'analyse de domaine, de la conception et de la vérification de l'ontologie. De ce fait, le développeur doit être à la fois expert du domaine et expert en conception d'ontologie (i.e. ontologiste[5]) afin d'assurer le succès de la conception, et surtout, pour assurer l'adoption de l'ontologie par la communauté d'utilisateurs. Ainsi, ce type d'approche dépend fortement de la créativité d'une seule personne, de son inspiration et de sa perception personnelle du domaine. Cette approche est souvent appliquée dans un domaine qui peut être cerné (compris) par un seul concepteur, où ce dernier est capable de dominer le processus de conception de l'ontologie.

[5] L'ontologiste est celui qui construit des ontologies. Son travail touche, d'une part, à l'informatique, à la logique, aux modèles de représentation de connaissances, aux normes et standards dans ce domaine et, d'autre part, à la linguistique et aux sciences cognitives.

Dans l'approche *synthétique*, un ensemble d'ontologies liées est identifié. Le concepteur d'ontologie synthétise alors les éléments de ces ontologies préexistantes avec les concepts du nouveau domaine à représenter, produisant ainsi une nouvelle ontologie unifiée.

L'approche *collaborative*. La marque d'une ontologie "moderne" est sa grande taille et sa haute complexité ; ces ontologies englobent un ensemble si riche de la connaissance que leur compréhension (complète) dépasse celle de n'importe quel développeur ou concepteur seul ou même d'une petite équipe de concepteurs. De ce fait, le développement d'une ontologie à grande échelle doit être le fruit d'un effort commun de plusieurs experts de domaine et des ingénieurs de la connaissance, voir même les futurs utilisateurs de l'ontologie.

Dans le cadre de notre travail nous nous positionnons par rapport à la troisième approche, c'est à dire à la conception coopérative d'une ontologie selon des points de vue multiple.

7.2 Cycle de vie d'une ontologie

Les ontologies étant destinées à être utilisées comme des composants logiciels dans des systèmes répondant à des objectifs opérationnels différents, leur développement doit s'appuyer sur les mêmes principes que ceux appliqués en génie logiciel. Ainsi, les ontologies doivent être considérées comme des objets techniques évolutifs et possédant un cycle de vie qui nécessite d'être spécifié [Furst, 2004]. Dans ce contexte, les activités liées aux ontologies sont d'une part des activités de gestion incluant la planification, le contrôle, et la garantie de qualité, et d'autre part des activités orientées développement regroupant les activités de pré-développement, de développement et de post-développement ; s'y ajoutent des activités transversales de support telles que l'évaluation, la documentation et la gestion de version.

Un cycle de vie inspiré du génie logiciel est proposé dans [Dieng et al., 2001, Gandon, 2006]. Il comprend une étape initiale d'évaluation des besoins, une étape de construction, une étape de diffusion, et une étape d'utilisation. Après chaque utilisation significative, l'ontologie et les besoins sont réévalués et l'ontologie peut être étendue et, si nécessaire, en partie reconstruite.

L'étape de construction peut être décomposée en trois principales étapes qui sont: la conceptualisation, la formalisation appelée également ontologisation et l'opérationalisation **(**C*f.* **§ 7.3)**. Le processus de construction peut être intégré au cycle de vie d'une ontologie comme indiqué en figure 1.8.

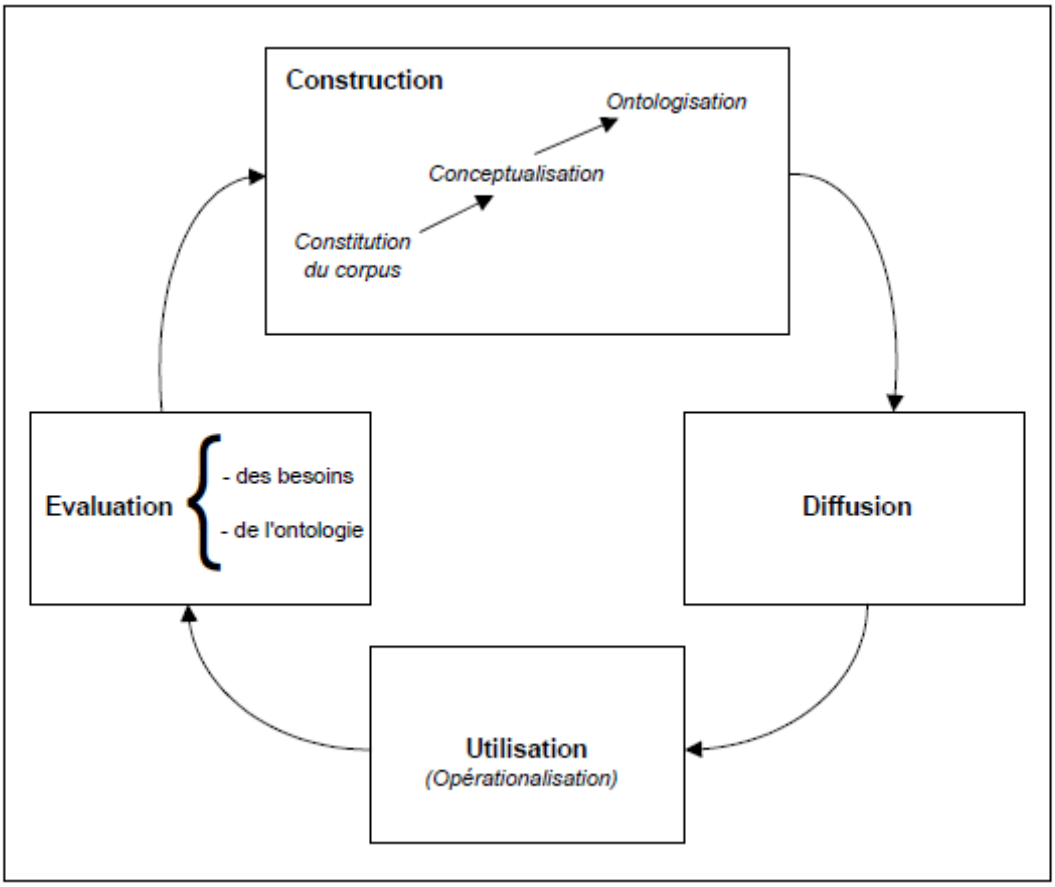


Figure 1.8- Le cycle de vie d'une ontologie [Furst, 2004]

7.3 Processus de construction d'une ontologie

Une analyse des travaux de [Gandon et Dieng-Kuntz, 2001] [Leclère et al, 2002a] sur les ontologies nous a permis de dégager un consensus sur le processus de conception d'une ontologie exploitable dans un système à base de connaissances. La figure 1.9 illustre ce consensus qui repose sur un enchaînement de trois étapes, permettant de passer des données brutes à l'ontologie opérationnelle. Les données brutes, constituant un corpus (exprimé *a priori* en langage naturel). En effet, pour construire une ontologie, il faut tout d'abord recueillir son corpus. La nature du corpus dépend du domaine d'application de l'ontologie. En effet, un corpus peut être sous forme de documents textuels, des interviews, des représentations conceptuelles, de sites Web, etc., ... Une fois le corpus prêt, la première étape de construction de l'ontologie peut être déclenchée.

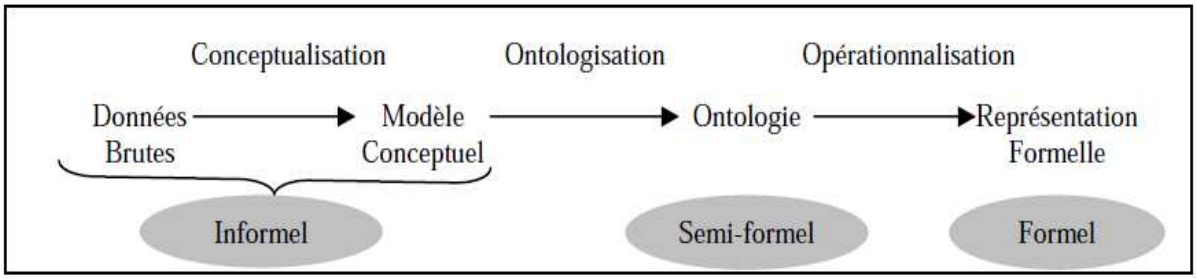


Figure 1.9- Processus général de construction d'une ontologie

Bien que ces étapes soient effectuées en séquence, il est évident que des « *retours en arrières* » sont possibles ; en d'autres termes, la réalisation d'une étape peut provoquer la remise en cause des choix adoptés aux étapes précédentes et, par voie de conséquence, conduire à compléter ou modifier les ontologies obtenues en amont du processus. Par la suite nous allons clarifier les diverses opérations menées à chaque étape ainsi que le statut de l'ontologie obtenue à chacune de ces étapes.

7.3.1 Conceptualisation

La première étape, appelée *conceptualisation*, permet d'aboutir à un modèle informel, donc sémantiquement ambiguë et généralement exprimé en langage naturel. Cette étape, consiste, à partir des données brutes, à dégager les concepts et les relations entre ces concepts permettant de décrire de manière informelle les entités cognitives du domaine. Elle est réalisée par un expert du domaine assisté de l'ingénieur de la connaissance et aboutit à un **modèle conceptuel**. Ainsi, le modèle obtenu consiste en un ensemble de termes désignant les entités du domaine de connaissances (concepts, relations, propriétés des concepts et des relations, etc.), assortis d'informations exprimant leur sémantique. La découverte des connaissances d'un domaine peut s'appuyer à la fois sur l'analyse de documents et sur l'interview d'experts du domaine. Ces activités doivent être raffinées au fur et à mesure que la conceptualisation émerge.

Il est à noter que certaines connaissances implicitement utilisées dans le domaine ne sont cependant jamais exprimées, ni dans le corpus, ni par les experts. Un des points les plus délicats de la conceptualisation consiste donc à identifier ces connaissances. La mise en évidence de ces connaissances implicites ne peut *a priori* se faire que lors de l'utilisation de l'ontologie, lors d'une phase de test opérationnel et/ou de test des questions de compétences [Leclère et al, 2002b]. Ceci montre que le processus de construction des ontologies ne peut être séquentiel et que des allers-retours entre les différentes étapes du processus sont à prévoir.

La problématique de la conceptualisation est essentiellement de reconnaître, dans les données brutes, les connaissances qui relèvent du domaine considéré, de les recenser de façon exhaustive. Les étapes suivantes du processus vont conduire progressivement à la formalisation de ces connaissances.

7.3.2 Ontologisation

La première étape de cette formalisation est l'*ontologisation*, qui consiste en une formalisation partielle, sans perte d'information, du modèle conceptuel obtenu dans l'étape précédente. Cette formalisation partielle facilite sa représentation ultérieure dans un langage complètement formel et opérationnel. De plus, il faut bien voir que l'ontologisation est une transcription des connaissances dans un certain formalisme de connaissances, ce formalisme devant être aussi générique que possible, mais présentant une sémantiquement clair. S'imposer de conserver toutes les connaissances conduit à intégrer, à l'ontologie du domaine, des connaissances qui ne peuvent être formalisées, ou dont la sémantique est ambiguë.

Cependant, le modèle obtenu pendant à ce stade est souvent qualifié de *semi-formel* pour justifier le fait que certaines connaissances ne peuvent pas être totalement formalisées. Le caractère semi-formel d'une ontologie lui interdit d'être utilisée telle quelle dans un SBC. En revanche, une ontologie, contenant toutes les connaissances d'un domaine, constitue le support idéal de communication et de partage des connaissances de ce domaine. En fait, cette étape produit un résultat en deux parties :

- *Une partie formelle*, disposant d'une sémantique précise ou du moins consensuelle;
- *Une partie informelle* qui ne dispose pas d'une sémantique claire ou consensuelle, ou tout ou moins d'une sémantique fixée *a priori* et donc exprimée dans un langage naturel ou semi-structuré.

7.3.3 Opérationalisation

La dernière étape, *appelée parfois représentation*, consiste à formaliser complètement l'ontologie obtenue précédemment dans le cadre d'un langage de représentation de connaissances, formel et opérationnel. Le terme opérationnel à la fois pour caractériser un modèle de représentation doté de mécanismes de raisonnement et d'une sémantique opérationnelle, et pour caractériser tout langage exécutable qui implémente un tel modèle. Cette étape est menée par un spécialiste du langage de représentation et par l'ingénieur de la connaissance. On obtient alors une **représentation formelle** des connaissances du domaine.

Ainsi, le caractère *formel* de l'ontologie permet à une machine, via cette ontologie, de manipuler des connaissances du domaine.

Il faut noter que la formalisation totale des connaissances peut imposer à ce stade une perte d'information [Hemam, 2005], du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation d'ontologies choisi.

Finalement, l'ontologie opérationnelle est intégrée en machine au sein d'un système manipulant le modèle de connaissances utilisé via le langage opérationnel choisi. Mais avant d'être livrée aux utilisateurs, l'ontologie doit, bien sûr, être testée par rapport au contexte d'usage pour lequel elle a été bâtie.

7.4 Méthodes et méthodologies de construction d'ontologies

La construction d'une ontologie peut être effectuée de plusieurs façons : construction à partir de zéro [Després et al, 2006], réutilisation et re-ingénierie (re-engineering) d'autres ontologies, ou fusion d'ontologies [Noy et Musen , 2003].

Dans la littérature, il n'existe pas encore de consensus à propos des meilleures pratiques à adopter lors du processus de construction ou même des normes techniques régissant le processus de développement des ontologies. Lors du processus de mise au point d'une ontologie, chaque équipe de développement suit habituellement ses propres principes, ses critères de conception et ses étapes d'élaboration.

Dans la suite nous allons présenter les principales méthodes et méthodologies de développement d'ontologies.

7.4.1 La méthodologie METHONTOLOGY

METHONTOLOGY [Fernandez et al., 1997] à été développée au laboratoire d'intelligence artificielle de l'université de Madrid. Elle vise la construction d'ontologie au niveau de connaissance. Ce projet a été motivé par le constat suivant : l'absence de méthodes ou de guides structurés est un obstacle à la construction d'ontologies partagées et consensuelles. Il est également un obstacle à l'extension d'une ontologie existante ou à sa réutilisation dans d'autres ontologies. L'approche METHONTOLOGY distingue les étapes suivantes :

1) *Cadrage :* Cette étape, consiste à cerner l'étendue de l'ontologie et le domaine à prendre en compte.

2) *Conceptualisation :* Le but de cette étape est d'identifier et de structurer les connaissances du domaine en utilisant un ensemble de représentations intermédiaires semi-formelles (des tables et des graphes), faciles à comprendre par les experts du domaine et qui sont indépendants du formalisme à utiliser pour représenter l'ontologie.
3) *Implémentation:* Cette étape consiste à formaliser le modèle conceptuel obtenu dans l'étape précédente par un formalisme de représentation d'ontologie. Puis, coder l'ontologie dans un langage d'ontologie formel.

WebODE [Corcho et al., 2005] a été construit pour donner un support technique à METHONTOLOGY. D'autres outils d'ontologie et de suites d'outils peuvent être utilisés pour construire des ontologies avec cette méthodologie, par exemple, Protégé [Noy et Fergerson, 2001], OntoEdit [Sure et al., 2002], etc.

METHONTOLOGY permet de caractériser les ontologies au niveau des connaissances et insiste sur la nécessité de travailler à partir de représentations intermédiaires des connaissances lors de la phase de conceptualisation. En effet, l'absence de cette étape, à travers laquelle le domaine de l'ontologie sera représenté par des représentations semis-formelles, qui sont d'une part faciles à comprendre et d'autre part sont indépendantes de tout formalisme de représentation, cause les problèmes suivants :

- Il est difficile pour les experts du domaine et les concepteurs d'ontologies de travailler en collaborations afin de bien structurer et vérifier les connaissances collectées.
- Le codage direct, résultat de l'acquisition de connaissances, est trop abrupt, spécialement en ce qui concerne les ontologies complexes.
- Les préférences des concepteurs d'ontologies pour un certain langage conditionnent l'implémentation des connaissances acquises.
- Les personnes qui mettent au point des ontologies (qui ne connaissent pas les langages dans lesquels les ontologies sont codées ou qui ne les maîtrisent pas bien), peuvent trouver des difficultés pour comprendre les ontologies implémentées ou même d'en construire une nouvelle.
- La réutilisation de l'ontologie par d'autres applications ou bien sa réutilisation dans d'autres ontologies, nécessite souvent un processus de réingénierie afin d'aboutir à son modèle conceptuel. Ce dernier sera restructuré puis formalisé.

7.4.2 La Méthode 101

La Méthode 101 [Noy et McGuinness, 2001] cherche à construire des ontologie formelle par la reprise et l'adaptation des ontologies déjà existantes, et propose de suivre les démarches ci-après :

- Déterminer le domaine et la portée de l'ontologie ;
- Considérer la réutilisation des ontologies existantes ;
- Enumérer les termes les plus importants dans l'ontologie ;
- Définir les classes et hiérarchie des classes ;
- Définir les propriétés des classes ;
- définir les facettes des attributs ;
- Construire les instance.

Cette méthode utilise comme support les outils Protégé-2000 [Noy et Fergerson, 2001] et Ontolingua [Farquhar et al., 1997].

7.4.3 La méthode ARCHONTE

La méthode ARCHONTE (ARCHitecture for ONTological Elaborating) proposée par B. Bachimont pour construire des ontologies s'appuie sur la sémantique différentielle [Bachimont et al., 2002]. La construction d'une ontologie selon cette approche comporte trois principales étapes :

- choisir les termes pertinents du domaine et normaliser leur sens puis justifier la place de chaque concept dans la hiérarchie ontologique en précisant les relations de similarités et de différences que chaque concept entretient avec ses concepts frères et son concept père ;
- formaliser les connaissances, ce qui implique par exemple d'ajouter des propriétés à des concepts, des axiomes, de contraindre les domaines d'une relation . . .
- l'opérationnalisation dans un langage de représentation des connaissances.

7.4.4 La méthodologie On-To-Knowledge

On-To-Knowledge [Sure et al., 2006] recommande un procédé itératif de développement, et comporte quatre phases principales : une phase de spécification de condition, une phase d'amélioration, une phase d'évaluation et une phase d'application et d'évolution. On-To-Knowledge propose l'acquisition des connaissances en spécialisant une ontologie générique.

elle propose de construire l'ontologie en tenant compte de la manière dont elle sera utilisée dans d'autre applications. Par conséquent, les ontologies développées avec cette méthodologie sont fortement dépendantes de l'application. On-To-Knowledge recommande la suite d'outils OntoStudio[6] comme support de développement des ontologies.

7.4.5 Les autres méthodes de développement d'ontologies

Il existent d'autres méthodes de développement d'ontologies utilisées dans d'autre domaines. Dans ce groupe, nous pouvons citer entre autres:

- **Les méthodes Entreprise** [Uschold et Grüninger, 1996] et **Toronto Virtual Enterprise** [Gruber, 1995] dont toutes les deux cherchaient à modéliser le domaine de l'entreprise. elles sont mois détaillées et n'ont été testées que dans le domaine des affaires (business).
- **La méthode Sensus** [Swartout et al., 1997] est utilisée pour construire le squelette d'une ontologie de domaine à partir d'une grande ontologie appelée l'ontologie Sensus. La méthode propose de relier les termes spécifiques du domaine à cette ontologie et de supprimer les termes qui ne seront pas détenus dans la nouvelle ontologie qu'on souhaite construire. Le résultat de ce processus est le squelette de cette nouvelle ontologie, qui est générée automatiquement en utilisant ce processus et l'outil OntoSaurus. Ainsi, conformément à cette méthode, la construction d'une ontologie dans un domaine donné doit suivre les étapes suivantes : 1) identifier les termes clés du domaine ; 2) relier manuellement les termes clés à SENSUS et 3) inclure tous les concepts qui se trouvent sur le chemin depuis le terme clé jusqu'à la racine de SENSUS.

D'autres méthodes existent, qui ne s'occupent en fait que d'une activité précise du processus de développement. Elle peuvent être classer selon les catégories suivantes:

- **Les méthodes d'étude des ontologies ou "ontologies learning"**. Cette catégorie de méthodes s'intéresse plus particulièrement à l'activité de l'acquisition de connaissances. Acquérir la connaissances pour la construction des ontologies exige souvent beaucoup de temps et de ressources. Pour cela, on a pensé à des méthodes d'étude d'ontologie pour alléger l'effort pendant le processus d'acquisition de la connaissances. Parmi ces méthodes, nous pouvons citer la méthode de l'équipe d'Aussenac-Gilles [Aussenac-

[6] http://www.ontoprise.de/en/products/ontostudio/

Gilles et al., 2000]. Ce type de méthodes peut être utilisé dans plusieurs buts : créer une ontologie à partir de zéro ou bien enrichir une ontologie avec de nouveaux concepts.

- **Les méthodes d'alignement et de fusion des ontologies**. Les ontologies ont pour but de capturer la connaissance consensuelle d'un domaine donné d'une générique et formelle pour être réutilisée et partagée à travers des applications et par des groupes de personnes. Nous pouvons trouver dans la littérature plusieurs ontologies qui modélisent, de différentes manières, le même domaine de connaissances. Généralement, on distingue deux principales approches pour unifier les terminologies des ontologies : alignement d'ontologie et fusionnement d'ontologie. Les méthodes d'alignement d'ontologies consistent à établir différents types de correspondance entre les ontologies, par conséquent cette option préserve les ontologies originales. Nous pouvons citer la méthode par ***AnchorPROMPT*** [Noy et Musen , 2003]. Par ailleurs, les méthodes de fusionnement d'ontologies proposent de produire une nouvelle et unique ontologie à partir d'ontologies déjà existantes. Dans cette catégorie nous pouvons citer par exemple la méthode ***FCA-Merge*** **[**Stumme et Maedche, 2001].

7.4.6 Discussions et comparaisons des méthodes présentées

L'exposé des méthodes et méthodologies ci-dessus permet de distinguer deux grandes phases :

1) une modélisation pour donner du sens, autrement dit, une modélisation des connaissances ontologiques conduisant à la définition d'une ontologie conceptuelle ; 2) une modélisation pour implémenter un système conduisant à une ontologie computationnelle.

D'autres auteurs proposent un certain nombre de critères et d'étapes pertinents pour la construction d'une ontologie dynamique, interopérable, facilement maintenable et indépendante du contexte. Il n'y a pas une seule et unique manière de modéliser un domaine de connaissances, il n'y a que des alternatives plus ou moins réussies. La plupart du temps, le choix d'une méthodologie adéquate dépend des objectifs et des buts poursuivis ainsi que de l'outil de construction utilisé.

La table 1.5 donne une comparaison des méthodes et méthodologies selon la stratégie de construction d'une ontologie. Le cadre de comparaisons est basé sur les critères suivants:

- **Proposition de cycle de vie.** Le cycle de vie identifie l'ensemble des étapes à travers lesquelles l'ontologie évolue pendant sa durée de vie Il décrit également quelles activités doivent être exécutées à chaque étape et comment les étapes sont liées.

- **Stratégie selon l'application.** Ce critère est lié au degré de dépendance de l'ontologie avec l'application qui l'utilise. Considérant ce critère, les méthodes et méthodologie peuvent être classées dans les type suivant :
  - Dépendantes: les ontologies sont construites sur la base des applications qui les utilisent.
  - Semi-dépendantes: les scénarios d'utilisation d'ontologie sont identifiés dans l'étape de spécification.
  - Indépendantes. le processus développement est totalement indépendant des utilisations de l'ontologie dans des applications.
- **Stratégie pour identifier les concepts**. Il y a trois stratégies pour identifier les concepts : partant du plus concret vers le plus abstrait (bottom-up), partant du plus abstrait vers le plus concret (top-down), ou partant du plus important vers le plus abstrait et le plus concret (middle-out).
- **Ontologie noyau.** Ce critère détermine s'il est possible ou non d'utiliser une ontologie noyau comme point de départ dans le développement de l'ontologie.
- **Supports techniques.** Ce critère consiste à déterminer quels sont les outils qui donnent un support complet ou partiel à la méthode ou méthodologie.

Table 1.5- Comparaisons des méthodes selon la stratégie de construction d'une ontologie.

	SENSUS	**METHONTOLOGY**	**101**	**ARCHONTE**	**On-To-Knowledge**
Proposition de cycle de vie	Non proposé	prototypes en évolution	Non proposé	Non proposé	Incrémentale et cyclique avec prototypes en évolution
Stratégie selon l'application	Non spécifiée	Indépendante	Indépendante	Dépendante	Dépendante
Stratégie d'identification des concepts	Semi-dépendante	Middle-out	Top-down	Middle-out	Top-down, Bottom-up, Middle-out
Utilisation d'une Ontologie noyau	Oui	Dépend des ressources disponibles	Dépend des ressources disponibles	Dépend des ressources disponibles	Dépend des ressources disponibles
Supports techniques	OntoSaurus	Protégé, OntoEdit, DOE	Ontolingua, Protégé	DOE	OntoStudio, OntoEdit

8 Conclusion

Dans ce chapitre nous avons discuté des ontologies et leur utilisation en ingénierie des connaissances. Les ontologies servent à décrire formellement des concepts et des relations entre concepts. Les concepts décrivent un système donné et participent à la signification des termes. Pour matérialiser ces concepts et relations, l'ontologie doit être construite et exprimée dans un langage. Elle doit, également, satisfaire un ensemble de propriétés (consensus, cohérence, réutilisation et partage).

Les apports de l'utilisation des ontologies sont divers. Les ontologies jouent un rôle important dans les systèmes à base de connaissance. Outre la réutilisation et le partage de connaissances, elles permettent de faciliter la communication entre les acteurs de différentes organisations. Elles permettent, en particulier, la réalisation de l'interopérabilité entre différents systèmes.

Les différents avantages qu'offrent les ontologies ne doivent pas dissimuler leurs limites. La principale est la notion de *point de vue* qui reste un point sombre et peu discuté dans la littérature de ce domaine. Cette notion traduit le fait qu'un concept n'est pas considéré de la même manière suivant le contexte dans lequel il est utilisé. Par exemple dans le domaine de l'urbanisme, la notion d'espace public va se limiter aux espaces extérieurs pour certains (rus, place,...) ; pour d'autres il s'agira également des lieux bâtis accessibles au public (gare, centre commercial,...) ; enfin le concept peut également faire référence aux espaces de négociation tels que lieux de réunion, de débats publics,... Difficile, par la suite, de savoir de quoi on parle exactement puisque personne n'a raison et personne n'a tort! On voit bien que suivant le point de vue selon lequel on se place, un concept n'a pas forcément les mêmes caractéristiques. Ce problème de la multi-représentation suscite de nombreuses recherches dans le domaine de la représentation des connaissances.

Dans le chapitre suivant, nous nous intéresserons à la notion de point de vue et les différentes façons dont elle a été traitée, sous l'angle de la représentation des connaissances.

Chapitre 2

Les points de vue et leurs représentations

"*When there are many meanings in a network, you can turn things around in your mind and look at them from different perspectives ; when you get stuck, you can try another view. That's what we mean by thinking*"
Minsky

1 Introduction

Une ontologie capture et structure la connaissance dans un domaine et, ce faisant, elle capture la signification des concepts qui sont spécifiques pour ce domaine. Il existe généralement plusieurs façons d'appréhender les connaissances d'un domaine, c'est-à-dire différents points de vue selon lesquels ces connaissances peuvent être représentées. Ainsi, lors de la construction d'une ontologie, il est souvent intéressant de prendre en compte la vision de chacun des utilisateurs. En effet, différentes personnes alimentent l'ontologie ou l'exploitent. Lors de l'alimentation, chacune de ces personnes peut vouloir formaliser ses connaissances selon son propre point de vue.

D'une manière générale, la notion de points de vue s'intéresse à la variété de perceptions portées sur un même univers de discours. Par opposition à une approche mono-point de vue ou monolithique, l'approche multi-points de vue permet de modéliser une même réalité selon des points de vue différents. Un point de vue dans son sens large, étant la perception qu'un individu ou un groupe d'individus en fait d'un monde observé.

Actuellement, on s'accorde à reconnaître l'intérêt de la notion de point de vue dans la conception et le développement des applications multi-utilisateurs qui requièrent la coopération de plusieurs experts voire les futures utilisateurs, avec chacun ses intérêts propres et ses connaissances.

Dans ce chapitre, nous nous intéressons plus particulièrement à cette notion qui apparaît, avec certaines variations, sous plusieurs termes et aspects selon son abord dans différents travaux traitant le problème de la multi-représentation. Ensuite, nous nous attachons à donner un aperçu de quelques travaux proposant une formalisation et une représentation multi-points de vue des connaissances. Nous abordons aussi une notion proche de celle de points de vue, qui est celle de contexte, telle qu'elle est considérée en représentation des connaissances.

2 Notion de point de vue

Lorsqu'on parle du point de vue de quelqu'un dans le langage courant, il s'agit généralement de la position que prend cette personne concernant un objet particulier. Ces deux termes (point de vue et position) relèvent de la même image : celle du lieu où se trouve la personne vis-à-vis de l'objet et aussi de l'angle selon lequel elle le considère.

Par exemple, sur le même objet de voiture que l'on veut mettre dans une base de données, on voudra fournir une base de données unifiée pouvant servir à la fois à un carrossier, un gestionnaire de locations, un mécanicien, un électronicien et un vendeur. Chacune de ces personnes met une chose différente derrière le terme « voiture », mais on sait que c'est du même objet qu'elles parlent. Simplement, chacune d'entre elles s'intéresse à certaines de ses caractéristiques en ignorant les autres.

Le concept de point de vue a été utilisé avec des sens divers dans différents domaines de l'informatique : bases de données, représentation des connaissances, analyse et conception, outils de génie logiciel, etc. Une des premières références à la notion de point de vue se trouve dans le travail de Minsky [Minsky, 1975] avec une connotation spatiale. Pour lui, un objet peut être vu par des observateurs différents à partir de divers points de vue ; ces observateurs regardent tous les mêmes attributs mais chacun peut les voir avec des valeurs différentes selon ses propres points de vue.

Notons que la notion de point de vue est apparue simultanément et indépendamment dans différents domaines de l'informatique tels que le génie logiciel, les bases de données ou la représentation des connaissances. Cette simultanéité se reflète dans la variété des appellations tels que **perspectives**, **vue**, **rôles**, **contextes** et **points de vue**. Dans la suite nous en donnons quelques définitions recueillies de travaux de recherche sur l'intégration des points de vue dans le domaine de la représentation des connaissances,

2.1 Qu'est-ce qu'un point de vue ?

Un des premiers travaux réalisés sur l'intégration explicite de la notion de point de vue dans une représentation orientée objet de connaissances est le travail de Olga Mariño Drews au cours du projet TROPES. Dans [Marino, 1993], un point de vue (appelé également perspective) est « *la perception qu'a une personne du monde observé (ou la perception d'une personne sur le monde observé*) ». Ces travaux se placent dans le contexte d'un monde unique, vu par des agents qui en ont une représentation partielle appelée perspective ou point de vue : « *les points de vue sont des visions partielles mais complémentaires du monde* ».

La notion de point de vue est donc proche de celle de perspective proposée par Minsky qui se rapproche du sens spatial du mot perspective : il existe un objet (unique) et chaque individu le regarde selon sa position, son point de vue.

TROPES a fourni une base de réflexion à d'autres projets. C'est le cas notamment des travaux développés dans [Ribière, 1998, Ribière, 1999] dans lesquels "*un point de vue est l'interface permettant l'indexation et l'interprétation d'une vue composée d'éléments de connaissances, il est caractérisé par un point de focalisation et un angle de vue*".

Dieng-Kuntz *et al.* (2000) précisent qu'en termes de construction, le point de vue permet d'indexer des connaissances afin de les rendre accessibles, dynamiques et réutilisables et qu'en termes de consultation, il constitue un filtre permettant de n'afficher à l'utilisateur que les informations pertinentes à ses yeux. Ils expliquent aussi ce que sont le point de focalisation et l'angle de vue de la définition de Ribière. " *Le point de focalisation permet de décrire le contexte dans lequel un expert se place, ainsi que son objectif alors que l'angle de vue décrit les caractéristiques*[1] *de ceux qui s'expriment selon un certain point de focalisation*". Dieng *et al.* (2000) stipulent qu'on retrouve deux types de points de vue :

1) les points de vue définissant des vues "**perspectives**", ils indexent des descriptions consensuelles d'un même objet par différents acteurs, les vues sont alors complémentaires et forment une vision cohérente du monde ;

2) les points de vue définissant des vues "**opinions**", ils indexent des vues non consensuelles relatant chacune une approche particulière d'un acteur, les vues représentent alors indépendamment les unes des autres des visions incomplètes du monde et peuvent collectivement être inconsistantes.

Selon Bach (2006), un point de vue correspond à un contexte ou à une situation, où des connaissances à propos d'un objet, d'un concept ou d'une entité sont exprimées et considérées comme valides et vraies selon ce point de vue. Un point de vue correspond aussi à une vue, où l'on examine des caractéristiques d'un concept ou d'une entité, qui sont considérées comme des caractéristiques pertinentes du concept ou de l'entité dans cette vue. Un point de vue peut provenir d'une personne, d'un groupe de personnes, d'une communauté dans une entreprise ou une organisation.

[1] Ces caractéristiques sont par exemple : le nom d'une personne, son domaine d'application, son niveau d'expertise, etc.

Pour Gilles Falquet, un point de vue est plutôt lié à un type de personne (métier, âge, niveau de formation, etc.) ou d'utilisation (une même personne pourra avoir un point de vue différent en fonction de la tâche qu'elle cherche à accomplir). Il envisage dans [Falquet et al, 2001] plusieurs types de relations entre points de vue:

- **Points de vue par niveau** : Selon le niveau de connaissance ou de formation par rapport à un domaine, on peut être plus ou moins sensible à certains détails. On peut donc avoir plusieurs hiérarchies de concepts avec une forme similaire, mais avec plus ou moins de niveaux. Si l'on prend par exemple le concept « cheval » dans le domaine de la zoologie, on peut avoir dans le point de vue « personne ordinaire », la classification suivante : *cheval* → *mammifère* → *animal* et dans le point de vue « zoologue » : *cheval* → onguligrade *avec nombre de doigts impair* → *onguligrade* → *mammifère* → *animal.* Le point de vue « personne ordinaire » représente donc un sous-ensemble du point de vue « zoologue ».
- **Point de vue = vue partielle :** Chaque type d'utilisateur va s'intéresser à un ensemble restreint de caractéristiques. Par exemple pour le domaine des voitures, dans le point de vue «mécanicien» les véhicules vont être définis selon le type de moteur, le diamètre des roues, etc. tandis que dans le point de vue « vendeur », les définitions vont plutôt contenir des informations sur le type de véhicule (tourisme, utilitaire, transport de groupes), la taille, la catégorie de prix, etc. Ce type de points de vue peut être envisagé comme une projection d'un espace vers un espace de plus petite dimension.
- **Points de vue avec chevauchement :** Des personnes travaillant dans des domaines voisins peuvent avoir des points de vue qui se recoupent. Par exemple si l'on considère le domaine des immeubles, on peut supposer qu'une partie des définitions de concepts sera identique pour un architecte et pour un ingénieur en génie civil, alors que certains autres concepts seront définis différemment (on peut aussi penser qu'une partie des concepts n'existeront que dans l'un ou l'autre des points de vue).

Enfin, dans [Benchikha, 2007] un point de vue est considéré comme une position conceptuelle mettant en liaison d'une part un acteur qui observe et d'autre part un phénomène (ou un monde) qui est observé. Les acteurs observant un même univers de discours peuvent alors avoir des points de vue complémentaires (ils ont des visions partielles mais complémentaires).

Pour terminer, nous considérons qu'un point de vue correspond donc à une catégorie d'utilisateurs du domaine ou bien au point de vue que peut prendre un utilisateur à un moment donné. En effet, lors de la modélisation de connaissances sur un objet du monde réel, les experts explicitent leurs connaissances selon leur vision sur cet objet en fonction de leur objectif et leur intérêt immédiat.

2.2 Pourquoi les points de vue ?

La notion des points de vues a été principalement utilisée pour facilité la conception des systèmes complexes. Ainsi, un point de vue peut être défini comme un objet local qui encapsule des connaissances partielles du système et du domaine [Finkelstein et Sommerville, 1996].

Benchikha, *et al.* (2005) précisent que les points de vue sont de plus en plus utilisés ces dernières années dans le développement de systèmes informatiques. Les objectifs visés par l'utilisation de points de vue sont multiples. Il est à noter qu'aucune utilisation de ce concept n'inclut l'ensemble des objectifs identifiés dans [Benchikha, 2007]. Ces objectifs sont :

1) Le point de vue comme un moyen de description multiple des entités. Le concept de point de vue semble résulter naturellement de la multiplicité des regards portés sur les objets d'un domaine d'étude. En effet, une entité du monde réel peut avoir plusieurs contextes comportementaux et plusieurs états d'où la notion de « description multiple ». Cette dernière confère à un même univers de discours plusieurs descriptions partielles. Chaque description fournit un point de vue. Les descriptions partielles sont complémentaires et produisent une description complète et cohérence des entités du monde réel.

2) Le point de vue comme un moyen pour maîtriser la complexité des systèmes. L'idée de diviser une représentation en modules représentant chacun un point de vue peut apporter beaucoup. Elle permet de construire une représentation relativement complexe à partir de briques de base indépendantes les unes des autres. Ainsi, l'un des objectif du point de vue est de présenter une version simplifiée de la connaissance. Cela permet de diminuer la charge pour l'utilisateur comme pour un programme.

3) Le point de vue comme une approche de modélisation et de développement décentralisé des systèmes. Plusieurs auteurs estiment que la modélisation des systèmes surtout complexes tels que définis dans ne peut s'appréhender avec les mêmes techniques que pour les petits systèmes dont le développement est maîtrisable par un nombre restreint de

personnes. De ce fait, différents travaux proposent donc une approche répartie de développement fondée sur les points de vue.

4) point de vue comme mécanisme avancé pour les technologies orientées objet. La technologie orientée objet a apporté un net progrès dans la modélisation des systèmes complexes par son pouvoir d'expression et sa meilleure réutilisabilité. Cependant, elle s'est vite accompagnée de l'émergence de nouveaux besoins tels que la nécessité de distinguer les données en provenance de différents concepteurs ou encore de l'évolution des objets ainsi que leur (re)classification multiple et dynamique. La rigidité d'un objet tant en ce qui concerne son état que son comportement a été très vite remise en cause via les perspectives de KRL [Bobrow et Winograd, 1977] et les points de vue dans TROPES [Mariño, 1993].

5) point de vue comme mécanisme pour résoudre des problèmes. Le concept de point de vue apporte des solutions simples et satisfaisantes à des problèmes épineux posés dans les différents domaines de l'informatique. En représentation des connaissances, par exemple, le point de vue est introduit dans la classification multiple des objets, dans la recherche de valeurs héritées, dans la modélisation de concepts indépendants et dans le traitement des conflits d'héritage multiple.

Dans les sections suivantes, nous abordons quelques travaux significatifs intégrant le concept de point de vue dans le domaine de la représentation des connaissances. En fait, c'est en représentation des connaissances que la notion de point de vue a tout d'abord été considérée. Nous présentons ci-dessous quelques propositions et des mécanismes permettant de la prendre en compte.

3 Les points de vue en représentation des connaissances

Il existe généralement plusieurs façons d'appréhender les éléments de connaissances relatifs à un domaine, c'est-à-dire différents points de vue ou perceptions selon lesquels ces connaissances peuvent être représentées. Par exemple, en fonction de la tâche à accomplir, une voiture sera considérée de différentes façons ou points de vue : l'intérêt pourra porter sur sa consommation et sur son prix d'achat selon le point de vue économique, sur le nombre de places et la taille du coffre si c'est l'aspect fonctionnel qui l'emporte, ou encore sur la couleur et la forme s'il est plus opportun de se focaliser sur l'esthétique de la voiture. Les travaux en représentation des connaissances se sont ainsi souvent intéressés à faire coexister plusieurs représentations alternatives dans une même base de connaissances, en intégrant de façon explicite la notion de point de vue.

Cette section donne un aperçu des grandes tendances dans les systèmes et les langages de représentation à points de vue multiples. Les travaux portant sur les points de vue et plus en rapport avec la programmation par objets ou les bases de données ne sont pas abordés ici.

3.1 Points de vue en représentation des connaissances par objets

C'est en représentation des connaissances par objets (RCO) [Masini *et al.*, 1989; Euzenat, 1998] qu'un grand nombre de systèmes de représentation de connaissances multi-points de vue a été développé. La représentation des connaissances par objets repose généralement sur la notion de *classe* pour représenter des ensembles d'objets. Les classes se voient associer des *attributs* et sont organisées en une hiérarchie. Une classe dans cette hiérarchie *hérite* les attributs de toutes ses superclasses. Certains systèmes autorisent qu'une classe ait plusieurs superclasses directes (héritage multiple). Un *objet* est instance d'une ou de plusieurs classes et hérite les attributs de cette ou de ces classes.

Dans les sous sections qui suivent nous discutons quatre systèmes qui exemplifient assez bien les différentes façons de représenter explicitement les points de vue dans les représentations par objets : KRL, LOOPS, TROPES et ROME.

3.1.1 KRL

Knowledge Representation Language, KRL, est l'un des premiers système destiné à représenter des connaissances avec Frames [Masini et al., 1989] et à reconnaître qu'un objet eut être considéré de plusieurs façons, selon le point de vue de l'observateur. Chaque point de vue est représenté dans un objet séparé, appelé *perspective*, de sorte que l'objet tout entier apparaisse comme agrégation de toutes ses perspectives.

Dans KRL il y a sept types différents d'unités : *Basic*, *Specialization*, *Individuals*, *Abstract*, *Manifestation*, *Relation* et *Proposition*. Les trois premières permettent le traitement des perspectives. Les classes *Basic* sont des racines des graphes des différents familles d'objets (tels que *Personne* et *Ecole*) qui établissent une première partition de l'univers du discours. Les classes *Specialization* sont des sous-classes d'une classe *Basic* ou d'une autre classe *Specialization.* Les classes *Individuals* décrivent des entités réelles du monde (i.e. des individus).

Les perspectives sont des notions centrales à la représentation des connaissances en KRL. Un objet peut être défini par plusieurs perspectives qui correspondent à autant de points de vue selon lesquels il peut être considéré [Bobrow et Winograd, 1977]. KRL les modélise avec le descripteur Perspective. Un individu a une première Perspective qui est la classe la

plus générale à laquelle il appartient, une unité de type Basic et il peut avoir d'autres perspectives parmi les unités de spécialisation de sa classe de base. Ainsi, en figure 2.1, la classe de base est la classe Membre qui peut être spécialisée en diverses unités, telles que Professeur, Chercheur, Assistant, Droit, Science, etc. L'unité individuelle, Benmouhamed, est un membre qui, selon la perspective Assistant est un assistant de classe A et, selon la perspective Droit est un enseignant de Droit public.

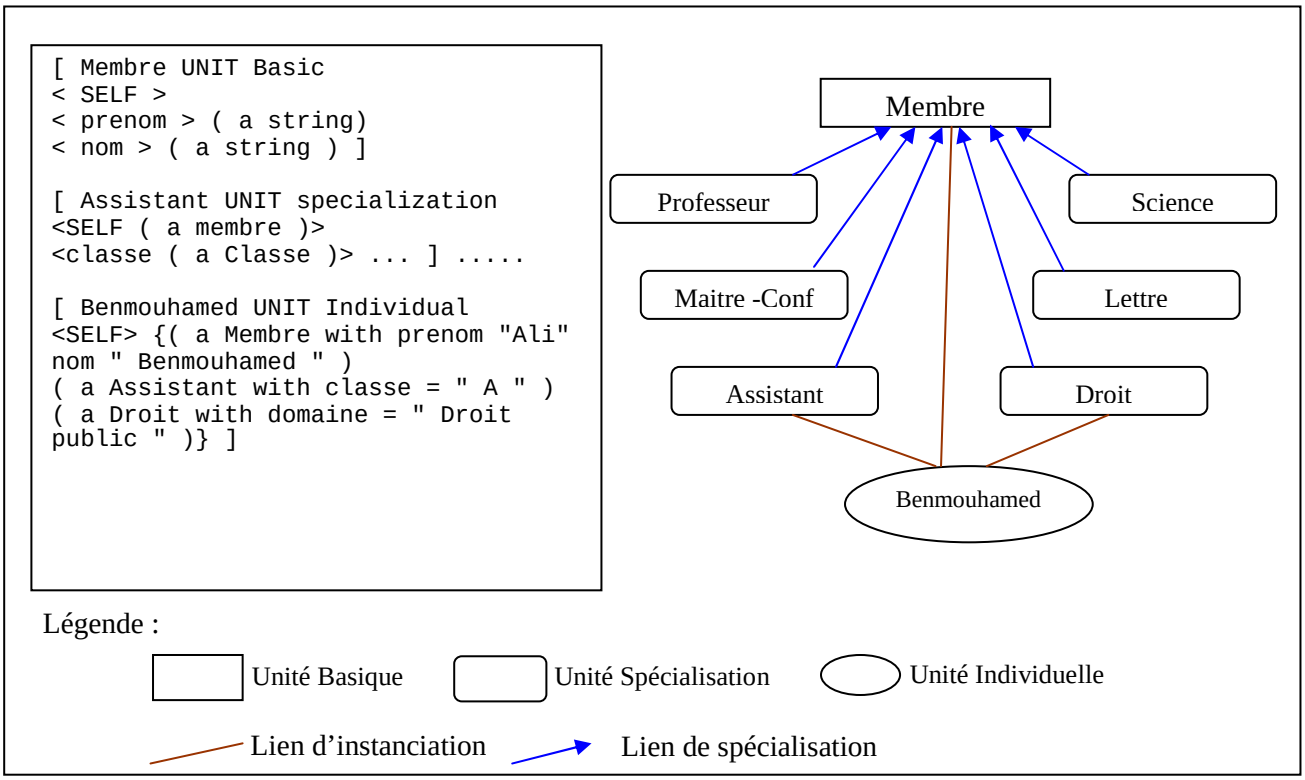


Figure 2.1. Les perspectives dans KRL

3.1.2 LOOPS

Alors que dans le modèle de KRL, une perspective d'un objet est un sous-ensemble d'attributs, dans le modèle LOOPS [Stefik et Bobrow, 1985], une perspective est un objet à part entière qui peut recevoir directement des messages.

Pour traiter les perspectives, LOOPS utilise deux classes abstraites appelées les *mixin* : Node et Perspective. Une classe qui décrit des objets ayant des points de vue, est une sous-classe de Node, alors qu'une classe décrivant des objets qui sont des perspectives d'autres objets doit être définie comme sous-classe de la classe Perspective. En figure 2.2, la classe Membre est une sous-classe du mixin Node à laquelle on associe des sous-classes de Perspective (Assistant et Droit) qui décrivent les perspectives d'un objet instance de Membre.

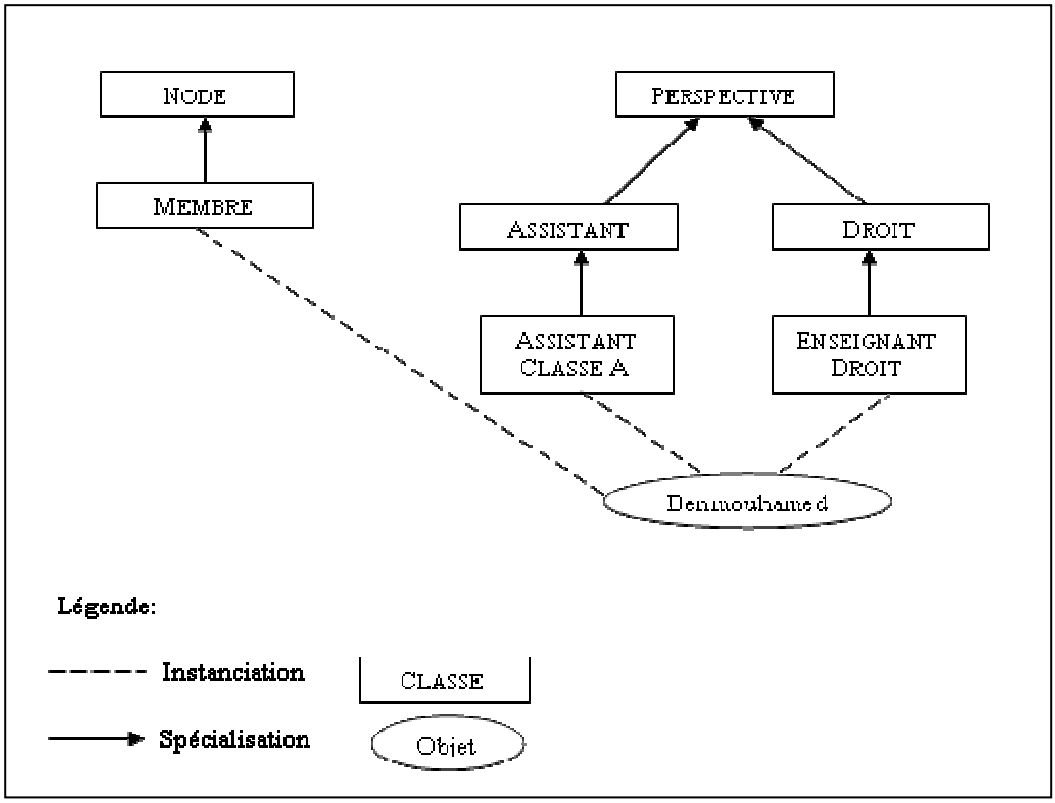


Figure 2.2- Perspectives d'un objet dans LOOPS

Les perspectives d'un objet LOOPS sont des objets indépendants, instances des sous-classes du mixin Perspective. L'objet même est membre d'une sous-classe du mixin Node. Un objet et ses perspectives sont une sorte d'objet composite, les composants étant les différentes perspectives. Chaque perspective est un objet indépendant pouvant recevoir des messages, ce qui autorise d'avoir des attributs de même nom avec des sens différents dans plusieurs perspectives.

En reprenant l'exemple précédent, l'individu Benmouhamed est lié à ses deux perspectives: Benmouhamed comme un assistant de classe A et Benmouhamed comme un enseignant de droit public. A la différence des objets composites, les perspectives sont créées dynamiquement à la demande. Ainsi, pourrait-on ajouter ensuite une perspective Genre, divisant les membres en deux classes, et considérer Benmouhamed comme Homme.

La modélisation des points de vue multiple dans LOOPS apporte de nouvelles idées sur le sujet. D'une part, le traitement indépendant de perspectives permet de regarder un objet selon un point de vue sans être saturé par l'information des autres points de vue. D'autre part, la dynamique permise par cette indépendance au niveau de la création et de la modification des

perspectives permet de stocker une instance dans la base et de la manipuler sans avoir l'information complète concernant les autres points de vue.

3.1.3 TROPES

Les travaux de thèse de Olga Mariño Drews, réalisés dans le cadre du projet TROPES, ont permis d'intégrer la modélisation de points de vue dans un modèle de représentation des connaissances par objets, le modèle TROPES [Mariño, 1993]. Dans son approche, à part des éléments classiques de la de représentation des connaissances par objets tels que classe, instance, attributs et facettes, TROPES emploie deux autres en plus : le ***concept*** et le ***point de vue*** (*cf* Figure 2.3). Chaque concept modélise une famille générique d'objets (objets dans le monde réel), et regroupe un ensemble d'instances qui n'appartiennent qu'à ce concept. Un concept est partitionné en différents points de vue sous lesquels chaque instance peut être considérée. Les points de vue d'un concept permettent d'attacher à une instance des considérations différentes. La consultation d'une instance dans un point de vue particulier permet d'obtenir les informations particulières à ce point de vue.

Dans chaque point de vue, les **classes** sont organisées en une hiérarchie de spécialisation et reliées par le lien '**sorte-de**'. Une classe fournit en intention une description d'instances particulières du concept; l'ensemble de ses instances constitue son extension. Une classe n'ayant qu'une seule sur-classe, dans un point de vue donné, l'héritage est simple. Les extensions des sous-classes directes d'une classe sont supposées être disjointes deux à deux et leur union être incluse dans l'extension de la classe. La classe-racine de chaque point de vue doit décrire toutes les instances du concept. Une instance est attachée *directement* à une seule classe dans chaque point de vue par le lien d'instanciation '**est-un**', mais appartient à chaque classe située sur le chemin allant de la classe-racine de la hiérarchie à cette classe. Une instance est rattachée à une classe dans chaque point de vue.

Entre des classes appartenant aux différents points de vue d'un même concept, il y a des liens, appelés passerelles. Cela permet d'exprimer des relations ensemblistes entre des ensembles d'instances possibles des classes de points de vue différents, et permet d'effectuer des raisonnements entre des points de vue tels que la vérification de la cohérence (une instance définie dans un point de vue doit satisfaire toutes les contraintes de ce point de vue, et aussi celles définies dans un autre point de vue si ces deux points de vue sont connectés par une passerelle).

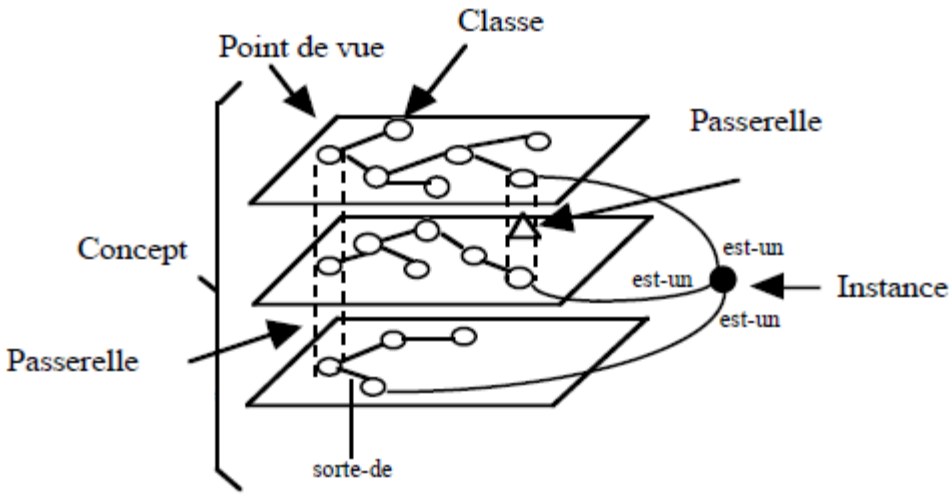


Figure 2.3 - Les principales entités de TROPES

Les travaux du projet TROPES sur l'intégration des points de vue au sein d'une base de connaissances orientée objet ont souvent fourni une base de réflexion pour d'autres apports. C'est le cas notamment des travaux développés dans [Ribière, 1998, Ribière, 1999] et le système KASIMIR [d'Aquin *et al.*, 2004]. Dans le premier cas, l'auteur utilise le formalisme des graphes conceptuels pour représenter et faire cohabiter différents points de vue d'experts sur un même sujet. Le système d'aide à la décision en cancérologie KASIMIR s'intéresse plus particulièrement à la représentation explicite de points de vue, utile en particulier pour prendre en compte les connaissances provenant de différentes disciplines de la cancérologie. En effet, la cancérologie est un domaine par essence pluridisciplinaire, faisant intervenir des médecins de différentes spécialités telles que la chirurgie, la radiothérapie, la chimiothérapie, etc. Cette diversité d'expertise entraîne une hétérogénéité et une complémentarité dans les connaissances. Ce travail a d'abord été étudié dans un cadre de la représentation par objets [d'Aquin *et al.*, 2004] puis a été implanté dans le cadre des logiques de descriptions distribuées et de C-OWL [d'Aquin *et al.*, 2005].

3.1.4 Synthèse

Les approches KRL, LOOPS et TROPES illustrent différentes solutions apportées à la gestion des points de vue, dans le domaine de la représentation des connaissances. Ces modèles reposent sur l'hypothèse qu'un point de vue est une représentation partielle d'un ensemble cohérent d'objets.

Nous pouvons comparer ces approches en fonction d'un ensemble de caractéristiques :

- La portée d'un point de vue. Dans les modèles de représentation de connaissances LOOPS et KRL, une perspective est uniquement considérée sur un objet. Dans le modèle TROPES, la notion de point de vue porte sur un objet et aussi sur une classe.
- Le type de représentation. Dans le langage KRL, la représentation multiple n'est pas complètement décentralisée. En effet, si les attributs d'un objet sont répartis dans des groupes tels que chaque groupe est relatif à une perspective, il n'est néanmoins pas possible d'accéder à une perspective indépendamment de l'objet. Par contre, dans LOOPS, nous pouvons considérer que la représentation multiple des objets est décentralisée, car une perspective d'un objet est un objet à part entière. Dans le modèle TROPES, la représentation multiple des objets n'est pas totalement décentralisée ; l'état de l'objet contient l'union des attributs décrivant l'entité dans tous les points de vue.
- A la différence de KRL et LOOPS, la notion de point de vue dans TROPES autorise l'expression de différentes taxonomies sur le même ensemble d'objets et permet leur confrontation. Cette confrontation est partiellement autorisée par la notion de passerelle dénotant par exemple l'équivalence entre classes de différents points de vue.

Le tableau ci-dessous, résume les principales caractéristiques des modèles en représentation des connaissances étudiés dans cette section.

Table 2.1- Comparaison d'approches en représentation de connaissances

Approche	Modèle de base	Portée d'un point de vue	Partage d'attributs entre points de vue
KRL	Modèles de frames	Objet	Non
LOOPS	Modèle de frames	Objet	Non
TROPES	Modèles de classes sans héritage multiple	Objet / classe	Oui

3.2 Points de vue et graphes conceptuels

Myriam Ribière propose un modèle conceptuel, nommé C-VISTA, pour la représentation multi-points de vue des connaissances basés sur le formalisme des graphes conceptuels [Ribière, 1999]. Nous rappelons qu'un graphe conceptuel est un graphe dont les nœuds peuvent être de deux types :

a. des concepts, un concept pouvant être décrit lui même par un graphe conceptuel s'il est complexe,
b. des relations conceptuelles (relations entre concepts).

Les nœuds du graphe de concept selon le modèle C-VISTA peuvent être des nœuds spécialisant le concept selon un point de vue. Ces nœuds sont appelés des nœuds point de vue. Les concepts qui se situent au-dessous d'un nœud point de vue sont dits « *orientés points de vue* ». Les concepts qui se situent au-dessus sont dits « *basiques* » (*cf.* Figure 2.4). Par ailleurs, un type de concept donné peut être à la fois basique et orienté point de vue (exemple en Figure 2.5).

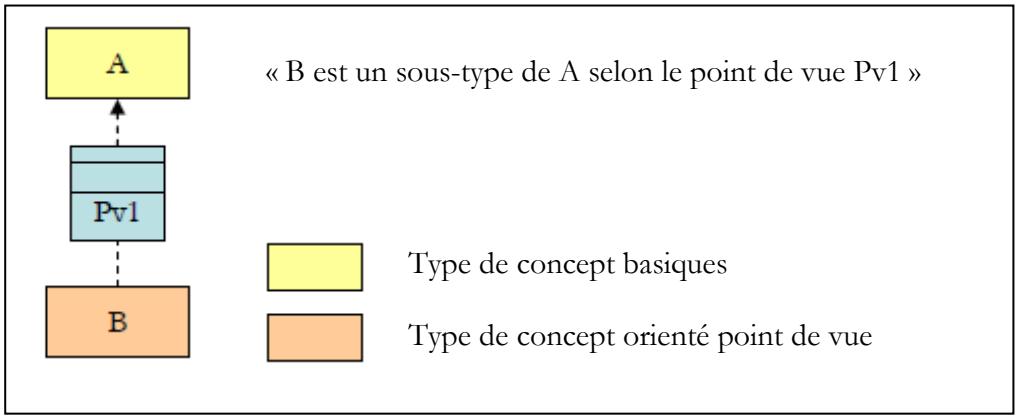


Figure 2.4 - Concepts selon C-VISTA

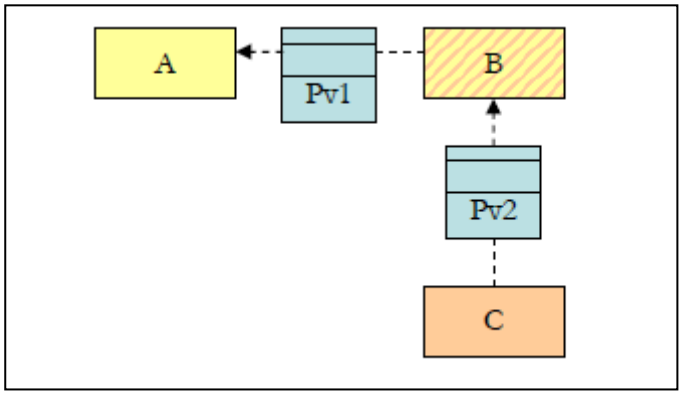


Figure 2.5 - Concepts selon C-VISTA(2)

La relation spécialisant un concept selon un point de vue peut être de deux sortes :

- la relation « perspective » qui spécialise les concepts dont la définition est consensuelle (partagée par les experts d'un même domaine ou de domaines différents),
- la relation « opinion » qui référence les concepts dont la définition est non consensuelle (non partagée par l'ensemble des experts d'un même domaine ou de domaines différents).

Dans le modèle C-VISTA, un point de vue est divisé en deux parties : le *focus* découle du contexte de travail de l'expert, de sa tâche et de ses objectifs. Plusieurs experts peuvent partager le même focus. *L'angle de vision*, au contraire, est propre à chaque expert, et représente ses capacités et compétences (*cf.* Figure 2.6).

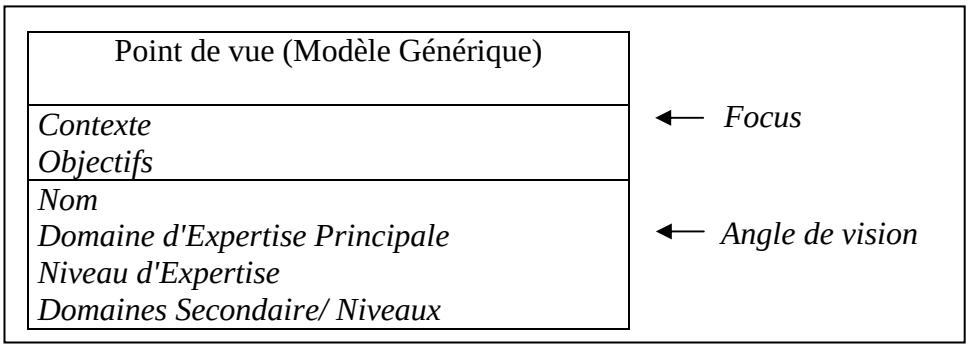


Figure 2.6 - Template d'un point de vue selon C-VISTA

C-VISTA permet l'expression de points de vue multiples au sein des hiérarchies qui forment le support des graphes conceptuels. Avoir plusieurs points de vue aboutit à la formation de terminologies différentes [Ribière, 2002]. Le modèle C-VISTA permet de tisser des liens entre ces terminologies. Ces liens sont de trois types :

- **équivalence** : les deux points de vue ont le même concept, mais l'appellent différemment. Conséquence : si l'un des concepts peut être utilisé comme représentation, l'autre aussi ;
- **inclusion** : Les deux points de vue n'ont pas la même granularité dans leur analyse, et certains concepts de l'un peuvent être regroupés dans un concept de l'autre. Si le concept inclus peut être utilisé comme représentation, alors nécessairement l'incluant aussi ;

- **exclusion** : Les deux points de vue ont des concepts qui ne peuvent en aucun cas être utilisés simultanément sur une même instance. Donc, si l'un des concepts peut être utilisé comme représentation, alors nécessairement l'autre non.

Notons que ces liens sont tissés par une équipe formée des experts et d'un cogniticien. Il s'agit donc d'une démarche d'intégration supervisée des connaissances. L'objectif est d'obtenir un modèle unique contenant les différents points de vue.

3.3 Points de vue dans les bases de données à objets

Le modèle objet MVDB (*MultiView DtaBase*) [Benchikha *et al.*, 2005, Benchikha, 2007] est proposé comme extension du modèle objet à base de classes avec le concept de point de vue dans le cadre des bases de données à objets. MVDB permet de considérer le schéma d'une base de données comme une spécification qui tient compte de plusieurs points de vues. Chaque point de vue représente un aspect de la description des données et est détenu par une base de données indépendante dite "partielle". Il s'agit donc d'un développement fondé sur l'élaboration décentralisée (répartie) de bases de données représentant un même univers de discours.

Le modèle MVDB adopte les principes de modélisation suivants:

- Le schéma d'une base de données est une description multiple d'un même univers de discours selon différents points de vue. Un point de vue est une abstraction d'une certaine perception (vision) des objets du monde. Il ne correspond pas à une classe mais c'est une spécification d'une hiérarchie de classes appelée schéma Point de Vue (*schéma-PV*). Le schéma d'une base de données est donc composé d'un ensemble de schémas point de vue et est appelé schéma *multi-point de vue*. Chaque schéma-Point de vue décrit un aspect de la description des données et est détenu par un système de base de données indépendant.
- La construction des schémas-PV est basée sur un schéma de base appelé le *référentiel*. Ce dernier détient les propriétés de base des entités de l'univers de discours et qui sont partagées par tous les schémas-PV. Chaque schéma-PV étend la description de tout l'ensemble ou d'un sous ensemble des entités du référentiel selon un point de vue donné.
- La représentation multiple est décentralisée afin de permettre une liberté de spécification des différentes représentations.
- Les différentes représentations partielles peuvent partager les informations via des relations de dépendances qui permettent à une représentation d'accéder à certaines informations spécifiées dans une autre.

- Les objets qui sont les instances dans les différents points de vue contiennent les données du domaine d'étude. Un objet peut être représentatif dans plusieurs points de vue. Le lien entre les différentes instances d'un objet est réalisé à travers la notion d'identité d'objet. Un objet est identifié d'une manière unique dans tous les schémas-PV.
- Les objets de la base de données ont une description de base dans le référentiel et une ou plusieurs descriptions selon les différents points de vue, on distingue un objet multi-points de vue et un objet point de vue respectivement.

3.4 Contexte et point de vue

La notion de contexte dans le domaine de la représentation des connaissances est très proche de la notion de point de vue [Bach, 2006]. Le travail dans [Homola et al., 2010] montre trois différentes formes du contexte dans la représentation des connaissances : (1) un contexte correspond à une partie, une vue partielle du domaine. La représentation des connaissances dans un contexte ne couvre donc qu'un sous-ensemble de connaissances dans le domaine ; (2) un contexte est une approximation. Les représentations des connaissances dans différents contextes ont différents niveaux d'approximation, différents niveaux de granularité ou d'abstraction ; (3) un contexte correspond à une perspective. Les représentations des connaissances dans différents contextes, donc différentes perspectives, dépendent des éléments extérieurs tels que le lieu, le moment… En ce sens, il semble que la notion de contexte soit plus générale que celle de point de vue. De ce fait, les travaux concernant la construction du modèle de représentation multi-contextes peuvent aussi être exploités pour représenter de multi-points de vue.

Paolo Bouquet et son équipe donnent dans [Bouquet *et al.*, 2003] une classification des théories du contexte (manière de prendre en compte le contexte). C'est ainsi qu'ils distinguent deux approches :

- la première voit le contexte comme étant un moyen de diviser la théorie globale du domaine lui donnant ainsi une structure interne plus articulée,
- la seconde voit le contexte comme une théorie locale (une représentation partielle) en relation avec d'autres théories locales.

La première vue, appelée aussi *«**diviser pour régner : LoC (Local Context)**»*, suppose l'existence d'une théorie globale du domaine. Cette théorie possède une structure interne articulée en une collection de contextes locaux. La seconde vue appelée par contre

«***composer pour régner***», suppose l'existence de plusieurs théories locales, chacune d'elle représente un point de vue différent du domaine.

3.4.1 Ontologie, contextes et points de vue

Les travaux sur la prise en compte des points de vue multiples au sein des ontologies ont pris deux directions principales. Dans la première, un point de vue est représenté dans une seule ontologie liée à d'autres ontologies relatives aux autres points de vue [Borgida et Serafini, 2003, Bouquet et al., 2004, Bach, 2006]. La deuxième tend plutôt à intégrer les points de vue dans une même ontologie [Benslimane et al., 2006]. Nous discutons ici les travaux basés sur les logiques de descriptions.

3.4.1.1 Logiques de Descriptions Distribuées

Borgida et Serafini proposent une Logique de Descriptions Distribuée (LDD) [Borgida et Serafini, 2003], qui généralise la logique de descriptions, avec une sémantique à modèles locaux pour représenter et raisonner au sujet des bases de connaissance (ontologies) dans les environnements distribués. Nous rappelons qu'une base de connaissances (BC) construite par un langage de concepts se compose généralement de deux niveaux de composantes : terminologique (TBox) et assertionnel (ABox). Dans une LDD, chaque ontologie correspond à une théorie de logique de description (TBox) et des ontologies sont liées par les mappings sémantiques. Quelques définitions de LDD seront brièvement résumées comme dans [Borgida et Serafini, 2003].

Syntaxe

Un réseau d'ontologies en DDL est composé de diverses TBox en logique de description, dont la syntaxe a été présentée au chapitre 1. Par ailleurs, les ontologies sont reliées entre elles en utilisant des règles de pont. Celles-ci sont représentées, dans la syntaxe abstraite comme suit.

Définition 1 (règle de pont). *Soient O_i et O_j deux ontologies. Une règle de pont de O_i vers O_j ($i \neq j$), est une expression de l'une des formes suivantes :*

- *$i:X \xrightarrow{\sqsubseteq} j:Y$ est une règle "intra" (into-bridge rule) ;*
- *$i:X \xleftarrow{\sqsupseteq} j:Y$ est une règle "extra" (onto-bridge rule) ;*
- *$i:a \longrightarrow j:b$ est une correspondance d'individus (individual correspondence).*

où $i:X$ et $j:Y$ sont soit des concepts soit des rôles de O_i et O_j respectivement et $i:a$ est un individu de O_i et $j:b$ est un individu de O_j .

Il est à noter qu'à l'origine, les règles de pont étaient conçues pour associer des concepts entre eux, ou bien des rôles entre eux, mais les derniers travaux sur DDL [Ghidini et Serafini, 2006, Ghidini *et al.*, 2007] permettent dorénavant d'associer un rôle et un concept et inversement. Dans ce cas, on parle de règles de pont hétérogènes.

Définition 2 (TBox distribuée) *une TBox distribuée (TBD)* $\mathcal{T} = (\{\mathcal{T}_i\}_{i \in I}, \mathcal{B})$ *se compose d'une collection des TBox* $\{\mathcal{T}_i\}_{i \in I}$ *et d'une collection des règles de pont* $\mathcal{B} = \{B_{ij}\}_{i \neq j \in I}$ *entre elles.*

Sémantique

Dans un réseau d'ontologies en DDL, on affecte à chaque ontologie une interprétation en logique de description. Il peut exister différentes logiques de description à chaque nœud du réseau. Pour relier les connaissances de deux ontologies, DDL utilise les relations de domaines.

Définition 3 (Relation de domaine). *Une relation de domaine* r_{ij} *de* Δ^{Ii} *à* Δ^{Ij} *est un sous ensemble de* $\Delta^{Ii} \times \Delta^{Ij}$ *. On note :*

$$r_{ij}(d) = \{d' \in \Delta^{Ij} \mid (d, d') \in r_{ij}\}$$

$$r_{ij}(D) = \cup_{d \in D} r_{ij}(d)$$

$$r_{ij}(R) = \cup_{(d, d') \in R} r_{ij}(d) * r_{ij}(d')$$

avec $D \subseteq \Delta^{Ii}$ et $R \subseteq \Delta^{Ii} \times \Delta^{Ij}$.

La relation de domaine représente la possibilité de l'envoi des individus de Δ^{Ii} à Δ^{Ij} du point de vue de l'ontologie O_j .

3.4.1.2 C-OWL : Context Ontology Web Language

C-OWL est une extension du langage OWL (*Cf.* Chapitre 3) pour la représentation d'ontologies contextualisées (ou contextuelles) [Bouquet *et al.*, 2004]. Les ontologies contextualisées sont des représentations locales, appelées contextes, en relation avec d'autres contextes au travers d'appariements (*cf.* Figure 2.7). En C-OWL, les connaissances sont contenues dans un ensemble de contextes, appelé espace de contextes. Chaque contexte de cet espace est une ontologie en OWL, possédant son propre langage et sa propre interprétation. Les appariements sont exprimés sous la forme de "passerelles" (*bridge rules en anglais*). Une passerelle entre deux ontologies O_i et O_j permet de déclarer une correspondance entre les éléments de connaissances de ces deux contextes. Sur la base de ces

correspondances, une partie des connaissances contenues dans O_i peut être interprétée et réutilisée dans O_j. Formellement, un espace de contextes contient un ensemble de contextes $\{O_i\}_{i \in I}$, I étant un ensemble d'index de contextes. Les index de I sont utilisés pour préfixer les expressions en OWL, associant ainsi chaque expression au contexte auquel elle appartient. Par exemple, i:C, i:∃ r.C, i:C ⊑ D, et i:C(a) sont des exemples d'expressions du contexte O_i.

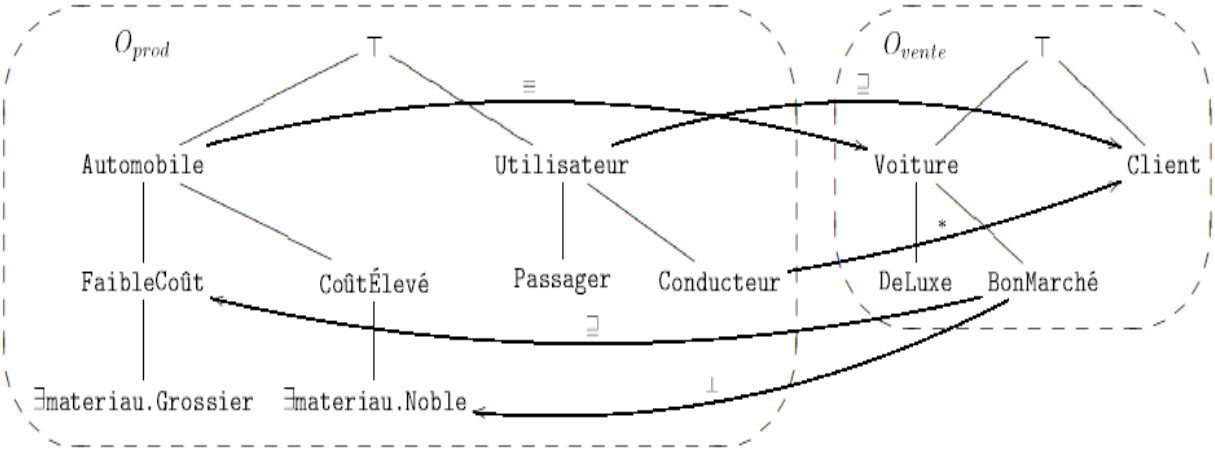


Figure 2.7- Exemple d'appariement entre deux contextes avec C-OWL [d'Aquin, 2005]

Il est à noter qu'une passerelle en C-OWL indique une relation directionnelle, établie du point de vue d'une ontologie, et non une assertion globale partagée par les ontologies mises en commun. Ainsi une correspondance C-OWL peut établir que, selon O_2, le concept A_1 de l'ontologie O_1 est perçu comme plus spécifique que le concept B_2 de O_2, mais cela n'indique pas que O_1 perçoit B_2 comme un sous-concept de A_1.

Syntaxe RDF/XML

C-OWL est une extension de OWL et doit à ce titre fournir une syntaxe qui soit, le plus possible, compatible avec celle de OWL. Chaque contexte d'un espace de contextes est représenté sous la forme d'une ontologie classique en OWL. La syntaxe RDF/XML standard de OWL est donc suffisante pour cela. Par ailleurs, dans cette syntaxe, la notion d'*espace des noms* est utilisée pour regrouper les identificateurs selon leur ontologie d'appartenance. Par exemple, dans l'URI

```
http://www.voiture.com/production#Automobile
```

La partie http://www.voiture.com/production fait référence à un espace des noms, celui de l'ontologie utilisée par exemple par les usines de production de voiture. Automobile dans ce cas est l'identificateur de la classe des voitures au sein de cet espace des noms. de cette façon, une autre ontologie pourra elle aussi utiliser l'identificateur Automobile, dans un autre espace des noms, sans confusion possible. Ce mécanisme est utilisé en C-OWL pour représenter les index des contextes. Par exemple, l'URI http://www.voiture.com/production#Automobile, pourra correspondre à la classe *prod:Automobile* du contexte O_{prod} et http://www.voiture.com/vente#Voiture à la classe *vente:Voiture* définie dans le contexte O_{vente}.

En suivant le même principe du langage C-OWL, Bach Thành Lê dans [Bach, 2006] propose un modèle de représentation des connaissances multi-points de vue, appelé MVP [2] (cf. Figure 2.8), et un langage d'ontologie multi-points de vue, qui est une extension du langage d'ontologie OWL, pour modéliser la sémantique d'un ensemble des connaissances produites indépendamment les unes des autres dans des contextes différents, où chaque contexte représente un point de vue différent du domaine. Pour cela, *Bach*, (2006) suppose que ces connaissances forment un réseau où à chaque nœud se trouve une ontologie, reliée aux autres ontologies par des correspondances formant donc des alignements d'ontologies (*cf.* Figure 2.9).

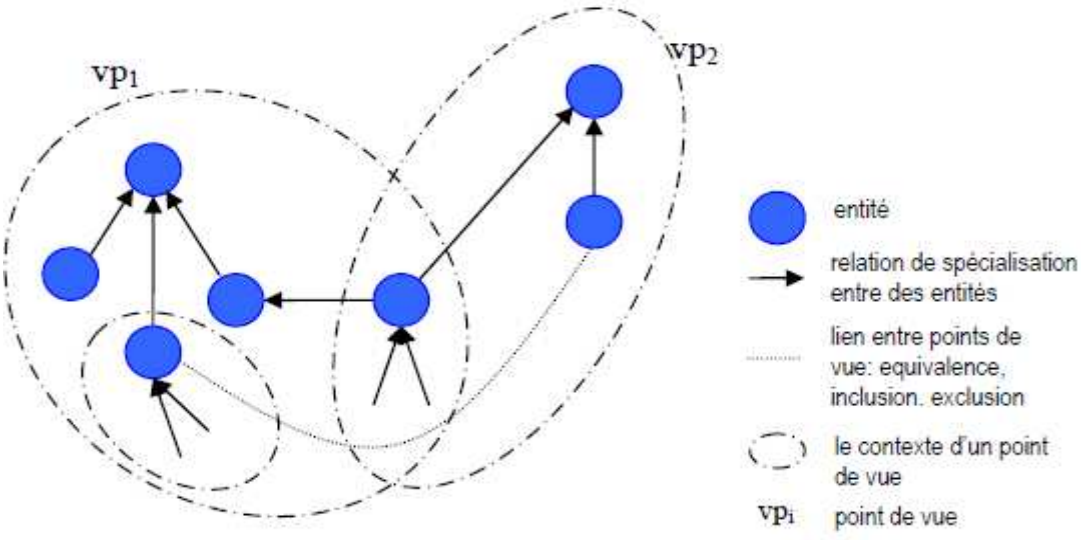


Figure 2.8 - Modèle multi-points de vue MVP par [Bach, 2006]

.

[2] Le modèle MPV est conçu pour représenter des connaissances dans une entreprise dans le cadre du Web sémantique.

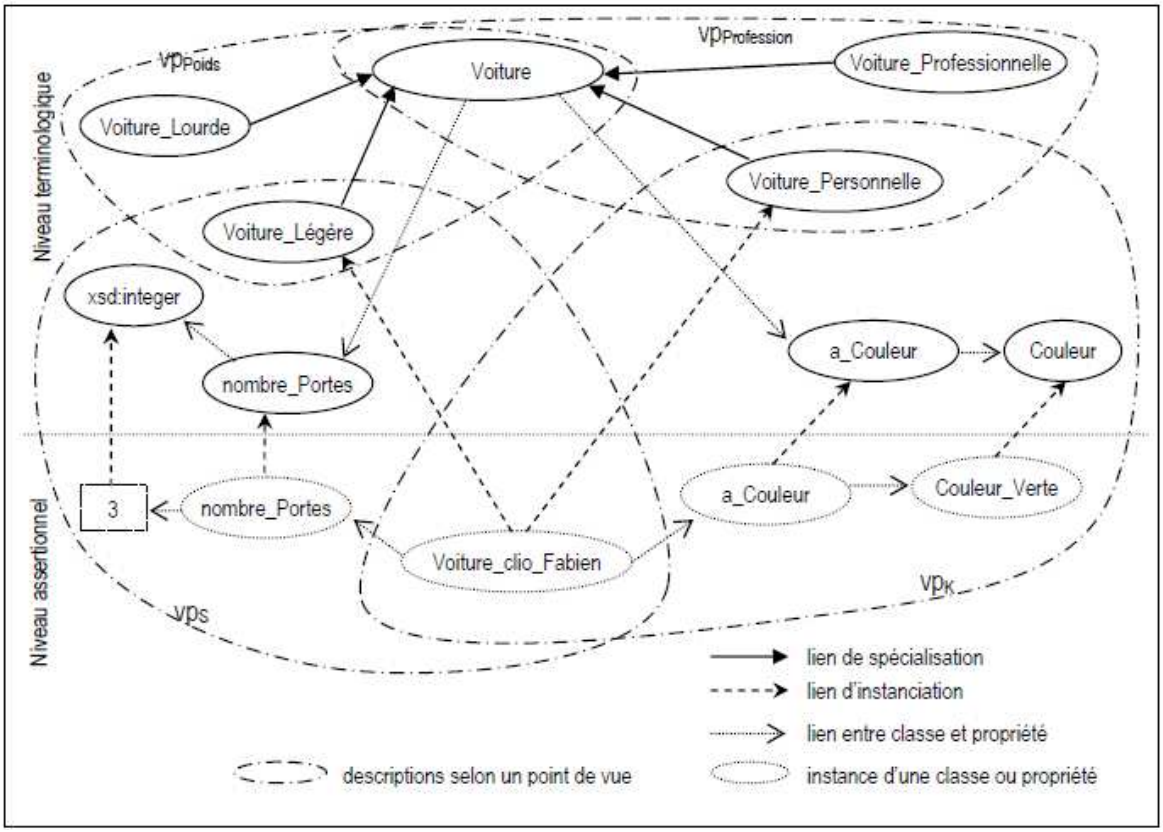


Figure 2.9- Représentation multi-points de vue selon le modèle proposé par [Bach, 2006]

Pour conclure, nous pouvons dire que l'objectif principal du modèle multi-points de vue (MVP) [Bach, 2006] et du langage C-OWL [Bouquet *et al.*, 2004] est de préserver l'indépendance de chaque ontologie locale. Pour travailler sur des ontologies multiples, les auteurs proposent la notion de pont sémantique pour établir des liens entre des ontologies disjointes. Dans ce cas, le modèle MVP et le langage C-OWL sont conçus pour faire face aux besoins d'alignement d'ontologies. Ceci est sensé permettre le raisonnement sur les appariement dans des ontologies disjointes. Notre approche est différente. Elle est motivée par le besoin de concevoir, dans certaines applications, une seule et unique ontologie représentant un même univers de discours et qui tient compte des différents points de vue des utilisateurs. Ainsi, dans notre approche le point de vue a un double rôle :

- il permet de voir le concept selon un certain angle et de ce fait, seuls les attributs du concept pertinents pour le point de vue sont visibles,
- il donne une structuration de la connaissance du concept sur lequel il est appliqué dans une hiérarchie significative pour le point de vue. Toutes les hiérarchies définissant un point de vue particulier d'un même concept ont le même concept racine et peuvent être liées par des passerelles.

3.4.1.3 MuRO : MultiRepresentation Ontology

Notre approche va dans la direction des travaux proposés par [Benslimane et al., 2006], dans lesquels le formalisme de la logique de descriptions est utilisé afin d'intégrer différents points de vue (ou *contextes*) dans une même ontologie. Cette dernière, appelée ontologie de multi-représentation (*MuRO :* MultiRepresentation Ontology), caractérise le concept ontologique (*concept contextuel*) par un ensemble de définitions qui varient selon le contexte. Autrement dit, un concept est défini avec plusieurs représentations de telle sorte qu'une représentation est garantie pour chaque contexte. Pour représenter une telle ontologie, *Benslimane et al.,* (2006) se sont inspirés du mécanisme d'*estampillage* utilisé dans le domaine des bases de données géographiques pour supporter la multi représentation d'une même donnée [Balley et al., 2004].

Balley et al., (2004) proposent une extension du modèle conceptuel entité-association nommé MADS. Un mécanisme d'estampillage est proposé dans ce cadre afin de permettre aux concepteurs d'associer à chaque entité, association ou propriété du domaine spatial, plusieurs descriptions selon la résolution sémantique ou spatiale.

Benslimane et al., (2006) ont adapté le mécanisme d'estampillage à une logique de description. C'est ainsi qu'ils ont proposé une extension des logiques de descriptions de type *ALCNR*, en appliquant l'estampillage sur les constructeurs. Ceci, permet aux attributs d'avoir des définitions différentes, i.e. différentes cardinalités ou différents domaines de valeurs (selon des contextes différents). Cette proposition peut être illustrée à travers un exemple simple. Soit à considérer deux représentations, d'un monde réel, identifiées par les estampilles S1 et S2. La description d'un concept estampillé **Homme_Marié,** dans deux cultures différentes (S1 et S2), est comme suit:

Type Homme_Marié (s1, s2) s1, s2: Nom: string s1: Epouse (1,1): Femme s2: Epouse (1, 4): Femme

Dans le contexte *s*1 un homme marié est un homme ayant exactement une seule épouse. Par ailleurs, dans le contexte *s2* un homme marié peut avoir au maximum 4 épouses. Dans les deux contextes (s*1*et s*2*) le concept **Homme_Marié** est décrit par l'attribut Nom. Ceci ce traduit en logique de description estampillée par l'expression logique suivante:

Homme_Marié ≡
$(\exists nom.String)[s_1, s_2] \sqcap \exists Epouse.Femme[s_1, s_2] \sqcap ((\leq 1Epouse)[s_1] \sqcup (\leq 4Epouse)[s_2])$

De notre point de vue, l'inconvénient majeur de cette approche est que les représentations variées d'un concept selon des contextes différents sont intégrées dans une même expression logique. Ainsi, un problème de positionnement du concept dans la hiérarchie de subsomption remet en cause les définitions multiples rattachées à ce concept. Par ailleurs, gérer l'évolution de ce type d'ontologies par l'ajout, la modification ou la suppression de contextes peut être relativement difficile.

Dans l'approche que nous proposons, il est plus facile de développer ce type d'ontologies et de suivre leur évolution, en utilisant le point de vue comme un moyen de modélisation et de représentation décentralisé. Cette décentralisation consiste à répartir les différentes représentations dans des unités, qu'on appelle des points de vue. Ceci facilite l'appréhension, par un utilisateur, d'unités plus petites et précisément adaptées à sa tâche tout en permettant à des mécanismes de raisonnement de fonctionner indifféremment sur ces unités ou sur les assemblages de ces unités.

4 Conclusion

L'idée de diviser une représentation en modules représentant chacun un point de vue peut apporter beaucoup. Elle permet de construire une représentation relativement complexe à partir de briques de base indépendantes les unes des autres. L'un des avantages du point de vue est de présenter une version simplifiée de la connaissance. Cela permet de diminuer la charge pour l'utilisateur comme pour un programme.

Dans le cadre de notre travail, nous adoptons le terme *d'ontologie multi-points de vue* afin de mettre l'accent sur l'importance de la notion de point de vue pour **1)** résoudre le problème de la représentation multiple **2)** avoir un meilleur accès et une meilleure visibilité des éléments ontologiques (concepts, rôles, individus) **3)** tirer profit de la représentation multi-points de vue des connaissances pour permettre leur évolution. Par ailleurs, pour prendre en compte la notion de point de vue, nous supposons que les différents points de vue sur un même univers de discours sont des visions partielles mais complémentaires. Leur union est une représentation complète et cohérente du monde.

Dans cette thèse, nous nous intéressons au problème de développement des ontologies multi-points de vue dans le cadre du Web sémantique. Le chapitre suivant s'intéresse pour cette raison à la présentation des principes et des technologies du Web sémantique.

Chapitre 3

Outils et technologies du Web sémantique

1 Introduction

A sa création par Tim Berners-Lee[1] au début des années 90, l'objectif du *World Wide Web* (ou *Web* ou *WWW*) était de permettre des échanges de savoirs entre individus distants. C'est dans ce but que fût créé le langage HTML - *HyperText Markup Language* - [Raggett *et al.*, 1999]. Il permet d'une part une mise en forme aisée et rapide de documents à l'aide de *balises* (*tags* en anglais). D'autre part, HTML offre une mise à disposition en réseau de ces documents, dits *hypertextes*, car connectés entre eux via des URIs.

Le Web a fortement évolué en l'espace de quinze ans. Victime de son succès, l'essort considérable qu'a connu le Web a abouti en ce début du XXI^{e} siècle à une prolifération de documents hétérogènes difficiles à consulter. Le langage HTML apparaît comme non adapté au Web d'aujourd'hui. En effet, ce langage fournit un jeu de balises destinées quasi-exclusivement à la mise-en-page de documents hypertextes. Il ne permet donc pas une organisation sémantique de l'information contenue dans les documents HTML.

Au cours de cette dernière décennie, une nouvelle vision du Web prend corps : celle d'un *Web Sémantique*. Pour la naissance du Web sémantique, le W3C4 - *World Wide Web Consortium* - se dote de nouveaux langages et outils, comme XML, RDF(S) puis OWL. Tous ont un objectif commun, celui de participer à une (re-)formalisation des savoirs selon une nouvelle architecture, et permettre ainsi une meilleure organisation de l'information pour un partage et une diffusion plus aisés.

Ce chapitre décrit la vision du Web sémantique (i.e. les composants principaux, l'architecture, les langages de représentation, etc.). A la fin, la dernière section de ce chapitre aborde l'utilisation des technologies du Web sémantique en vue de matérialiser une mémoire d'entreprise en un Web sémantique d'entreprise (WSE).

[1] http://www.w3.org/People/Berners-Lee/

2 Le Web à la recherche d'une sémantique

Depuis sa création, le Web a connu un succès gigantesque et est en train de devenir peu à peu le premier outil pour la production, la publication, la diffusion et le partage de l'information. Cependant la répartition à travers le monde d'un tel réseau d'informations, la croissance accrue du nombre de publications et la liberté totale d'y accéder ont révélé plusieurs limites et inconvénients. En effet, le web actuel ne dispose pas d'outils pour décrire et structurer ses ressources de manière satisfaisante afin de permettre un accès pertinent à l'information. Par exemple, les liens entre les pages web, bien que porteurs de sens pour les utilisateurs, n'ont aucune signification exploitable par les machines.

C'est pour pallier ces insuffisances que Sir Tim Berners Lee a proposé dans [Berners-Lee et al., 2001] d'étendre le web actuel vers un web où l'information possédera un sens bien défini permettant ainsi aux applications d'exploiter directement la sémantique des ressources et de coopérer avec l'utilisateur afin de lui faciliter ses tâches (recherche, commerce électronique…).

Ce futur web baptisé Web sémantique a été défini comme un « Web intelligent » où les informations stockées dans les machines seraient en plus comprises par ces dernières afin de répondre efficacement aux requêtes lancées par les utilisateurs.

2.1 Les principales composantes du Web sémantique

Le Web Sémantique a été proposé en se basant sur les critiques adressées au web actuel : (i) certes HTML a permis de tisser tout un réseau d'informations par ses liens hypertextes, mais il n'a donné aucune sémantique à ces liens ce qui les rend pratiquement inexploitables par les machines, (ii) les métadonnées utilisées sont non structurées et limitées dans leurs usages, et (iii) il est difficile de faire des inférences et des raisonnements sur les connaissances décrites dans les documents publiés sur le web vu l'absence de modèles permettant la représentation sémantique de ces connaissances.

Tous ces problèmes ont fait l'objet de différents travaux de recherche qui ont convergé vers plusieurs solutions parmi lesquelles nous présentons celles qui nous semblent les plus essentielles :

- Proposer des langages et des formalismes de représentation et de structuration des connaissances. Ces langages permettent de modéliser et de représenter le contenu sémantique des ressources du web.

- Rendre disponibles des ressources conceptuelles (des modèles) représentées dans ces langages modélisant les connaissances et facilitant leur accès et leur partage : les ontologies.
- Proposer des métadonnées explicites, c'est-à-dire qui suivent un modèle et qui sont exprimées dans des langages définis formellement.

2.2 Architecture du Web sémantique

Le fonctionnement du Web Sémantique est fondé sur le fait que les machines puissent accéder à l'ensemble des informations éparpillées sur le web. Le W3C (World Wide Web Consortium) ainsi que les chercheurs travaillant dans le domaine de l'intelligence artificielle ont beaucoup travaillé sur ce point et ont proposé plusieurs langages de représentation des connaissances afin de faciliter cet accès. La figure 3.1 présente une des visions de ces langages, organisés en couches d'expressivité croissante. Cet ensemble de « couches » est appelé, par le W3C, la *pyramide des langages* du Web sémantique.

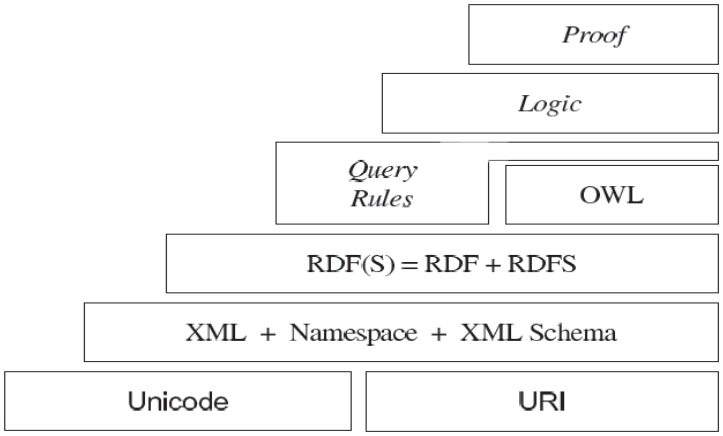


Figure 3.1- La pyramide des langages du Web sémantique, par le W3C

Deux types de bénéfices sont motivés par cette organisation en couches des langages du Web sémantique. En effet, chaque niveau repose sur les résultats définis au niveau inférieur, c'est-à-dire que chaque niveau est progressivement plus spécialisé et plus complexe que son niveau précédent. D'autre part, tout niveau est indépendant des niveaux supérieurs afin qu'il puisse être développé et rendu opérationnel de manière autonome par rapport aux développement des niveaux supérieurs.

il est à noter que les mots en italique, dans cette pyramide, ne sont pas des langages à proprement parler, mais ils caractérisent soit des langages en phase d'élaboration soit d'hypothétiques couches.

Une caractéristique commune à ces langages est d'être exprimables selon une syntaxe XML, et d'avoir une capacité à identifier des ressources (sur le Web notamment) en utilisant le système d'adressage par *Universal Resource Indicator* (URI). Il s'agit d'un mécanisme de base du Web où tous les hyperliens sont exprimés sous forme d'URI. Une URI est une chaîne de caractères qui permet d'identifier de manière unique une entité élémentaire abstraite ou physique, comme une page sur le Web.

Une ressource, dans le milieu du Web sémantique, est une entité élémentaire servant à représenter des connaissances. Elle peut donc appartenir au niveau structurel ou au niveau factuel d'une modélisation. L'apparition du langage XML a permis de décrire les ressources portant sur un domaine donné au moyen d'un vocabulaire adapté à ce domaine. En effet, contrairement au HTML qui propose un ensemble limité de balises pour décrire des ressources, l'utilisateur à la possibilité de définir ses propres balises. L'arrivée du langage RDF donne un cadre à la façon de décrire les ressources sur le Web, en proposant l'agencement de ces ressources au sein de *triplets*. C'est avec l'utilisation conjointe des langages RDF et RDFS que les ressources du Web peuvent être organisées sur deux niveaux conceptuels pour représenter les connaissances d'un domaine ; le langage RDF pour représenter les connaissances factuelles d'un domaine et le langage RDFS pour les connaissances structurelles. Ces connaissances structurelles se composent d'un ensemble de *classes* et de *propriétés* (des relations) organisées en *taxonomies*. Pour offrir plus de «richesse» en matière de modélisation, le W3C préconise le langage OWL qui permet de créer des *ontologies* des domaines modélisés. Une *ontologie* offre une granularité plus fine pour décrire et définir les connaissances structurelles d'un domaine par rapport à une taxonomie. Une ontologie permet d'organiser classes et propriétés non seulement en hiérarchies mais aussi selon des liens dits horizontaux, comme la disjonction, l'équivalence, *etc.*

3 Les langages XML, RDF et RDFS

3.1 eXtensible Markup Language

Recommandation du W3C depuis le 10 février 1998, le langage eXtensible Markup Language (XML) connait depuis ses débuts un succès indéniable. Défini dés son origine comme un métalangage facilitant l'élaboration de langages à balises spécialisés, XML a conquis depuis de multiples formats, les nouveaux comme les anciens. De xhtml au format de documents openoffice.org, nombreux sont les documents qui profitent aujourd'hui du cadre de XML.

Les normes strictes qui gèrent la syntaxe et la structure de XML rendent le langage et son utilisation plus aisés, favorisant notamment la factorisation des travaux de développement d'analyseurs syntaxiques. XML fournit donc un cadre de structuration des données qui peut être employé pour la définition rapide et sans ambiguïté syntaxique d'un format de document.

3.1.1 Structure d'un fichier XML

Une source de données est un document XML si elle est « bien formée », c'est à dire si elle correspond parfaitement à la spécification de XML. Un document XML dispose alors de deux structures : une structure logique et une structure physique, les deux correspondant parfaitement.

Un document XML est représenté physiquement sous la forme d'un fichier texte structuré en éléments, à l'aide de balises éventuellement imbriquées. En en-tête du document doit figurer un « prologue », une déclaration qui identifie le document comme un document XML. Ce prologue indique la version de XML employée, le codage de caractères, et si le document est associé à une DTD[2] ou s'il est autonome. Il existe un élément particulier : l'élément « racine », encore appelé « élément document ». Cette racine doit contenir tous les autres éléments du document et ne peut apparaître qu'une fois dans un document XML. En conséquence, aucun autre élément ne contient la racine.

En complément des principes fondamentaux de XML expliqués ci-dessus, on peut ajouter quelques règles syntaxiques du corps d'un document XML :

- chaque élément doit commencer par une balise ouvrante et se termine par une balise fermante. Eventuellement, un élément vide peut être représenté par une balise simple.
- l'imbrication des éléments du document se fait sans chevauchement. Concrètement, cela signifie que si la balise ouvrante d'un élément se trouve à l'intérieur d'un élément parent, la balise fermante se trouvera dans le même élément.
- la valeur d'un attribut doit être encadrée de guillemets, simples ou doubles.

La figure 3.2(a) présente un fichier XML qui décrit les membres d'une équipe. Ce fichier a pour en-tête les deux premières lignes qui indiquent que c'est un document XML suivant la version 1.1 de XML, et qu'il respecte la DTD "equipe.dtd". Les informations contenues dans le corps de ce fichier sont structurées à l'intérieur de balises XML. L'élément racine de ce document, celui qui contient les autres élément, est défini ici entre une balise ouvrante <équipe> et une fermante </équipe>. Cette équipe est définie d'une part par un attribut nom

[2] Une DTD - *Document Type Definition* - permet la définition simple de la syntaxe et de la grammaire du nouveau langage basé sur XML

ayant pour valeur "SI&BC" et d'autre part par ses éléments enfants. Ces éléments enfants définissent des membres (balises <membre>). Chaque membre est caractérisé par les éléments enfants nom, prénom et âge. Ces trois éléments sont caractérisés par leurs données imbriquées respectives. Ainsi, l'équipe "SI&BC" contient deux membres. Le second par exemple a pour nom Hemam, pour prénom Mounir et pour âge 35.

Ce fichier XML bien formé, car respectant la syntaxe XML, est de plus valide car il respecte les contraintes imposées par le DTD "equipe.dtd". En effet, cette DTD, présentée en figure 3.2(b), indique qu'un élément équipe est composé d'un ou plusieurs (*cf.* le signe +) éléments membre. Il peut de plus posséder un attribut nom qui est facultatif (*cf.* le motclé#IMPLIED). Un élément membre est composé d'un élément nom, d'un élément prénom, et éventuellement (*cf.* le signe ?) d'un élément âge.

```
1 <?xml version="1.1" encoding="utf-8" ?>
2 <!DOCTYPE équipe SYSTEM "equipe.dtd">
3       <équipe nom = "SI&BC">
4               <membre>
5                       <nom>Gharzouli</nom>
6                       <prénom>Mouhamed</prénom>
7               </membre>
8               <membre>
9                       <nom>Hemam</nom>
10                      <prénom>Mounir</prénom>
11                      <âge>35</âge>
12              </membre>
13    </équipe>
```

(a) Un fichier XML *valide* : *bien formé* et vérifiant la DTD "equipe.dtd".

```
1 <!ELEMENT équipe (membre+)>
2 <!ATTLIST équipe nom CDATA #IMPLIED>
3
4 <!ELEMENT membre (nom, prénom, âge?)>
5 <!ELEMENT nom (#PCDATA)>
6 <!ELEMENT prénom (#PCDATA)>
7 <!ELEMENT âge (#PCDATA)>
```

(b) La DTD "equipe.dtd".

Figure 3.2- Un fichier XML et sa DTD.

De manière plus formelle, XML est un méta-langage[3] facilitant l'élaboration de langages à balises spécialisées. XML fournit un cadre de définition et de structuration des *notions* qui constituent un format de document. Cependant, XML n'impose aucune contrainte sémantique

[3] Un *méta-langage* se dit d'un langage utilisé pour définir un autre langage.

à la signification de ces documents, l'interopérabilité syntaxique n'est pas suffisante pour qu'un logiciel puisse "comprendre" le contenu des données et les manipuler d'une manière significative.

La couche RDF + RDFS est considérée comme étant la première fondation de l'interopérabilité sémantique. En effet, Elle permet de décrire les taxonomies des concepts et des propriétés (avec leurs signatures). RDF fournit un moyen d'insérer de la sémantique dans un document, l'information est conservée principalement sous forme de déclarations RDF. Le RDFS décrit les hiérarchies des concepts et des relations entre les concepts, les propriétés et les restrictions domaine/co-domaine pour les propriétés. Dans ce qui suit nous allons présenter plus en détail les deux langages RDF et RDFS .

3.2 RDF : Resource Description Framework

Le W3C a publié la première recommandation de RDF[4] en 1999 pour répondre aux besoins du Web Sémantique. Sa dernière version en date est celle de février 2004[5]. L'objectif premier de RDF est la description de « ressources » [Baget *et al.,* 2004]. Un document RDF peut contenir plusieurs descriptions. Une **description** correspond à un ensemble d'**énoncés** (ou «statements») au sujet d'une **ressource**. Un énoncé RDF est aussi appelé **triplet** car il est composé de trois éléments, sujet-prédicat- objet, où :

1) *le sujet* représente **la ressource décrite**, i.e. tout document accessible sur le Web comme les pages HTML, les documents textuels (PDF, Ms Word) ou multimédias (images, vidéo), etc., mais aussi tout objet, abstrait ou non, du monde réel. Les ressources sont nommées en utilisant une URI.

2) *le prédicat* représente **la propriété descriptive**, i.e. une caractéristique spécifique, un attribut ou une relation, utilisée pour décrire une ressource.

3) *l'objet* représente **la valeur de cette propriété**, soit une valeur littérale, comme un nombre entier ou une chaîne de caractère, soit une autre ressource accessible par son URI. Par contre, une valeur littérale ne peut en aucun cas être le sujet d'un énoncé.

Un triplet peut s'écrire « prédicat(sujet, objet) » ou encore « propriété(sujet, valeur) ». Par exemple, la phrase « Francis Coppola est né à Détroit » sera traduite par le triplet « né(Francis Coppola, Détroit) ». Ici, l'objet « Détroit » correspond soit à une simple chaîne de caractères, soit une ressource identifiée par le site Web de la ville. D'autres notations peuvent être utilisées pour afficher des données RDF. En particulier, les graphes étiquetés orientés sont

[4] http://www.w3.org/RDF/
[5] http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/

parfaitement adaptés à la représentation des énoncés RDF puisque le modèle RDF a été influencé par les réseaux sémantiques [Baget *et al.*, 2004]. Dans ces graphes, un nœud représente un sujet ou un objet et l'arc un prédicat dont l'origine est le sujet et la destination l'objet de l'énoncé (*cf.* Figure 3.3). RDF peut aussi être exprimé en XML afin notamment d'échanger ses données avec les agents logiciels du Web ou autres.

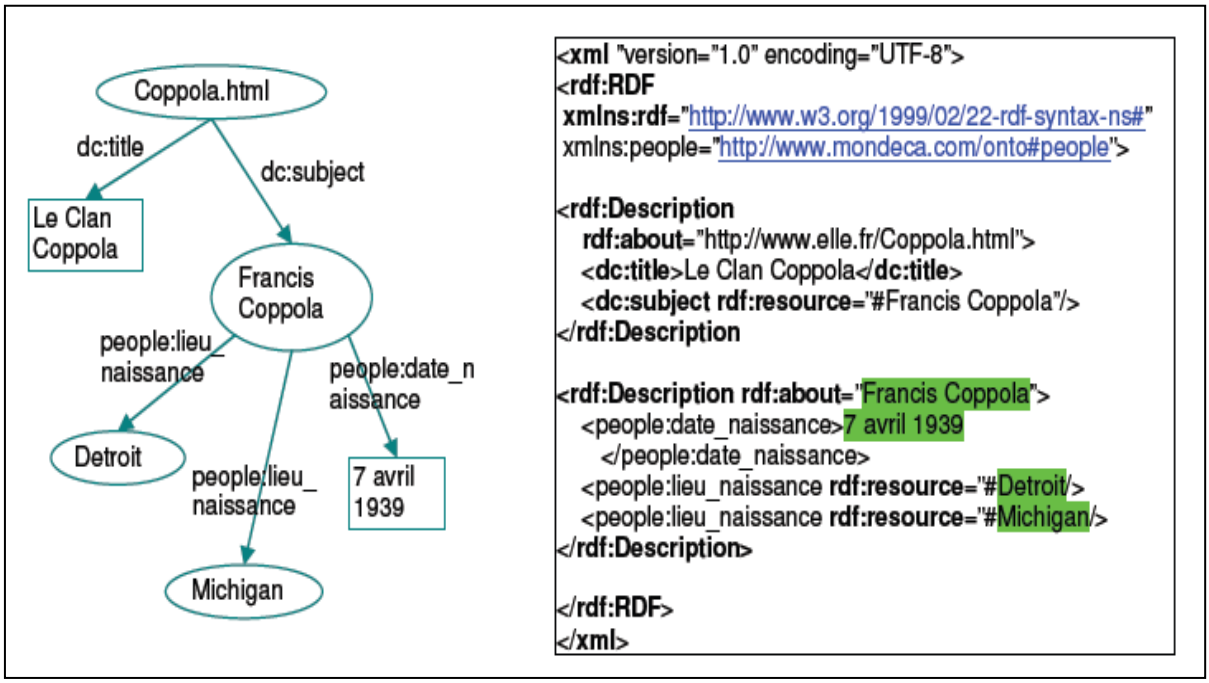


Figure 3.3 - Exemple de représentation en RDF (notation graphique à gauche et XML à droite)

A propos du langage, un document RDF est délimité par l'élément `rdf:RDF` qui comporte un ou plusieurs éléments `rdf:Description` pour chacune des descriptions de ressources comprises dans le document. Chaque description comprend un attribut `rdf:about` qui pointe vers l'URI de la ressource à décrire et un à plusieurs éléments représentant chacun un prédicat. Lorsqu'un prédicat a pour valeur une autre ressource, l'attribut rdf:resource pointera vers son URI, comme c'est le cas pour le prédicat « dc:subject » dans la figure ci-dessous. Par contre, si c'est un littéral, la valeur est insérée dans l'élément, cf. « dc:title » dans la Figure 3.

Les autres caractéristiques de RDF permettent la composition des énoncés en structures plus complexes comme le groupement de sujets et/ou d'objets en listes énumérées ou bien la réification d'énoncés, i.e. la création de nouveaux énoncés à partir d'énoncés existants.

RDF répond aux besoins de la plupart des outils d'annotation. En effet, les documents RDF sont des documents XML valides, leur modélisation sous forme de réseau sémantique apporte

une flexibilité nécessaire et il est possible de réutiliser des énoncés existants pour composer des documents RDF plus complexes. Par contre, RDF ne fournit pas de mécanisme de contrainte de classes ou de types pour les différentes parties du triplet. Il n'est donc pas assez puissant pour représenter de vraies ontologies avec un système de raisonnement approprié.

3.3 RDFS : Resource Description Framework Schema

Comme son nom l'indique, RDFS[6] est un langage utilisé pour la définition de schémas RDF. Son modèle de données est basé sur celui des Frames. Alors qu'RDF exprime les relations sémantiques au niveau des instances sous la forme de triplets, RDFS exprime donc les relations au niveau des classes et des propriétés (les prédicats RDF), contraignant ainsi les instances possibles dans les triplets RDF [Baget *et al.*, 2004]. Par conséquent, RDFS marque une étape de plus vers la conception d'un formalisme de représentation plus riche et introduit les primitives de bases de la modélisation ontologique pour le Web Sémantique.

RDFS est construit au dessus de RDF, c'est-à-dire que ce langage réutilise la syntaxe des triplets RDF et l'étend avec de nouveaux éléments pour définir les classes, `rdfs:class`, et les propriétés, `rdfs:property`. L'élément `rdfs:subClassOf` permet de spécifier la taxonomie des classes. La figure 3.4 montre les éléments incluses dans le RDF(S).

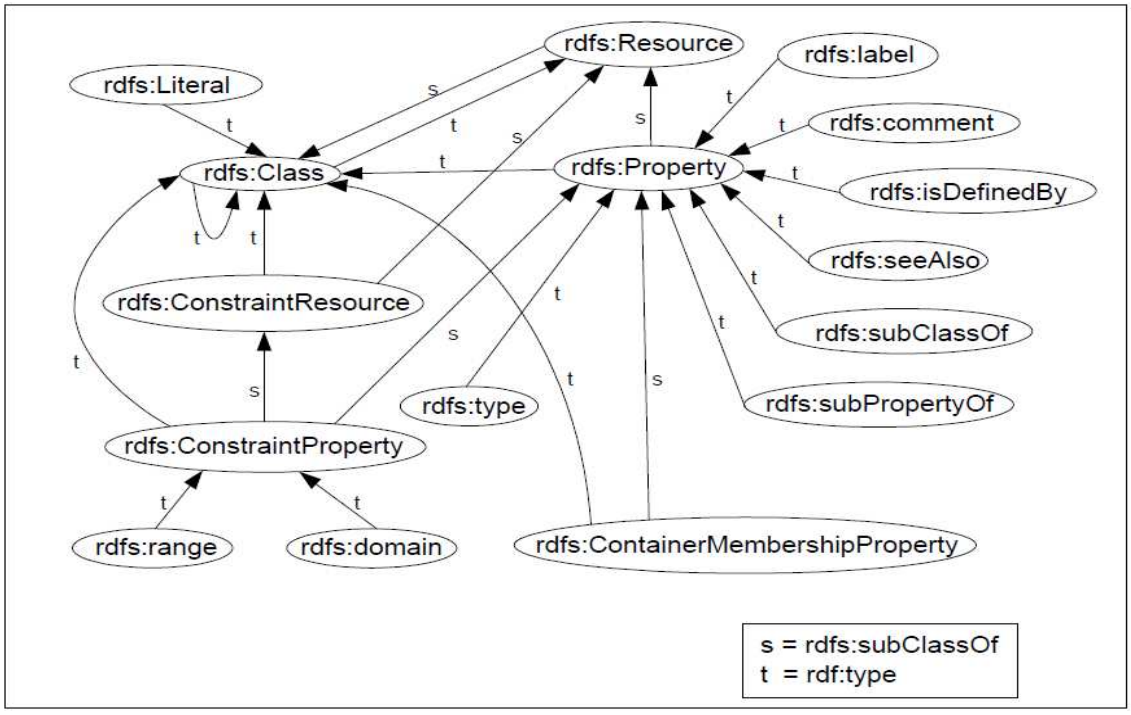


Figure 3.4 - Le schéma de RDF(S).

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

Dans la Figure 3.5, «Personne» et «Personnalité» sont définies comme étant des classes de l'ontologie, « Personnalité » étant une sous-classe de «Personne». Enfin la ressource «Francis Coppola» est déclarée comme étant une instance de la classe «Personnalité». Les hiérarchies de propriétés sont définies par l'élément `rdfs:subPropertyOf`. Les définitions de propriétés sont restreintes par deux contraintes:

1) le domaine de la propriété, représenté par `rdfs:domain` et indiquant la classe à laquelle cette propriété s'applique et,

2) la portée de la propriété, représentée par `rdfs:range` et indiquant la classe dont les instances seront les valeurs autorisées pour cette propriété.

Ainsi, dans l'exemple de la Figure 3.5, nous définissons la propriété « Lieu_Naissance » avec la classe « Personne » comme domaine et « Lieu » comme sa portée. Ceci signifie donc que cette propriété s'applique à toute instance de la classe « Personne » et que sa valeur doit être une instance de la classe « Lieu ».

```
<xml "version="1.0" encoding="UTF-8" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

<rdf:Description rdf:about="http://www.elle.fr/Coppola.html">
   <rdf:type rdf:about="#Article"/>
   <rdfs:label>Le Clan Coppola</rdfs:label>
   <dc:subject rdf:resourceRef="#FFCoppola"/>
</rdf:Description>

<rdfs:Class rdf:ID="#Lieu">
<rdfs:Class rdf:ID="#Personne">
<rdfs:Class rdf:ID="#Personnalité">
   <rdfs:subClassOf rdf:resource ="#Personne"/>
</rdfs:Class>

<rdf:Property rdf:about="lieu_naissance">
   <rdfs:domain rdf:resource="#Personne"/>
   <rdfs:range rdf:resource="#Lieu"/>
</rdf:Property>
<rdf:Property rdf:about="date_naissance">
   <rdfs:domain rdf:resource="#Personne"/>
   <rdfs:range rdf:resource="http://www.w3.org/2000/10/XMLSchema#string"/>
</rdf:Property>

<Personnalité rdf:about="FFCoppola">
   <rdfs:label>Francis Coppola</rdfs:label>
   <lieu_naissance rdf:resource="#Detroit"/>
   <lieu_naissance rdf:resource="#Michigan"/>
   <date_naissance>7 avril 1939</date_naissance>
</Personnalité>
</xml>
```

Figure 3.5 - Exemple de représentation sémantique basée sur un schéma RDFS

Comme RDF et RDFS sont des recommandations du W3C, ils ont été largement acceptés pour l'annotation des ressources du Web. Les Schémas RDF apportent des caractéristiques importantes à l'annotation sémantique en RDF par la définition d'une ontologie relativement simple à travers une taxonomie de classes, de leurs propriétés ainsi que la restriction du domaine et de la portée des propriétés. Ces aspects sont particulièrement importants pour l'annotation sémantique car ils offrent assez d'expressivité pour mettre en place un premier niveau sémantique. Par contre, il n'est pas suffisant de définir un ensemble de classes et de propriétés RDFS pour constituer une véritable ontologie formelle. Parmi les caractéristiques manquantes, citons la restriction des cardinalités par des quantifieurs, la combinaison de classes par des opérateurs ensemblistes, l'équivalence entre classes, etc.

4 OWL : Web Ontology Language

Afin d'étendre le pouvoir d'expression de RDF-Schema, deux initiatives ont été lancées. La première est une initiative américaine lancée par la DARPA (Defense Advanced Research Projects Agency) qui a permis la spécification du langage DAML-ONT [Stein et al., 2000]. La seconde est une initiative sponsorisée par la communauté européenne qui a permis la spécification du langage OIL [Fensel et al., 2001]. La fusion de ces deux langages a donné naissance à DAML+OIL [Connolly et al., 2001] qui a servi de base pour la conception du langage OWL (*Web Ontology Language* ou *Langage d'Ontologie Web*) [Dean *et al.*, 2004].

Le langage OWL fournit des mécanismes pour créer tous les composants d'une ontologie : classes, instances, propriétés et axiomes. OWL repose également sur la syntaxe des triplets RDF et réutilise certaines des constructions RDFS. Comme en RDFS, les classes peuvent avoir des sous-classes, fournissant ainsi un mécanisme pour le raisonnement et l'héritage des propriétés. Par contre, en OWL, on distingue :

1) les propriétés objet (*object property*), i.e. les relations, qui relient des instances de classes à d'autres instances de classes. C'est l'équivalent des triplets RDF dont l'objet est une ressource.

2) les propriétés type de données (*datatype property*), i.e. les attributs, qui relient des instances de classes à des valeurs de types de données (nombres, chaînes de caractères,…). C'est l'équivalent des triplets RDF dont l'objet est une valeur littérale.

Les axiomes fournissent de l'information au sujet des classes et des propriétés, spécifiant par exemple l'équivalence entre deux classes.

4.1 Les trois sous langages de OWL

Le langage OWL est assez complexe, voila pourquoi il se compose de trois sous langages qui proposent une expressivité croissante, chacun conçu pour des communautés de développeurs et des utilisateurs spécifiques : OWL Lite, OWL DL, OWL Full. Chacun est une extension par rapport a son prédécesseur plus simple:

- **OWL Lite** est le sous langage de OWL le plus simple. Il répond à des besoins de hiérarchie de classification et de fonctionnalités de contrainte simples de cardinalité´e 0 ou 1. Une cardinalité 0 ou 1 correspond à des relations fonctionnelles, par exemple, une personne a une adresse. Toutefois, cette personne peut avoir un ou plusieurs prénoms, OWL Lite ne suffit donc pas pour cette situation.
- **OWL DL** est plus complexe que OWL Lite, permettant une expressivité bien plus importante. OWL DL est fondé sur la logique de descriptions (d'où son nom, OWL Description Logics) et conférant donc à OWL DL son adaptation au raisonnement automatisé. Malgré sa complexité relative face à OWL Lite, OWL-DL garantit la complétude des raisonnements (toutes les inférences sont calculables) et leur décidabilité (leur calcul se fait en une durée finie).
- **OWL Full** est la version la plus complexe d'OWL, mais également celle qui permet le plus haut niveau d'expressivité. Le langage OWL Full se destine aux personnes souhaitant une expressivité maximale, ainsi que la liberté syntaxique de RDF, mais sans garantir la complétude et la décidabilité des calculs liés à l'ontologie. En OWL Full, une classe peut également être un individu, il n’y a pas de séparation des types. C’est pour cela que la calculabilité ne peut être garantie.

Il est à noter que entre ces trois sous langage il existe une dépendance de nature hiérarchique. En effet, toute ontologie OWL Lite valide est également une ontologie OWL DL valide, et toute ontologie OWL DL valide est également une ontologie OWL Full valide.

4.2 Eléments du langage OWL

Cette partie ne va pas reprendre toutes les finesses de OWL Lite, OWL DL et OWL Full, mais uniquement les plus importantes. Pour des explications exhaustives du rôle de chacun des éléments composant le vocabulaire d'OWL, il est recommandé de se reporter à la Recommandation du W3C « OWL Web Ontology Language Reference » [Patel-Schneider *et al.*, 2004].

4.2.1 L’élément classe

Une classe définit un groupe d'individus qui sont réunis parce qu'ils ont des caractéristiques similaires. L'ensemble des individus d'une classe est désigné par le terme « extension de classe », chacun de ces individus étant alors une « instance » de la classe. Les trois versions d'OWL comportent les mêmes mécanismes de classe, à ceci près que OWL FULL est la seule version à permettre qu'une classe soit l'instance d'une autre classe (d'une métaclasse). A l'inverse, OWL Lite et OWL DL n'autorisent pas qu'une instance de classe soit elle-même une classe.

Déclaration de classe

Dans OWL, la déclaration d'une classe se fait par le biais du mécanisme de « description de classe », qui se présente sous diverses formes. Une classe peut être définie comme suit :

- **l'indicateur de classe**. La description de la classe se fait, dans ce cas, directement par le nommage de cette classe. Par exemple, une classe « Doctorant » se déclare de la manière suivante :

```
<owl:Class rdf:ID="Doctorant" />
```

Il est à noter que ce type de description de classe est le seul qui permette de nommer une classe. Dans les autres cas, la description représente une classe dite « anonyme », crée en plaçant des contraintes sur son extension.

- **l'énumération des individus composant la classe**. Dans ce cas la définition d'une se fait en énumérant les instances de la classe, à l'aide de la propriété `owl:oneOf`. Par exemple :

```
<owl:Class>
   <owl:oneOf rdf:parseType="Collection">
      <owl:Thing rdf:about="#Djamel" />
      <owl:Thing rdf:about="#Mouhamed" />
      <owl:Thing rdf:about="#Mounir" />
      <owl:Thing rdf:about="#Leila" />
   </owl:oneOf>
</owl:Class>
```

Si ce mécanisme est utilisable avec OWL DL et OWL FULL, il ne fait cependant pas partie de OWL Lite.

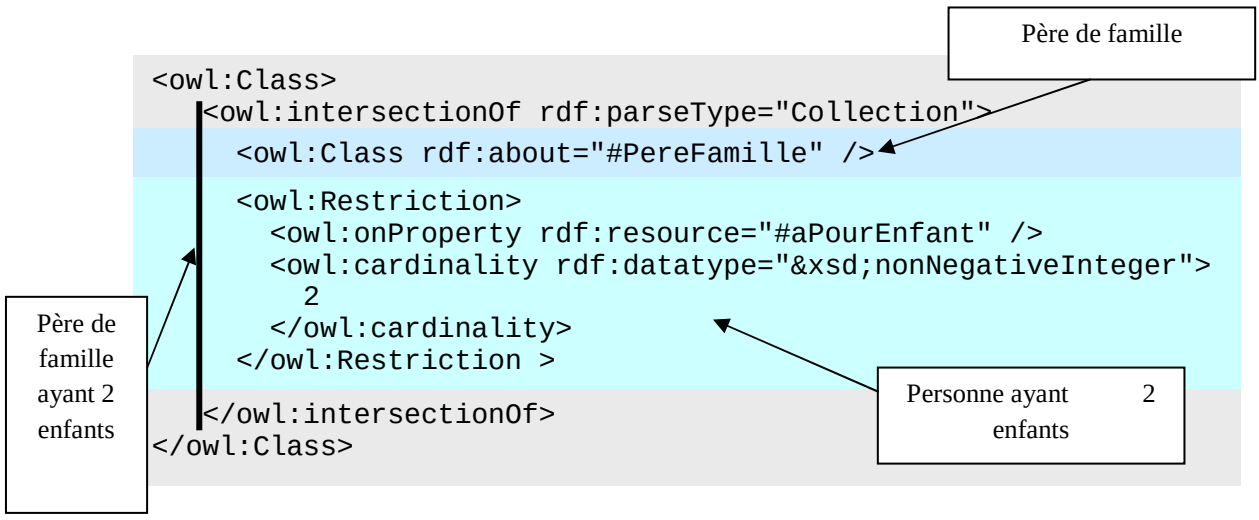
- **les operateurs de restrictions sur les valeurs de propriétés.** Ce type de définition consiste à utiliser une restriction OWL qui définit une classe en spécifiant une contrainte associée à une propriété. Cette classe est l'ensemble des instances satisfaisant cette contrainte. Celle-ci peut être de trois sortes :

– une contrainte de cardinalité ; elle définit le nombre minimum, exact ou maximum de valeurs que doivent définir les instances pour la propriété contrainte ;

– une contrainte de codomaine ; *allValuesFrom* (resp. *someValuesFrom*) permet de créer une classe dont les instances ne peuvent prendre pour valeur de la propriété contrainte que des (resp. au moins une) instances d'une classe spécifiée ;

– une contrainte de valeur ; *hasValue* permet de créer une classe dont les instances ont une valeur spécifiée pour la propriété contrainte.

- **les operateurs orientés ensemble**. Ce type de définition est d'appliquer des opérations booléennes à une ou plusieurs classes déjà définies (*intersectionOf, unionOf, complementOf*) pour formuler une nouvelle classe:



4.2.2 L'élément propriété

OWL fait la distinction entre deux types de propriétés :

- les propriétés d'objet permettent de relier des instances à d'autres instances;
- les propriétés de type de donnée permettent de relier des individus à des valeurs de données.

Une propriété d'objet est une instance de la classe owl:ObjectProperty, une propriété de type de donnée étant une instance de la classe owl:DatatypeProperty. Ces deux classes sont elles-mêmes sous-classes de la classe RDF rdf:Property.

```
<owl:ObjectProperty rdf:ID="habite">
        <rdfs:domain rdf:resource="#Humain" />
        <rdfs:range rdf:resource="#Pays" />
</owl:ObjectProperty>
```

Dans l'exemple ci-dessus, on apprend que la propriété habite a pour domaine la classe Humain et pour co-domaine la classe Pays : elle relie des instances de la classe Humain à des instances de la classe Pays. Dans le cas d'une propriété de type de donnée, le co-domaine de la propriété peut être un type de donnée, comme défini dans le Schéma XML. Par exemple, on peut définir la propriété de type de données anneeDeNaissance :

```
<owl:Class rdf:ID="dateDeNaissance" />
    <owl:DatatypeProperty rdf:ID="anneeDeNaissance">
    <rdfs:domain rdf:resource="#dateDeNaissance" />
    <rdfs:range rdf:resource="&xsd;positiveInteger"/>
</owl:DatatypeProperty>
```

Dans ce cas, anneDeNaissance fait correspondre aux instances de la classe de dateDeNaissance des entiers positifs.

4.2.3 La relation de sous-classe

Il existe dans toute ontologie OWL une superclasse, nommée *Thing*, dont toutes les autres classes sont des sous-classes. Ceci nous amène directement au concept d'héritage, disponible à l'aide de l'élément *rdfs:subClassOf* :

```
<owl:Class rdf:ID="Humain">
  <rdfs:subClassOf rdf:resource="&etre;#EtreVivant" />
</owl:Class>
<owl:Class rdf:ID="Homme">
  <rdfs:subClassOf rdf:resource="#Humain" />
</owl:Class>
```

Enfin, il existe aussi une classe nommée *noThing*, qui est sous-classe de toutes les classes OWL. Cette classe ne peut avoir aucune instance.

4.2.4 Les instances de classe

La définition d'un individu consiste à énoncer un « fait », encore appelé « axiome d'individu ». On peut distinguer deux types de faits [Lacot, 2005] :

- les faits concernant l'appartenance à une classe

La plupart des faits concerne généralement la déclaration de l'appartenance à une classe d'un individu et les valeurs de propriété de cet individu. Un fait s'exprime de la manière suivante :

```
<Humain rdf:ID="Leila">
        <aPourPere rdf:resource="#Ali" />
        <aPourFrere rdf:resource="#Karim" />
</Humain>
```

Le fait écrit dans cet exemple exprime l'existence d'un Humain nommé « Leila » dont le père s'appelle « Ali », et qu'il a un frère nommé « Karim ». On peut également instancier un individu anonyme en omettant son identifiant :

```
<Humain >
        <aPourPere rdf:resource="#Ali" />
        <aPourFrere rdf:resource="#Karim" />
</Humain>
```

Ce fait décrit, dans ce cas, l'existence d'un Humain dont le père se nomme «Ali » et qui a un frère nommé «Karim ».

- les faits concernant l'identité des individus

Une difficulté qui peut éventuellement apparaître dans le nommage des individus concerne la non-unicité éventuelle des noms attribués aux individus. Par exemple, un même individu pourrait être désigné de plusieurs façons différentes. C'est la raison pour laquelle OWL propose un mécanisme permettant de lever cette ambiguïté, à l'aide des propriétés `owl:sameAs`, `owl:diffrentFrom` et `owl:allDifferent`. L'exemple suivant permet de déclarer que les noms « Louis_XIV » et « Le_Roi_Soleil » désignent la même personne :

```
<rdf:Description rdf:about="#Louis_XIV">
        <owl:sameAs rdf:resource="#Le_Roi_Soleil" />
</rdf:Description>
```

4.3 OWL et les logiques de descriptions

De part leur nature, les recherches sur les ontologies intègrent plusieurs domaines de l'informatique tels que l'ingénierie des connaissances, la représentation des connaissances, la recherche d'informations ou les bases de données [Staab et Studer, 2004]. Nous nous

intéressons plus spécifiquement ici au langage de représentation d'ontologies OWL dont les origines proviennent de plusieurs langages de représentation des connaissances [Horrocks et al., 2003]. En tant que langage de représentation d'ontologies pour le Web sémantique, OWL doit être doté de certaines caractéristiques, en particulier :

– Il doit être associé à une sémantique standard et formellement définie, permettant la mise en œuvre de mécanismes de raisonnements bien maîtrisés.

– Il doit être très expressif, pour prendre en compte la variété des utilisateurs, des domaines et des applications présents sur le Web et le Web sémantique.

Pour ces raisons, une des influences les plus importantes ayant conduit à OWL provient des logiques de descriptions.

Une base de connaissances en logique de descriptions contient la description des concepts, des rôles et des individus du domaine d'application. En OWL, on dira plutôt qu'une *ontologie* contient la description de *classes*, de *propriétés* et d'*individus*, respectivement. Pour simplifier, on adoptera dans la suite le vocabulaire associé à OWL. Les *individus* sont utilisés pour représenter les objets du domaine. Les *propriétés* correspondent à des relations binaires entre ces objets. Les *classes* représentent des ensembles d'objets, possédant des caractéristiques communes.

Ainsi, OWL étant une quasi-réécriture de certaines logiques de descriptions. La correspondance entre les constructeurs, axiomes et assertions des LD et la syntaxe du langage OWL-DL est donnée dans les tables 3.1 et 3.2.

Table 3.1– Relations entre les constructeurs de OWL-DL et les LDs [Napoli *et al.*, 2004]

Constructeur	Syntaxe LD	Exemple
intersectionOf	$C_1 \sqcap C_2$	$\texttt{Personne} \sqcap \texttt{Masculin}$
unionOf	$C_1 \sqcup C_2$	$\texttt{Docteur} \sqcup \texttt{Professeur}$
complementOf	$\neg C$	$\neg\texttt{Homme}$
oneOf	$\{x_1, x_2, \ldots, x_n\}$	$\{\texttt{Paolo}, \texttt{Maria}\}$
allValuesFrom	$\forall r.C$	$\forall \texttt{aEnfant.Homme}$
someValuesFrom	$\exists r.C$	$\exists \texttt{aEnfant.Femme}$
hasValue	$\exists r.\{x\}$	$\exists \texttt{citoyenDe}.\{\texttt{Europe}\}$
minCardinality	$\geq n\, r$	$(\geq 2\ \texttt{aEnfant})$
maxCardinality	$\leq n\, r$	$(\leq 1\ \texttt{aEnfant})$
inverseOf	r^-	$\texttt{aEnfant}^-$

Table 3.2– Relations entre les axiomes de OWL-DL et les LDs [Napoli *et al.*, 2004]

Axiome	Syntaxe LD	Exemple
subClassOf	$C_1 \sqsubseteq C_2$	$\texttt{Homme} \sqsubseteq \texttt{Personne}$
equivalentClass	$C_1 \equiv C_2$	$\texttt{Homme} \equiv \texttt{Masculin} \sqcap \texttt{Personne}$
subPropertyOf	$r_1 \sqsubseteq r_2$	$\texttt{aFils} \sqsubseteq \texttt{aEnfant}$
equivalentProperty	$r_1 \equiv r_2$	$\texttt{cout} \equiv \texttt{prix}$
disjointWith	$C_1 \sqsubseteq \neg C_2$	$\texttt{Masculin} \sqsubseteq \neg\texttt{Feminin}$
sameAs	$\{x_1\} \equiv \{x_2\}$	$\{\texttt{Paris}\} \equiv \{\texttt{Parigi}\}$
differentFrom	$\{x_1\} \sqsubseteq \neg\{x_2\}$	$\{\texttt{Paolo}\} \sqsubseteq \neg\{\texttt{Maria}\}$
transitiveProperty	$r \in R_+$	$\texttt{aAncetre}^+ \in R_+$
functionalProperty	$\top \sqsubseteq (\leq 1\, r)$	$\top \sqsubseteq (\leq 1\, \texttt{aMaman})$
inverseFunctionalProperty	$\top \sqsubseteq (\leq 1\, r^-)$	$\top \sqsubseteq (\leq 1\, \texttt{estMamanDe}^-)$
symmetricProperty	$r \equiv r^-$	$\texttt{estVoisinDe} \equiv \texttt{estVoisinDe}^-$

4.4 Sémantique logique de OWL

À un document OWL est associée une formule logique dans le langage $\mathcal{SHIF}(\mathcal{D})$ pour OWL Lite et dans le langage $\mathcal{SHOIN}(\mathcal{D})$ pour OWL DL [Horrocks et Patel-Schneider, 2004]. Une classe en OWL correspond à un concept en logique de description, une propriété concept en OWL correspond à un rôle en DL. Les datatypes et propriétés dadatypes sont nouvellement introduits ; une propriété datatype lie une classe à un datatype. L'interprétation I_{OWL}, qui associe une sémantique à un document OWL, étend l'interprétation I définie dans le chapitre 1 en Section 5.3.2, de façon à prendre en compte les datatypes. L'interprétation $I_{OWL} = (\Delta^I, \Delta^D, .^I)$ consiste en un ensemble non vide Δ^I, appelé cette fois domaine de l'interprétation des instances, en un ensemble Δ^D disjoint de Δ^I, appelé domaine de l'interprétation des datatypes, et une fonction d'interprétation[7] $.^I$ qui fait correspondre : à une classe *un sous-ensemble de* Δ^I, *à une propriété concept un sous-ensemble de* $\Delta^I \times \Delta^I$, *à un individu un élément de* Δ^I, *à un datatype un sous-ensemble* Δ^D, *à une propriété datatype un sous-ensemble de* $\Delta^I \times \Delta^D$, *et à une valeur un élément de* Δ^D, de telle sorte que les expressions en colonne « Sémantique » du Tableau 3.3 soient satisfaites.

[7] Certains auteurs dissocient cette fonction d'interprétation en deux sous-fonctions : $.^I$ pour interpréter les concepts, les propriétés concepts et les individus, et $.^D$ pour interpréter les datatypes, les propriétés datatypes et les valeurs.

Table 3.3 – Les datatypes et leurs valeurs en OWL [Raimbault, 2008]

Constructeur	Syntaxe	Sémantique
hiérarchie de datatypes	$D_1 \sqsubseteq D_2$	$(D_1 \sqsubseteq D_2)^{\mathcal{I}} = \{d \in \Delta_D \mid d \in D_1^{\mathcal{I}} \rightarrow d \in D_2^{\mathcal{I}}\}$
quantificateur universel	$\forall u.D$	$(\forall u.D)^{\mathcal{I}} = \{x \in \Delta_{\mathcal{I}} \mid \forall d \in \Delta_D, (x,d) \in u^{\mathcal{I}} \rightarrow d \in D^{\mathcal{I}}\}$
quantificateur existentiel	$\exists u.D$	$(\exists u.D)^{\mathcal{I}} = \{x \in \Delta_{\mathcal{I}} \mid \exists d \in \Delta_D, (x,d) \in u^{\mathcal{I}} \wedge d \in D^{\mathcal{I}}\}$
restrictions de nombre	$\geq n\ u$	$(\geq n\ u)^{\mathcal{I}} = \{x \in \Delta_{\mathcal{I}} \mid \text{card}(\{d \in \Delta_D \mid (x,d) \in u^{\mathcal{I}}\}) \geq n\}$
(au-plus et au-moins)	$\leq n\ u$	$(\leq n\ u)^{\mathcal{I}} = \{x \in \Delta_{\mathcal{I}} \mid \text{card}(\{d \in \Delta_D \mid (x,d) \in u^{\mathcal{I}}\}) \leq n\}$
type énuméré	$\{v_1, ..., v_n\}$	$\{v_1, ..., v_n\}^{\mathcal{I}} = \{v_1^{\mathcal{I}}, ..., v_n^{\mathcal{I}}\}$
Axiome	**Syntaxe**	**Sémantique**
définition de valeur	$D(v)$	$D(v)^{\mathcal{I}} = v^{\mathcal{I}} \in D^{\mathcal{I}}$
affectation de valeur	$u(a,v)$	$u(a,v)^{\mathcal{I}} = (a^{\mathcal{I}}, v^{\mathcal{I}}) \in u^{\mathcal{I}}$

D, D1 et *D2* sont des datatypes, *u* et u_i des propriétés datatypes, a un individu, et *v et* v_j des valeurs.

4.5 PROTÉGÉ : Un éditeur d'ontologies OWL

Les outils d'édition constituent une aide pour l'implémentation d'une ontologie dans laquelle les principaux choix ont déjà été faits. Les outils peuvent se subdiviser en deux groupes : les outils dont le formalisme est proche des frames et ceux dont le formalisme est proche des logiques de description. Dans la première catégorie, le plus connu est ONTOEdit[8] tandis que l'outil PROTÉGÉ peut être classé dans les deux catégories.

PROTÉGÉ[9] est distribué en open source par le Stanford Medical Informatics de l'université de médecine de Stanford depuis 1995. Il est construit autour d'un modèle de connaissances inspiré par le paradigme des frames : classes, slots (attributs) et facets (contraintes sur les attributs) sont les primitives de modélisation proposées. Ce modèle autorise une liberté de conception importante, puisque le contenu des formulaires de spécification des classes peut être modifié suivant les besoins, via un système de méta-classes, qui constituent des sortes de « patrons » pour les classes du modèle du domaine. Il est adapté à la construction d'ontologies depuis la version PROTÉGÉ2000. L'interface très complète ainsi que l'architecture logicielle bien pensée permettant l'insertion de plugins, notamment

[8] ONTOEdit n'est pas disponible gratuitement dans sa version complète

[9] Auparavant appelé PROTÉGÉ 2000, cet éditeur a repris le nom de l'outil d'acquisition des connaissances qui l'a précédé. PROTÉGÉ est disponible à l'adresse suivante : http://protege.stanford.edu/.

des plugins pour gérer les représentations sous forme graphique, par exemple OWLViz[10], ont grandement contribué au succès de PROTÉGÉ .

En quelques années, cet éditeur s'est imposé comme la référence, avec une communauté d'utilisateurs extrêmement importante et active. Ses nombreuses extensions lui permettent en particulier de gérer des langages standards comme RDF et surtout OWL, de créer des axiomes formels de manière intuitive, d'accéder aux ontologies par des interfaces graphiques évoluées, de comparer et fusionner des ontologies avec la suite PROMPT [Noy & Musen, 2003] . Il est également possible de faire fonctionner des raisonneurs, comme RACER (Renamed ABox and Concept Expression Reasoner) [Haarslev et Müller, 2001] pour le langage OWL par exemple, pour vérifier la cohérence et la consistance de la structure ontologique.

5 De la mémoire d'entreprise au Web sémantique d'entreprise

A l'heure actuelle, les sources d'informations et de connaissances dans les entreprises deviennent de plus en plus importantes et elles jouent aussi un rôle crucial pour le développement de l'entreprise. Le partage du travail nécessite un partage de connaissances au sein des entreprises et ce besoin peut devenir encore plus crucial dans l'avenir. Pour réussir, les entreprises doivent bien gérer ces sources de connaissances et supporter l'utilisation effective des connaissances.

Les auteurs dans [Dieng-Kuntz et al., 2005] abordent une approche particulière de la gestion des connaissances qui est la constitution d'une mémoire d'entreprise ou mémoire organisationnelle, matérialisant et indexant les connaissances et informations cruciales de l'organisation (i.e. une entreprise, une institution ou une communauté de pratique) afin d'améliorer leur accès, partage et réutilisation voire de permettre la création de nouvelles connaissances par les membres de l'organisation dans leurs tâches individuelles et collectives. La construction d'une telle mémoire d'entreprise entraîne de plus en plus l'utilisation effective des connaissances partagées au sein de l'entreprise. Avec l'évolution du Web vers le Web sémantique, les nouvelles technologies pour ce dernier peuvent aussi être appliquées pour l'organisation. Les ressources de cette mémoire d'entreprise sont similaires à celles du Web, elles ont besoin d'être annotées sémantiquement pour mieux découvrir et utiliser les connaissances de ces ressources.

[10] OWLViz est téléchargeable à l'adresse suivante :
http://www.co-ode.org/downloads/binaries/OWLViz_Build_17.zip.

En faisant l'analogie entre les ressources du web et les ressources d'une entreprise, [Dieng-Kuntz, 2005] propose de matérialiser la mémoire d'entreprise à travers un Web Sémantique d'Entreprise (WSE) (ou Web Sémantique d'Organisation (WSO)) en utilisant les ontologies qui fournissent un cadre formel pour décrire les différentes sources de connaissances de l'organisation et qui guident la création d'annotations sémantiques facilitant la description, le partage et l'accès à ces sources. Les principales composantes de ce Web sémantique d'entreprise sont les suivantes :

- Les *ressources* : il peut s'agir de bases de données, de documents (dans toutes sortes de formats), de services/logiciels, voire de personnes, etc.
- Les *ontologies* : elles décrivent le vocabulaire partagé par les différentes communautés de l'entreprise.
- Les *annotations sémantiques* : elles décrivent des méta-données sur les ressources en se basant sur les concepts et les relations de l'ontologie.

La Figure 3.6 présente l'architecture d'un WSE dans lequel les technologies du Web sémantique sont utilisées pour réaliser les tâches du Web sémantique d'entreprise. Dans cette architecture, nous avons des ontologies communes décrivant d'une manière formelle des connaissances du domaine de l'entreprise. Les annotateurs annotent sémantiquement des ressources d'entreprise telles que la description du contenu des documents, les compétences des personnes, les caractéristiques des services… en employant le vocabulaire commun et partagé au sein de l'entreprise (i.e. l'ontologie). Avec l'aide d'un système de gestion des connaissances, les utilisateurs dans l'entreprise peuvent alors utiliser des annotations créées grâce aux significations des concepts et des propriétés prédéfinies dans l'ontologie.

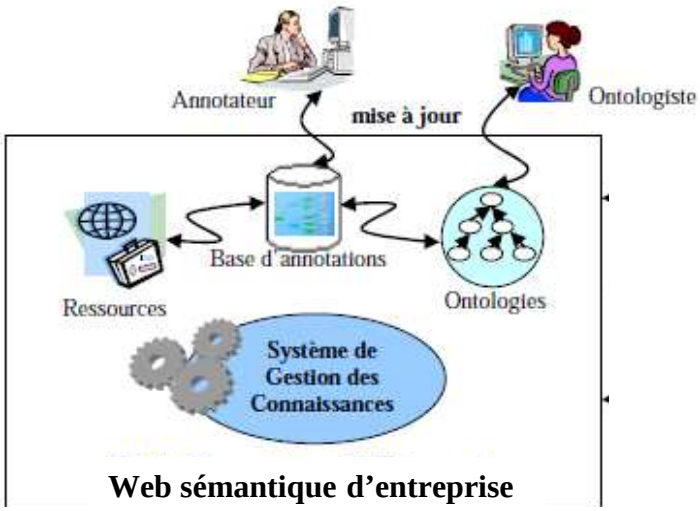


Figure 3.6 - Architecture d'un Web sémantique d'entreprise [Dieng-Kuntz et al., 2005]

6 Conclusion

Nous avons fait dans ce chapitre une revue non exhaustive des notions liées au Web Sémantique. Nous avons décrit quelques langages de représentation des ontologies utilisés dans le cadre du Web sémantique. Nous avons vu que RDF(S) ne supportait pas certaines notions à propos des ontologies. OWL permet d'étendre RDF(S) en fournissant des primitives pour exprimer les notions qu'il n'était pas possible d'exprimer avec RDF(S). Ce langage dispose d'une sémantique formelle et fournit un support pour les inférences grâce à son lien avec les logiques de descriptions.

Du point de vue des organisations, quel intérêt apporte le Web sémantique? Cette question rejoint les travaux réalisés dans le domaine de la gestion des connaissances et en particulier dans le domaine des mémoires d'entreprise ou d'organisation. Ces travaux intègrent les techniques du Web sémantique (ontologies, annotations sémantiques et langages formels de représentation, etc.) pour construire le Web sémantique d'entreprise visant à faciliter son accès, son partage et sa réutilisation des connaissances organisationnelles. Cependant, les organisations vivent généralement dans un environnement dynamique et hétérogène. De ce fait, lors de la formalisation et de l'exploitation des connaissances ontologiques au sein d'une organisation, il est souvent intéressant de prendre en compte la vision de chacun des utilisateurs. En effet, différentes personnes alimentent l'ontologie ou l'exploitent. Lors de l'alimentation, chacune de ces personnes peut vouloir formaliser ses connaissances selon un point de vue ou une perception particulière. De même, lors de l'exploitation de l'ontologie, il est préférable de fournir à chaque utilisateur un moyen permettant de s'approprier et de personnaliser l'exploitation des connaissances mémorisées.

Dans le chapitre suivant, nous présenterons notre approche pour le développement d'une ontologie dans une organisation hétérogène en prenant en compte différents points de vue et terminologies des utilisateurs au sein de cette organisation.

Chapitre 4

Vp-MethOnto : Une méthode pour le développement des ontologies multi-points de vue

« Une petite colline te fait arriver à une grande »

Proverbe africain

1 Introduction

L'une des caractéristiques essentielles des ontologies est qu'elles fournissent des connaissances consensuelles sur un domaine donné. De nos jours, les ontologies englobent un ensemble si riche de la connaissance que leur compréhension (complète) dépasse celle de n'importe quel développeur ou concepteur seul ou même d'une petite équipe de concepteurs. De ce fait, le développement d'une ontologie à grande échelle doit être le fruit d'un effort commun et collaboratif de plusieurs experts de domaine voir même les futures utilisateurs. Ainsi, l'approche multi-points de vue pour le développement d'ontologie est la mieux adaptée pour cette tâche par rapport autres approches mono-point de vue.

Ce chapitre présente notre contribution au problème posé par cette thèse, à savoir le développement d'ontologies multi-points de vue. Ceci, nécessite une méthode et une démarche qui se situent des les premières phases de développement d'une ontologie, permettant à des experts du domaine de modéliser les différences de points de vue sur le sens des termes. Notre approche fait intervenir les experts du domaine dans la conception de l'ontologie multi-points de vue en leur permettant d'aller vers le consensus.

Dans la suite, nous adoptons le terme *d'ontologie multi-points de vue* afin de mettre l'accent sur l'importance de la notion de point de vue pour **(a)** résoudre le problème de la représentation multiple **(b)** avoir un meilleur accès et une meilleure visibilité des éléments ontologiques (concepts, rôles, individus) **(c)** tirer profit de la représentation multi-points de vue des connaissances pour permettre leur évolution. Par ailleurs, pour prendre en compte la notion de point de vue, nous supposons que les différents points de vue sur un même univers de discours sont des visions partielles mais complémentaires. Leur union est une représentation complète et cohérente du monde.

2 Ontologie et point de vue

Une Ontologie représente et structure la connaissance du monde réel. On a vu auparavant que cette connaissance ressort de domaines différents, des familles différentes d'objets. Si la connaissance du monde est variée, il en est de même des observateurs qui veulent la manipuler : chacun a ses intérêts particuliers ; chacun regarde des propriétés et relations particulières des objets du monde, et chacun organise les objets selon sa propre perspective. Dans notre approche, cette diversité est reflétée dans l'ontologie par les points de vue : un point de vue décrit une perspective particulière d'observation du monde représenté. Les points de vue suivent l'hypothèse d'un monde unique vu de façons complémentaires par les divers observateurs.

2.1 Ontologie multi-points de vue

Dans ce travail, nous nous intéressons au problème de développement d'une ontologie dans une organisation hétérogène en prenant en compte différents points de vue et terminologies des communautés au sein de cette organisation. Une telle ontologie, que nous appelons ontologie Multi-Points de Vue (MPV), confère à un même univers de discours plusieurs descriptions partielles telles que chacune soit relative à un point de vue. De plus, les différentes descriptions partielles partagent à un niveau global des éléments ontologiques consensuels et des passerelles. Ces dernières établissent les communications entre les points de vue et représentent ainsi la collaboration interdisciplinaire (*Cf.* **Figure 4.1**).

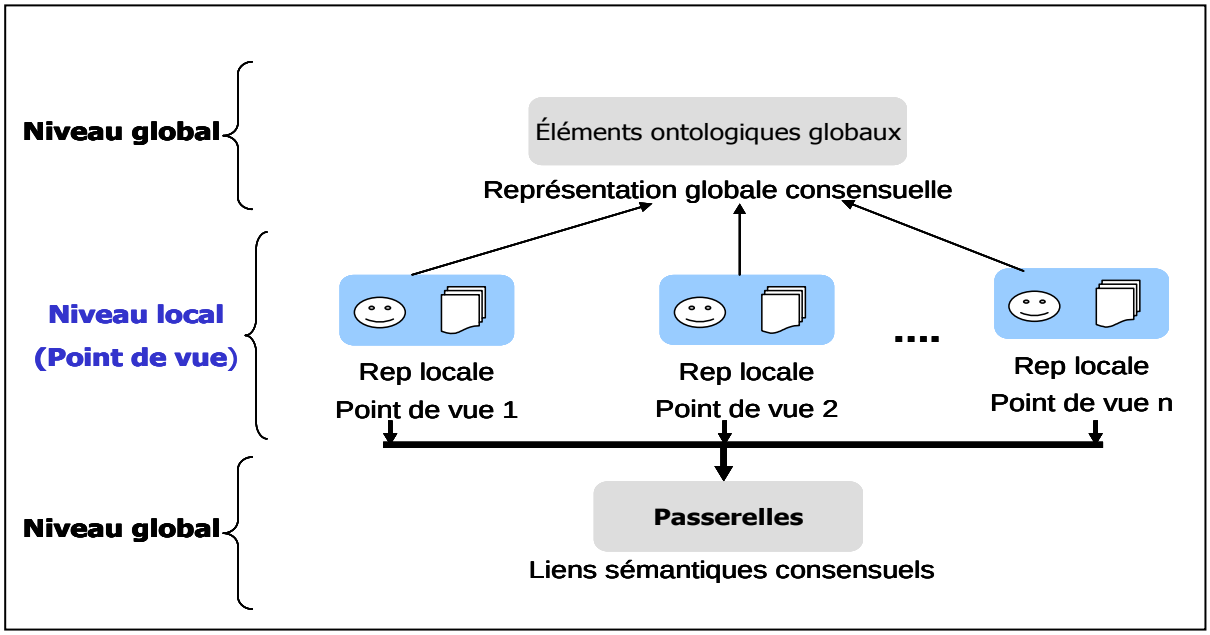


Figure 4.1- Ontologie multi-points de vue

3 Description de la méthode proposée

Notre objectif est de proposer une méthode (*cf.* **Figure 4.2**) permettant la construction d'ontologies, en prenant en compte la notion de point de vue. Pour définir l'ontologie multi-points de vue, nous nous sommes inspirés de travaux qui traitent de la représentation des connaissances par objets auxquels sont associés des points de vue [Mariño, 1993]. Pour cela, nous utilisons un sous-langage de la Logique de Descriptions (LD$)\ \mathcal{SHOQ}\ (\mathcal{D})$ [Baader et al., 2003a] pour exprimer les notions inhérentes aux points de vue telles que les concepts et les rôles globaux et locaux, les passerelles, les estampillages, …

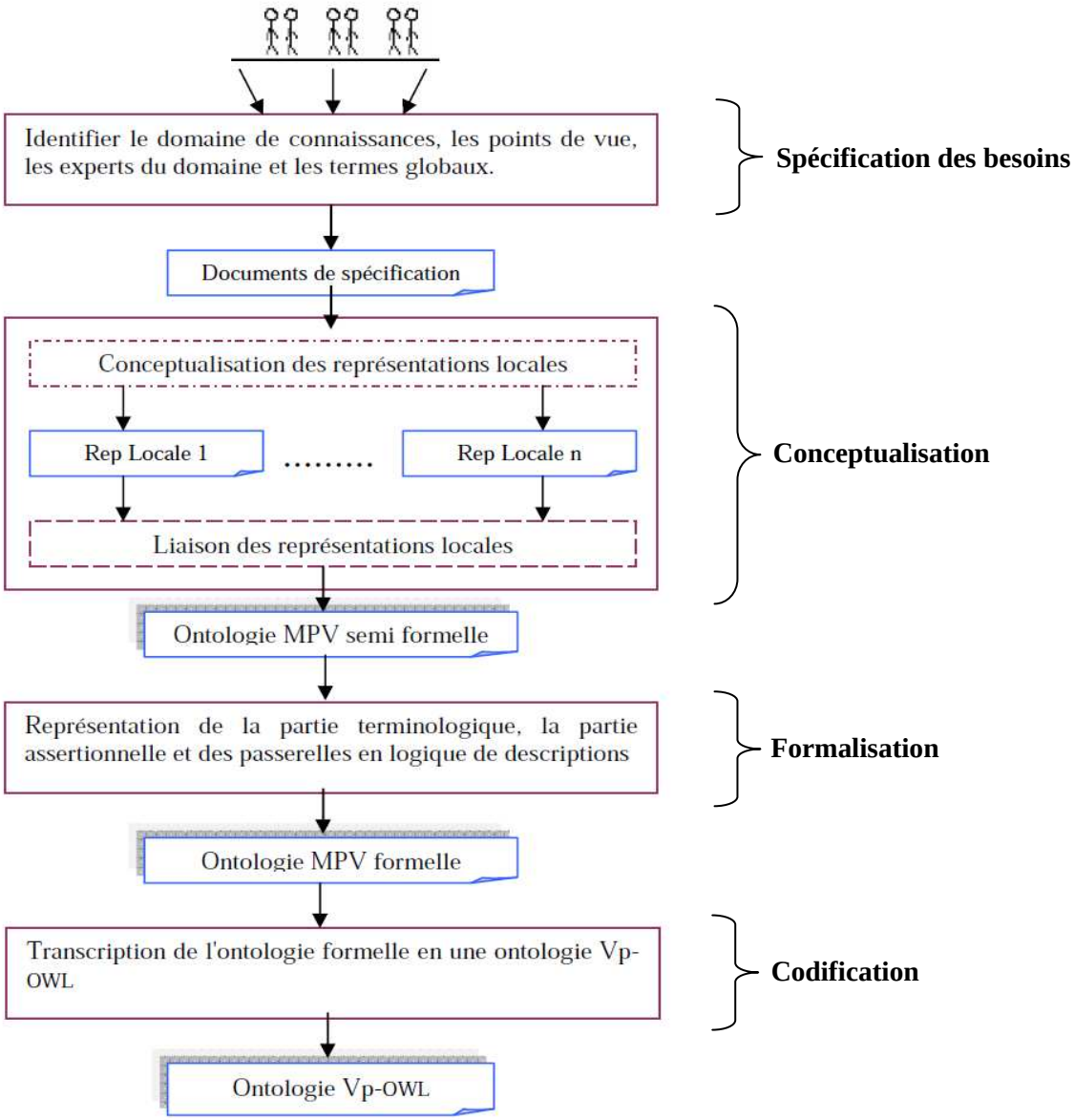


Figure 4.2- Les étapes de la méthode Vp-MethOnto

La méthode Vp-MethOnto que nous proposons est complète, dans la mesure où, partant de données brutes elle permet d'arriver à une ontologie multi-points de vue représentée par le langage Vp-OWL. Pour ce faire, quatre principales étapes sont suivies afin d'expliciter et de guider la construction de l'ontologie multi-points de vue. Nous allons à présent détailler chacune de ces quatre étapes constituant le processus de construction d'ontologies multi-points de vue, en mettant l'accent sur les objectifs de chacune de ces étapes ainsi que les actions qui doivent être mener. Pour illustrer notre approche, une étude de cas est présentée pour le domaine d'immobilier.

3.1 Étape de spécification des besoins

3.1.1 Présentation

Le but visé par cette étape est d'établir un document de spécification des besoins. Ce dernier permet de décrire l'ontologie à construire à travers les quatre aspects (i.e. propriétés) suivants :

1. Le domaine de connaissance

Cet aspect consiste à délimiter aussi précisément que possible le domaine que va couvrir l'ontologie.

2. Les points de vue

Il s'agit ici de déterminer quels sont les points de vue à représenter. En effet, lorsqu'un domaine est suffisamment vaste et complexe, il est souvent organisé selon plusieurs services, plusieurs tâches, plusieurs groupes de travail ou encore plusieurs communautés. Cette organisation apporte une division *a priori* du domaine en points de vue.

Exemple : Dans le domaine « Immobilier » on peut distinguer les points de vue : « *Finance* », « *Taille* » et « *Localisation* ».

3. Les experts du domaine

Cet aspect consiste à déterminer parmi les experts du domaine, ceux qui sont le mieux à même de modéliser les connaissances relatives à chacun des points de vue, selon leurs spécialités.

4. La portée de l'ontologie

Cet aspect consiste à déterminer à priori la liste des termes globaux les plus importants désignant les entités du domaine de connaissances à représenter.

Exemple : Dans l'exemple du domaine « Immobilier » on peut déterminer les termes globaux suivants: {Habitat, Appartement, Locataire, Agence}.

3.1.2 Résultat

Le résultat de cette étape est un document de spécification des besoins représenté dans le langage RDF. Le langage RDF permet de décrire les ressources qui sont disponibles sur le Web en terme de trois types d'objets : Une **ressource** (correspond a l'ontologie MPV à construire) définie par un ensemble de **propriétés** (correspondent aux quatre aspects) et l'association d'une ressource à une propriété par une **valeur** de propriété (correspond au résultat de chaque aspect) (*cf.* **Table 4.1**).

Table 4.1- Les trois parties d'une déclaration RDF

Sujet (ressource)	*Une ontologieMPV, identifié par une URI*
Prédicat (propriété)	*Un aspect*
Objet (valeur littérale ou bien une ressource)	*Le résultat d'un aspect*

On peut résumer cette étape par les points suivants :

Données en entrée	Besoins
Données en sorties	Spécification des besoins en RDF
Intervenants	Ontologiste

3.1.3 Description

Pour représenter le résultat de cette étape (qui est un document de spécification des besoins), nous utilisons tout d'abord le modèle élémentaire de *RDF (*également appelé diagramme arcs et nœuds), pour décrire de manière graphique le résultat de chaque aspect. Dans ce diagramme, les nœuds (représentés par des ovales) représentent les ressources, tandis que les arcs symbolisent les propriétés (i.e. aspects). Les valeurs littérales sont quant à elles représentées par des rectangles. Ensuite nous utilisons le langage RDF qui propose une syntaxe basé sur le XML, afin d'obtenir un document au format RDF.

3.1.3.1 Représentation par le modèle de données de RDF

Cette phase consiste à représenter le résultat de chaque aspect par le modèle de données RDF (graphes RDF).

- La figure 4.3 illustre le diagramme permettant de représenter l'assertion suivante : « ***le domaine*** *de l'ontologie à créer est : 'chaîne de caractères'* »

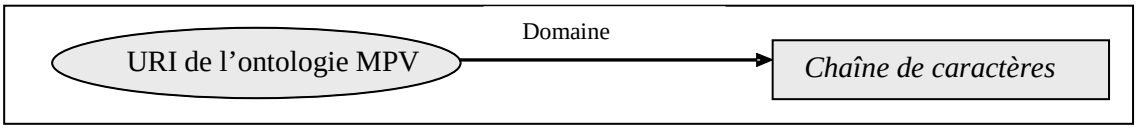


Figure 4.3- Un graphe RDF concernant le premier aspect (le domaine de connaissance)

- Pour représenter maintenant l'assertion suivante : « ***les points de vue*** *à représenter sont : VP_1, VP_2,….., VP_n*», nous utilisons l'objet conteneur **Bag**. Ce dernier est utilisé pour déclarer qu'une propriété possède plusieurs valeurs et qu'il n'y a pas de sens pour l'ordre dans lequel elles sont donnéesLa forme généralisée '**rdf:li**' permet de définir les différentes valeurs du conteneur. (*cf.* Figure.4.4).

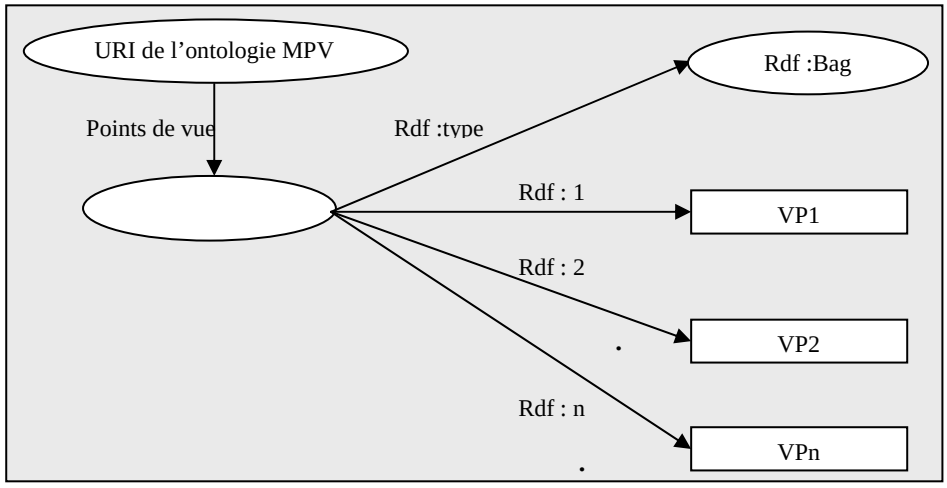


Figure 4.4- Un graphe RDF concernant le troisième aspect (les points de vue)

- Le résultat concernant le quatrième aspect est une liste de termes les plus représentatifs (littéraux). La figure 4.5 illustre le diagramme permettant de représenter, par exemple l'assertion: « ***Les termes*** *les plus représentatifs sont : T1, T2,…Tn »*

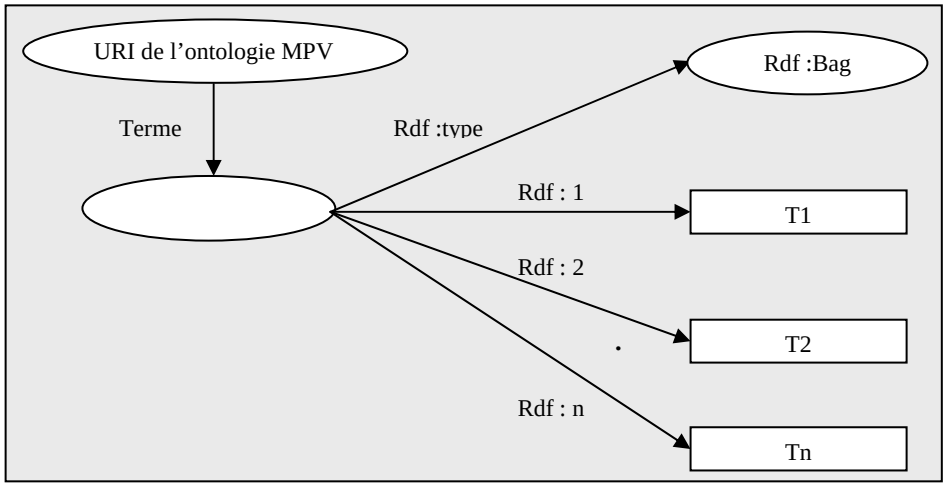


Figure 4.5 - Un graphe RDF concernant le quatrième aspect (la portée)

3.1.3.2 Représentation par la syntaxe RDF

Le RDF propose également une syntaxe XML. Cette phase consiste donc à coder en RDF les différents diagrammes (graphes RDF) présentés dans la phase précédente.

- La sérialisation en XML/RDF du graphe présenté dans la figure 4.3 est comme suit :

```
<rdf:Description about=" URI de l'ontologie MPV">
<Domaine> chaîne de caractère </Domaine>
</rdf:Description>
```

- La sérialisation en XML/RDF du graphe présenté dans la figure 4.4 est comme suit :

```
<rdf:Description about=" URI de l'ontologie MPV">
<Points de vue>
<rdf:Bag>
   <rdf:1 PV1>
   <rdf:2 PV2>
   .
   .
   <rdf:n PVn>
</rdf:Bag></Points de vue >
</rdf:Description>
```

3.2 Étape de conceptualisation

La conceptualisation mérite une attention particulière parce qu'elle détermine le reste de la construction de l'ontologie. L'objectif est d'organiser et de structurer la connaissance acquise durant l'activité d'acquisition de connaissance, en utilisant des représentations semi formelles (tables et graphes) qui sont indépendantes des paradigmes de la représentation de connaissances dans lesquels l'ontologie va être formalisé. Durant cette étape, nous construisons, pour chaque point de vue PV_i, une représentation locale Rep_i selon la perception des experts par rapport au point de vue considéré, ensuite les différentes représentations locales seront connectées par des liens intermédiaires (*cf.* **Figure 4.6**). Pour ce faire, nous distinguons les principales activités suivantes :

1) Construire *un* ***glossaire de concepts globaux***.

2) Etablir, pour chaque point de vue PV_i, une représentation locale :

- Construire *un* ***glossaire de concepts****;*
- Classifier les concepts dans *une* ***hiérarchie de concepts***;
- Représenter les relations qui existent entre les différents concepts par *un* ***diagramme de relations binaires***;
- Identifier les concepts par leurs instances, leurs attributs ainsi que leurs concepts synonyme dans *un* ***dictionnaire de concepts***;
- Décrire les relations dans *une* ***table de relations binaires***;
- Spécifier des contraintes sur les attributs dans *une* ***table d'attributs***;
- Décrire les instances des concepts dans *une* ***table d'instances***;
- Décrire les assertions entre instances dans *une* ***table des assertions***.

3) Etablir des liaisons entre les différentes représentations locales :

- Représenter les liens sémantiques qui existent entre les concepts des différents points de vue dans *une* ***table d'axiomes logiques***;
- Représenter les relations qui existent entre les différents concepts locaux des différents points de vue par *un* ***diagramme de relations globales***.

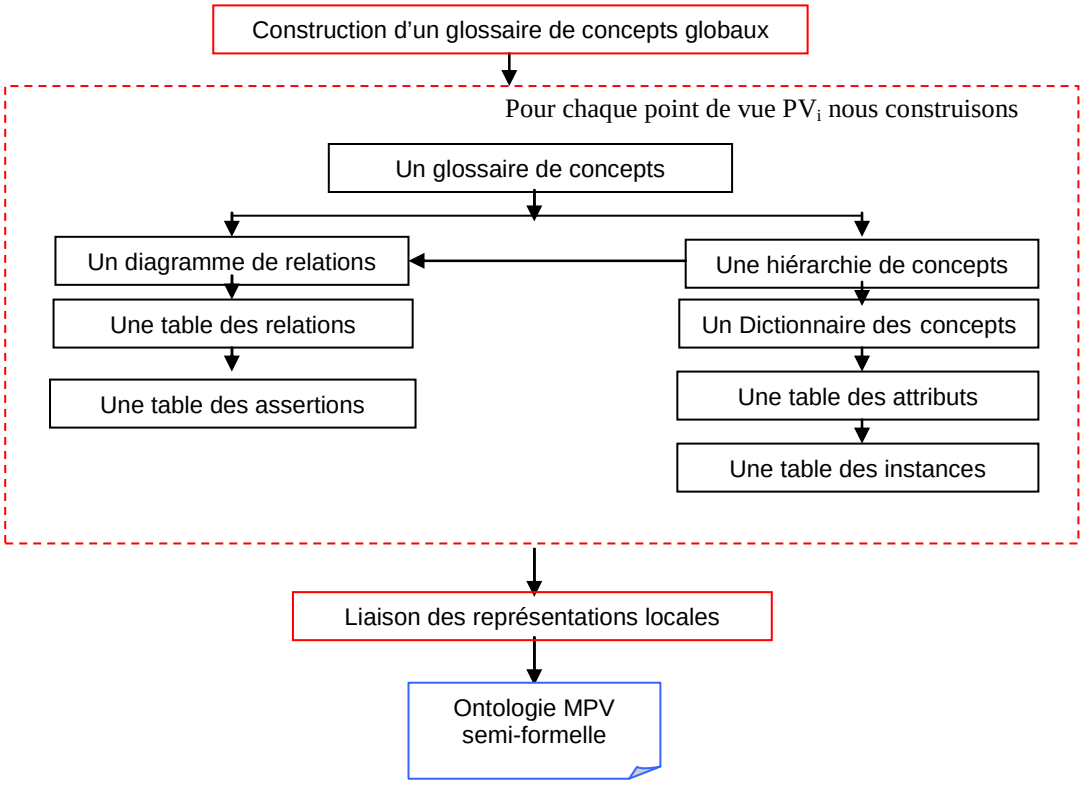


Figure 4.6- Le processus de structuration des connaissances (conceptualisation)

On peut résumer cette étape par les points suivants :

Données en entrée	Termes globaux
Données en sorties	Tables + graphes
Intervenants	Ontologiste + experts

3.2.1 Construction d'un glossaire de concepts globaux (GCG)

Cette étape consiste à construire un *GCC*. Ce dernier recueille les concepts les plus importants et qui sont utiles et potentiellement utilisables dans le domaine que l'on investit et associe à chaque concept identifié une description en langage naturel. Toutes ces sélections et ces descriptions des concepts globaux sont effectuées selon un *point de vue global* pour le domaine considéré.

- **Comment construire le glossaire de concepts globaux ?**

Pour chaque concept global identifié dans le domaine, nous devrons compléter les champs du glossaire suivants:

- *Terme*: Ce champ contient un élément lexical du concept,
- *Description en langage naturel*: Ce champ associe une description du concept en langage naturel afin de fournir le sens du terme,
- *Référence* : Ce champ associe des liens vers les parties du corpus (*sources d'informations*) qui mettent en évidence le sens du concept. Ce qui permet au cas où une ambiguïté sémantique demeure, de revenir au corpus.

Exemple

Table 4.2 - Glossaire global de termes

Terme	Description en langage naturel	Référence
Habitat	Est un endroit qui sert de logement à quelqu'un .	
Appartement	Est un habitat composé d'un ensemble de pièces	-
Locataire	Est une personne qui prend à loyer un appartement	
Agence	Est une entreprise proposant des services pour ses clients	
……	……	

3.2.2 Construction des représentations locales

Pour chaque point de vue PV_i, nous construisons une représentation locale Rep_i selon la perception des experts par rapport au point de vue considéré. Une fois les concepts identifiés par leurs termes, il faut en décrire la sémantique dans un langage semi-formel, en indiquant leurs attributs, leurs instances connues et les liens qu'ils entretiennent entre eux. Pour ce faire, nous suivons les activités suivantes :

3.2.2.1 Construction du glossaire local de concepts

Le glossaire local de concepts (i.e. spécifique à un point de vue), consiste à ne retenir que les concepts du domaine qui sont intéressants dans le cadre du point de vue visé. Les experts du domaine, relatifs à un point de vue, peuvent faire de nouvelles propositions et signaler des concepts qui ne sont pas apparus dans le glossaire de concepts globaux (GCG).

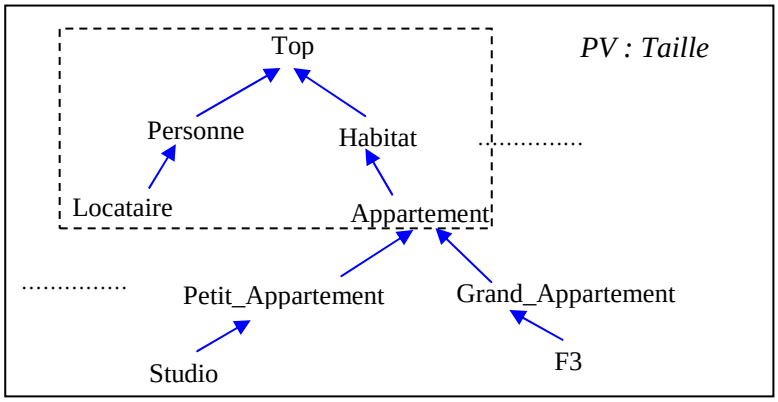
Exemple 1 : Selon le point de vue « *Taille* » nous pouvons recueillir les concepts suivants : {Petit_Appartement, Grand_Appartement, Studio, F1, F2, F3...}.

Exemple 2 : Selon le point de vue « *Finance* » nous pouvons recueillir les concepts suivants: {Appartement_Cher, Appartement_PasCher, HLM, Appartement_LoyerHaut, Locataire_Riche…}.

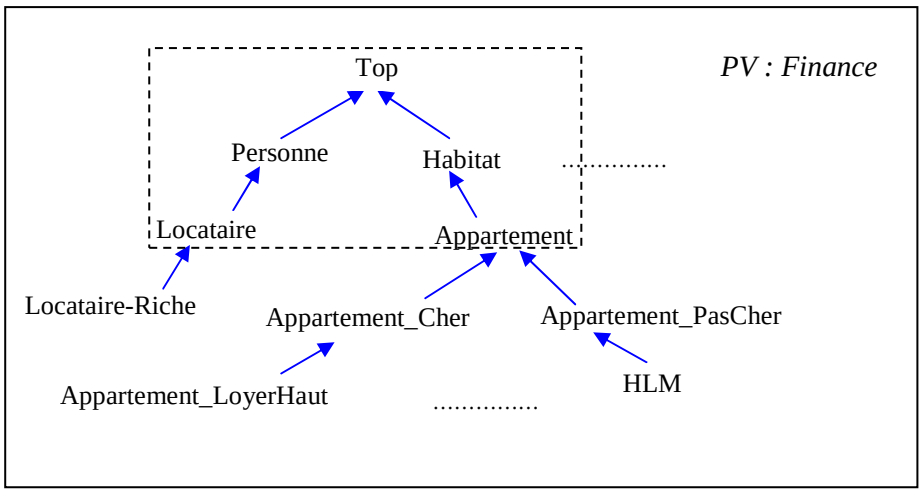
3.2.3.2 Construction de la hiérarchie de concepts

Une hiérarchie de concepts organise un groupe de concepts entre eux sous forme d'une taxonomie en utilisant la relation de généralisation (i.*e. classe/sous-classe*). Dans une hiérarchie de concepts, la relation de généralisation est représentée par un arc (le sens d'un arc est dirigé vers le concept général). Par ailleurs, un concept universel « Top», qui généralise tous les concepts racines, peut être utilisé pour former une seule hiérarchie, afin d'éviter d'avoir des concepts isolés.

Exemple 1 : Une hiérarchie locale de concepts selon le point de vue « *Taille* »



Exemple 2 : Une hiérarchie locale de concepts selon le point de vue « *Finance*»



3.2.3.3 Construction du dictionnaire de concepts

Un concept représente une famille d'individus ayant des caractéristiques communes. Un point de vue du concept reflète un intérêt particulier dans l'observation des individus de ce concept et de leurs caractéristiques ; chaque point de vue considère un sous-ensemble de ces caractéristiques. Ainsi, le dictionnaire de concepts consiste à décrire tous les concepts représentés dans la hiérarchie de concepts, en représentant pour chaque concept ses instances connues, ses concepts synonymes et ses attributs qui sont visibles par rapport au point de vue considéré. Un attribut marqué par * est un attribut visible dans tous les points vue. Par ailleurs, l'ensemble des attributs marqués par * constitue ce qu'on appelle la clé du concept. Cet ensemble permet de distinguer une instance de toutes les autres instances de son concept.

Par exemple, pour le concept *Appartement*, le point de vue *Taille* contient, en plus des attributs clés (i.e. numéro, adresse et étage), les attributs qui concernent la conception, l'architecture et la distribution de l'appartement, tandis que le point de vue *Financier*

s'occupe des dépenses de location de l'appartement. Par exemple l'attribut *surface* est visible du point de vue *Taille*, mais aussi du point de vue *Financier* , car il est nécessaire pour calcules les *impôts* (*Cf.* Figure 4.6, Table 4.3, Table 4.4).

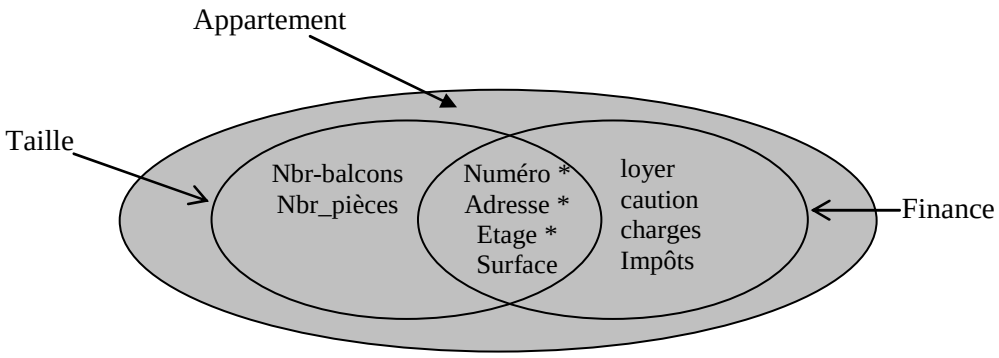


Figure 4.6 - Les attributs clés sont visibles depuis tous les points de vue, les autres attributs sont depuis un ou plusieurs points de vue.

Table 4.3 - Le concept « Appartement » est décrit du point de vue « Taille » comme ne possédant que les attributs qui sont pertinents pour ce point de vue : *surface* et *nbr_pièces*,

PV	Nom du concept	Concepts synonymes	Attributs	Instances
Taille	Appartement	Logement	Numéro * Adresse * Etage * Surface Nbr_pièces Nbr-balcons	Chez-Benali

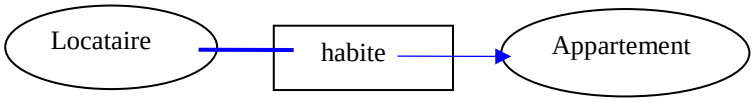
Table 4.4 - Du point de vue « Financier », le concept « Appartement » contient les attributs *loyer*, *charges...*

PV	Nom du concept	Concepts synonymes	Attributs	Instances
Finance	Appartement	Logement	Numéro * Adresse * Etage * Loyer Charges Impôts Surface	-

3.2.3.4 Construction du diagramme de relations

Le diagramme de relations permet de représenter de manière graphique les diverses relations qui existent entre les divers concepts représentés dans la hiérarchie locale de concepts. Dans ce diagramme, les relations sont représentées par des rectangles et des arcs (le sens d'un arc est dirigé vers le concept cible), tandis que les concepts sont représentés par des ovales.

Exemple : La relation « habite » pourrait être employée pour représenter le rapport entre le concept «Locataire», qui est indiqué comme son domaine, et le concept «Appartement », qui est son co-domaine.



3.2.3.5 Constructions de la table des attributs

Les instances d'un concept sont décrites par un ensemble d'attributs. Ces attributs sont définis au niveau du concept. La description complète d'un attribut pour un concept est donnée en terme des descripteurs de contraintes de type, domaine et cardinalité. Ainsi, pour chaque attribut identifié dans le dictionnaire de concepts, la table des attributs consiste à définir le nom de cet attribut, son concept, son type, sa cardinalité (nombre min ou max de valeurs) et son domaine de valeurs (*Cf.* Table 4.5).

Table 4.5- Description de l'attribut *Nbr_pièces* pour les concepts *Petit_Appartement* et *F1*

PV	Nom de l'attribut	Concept	Type	Cardinalité (Min/Max)	Domaine des valeurs
Taille	Nbr_pièces	Petit_Appartement	Entier	1..1	{1, 2}
	Nbr_pièces	F1	Entier	1..1	1

Il est à noter que, la description complète d'un attribut pour un concept détermine le type de cet attribut pour le concept, l'ensemble des valeurs possibles. Affiner un attribut dans un concept C consiste à déterminer un sous-ensemble de l'ensemble des valeurs possibles du sur-concept directe de C.

Par exemple, si on définit un concept *Appartement_bas*, sous le point de vue *Localisation comme étant un* sous-concept du concept *Appartement*, on affine donc l'attribut étage pour inclure seulement les appartements des 5 premiers étages alors, le type de l'attribut étage pour ce concept est tout le domaine {0, 1, 2, 3, 4, 5} (*Cf.* Table 4.6).

Table 4.6- Le concept *Appartement_bas* sous-concept de *Appartement*

PV	Nom de l'attribut	Concept	Type	Cardinalité (Min/Max)	Domaine des valeurs
Localisation	étage	Appartement-Bas	Entier	1..1	[0, 5]

3.2.3.6 Construction de la table des instances

Une instance d'un concept représente un objet de l'ensemble du monde décrit par le concept. Une instance complète du concept, c'est-à-dire une représentation complète de l'individu du monde, a tous les attributs représentés dans le concept. Si l'on regarde une instance d'un concept de façon globale, on peut voir toute son information, les valeurs qu'elle a pour tous les attributs du concept. Par contre, lorsque l'on regarde une instance d'un point de vue particulier, on ne voit que les attributs de l'instance qui sont pertinents pour (connus par) ce point de vue ; on a une vision partielle de l'instance (le point de vue établit une sorte de masque sur l'instance).

Par exemple, L'instance « *chez_Benali* » est considéré comme un *F1* d'après le point de vue « *Taille* » et ne possédant que les attributs qui sont pertinents pour ce point de vue: *surface, nbr_pièces*, etc., tandis que le point de vue *Financier* contient les attributs *loyer, charges*, etc (*Cf.* Table 4.7).

Il est important de noter que, dans chaque point de vue, une instance appartient à son concept local de rattachement et à tous les sur-concepts de celui-ci. Une instance incomplète, pour laquelle on n'a aucune information autre que la clé, appartient au concept global. Lorsque l'on obtient plus d'information sur l'instance, celle-ci peut "migrer" dans les différents points de vue pour être rattachée aux concepts locaux les plus spécialisés (*Cf.* Figure 4.7).

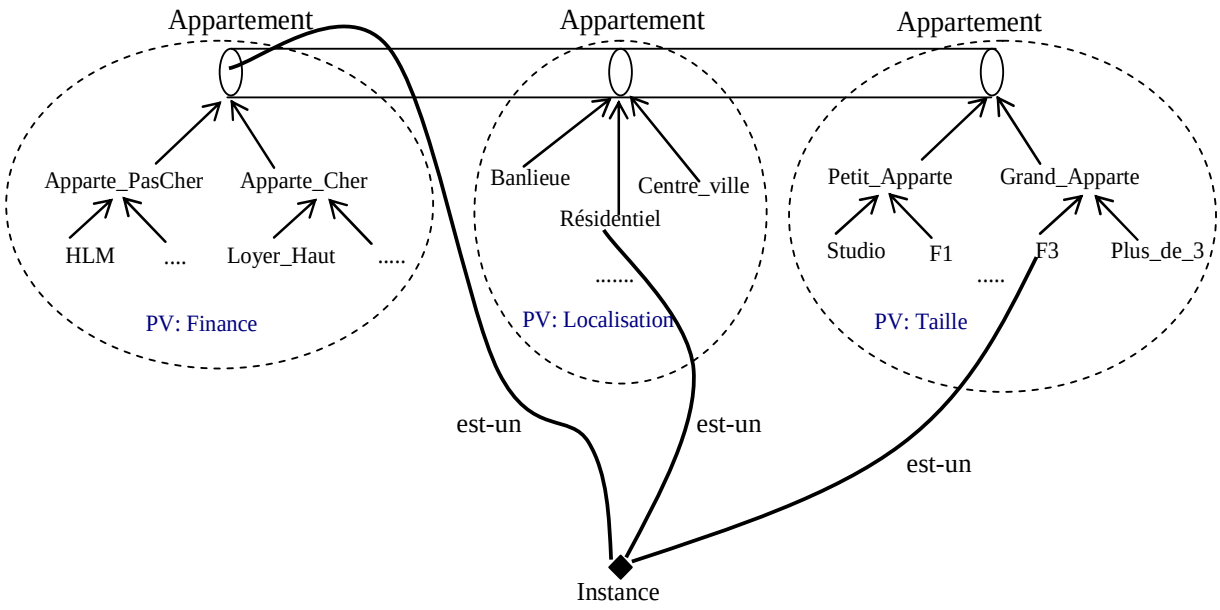


Figure 4.7- Hiérarchies locales des points de vue *Localisation*, *Taille* et *Financier* et description d'un appartement de plus de 3 pièces dans un quartier résidentiel, pour lequel on n'a pas d'information financière.

Table 4.7 - Instance *Chez_Benali* du concept *Appartement* vue des points de vue *Taille* et *Financier*. Les attributs clés (Numéro, Adresse et Etage) sont visibles des deux points de vue.

PV	Nom de l'instance	Concept	Attributs	Valeurs
Taille	Chez_Benali	F1	Numéro * Adresse * Etage * Surface Nbr_pièces	112 6 rue benbadis 25000 4 55 1
Finance	Chez_Benali	Appartement_PasCher	Numéro * Adresse * Etage * Loyer Charge	112 6 rue benbadis 25000 4 3000DA 250DA

- **L'identification d'une instance**

Toute instance appartient à un et un seul concept global. Une instance est identifiée à l'intérieur de son concept global par sa **clé** : les valeurs qu'elle a pour l'ensemble d'attributs qui forment la clé du concept global (une valeur par attribut). La clé est unique, c'est-à-dire que deux instances ayant les mêmes valeurs pour les attributs de la clé représentent la même

instance[1]. Outre l'identification sémantique d'une instance, on peut lui associer un nom (i.e. une étiquette).

Par exemple, une instance du concept *Appartement* est identifiée par les valeurs des attributs clés du concept : numéro, adresse, étape et éventuellement un nom. Par exemple, l'instance (112, "6 rue benbadis 25000", 4) du concept *Appartement* peut être identifiée par le nom "chez_Benali".

- **Appartenance d'une instance à un concept**

Toute instance d'un concept global est visible, par sa clé, depuis tous les points de vue et dans chaque point de vu, elle et rattachée au concept le plus spécialisé pour lequel l'instance satisfait toutes les contraintes sur les attributs. Ainsi, pour qu'une instance puisse être rattachée à un concept, sa valeur[2] doit satisfaire le type du concept. Une instance satisfait le type d'un concept `C`, si les valeurs de ses attributs satisfont les types de ces attributs pour le concept `C`.

3.2.3.7 Construction de la table des relations

Une relation binaire est un composant du niveau terminologique d'une modélisation des connaissances qui associe deux concepts entre eux. La table des relations définie pour chaque relation binaire, qui apparaît dans le diagramme de relations, les noms de ses concepts source et cible, le nom de sa relation inverse (si elle existe) et ses cardinalités (nombre max ou min d'occurrences du concept) source et cible.

Par exemple, la table 4.8 représente la relation binaire *habite* ayant pour domaine et co-domaine respectivement les concepts *Locataire* et *Appartement*. Par ailleurs, la relation *habité-par* étant la relation inverse de la relation *habite*.

Table 4.8- Relation binaire entre deux concepts

Nom de la relation	***Concept source***	***Cardinalité source***	***Concept cible***	***Cardinalité cible***	***Relation inverse***
habite	Locataire	1..n	Appartement	1..1	habité-par

[1] Une instance i_1 est dite identique à une autre instance i_2 si et seulement si la clé de l' instance i_2 identifie aussi i_1.

[2] La valeur de l'instance est la liste des couples attribut-valeur, correspondant aux attributs de son concept d'appartenance.

3.2.3.8 Construction de la table des assertions

Un lien entre deux instances est une assertion d'une relation binaire. Cette relation constitue le type du lien. Un lien obéit à la définition de son type, c'est-à-dire que le premier argument du lien est une instance dans l'extension du domaine de la relation et le second argument une instance dans l'extension du co-domaine. Ainsi, la table des assertions définie pour chaque concept source **Cs** représenté dans la table des relations et pour chaque instance **I** de **Cs**, définit dans la table des instances, les instances du concept cible **Cc** qui sont en relation par **R** avec l'instance **I**.

Exemple : Dans le point de vue *Taille*, l'instance « *Benali* » est en relation avec l'instance « *Chez_Benali* » par la relation « *habite* »

Nom de la relation	Instance source	Instance cible
habite	Benali	Chez_Benali

3.2.4 Liaison des représentations locales

Les différents points de vue produisent des hiérarchies de concepts différentes. Cependant ces hiérarchies ne sont pas complètement indépendantes les unes des autres. Ainsi, cette dernière activité consiste à lier les différentes représentations locales des différents points de vue par des liens intermédiaires ; pour cela deux types de liens sont distinguées : *Passerelle* et *Relation globale*.

3.2.4.1 Passerelles entre points de vue

Une passerelle décrit une règle entre un concept source (ou un ensemble de concepts sources) et un concept cible de deux (ou plusieurs) points de vue différents. En général, une passerelle peut être vue comme une implication entre le concept à l'origine de la passerelle (concept source) et celui situé à l'autre extrémité (concept destination) : être individu du (ou des) concept(s) source(s) de la passerelle implique être individu du concept destination. La notion de passerelle se rapproche de la notion de contrainte. En effet, le fait qu'un individu du concept source implique que l'objet créé soit également instance du concept destination permet de représenter des contraintes entre les concepts des différents points de vue.

La description des différentes passerelles entre les différents concepts des différents points de vue, se fait à travers une table d'axiomes logiques. Chaque axiome comporte, la liste des concepts sources et le concept destination sur lesquels porte l'axiome, une définition en

langage naturel et une expression logique. Par la suite, nous explicitons les différents types de passerelles.

1) **Passerelle d'inclusion unidirectionnelle.** Ce type de passerelle exprime l'inclusion ensembliste entre l'extension d'un concept d'un point de vue, source de la passerelle, et celle d'un concept d'un autre point de vue, destination de la passerelle. En termes de logique, la passerelle peut être vue comme une implication: être instance du concept source implique être instance du concept destination. Ainsi, si on instancie le concept source, l'instance créée est automatiquement rattachée au concept destination.

   Soit PV_1 et PV_2 deux points de vue différents et soient E un concept défini sous le point de vue PV_1 (noté PV_1:E) et D un concept défini sous le point de vue PV_2 (noté PV_2:D). Une passerelle d'inclusion unidirectionnelle de E vers D décrite par :

   pass
   {
   de = PV_1:E
   vers = PV_2:D
   }

   correspond aux assertions :

   $\forall x, x \in PV_1:E \Rightarrow x \in PV_2:D$

   $\forall x \in C$, x satisfait les contraintes de PV_1:E $\Rightarrow$ x satisfait les contraintes de PV_2:D

2) **Passerelle d'inclusion avec plusieurs sources.** Une passerelle unidirectionnelle établit une implication entre un concept source et un concept destination : toute instance du concept source est instance du concept destination. Dans certains domaines d'application, l'instance doit appartenir à plusieurs concepts de différents points de vue pour que l'on puisse déduire son appartenance à un concept destination d'un autre point de vue. Dans ce cas, qui généralise le cas précédent, une passerelle est décrite par la liste de ses concepts sources des différents points de vue et le concept destination d'un autre point de vue.

   Soit E_1 ,…, E_i des concepts définis respectivement dans les points de vue PV_1 , …, PV_i, E_i étant un concept du point de vue PV_i (noté PV_i:E_i) et soit D un concept défini dans le point de vue PV_n, n≠1,…,i. Une passerelle unidirectionnelle des E_i vers D décrite par :

   pass
   {
   de = { PV_1:E_1, PV_2:E_2,... , PV_i:E_i } ;
   vers = PV_n :D
   }

   correspond aux assertions :

$\forall x, x \in \{ PV_1:E_1 \wedge x \in PV_2:E_2 \wedge \ldots \wedge x \in PV_i:E_i \Rightarrow x \in PV_n:D$

$\forall x$, x satisfait les contraintes de $PV_1:E_1$ et celles de $PV_2:E_2$ et celles de $PV_i:E_i \Rightarrow$ x satisfait les contraintes de $PV_n:D$.

Exemple 1 : Passerelle d'inclusion avec plusieurs sources

Concept & PV source	**Concept & PV Cible**	**Description de la passerelle**	**Expression logique**
Plus_TroisPièce (*PV: Taille*) Appart_CentreVille *(PV : Localisation)*	Appartement_Cher (*PV: Financier*)	Tous les appartements de plus de trois pièces qui se trouvent au centre-ville sont des appartements chers	∀ x, Plus_TroisPièce (x) ∧ Appartement_CentreVille(x) ⇒ Appartement_Cher (x)

3) **Passerelle d'inclusion bidirectionnelle**. Une passerelle d'inclusion bidirectionnelle, entre deux concepts C et D de deux points de vue différents, représente l'égalité ensembliste ; une telle passerelle est équivalente à deux passerelles unidirectionnelles : l'une ayant comme source le concept C et comme destination le concept D et l'autre, l'inverse, ayant comme source le concept D et comme destination le concept C. La définition d'une passerelle bidirectionnelle est donc équivalente à la définition des deux passerelles unidirectionnelles correspondantes.

Soit E et D deux concepts définis respectivement dans les deux points de vue PV_1 et PV_2. Une passerelle d'inclusion bidirectionnelle de E vers D peut être décrite comme suit :

pass
{
de = $PV_1:E$
vers = $PV_2:D$
}
et
pass
{
de = $PV_2:D$
vers = $PV_1:E$
}

Ainsi, ce cas de passerelle est considéré comme une double implication logique :

$\forall x, x \in E_1 / PV_1 \Leftrightarrow x \in PV_2:D$

Exemple 2: Passerelle d'inclusion bidirectionnelle

Concept & PV source	**Concept & PV Cible**	**Description de la passerelle**	**Expression logique**
HLM (*PV: Financier*)	Appartement_banlieue *(PV : Localisation)*	Tous les HLM sont dans la banlieue et tous les appartements de banlieue sont des HLM	∀ X, HLM(X) ⇔ Appartement_banlieue(X)

4) **Passerelle d'exclusion bidirectionnelle.** Ce type de passerelle exprime un lien entre deux concepts, de deux points de vue différents, pour lesquels il ne sera pas possible, pour une même instance, d'appartenir en même temps aux deux ensembles d'instances correspondant à ces deux concepts.

Une passerelle d'exclusion entre deux concepts E et D peut être décrite comme suit :

pass
{
de = PV_1:E
vers ≠ PV_2:D
}

et

pass
{
de = PV_2:D
vers ≠ PV_1:E
}

Exemple 3 : Passerelle d'exclusion

Concept & PV source	**Concept & PV Cible**	**Description de la passerelle**	**Expression logique**
Appartement_banlieue *(PV : Localisation))*	Appartement_TrèsCher (*PV: Financier*)	Les appartements de banlieue ne sont pas très cher et les appartements très cher ne sont pas en banlieue	∀ x, Appartement_banlieue(x) ⇒¬Appartement_TrèsCher(x) ∨ Appartement_TrèsCher (x) ⇒ ¬Appartement_banlieue(x)

- **Passerelle d'inclusion et spécialisation de concepts**

Les concepts d'un point de vue sont structurés dans un hiérarchie locale induite par la relation de spécialisation, ≤. Cette relation représente l'inclusion stricte des ensembles potentiels décrits pas les concepts. Ainsi, PV_i:B ≤ PVi:A si l'ensemble potentiel des instances de B est un sous-ensemble strict de l'ensemble potentiel des instances de A .

Une passerelle d'inclusion unidirectionnelle entre deux concepts de deux points de vue différents établit aussi une relation d'inclusion ensembliste. La propriété de transitivité de la relation d'inclusion ensembliste permet l'identification des passerelles implicites entre des concepts de différents points de vue.

Soit E un concept d'un point de vue PV_j et B et A deux concepts d'un même point de vue PV_i i≠j, A étant sur-concept de B (noté PV_i:B ≤ PVi:A). S'il existe une passerelle d'inclusion de PVi:A vers PV_j:E (noté PV_i:A => PV_j:E), alors l'ensemble potentiel d'instances de PVi:A est inclus dans celui de PV_j:E et, comme l'ensemble potentiel d'instances de PV_i:B est inclus dans celui de A/PVi , on déduit, par transitivité, que l'ensemble potentiel d'instances de PV_i:B est inclus dans celui de PV_j:E, donc qu'il y a une passerelle d'inclusion de PV_i:B vers PV_j:E:

Si ((PV_i:B ≤ PV_i:A) et (PV_i:A => PV_j:E)) alors (PV_i:B => PV_j:E)

En général, s'il y a une passerelle d'inclusion allant d'un concept source B d'un point de vue PV_i vers un concept destination E d'un autre point de vue PV_j, alors il y a une passerelle d'inclusion implicite de chacun des sous-concepts de B vers E (Cf. Figure 4.8).

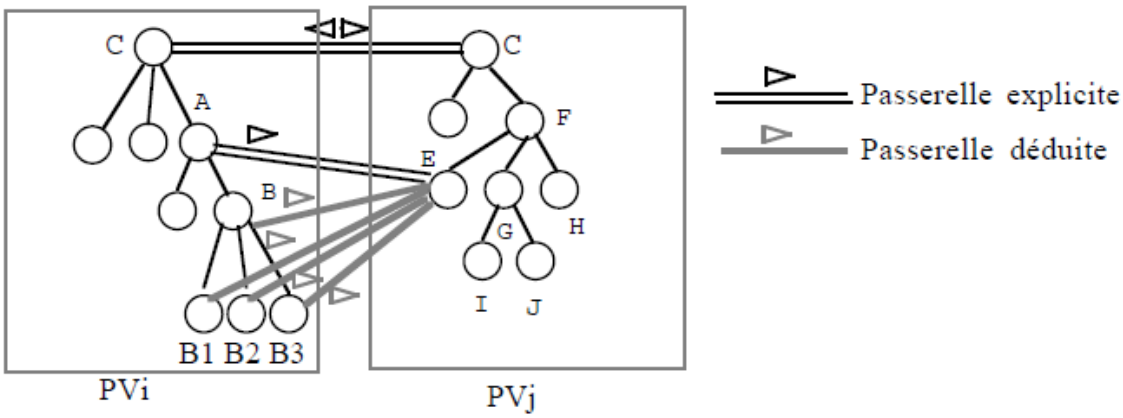


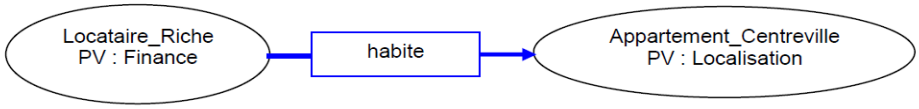
Figure 4.8- La passerelle explicite entre le concept PV_i:A et la concept PV_j:E entraîne l'existence des passerelles implicites de chaque sous-concept de PV_j:E vers le concept PV_j:E.

3.2.4.2 Relations globales

Une relation globale ***R*** est une relation lexicale, qui permet de relier les sous-concepts hiérarchisés différemment selon des points de vue différents et permet d'exprimer un fait général à propos des membres des concepts qui participent à cette relation. Elle est définie par un concept origine ***C*** appelé le domaine de la relation ***R*** et un ensemble un concept destination ***D*** appelé le co-domaine de la relation ***R***. Cela correspond à l'assertion suivante:

$\forall$ X $\in$ PV_{source}:C, $\exists$ Y$\in$ $PV_{destination}$:D, tel que l'instance X est liée à l'instance Y par la relation R

Exemple : un locataire riche est une personne qui ***habite*** dans un appartement au centre-ville



3.3 Étape de formalisation

L'ontologie conceptuelle obtenue dans l'étape précédente doit alors être formalisée. Cette formalisation facilite sa représentation ultérieure dans un langage complètement formel et opérationnel (OWL). Le formalisme de représentation utilisé à cette étape est la logique de descriptions. Une ontologie exprimée en LD contient la description des concepts, des rôles et des individus. Les concepts représentent des ensembles d'objets, possédant des caractéristiques communes. Les individus sont utilisés pour représenter les objets du domaine. Les rôles correspondent à des relations binaires entre ces objets.

Il est à noter que le passage du modèle conceptuel (i.e. l'ontologie semi-formelle) à une ontologie formalisée en LD, s'exprime selon les règles qui sont décrites dans la table 4.9 suivante :

Table 4.9 - Règles de passage du modèle conceptuel à une ontologie formalisée en LD

Formalisation	***Conceptualisation***
Concept	Le nom d'un concept qui apparaît dans le dictionnaire de concepts
Subsumants	Les sur-concepts auxquels le concept est relié dans la hiérarchie de concepts
Subsumés	Les sous-concepts auxquels le concept est relié dans la hiérarchie de concepts
individu	Une instance définie dans la table des instances
Rôle	Une relation binaire définie dans la table des relations ou bien un attribut qui figure dans la table des attributs
Restriction de cardinalité	Cardinalité Maximum (Minimum) exprimée dans le champ "cardinalité" de la table des attributs .
Domaine (co-domaine) d'un rôle	Concept source (cible) d'une relation binaire, défini dans la table des relations binaires.

3.3.1 Formalisation d'ontologies MPV en logique de descriptions

Pour les besoins de la formalisation des ontologies multi-points de vue, nous introduisons, dans la logique de descriptions, les notions suivantes :

1 *Une ontologie multi-points de vue* est une description multiple d'un même univers de discours selon différents points de vue. Elle est définie par un quadruplet $O = <\mathcal{C}^{\mathcal{G}}, \mathcal{R}^{\mathcal{G}}, \mathcal{V}p, \mathcal{M}>$, où :

- $\mathcal{C}^{\mathcal{G}}$ est l'ensemble des concepts globaux,
- $\mathcal{R}^{G}$ est l'ensemble des rôles globaux
- $\mathcal{V}p$ est l'ensemble des points de vue
- $\mathcal{M}$ est l'ensemble des passerelles.

2 *Un point de vue* est une description partielle d'un univers de discours selon une perception particulière. Un point de vue est défini par un triplet $\mathcal{VP}_{\mathcal{K}} = <\mathcal{C}^{\mathcal{L}}, \mathcal{R}^{\mathcal{L}}, \mathcal{A}^{\mathcal{L}}>$, où :

- $\mathcal{C}^{\mathcal{L}}$ est l'ensemble des concepts locaux
- $\mathcal{R}^{\mathcal{L}}$ est l'ensemble des rôles locaux
- $\mathcal{A}^{\mathcal{L}}$ est l'ensemble des individus locaux.

3 *Concept global* est un concept vu par l'ensemble des points de vue avec certaines propriétés[3] communes. Ces dernières, sont visibles par tous les points de vue et constituent ce qu'on appelle la clé du concept global.

4 *Concept local* est un concept qui est vu et décrit localement selon un point de vue donné.

5 *Rôle global* est une relation entre deux concepts locaux définis dans deux points de vue différents.

6 *Rôle local* est une relation entre deux concepts locaux définis dans le même point de vue.

7 *Estampill*e: Nous adaptons le mécanisme d'estampillage[4] utilisé dans [Benslimane et al., 2006] pour permettre la multi représentation des concepts dans le formalisme de la logique de descriptions. Dans notre approche, une estampille (i.e. label) permet de

[3] Le terme propriété est pris au sens large et inclut les relations binaires entre concepts globaux et les relations unaires (attributs).

[4] Les estampilles sont des concepts de modélisation utilisés afin de différencier les représentations multiples d'une même réalité [Balley et al., 2004].

reconnaître pour chaque élément ontologique (i.e. concept, rôle, individu) le point de vue auquel il appartient.

8 *Hiérarchie locale :* Sous un point de vue l'ensemble des concepts locaux sont liés par la relation de *subsomption* (ou de généralité). Cette dernière, permet de les organiser en hiérarchie locale propre au point de vue. Par ailleurs, chaque concept racine (i.e. sommet de la hiérarchie locale) est associé à (subsumé par) un type de concept global.

9 *Passerelle:* L'une des particularités de la représentation multi-points de vue est l'existence d'un canal de communication entre les différents points de vue. Ce canal de communication, appelé passerelle, permet de représenter des liens consensuels entre les concepts locaux de différents points de vue.

10 *Instanciation multiple:* Le mécanisme d'instanciation multiple permet à un individu d'être une instance directe d'un ou de plusieurs concepts. Dans le contexte de notre travail, un individu possède la propriété suivante:

- ***Propriété:*** *un individu est une instance d'un concept global et une instance d'un ou de plusieurs concepts locaux définis dans un ou plusieurs points de vue.*

Un individu possède donc une description de base (i.e. description globale) et peut être décrit partiellement selon un ou plusieurs points de vue.

3.3.1.1 Syntaxe

Définition 1 (Syntaxe d'un concept global).

Soit $S= \{vp_1,\ldots,vp_k,\ldots,vp_m\}$ *l'ensemble des noms de points de vue. Un concept global noté par* $\boldsymbol{C^{\hat{o}}}$*, peut être formé en utilisant les constructeurs de conjonction et de disjonction ainsi que les constructeurs de restriction globaux*[5] *décrits dans la table 4.10 suivante :*

[5] Permet à un rôle d'avoir plusieurs cardinalités et plusieurs domaines de valeurs selon des points de vue différents.

Table 4.10 - Constructeurs de restriction globaux.

Constructeurs de restriction globaux	**Descriptions**
$\forall_{vp1,\ldots,vp\,k}\ d.T$	Définit un nouveau concept global dont toutes ses instances sont décrites, sous les points de vue vp_1 à vp_k, par une propriété de donnée *d* de type T.
$\forall_{vp1,\ldots,vp\,k}\ p.C$	Définit un nouveau concept global qui est relié au concept global C, dans les points de vue vp_1 à vp_k, par la propriété d'objets *p*.
$\leq_{vp1,\ldots,vpk}\ n\ r$	Spécifie la cardinalité minimale ou maximale de la propriété *r* dans les points de vue vp_1 à vp_k.
$\geq_{vp1,\ldots,vpk}\ n\ r$	
$=_{vp1,\ldots,vpk}\ n\ r$	

***C** est un nom de concept global, **T** est un type de données, **d** est une propriété de données (DatatypeProperty), **p** est une propriété d'objets (ObjectProperty), **r** est une propriété (DatatypeProperty ou ObjectProperty) et **n** un nombre entier*.

Définition 2 (Syntaxe d'un concept local).

*Soit $vp_i \in S$. Un concept local, noté **vp_i: Concept,** est défini de la manière suivante :*

< vp_i: Concept > $\rightarrow$ < Concept-Global > | <Concept > $\sqcap$ < Concept > |

< Concept > $\sqcup$ < Concept > | $\neg$< Concept > | $\forall$ <rôle>.< Concept > |

$\exists$ <rôle>. < Concept > | <rôle>.{a, b …}

$\geq$ n <rôle>. < Concept> | $\leq$ n <rôle>. <Concept>

Définition 3 (Syntaxe d'un rôle local).

*Un rôle local, noté **vp_i: R,** peut être défini selon la forme suivante:*

$$vp_i: R\ (C, D)$$

*où R est un nom de rôle local défini dans le point de vue **VP_i**, C et D sont deux concepts locaux définis dans le même point de vue **VP_i***

Définition 4 (Syntaxe d'un rôle global).

*Un rôle global, noté par **$R^{\hat{0}}$**, peut être défini selon la forme suivante:*

$$R^{\hat{0}}\ (vp_i: C, vp_j: D)$$

où R est un nom de rôle global, C et D sont deux concepts locaux définis dans deux points de vue différents.

Définition 5 (**Syntaxe d'une relation de subsomption**).

Sous un point de vue ***VP$_i$****, une hiérarchie locale, notée* ***vp$_i$/$\mathcal{H}$****, est définie par le triplet (*$\mathcal{C}^L$*, ∂, ⊑) où* :

- $\mathcal{C}^L$ est l'ensemble des concepts locaux,
- ∂ est une fonction de $\mathcal{C}^L$ dans $\mathcal{C}^G$ qui associe chaque concept racine (*i.e.* le plus général) S de $\mathcal{C}^L$ à un seul concept global $\mathcal{C}^{\hat{0}}$ de $\mathcal{C}^G$,
- ⊑ est la relation de subsomption utilisée pour exprimer explicitement un lien d'ordre direct selon les deux formes suivantes :

$$\text{vpi: D} \sqsubseteq \text{vpi: C} \qquad \textbf{(1)}$$

où C et D sont deux concepts locaux définis dans le même point de vue ***VP$_i$****,*

$$\text{vp}_i\text{: S} \sqsubseteq \text{C}^{\hat{0}} \qquad \textbf{(2)}$$

où S est le concept le plus général défini dans le point de vue ***VP$_i$*** *et $C^{\hat{0}}$ est un nom de concept global.*

Définition 6 (**Syntaxe d'une passerelle).**

Deux types de passerelles sont distingués:

- *Passerelle unidirectionnelle représente une relation d'inclusion.*
- *Passerelle bidirectionnelle représente une relation d'égalité ou bien une relation d'exclusion*

Une passerelle (uni ou bidirectionnelle) s'exprime de quatre manières :

$\text{vpi: X} \xrightarrow{\sqsubseteq} \text{vpj: Y}$ (***Passerelle d'inclusion unidirectionnelle***) **(1)**

Signifie qu'un individu qui est une instance du concept source X sous le point de vue VPi est aussi une instance du concept destination Y sous le point de vue VPj.

$vp1: X_1 \sqcap \ldots \sqcap vpk: X_k \xrightarrow{\sqsubseteq} vpj: Y$ ***(Passerelle d'inclusion avec plusieurs sources)*** (**2**)

Signifie qu'un individu qui est une instance de chacun des concepts sources ($vp_1:X_1 ... vp_k:X_k$) définis sous des points de vue disjoints est aussi une instance du concept destination Y sous le point de vue VPj.

$vpi: X \xleftrightarrow{\equiv} vpj: Y$ (***Passerelle d'inclusion bidirectionnelle***) (**3**)

Exprime l'égalité entre les deux ensembles d'extensions des deux concepts locaux X et Y définis sous deux points de vue différents.

$vpi: X \xleftrightarrow{\perp} vpj: Y$ (***Passerelle d'exclusion bidirectionnelle***) (**4**)

Signifie que les deux concepts X et Y sont incompatible.

Définition 7 (**Syntaxe d'un individu local**)**.**

Un individu local est une instance d'un concept local défini sous un point de vue donné. Chaque individu local est décrit selon la forme suivante:

$$vp_i: C\ (a)$$

*où **C** est un concept local défini dans le point de vue **vp_i** et **a** est un nom d'individu local.*

3.3.1.2 Sémantique

La sémantique du langage étendu est fournie par une interprétation globale $\mathcal{I}_{\boldsymbol{G}}$, un ensemble d'interprétations locales $\mathcal{I}_{\boldsymbol{vp}}= \{\mathcal{I}_1,\ldots, \mathcal{I}_k,\ldots \mathcal{I}_m\}$, et un ensemble $\{r_{ij}\}_{i\neq j}$ de relations de domaine :

Définition 8 (Interprétations locales).

Pour chaque point de vue $\boldsymbol{VP_K} = \{C^L, R^L, A^L\}$, nous associons une interprétation locale $\mathcal{I}_k= (\Delta^{I_k}, \cdot^{I_k})$ où Δ^{I_k} est un ensemble appelé domaine d'interprétation locale et où $\cdot^{I_k}$ est appelée fonction d'interprétation locale. Cette fonction associe pour chaque concept local $\boldsymbol{A} \in C^L$ un sous-ensemble A^{I_k} de Δ^{I_k}, pour chaque nom de rôle local $\boldsymbol{r} \in R^L$ un sous-ensemble R^{I_k} de $\Delta^{I_k} \times \Delta^{I_k}$ et pour chaque individu local $\boldsymbol{a} \in A^L$ un élément de Δ^{I_k}.

Définition 9 (Interprétation globale).

Une interprétation globale $\mathcal{I}_G = (\Delta^I, \cdot^I)$ *consiste en un domaine d'interprétation globale* $\Delta^I = \Delta^{I_1} \cup \ldots \Delta^{I_k} \ldots \cup \Delta^{I_m}$, *et en une fonction d'interprétation globale* $\cdot^I$ *qui fait correspondre à chaque concept global* $\boldsymbol{C^{\hat{0}}} \in C^G$ *un sous-ensemble de* Δ^I *et à chaque rôle global* $\boldsymbol{r^{\hat{0}}} \in R^G$ *un sous-ensemble de* $\Delta^{I_i} \times \Delta^{I_k}$ *où* $i \neq k$.

Définition 10 (Relation de domaine).

Une relation de domaine r_{ij} *de* Δ^{I_i} *à* Δ^{I_j}, *définit comment deux points de vue différents (**VP$_i$** et **VP$_j$**) interagissent et elle est nécessaire pour évaluer la satisfiabilité des passerelles.*

Nous utilisons la notation fonctionnelle r_{ij} (a) pour dénoter l'ensemble $\{ a' \in \Delta^{I_j} \mid \langle a, a' \rangle \in r_{ij}\}$, le couple $\langle a, a' \rangle$ appartenant à r_{ij} signifie que à partir du point de vue **VP$_j$**, l'élément a de Δ^{I_i}, correspond à l'élément a' de Δ^{I_j}. Par ailleurs, pour chaque sous-ensemble A^{I_i} de Δ^{I_i} nous utilisons r_{ij} (A^{I_i}) pour dénoter $\bigcup_{a \in A^{I_i}} r_{ij}(a)$. En d'autres termes, r_{ij} (A^{I_i}) est une notation pour l'interprétation du concept local A, défini dans le point de vue **VP$_i$**, telle que considéré dans le domaine d'interprétation du point de vue **VP$_j$**.

Définition 10 (Satisfiabilité des passerelles).

Les passerelles sont interprétées, en utilisant la relation de domaine, selon les définitions suivante :

1. $\langle \mathcal{I}_i, r_{ij}, \mathcal{I}_j \rangle \models \text{vp}_i: X \xrightarrow{\sqsubseteq} \text{vp}_j: Y$, Si $r_{ij}(X^{I_i}) \subseteq Y^{I_j}$
2. $\langle \{\mathcal{I}_1, \ldots, \mathcal{I}_k\}, r_{ij}, \mathcal{I}_j \rangle \models \text{vp}_1: X_1 \sqcap \ldots \sqcap \text{vp}_k: X_k \xrightarrow{\sqsubseteq} \text{vp}_j: Y$, Si $r_{1j}(X_1^{I_1}) \cap \ldots \cap r_{kj}(X_k^{I_k}) \subseteq Y^{I_j}$
3. $\langle \mathcal{I}_i, r_{ij}, \mathcal{I}_j \rangle \models \text{vp}_i: X \xleftrightarrow{\equiv} \text{vp}_j: Y$, Si $r_{ij}(X^{I_i}) \subseteq Y^{I_j}$ et $r_{ji}(Y^{I_j}) \subseteq X^{I_i}$
4. $\langle \mathcal{I}_i, r_{ij}, \mathcal{I}_j \rangle \models \text{vp}_i: X \xleftrightarrow{\perp} \text{vp}_j: Y$, Si $r_{ij}(X^{I_i}) \cap r_{ji}(Y^{I_j}) = \varnothing$

3.3.1.3 Exemple de modélisation

Dans la table 4.11 nous présentons l'exemple d'une ontologie multi-points de vue représentée en LD. Dans cet exemple, trois points de vue sont considérés: Taille, Finance et Localisation désignés respectivement par vp_1, vp_2 et vp_3.

Table 4.11 - Exemple d'une ontologie multi-points de vue en LD

Concept global
Appartement $\hat{o} \equiv (\forall_{vp1}$ Nbr_pièces.Number) $\sqcap$ ($\forall_{vp2}$ Loyer.Number) $\sqcap$ ($\forall_{vp1,vp2}$ Surface.Number) ($\forall_{vp1, vp2, vp3}$ Adresse.String) $\sqcap$ ($\geq_{vp1, vp2, vp3}$ 1 Adresse) $\sqcap$ ($\leq_{vp1, vp2, vp3}$ 1 Adresse)
Définit un concept global avec un attribut Nbr_pièces *selon* vp_1*, un attribut* Loyer *selon* vp_2 *, un attribut* Surface *selon* vp_1 *et* vp_2 *et un attribut* Adresse *selon les trois points de vue* vp_1, vp_2 *et* vp_3
Concept local
vp_1: Petit_Appartement $\equiv$ Appartement $\hat{o} \sqcap$ (Nbr_pièces. {1, 2})
Définit un concept local, dans le point de vue vp_1*, comme étant un appartement et dont la valeur de l'attribut* Nbr_pièces *est dans l'ensemble* {1, 2}.
Relation de subsomption
vp_2 : HLM $\sqsubseteq$ vp_2: Appartement_PasCher
Exprime un lien de subsomption entre deux concepts locaux définis dans le même point de vue. En effet, sous le point de vue vp_2 *, tous les HLM sont des appartements pas cher.*
vp_2: Appartement_PasCher $\sqsubseteq$ Appartement $\hat{o}$
Exprime un lien de subsomption entre le concept local Appartement_PasCher, défini sous *le point de vue* vp_2*, et le concept global* Appartement $\hat{o}$
Rôle local /global
vp_2: habite_par (Appartement_Cher, Locataire_Riche)
Définit un rôle local entre deux concepts locaux définis dans le même point de vue vp_2
habite $\hat{o}$ (vp_2: Locataire_Riche, vp_3: Appartement_CentreVille)
Définit un rôle global entre deux concepts locaux définis dans deux PV différents (vp_2 *et* vp_3)
Passerelle unidirectionnelle/bidirectionnelle
vp_2 : HLM $\overset{\equiv}{\longleftrightarrow}$ vp_3: Appartement_Banlieue
Exprime que les deux concepts locaux, définis dans deux points de vue différents, sont équivalents. En effet, tous les HLM sont dans la banlieue et tous les appartements de banlieue sont des HLM
vp_1: Plus_TroisPièce $\sqcap$ vp_3: Appartement_CentreVille $\overset{\sqsubseteq}{\longrightarrow}$ vp_2: Appartement_Cher
Signifie que tous les appartements de plus de trois pièces qui se trouvent au centre-ville sont des appartements chers
Multi-instanciation
vp_1: Petit_Appartement (chez-Benali) vp_3: Appartement_Banlieue (chez-Benali)
Indiquent que l'individu chez-Benali *est une instance de* Petit_Appartement *sous le point de vue* vp_1 *et aussi une instance de* Appartement_Banlieue sous le point de vue vp_3

3.4 Étape de codification

Il reste à opérationnaliser (coder) l'ontologie formelle. Ainsi, cette dernière étape consiste à traduire l'ontologie obtenue à l'étape précédente en une ontologie destinée à servir dans un système informatique. Pour cela, elle doit être spécifiée en un langage de représentation des connaissances doté de capacité d'inférence. Pour ce faire, nous avons choisi d'utiliser le langage OWL-DL qui répond parfaitement à nos besoins en termes d'expressivité et nous avons ajouté quelques nouveaux constructeurs pour le support des points de vue. Ceci fera l'objet du chapitre 5.

4 Conclusion

Les notions *ontologie* et *point de vue* sont complémentaires. En effet une ontologie représente les connaissances du domaine partagées par plusieurs utilisateurs et le point de vue représente les connaissances du domaine qui sont pertinentes dans un contexte donné.

Dans ce chapitre, nous avons présenté une approche pour la représentation d'une ontologie avec des points de vue différents. L'élément clé de notre approche est de permettre la description de ce type d'ontologies, sans éliminer l'hétérogénéité mais en faisant cohabiter l'hétérogénéité (au niveau local) et le consensus (au niveau global). A chaque point de vue correspond une représentation locale. Les différents points de vue partagent à un niveau global, des éléments ontologiques (i.e. concepts et rôles globaux) et des passerelles. Ces dernières, permettent de lier différents concepts locaux provenant de différents points de vue et ainsi d'inférer des informations provenant d'un point de vue en fonction de celles connues dans un autre. Par ailleurs, dans le modèle multi-points de vue proposé, une instance représente un individu particulier d'un concept global, qui est rattaché, dans chaque point de vue, à son concept local d'appartenance le plus spécialisé, par le lien d'appartenance est-un. De ce fait, un individu peut être manipulé à partir d'un seul point de vue sans se préoccuper des autres, ce qui réduit le nombre d'attributs (de propriétés) et de concepts à regarder et simplifie ainsi la hiérarchie de spécialisation (subsomption) de concepts. Enfin, une définition formelle de la syntaxe et de la sémantique, en logique de descriptions, des différents concepts introduits pour le support des points de vue a été présentée dans ce chapitre.

Dans le chapitre suivant, nous présentons le langage Vp-OWL (*Viewpoint Ontology Web Language*) qui est une extension du langage OWL-DL, permettant la représentation des ontologie multi-points de vue dans le cadre du Web sémantique.

Chapitre 5

Vp-OWL : Un langage basé point de vue pour la représentation d'ontologies

«*J'écoute et j'oublie, je vois et je me souviens, je fais et je comprends* »

Proverbe chinois

1 Introduction

Afin de concrétiser le Web sémantique, il faut à la fois disposer des langages capables de représenter les connaissances et formaliser des corpus entiers de ces connaissances (relatives à un domaine particulier) en définissant des ontologies. A cet effet, différents langages de représentation des ontologies ont été proposés. Le plus adopté dans le contexte du Web sémantique est le langage OWL (*Ontology Web Language*).

Dans le Web sémantique, une ontologie, en tant que modèle de connaissance pour un domaine de discours, devrait être construite dans un environnement multi-points de vue, de manière à prendre en compte la diversité des sources de connaissances et les différentes catégories d'utilisateurs.

Dans ce chapitre, nous présentons notre proposition du langage d'ontologie permettant de représenter une ontologie multi-points de vue dans le cadre du web sémantique. Le langage d'ontologie proposé, nommé Vp-OWL, est basé sur le langage d'ontologie OWL recommandé par le W3C[1], et est basé sur le modèle multi-points de vue que nous venons de présenter dans le chapitre 4. Vp-OWL est une extension de OWL-DL, par l'ajout de nouveaux constructeurs permettant d'exprimer les concepts et les notions inhérentes aux points de vue.

[1] http://www.w3.org/TR/2004/REC-owl-ref-20040210/

2 Description des nouveaux constructeurs

Le langage d'ontologies multi-points de vue (Vp-OWL), que nous proposons comme une extension du langage OWL-DL, supporte le même ensemble de constructeurs du langage OWL-DL (*owl: Class, rdfs: subClassOf, etc...*). Par ailleurs, pour permettre la prise en compte de la notion de point de vue dans les ontologie du Web, Vp-OWL étend OWL-DL par l'ajout de nouveaux constructeurs qui sont décrits dans la table 5.1 ci-dessous. Pour chacun de ces constructeurs, ce tableau décrit le nom, l'élément de modélisation associé et une description informelle de sa sémantique exprimant son usage.

Table 5.1 - Nouveaux constructeurs supportés par Vp-OWL

Nouveaux constructeurs	Elément de modélisation	Usage
vpowl:Viewpoint	Ressource	Utilisé pour définir un point de vue.
vpowl:GlobalClass vpowl:LocalClass	Classe	Utilisé pour définir un concept global ou un concept local respectivement.
vpowl:GlobalProperty vpowl:LocalProperty	Propriété	Utilisé pour définir respectivement une propriétés globale ou une propriétés locale.
vpowl:underViewpoint vpowl:onViewpoint vpowl:belongtoViewpoint	Propriété	Utilisé pour spécifier le point de vue des éléments ontologique locaux (classe, propriété ou individu).
vpowl:InclusionBridge vpowl:EquivalenceBridge vpowl:ExclusionBridge	Propriété	Utilisé pour affirmer l'existence d'un mapping entre deux ou plusieurs classes définies dans différents points de vue.

Il est important de préciser, que les différents constructeurs vpowl présentés ci-dessus, sont tous des spécialisations de leurs prédécesseurs OWL. Ainsi, la Figure 5.1 montre le lien de sous-classe (i.e. lien de généralité) qui existe entre les constructeurs du langage Vp-OWL et ceux du langage OWL-DL.

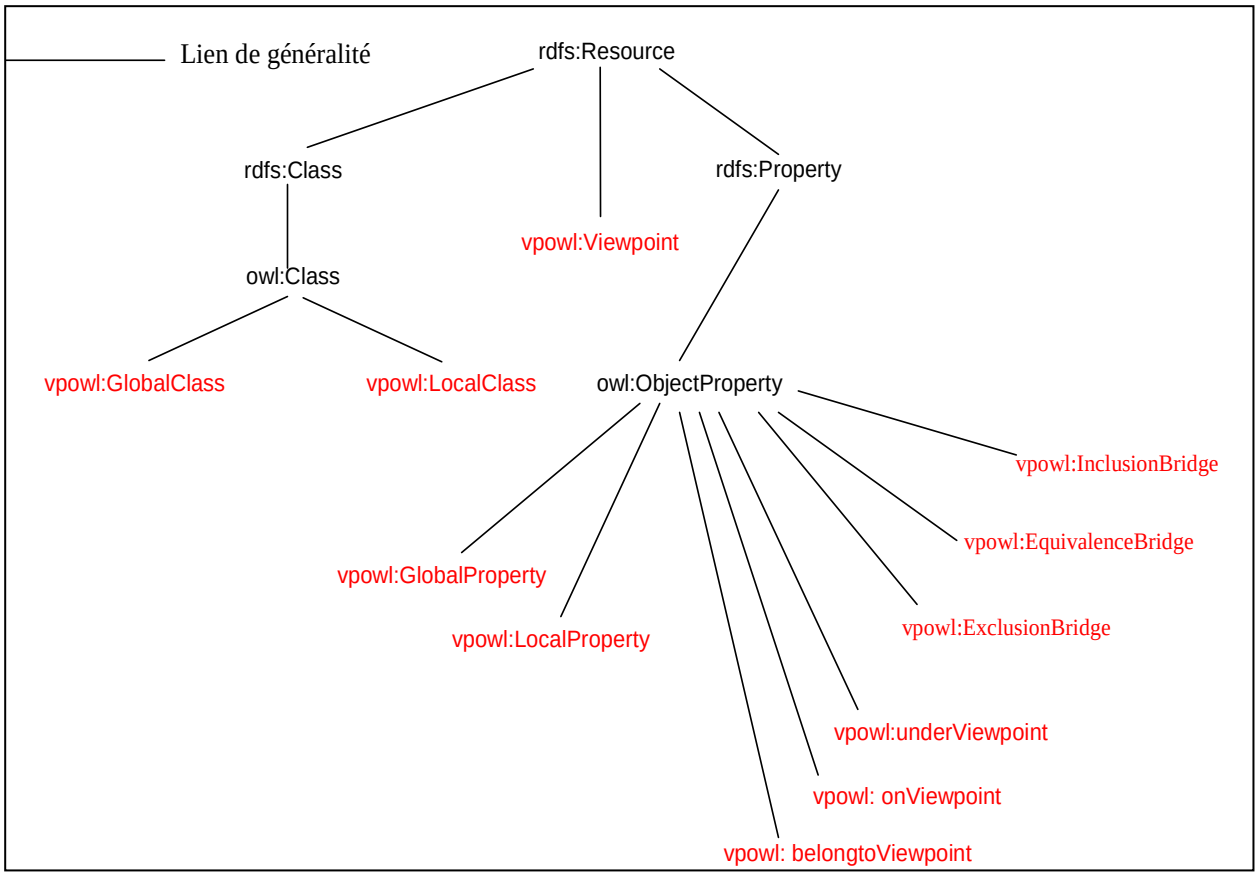


Figure 5.1- Lien de généralité entre Vp-OWL and OWL

2.1 L'élément Viewpoint

L'élément *Viewpoint* est une *ressource*[2] rdfs qui est tout simplement modélisé comme étant une classe OWL. La seule propriété pertinente pour la classe Viewpoint est le nom de ce point de vue, qui est pris en compte à travers la propriété *ViewpointName* (*Cf.* Figure 5.2). Par ailleurs, l'usage de la contrainte *FunctionalProperty* spécifié qu'un point de vue doit avoir un et seul nom.

```
<owl:Class rdf:ID="Viewpoint"/>
        <rdfs:label> Viewpoint </rdfs:label>
</owl:Class>
  <owl:DatatypeProperty rdf:ID="ViewpointName">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Viewpoint"/>
        <rdfs:range rdf:resource="&xsd;string"/>
  </owl:DatatypeProperty>
```

Figure 5.2 - Représentation en OWL de l'élément *Viewpoint*

[2] La *ressource* « rdfs:Resource » est le concept de haut niveau de toute ontologie OWL. Elle représente toute chose décrite par RDFS et OWL

2.2 Les éléments d'estampillage

Les trois constructeurs d'estampillage *vpowl:underViewpoint*, *vpowl:onViewpoint*, *vpowl:belongtoViewpoint* du langage Vp-OWL sont modélisés comme des propriétés de type objet s'appliquant aux classes *LocalClass*, *LocalProperty* et *owl:Thing* respectivement et prenant leurs valeurs dans des objets de la classe *Viewpoint* (*Cf.* Table 5.2, Figure 5.3).

Table 5.2 - Les éléments d'estampillage du langage Vp-OWL

Eléments d'estampillage	Type	Domaine	Co-domaine
vpowl:underViewpoint	rdf: Property	vpowl:LocalClass	vpowl:Viewpoint
vpowl:onViewpoint	rdf: Property	vpowl:LocalProperty / owl:DatatypeProperty	vpowl:Viewpoint
vpowl:belongtoViewpoint	rdf: Property	owl:Thing	vpowl:Viewpoint

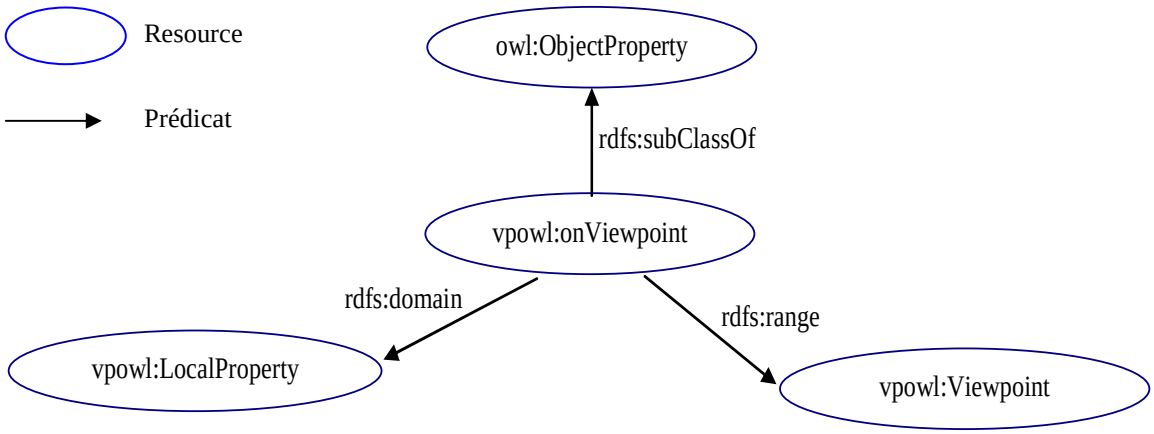


Figure 5.3 - Modélisation de l'élément *vpowl:onViewpoint*, en graphe RDF, comme étant une sous-classe de l'élément *owl:objectproperty* et dont les types de domaine et co-domaine sont les éléments *vpowl:LocalProperty* et *vpowl:Viewpoint*, respectivement.

2.3 Les éléments GlobalClass et LocalClass

Les classes fournissent un mécanisme d'abstraction permettant de regrouper des individus avec des caractéristiques (i.e. des propriétés) similaires. Le langage Vp-OWL distingue deux types de classes: classe globale et classe locale. Ainsi, le premier type permet de regrouper un ensemble d'individus ayant certaines propriétés qui sont vues et partagées par l'ensembles des points de vue. Tandis que, le deuxième type (i.e. classe local) permet de représenter des individus par un ensemble de propriétés qui sont vues et décrites localement sous un point de vue donné. De ce fait, ces deux nouveaux constructeurs sont modélisés comme étant des sous-classes du constructeurs *owl: class*. Par ailleurs, une propriété *vpowl:underViewpoint* est utilisée pour spécifier le point de vue sous lequel une classe locale est définie (*Cf*. figure 5.4).

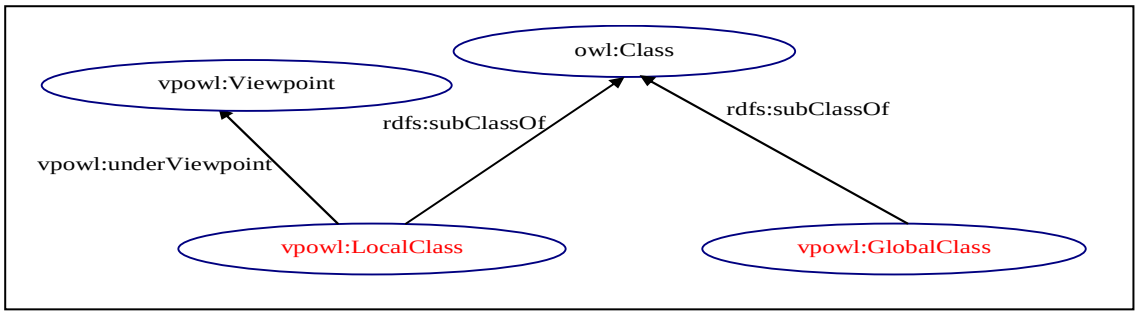


Figure 5.4- Modélisation des éléments vpowl:GlobalClass et vpowl:LocalClass en graphe RDF

2.4 Les éléments GlobalProperty et LocalProperty

L'élément *GlobalProperty* nous permet d'affirmer un fait général (i.e. global) sur les membres de deux classes locales qui sont définies dans deux points de vue différents. Tandis que, l'élément *LocalProperty* permet d'affirmer un fait local sur les membres de deux classes locales qui sont définies dans le même point de vue. La figure 5.5 , représente la modélisation en graphe RDF de ces deux nouveaux constructeurs.

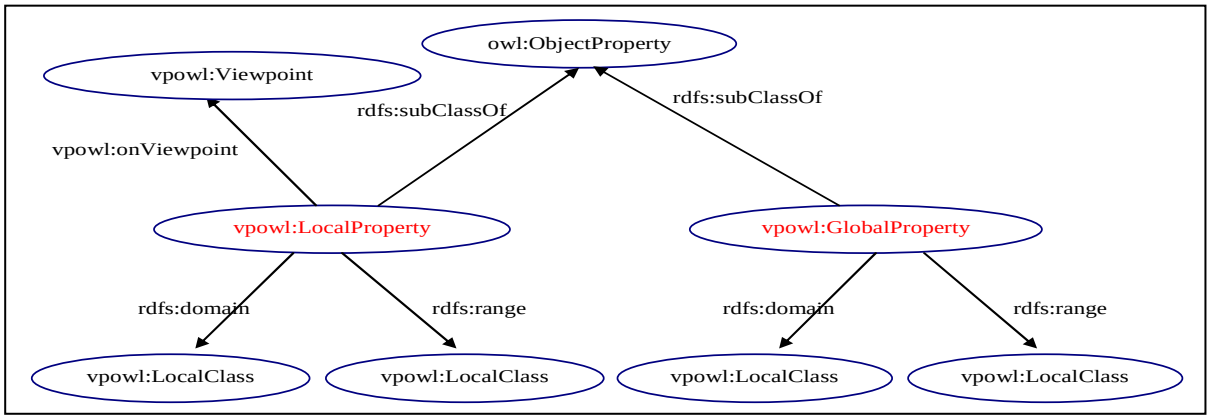


Figure 5.5- Modélisation des éléments *GlobalProperty* et *LocalProperty* en graphe RDF

2.5 Liens entre point de vue

Le langage Vp-OWL utilise les passerelles (*bridge, en anglais*) pour exprimer des rapports entre les classes locaux définis selon différents points de vue, tels que le rapport d'inclusion, le rapport d'équivalence et le rapport d'exclusion. Pour ce faire, Vp-OWL emploie trois constructeurs *vpowl:InclusionBridge vpowl:EquivalenceBridge vpowl:ExclusionBridge* pour exprimer ces types de relation, respectivement. Enfin, ces trois nouveaux constructeurs qui sont modélisés, selon le vocabulaire du langage OWL, comme étant des éléments de type *rdf:Property* permettent d'exprimer un lien sémantique entre des classes locaux définies dans des points de vue différents (*Cf.* Table 5.3, Figure 5.6, Figure 5.7) .

Table 5.3- Les passerelles en langage Vp-OWL

Eléments passerelles	Type	Domaine	Co-domaine
vpowl:InclusionBridge	rdf:Property	vpowl:LocalClass	vpowl:LocalClass
vpowl:EquivalenceBridge	rdf:Property	vpowl:LocalClass	vpowl:LocalClass
vpowl:ExclusionBridge	rdf:Property	vpowl:LocalClass	vpowl:LocalClass

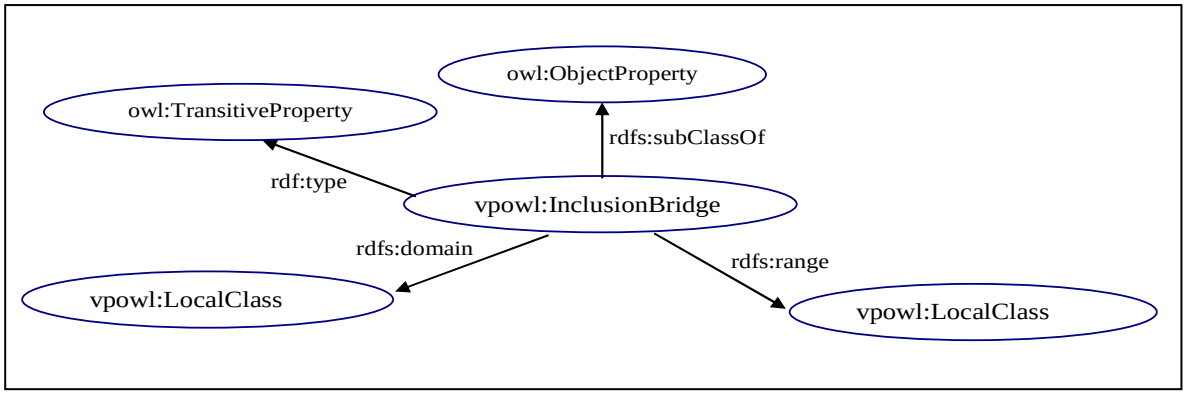


Figure 5.6- Modélisation de l'élément *vpowl:InclusionBridge* en graphe RDF

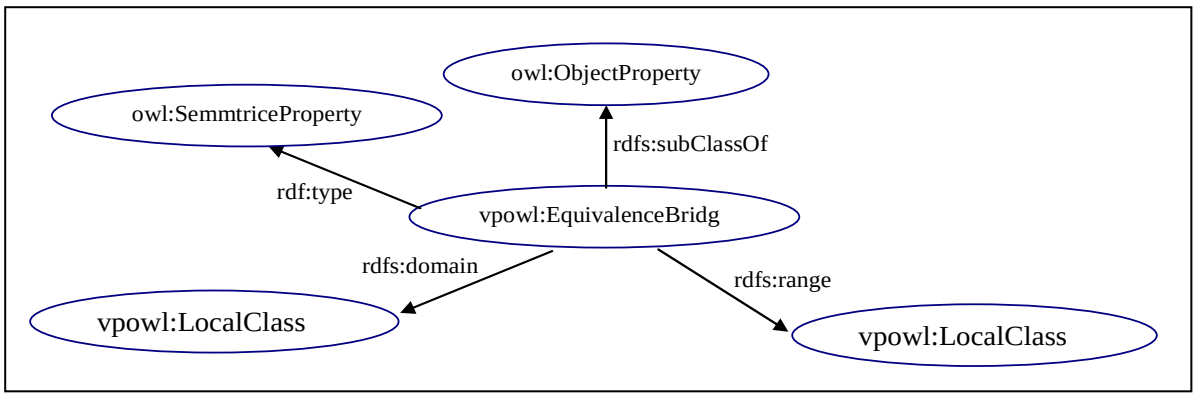


Figure 5.7- Modélisation de l'élément *vpowl:EquivalenceBridge* en graphe RDF

3 Implémentation du langage Vp-OWL

La description des différents constructeurs du nouveau langage Vp-OWL se fait à travers une implémentation d'une ontologie de représentation (i.e. une méta-ontologie), que nous appelons **Onto-Vp-OWL**, permettant d'exprimer le vocabulaire du langage Vp-OWL. Ainsi, une ontologie multi-points de vue d'un domaine donné peut être écrite en utilisant les entités d'une telle ontologie.

Pour la codification de *Onto-Vp-OWL* en OWL-DL, on s'est servi de l'outil Protégé-OWL. Il s'agit d'un environnement (open source) de développement d'ontologies et de systèmes à base de connaissances, qui peut être installé sur différents systèmes d'exploitation. Le plugin OWL est une extension pour protéger, afin de supporter le langage OWL. Protégé-OWL autorise la définition de méta-classes, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de bâtir une ontologie. Par ailleurs, cet outil fournit une interface modulaire facile à exploiter. L'interface utilisateur de sa version courante (3.4.7)[3], illustrée dans la figure 5.8, contient divers onglets associés à des tâches spécifiques pour créer, éditer, visualiser et enregistrer des ontologies OWL.

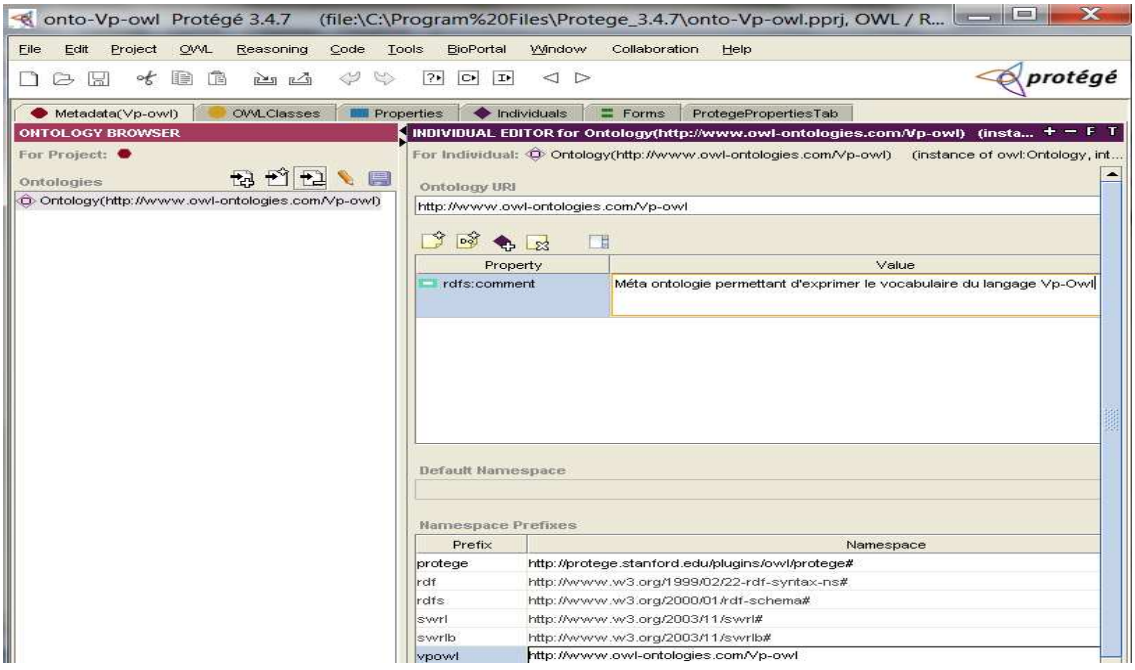


Figure 5.8- Copie d'écran de l'interface principale de l'outil Protégé-OWL (version 3.4.7)

[3] http://protege.stanford.edu/plugins/owl/down

3.1 Méta-classes du langage Vp-OWL

La figure 5.9 montre une extension de la hiérarchie des méta-classes, pour l'implémentation du modèle multi-points de vue (i.e. le langage Vp-OWL), avec GlobalClass et LocalClass comme étant des sous-types de la méta-classe owl:Class, Viewpoint comme étant un sous-type de owl:Thing et les autres (i.e. ClassOfbelongoViewpoint, ClassOfonViewpoint,...) comme étant des sous-types de la méta-propriété owl:ObjectProperty.

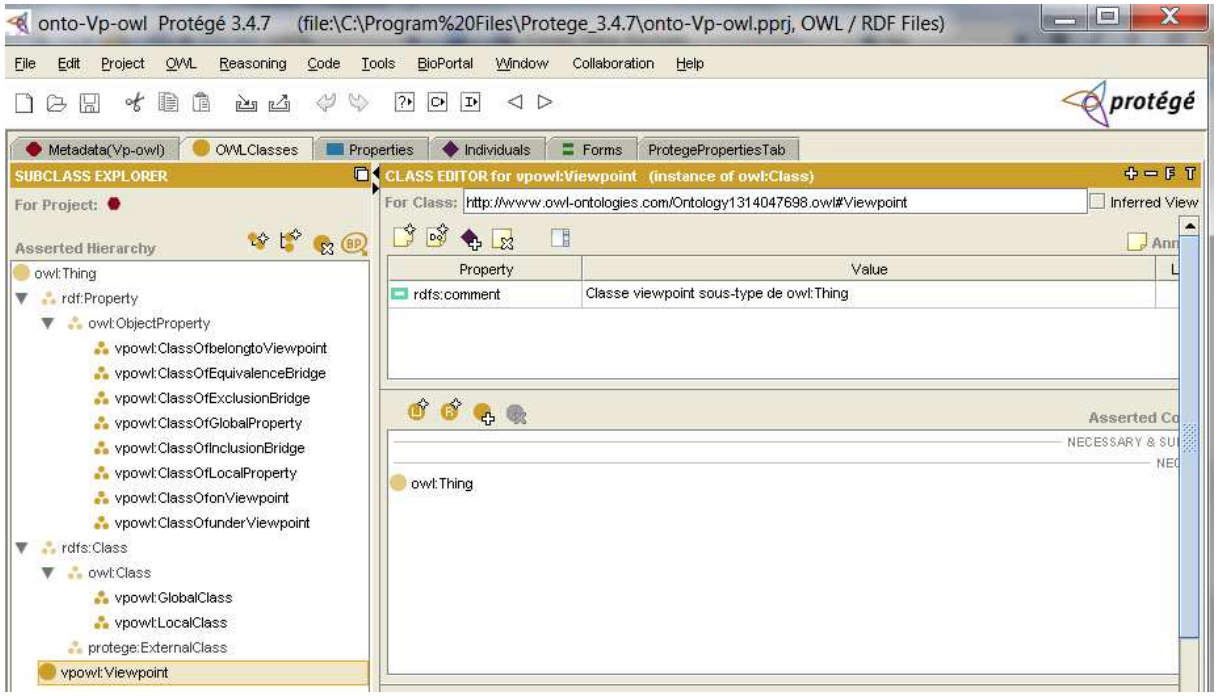


Figure 5.9- La hiérarchie des méta-classes du langage Vp-OWL

3.2 Propriétés du langage Vp-OWL

Une fois les méta-classes (sous-classes de *owl-ObjectProperty*) ont été construites, les différents nouveaux constructeurs (*Cf.* figure 5.10), qui sont de type propriété, peuvent alors être crées comme étant des instances de ces méta-classes. La création d'une propriété Vp-owl nécessite de cliquer dans l'onglet « Individuals » de l'interface de Protégé-OWL, sélectionner la méta-classe à instancier puis créer la nouvelle propriété. Les champs de celle-ci à compléter sont le nom de la propriété ainsi que son domaine et co-domaine.

La capture d'écran de la figure 5.11, illustre la représentation du constructeur vpowl:underViewpoint. Ce dernier, est défini comme étant une instance de la méta-classe vpowl:ClassOfunderViewpoint, qui une sous classe de owl:objectProperty, avec domaine et

co-domaine (i.e. *range*) vpowl:LocalClass et vpowl:Viewpoint respectivement. Ceci, correspond au code source, RDf/XML de OWL, qui est représenté dans la figure 5.12.

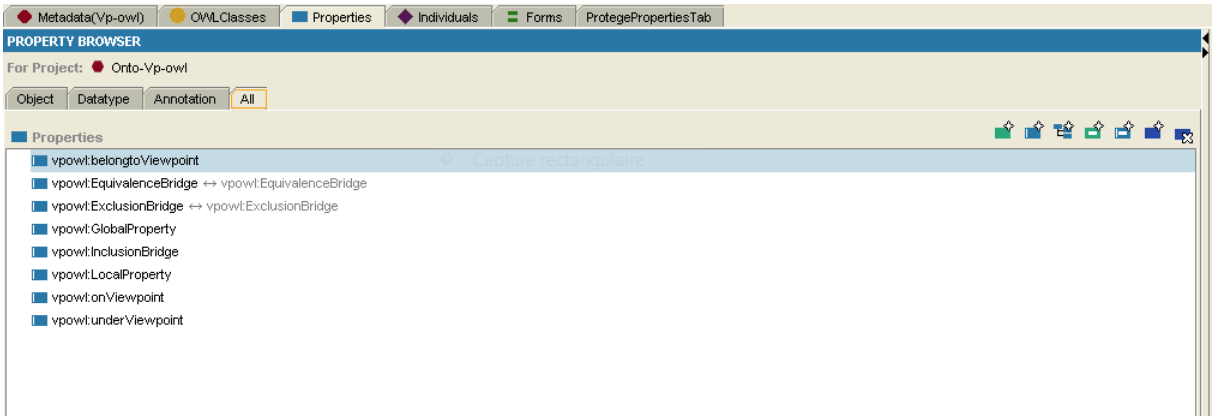


Figure 5.10 - La hiérarchie des propriétés du langage Vp-owl

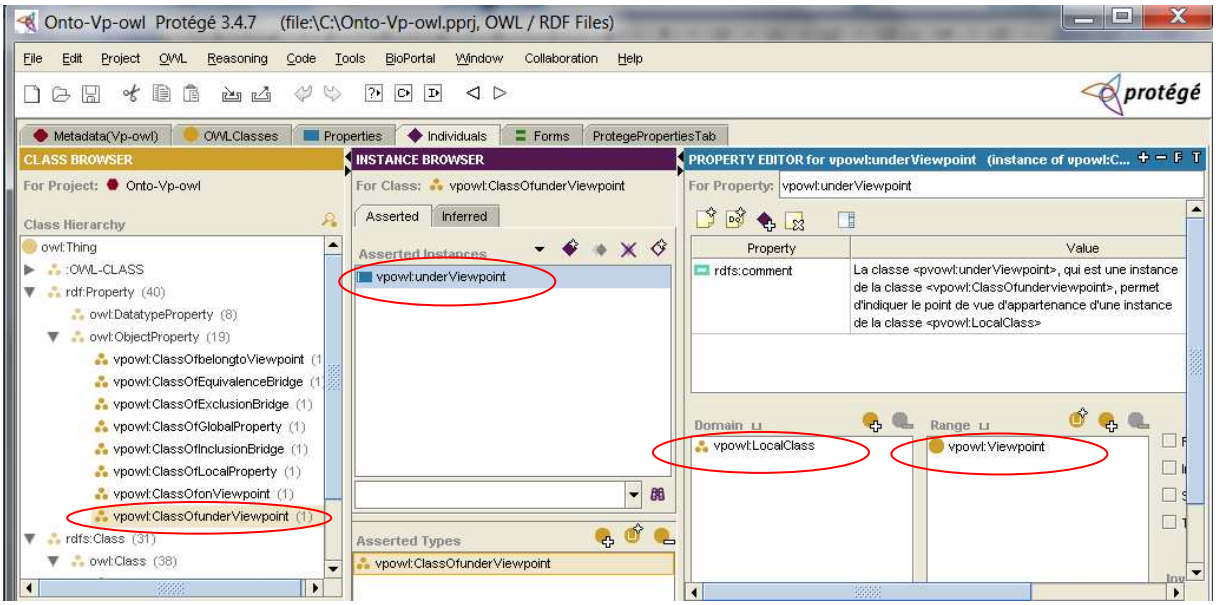


Figure 5.11- Représentation dans Protégé-OWL de l'élément *vpowl:underViewpoint*

```
<owl:Class rdf:about="&vpowl;ClassOfunderViewpoint">
    <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
</owl:Class>
<vpowl:ClassOfunderViewpoint rdf:about="&vpowl;underViewpoint">
    <rdfs:domain rdf:resource="&vpowl;LocalClass"/>
    <rdfs:range rdf:resource="&vpowl;Viewpoint"/>
     <rdfs:comment rdf:datatype="&xsd;string"
     permet indiquer le point de vue d'appartenance d'une instance de la classe
     pvowl:LocalClass </rdfs:comment>
 </vpowl:ClassOfunderViewpoint>
```

Figure 5.12 - Représentation de *vpowl:underViewpoint* dans la syntaxe RDF/XML de OWL

Enfin, la capture d'écran de la figure 5.13 résume l'ensemble des propriétés du constructeur *pvowl:EquivalenceBridge*. Ce dernier est décrit, par exemple, à travers la propriété *owl:equivalenteProperty* pour spécifier qu'il est équivalent au constructeur *owl:EquivalentClass*. Par ailleurs, les propriétés (rdfs:domain et rdfs:range) sont utilisées pour enrichir le sens de notre constructeur *pvowl:Equivalence Bridge* en spécifiant un domaine et un objet. Ainsi, la propriété *rdfs:domain* est utilisée pour déclarer que des objets de la méta-classe *pvowl:LocalClass* doivent être utilisés comme sujet de la propriété *pvowl:EquivalenceBridge*. La propriété *rdfs:range*, quant à elle, permet de spécifier que des objets de la méta-classe *pvowl:Viewpoint* doivent être utilisés comme objet de la propriété *pvowl:EquivalenceBridge*. Enfin, l'usage de *owl:Symmetricproperty* sert à affirmer que la propriété pvowl:EquivalenceBridge est de type symétrique (i.e. *lien bi-directionnel*). Ceci, correspond au code source, RDf/XML de OWL, qui est représenté dans la figure 5.14.

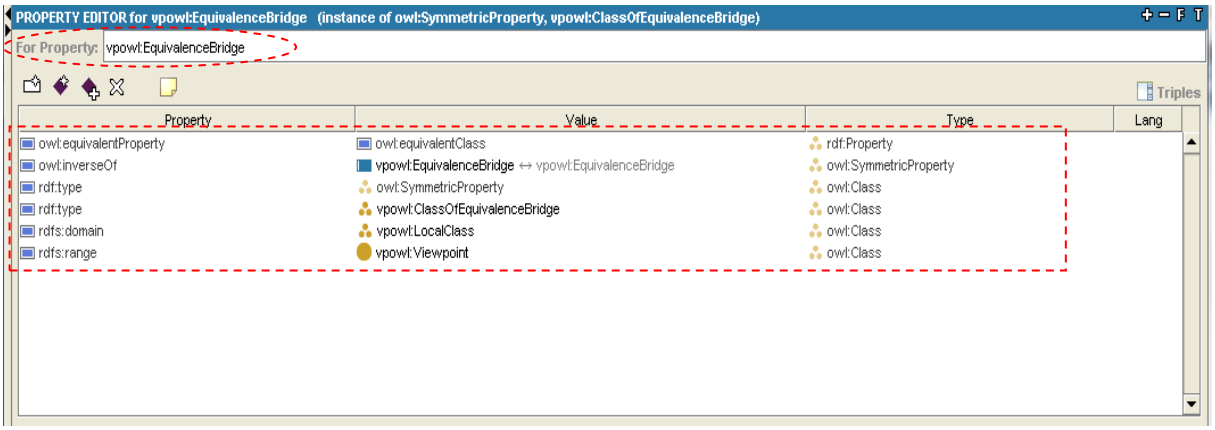


Figure 5.13- Edition des propriétés du nouveau constructeur *vpowl:EquivalenceBridge*

```
<owl:Class rdf:about="&vpowl;ClassOfEquivalenceBridge ">
        <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
</owl:Class>

<vpowl:ClassOfEquivalenceBridge rdf:about="&vpowl;EquivalenceBridge">
   <rdf:type rdf:resource="&owl;SymmetricProperty"/>
   <rdfs:range rdf:resource="&vpowl;LocalClass"/>
   <rdfs:domain rdf:resource="&vpowl;LocalClass"/>
   <owl:equivalentProperty rdf:resource="&owl;equivalentClass"/>
   <owl:inverseOf rdf:resource="&vpowl;EquivalenceBridge"/>
</ vpowl:ClassOfEquivalenceBridge>
```

Figure 5.14- Représentation de *vpowl:EquivalenceBridge* dans la syntaxe RDF/XML de OWL

4 Codification de l'ontologie multi-points de vue en Vp-OWL

Dans cette section, nous montrons les différentes étapes à suivre pour implémenter une ontologie multi-points de vue dans le langage Vp-OWL en utilisant l'outil Protégé-OWL.

4.1 Importation de la méta-ontologie

Une fois la méta-ontologie (i.e. *Onto-Vp-OWL*) à été construite et enregistrée, celle-ci sera importée dans un nouveau projet OWL, concernant l'implémentation d'une ontologie-multi-point de vue. Ainsi, il est alors possible d'utiliser les méta-classes et les propriétés, qui sont définies dans la méta-ontologie importée, pour pouvoir décrire les différents éléments ontologiques de l'ontologie multi-points de vue. Pour ce faire, il suffit de spécifier l'espace de nommage (*namespace*) de la méta-ontologie, c'est-à-dire son URI, qui est supposé être associé au préfixe vpowl (*Cf.* figure 5.15).

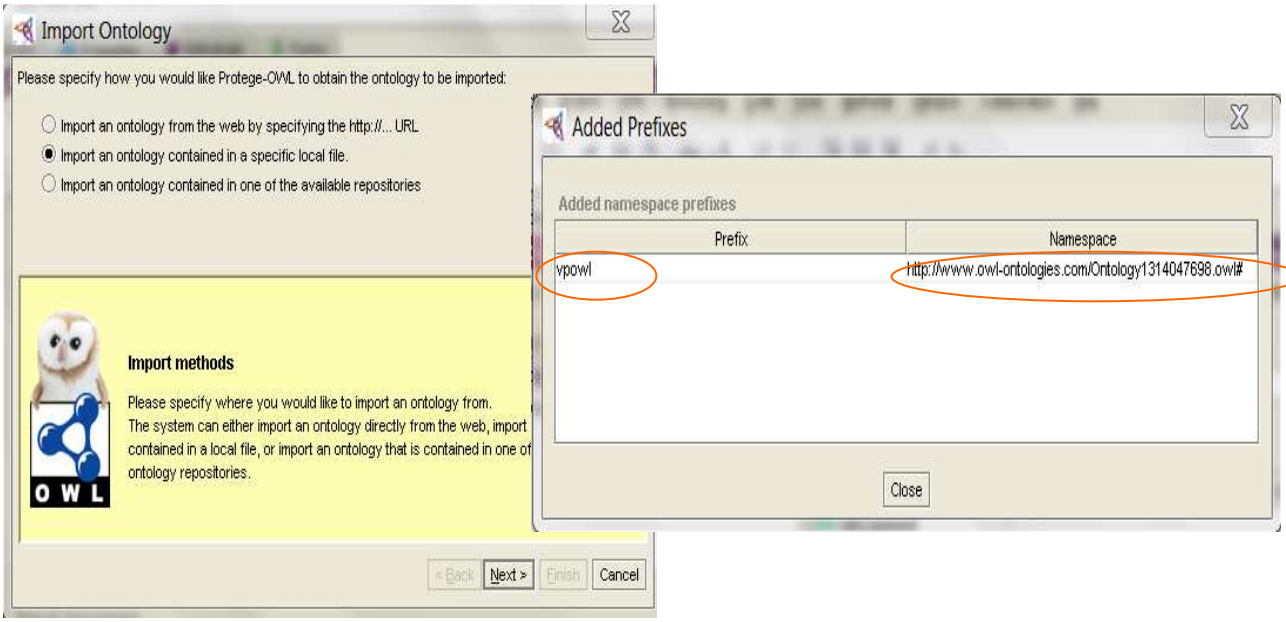


Figure 5.15 - Import de la méta-ontologie Onto-Vp-OWL

4.2 Définition des points de vue

La première étape à effectuer, pendant la codification de l'ontologie multi-points de vue, consiste à définir les différents points de vue liés aux différentes représentations locales. Chaque point de vue donné (i.e. un objet point de vue) est modélisé comme étant une instance de la méta-classe *vpowl:Viewpoint*.

Ainsi, la capture d'écran de la figure 5.16 représente la définition, dans Protégé-OWL, des trois points de vue: *Taille*, *Finance* et *Localisation*, comme instances des classes *VP1*, *VP2* et

VP3, sous-classes de *vpowl:Viewpoint*. Par ailleurs, on indique à l'aide du champ *ViewpointName* le nom du point de vue et le champ *rdfs:comment* correspond à la description en langage naturel de ce point de vue.

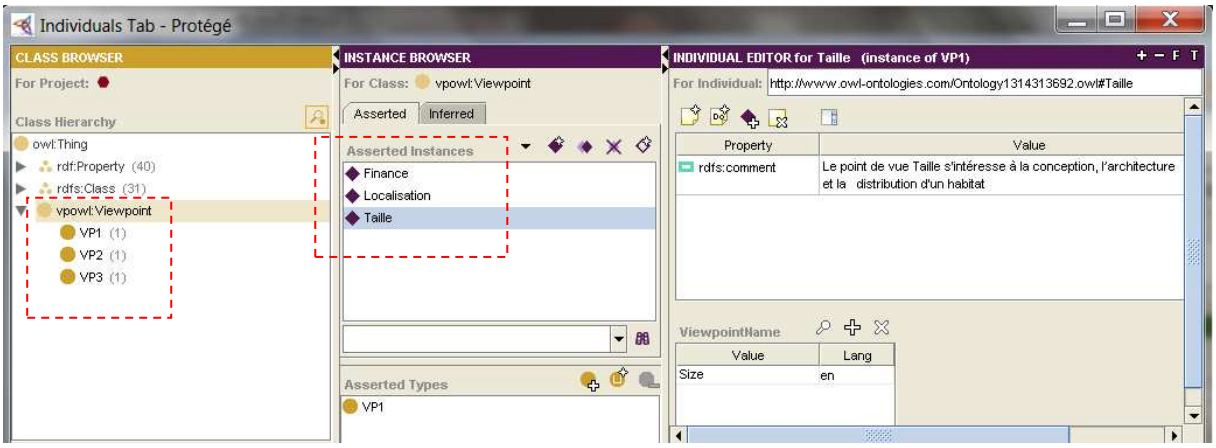


Figure 5.16- Définition des points de vue instances de la classe *vpowl:Viewpoint*

4.3 Définition de la hiérarchie des classes globales

La définition de la hiérarchie des classes globales de l'ontologie-multi-points de vue n'est pas faite selon une approche purement descendante ou ascendante, mais plutôt combinée. Une classe universelle (owl :Thing) est utilisée comme racine pour cette hiérarchie et la création des sous-classes se fait en instanciant la méta-classe *vpowl:GlobalClass*.

La capture d'écran de la figure 5.17, représente la définition d'un ensemble de classes globales (*Habitat*, *Appartement*, *Locataire* et *Agence*). Ces dernières, sont organisées entre elles, sous forme d'une hiérarchie, à travers la relation *rdfs:subClassOf*.

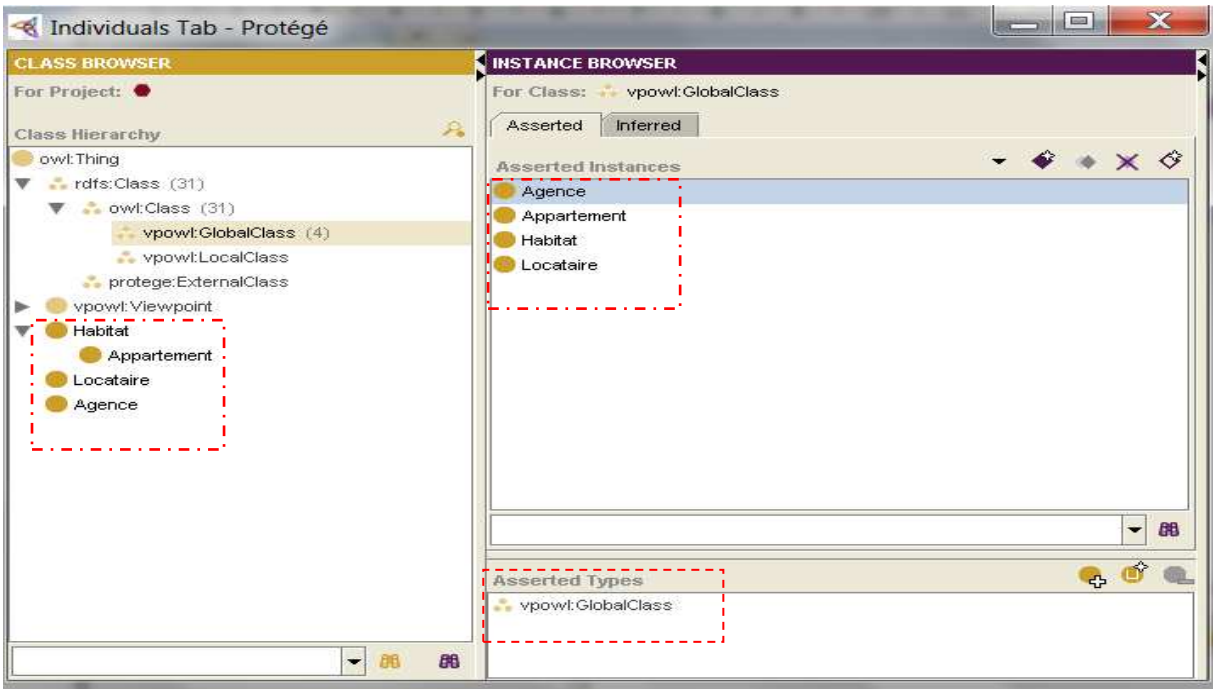


Figure 5.17- Définition des classes globales et de leur hiérarchie

4.4 Définition des propriétés des classes globales

Les propriétés d'une classe globale peuvent être de deux types différents: attribut et relation qui correspondent à '*DatatypeProperty*' et '*ObjectProperty*', respectivement . La création d'une propriété, pour une classe globale particulière, consiste à choisir la classe en question, puis cliquer sur le bouton de création d'une nouvelle propriété.

Les caractéristiques de cette propriété ('datatypeProperty' ou 'objectProperty') peuvent être renseignés; Il s'agit de son domaine (rdfs:domain) et co-domaine (rdfs: range), la relation inverse qui lui correspond s'il s'agit d'une relation, ses valeurs possibles s'il s'agit d'un attribut, un commentaire, etc... Les types de données possibles pour les attributs sont ceux offerts par XML schéma (xsd:boolean, xsd:integer, xsd:positiveInteger, xsd:float, xsd: string, xsd:decimal, etc).

Soit par exemple, la classe *Appartement.* On peut donc lui associer les propriétés adresse (*DatatypeProperty*) et habité-par (*ObjectProperty*). Ainsi, les deux captures d'écran suivantes (*Cf.* figure 5.18 , figure 5.19) rendent compte de la définition de la relation *habité-par* et de l'attribut *adresse*.

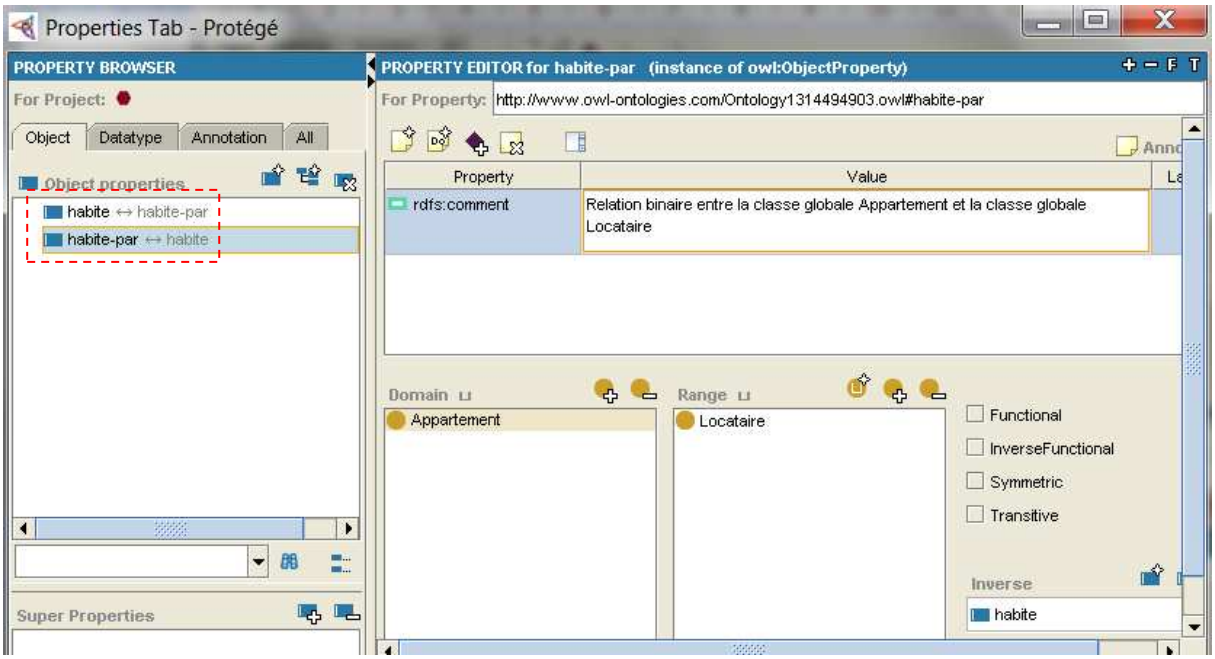


Figure 5.18 - Définition d'une nouvelle propriété (*habité-par*)

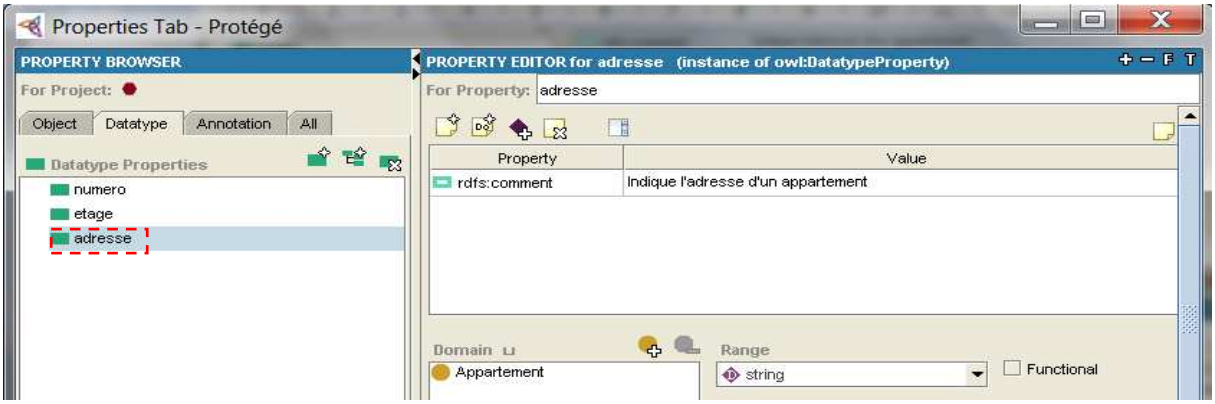


Figure 5.19 - Définition d'un nouvel attribut (*adresse*)

Enfin, la capture d'écran de la figue 5.20 représente la définition, pour la classe globale *Appartement*, d'un nouvel attribut *surface*. Ce dernier est considéré comme étant visible uniquement dans les deux points de vue *Taille* et *Finance*.

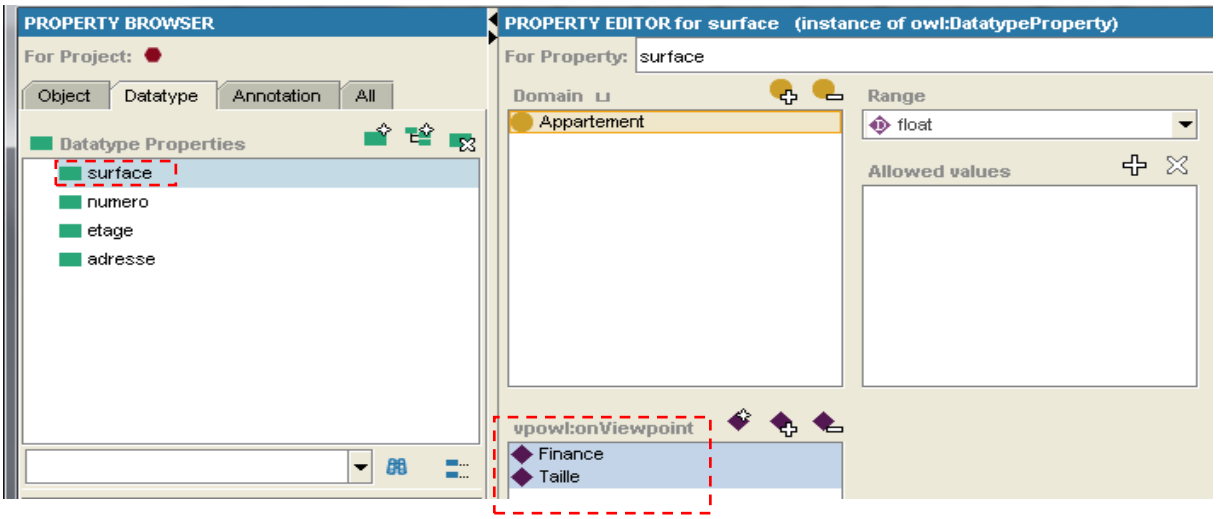


Figure 5.20- Définition d'un nouvel attribut *surface*, visible dans les deux points de vue *Finance* et *Taille*.

4.5 Création des classes locales

Chaque classe locale est crée comme étant une instance de la méta-classe *vpowl:ClassLocal*. Elle est associée à un point de vue donné à travers la propriété *vpowl:underViewpoint*. Sous un point de vue, les classes locales sont organisées entre elles, à travers la relation *rdfs:subClassOf*, formant ainsi, ce que nous appelons une hiérarchie locale. Par ailleurs, chaque classe locale racine (i.e. sommet de la hiérarchie locale) est associée, d'une part, à sa classe globale la plus spécialisée et d'autre part à la classe universelle (owl :Thing), à travers la relation *rdfs:subClassOf*.

Par exemple, dans la figure 5.21, une classe locale *Petit-Appartement* est crée, sous le point de vue *taille*, comme étant une instance de *vpowl:LocalClass*. Par ailleurs, étant donné que la classe locale *Petit-Appartement* est une racine dans sa hiérarchie locale, elle est donc définie comme une sous-classe directe de la classe globale *Appartement* et de la classe *owl:Thing*.

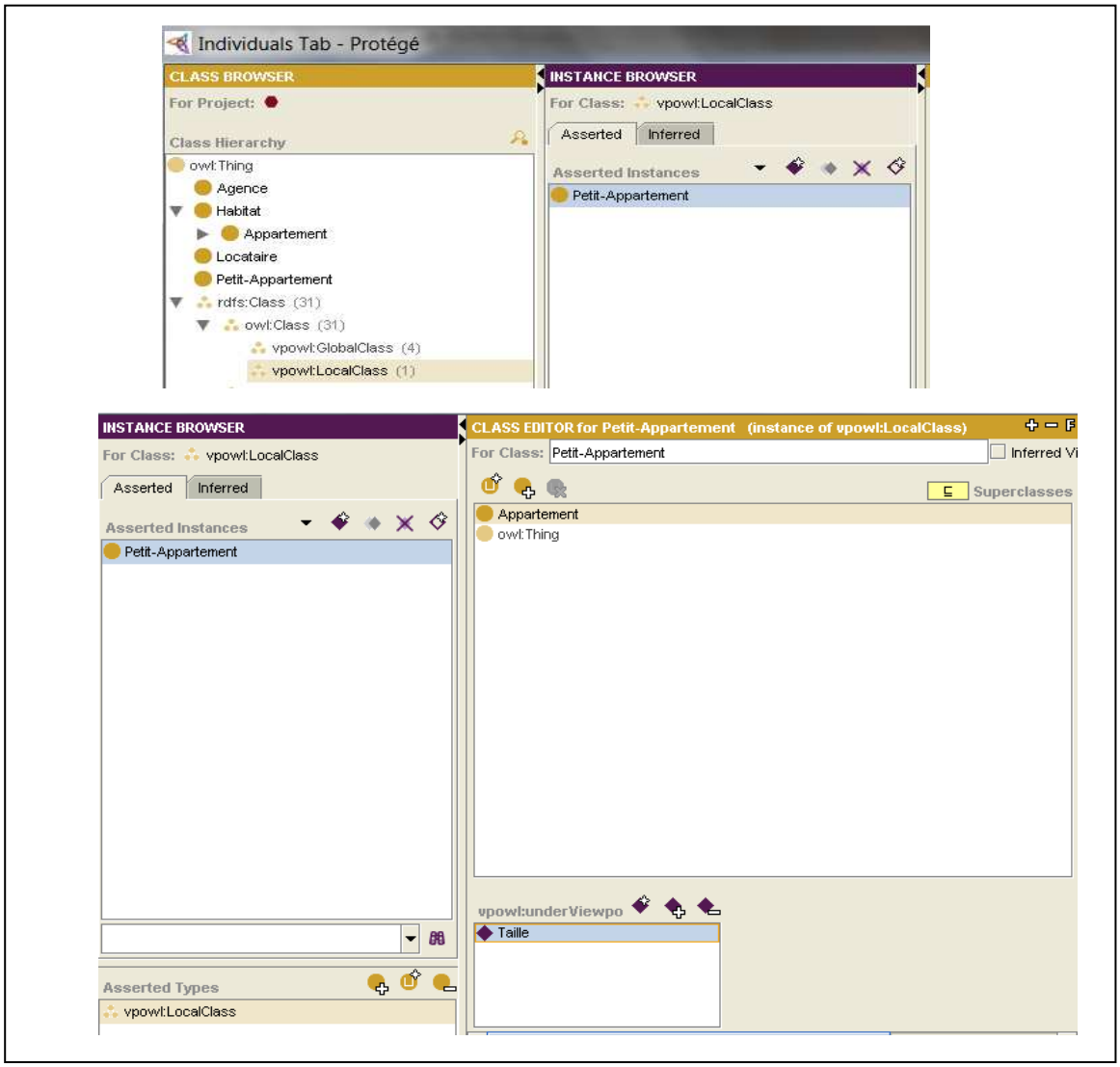


Figure 5.21- Création de la classe locale *Petit-Appartement* sous le point de vue *Taille*

4.6 Définition des passerelles

Une fois les différentes classes locales des différents points ont été définies, il est maintenant possible de spécifier les différentes passerelles qui peuvent exister entre ces classes. Les quatre types de passerelle s'expriment à travers des propriétés de type objet (*i.e.* objectproperty). De ce fait, la définition d'une passerelle donnée consiste d'une part à spécifier son domaine c'est-à-dire la (ou les) classe(s) source(s) et d'autre part, à spécifier son co-domaine c'est-à-dire la classe cible (*Cf.* figure 5.22).

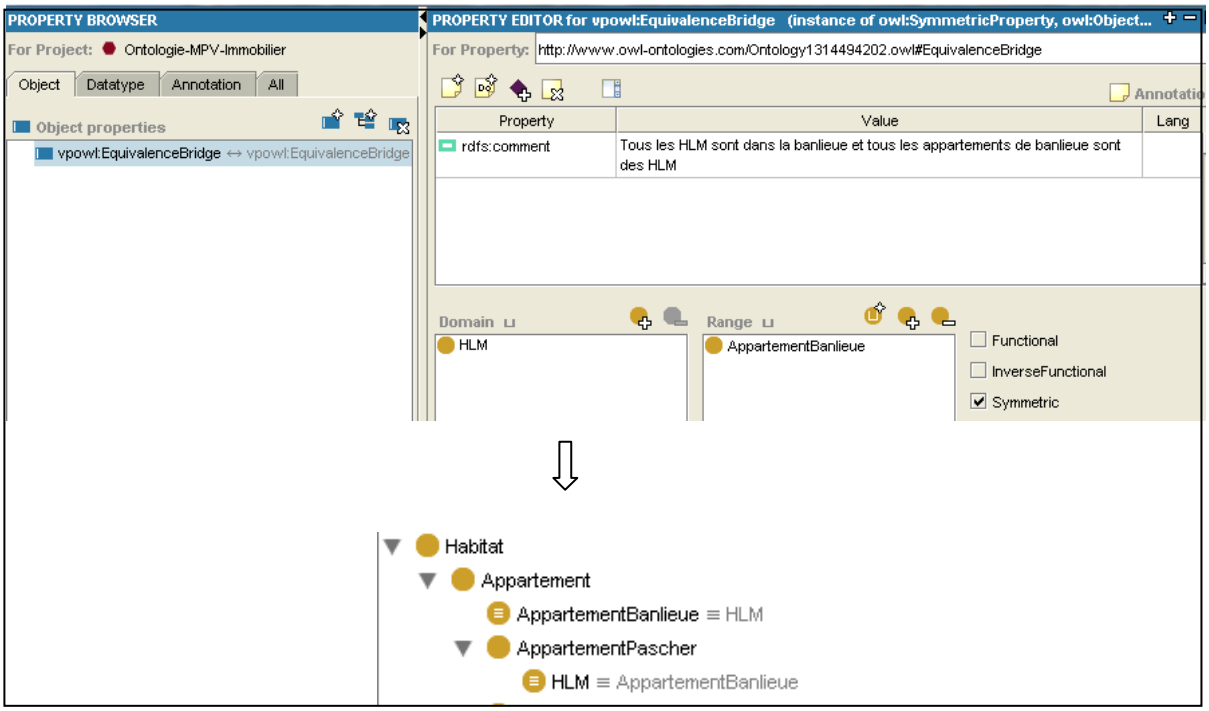


Figure 5.22 - Définition d'une passerelle d'inclusion bidirectionnelle (*EquivalenceBridge*)

4.7 Définition des instances

Une fois que l'ontologie multi-points de vue a été construite, des individus peuvent être crées pour les classes locales de celle-ci. Par ailleurs, la création d'un nouveau individu peut être perçue comme le passage d'un état initial (de l'individu à créer et de sa classe globale d'appartenance, ainsi que ses valeurs valides pour l'ensemble des propriétés communes), à un état final dans lequel l'individu est rattaché aux différentes classes locales des différents points de vue impliqués.

La création d'un individu nécessite de cliquer dans l'onglet « Individuals » de l'interface de Protégé-OWL, sélectionner la classe globale à instancier puis créer une nouvelle instance. Les champs (i.e. propriétés communes) de celle-ci sont à compléter, soit par la valeur de l'attribut, soit par le(s) nom(s) de(s) individus avec lequel (lesquels) cet individu est relié par la relation en question. A la fin, l'individu sera rattaché, à travers la relation de typage (*rdf:type*), à une ou plusieurs classes locales et la propriété *vpowl:belontoViewpoint* sera complétée par la liste des différents points de vue pour lesquels le nouvel individu à été crée.

4.8 Interrogation de l'ontologie multi-points de vue

Notre ontologie multi-points de vue étant désormais constituée, il nous est possible d'effectuer des requêtes avec Protégé-OWL .

- Un premier exemple de requête pourrait être le suivant: Quelles sont les classes locales qui sont définies sous le point de vue Taille? Avec Protégé- OWL, une telle interrogation se formule très simplement. Il nous suffit en effet d'indiquer la classe (*vpowl:LocalClass*), la propriété concernée (*vpowl:UnderViewpoint*) et l'objet désiré (ici, *Taille*) (*Cf.* figure 5.23).

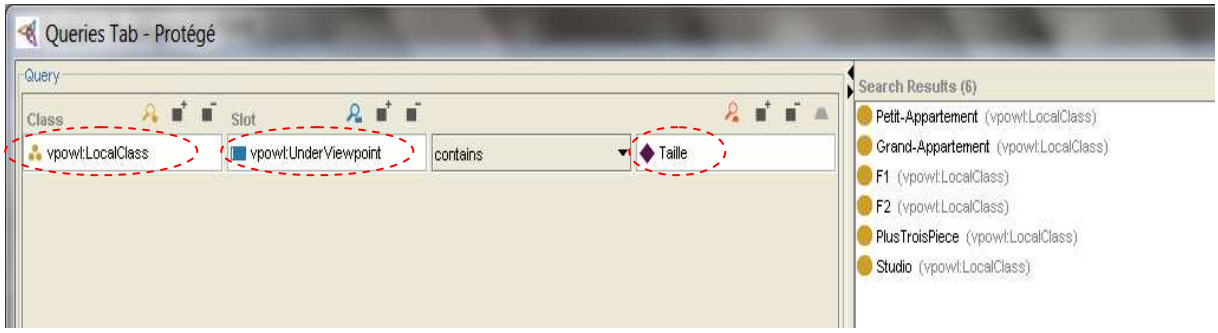


Figure 5.23 - Requête des classes locales qui sont définies sous point de vue Taille

- On pourrait également formuler la requête suivante:

Quelles sont les instances appartenant au point de vue *Localisation*? Comme précédemment, il suffit d'indiquer la classe (*owl:Thing*), la propriété (*vpowl:BelongToViewpoint*) et de choisir l'objet point de vue *Localisation* (*Cf.* figure 5.24).

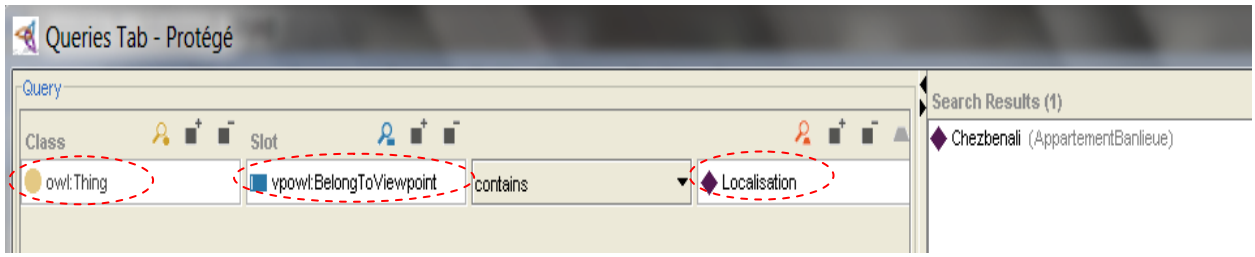


Figure 5.24 - Requête des instances qui appartiennent au point de vue Localisation

4.9 Le test et la validation de l'ontologie multi-points de vue

La conception d'ontologies est un processus complexe. Souvent, les développeurs ont besoin d'appliquer plusieurs itérations avant de pouvoir qualifier l'ontologie de complète. Un outil de développement doit assister l'évolution de l'ontologie et aide l'utilisateur à éviter les erreurs de conception. Dans Protégé-OWL, des approches prometteuses ont été exploitées pour la maintenance des ontologies, basées sur le maintien d'une librairie de jeux de tests à exécuter de temps à autre, afin de vérifier qu'aucun nouveau changement n'a affecté une fonctionnalité existante. Ces jeux de test sont reliés aux définitions de classes formelles dans les logiques de description tel que OWL-DL.

L'une des caractéristiques clés des ontologies multi-points de vue représentées selon notre langage Vp-OWL est qu'elles peuvent être exploitées par des raisonneurs, tel que le système RACER[4] offrant des services d'inférences associés aux logiques de descriptions. Le système RACER, que nous avons utilisé, permet de lire un document au format Vp-OWL (i.e. une ontologie multi-points de vue) puis fournir des services d'inférences pour aider à construire et à maintenir l'ontologie multi-points de vue. Ceci par la détection des inconsistances, des redondances, des dépendances cachées, et des erreurs de classification.

4.9.1 Raisonnement basé sur les logiques de description

Protégé-OWL fournit un accès direct au raisonneur RACER via l'interface DIG[5], qui est une interface/protocole standard pour dialoguer avec les raisonneurs de la logique de description [Turhan, 2011]. Une fois installé, Racer se lance par double clique sur son icône d'application qui ouvre une fenêtre de console et lance le raisonneur avec une communication HTTP activée. L'interface utilisateur courante supporte deux types de raisonnement : le test de consistance « check consistency » et la classification (classify taxonomy), qui peuvent être invoqués via le menu « resoning ».

Le test de consistance consiste à déterminer si une classe peut être instanciée ou non. Il peut être invoqué, pour toutes les classes de l'ontologie ou pour une classe spécifique. Les classes inconsistances sont marquées avec des icônes entourées en rouge, ce qui n'est pas le cas pour notre ontologie.

La classification consiste à inférer une nouvelle hiérarchie de classes, à partir des définitions de celles-ci. Un test de consistance complet est exécuté à l'avance, car les classes inconsistantes ne peuvent pas être classifiées correctement. Finalement, les résultats sont stockés jusqu'à la prochaine invocation du raisonneur et peut être visualisés séparément. L'interface de Protégé-OWL permet de visualiser en parallèle la hiérarchie construite par l'utilisateur et celle inférée par le raisonneur (*Cf.* figure 5.25). Les classes qui ont changé de subsumant sont affichées en bleu, avec la possibilité d'afficher les explications qui les concernent.

[4] http://www.sts.tu-harburg.de/~r.f.moeller/racer/

[5] Description Logic Implementers Group

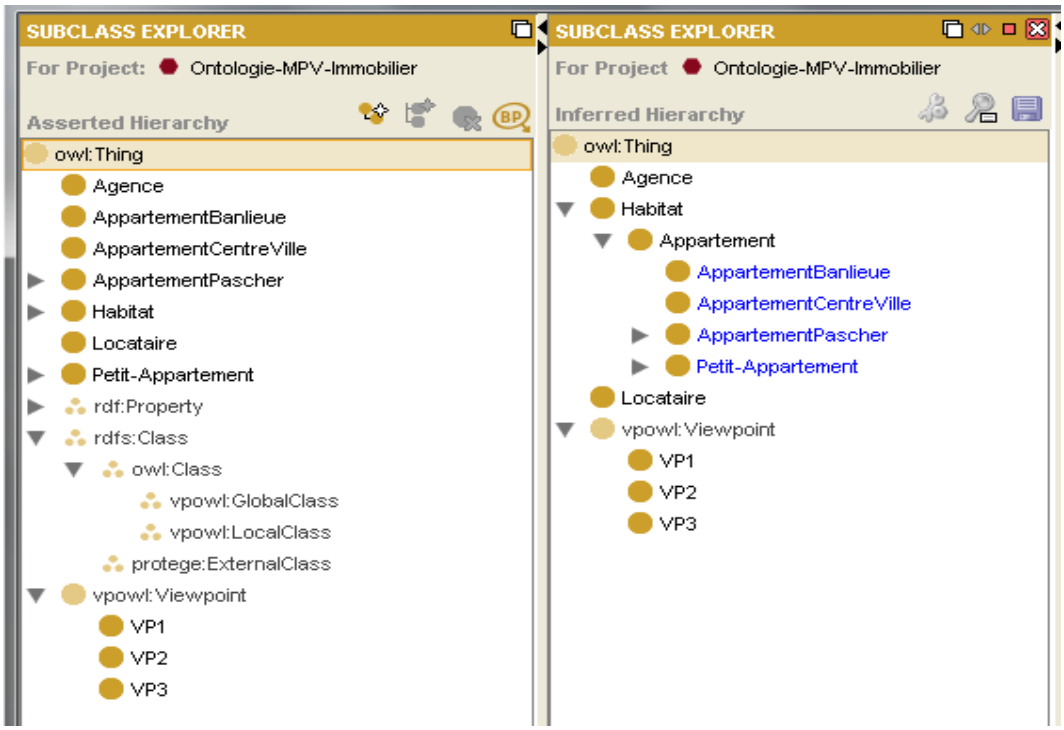


Figure 5.25 - Le résultat de la classification par le raisonneur RACER

4.9.2 Test de l'ontologie

Le plugin OWL fournit un mécanisme pour exécuter des tests supplémentaires (configurés) sur l'ontologie construite. Ces tests sont des petits programmes java prédéfinis (liste disponible à travers le menu « OWL », test settings), qui peuvent être étendues par les programmeurs. Ils prennent une classe, une propriété, un individu ou l'ontologie comme entrée, vérifient des conditions sur celles-ci, et retournent un message d'erreur dans le cas d'échec. Un exemple de test prédéfini est celui qui assure l'invariante « l'inverse d'une propriété transitive et aussi transitive ». Si une propriété, dans l'ontologie, a violé les conditions qui lui sont attachées, le système affiche une erreur et peut même offrir la possibilité de la corriger.

5 Code Vp-OWL généré par Protégé-OWL

Une fois notre ontologie multi-points de vue construite et enregistrée, nous obtiendrons un document au format Vp-OWL. Dans ce qui suit, nous présentons quelques déclarations selon la syntaxe Vp-OWL:

Points de vue

```
<owl:Class rdf:ID="VP1">
    <rdfs:subClassOf rdf:resource="&vpowl;Viewpoint"/>
</owl:Class>
<owl:Class rdf:ID="VP2">
    <rdfs:subClassOf rdf:resource="&vpowl;Viewpoint"/>
</owl:Class>
<owl:Class rdf:ID="VP3">
    <rdfs:subClassOf rdf:resource="&vpowl;Viewpoint"/>
</owl:Class>
<VP1 rdf:ID="Taille">
    <vpowl:ViewpointName xml:lang="en">Size</vpowl:ViewpointName>
</VP1>
<VP2 rdf:ID="Finance">
    <vpowl:ViewpointName xml:lang="en">Finance</vpowl:ViewpointName>
</VP2>
<VP3 rdf:ID="Localisation">
    <vpowl:ViewpointName xml:lang="en">Localisation</vpowl:ViewpointName>
</VP3>
```

Définit trois points de vue: Taille, Finance et Localisation, instances de VP1, VP2 et VP3.

Classe globale

```
< vpowl:GlobalClass rdf:ID="Appartement">
    <rdfs:subClassOf rdf:resource="#Habitat"/>
</ vpowl:GlobalClass >
<owl:DatatypeProperty rdf:ID="nombre_piece">
    <rdfs:domain rdf:resource="#Appartement"/>
    <rdfs:range rdf:resource="&xsd;positiveInteger"/>
    <vpowl:onViewpoint  rdf:resource="#Taille"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="adresse">
    <rdfs:domain rdf:resource="#Appartement"/>
    <rdfs:range rdf:resource="&xsd;string"/>
    <vpowl:onViewpoint  rdf:resource="#Finance"/>
    <vpowl:onViewpoint  rdf:resource="#Taille"/>
    <vpowl:onViewpoint  rdf:resource="#Localisation"/>
</owl:DatatypeProperty>
```

Définit une classe globale *Appartement* avec un attribut *nombre_piece* selon le point de vue *Taille* et un attribut commun adresse selon les points de vue *Taille*, *Finance* et *Localisation*.

Classe locale
<vpowl:LocalClass rdf:ID="**Petit-Appartement**"> <vpowl:UnderViewpoint rdf:resource="**#Taille**"/> <rdfs:subClassOf rdf:resource="**#Appartement**"/> <owl:Restriction> <owl:onProperty rdf:resource="**#nombre-piece**"/> <owl:oneOf rdf:parseType="Collection"> <owl:hasValue rdf:resource="**#1**"> <owl:hasValue rdf:resource="**#2**"> </owl:Restriction> ..... </vpowl:LocalClass>
Définit une classe locale *Petit-Appartement*, sous le point de vue *Taille*, comme étant une sous classe de la classe globale *Appartement* et dont l'image de l'attribut *nombre-piece* est l'ensemble des éléments de la liste de valeurs entières {1, 2}.

Propriété locale
<vpowl:classOfLocalProperty rdf:ID="**habite-par**"> <rdfs:domain rdf:resource="**#AppartementCher**"/> <rdfs:range rdf:resource="**#Locataire-Riche**"/> <vpowl:onViewpoint rdf:resource="**#Finance**"/> </vpowl:classOfLocalProperty>
Définit une propriété locale entre les classes locales *AppartementCher* et *Locataire riche* qui sont définies sous le point de vue *Finance*.
Relation de subsomption
<vpowl:LocalClass rdf:ID=" **AppartementCher** "> <rdfs:subClassOf rdf:resource="**#Appartement**"/> </vpowl:LocalClass>
Exprime un lien de subsomption entre la classe locale *AppartementCher*, définie sous le point de vue *Finance*, et la classe globale *Appartement*.
<vpowl:LocalClass rdf:ID="**HLM**"> <rdfs:subClassOf rdf:resource="**#AppartementPasCher**"/> </vpowl:LocalClass>
Exprime un lien de subsomption entre deux classes locales définies sous le même point de vue *Finance*.

Passerelle d'inclusion bidirectionnelle
<vpowl:LocalClass rdf:ID="**HLM**"> <vpowl:EquivalenceBridge rdf:resource="**#AppartementBanlieue** "/> </vpowl:LocalClass>
Exprime une passerelle d'inclusion bidirectionnelle entre les deux classes locales *HLM* et *AppartementBanlieue*.

Passerelle d'inclusion unidirectionnelle
<owl:Class> <owl:intersectionOf rdf:parseType="collection"> <vpowl:LocalClass rdf:about="# PlusTroisPiece"/> <vpowl:LocalClass rdf:about="# AppartementCentreVille"/> </owl:intersectionOf> <vpowl:InclusionBridge rdf:resource="#AppartementCher"/> </owl:Class>
Exprime une passerelle d'inclusion entre la classe source, définie par l'intersection des deux classes locales *PlusTroisPiece* et *AppartementCentreVille*, et la classe cible *AppartementCher*.

Multi-instanciation
<Appartement rdf:about="#ChezBenali"> <vpowl:belongtoViewpoint rdf:resource="#Taille"/> <rdf:type rdf:resource="#Petit-Appartement"/> <vpowl:belongtoViewpoint rdf:resource="#Localisation"/> <rdf:type rdf:resource="#AppartementBanlieue"/> </Appartement >
Affirme que l'individu *ChezBenali*, qui est une instance de la classe globale *Appartement*, est rattaché d'une part à la classe locale *Petit-Appartement*, sous le point de vue *Taille* et d'autre part à la classe *AppartementBanlieue* sous le point de vue *Localisation*.

6 Conclusion

Dans ce chapitre, nous nous sommes intéressés à l'implémentation d'une ontologie multi-points de vue. D'abord nous avons présenté notre proposition du langage Vp-OWL permettant de représenter une ontologie multi-points de vue dans le cadre du web sémantique. Le langage Vp-OWL a été proposé comme une extension de OWL-DL, par l'ajout de nouveaux constructeurs. Par ailleurs, la description de ces nouveaux constructeurs a été fait à travers une implémentation d'une méta-ontologie, que nous avons appelé Onto-Vp-OWL, permettant d'exprimer le vocabulaire du langage Vp-OWL. Enfin, nous avons décrit en détaille les différentes étapes à suivre pour implémenter une ontologie multi-points de vue dans le langage Vp-OWL en utilisant l'outil Protégé-OWL.

Le chapitre suivant conclut notre travail mais permet surtout de dresser des perspectives à la recherche entreprise.

Conclusion générale et perspectives

« Ce n'est pas la fin.
Ce n'est même pas le commencement de la fin.
Mais, c'est peut-être la fin du commencement »
Winston Churchill

Le Web sémantique repose essentiellement sur la notion d'ontologie. Les ontologies sont une manière de représenter de façon formelle la connaissance d'un domaine donné. Un des buts principaux des ontologies est d'être partagées entre un groupe de personnes pour fixer une terminologie et les relations entre concepts, aussi bien pour une utilisation humaine que pour une machine.

Dans des domaines complexes et multi-disciplinaires, co-existent souvent différentes façons d'appréhender les éléments de connaissances, c'est-à-dire différents points de vue (ou perceptions) selon lesquels ces connaissances peuvent être représentées. Ainsi, le même domaine peut avoir plusieurs représentations alternatives, où chacune d'entre elles est décrite selon un point de vue ou une perception particulière.

Dans ce mémoire de thèse, nous avons abordé la problématique de la multi-représentation dans les ontologies. L'objectif visé est de proposer une approche de développement des ontologies qui tiennent compte des différents points de vue des utilisateurs. Ainsi, pour atteindre l'objectif de la thèse qui est de permettre le développement des ontologies multi-points de vue dans le cadre du Web sémantique, notre démarche a été la suivante :

- Au niveau conceptuel, nous avons étudier et proposer un modèle de représentation des connaissances en prenant en compte la notion de point de vue : *modèle multi-points de vue*. Ensuite, au niveau formel, le modèle multi-points de vue à été appliquer pour formaliser l'ontologie multi-points de vue en logique de descriptions. Pour cela, nous avons utilisé un sous-langage de la logique de descriptions pour exprimer les notions inhérentes aux points de vue telles que les concepts et les rôles globaux et locaux, les passerelles, les estampillages, …Une définition formelle de la syntaxe et de la sémantique, en logique de descriptions, des différentes notions introduites pour le support des points de vue a été présentée.
- Au niveau opérationnel et sur la base du formalisme de logique de description développé pour la formalisation de l'ontologie multi-points de vue, un langage d'ontologies multi-

points de vue, nommé Vp-OWL, à été construit. Ce dernier, permet de représenter des ontologies multi-points de vue par un langage qui est une extension du langage d'ontologie OWL, recommandé par W3C. A cet effet, de nouveaux constructeurs ont été introduits pour prendre en compte l'aspect multi-points de vue. Ainsi, nous avons défini une ontologie de représentation (une méta-ontologie), nommée Onto-Vp-OWL permettant de décrire la sémantique des différents constructeurs du langage Vp-OWL selon la syntaxe RDF/XML de OWL.

– Enfin, nous avons proposé la méthode Vp-MethOnto permettant le développement des ontologies multi-points de vue à travers un processus qui comporte quatre principales étapes : une étape de spécification des besoins, une étape de conceptualisation, une étape de formalisation et une étape de codification. Vp-MethOnto utilise l'outil Protégé-OWL comme support technique pour la codification des ontologies en Vp-OWL.

Notre approche pour le développement des ontologies multi-points de vue, a un intérêt flagrant en terme d'ingénierie et de gestion des connaissances. L'approche proposée consiste en effet à construire un ensemble de représentations locales, correspondant chacune à la vision d'une partie du domaine, destinée à une tâche, une application ou un groupe de personnes particulier. Par ailleurs, ces représentations partages à un niveau global des éléments ontologiques et des passerelles permettant d'exprimer un certain consensus global entre les différents points de vue. De cette façon, chaque point de vue peut se concentrer sur une représentation simple, homogène et la plus exhaustive possible des connaissances qu'il juge utiles selon son propre intérêt. Par ailleurs, l'ensemble des représentations locales correspondant aux points de vue présents dans le domaine est plus simple à maintenir qu'une ontologie unique agrégeant l'ensemble de ces points de vue. En effet, une évolution dans les connaissances d'un point de vue particulier peut dans ce cas être réalisée localement, sans avoir à remettre en cause l'ensemble de la représentation. De la même façon, la représentation multi-points de vue simplifie l'utilisation et l'accès aux connaissances. Les acteurs du domaine, selon le point de vue qu'ils adoptent, ont la possibilité de se concentrer uniquement sur les représentations locales qu'ils jugent pertinents et obtenir de cette façon une représentation des connaissances adaptée à leurs usages. Le partage de connaissances avec d'autres points de vue est alors réalisé, à travers la notion de passerelle.

Perspectives

Au terme de cette thèse, nous pouvons envisagé en perspectives un certain nombre d'extension de note travail.

- Intégrer la notion de flou dans la représentation des ontologies multi-points de vue. Cette notion intervient au moment de la liaison des représentations locales des différents points de vue. En effet, il n'est pas toujours possible de spécifier avec certitude et avec précision les liens sémantiques qui existent entre les différentes représentations des différents points de vue. Pour cette raison, nous envisagerons d'intégrer la notion de floue dans notre approche pour représenter ce type d'ontologie. Nous nous basons sur les travaux de la théorie des possibilités et de la logique floue, en particulier les travaux qui ont proposé d'introduire du flou dans les Logique de Descriptions.
- Proposer une approche *d'annotation sémantique* des ressources dans une organisation hétérogène, en prenant en considération un aspect multi-points de vue. Pour répondre à ce requis, l'approche à proposer sera basée sur l'exploitation et l'instanciation d'une ontologie multi-points de vue. Les annotateurs ou les utilisateurs finaux de cette ontologie peuvent faire des énoncés (i.e. des annotations) comme ils veulent et selon leurs niveaux de connaissances et leurs points de vue visé. Ainsi, a chaque point de vue correspond une annotation sémantique locale ayant son propre langage et surtout sa propre interprétation vis-à-vis au contenu et à la fonctionnalité de la ressource considérée.
- Dans le modèle multi-points de vue proposé, nous n'avons pas introduit de relations entre les points de vue. Par exemple, un point de vue ne peut pas être défini comme sous-point de vue d'un autre point de vue. Cette capacité pourrait être affectée en considérant comme qu'un point de vue correspond à un ensemble de critères qui caractérisent le contexte défini par le point de vue, et que l'ajout d'autres critères (caractéristiques) à cet ensemble créera un autre point de vue, qui sera un sous-point de vue du point de vue en question.
- Proposer une approche d'alignement entre des ontologies «classiques » et des ontologies multi-points de vue. Un scénario à considérer est celui d'une organisation possédant déjà une ontologie multi-points de vue, et intégrant, au cours de sa croissance, d'autres communautés ou d'autres groupes d'utilisateurs qui possèdent eux-mêmes leurs propres ontologies. Le problème est donc de mettre en correspondance non seulement entre des ontologies classiques, mais aussi entre une ontologie multi-points de vue et une ontologie classique voire entre des ontologies multi-points de vue.

- Enfin, une autre perspective de ce travail consiste à la réalisation d'un système complet permettant d'une part de gérer des ontologies multi-points de vue par la définition, la modification, la suppression des points de vue, des entités de l'ontologie selon des points de vue ; et d'autre part de filtrer des connaissances selon des points de vue des utilisateurs.

Ces quelques commentaires montrent que le sujet est encore loin d'être épuisé. Nous envisageons donc de continuer à explorer les différentes possibilités encore ouvertes.

ANNEXE

Code OWL de la méta ontologie (Onto-Vp-OWL-Language)

```
<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
   <!ENTITY protege "http://protege.stanford.edu/plugins/owl/protege#" >
   <!ENTITY xsp "http://www.owl-ontologies.com/2005/08/07/xsp.owl#" >]>

<rdf:RDF xmlns="http://www.owl-ontologies.com/Vp-OWL-Language.owl#"
    xml:base="http://www.owl-ontologies.com/Vp-OWL-Language.owl"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:xsp="http://www.owl-ontologies.com/2005/08/07/xsp.owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#">
   <owl:Ontology rdf:about="">
      <rdfs:comment rdf:datatype="&xsd;string">
        Méta ontologie permettant d'exprimer le vocabulaire du langage Vp-OWL</rdfs:comment>
   </owl:Ontology>

   <owl:Class rdf:ID="Viewpoint"/>
   <owl:DatatypeProperty rdf:ID="ViewpointName">
      <rdf:type rdf:resource="&owl;FunctionalProperty"/>
      <rdfs:domain rdf:resource="#Viewpoint"/>
      <rdfs:range rdf:resource="&xsd;string"/>
   </owl:DatatypeProperty>

   <owl:Class rdf:ID="GlobalClass">
      <rdfs:subClassOf rdf:resource="&owl;Class"/>
   </owl:Class>
   <ClassOfbelongtoViewpoint rdf:ID="belongtoViewpoint">
      <rdfs:range rdf:resource="#Viewpoint"/>
   </ClassOfbelongtoViewpoint>

   <owl:Class rdf:ID="ClassOfbelongtoViewpoint">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>

    <owl:Class rdf:ID="ClassOfEquivalenceBridge">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>

   <owl:Class rdf:ID="ClassOfExclusionBridge">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>

   <owl:Class rdf:ID="ClassOfGlobalProperty">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>
   <owl:Class rdf:ID="ClassOfInclusionBridge">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>

   <owl:Class rdf:ID="ClassOfLocalProperty">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
   </owl:Class>
```

```
    <owl:Class rdf:ID="ClassOfonViewpoint">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
    </owl:Class>

    <owl:Class rdf:ID="ClassOfunderViewpoint">
      <rdfs:subClassOf rdf:resource="&owl;ObjectProperty"/>
    </owl:Class>

    <ClassOfEquivalenceBridge rdf:ID="EquivalenceBridge">
      <rdf:type rdf:resource="&owl;SymmetricProperty"/>
      <rdf:type rdf:resource="&owl;TransitiveProperty"/>
      <rdfs:domain rdf:resource="#LocalClass"/>
      <owl:equivalentProperty rdf:resource="&owl;equivalentClass"/>
      <owl:inverseOf rdf:resource="#EquivalenceBridge"/>
      <rdfs:range rdf:resource="#LocalClass"/>
    </ClassOfEquivalenceBridge>

    <ClassOfExclusionBridge rdf:ID="ExclusionBridge">
      <rdf:type rdf:resource="&owl;SymmetricProperty"/>
      <rdfs:domain rdf:resource="#LocalClass"/>
      <owl:equivalentProperty rdf:resource="&owl;disjointWith"/>
      <owl:inverseOf rdf:resource="#ExclusionBridge"/>
      <rdfs:range rdf:resource="#LocalClass"/>
    </ClassOfExclusionBridge>

<ClassOfInclusionBridge rdf:ID="InclusionBridge">
      <rdfs:domain rdf:resource="#LocalClass"/>
      <owl:equivalentProperty rdf:resource="&rdfs;subClassOf"/>
      <rdfs:range rdf:resource="#LocalClass"/>
    </ClassOfInclusionBridge>
    <owl:Class rdf:ID="LocalClass">
      <rdfs:subClassOf rdf:resource="&owl;Class"/>
    </owl:Class>

<ClassOfGlobalProperty rdf:ID="GlobalProperty">
      <rdfs:domain rdf:resource="#LocalClass"/>
      <rdfs:range rdf:resource="#LocalClass"/>
    </ClassOfGlobalProperty>

    <ClassOfLocalProperty rdf:ID="LocalProperty">
      <rdfs:domain rdf:resource="#LocalClass"/>
      <rdfs:range rdf:resource="#LocalClass"/>
    </ClassOfLocalProperty>
    <ClassOfonViewpoint rdf:ID="onViewpoint">
      <rdfs:domain>
        <owl:Class>
          <owl:unionOf rdf:parseType="Collection">
            <owl:Class rdf:about="&owl;DatatypeProperty"/>
            <owl:Class rdf:about="#ClassOfLocalProperty"/>
          </owl:unionOf>
        </owl:Class>
      </rdfs:domain>
      <rdfs:range rdf:resource="#Viewpoint"/>
   </ClassOfonViewpoint>

    <ClassOfunderViewpoint rdf:ID="underViewpoint">
      <rdfs:domain rdf:resource="#LocalClass"/>
      <rdfs:range rdf:resource="#Viewpoint"/>
    </ClassOfunderViewpoint>

</rdf:RDF>
```

Références Bibliographiques

[Amardeilh, 2007] Amardeilh., F. (2007). "Web Sémantique et Informatique Linguistique : propositions méthodologiques et réalisation d'une plateforme logicielle". Thèse de doctorat, Université Paris X- Nanterre.

[**Aussenac-Gilles et al., 2000]** Aussenac-Gilles., N., B. Biébow., B. et Szulman., S. (2000) Revisiting ontology design: a methodology based on corpus analysis. In *Proc. of the 12th International Conference, (EKAW'2000)*, LNAI 1937, pp. 172-188.

[Baader et Hollunder, 1991] Baader., F. et Hollunder., B. (1991). "KRIS: Knowledge Representation and Inference System". SIGART Bulletin, Vol.2, N°.3, pp.8–15.

[Baader et al., 2003] Baader, F., Horrocks, I., et Sattler, U. (2003)." Description logics as ontology languages for the semantic web". In D. Hutter and W. Stephan, editors, Festschrift in honor of Jorg Siekmann. LNAI. Springer-Verlag, pp. 228-248.

[Baader et al., 2007] Baader, F., Horrocks, I., et Sattler, U. (2007)." Description Logics". Frank van Harmelen, Vladimir Lifschitz, et Bruce Porter, editors, Handbook of Knowledge Representation, pp. 135–179. Elsevier.

[Baader., 2009] Baader., F (2009). "Description Logics". In Reasoning Web: Semantic Technologies for Information Systems, 5th International Summer School 2009, Vol.5689 of Lecture Notes in Computer Science, pp. 1–39.

[Baader et al., 2011] Baader., F., Binh., N., Borgwardt., S et Morawska., B. (2011). "Unification in the Description Logic EL without the Top Concept". In Proc of the 24th International Workshop on Description Logics, Vol. 745 of CEUR-WS, pp. 26–36.

[Bach, 2006] Bach., T. L. (2006). "Construction d'un Web sémantique multi-points de vue ". Thèse de doctorat en sciences, École des Mines de Paris, Sophia Antipolis.

[Bachimont et al., 2002] Bachimont B., Isaac., A. et Troncy., R. (2002). Semantic commitment for designing ontologies : A proposal. In 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW), volume LNAI 2473 of Lecture Notes in Artificial Intelligence, p. 114–121

[Baget et al., 2004] Baget., J.-F., Canaud., E., Euzenat., J. et Hacid., M.-S. (2004). "Les langages du Web Sémantique", In Le Web sémantique, CHARLET J., LAUBLET P. & REYNAUD C. (Ed.), Revue Information - Interaction - Intelligence, Vol.4, N°.1, pp. 21-43.

[Balley et al., 2004] Balley, S., Parent, C. & Spaccapieta, S. (2004). "Modeling Geographic Data with Multiple Representation" In journal Geographical Information science, Vol 18, p. 327-352.

[Benchikha et al. 2005] Benchikha, F., Boufaïda, M. et Seinturier, L. (2005). "Viewpoints: a framework for object oriented database modeling and distribution". Data Science Journal, Vol. 4, pp. 92-107.

[Benchikha, 2007] Benchikha, F. (2007). " Intégration des points de vue dans les bases de données à objets : Le modèle Multi-Viewpoint DataBase", Thèse de Doctorat, Université de Constantine.

[Benerecetti et al., 2001] Benerecetti., M., Bouquet., P. et Ghidini., C. (2001). "On the Dimensions of Context Dependence: Partiality, Approximation, and Perspective". In Proc. of CONTEXT 2001, pp.59–72

[Benslimane et al., 2006] Benslimane, D., Arara, A., Falquet, G., Maamar, Z., Thiran, P. and Gargouri, F. (2006) "Contextual Ontologies: Motivations, Challenges, and Solutions", i*n Fourth Biennial International Conference on Advances in Information Systems (ADVIS-2006),* pp.168–176, Springer Verlag.

[Berners-Lee et al., 2001] Berners-Lee., T., Hendler., J., et Lassila., O. (2001). "The semantic web". Scientific American, Vol.284, N°.5, pp.34–43.

[Bobrow et Winograd, 1977] Bobrow., D. G. et Winograd., T. (1977). "An Overview of KRL, a Knowledge Representation Language". Cognitive Science, Vol. 1, pp. 3–46.

[Borgida et Serafini, 2003] Borgida., A. et Serafini. L. (2003). "Distributed description logics: Assimilating information from peer sources". Journal of Data Semantics, Vol. 1, pp.153–184.

[Bouquet et al., 2003] Bouquet, P., Ghidini, C., Giunchiglia, F., Blanzieri, E. (2003). "Theories and uses of context in knowledge representation and reasoning". Journal of Pragmatics, special issue on Context, Vol.35, N°.3, pp. 455–484.

[Bouquet et al., 2004] Bouquet., P., Giunchiglia., F., van Harmelen., F., Serafini., L. et Stuckenschmidt., H. (2004). "Contextualizing Ontologies". Journal of Web Semantics, Vol.1, N°4, pp. 325–343.

[Brachman et Schmolze, 1985] Brachman., R. J. et Schmolze., J. G. (1985). "An overview of the KL-ONE knowledge representation system". Cognitive Science, Vol.9, N°.2, pp.171–216.

[Brachman et Levesque 1985] Brachman., R. et Levesque., H. Eds., "Readings in Knowledge Representation, Morgan Kaufmann Publishers", Los Altos (CA), USA.

[Bray *et al.*, 2006] Bray., T., Paoli., J., Sperberg-McQueen., C. M., et Cowan., John. (2006). " Extensible Markup Language (XML) 1.1" (Second Edition). Technical report, 2006. W3C Recommendation http://www.w3.org/TR/xml11/.

[Brickley et Guha, 2004] Brickley. D. et Guha., R. V. (2004). "RDF VocabularyDescription Language 1.0: RDF Schema". Technical report. W3C Recommendation. http://www.w3.org/TR/rdf-schema/.

[Brisson, 2004] Brisson, L. (2004). "Mesures d'intérêt subjectif et représentation des connaissances". Rapport de recherche, ISRN I3S/RR–2004-35–FR, Projet EXECO.

[Chabalier, 2004] Chabalier, J. (2004). "Acquisition incrémentale et représentation des systèmes intégrés bactériens par une approche orientée-objet ". Thèse de doctorat, Université de Provence, Aix-Marseille I.

[Connolly et al., 2001] Connolly., D., van Harmelen., F., Horrocks., I., McGuinness., D. L., Patel-Schneider., P. F., et Stein., L. A. (2001). "DAML+OIL Reference Description. World Wide Web Consortium". http://www.w3.org/TR/daml+oil-reference.

[Corcho et al., 2005] Corcho., O., Fernández., M., Gómez-Pérez., A., et López-Cima., A. (2005). "Building Legal Ontologies with METHONTOLOGY and WebODE". In : Law and the Semantic Web Heidelberg, DE: Springer (2005), p. 142-157.

[d'Aquin et al., 2004] d'Aquin., M., Lieber., J., et Napoli., A. (2004). "Représentation de points de vue pour le raisonnement à partir de cas". In Langages et Modèles à objets (LMO'04), pp. 245-258.

[d'Aquin et al., 2005] d'Aquin., M., Lieber., J., et Napoli., A. (2005). "Decentralized Case-Based Reasoning for the Semantic Web". In Proceedings of the 4th International Semantic Web Conference (ISWC'05), Springer.

[d'Aquin, 2005] d'Aquin., M. (2005). "Un portail sémantique pour la gestion des connaissances en cancérologie". Thèse de Doctorat, Université Henri Poincaré- Nancy 1.

[Dean et al., 2004] Dean., M., Connolly., D., Harmelen., F., Hendler., J., Patel-Schneider., P. F. et Stein., L. A. (2004). "OWL Web Ontology Language Reference". Technical report, W3C Recommendation. http://www.w3.org/TR/owl-ref/.

[Després et al, 2006] Després., S. et Szulman., S. (2006). "Terminae method and integration process for legal ontology building". In Proceedings of the 19th International Conference IAE/AIE. pp. 1014-1023.

[Dieng et al., 2000] Dieng., R., Corby., O., Grandon., F., Giboin., A., Golebiowska., J., Matta., N. et Ribière., M. (2000). "Méthodes et outils pour la gestion des connaissances, chapitre 6: Gestion de multiples points de vue". Dunod, Paris.

[Dieng. et al, 2001] Dieng., R., Corby., O., Gandon., F., Giboin., A. et Golebiowska., J. (2001). "Méthodes et outils pour la gestion des connaissances : une approche pluridisciplinaire du knowledge management", (2ième édition), Dunod Edition Informatiques Séries Systèmes d'Information.

[Dieng. et al, 2004] Dieng., R., Corby., O., Gandon., F., Giboin., A., Golebiowska., J., Matta., N. et Ribiere., M. (2004). "Méthodes et outils pour la gestion des connaissances : une approche pluridisciplinaire du knowledge management". Dunod Edition, Informatique, Séries Systèmes d'Information.

[Dieng et al., 2005] Dieng., K. R., Corby., O., Gandon., F., Giboin., A., Golebiowska., J., Matta., N. et Ribière, M. (2005). "Knowledge Management-Methodes et outils pour la gestion des connaissances; une approche pluridisciplinaire du Knowledge Management". 2e dition, Dunod.

[Dieng, 2005] Dieng., K. R. (2005). "Corporate Semantic Webs". In Encyclopaedia of Knowledge Management, D. Schwartz ed, Idea Publishing Group.

[Dietz, 2006] Dietz, Jan L.G. (2006). "Entreprise Ontology: Theory and Methodology". Computer Science. Berlin Heidelberg. Springer-Verlag.

[Distel, 2010] Distel., F. (2010). "An Approach to Exploring Description Logic Knowledge Bases". In Proc of the 8th International Conference on Formal Concept Analysis, Vol. 5986 of Lecture Notes in Artificial Intelligence, pp. 209–224.

[Djakhdjakha et al., 2009] Djakhdjakha., L., Hemam., M. et Boufaida., Z. (2009). "Une Approche d'Alignement d'une Ontologie Multi-Points de Vue et une Ontologie Classique". In the International Conference on Applied Informatics (ICAI'09), pp. 390–396.

[Djakhdjakha et Hemam, 2010] Djakhdjakha., L. et Hemam., M. (2010). "Optimisation d'alignement d'une ontologie multi-points de vue et une ontologie classique". In Proceedings of COSI'10 (Colloque International sur l'Optimisation et les Systèmes d'Information), pp. 332–343.

[Djellal et al., 2010] Djellal, A., Hemam., M. et Boufaida., Z. (2009). "La notion de flou dans les ontologies multi-points de vue", In the International Conference on Applied Informatics (ICAI'09) pp. 170–174.

[Djellal et al., 2010] Djellal, A., Hemam., M. et Boufaida., Z. (2010). "An Extension of the Ontology Web Language with Viewpoint and Fuzzy Notions", In Proceedings of MISC'2010, International Symposium on Modelling and Implementation of Complex Systems, pp. 149–159.

[Euzenat, 1998] Euzenat., J. (1998). "Représentation de connaissances par objets", R. Ducournau, J. Euzenat, G. Masini, A. Napoli (eds), Langages et modèles à objets — Etat des recherches et perspectives, Collection Didactique D-019, INRIA, Le Chesnay, pp. 293-319.

[Falquet et al., 2001] Falquet, G. Mottaz, C.L. (2001). "Navigation hypertexte dans une ontologie multi-points de vue". in Actes de la conférence NîmesTIC'2001.

[Farquhar et al., 1997] Farquhar., A., Fikes., R. et Rice., J. (1997). "The Ontolingua Server: a Tool for Collaborative Ontology Construction". In. International Journal of Human-Computer Studies, Vol. 466, pp. 707-727.

[Fensel et al., 2001] Fensel, D., van Harmelen, F., Horrocks, I., McGuinness, D. L., et Patel-Schneider, P. F. (2001). "OIL: An Ontology Infrastructure for the SemanticWeb". IEEE Intelligent Systems, Vol.16, N°.2, pp.38–45.

[Fernandez et al., 1997] Fernandez, M., Asuncion Gomez-Perez, A. et Juristo, N. (1997). "METHONTOLOGY: from ontological art towards ontological engineering". In Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering, pp. 33–40.

[Finkelstein et Sommerville, 1996] Finkelstein., A. et Sommerville., I. (1996). "The Viewpoints FAQ". Software Engineering Journal, Vol. 11, N°1, pp. 2-4.

[Furst, 2002] Furst, F. (2002). "L'ingénierie ontologique". Rapport technique, Institut de recherche en Informatique de Nantes.

[Furst, 2004] Furst., F. (2004). "Contribution à l'ingénierie des ontologies : une méthode et un outil d'opérationnalisation". Thèse d'Informatique, Université de Nantes.

[Gandon, 2006] Gandon., F. (2006). "Ontologies informatiques". Interstices, Journal en ligne de l'INRIA. Disponible à http://interstices.info/display.jsp?id=c_17672

[Gandon et Dieng-Kuntz, 2001] Gandon., F. et Dieng-Kuntz., R. (2001). "Ontologie pour un système multi-agents dédié à une mémoire d'entreprise", In Actes des journées francophones d'Ingénierie des Connaissances IC'2001, pp.1-20, Presses Universitaires de Grenoble.

[Gangemi et al., 2003] Gangemi., A., Guarino., N., Masolo., C. et Oltramari., A. (2003). "Sweetening WORDNET with DOLCE". AI Magazine, Vol. 24, N°3, pp.13–24.

[Ghidini et Serafini, 2006] Ghidini., C., et Serafini., L. (2006). "Reconciling Concepts and Relations in Heterogeneous Ontologies". In The Semantic Web : Research and Applications, 3rd European SemanticWeb Conference, ESWC 2006, Proceedings, volume 4011 de Lecture Notes in Computer Science, pp.50–64. Springer-Verlag.

[Ghidini et al., 2007] Ghidini., C., Serafini., L., et Tessaris., S. (2007). "On Relating Heterogeneous Elements from Different Ontologies. In Modeling and Using Context". In Proceedings of the 6th International and Interdisciplinary Conference, CONTEXT 2007, pp.234–247. Springer-Verlag.

[Gruber, 1995] Gruber., T.R. (1995). "Toward principles for the design of ontologies used for knowledge sharing". International Journal of Human Computer Studies.

[Guarino et Giaretta, 1995] Guarino., N. et Giaretta., P. (1995). "Ontologies and knowledge bases: Towards a terminological clarification". In N MARS, réd., Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing, pp. 25–32. IOS Press.

[Haarslev et Müller, 2001] Haarslev., V. et Müller., R. (2001). "Racer system description". Dans Proceedings of IJCAR'2001, Vol.2083 of Lecture Notes in Artificial Intelligence (LNAI), pp. 701–705. Springer.

[Hemam et Boufaida, 2004] Hemam., M. et Boufaida., Z. (2004). "An Ontology Development Process for the Semantic Web". Workshop on Knowledge Management and the Semantic Web (EKAW'04), UK.

[Hemam, 05] Hemam., M. (2005). "Un processus de développement d'ontologie dans le cadre du Web sémantique". Mémoire de Magister en informatique, universitaire Larbi Ben M'hidi-Oum El Bouaghi, institut des sciences exactes.

[Hemam et Boufaida, 2008a] Hemam., M. et Boufaida., Z. (2008). "Conceptualisation d'ontologies multi-points de vue ", 6ème Séminaire National en Informatique (SNIB'08), pp.5–10.

[Hemam et Boufaida, 2008b] Hemam., M. et Boufaida., Z. (2008). "Prise en compte des points de vue dans la construction des ontologies en logique de descriptions". Actes du Colloque International sur l'Optimisation et les Systèmes d'Information (COSI'08), pp.287–298.

[Hemam et Boufaida, 2009a] Hemam., M. et Boufaida., Z. (2008). "Représentation d'ontologies multi-points de vue: une approche basée sur la logique de descriptions". Papier court dans 20es Journées Francophones d'Ingénierie des Connaissances (IC'09).
http://ic2009.inria.fr/docs/posters/HemamBoufaida_Poster_IC2009.pdf.

[Hemam et Boufaida, 2009b] Hemam., M. et Boufaida., Z. (2009). "Raisonnement par classification sur une ontologie multi-points de vue". In 3èmes Journée Francophone sur les Ontologies (JFO'09), ACM Edition, pp.149–156.

[Hemam et Boufaida, 2010] Hemam., M. et Boufaida., Z. (2010). "MVp-OWL: A Multi-Viewpoints Ontology Language for the Semantic Web". In Proceedings of the Third International Conference on Web and Information Technologies (ICWIT'10), pp.120–131.

[Hemam et al., 2010] Hemam., M., Djema., O. et Boufaida., Z. (2010). "Vers une approche d'annotation sémantique multi-points de vue". Actes des Journées d'Etudes Doctorales (JED'10), pp.13–18.

[Hemam et Boufaida, 2011] Hemam., M. et Boufaida., Z. (2011). "MVp-OWL: A Multi-Viewpoints Ontology Language for the Semantic Web". Int. Journal Reasoning-based Intelligent Systems (IJRIS). Inderscience Publishers**,** Vol. 3, N° 3/4, pp. 147–155.

[Héon, 2010] Héon., M. (2010). "OntoCASE: Méthodologie et assistant logiciel pour une ingénierie ontologique fondée sur la transformation d'un modèle semi-formel". Thèse de doctorat, Université du Québec, Montréal

[Homola et al., 2010] Homola, M., Serafini, L., & Tamilin, A. (2010). "Modeling Contextualized Knowledge". In Proc. of International Workshop on Context, Information And Ontologies (CIAO'10).

[Horrocks, 1998] Horrocks., I. (1998). "Using an Expressive Description Logic: FaCT or Fiction ?". Dans Proceedings of KR'98, pp.636–649.

[Horrocks et al., 2003] Horrocks., I., Patel-Schneider., P.F. et van Harmelen., F. (2003). "From SHIQ and RDF to OWL : the making of a Web Ontology Language". Journal of Web Semantics, Vol.1, N°1, pp.7–26.

[Horrocks et Patel-Schneider, 2004] Horrocks, I., et Patel-Schneider, F. P. (2004). "Reducing OWL entailment to description logic satisfiability". In Proceedings of the second International Semantic Web Conference ISWC'04.

[Horrocks et al., 2005] Horrocks., I., Baader., F. et SattlerL., U. (2003). "Description Logics as Ontology Languages for the Semantic Web". In Festschrift in hoor of Jorg Siekmann, LNAI. Vol. 2605, Springer-Verlag.

[Horrocks et Sattler, 2007] Horrocks., I et Sattler., U. (2007). "A Tableaux Decision Procedure for SHOIQ". *Journal of Automated Reasoning,* Springer Verlag, Vol.39, N°3, pp.245-429.

[Joseph et al., 2009] Joseph, M., Serafini, L. et Tamilin, A. (2009). "Context shifting for effective search over large knowledge bases. In: Workshop on Context, Information And Ontologies (CIAO-2009).

[Keita, 2007] Keita., A. K. (2007). "Conception coopérative d'ontologies pré-consensuelles: Application au domaine de l'urbanisme". Thèse de doctorat, Institut National des Sciences Appliquées de Lyon.

[Kifer et al., 1995] Kifer., M., Lausen., G. et Wu., J.(1995). "Logical Foundations of Object-Oriented and Frame-Based Languages". Journal of the ACM, Vol.42, N°.4, pp. 741–843.

[Klyne et Carroll, 2004] Klyne., G. et Carroll., J. J. (2004). "Resource Description Framework (RDF): Concepts and Abstract Syntax". Technical report. W3C Recommendation. http://www.w3.org/TR/rdf-concepts/.

[Konev et al., 2008] Konev., B.,Lutz., C., Walther., D. et Wolter., F. "Semantic Modularity and Module Extraction in Description Logics". In Proceedings of the 18th European Conference on Artificial Intelligence PP. 55–59. IOS Press.

[Lacot, 2005] Lacot., X. (2005). "Introduction à OWL, un langage XML d'ontologies web". http://lacot.org/public/introduction_a_owl.pdf.

[Lahna et al., 2009] Lahna., B., Roudiès., O. et Giraudin., J.P. (2009). "Approches par points de vue pour l'ingénierie des Systèmes d'information". e-TI - la revue électronique des technologies d'information, N°5. http://www.revue-eti.netdocument.php?id=1949.

[Leclère et al, 2002a] Leclère., L., Trichet., T. et Frûst., F. (2002). "Construction of an ontology related to the projective geometry". RFIA 13th congrès des Reconnaissance des Frames et Intelligence Artificielle, France.

[Leclère et al, 2002b] Leclère., L., Trichet., T. et Frûst., F. (2002). "Operationalising domain ontologies : towards an ontological level for the SG family", in Foundations and Applications of Conceptual Structure, contributions to the International Conference on Conceptual Structures. 2002.

[Liao et Tu, 2007] Liao, H.C. et Tu, C.C. (2007). "A rdf and owl-based temporal context reasoning model for smart home". Information Technology Journal Vol.6, pp.1130-1138.

[Lopez et al, 1999] Lopez., M. F., Gomez-Perez., A., Sierra., J. P. & Sierra., A. P. (1999). "Building a chemical ontology using methontology and the ontology design environment". IEEE Intelligent Systems, Vol.14, N°.1, pp.37–46.

[Luong, 2007] Luong, P. (2007). " Gestion de l'évolution d'un Web sémantique d'entreprise". Thèse de doctorat de Mines de Paris.

[Mariño, 1993] Mariño, O. (1993). *Raisonnement classificatoire dans une représentation à objets multi-points de vue*. Thèse de l'université Joseph Fourier, Grenoble 1.

[Masini et al., 1989] Masini., G. Napoli., A. Colnet., D. Léonard., D. et Tombre., K. (1989). "Les langages à objets". InterEditions, Paris

[Mellal, 2007] Mellal, N. (2007). "Réalisation de l'interopérabilité sémantique des systèmes, basée sur les ontologies et les flux d'information". Thèse de doctorat, Université Polytech'Savoie.

[Minsky, 1975] Minsky., M. (1975). "A Framework for Representing Knowledge". in The Psychology of Computer Vision New York:6,156-189. P.H. Winston, McGrawHill.

[Napoli, 97] Napoli., A. (1997). "Une introduction aux logiques de descriptions". Rapport de recherche N° 3314.

[Napoli et al., 2000] Napoli., A., Euzenat., J., et Ducournau., R. (2000). "Les représentations des connaissances par objets". Technique et science informatiques, Vol.19, N° 3, pp.387–394.

[Napoli et al., 2004] Napoli., A., Carré., B., Ducournau., R., Euzenat., J. et Rechenmann., F. (2004). "Objets et représentation, un couple en devenir". In Revue des Sciences et Technologies de l'Information. Vol.10, N°4, pp.61-81.

[Niles et Pease, 2001] Niles., I. et Pease., A. (2001). "Towards a standard upper ontology". In Proceedings of the 2nd International Conference on Formal Ontology in Information Systems pp. 2–9.

[Noy et McGuinness, 2001] Noy., N. F. et McGuinness., D. L. (2001). "Ontology Development 101: A Guide to Creating Your. First Ontology". Technical Report KSL-01-05Stanford: Knowledge Systems Laboratory, March, 200.1

[Noy et Fergerson, 2001] Noy., N. F. et Fergerson., R. W. (2001). "The knowledge model of Protege-2000: Combining interoperability and flexibility". In the 12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00), Lecture Notes in Artificial Intelligence LNAI, pp. 17-32.

[Noy et Musen , 2003] Noy., N. F. et Musen., M. A. (2003). "The PROMPT suite : interactive tools for ontology merging and mappin". International Journal of Human-Computer-Studies, Vol.59, N°.6.

[Oberle, 2006] Oberle., D. (2006). "Semantic Management of Middleware ". Semantic Web And Beyond Computing for Human Experience: Springer US.

[Patel-Schneider et al., 2004] Patel-Schneider., P. F., Hayes., P. et Horrocks., I. (2004). "OWL Web Ontology Language; Semantics and Abstract Syntax". Technical report. W3C Recommendation. http://www.w3.org/TR/owl-semantics/.

[Paquette, 2007] Paquette., G. (2007). "Graphical Ontology Modeling Language for Learning Environments". Technology, Instruction., Cognition and Learning, Vol. 5, pp. 133– 168.

[Pierra, 2003] Pierra., G. (2003). "Context-Explication in Conceptual Ontologies: The PLIB Approach", Proceedings of CE'2003, Special track on Data Integration in Engineering, Madeira, Portugal, pp. 243-254.

[Raggett et al., 1999] Raggett, D., Hors, A. L., et Jacobs, I. (1999). "HTML 4.01 Specification". Technical report. W3C Recommendation. http://www.w3.org/TR/html401/.

[Raimbault, 2008] Raimbault., T. (2008). "Transition de modèles de connaissances. Un système de connaissance fondé sur OWL, Graphes Conceptuels et UML". Thèse de doctorat, Université de Nantes.

[Reiter, 1980] Reiter., R. (1980). "A logic for default reasoning". Artificial Intelligence, Vol.13, pp.81–132.

[Ribière, 1998] Ribière., M. (1998). "Using Viewpoints and CG for the Representation and Management of a corporate memory in Concurrent Engineering". Proc. of the 6th Int. Conference on Conceptual Structures (ICCS'98), Montpellier, Springer-Verlag, LNAI 1453.

[Ribière, 1999] Ribière., M. (1999). "Représentation et gestion de multiples points de vue dans le formalisme des graphes conceptuels". Thèse de doctorat en informatique, Nice-Sophia Antipolis.

[Ribière, 2002] Ribière, M., Dieng, R. (2002). "A Viewpoint Model for Cooperative Building of an Ontology". In Proc. of the 10th International Conference in Conceptual Structures, (Borovetz, Bulgarie. Berlin : Springer-Verlag, 2002, pp. 220-234.

[Roche, 2005] Roche., C. (2005). " Terminologie et Ontologie". Revue Langages, N° 157, pp. 48– 62, Editions Larousse.

[Staab et Studer, 2004] Staab., S. et Studer., R. (2004). Handbook on Ontologies. Springer, 2004.

[Sertkaya, 2009] Sertkaya., B. (2009). "OntoComP: A Protege Plugin for Completing OWL Ontologies". In Proc of the 6th European Semantic Web Conference, Vol. 5554 of Lecture Notes in Computer Science, PP. 898–902.

[Stefik et Bobrow, 1985] Stefik., M. Bobrow., D.G. (1985). Object-Oriented programming : Themes and variations. in *A.I. magazine*, Vol.6, N°4, pp-40-62.

[Stein et al., 2000] Stein., L. A., Connolly., D., et McGuinness., D. L. (2000). "DAML-ONT Initial Release". http://www.daml.org/2000/10/daml-ont.html.

[Stumme et Maedche, 2001] Stumme., G. et Maedche., A. (2001). "FCA-MERGE: Bottom-Up Merging of Ontologies". In 17th International Joint Conferences on Artificial Intelligence, pp.225-230.

[Sowa, 2000] Sowa., j. (2000). "Knowledge Representation: Logical, Philosophical, and Computational Foundations". Pacific Grove, CA: Brooks/Cole.

[Sure et al., 2006] Sure., Y., Tempich., C et Vrandecic., D. (2006). "Ontology Engineering Methodologies". In Semantic Web Technologies: Trends and Research in Ontology-based Systems, pp. 171-187.

[Sure et al., 2002] Sure, Y., Juergen Angele., J. et Steffen Staab., S. (2002). "OntoEdit: Guiding Ontology Development by Methodology and Inferencing". In Proceedings of the International Semantic Web Conference (ISWC 2002), Springer, pp. 1205-1222.

[Swartout et al., 1997] Swartout, B., Ramesh, P., Knight, K. et Russ, T. (1997). "Towards Distributed Use of Large-Scale Ontologies". Ontological Engineering. AAAI-97 Spring Symposium Series, pp. 138-148.

[Turhan, 2011] Turhan., A. Y. (2011). "Description Logic reasoning for Semantic web ontologies". Proc. of the first International Conference on Web Intelligence, Mining and Semantics. ACM edition.

[Uschold et Grüninger, 1996] Uschold., M. et Grüninger., M. (1996). "Ontologies: principles, methods, and applications". Knowledge Engineering Review, Vol.11, N°2, pp. 93–155.

[Valery, 2004] Valery., P. (2004). "Proposition d'une méthode d'ingénierie ontologique pour les EIAH : application aux systèmes auteurs". Programme de doctorat en informatique, Mai 2004 Université du Québec à Montréal Canada.