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 d'Informatique

Année : 2008/2009
N° d'ordre :
Série :

Mémoire de Magister

Discipline
Informatique

Option
Systèmes distribués

Présenté par

M^elle^ Souheila KHALFI

Thème

****

CONSTRUCTION D'UNE ONTOLOGIE POUR LA PRISE EN CHARGE DES PATIENTS À DOMICILE

****

Soutenu publiquement le 28/10/2009 devant le jury composé de :

Président:	M^me^ Zizette Boufaida	Professeur	Université Mentouri de Constantine
Rapporteur:	M^r^ Nacereddine Zarour	Maitre de conférences	Université Mentouri de Constantine
Examinateurs:	M^me^ Faiza Belala	Maitre de conférences	Université Mentouri de Constantine
	M^me^ Nadia Zeghib	Maitre de conférences	Université Mentouri de Constantine

---

Préparé au sein du SIBC, Laboratoire LIRE
Equipe Systèmes d'Information & Bases de Connaissances, Laboratoire d'Informatique REpartie

À ma chère mère.

REMERCIEMENTS

Je voudrais avant tout remercier DIEU, le tout puissant, pour tous ses bienfaits trop souvent négligés.

Je remercie mon encadreur le Dr. Nacereddine ZAROUR pour l'aide et les conseils si précieux qu'il m'a prodigués pour l'élaboration de ce mémoire. Je tiens à lui exprimer ma profonde gratitude pour le respect et la sympathie dont je fus l'objet. Vous demeurerez pour moi un exemple à suivre pour votre savoir, votre compétence, vos qualités scientifiques et humaines.

J'ai une dette de reconnaissance envers le Pr. Zizette Boufaida d'avoir présidé le jury.

Je dois également de chaleureux et sincères remerciements aux Drs. Faiza Belala et Nadia Zeghib d'avoir accepté d'être examinatrices de ce modeste mémoire.

J'adresse également mes remerciements à tous ceux qui ont cette pénible tâche de soulager les gens et diminuer leur souffrance.

Je remercie aussi ma famille pour son aide, sa générosité et son soutien moral qui ont été pour moi une source de courage et de confiance.

Merci Saloua pour les moments qu'on a passés ensemble, pour ton soutien et ta disponibilité.

Enfin, je remercie tous mes enseignants du début à la fin de mes études.

ملخص

حاليا, مجال التكفل المنزلي بالمريض عرف تطورا كبيرا بفضل استعمال تقنيات المعلومات و الاتصال الجديدة. هذا النوع من العناية يحتاج إلى تنظيم متعدد التخصصات والذي يضم مجموعة من المساهمين بقرب المريض. هؤلاء يستعملون لغة طبية تتميز بمفردات غنية وصعبة الاستعمال.الحصول على تبادل فعال و خال من التناقضات للمعلومات بين مختلف أخصائي الصحة يتطلب توحيد اللغة المستعملة بينهم. في هذا المعنى, الانطولوجيا تبدو الحل المناسب بكل ما تحمله من ايجابيات.
الهدف من هذا العمل هو بناء انطولوجيا ميدان التكفل المنزلي بالمرضى. المنهجية المتبعة تمت على ثلاث مراحل أساسية:1) وضع نموذج تصوري للانطولوجيا و الذي اعتمدنا خلاله على طريقة *METHONTOLOGY* 2) صياغة الانطولوجيا المحصل عليها بلغة *SHOIN(D)* 3) تحويل هذه الأخيرة إلى انطولوجيا عملية متناسقة و متماسكة.

كلمات مفتاحية: التكفل المنزلي بالمريض, انطولوجيا ,*SHOIN(D)*, OWL.

RÉSUMÉ

De nos jours, le domaine de la Prise en charge des patients À Domicile (PAD) a grandement progressé grâce à l'utilisation des nouvelles Technologies de l'Information et de la Communication (TIC). Un tel mode de prise en charge nécessite une organisation pluridisciplinaire où plusieurs acteurs interviennent auprès du patient. Ces derniers utilisent un langage médical caractérisé par un vocabulaire extrêmement riche et difficile à manipuler. Une communication efficace et dépourvue d'ambiguïté d'informations médicales entre les différents professionnels de santé nécessite un minimum de standardisation du langage utilisé. A ce titre, les ontologies se révèlent la solution informatique appropriée dont les bénéfices sont incontestables.

L'objectif de ce mémoire consiste à la construction d'une ontologie du domaine de la PAD, dont le processus de construction s'est réalisé en trois phases fondamentales : i) la conceptualisation essentiellement fondée sur la méthode *METHONTOLOGY* qui a donné lieu à une ontologie conceptuelle ; ii) la formalisation en logique de description *SHOIN(D)* de ce cette dernière ; iii) l'opérationnalisation qui a aboutit à une ontologie opérationnelle cohérente et consistante.

Mots-clés : Prise en charge des patients à domicile, Ontologie, *SHOIN(D)*, OWL.

ABSTRACT

Nowadays, the field of Home Care (HC) has widely progressed thanks to the use of new Technologies of Information and Communication (TIC). Such kind of care needs a multidisciplinary organization, where several actors are involved. Those one use a medical language characterized by an extremely rich vocabulary and difficult one to manipulate. Efficient and unambiguous communication of medical information between different health care professionals needs a minimal standardization of used language. In this context, the ontologies seem to be the appropriate solution whose benefits are indisputable.

The objective of this thesis consist on the construction of a home care domain ontology, where the development process is realized in three fundamental phases: i) the conceptualization essentially founded on *METHONTOLOGY* methodology that lead to a conceptual ontology; ii) the formalization of that one in description logic *SHOIN(D)*; iii) the implementation witch resulted in a coherent and consistent operational ontology.

Key-words: Home Care, Ontology, *SHOIN(D)*, OWL.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ACRONYMES

ABAH	Agent Based Architecture For Homecare
AL	Attributive Language
ARCHONTE	ARCHitecture for ONTological Elaborating
CIM	Classification statistique Internationale des Maladies et des problèmes de santé connexes
COQUAS	COordination pour la QUAlité de Soins
CRM	Common Reference Model
DAML	DARPA Agent Markup Language
DL	Description Logic
FaCT	Fast Classification of Terminologies
FLogic	Frame Logic
FMA	Foundational Model of Anatomy
GALEN	General Architecture for Language, Encyclopedia and Nomenclature
GC	Graphe Conceptuel
GCI	General Concept Inclusion
GRAIL	Galen Representation And Integration Language
HAD	Hospitalisation À Domicile
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IA	Intelligence Artificielle
IC	Ingénierie des Connaissances
ICD	International Statistical Classification of Diseases and related health problems
IO	Ingénierie Ontologique
KACTUS	modelling Knowledge About Complex Technical systems for multiple USe
KAON	KArlsruhe Ontology
KIF	Knowledge Interchange Format
LD	Logique de Description
MAD	Maintient À Domicile
NLM	National Library of Medicine
NLP	Natural Language Processing
OCML	stands for Operational Conceptual Modeling Language.
OIL	Ontology Inference Layer
OntoPAD	Ontologie pour la Prise en charge des patients A domicile
OntoPneumo	Ontologie pour le domaine de la Pneumologie
OWL	Web Ontology Language
PAD	Prise en charge des patients À Domicile
PDA	Personnal Digital Assistant
RACER	Renamed Abox And Concept Expression Reasoner
RDF	Resource Description Framework
RDF(S)	Resource Description Framework Schema
RTO	Ressources Terminologiques et Ontologiques
SAD	Soins infirmiers À Domicile

SHOE	Simple HTML Ontology Extension
SNOMED	Systematized Nomenclature of Medicine
TCP	Transmission Control Protocol
TIC	Technologies de l'Information et de la Communication
TOVE	Toronto Virtual Enterprise
UMLS	Unified Medical Language System
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WWW	World Wide Web
XML	eXtensible Markup Language
XOL	Ontology Exchange Language

INTRODUCTION GÉNÉRALE

1 Contexte de travail

Le domaine **de la Prise en charge des patients À Domicile** (PAD) s'organise autour d'une coopération complexe d'intervenants qui se succèdent au domicile du patient. Une structure de telle organisation évolue continuellement durant son existence puisque de nouveaux partenaires arrivent alors que d'autre sortent selon la situation et plus précisément selon l'état du patient. Les performances et l'efficacité de l'organisation de la PAD résultent de la coopération mise en œuvre, des décisions prises, des capacités et compétences et les ressources offertes par les différents intervenants. Avec l'avancement accru de l'Internet et les nouvelles technologies de l'information et de la communication, la PAD a connu une grande amélioration.

Depuis quelques années, on assiste à l'émergence des **ontologies** qui ont suscité beaucoup d'intérêt du fait qu'elles permettent de représenter les connaissances de plusieurs domaines. Ces représentations de connaissances correspondent à « une spécification explicite et formelle d'une conceptualisation partagée » (Studer, *et al*., 1998). Elles sont au cœur des travaux menés en Ingénierie des Connaissances (IC) et son champs d'applications ne cesse de s'élargir et couvre plusieurs disciplines notamment le web sémantique. Visant à établir des représentations à travers lesquelles les machines puissent manipuler la sémantique des informations, la construction des ontologies 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. Il faut cependant noter que, depuis la naissance de l'intelligence artificielle, plusieurs formalismes de représentation de connaissances ont été développés, permettant de formaliser les connaissances d'un domaine, puis de mettre en œuvre des raisonnements sur ces représentations. Parmi ces formalismes développés au niveau conceptuel pour la modélisation d'ontologie, trois grands modèles sont distingués: les graphes conceptuels, les langages de frames et les logiques de descriptions. Ces formalismes permettent de représenter les concepts sous-jacents à un domaine de connaissance, les relations qui les lient, et la sémantique de ces relations. Par la suite, ces formalismes sont évolués sous l'influence de xml pour être adapté au web sémantique. Au fur et à mesure des expérimentations, des méthodes de construction et des outils de développement adéquats sont apparues. Emergeant des pratiques artisanales initiales, une véritable ingénierie se constitue autour des ontologies, ayant pour but leur construction mais plus largement leur gestion tout au long d'un cycle de vie.

2 Problématique et motivations

Comme dans la plupart des domaines de recherche, le domaine médical est un domaine très complexe, caractérisé par un vocabulaire extrêmement riche, en termes de quantité d'informations médicales importantes véhiculées entre les différents professionnels de santé, et difficile à manipuler. De ce fait, la communauté médicale a visé très lentement la nécessité de modéliser leurs connaissances et de les rendre explicites pour des besoins de partage et réutilisation. Par conséquent, plusieurs **Ressources Terminologiques et Ontologiques** (RTO) ont été proposées. Chacune répond à des besoins précis et divers et modélise une partie du domaine. On cite entre autres : UMLS (Bodenreider & Burgun, 2005), GALEN (Rector, *et al.*, 1995), MENELA (Zweigenbaum, 1994), FMA (Rosse & Mejino, 2003), NAUTILUS (Dieng-Kuntz, et al., 2006), OntoPneumo (Baneyx, 2007),...Etc.

La prise en charge à domicile d'un patient implique une coopération de plusieurs acteurs. Ces derniers partagent un nombre important d'informations médicales. La diversité des intervenants impliqués dans la gestion de soins en situation de PAD et de leurs spécialités (médecins, infirmiers, assistantes sociales, professionnels de rééducation et réadaptation, etc.) peut résulter en des conflits sémantiques lors de l'interprétation des informations médicales transmises. Le problème est qu'il n'y a pas de consensus établi sur la définition des différents termes utilisés. Ces termes dénotant les concepts risquent d'aboutir aux multiples interprétations. Ce type d'ambiguïté sémantique peut résulter en une mauvaise compréhension. Pour cela, l'ontologie est vue comme une solution candidate.

D'après la littérature que nous avons consultée, des architectures ont été proposées pour la PAD dont certaines prévoient l'ontologie, mais qui n'a pas été développée. Dés lors, une ontologie du domaine permet à tous les intervenants de la PAD (professionnels de santé, patients et familles) de partager un vocabulaire compréhensible commun dépourvu de toute ambigüité et leur fournit alors une même vision du domaine. Bien que cette ontologie soit orientée pour la communication entre personnes, mais son intégration au sein d'un système informatique en vue d'assurer l'interopérabilité entre agents logiciels est aussi envisageable.

3 Objectifs de travail

Notre contribution consiste alors en la construction d'une ontologie de domaine de la prise en charge des patients à domicile. Le développement de cette ontologie doit suivre un processus de construction d'ontologies comptant un ensemble de phases spécifiées de façon très détaillée afin

de cerner l'étendue et d'aboutir à une ontologie qui répond aux besoins. Nous accomplirons dans ce manuscrit deux objectifs:

- ***le premier objectif*** consiste à décrire un processus de construction d'ontologie partant de connaissances brutes et arrivant à une ontologie opérationnelle représentée par le langage OWL. Pour ce faire, plusieurs étapes sont suivies afin d'expliciter et de guider le développement d'ontologie. Il consiste en un enchainement de cinq phases : spécification, conceptualisation, formalisation, opérationnalisation et évaluation. En effet, la réalisation de ces différentes étapes s'appuie sur les résultats des travaux réalisés en Ingénierie Ontologique (IO), appliqués dans la pratique à la construction d'une ontologie. Ces travaux mettent l'accent sur l'aspect de représentation formelle et de raisonnement, et utilisent en particulier le formalisme de la logique de descriptions qui possède une sémantique claire et procure des services d'inférences et des mécanismes de raisonnement puissants. Par ailleurs, ce processus prend en considération les conseils et les principes d'IO acceptés par la communauté ontologique.
- ***le deuxième objectif*** consiste à bâtir l'ontologie du domaine en suivant les étapes du processus défini. Nous commençons par la phase de spécification qui consiste à établir un document de spécification des besoins servant à décrire l'ontologie à construire. Ensuite, nous organiserons et structurerons les connaissances acquises en utilisant des représentations intermédiaires semi formelles qui sont faciles à comprendre et indépendantes de tout langage d'implémentation. Après, nous utiliserons le formalisme des logiques de description pour représenter l'ontologie dans un langage formel et par la suite nous coderons l'ontologie en langage OWL en nous appuyant sur outil PROTÉGÉ-OWL. Finalement, nous évaluerons la consistance et la cohérence de l'ontologie formelle et opérationnelle, résultat de la phase d'opérationnalisation, à travers le raisonneur RACER.

4 Organisation du mémoire

Ce mémoire est structuré de la manière suivante : les chapitres I et II présentent l'état de l'art relatif aux domaines en lien avec notre travail et le chapitre III décrit notre contribution.

- *Le **chapitre I*** s'intéresse à présenter le concept Homecare notamment en France dans le contexte de l'hospitalisation à domicile. Nous nous concentrons sur son intérêt, ses différents acteurs impliqués et par une brève description des projets et des travaux abordant la PAD.

- *Le **chapitre II*** est dédié à éclaircir la notion de l'ontologie plus particulièrement en ingénierie des connaissances et ses différentes classifications. Nous étudierons également le problème délicat de la construction d'une ontologie en proposant une revue des méthodologies proposées. Nous nous intéressons aussi aux principaux formalismes de représentation de connaissances, aux langages existants pour exprimer les ontologies et aux différents outils existants développés autour de l'IO aidant à la construction d'ontologies. Enfin, nous clôturons par un bref aperçu sur quelques ontologies médicales existantes.
- *Le **chapitre III*** est consacré à la construction de l'ontologie du domaine de la prise en charge des patients à domicile. Nous nous attachons dans ce chapitre à :
  - Décrire un processus de développement d'une ontologie de domaine afin de faciliter la construction de l'ontologie du domaine de la PAD.
  - Construire l'ontologie du domaine en suivant le processus décrit.
- Ce mémoire s'achève par une conclusion générale en présentant un récapitulatif du contexte de recherche de notre étude et notre contribution tout en planifiant les perspectives que nous envisageons pour compléter ce travail.

CHAPITRE I

LA PRISE EN CHARGE DES PATIENTS À DOMICILE

1.1 Introduction

La Prise en charge des patients À Domicile (PAD) pour tout type de soins en une situation clinique donnée, de toute personne en situation de dépendance temporaire ou permanente quel que soit son âge, relève des soins médicaux, infirmiers et de rééducation, mais renvoie également à une prise en charge médico-sociale et dans certains cas éducatives, et aux aides à l'accomplissement des actes de la vie quotidienne. Elle requiert l'intervention coordonnée de l'ensemble des participants (infirmiers, médecins, et autres professionnels) engagé dans la coordination et la dispensation des soins à domicile de manière professionnelle.

Les services à domicile ne forment pas un nouveau champ ajouté aux services de santé et des services sociaux, mais bien une nouvelle manière de répondre aux exigences, adaptée d'avantage à la réalité d'aujourd'hui. De ce fait, ils ne peuvent être vus seuls, comme un secteur en soi, mais plutôt attachés à d'autres services, offerts par le système de santé et de services sociaux.

Dans ce chapitre, nous débutons par la présentation du concept HomeCare ainsi que la définition de l'hospitalisation à domicile. Ensuite, nous expliquons l'intérêt de l'Hospitalisation À Domicile (HAD) et nous décrivons le rôle de chaque acteur impliqué dans l'HAD. Enfin, nous exposons plusieurs systèmes supportant la PAD.

1.2 Le concept HomeCare

Le concept Américain 'Homecare' est émis par le Pr Bluestone aux Etats-Unis en 1947 faisant de lui le créateur de la prise en charge des patients à domicile. Celle-ci s'est développée rapidement ces dernières années dans plusieurs pays : France, Japon, pays scandinaves et anglo-saxons,...etc. Certes, les modes d'organisation de soins dans ces pays se différencient d'un pays à un autre ou encore au sein d'un même pays (c-à-d d'une région d'un même pays à une autre), mais ils partagent tous les mêmes caractéristiques essentielles. Cette distinction est remarquable au niveau des définitions aussi dans les appellations qui lui sont attribuées.
Dans la littérature:

- Au Etats-Unis: Home Health Care, Home care;
- En Canada: home care

- En France: prise en charge des patients à domicile PAD, la santé à domicile. Ces termes se réfèrent à une variété de modalités de soins, notamment : une Hospitalisation À Domicile HAD (ou bien hospitalisation hors murs), Soins infirmiers À Domicile SAD (ou services de soins à domicile), Maintient À Domicile MAD (soins infirmiers pour les personnes âgées) …etc.
- En Italie: assistance à domicile ou traitement à domicile,
- En Australie: hospital in the home
- En JAPAN: home care services
- Etc.

Ainsi, ce mode de prise en charge est structuré de différentes façons, en fonction de plusieurs critères : règlements énoncés par la tutelle, son fonctionnement général (administratif, financier), les différents intervenants à domicile (nature, statut), population concernée, les services fournies et de même que le matériels utilisés …etc.

Vue l'inexistence d'une organisation standard de la prise en charge des patients à domicile, et sa diversité dans les différents pays, notre étude sera consacrée à la PAD en France dans le contexte de l'hospitalisation à domicile HAD.

1.3 Définition du champ d'HAD (ARH, 2006)

En 1986 : « l'hospitalisation à domicile recouvre l'ensemble des soins médicaux et paramédicaux délivrés à domicile à des malades dont l'état ne justifie pas le maintien au sein d'une structure hospitalière. Ces soins doivent être d'une nature et d'une intensité comparables à ceux qui étaient susceptibles de leur être prodigués dans le cadre d'une hospitalisation traditionnelle ».

En 1992 les structures d'HAD sont définies comme : « des structures permettant d'assurer au domicile du malade, pour une période limitée mais révisable en fonction de son état de santé, des soins médicaux et paramédicaux continus et nécessairement coordonnés. Ces soins se différencient de ceux habituellement dispensés à domicile par la complexité et la fréquence des actes».

En 2000 : « l'hospitalisation à domicile concerne les malades atteints de pathologies graves, aiguës ou chroniques, évolutives et/ou instables qui, en l'absence d'un tel service, seraient hospitalisés en établissement de santé. L'HAD a pour objectif d'éviter ou de raccourcir l'hospitalisation en service de soins aigus ou de suite et réadaptation, lorsque la prise en charge à domicile est possible ».

1.4 Intérêt de la PAD

La PAD a pour objectif principal d'améliorer le confort du patient dans de bonnes conditions de soins et de (ANAES, 2004):

- Poursuivre le traitement et les soins.
- Donner des soins continus à l'occasion d'une maladie chronique ou évoluant par poussées. ou grave d'emblée conduisant à des soins palliatifs pouvant aller jusqu'à la phase ultime.
- Prévenir la maladie et les complications.
- Réduire le risque de survenue d'un problème de santé.
- Adapter le fonctionnement de la personne à l'environnement.
- Soutenir et soulager la famille qui aide un proche en situation de dépendance.

1.5 L'organisation de l'HAD

Les différents acteurs intervenants dans un processus d'HAD sont les suivants (DirectionRégionaleDuServiceMédical, *et al.*, 2003) (ANAES, 2003) :

- **Le médecin hospitalier**

Le médecin hospitalier peut avoir deux rôles distincts. D'une part, un rôle de soins personnalisés aux malades dont il a la charge. Dans ce cadre, il transmet au médecin coordonnateur du service d'HAD et au médecin traitant les informations médicales concernant son patient, il élabore le projet thérapeutique en lien avec l'équipe de l'HAD, il s'engage à suivre le patient au niveau hospitalier et à le ré-hospitaliser si nécessaire. D'autre part, un rôle d'expert pour certaines pathologies dont les traitements complexes sont fréquents en HAD, comme la cancérologie, la cardiologie et la neurologie, etc.

- **Le médecin traitant**

Le médecin traitant, praticien exerçant à titre libéral, est choisi librement par le malade. Qu'il soit ou non prescripteur de l'HAD. Il est le pivot de la prise en charge du patient à domicile et il est responsable du suivi du malade. L'hospitalisation à domicile ne peut se réaliser qu'avec son accord, au vu du projet thérapeutique. Il réévalue, avec l'équipe d'HAD, l'état de santé du patient et adapte les prescriptions en fonction de son évolution, en lien, si besoin, avec le service hospitalier où a été hospitalisé le patient. Il décide de l'hospitalisation en milieu hospitalier traditionnel, si nécessaire.

La rencontre du médecin traitant et de l'équipe soignante du service d'HAD est de nature à faciliter la prise en charge et le suivi du patient.

- **Le médecin coordonateur**

Le médecin coordonnateur est le référent médical de la structure. Il émet un avis médical pour toute admission et sortie d'un patient de la structure d'HAD, il s'appuiera pour cette décision médicale sur le projet thérapeutique du patient, proposé par le médecin prescripteur de l'HAD. Il contribue à l'échange d'informations nécessaires à une prise en charge globale et coordonnée du patient et assure les contacts avec les médecins libéraux et hospitaliers. Par ailleurs, il a un rôle de formateur auprès de l'équipe soignante, participe à l'évaluation de la qualité du service d'HAD, et aux décisions stratégiques de la structure.

- **L'infirmière coordinatrice**

L'infirmière coordonnatrice est l'interlocuteur privilégié pour les médecins prescripteurs hospitaliers ou libéraux. Lors de l'admission d'un patient, elle le rencontre lui et sa famille pour recueillir leurs attentes, évaluer la faisabilité du retour au domicile et anticiper l'organisation à mettre en place pour assurer la sécurité du patient. En ce sens, elle vérifie l'existence et l'accord du médecin traitant. Elle collecte les données administratives et médicales nécessaires a la prise en charge, recueille le projet thérapeutique auprès de l'équipe médicale et complète les informations par des données personnalisées recueillies auprès de l'équipe paramédicale. L'analyse de ces informations lui permet de déterminer les modalités nécessaires à la prise en charge, le nombre d'intervenants et leur qualification, le nombre de passage, les besoins en matériel et aides diverses. Dans l'évaluation des situations complexes, la coordinatrice peut intervenir en binôme avec un psychologue ou une assistante sociale en lien avec le médecin coordonnateur.

- **Le psychologue**

Un soutien psychologique est donc parfois nécessaire, tant pour le malade que pour la famille qui peut vivre parfois des situations difficiles lorsqu'elle est confrontée à la maladie et, notamment, à la mort de son proche à domicile.

D'autres intervenants paramédicaux salariés ou libéraux peuvent être sollicités pour compléter la prise en charge des patients (kinésithérapeutes, orthophonistes, diététiciens,…etc.).

1.6 Le déroulement d'une HAD

Le déroulement d'une HAD suit les étapes suivantes[1] :

a) La préparation de l'admission

Une infirmière coordinatrice procède à une évaluation des besoins du patient avec l'équipe médicale hospitalière et de ville afin de déterminer la faisabilité de l'admission en adéquation

[1] www.santeservice.aso.fr

avec les critères définis par les autorités sanitaires. Elle rencontre le patient et son entourage, leur délivre les informations utiles sur les conditions de sa prise en charge. Si nécessaire une assistante sociale se déplace au domicile pour aider à la mise en place de l'HAD.

b) L'admission

L'admission est prononcée après avis du médecin coordonateur sur la base du projet thérapeutique. En fonction du lieu de son domicile le patient dépend d'un secteur de soin.

c) La conduite de la prise en charge

Une fois le malade pris en charge, les intervenants s'engagent dans un projet thérapeutique et coordonnent les actes pratiqués au domicile du patient. Les intervenants se font autour d'une évaluation globale et permanente des besoins du malade, et le transfert des informations est assuré an amont et en aval de la prise en charge.

Une réévaluation des besoins est faite dés le premier jour de la prise en charge. Le matériel médical (lits médicalisés, oxygène, appareils respiratoires, etc) et livré la veille. L'équipe pluridisciplinaire intervienne pour mettre en place des protocoles de soins dans le respect des règles de sécurité, d'hygiène, de qualité et de confidentialité. La présence d'un dossier de soins et de transmission au chevet du malade facilite la coordination entre l'équipe soignante, l'encadrement, le médecin et l'hôpital.

d) Les évaluations régulières de l'HAD

Toutes les semaines, les équipes de soins se retrouvent avec leurs cadres pour évaluer les besoins des patients ainsi que le déroulement de la prise en charge. L'évaluation de la charge en soins du personnel soignant salarié de la structure se fait de manière quotidienne ce qui permet un réajustement en temps réel des réponses aux besoins des patients.

A tout moment, l'évaluation clinique faite par le médecin traitant ou le médecin hospitalier pourra conduire à la poursuite de cette prise en charge ou à son arrêt pour ré-hospitalisation ou relais vers une autre structure.

e) Le patient

Le patient est au centre de la prise en charge coordonnée à domicile.

f) La sortie

La prise en charge terminée, alors un relais vers une nouvelle structure est organisé. C'est le médecin hospitalier ou le médecin traitant qui décide de l'arrêt pour ré-hospitalisation ou un relais vers une autre structure.

La sortie du patient est planifiée, coordonnée aussi un accompagnement spécifique peut être mis en place en fonction des situations rencontrées.

1.7 Systèmes de prise en charge des patients à domicile

1.7.1 Travail d'ITABASHI et ses collègues

Itabashi et ses collègues (Itabashi, *et al.*, 2005) ont développé un système (voir figure 1.1) qui soutienne la prise en charge des patients à domicile, accessible via les PDAs des différents acteurs. L'architecture proposée est composée de trois agents, à savoir : 'interface agent', 'Scheduler agent' et le 'Helper agent'. La planification des soins est réalisée de façon autonome par négociation entre agents. Une fois acceptée par les personnels d'aide ainsi que les patients, la planification sera fixée. En cas de désagrément entre les différents acteurs du système, les plans d'intervention seront reconstruits.

Les auteurs affirmaient que le système proposé permet de réduire le coût total de la PAD, et ils suggéraient l'utilisation d'une ontologie qui assurera une négociation plus flexible des plans d'intervention.

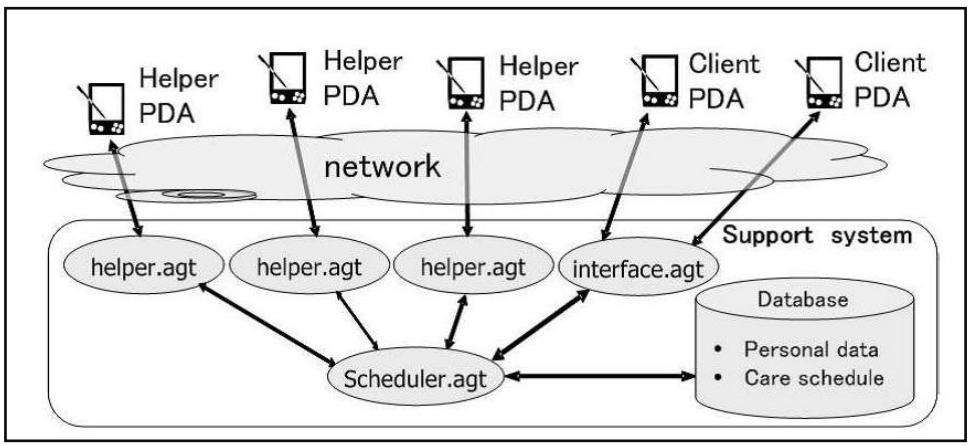

Figure 1.1 Configuration du système.

1.7.2 COQUAS (COordination pour la QUAlité de Soins)

COQUAS est un projet National Français proposé par Bricon-Souf et ses collègues (Bricon-Souf, *et al.*, 2005) qui a pour but l'amélioration de la coopération entre les professionnels de santé, patients et famille durant un processus PAD. Un système prototype modulaire est conçu pour prendre en considération les aspects de coopération et d'interopérabilité. L'interface (voir figure 1.2) est construite de façon dynamique en se basant sur les méta-descriptions des actions et les informations résultantes de l'analyse cognitive effectuée.

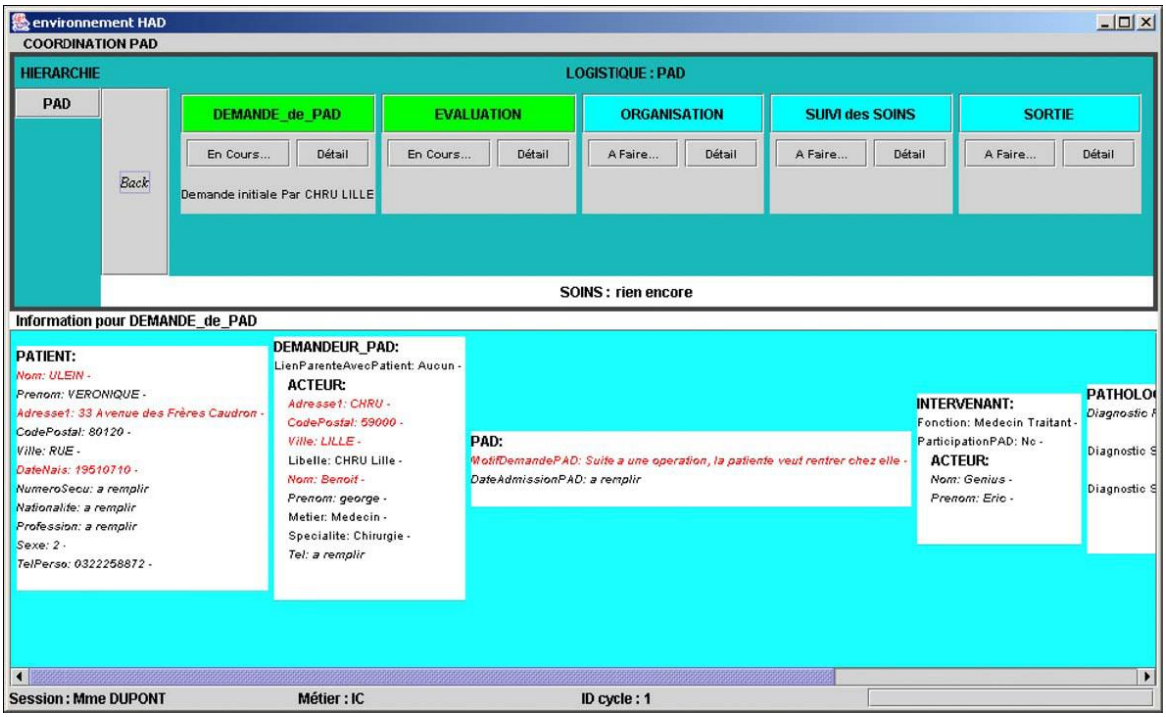

Figure 1.2 Interface de coordination.

1.7.3 Le projet K4CARE

Le projet K4Care (Isern, *et al.*, 2007) a donné naissance à une plateforme multi-agent accessible via le Web (voir figure 1.3). Cette dernière est issue d'un modèle générique dont le but est l'unification de l'organisation de la PAD entre les pays européens. La propriété clé du system proposé est le potentiel d'adaptation de l'accès aux profils de ses éventuels utilisateurs. Pour ce faire, deux ontologies sont utilisées : '*Actor Profile Ontology*' et '*Case Profile Ontology'*.

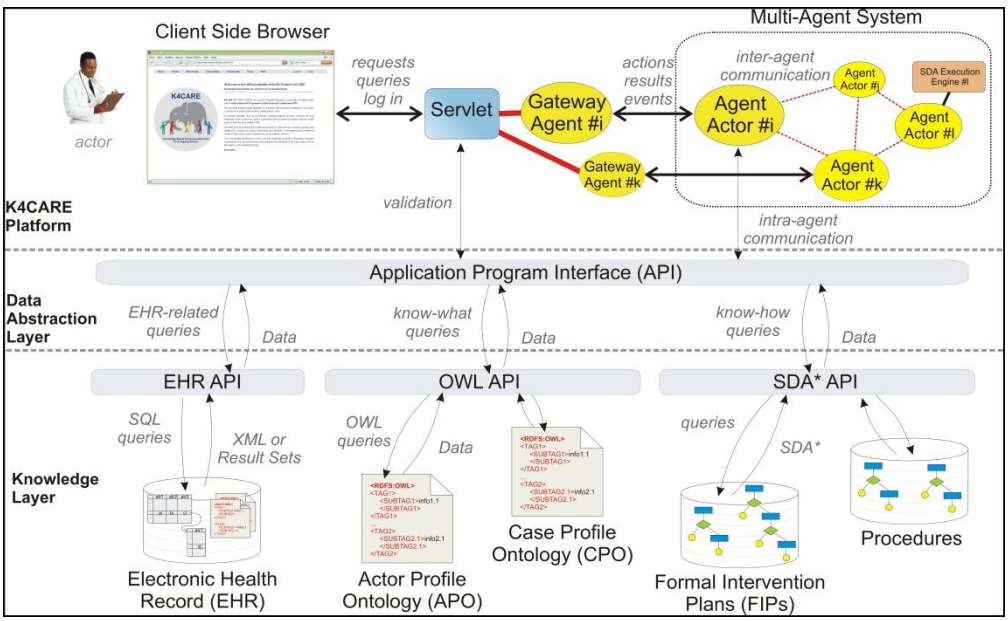

Figure 1.3 Architecture du K4CARE.

1.7.4 Travail de Koutkias et ses collègues

Koutkias et ses collègues (Koutkias, *et al.*, 2005) ont implémenté un système où l'architecture est basée sur le paradigme agent. Le système multi-agent présenté a pour objectif d'accroitre le monitorage, la surveillance et les services éducationnelles d'un centre de contacte médicale de gestion des maladies chroniques. Dans tel scenario d'une prise en charge à domicile, un suivi efficace permet de gérer et d'interpréter effectivement le volume large des données médicales collectés durant la session du patient avec le système ainsi que d'évaluer l'utilisation des ressources du centre de contacte médicale. Spécifiquement, l'objectif du système est de surveiller l'environnement du centre de contacte médicale, de détecter les cas importants et d'informer les professionnels de santé et le personnel administratifs à travers des messages d'alertes, des notifications et des rapports pour agir en fonction de la situation.

1.7.5 Travail de K.Zarour et N.Zarour

K.Zarour et N.Zarour (Zarour & Zarour, 2007) ont mis en œuvre une architecture basée agent baptisée ABAH (Agent Based Architecture For Homecare) supportant la PAD et comportant tous les composants nécessaires pour son fonctionnement. L'objectif de cette architecture est de rendre plus efficace la coopération des acteurs principaux et leur permet d'exercer au maximum leurs compétences et leurs capacités afin que le système de la PAD dispense des soins appropriés, continus et d'augmenter la qualité des services offerts aux patients.

1.8 Conclusion

Les services à domicile représentent certainement une solution d'avenir, ils connaîtront, selon toute vraisemblance, une croissance importante au cours des prochaines années. Il est important ; aujourd'hui, de se donner une organisation forte de services à domicile et de mettre en place les conditions nécessaires pour assurer leur développement.

En effet, les Technologies de l'Information et de la Communication (TICs) jouent un rôle primordial dans la PAD pour améliorer l'accès à ses services, échanger facilement l'information et partager les services, une meilleure utilisation des ressources, de façon à améliorer la circulation de l'information et mieux organiser les services.

Le chapitre suivant est consacré à l'ingénierie ontologique et à la présentation de quelques ontologies médicales.

CHAPITRE II

INGÉNIERIE ONTOLOGIQUE

2.1 Introduction

Durant cette dernière décennie, Nous avons remarqué qu'une attention croissante a été concentrée sur l'ingénierie ontologique où l'ontologie est l'objet fondamental sur lequel il faut se pencher. Les ontologies sont largement utilisées et ont prouvé leurs utilités dans de nombreux domaines tels que : l'ingénierie de connaissances, l'intelligence artificielle, l'intégration des sources de données, la recherche d'information, l'e-commerce et sont au cœur du Web Sémantique[1]. Cet engouement est motivé par le fait que les ontologies sont un moyen efficace pour la gestion et le partage des connaissances d'un domaine particulier entre personnes et\ou systèmes.

Ce chapitre a pour titre ingénierie ontologique qui est une discipline de l'informatique référant à l'ensemble des activités qui concernent le processus de développement d'ontologie, son cycle de vie, les méthodes et les cadres méthodologiques de sa construction ainsi que les outils et les langages d'implémentation d'ontologies.

Nous allons donc présenter dans ce chapitre, en premier lieu la notion d'ontologie plus particulièrement en ingénierie des connaissances, en relevant quelques définitions qui lui ont été attribuées et les différents éléments qui la constituent. Par la suite, nous présenterons ses différentes classifications en se concentrant sur la richesse de la structure interne de l'ontologie, l'objet de conceptualisation, le niveau de granularité et le degré de formalisation. Nous étudierons également le problème délicat de la construction d'une ontologie en proposant une revue des méthodologies tout en distinguant entre celles dont la conception est effectuée manuellement et les autres qui se focalisent sur les outils de traitement de langue pour rendre cette conception semi-automatique ou bien automatique. Ainsi, nous faisons un étalage des principes d'ingénierie ontologique abordés dans la littérature qui permettent d'orienter la construction d'une structure ontologique. En outre, nous faisons un survol des principaux formalismes de représentation de connaissances à savoir les frames, les graphes conceptuels et les logiques de description qui sont à

[1] D'après Tim Berners-Lee, Le Web Sémantique se veut une extension du Web actuel dans laquelle l'information est décrite de manière bien définie, afin de permettre aux agents logiciels de procéder à des traitements automatiques portant sur le contenu des ressources du Web et sur la signification de ce contenu.

l'origine des langages permettant d'exprimer des ontologies. Nous verrons alors comment ces formalismes ont évolué tout récemment, sous l'influence du langage XML, pour être adaptés au Web sémantique. De plus, nous donnons un bref aperçu de quelques outils existants permettant la construction de l'ontologie. Enfin, nous clôturons par un panorama de quelques ontologies médicales existantes.

2.2 Qu'est-ce que l'ontologie en Ingénierie de connaissances

Les ontologies, à l'origine d'une branche de la philosophie qui s'intéresse à la nature et l'organisation de la réalité, correspondent à ce qu'Aristote appelait la Philosophie première, c'est-à-dire la partie de la métaphysique qui s'intéresse à l'être en tant qu'être, par opposition aux philosophies secondes qui s'intéressent à l'étude des manifestations de l'être (Garf, 1996). En informatique, la littérature fournit un tas de définitions du mot ontologie. Ces définitions, dans leur diversité, offrent des points de vues à la fois différents et complémentaires. Dans la section suivante, nous décrivons quelques unes.

2.2.1 Quelques définitions

Neches et ses collègues (Neches, *et al.*, 1991) ont été les premiers à en proposer une définition :

> « *An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary*».

[Traduction : une ontologie définit les termes et les relations de base du vocabulaire d'un domaine ainsi que les règles qui indiquent comment combiner les termes et les relations de façon à pouvoir étendre le vocabulaire]. Cette définition descriptive donne un premier aperçu sur la manière de construire une ontologie, à savoir l'identification des termes de bases d'un domaine et les relations entre ces termes ainsi que les règles pouvant s'appliquer sur ces derniers.

En 1993, Gruber (Gruber, 1993a) a proposé une définition à cette notion qui est la plus célèbre couramment citée dans la littérature :

> « *An ontology is an* ***explicit specification*** *of a* ***conceptualization*** ».

[Traduction: une ontologie est une spécification explicite d'une conceptualisation]. Il a introduit la notion de 'conceptualisation' qui réfère à un modèle abstrait d'un certain domaine du monde réel en identifiant les concepts pertinents décrivant ce domaine. Le terme 'explicite' signifie que les concepts utilisés ainsi que les contraintes sur leur emploi, sont réellement définis d'une manière claire et précise.

Depuis, plusieurs raffinements à cette définition ont été proposés.

En 1997, Borst (Borst, 1997) a modifié légèrement la définition de Gruber en citant qu'une ontologie est définie comme étant :

> *« An ontology is a **formal specification** of a **shared conceptualization** ».*

[Traduction: une ontologie est une spécification formelle d'une conceptualisation partagée]. Cette définition précise d'une part, le fait que l'ontologie doit être 'formelle', c'est à dire exprimée sous forme d'une logique pouvant être exploitable par une machine. D'autre part, elle doit être 'partagée' dans la mesure où elle doit capturer des connaissances partagées entre différents individus.

En 1998, Studer et ses collègues (Studer, *et al.*, 1998) ont rassemblé ces deux définitions (celles de Gruber et Borst) dans une seule qui est :

> *« An ontology is a **formal**, **explicit specification** of a **shared conceptualization.** Conceptualization refers to an abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon. Explicit means that the type of concepts used, and the constraints on their use are explicitly defined. Formal refers to the fact that the ontology should be machine-readable. Shared reflects the notion that an ontology captures consensual knowledge, that is, it is not private of some individual, but accepted by a group ».*

[Traduction: une ontologie est une spécification formelle et explicite d'une conceptualisation partagée]. Ils l'expliquent comme suit :

- ***Spécification explicite*** signifie que les concepts, les propriétés, les relations, les fonctions, les restrictions et les axiomes de l'ontologie sont définis de façon déclarative ;
- ***Formelle*** réfère au fait qu'une ontologie doit être traduite dans un langage interprétable par une machine (*machine-readable*) ;
- ***Conceptualisation*** réfère à un modèle abstrait d'un phénomène du monde en identifiant les concepts appropriés à ce domaine ;
- ***Partagé*** réfère au fait qu'une ontologie capture la connaissance consensuelle c'est-à-dire non réservée à quelque individus, mais partagée par un groupe ou une communauté.

En 1995, Guarino et Giaretti (Guarino & Giaretta, 1995) proposent leur définition :

> *« A **logical theory** which gives an **explicit**, **partial account** of a **conceptualization** ».*

[Traduction : Une ontologie est une théorie logique proposant une vue explicite et partielle d'une

conceptualisation]. Une ontologie est dite ***partielle*** dans le sens où une conceptualisation ne peut pas toujours être entièrement formalisée dans un tel cadre, du fait d'ambiguïtés ou du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation choisi (Fürst, 2002).

En 2000 Aussenac-Gilles et ses collègues (Aussenac-Gilles, *et al.*, 2000) énonce que :

> *« Une ontologie organise dans un réseau des concepts représentant un domaine. Son contenu et son degré de formalisation sont choisis en fonction d'une application ».*

Cette définition souligne la dépendance entre le degré de formalisation de l'ontologie et son contenu avec l'application dans laquelle elle va être utilisée.

Il existe un autre groupe de définitions basées sur le processus de construction d'une ontologie, alors que celles précitées sont indépendamment de ce dernier. Ce type de définition souligne la relation entre une ontologie et une base de connaissances.

En 1996 Bernaras et ses collègues (Bernaras, *et al.*, 1996) au sein du projet KAKTUS propose :

> *« It [an ontology] provides the means for describing explicitly the conceptualization behind the knowledge represented in a knowledge base ».*

[Traduction: Une ontologie fournit le moyen pour décrire d'une manière explicite la conceptualisation des connaissances représentées dans les bases de connaissances]. Cette définition propose l'extraction d'ontologie à partir d'une base de connaissances, ce qui reflète l'approche utilisée par les auteurs pour construire une ontologie.

Dans le même ordre d'idée, en 1997 Swartout et ses collègues (Swartout, *et al.*, 1997) au sein du projet SENSUS :

> *« An ontology is a hierarchically structured set of terms for describing a domain that can be used as a skeletal foundation for a knowledge base ».*

[Traduction: une ontologie est un ensemble de termes hiérarchiquement structurés, conçu afin de décrire un domaine qui peut être utilisé comme un squelette de base pour les bases de connaissances]. Selon cette définition, une ontologie peut servir à construire plusieurs bases de connaissances qui peuvent partagés la même taxonomie.

Selon Gomez et ses collègues (Gómez-Pérez, *et al.*, 2004), les ontologies visent à capturer les connaissances ***consensuelles*** de façon générique ainsi que la façon de leur réutilisation et leur partage en travers des applications et des groupes de personnes.

2.2.2 Composantes de l'ontologie

Les ontologies rassemblent les connaissances propres à un domaine particulier. Ces connaissances sont formalisées en mettant en jeu les composants suivants: concepts ; relations ; axiomes et instances.

2.2.2.1 Concepts

Un concept peut représenter un objet, une idée, ou bien une notion abstraite (Uschold & King, 1995). Ils sont appelés aussi classes de l'ontologie dans certain travaux. Un concept peut être divisé en trois parties : ***un terme (***ou plusieurs***)***, ***une notion*** et un ***ensemble d'objets***.

- Le terme (ou bien label) d'un concept est l'expression linguistique utilisée couramment pour y faire référence.
- La notion désigne ce qui est appelé, au sens de la représentation des connaissances, ***l'intension*** du concept. Elle contient sa sémantique qui est définie à l'aide de propriétés (relations et attributs), de règles et de contraintes.
- L'ensemble d'objets définis par le concept forme ce qui est appelé ***l'extension*** du concept. Il s'agit des objets auxquels le concept fait référence, autrement dit, de ses instances.

L'intension et l'extension d'un concept sont deux aspects bien distincts : deux extensions peuvent ne pas être des ensembles disjoints tandis que deux intensions ont comme propriété de s'exclure mutuellement. Cependant, deux concepts même désignés avec le même terme, peuvent partager une même extension mais pas une même intension. Cette difficulté est due aux différents points de vue que l'on peut avoir sur un même objet. C'est le cas pour « table » qui est à la fois un meuble constitué d'un plateau et de quatre pieds et un meuble sur lequel on peut poser des objets et autour duquel on peut s'asseoir : les extensions sont les mêmes (l'ensemble des objets ayant une de ces deux descriptions) mais les intensions sont différentes. Il s'agit d'un cas typique d'homonymie de termes. Dans le même ordre d'idées, il paraît indispensable de gérer les synonymies. Ces deux problèmes compliquent la tâche de l'ingénieur des connaissances lorsqu'il doit choisir comment désigner, dans la langue, un concept.

Certains auteurs, parmi lesquels (Gómez-Pérez, *et al.*, 2004) souligne la nécessité de désigner un concept par plusieurs termes. D'autres tels que (Baneyx, 2007) proposent plutôt de désigner un concept par un seul label et de relier ce label à un ensemble de termes préférés. Cela a l'avantage de marquer une différence nette entre le statut de label et celui de terme utilisé dans la langue.

2.2.2.2 Relations

Les relations sémantiques unissent les concepts du segment analysé de la réalité entre eux et

traduisent les associations existantes. Formellement, elles sont définies comme étant tout sous-ensemble d'un produit de *n* ensembles, c'est-à-dire: $R \subset C1 * C2 * \ldots Cn$.

Les ontologies généralement contiennent que des relations binaires. Le premier argument d'une relation binaire est dit domaine, alors que le deuxième argument est dit co-domaine. Cela permet de designer la façon dont la relation doit être lue. Ces relations sont caractérisées par un terme (voire plusieurs), une signature qui précise le nombre d'instances de concepts que la relation lie et leurs types. Elles englobent les associations suivantes : sous-classe de (généralisation-spécialisation) ; partie de (agrégation ou composition) ; associée-à ; instance de ; est un ; etc.
On distingue alors, les relations taxonomiques (dite aussi de subsomption) et les relations associatives.

a) Relation de subsomption

La relation de subsomption « est-un » (is-a) a un statut particulier car elle structure **la hiérarchie ontologique**. À ce titre, elle est implicite. Un concept C1 (concept père) subsume un concept C2 (concept fils) si toute propriété sémantique de C1 est également une propriété sémantique de C2 et si C2 est plus spécifique que C1. Ainsi, l'extension d'un concept est forcément plus réduite que celle de son concept père. Son intension est par contre plus riche. Selon (Baneyx, 2007), la relation de subsomption est définie dans la littérature de plusieurs manières :

- Définition intensionnelle : un concept C1 subsume un concept C2 si tout individu décrit par C2 l'est aussi par C1, autrement dit si l'ensemble des propriétés d'un individu dont la description est définie par C2 contient l'ensemble des propriétés spécifiées par C1.
- Définition extensionnelle : un concept C1 subsume un concept C2 si l'ensemble des individus dénotés par C1 contient l'ensemble des individus dénotés par C2.
- Définition logique : un concept C1 subsume un concept C2, si être un individu décrit par C2 implique être un individu décrit par C1.

La relation de subsomption n'est pas la seule relation qui permette de structurer la hiérarchie ontologique, la relation de méronymie, « partie-tout » (part-of) est souvent utilisée.

L'**héritage multiple** est une propriété qui peut être définie sur la relation de subsomption : un concept d'une ontologie peut avoir plusieurs pères par la relation de subsomption. L'héritage multiple implique que le concept hérite des propriétés de tous ses pères.

b) Relation associative

Les relations « associatives » sont des relations d'interaction entre deux concepts qui ne sont pas la relation de subsomption. La désignation « relation associative » est empruntée aux domaines de

la bio-informatique. Elles correspondent à la notion de rôle en Logique de Description (voir la section 2.7.3) et permettent de typer les concepts reliés.

Les relations binaires sont parfois utilisées pour exprimer les attributs d'un concept. Les attributs sont distingués des relations par leur co-domaine qui est un type de donné tel que: nombre, chaîne de caractères….etc. Tandis que le co-domaine d'une relation est d'un genre plus complexe puisqu'il s'agit d'un autre concept présent dans l'ontologie (Gómez-Pérez, *et al.*, 2004).

2.2.2.3 Axiomes

Les axiomes permettent de modéliser des assertions acceptées comme vraies, à propos des abstractions du domaine traduites par l'ontologie. Leur inclusion dans une ontologie peut avoir plusieurs objectifs: interviennent dans la définition des significations des composants d'ontologie, les contraintes sur les valeurs des attributs, les arguments de relations et dans l'inférence de nouvelles informations, etc. D'après (Gómez-Pérez, *et al.*, 2004), Les axiomes formels sont utilisés pour vérifier la consistance de l'ontologie.

2.2.2.4 Instances (ou individus)

Elles constituent la définition extensionnelle de l'ontologie. Ils représentent les éléments singuliers véhiculant les connaissances à propos du domaine.

2.3 Classification des ontologies

Les ontologies peuvent être classifiées en fonction de plusieurs dimensions. Parmi celle-ci, nous examinons quatre, à savoir :

- La richesse de la structure interne des ontologies ;
- L'objet de conceptualisation ;
- Le niveau de granularité ;
- Le niveau de formalisation de la représentation des connaissances.

2.3.1 Typologie selon la richesse de la structure interne des ontologies

Lassila et McGuinness (Lassila & McGuinness, 2001) proposent une classification des ontologies selon la richesse de leur structure interne. Telle qu'illustrée à la figure 2.1, cette classification consiste en un continuum allant du vocabulaire contrôlé à l'ontologie avec des contraintes logiques. Lorsque nous nous déplaçons tout le long du continuum, le sens des termes devient plus précis et le degré de formalisme augmente ce qui réduit l'ambiguïté. Ce critère est connu aussi

dans d'autres travaux « le degré d'**engagement sémantique**[2] **»** qui correspond au niveau de spécification formelle permettant de restreindre l'interprétation de chaque concept et ainsi d'en donner la sémantique (Bachimont, 2000). Autrement dit, le degré d'engagement sémantique fait en particulier référence au niveau sémantique des connaissances que l'ontologie représente.

Les principales catégories de cette classification sont (Lassila & McGuinness, 2001):

- Vocabulaire contrôlé ;
- Glossaire ;
- Thésaurus ;
- Hiérarchie informelle (is-a) ;
- Hiérarchie formelle (is-a) ;
- Hiérarchie formelle (is-a) avec instances du domaine ;
- Frame ;
- Ontologies avec restrictions de valeur ;
- Ontologies avec contraintes logiques.

a) ***Vocabulaire contrôlé*****:** est une liste finie de termes définis par un groupe de personnes ou une communauté. La signification des termes n'est pas forcément définie et il n'y a pas d'organisation logique entre les termes. Les catalogues sont des exemples de cette catégorie.

b) ***Glossaire*****:** est une liste de termes avec leurs significations spécifiées en langage naturel. Cette représentation apporte plus d'informations car une personne peut lire la définition, cependant elle n'est pas interprétable par l'ordinateur.

c) ***Thésaurus*****:** il fournit une sémantique supplémentaire entre les termes tels que la relation de synonymie. Mais, il n'offre pas la structure hiérarchique explicite.

d) ***Hiérarchie informelle (is-a)*****:** est une hiérarchie où la notion vague de généralisation et de spécialisation est fournie bien que ce ne soit pas une hiérarchie qui correspond à la stricte notion de subsomption. Ce qui revient à dire que la structure de ce type d'hiérarchie est basé sur la proximité des concepts.

e) ***Hiérarchie formelle (is-a)*****:** est une hiérarchie dont la structure est déterminée par des relations de subsomption. Dans cette catégorie, si A est une super-classe de B, alors si un objet est une sous-classe de B, il est nécessairement le cas qu'il est ainsi une sous-classe de A. De même, si A est une super-classe de B, alors si un objet est une instance de B,

[2] Un engagement sémantique est d'expliciter clairement le sens des connaissances. Les principes différentiels proposés par Bachimont (voir la section 2.5.2.1) permettent d'exprimer cet engagement sémantique.

alors il est nécessairement le cas qu'il est une instance de A. Cette ontologie ne peut inclure que des noms des classes.

f) ***Hiérarchie formelle (is-a) avec instances du domaine*****:** similaire à la catégorie précédente mais incluant des instances.

g) ***Frame*****:** est une ontologie incluant des classes avec leurs propriétés. Le fait d'inclure des propriétés dans la description d'une classe devient intéressante dans la mesure où ces propriétés pouvant être héritées par les sous classes de la taxonomie formelle « is a ».

h) ***Ontologies avec restrictions de valeur*****:** sont des ontologies pouvant contenir des restrictions sur les valeurs des propriétés.

i) ***Ontologies avec contraintes logiques*****:** sont les ontologies les plus expressives. Ces ontologies pouvant contenir des contraintes entre constituants (exemple relations) définies dans un langage logique. L'ontologiste peut exprimer les contraintes dans un langage d'ontologie expressif tel qu'OWL.

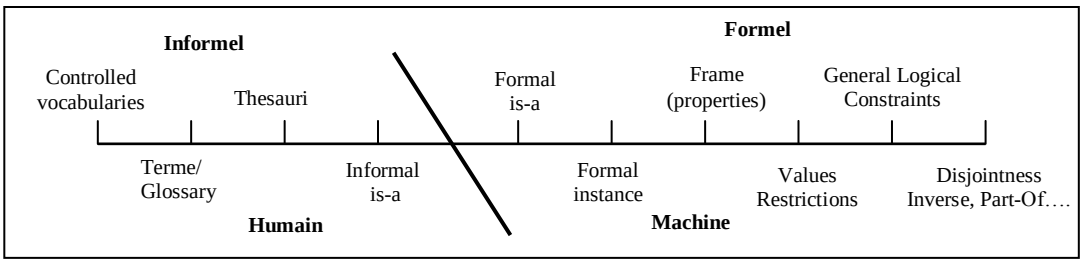

Figure 2.1 Classification de Lassila et McGuinness (Lassila & McGuinness, 2001).

Peu de gens considèrent les quatre premières catégories (le vocabulaire contrôlé, le glossaire, le thésaurus et l'hiérarchie informelle is-a) comme des ontologies et leur justification est qu'ils fournissent une conceptualisation consensuelle d'un domaine. À ce titre, la notion d'ontologie est diluée.

En revanche, Les autres qui sont les plus nombreux, les considèrent comme des **ressources terminologiques**[3]. Notamment, ils préfèrent avoir une hiérarchie formelle incluse avant de considérer quelque chose comme une ontologie. Dans ce même ordre d'idée, Gomez et ses collègues (Gómez-Pérez, *et al.*, 2004) ont distingué les **ontologies légères** (*lightweight ontologies*) et **les ontologies lourdes** (heavyweight ontologies). Les ontologies légères incluent des concepts comprenant des propriétés et qui sont organisés en taxonomies avec des relations

[3] Ou bien une terminologie. Est une ressource faisant intervenir des termes au lieu des concepts. Les termes ne font pas référence à des notions (intention du concept) et des objets (extension du concept) mais définissent uniquement le vocabulaire lié à la connaissance représentée.

conceptuelles. Tandis que les ontologies lourdes ajoutent aux ontologies légères des axiomes et des restrictions clarifiant le sens. Les ontologies lourdes modélisent un domaine de façon plus profonde avec plus de restrictions sur la sémantique du domaine.

2.3.2 Typologie selon l'objet de conceptualisation

Gomez et ses collègues (Gómez-Pérez, *et al.*, 2004) proposent une classification selon le sujet de conceptualisation des ontologies. Cette classification est une extension des travaux de Mizoguchi (Mizoguchi, *et al.*, 1995), ceux de Van Heijst (Van Heijst, *et al.*, 1997), et ceux Guarino (Guarino, 1998) :

- ontologies de représentation de connaissances ;
- ontologies générique / générale / commune ;
- ontologies de haut niveau / de niveau supérieur ;
- ontologies du domaine ;
- ontologies de tâche ;
- ontologies d'application.

a) ***Ontologies de représentation de connaissances (knowledge representation ontology or meta-ontologies)*** (Van Heijst, *et al.*, 1997) **:** elles regroupent les primitives utilisées pour formaliser les connaissances sous un paradigme de représentation de connaissances. L'exemple le plus expressif est l'ontologie de frame (*Frame Ontologie*) (Gruber, 1993a) qui intègre les primitives de représentation des langages à base de *frames*: classes, instances, facettes, propriétés/slots, relations, etc.

b) ***Ontologies générale ou commune ou générique (general*** (Van Heijst, *et al.*, 1997) ***or common*** (Mizoguchi, *et al.*, 1995) ***ontologies):*** les connaissances modélisées dans ce type d'ontologie doivent être générales pour être réutilisées dans différents domaines. Elle Comprend le vocabulaire relatif au temps, espace, unités, etc.

c) ***Ontologies de niveau supérieur ou de haut niveau (top-level or upper-level ontologies)****:* Ce type d'ontologies modélise des concepts de haut niveau auxquels ces derniers doivent être reliés au sommet des ontologies de plus bas niveaux. Cependant, il existe plusieurs ontologies de haut niveau qui se différent par le critère utilisé pour classifier les concepts généraux de la taxonomie.

d) ***Ontologies du domaine (Domain Ontologies)*** (Mizoguchi, *et al.*, 1995) (Van Heijst, *et al.*, 1997) : elles sont réutilisables au sein d'un domaine donné, mais pas d'un domaine à un autre. Les connaissances représentées dans ce type d'ontologies sont spécifiques à un domaine particulier. Elle fournit un vocabulaire d'un domaine spécifique au travers de

concepts et de relations qui modélisent les principales activités, les théories du domaine en question. Les concepts et les relations des ontologies de domaine sont souvent des spécialisations de concepts et des relations définis dans des ontologies de haut niveau.

e) ***Ontologies de tâches (Task Ontologies)*** (Mizoguchi, *et al.*, 1995) (Guarino, 1998) **:** ce type d'ontologie est utilisé pour décrire un vocabulaire relatif à une tâche ou une activité générique (faire un diagnostic, planifier une activité . . .) en spécialisant certains termes des ontologies de haut niveau. Ces ontologies fournissent un ensemble de termes au moyen desquels on peut décrire, au niveau générique, comment résoudre un type de problème.

f) ***Ontologies d'application (Application Ontologies)*** (Van Heijst, *et al.*, 1997) : Ce sont les ontologies les plus spécifiques. Contrairement à l'ontologie de domaine, l'ontologie d'une application donnée ne peut pas être réutilisée pour d'autres applications. Elle contient les connaissances requises pour une application particulière. Ce type d'ontologie décrit des concepts qui dépendent à la fois d'un domaine particulier et d'une tâche particulière. Par conséquent, elle spécialise souvent des ontologies de domaine et des ontologies de tâches pour une application donnée.

2.3.3 Typologie selon le niveau de granularité

On peut distinguer les ontologies selon le niveau de détail utilisé lors de la conceptualisation de l'ontologie en fonction de l'objectif opérationnel envisagé pour l'ontologie, deux catégories peuvent être identifiées : granularité fine et granularité large (Psyché, *et al.*, 2003) :

a) ***Granularité fine :*** elle correspond à une ontologie très détaillée, possédant ainsi un vocabulaire plus riche capable d'assurer une description détaillée des concepts pertinents d'un domaine ou d'une tâche;

b) ***Granularité large :*** elle correspond à un vocabulaire moins détaillé. Les ontologies génériques possèdent une granularité large, compte-tenu du fait que les notions sur lesquelles elles portent peuvent être raffinées par des notions plus spécifiques (Fürst, 2002).

2.3.4 Typologie selon le niveau de formalisation utilisé

Les ontologies peuvent être distinguées en fonction du degré de formalisme utilisé pour les exprimer. Uschold et Grüninger (Uschold & Grüninger, 1996) proposent une classification contenant les quatre catégories : Hautement informelles ; Semi-informelles ; Semi-formelles et Rigoureusement formelles.

a) ***Hautement informelle :*** elle est exprimée en langue naturelle (sémantique ouverte).

b) ***Semi-informelle :*** elle est exprimée dans une forme restreinte et structurée de langage naturel.

c) ***Semi-formelle*** : elle est exprimée dans un langage artificiel défini formellement.

d) ***Rigoureusement formelle :*** l'ontologie est exprimée dans un langage contenant une sémantique formelle, des théorèmes, et des preuves pour vérifier les propriétés telles que la validité et la complétude.

Il est évident qu'il est difficile de faire la différence, pour une ontologie considérée, entre ces quatre différentes classifications et de choisir celle qui correspond

2.4 Cycle de vie d'une ontologie

Puisque les ontologies sont 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 possédants un cycle de vie qui nécessite d'être précisé. Dans ce contexte, les activités liées aux ontologies sont d'une part, des activités de gestion de projet (planification, contrôle, assurance qualité) et d'autre part, des activités de développement (spécification, conceptualisation, formalisation); s'y ajoutent des activités transversales de support telles que l'évaluation, la documentation, la gestion de la configuration (Blazquez, *et al.*, 1998). Un cycle de vie inspiré du Génie Logiciel est proposé dans (Baneyx, 2007) basé sur (Dieng, *et al.*, 2001) et (Gandon, 2006).

Il comprend une étape initiale de détection et de spécification des besoins qui permet notamment de cerner précisément le domaine de connaissances, une étape de conception qui se subdivise en trois phases, une étape de déploiement et de diffusion, une étape d'utilisation, une étape incontournable d'évaluation et enfin une sixième étape consacrée à l'évolution et à la maintenance du modèle. Après chaque utilisation significative, l'ontologie et les besoins doivent être réévalués et l'ontologie peut être étendue et si nécessaire en partie reconstruite. La validation du modèle de connaissances est au centre du processus et se fait de manière itérative (Baneyx, 2007).

Fernandez et ses collègues (Fernández-López, *et al.*, 1997) insiste sur le fait que les activités de documentation et d'évaluation sont nécessaires à chaque étape du processus de construction. L'évaluation précoce permettant de limiter la propagation d'erreurs. Le processus de construction peut être intégré au cycle de vie d'une ontologie comme l'indique la figure 2.2.

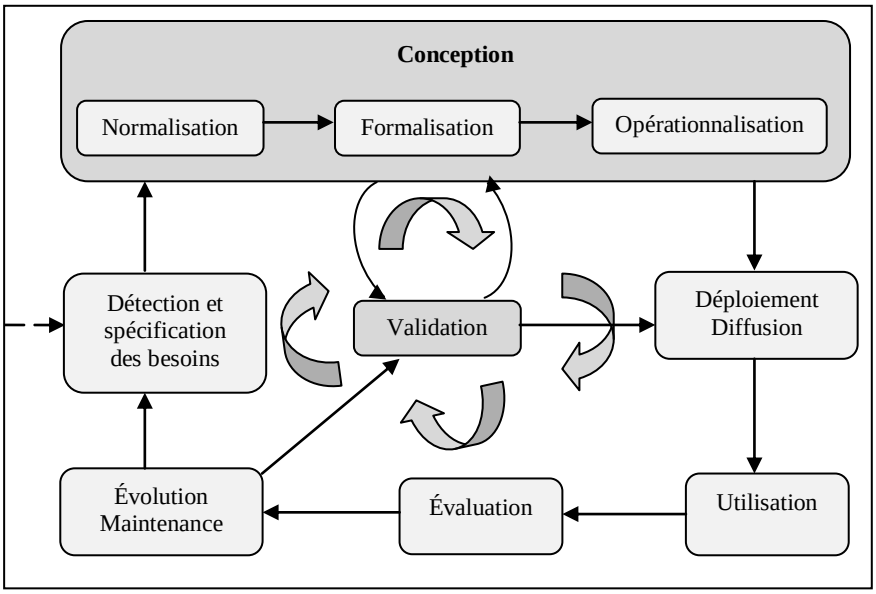

Figure 2.2 Cycle de vie d'une ontologie (Baneyx, 2007).

Loin d'une méthode concrète ou cadre méthodologique pour construire une ontologie, il en ressort que la phase de construction est un enchainement de trois étapes : conceptualisation, formalisation, opérationnalisation. L'étape de formalisation peut être complétée par une étape d'intégration au cours de laquelle une ou plusieurs ontologies vont être importées et réutiliser dans l'ontologie à construire.

2.5 Méthodes et méthodologies d'ingénierie ontologique

Le processus de développement d'une ontologie est un processus complexe où plusieurs acteurs interviennent dans les différentes étapes du processus. Il s'agit donc d'une équipe pluridisciplinaire. Pour cela, il est nécessaire d'utiliser des méthodes ou méthodologies pour seconder le processus de construction des ontologies. Cependant, selon (Corcho, *et al.*, 2003), il n'existe pas une méthodologie parmi celles proposées dans la littérature qui est complètement maturée par rapport aux méthodologies du génie logiciel ou de l'ingénierie des connaissances.

Les méthodes et les méthodologies recensées permettent la construction d'ontologies à partir de zéro (from scratch) c.-à-d. à partir des données brutes ou par réutilisation d'autres ontologies, la ré-ingénierie, l'intégration ou fusion avec d'autres ontologies, la construction collaborative ainsi que l'évolution des ontologies construites.

Jusqu'en 1995, les premières ontologies ont été développées de façon complètement artisanale, sans suivre de méthode prédéfinie. Des premiers projets sont issus des listes de recommandations constituant des ébauches de méthodes, ou cadres méthodologiques. Depuis 1998, on assiste à la naissance de cadres méthodologiques plus élaborés inspirés des méthodes de l'Ingénierie des

connaissances (ex : METHONTOLOGY) ou fondés sur la linguistique (ex : TERMINAE), etc.

2.5.1 Conception manuelle des ontologies

Les méthodes présentées dans cette première section sont celles utilisées pour la construction des ontologies à partir de zéro (from scratch) ou par réutilisation d'autres ontologies où la conception est réalisée manuellement.

2.5.1.1 La méthode d'Uschold et King (Uschold & King, 1995)

Ils ont proposé une première méthode de construction d'ontologie inspirée de leur expérience acquise lors du développement des ontologies dans le domaine de la gestion des entreprises (Enterprise Ontology). La figure 2.3 présente le processus de la méthode. Cette dernière repose sur les quatre étapes suivantes:

***Etape1* :** Identifier le but et la portée de l'ontologie dont les raisons pour lesquelles l'ontologie en cours de construction sont clarifiées ainsi que les utilisateurs potentiels de l'ontologie;

***Etape2* :** Construire l'ontologie. Cette étape est divisée en trois activités qui sont :

- Capture de l'ontologie: identifier les concepts et les relations fondamentaux, produire des définitions précises et non ambiguës à ces éléments en langage naturel, identifier les termes dénotant ces éléments et enfin essayer d'arriver à un agrément. Trois stratégies d'identification des concepts ont été proposées (Uschold & Grüninger, 1996):
  - Approche ascendante (bottom-up strategy) : les concepts les plus spécifiques sont identifiés, par la suite, ils sont généralisés en concepts plus abstraits.
  - Approche descendante (top-down strategy): les concepts les plus abstraits sont identifiés, par la suite, ils sont spécialisés en plus spécifiques.
  - Approche centrifuge (middle-out strategy): les concepts les plus importants sont identifiés (centraux), par la suite, ils sont généralisés et spécialisés comme il est nécessaire.
- Codage de l'ontologie : la représentation explicite de la conceptualisation dans un langage formel ;
- réutiliser et intégrer éventuellement des ontologies existantes. Cette activité peut être effectuée en parallèle avec l'activité de capture et/ou de codage ;

***Etape3* :** évaluer l'ontologie ;

Etape4 : documenter l'ontologie.

L'outil Enterprise Toolset a été développé en suivant cette méthode.

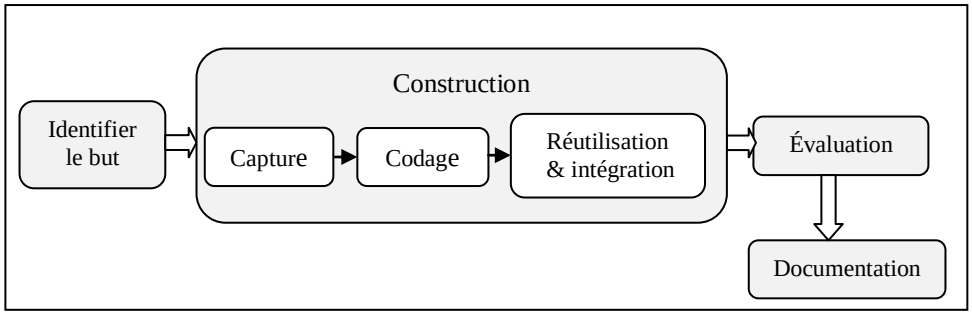

Figure 2.3 Processus de la méthode d'Uschold et King.

2.5.1.2 La méthode de Grüninger et Fox (Grüninger & Fox, 1995)

Cette méthode est basée sur l'expérience du développement de TOVE (*Toronto Virtual Enterprise*) *project Ontology*, qui est une ontologie dans le domaine de la modélisation des activités et des processus d'affaires. La figure 2.4 présente le processus de la méthode. Cette dernière se compose de six étapes :

Etape1 : capturer les scénarios motivants : Selon Grüninger et Fox, le développement d'ontologies est motivé par des scénarios qui se présentent dans l'application. Les scénarios motivants sont des problèmes qui ne sont pas abordés de manière adéquate par les ontologies existantes. Un scénario motivant propose ainsi un ensemble de solutions possibles intuitivement pour les problèmes présentés dans ces scénarios. Ces solutions offrent une sémantique informelle destinée aux objets et aux relations qui seront ultérieurement inclus dans l'ontologie. Toute proposition d'une nouvelle ontologie ou d'une extension d'une ontologie doit décrire un ou plusieurs scénarios motivants ;

Etape2 : formuler les questions de compétence de façon informelle : ils sont basés sur les scénarios obtenus dans l'étape précédente. Une ontologie doit être en mesure de représenter ces questions en utilisant sa terminologie, et être en mesure de caractériser les réponses à ces questions en utilisant les axiomes et les définitions. Ces questions de compétences sont informelles, car elles ne sont pas encore exprimées dans un langage formel d'ontologie;

Etape3 : spécification de la terminologie de l'ontologie en langage formel:

- Obtenir une terminologie informelle : Une fois les questions de compétence informelles disponibles, l'ensemble des termes utilisés pourront être extraits à partir de ces questions. Ces termes serviront de base pour la spécification de la terminologie en un langage formel ;
- Spécification d'une terminologie formelle : La terminologie de l'ontologie est spécifiée en utilisant un formalisme tel que KIF. Ces termes permettront aux définitions et aux contraintes d'être à la suite exprimées par le biais d'axiomes;

Etape4 : formuler les questions de compétence de façon formelle, en utilisant la terminologie de

l'ontologie : Une fois que les questions de compétences ont été posées d'une manière informelle et la terminologie de l'ontologie a été définie, les questions de compétences sont définies formellement ;

Etape5 : spécification des axiomes et des définitions pour les termes de l'ontologie en langage formel : Les axiomes de l'ontologie spécifient les définitions des termes et les contraintes sur leur interprétation. Les axiomes doivent être fournis pour définir la sémantique, ou le sens de ces termes;

Etape6 : établir des conditions pour caractériser la complétude de l'ontologie : Une fois que les questions de compétences ont été formellement décrites, il faut définir les conditions dans lesquelles les réponses à toutes les questions soient complètes.

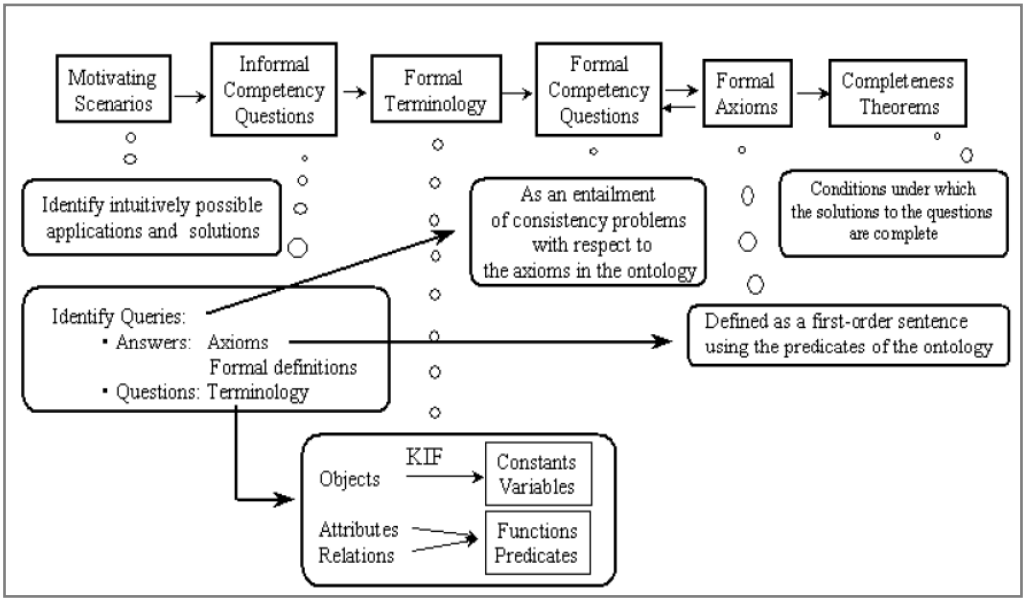

Figure 2.4 Processus de la méthode de Grüninger et Fox**.**

2.5.1.3 La méthode de KACTUS (modelling Knowledge About Complex Technical systems for multiple USe**)** (KACTUS, 1996):

Elle est proposée par Amaya Bernaras et ses collègues dans le cadre du projet KAKTUS. Elle est conditionnée au développement d'une application. Ainsi, chaque fois qu'une application est construite, l'ontologie qui représente les connaissances nécessaires pour l'application est aussi construite. En suivant cette méthode, l'ontologie peut être développée en réutilisant d'autres ontologies existantes et peut aussi être intégrée dans des ontologies d'applications futures. Elle repose sur trois étapes :

Etape 1 : Spécifier l'application basée sur l'ontologie en particulier les termes à collecter et les tâches à effectuer en utilisant cette ontologie;

Etape 2 : Organiser les termes en utilisant les méta-catégories (concept, relation, attribut, etc.) et

importer, adapter et étendre les ontologies existantes et pertinentes ;

Etape 3 : Affiner l'ontologie et la structurer selon des principes de modularisation et organisation hiérarchique.

2.5.1.4 La méthode basée sur SENSUS (Swartout, *et al.*, 1997)

C'est une méthode qui consiste à construire le squelette d'une ontologie de domaine (arbre conceptuel de l'ontologie) à partir d'une plus grande ontologie, l'ontologie SENSUS. La figure 2.5 présente le processus de la méthode. Cette dernière propose de relier les termes spécifiques du domaine à SENSUS et de tailler dans SENSUS, ceux qui ne sont pas pertinents dans la nouvelle ontologie qu'on souhaite construire. Le résultat de ce processus est le squelette de cette future ontologie, qui est générée automatiquement en utilisant le processus décrit au dessous et l'éditeur OntoSaurus. Conformément à cette méthode, pour construire une ontologie dans un domaine spécifique, il faut suivre les étapes suivantes :

Étape 1*:* identifier les termes clés du domaine ;

Etape 2 : relier manuellement les termes clés à SENSUS ;

Étape 3: inclure tous les concepts qui se trouvent sur le chemin depuis le terme clé jusqu'à la racine de SENSUS ;

Étape 4: ajouter les nouveaux termes de domaine. Dans cette étape, on ajoute manuellement tous les termes qui sont relevants pour le domaine et qui ne sont pas encore apparus. Les étapes 2 et 3 sont répétées pour inclure les concepts qui se trouvent dans le chemin depuis les nouveaux termes jusqu'à la racine de SENSUS ;

Étape 5: Ajouter le sous arbre entier. Dans cette étape, celui qui fait l'ontologie doit faire attention aux nœuds qui ont un grand nombre de chemins à travers eux dans le nouvel arbre généré. Pour les sous arbres au dessous de ces nœuds, c'est l'ontologiste qui décide l'ajout ou la suppression d'un nœud.

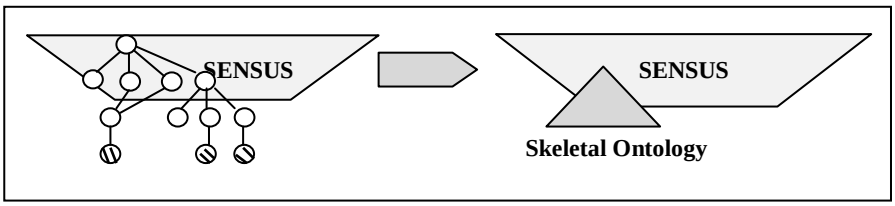

Figure 2.5 Processus de la méthode de SENSUS.

2.5.1.5 La méthode METHONTOLOGY (Fernández-López, *et al.*, 1997)

Cette méthode a été développée au sein du groupe d'ontologie à l'université polytechnique de Madrid. La figure 2.6 présente le processus de la méthode. Cette dernière comprend deux étapes :

Etape1 : identification du processus de développement de l'ontologie avec

- activités de gestion de projet (prévision, contrôle, assurance qualité) ;
- activités orientées-développement (spécification, conceptualisation, formalisation, implémentation, maintenance) ;
- et des activités du support (acquisition de connaissances, intégration, évaluation, documentation, gestion de la configuration).

Etape2 : cycle de vie de l'ontologie basé sur des prototypes évolutifs.

Cette méthodologie attire l'attention pour les raisons suivantes :

- la complétude de la méthode : les activités intégrées dans le processus de développement de la méthode sont complètes. Ainsi, elle présente un certain nombre de phases spécifiées de manière très détaillée entre autre la phase de conceptualisation.
- La phase d'évaluation de la méthode qui permet de s'assurer que l'ontologie créée est correctement construite et elle modélise réellement le vrai monde pour lequel elle a été créée. Pour ce faire, une liste de critères de vérification et de validation de l'ontologie a été identifiée.
- La conformité aux activités du standard IEEE pour le développement des logiciels.
- Le cycle de vie de l'ontologie basé sur des prototypes évolutifs permet à l'ontologiste de faire un retour arrière de n'importe quel état à un autre si quelques définitions sont incomplètes ou incorrectes. Donc, ce cycle de vie permet l'inclusion, la suppression ou la modification des définitions de l'ontologie à tout moment.

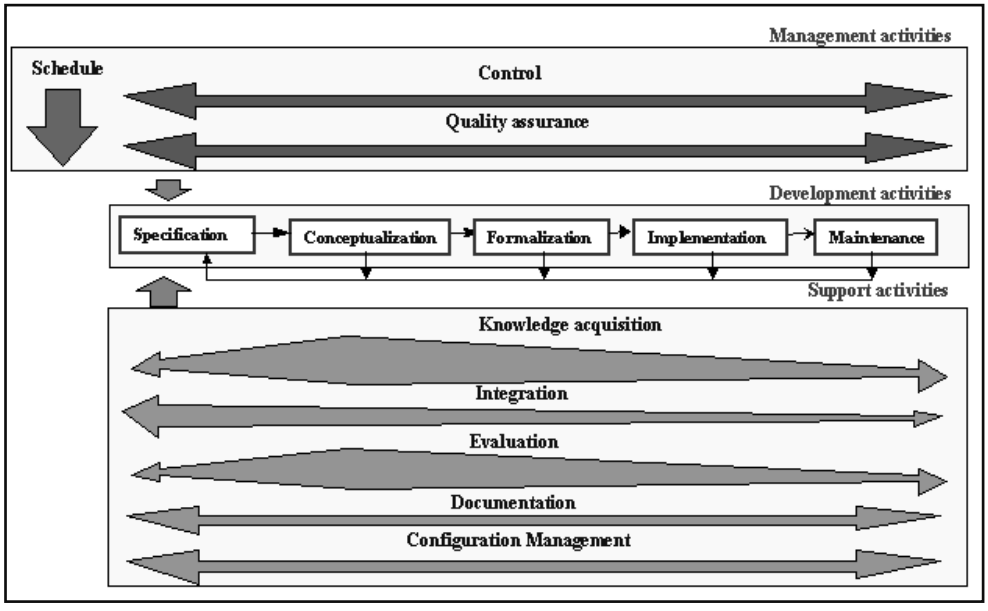

Figure 2.6 Processus de développement et le cycle de vie de METHONTOLOGY.

Plusieurs ontologies ont été créées à partir de cette méthodologie, entre autre, Chemical ontology, qui contient des connaissances dans le domaine de la chimie.

Les outils ODE et WeODE ont été construits pour donner un support technique à METHONTOLOGY. D'autres outils peuvent aussi être utilisés pour construire une ontologie en suivant cette méthodologie, par exemple, PROÉTGÉ, OntoEdit, KAON, etc.

2.5.1.6 La méthodologie ON-TO-KNOWLEDGE (Staab, *et al.*, 2001)

La méthodologie On-To-Knowledge propose de construire une ontologie en tenant compte de comment l'ontologie va être utilisée par l'application plus tard. Par conséquent, les ontologies développées en suivant cette méthodologie sont très dépendantes de l'application. La figure 2.7 présente le processus de la méthodologie. Cette dernière se présente en cinq étapes. La première étape porte sur l'identification du problème à résoudre alors que les quatre dernières portent sur le développement de l'ontologie:

Étape 1: Étude de faisabilité : Conformément à On-To-Knowledge, cette étude est appliquée à l'application entière et, donc, peut être effectuée avant le développement de l'ontologie. Elle permet d'identifier le problème, les opportunités et les solutions potentielles. En fait, cette étude sert de base à l'étape de Kickoff ;

Etape 2 : la phase Kickoff : le résultat de cette étape est un document de spécification des besoins de l'ontologie qui décrit :

- Le domaine, la portée et l'objectif de l'ontologie ;
- Les directives de conception (telles que les conventions de nommage) ;
- Les sources d'informations disponibles (livres, interviews…etc.) ;
- Les utilisateurs potentiels et les cas d'utilisations ainsi que les applications supportées par l'ontologie.

Les questions de compétences peuvent être utiles à l'élaboration du document de spécification de besoins. La spécification de besoin devrait permettre au concepteur d'ontologie de décider en ce qui concerne l'inclusion ou l'exclusion des concepts dans l'ontologie et de leur structure hiérarchique. En fait, cette spécification est utile pour élaborer la version initiale 'baseline taxonomy' contenant peu mais d'importants termes.

Dans cette étape les développeurs devraient chercher des ontologies existantes qui sont potentiellement réutilisables.

Étape 3: Raffinement :

L'objectif de cette étape est de produire une ontologie orientée application conformément aux spécifications données à l'étape de kickoff. L'étape de raffinement est divisée en deux activités :

- Raffiner la taxonomie de base obtenue à l'étape précédente auprès des experts du domaine. Une fois cette activité est effectuée, les axiomes sont identifiés et modélisés ;
- Implémenter l'ontologie dans un langage d'ontologie. Un tel langage est sélectionné selon

les besoins spécifiques de l'application envisagée.

Étape 4: Évaluation

Cette étape sert à prouver l'utilité du développement de l'ontologie et les applications associées. Deux activités doivent être effectuées :

- Contrôler si l'ontologie satisfait le document de spécification des besoins et si elle répond aux questions de compétences.
- Tester l'ontologie dans le cadre de son environnement d'application.

Cette étape d'évaluation est fortement liée à celui de raffinement. En effet plusieurs allers retours sont nécessaires avant d'atteindre le niveau de satisfaction souhaité.

Étape 5: Maintenance de l'ontologie.

Un éditeur d'ontologies est associé à cette méthodologie, il s'agit d'OntoEdit.

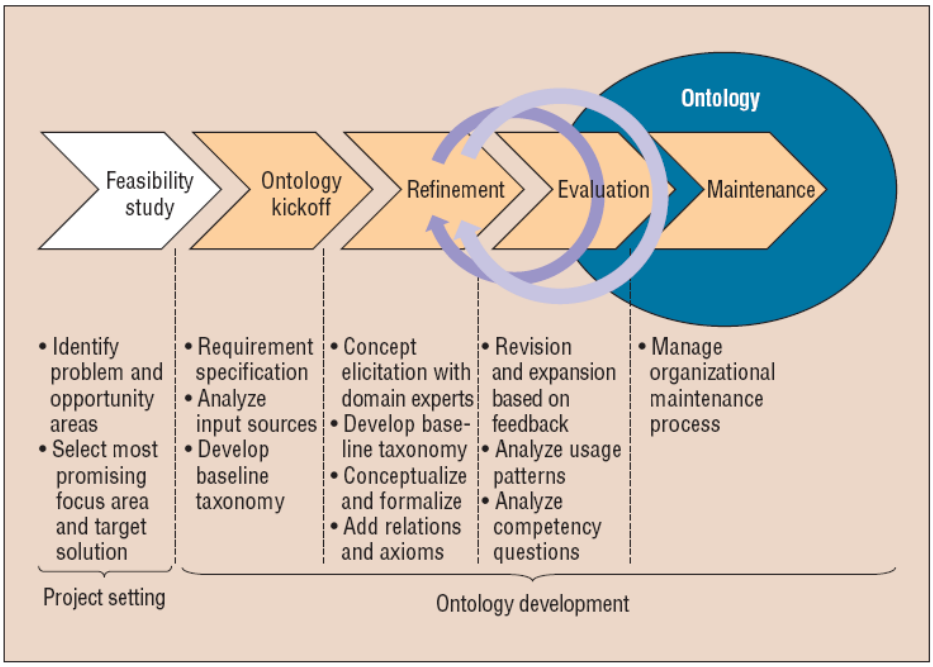

Figure 2.7 Processus de la méthodologie On-To-Knowledge.

2.5.2 Conception des ontologies à l'aide des outils de traitement de langue (Natural Langage Processing tools)

Il existe un autre axe de recherche, concentré sur les méthodes, faisant appel à d'autres disciplines, qui permettent la construction semi-automatique d'ontologies à l'aide des outils de traitement automatique de textes (NLP tools). Ces méthodes focalisent sur l'étape d'acquisition automatique afin de minimiser le coût et l'effort du travail manuel. Ainsi, Ces outils permettent

d'extraire les termes candidats d'un domaine et leurs relations à partir d'un corpus[4] textuel. Néanmoins, cela nécessite l'intervention d'un expert du domaine pour la sélection et la validation des termes obtenus.

Au ce sujet, on peut citer à titre d'exemple la méthode ARCHONTE et le travail d'Audrey Baneyx basé sur cette dernière ainsi que TERMINAE. Nous décrivons ces méthodes dans cette deuxième section.

2.5.2.1 La méthode ARCHONTE

La méthode ARCHONTE (ARCHitecture for ONTological Elaborating) a été mise au point par B. Bachimont, au sein du groupe Terminologie et Intelligence Artificielle, pour construire des ontologies s'appuie sur la sémantique différentielle[5] (Bachimont, 2000) (Bachimont, *et al.*, 2002). La figure 2.8 présente le processus de la méthode. La construction d'une ontologie comporte trois étapes:

Etape1 : normalisation

Choisir les termes pertinents du domaine et normaliser leur sens, à partir d'un corpus textuel qui est la source privilégiée permettant de caractériser les notions utiles à la modélisation ontologique et le contenu sémantique qui leur est associé, 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;

Les principes différentiels sont (Bachimont, 2000) (Bachimont, *et al.*, 2002):

- *le principe de communauté avec le père :* il faut expliciter en quoi le fils est identique au père qui le subsume ;
- *le principe de différence avec le père :* il faut expliciter en quoi le fils est différent du père qui le subsume. Puis que le fils existe, c'est donc qu'il est distinct du père ;
- *le principe de différence avec les frères :* il faut expliciter la différence de la notion considérée avec chacune des notions sœurs car toute notion doit se distinguer de ses sœurs si non il n'y aurait pas lieu de la définir ;
- *Le principe de communauté avec les frères:* il faut expliciter la communauté existante entre la notion considérée et chacune des notions sœurs. Ce principe de communauté doit être différent du principe de communauté existant avec le parent. La communauté entre

[4] Est un ensemble de textes extraits à partit des documents attestés dans la pratique d'un domaine, qui doit être représentatif du domaine (complet), préparé à être traité par un ordinateur et accepté par les experts du domaine.
[5] Elle permet d'attribuer un sens aux termes grâce à la définition de traits sémantiques génériques et spécifiques (les principes différentiels).

les notions filles doit permettre de définir des différences mutuellement exclusives entre les notions filles.

Ces principes différentiels permettent de fixer le cadre interprétatif des concepts. Cela revient à associer aux termes une signification qui fasse abstraction des variations de sens liées aux différents contextes textuels dans lesquels ils peuvent apparaître. Les concepts sont donc normés puisqu'ils sont décrits selon un certain point de vue, en l'occurrence celui de la tâche à réaliser. Par conséquent, cela permet de passer à «l'ontologie différentielle».

Etape2 : formalisation

Formaliser les connaissances, ce qui implique par exemple d'ajouter des propriétés à des concepts, des axiomes, de contraindre les domaines d'une relation… En d'autres termes, il s'agit de définir des concepts selon une sémantique formelle et extensionnelle (c.à.d. les concepts sont liés à un ensemble de référents dans le monde) et de formaliser les relations qui existent entre les concepts en définissant leur arité et les ensembles d'extensions de concepts qu'elles relient. Il faut passer de la dimension linguistique et interprétative de la taxinomie à «l'ontologie référentielle» ou «l'ontologie formelle» composée de concepts dont le sens est décontextualisé.

Etape3 : opérationalisation

L'opérationnalisation dans un langage de représentation des connaissances. Cette étape marque le passage à « l'ontologie computationnelle ».

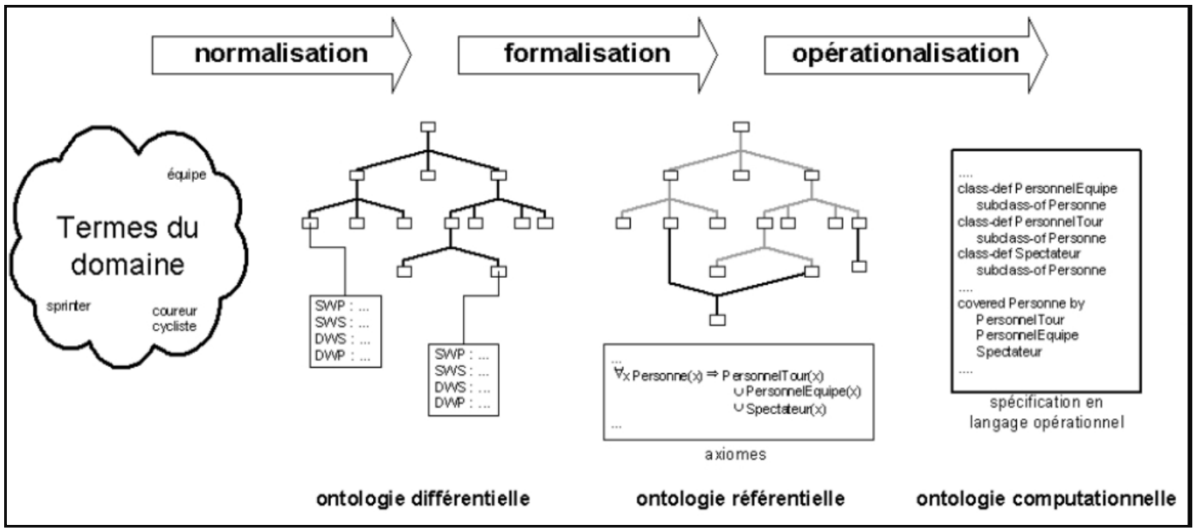

Figure 2.8 Les trois étapes d'ARCHONTE telles que proposées par B. Bachimont.

Une extension de la méthode ARCHONTE proposée par Audrey Baneyx, comme exprimé à la figure2.9, se trouve dans (Baneyx, 2007). Elle a bien précisé les trois étapes d'ARCHONTE et d'en ajouter une première consacrée à la constitution du corpus des connaissances et à son analyse par des outils de traitement automatique du langage.

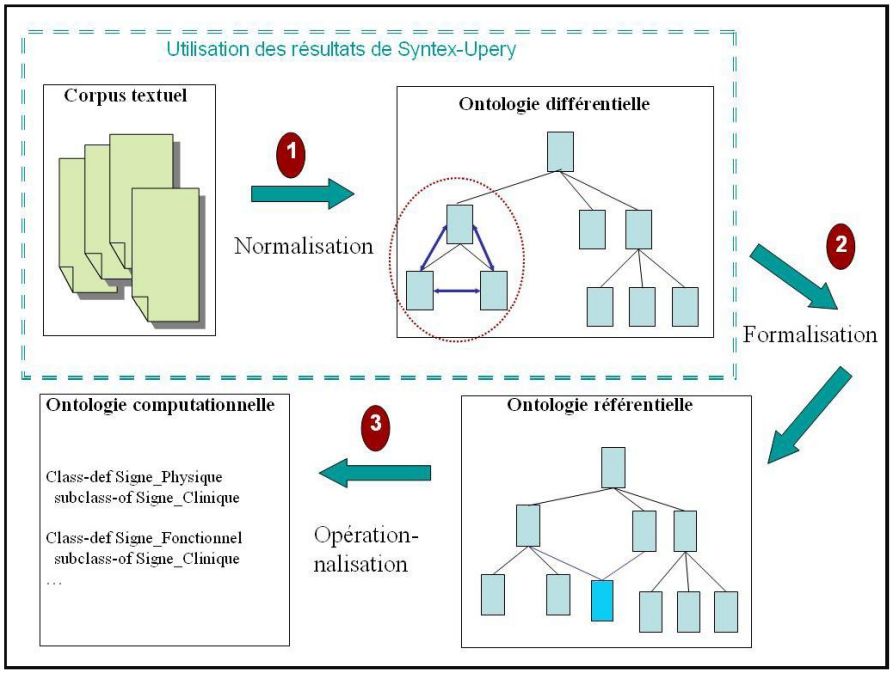

Figure 2.9 Une autre vue des étapes d'ARCHONTE d'après (Baneyx, 2007).

2.5.2.2 La méthode TERMINAE (Aussenac-Gilles, *et al.*, 2000)

La méthode TERMINAE est une méthode qui repose sur l'analyse de corpus linguistique. Elle utilise pour cela, comme nous avons dit précédemment, des outils de traitement automatique des langues analysant les termes de textes et les relations lexicales. Les termes sont regroupés suivant leur contexte et facilitent la création de concepts et de relations sémantiques. Les concepts et relations sont ensuite formalisés dans un modèle. Telle qu'illustrée à la figure 2.10, cette méthode est composée de quatre étapes :

Etape1 : la construction du corpus

Les textes sont sélectionnés à partir des différents documents techniques disponibles. Ces derniers sont récoltés sous l'intervention d'un expert du domaine. Le résultat de cette activité est un corpus sur lequel les outils de traitement automatique de langues seront réalisés. Le corpus doit être complet dans le sens où il doit couvrir entièrement le domaine traité par l'application.

Etape2 : l'analyse linguistique

Elle consiste à la sélection des outils linguistiques adéquats et les appliquer au corpus obtenu précédemment afin d'extraire les termes et leurs relations lexicales et syntaxiques.

Les outils utilisés nécessitent l'intervention d'experts du domaine afin de sélectionner et de valider les candidats. A la fin de cette étape, un ensemble de termes, de relations lexicales entre ces termes et de regroupements est obtenu,

Etape4 : La normalisation

Elle vise à conceptualiser les résultats de l'étape précédente. Elle est divisée en deux étapes :

- Étape linguistique : les termes à conserver sont sélectionnés en fonction de leur contexte et définis à partir d'une définition en langage naturel.
- Étape conceptuelle : les concepts sont ensuite identifiés ainsi que les relations sémantiques entre eux. Ils sont représentés sous forme d'un réseau sémantique.

Etape4 : la formalisation

Le réseau sémantique précédemment obtenu est traduit dans un langage formel.

Ces étapes sont précédées généralement par une première qui consiste en la description des besoins (utilisation de l'ontologie, connaissance à représenter…).

Un outil porte le même nom que la méthode est associé à cette dernière. Il a été développé au LIPN de l'Université Paris-Nord2. Il offre un support méthodologique qui permet de faire évoluer progressivement une ontologie en conservant des liens entre les textes et les niveaux linguistiques et conceptuels. Le modèle de représentation de l'outil TERMINAE est celui des Logiques de description mais une traduction des ontologies dans le langage OWL est possible.

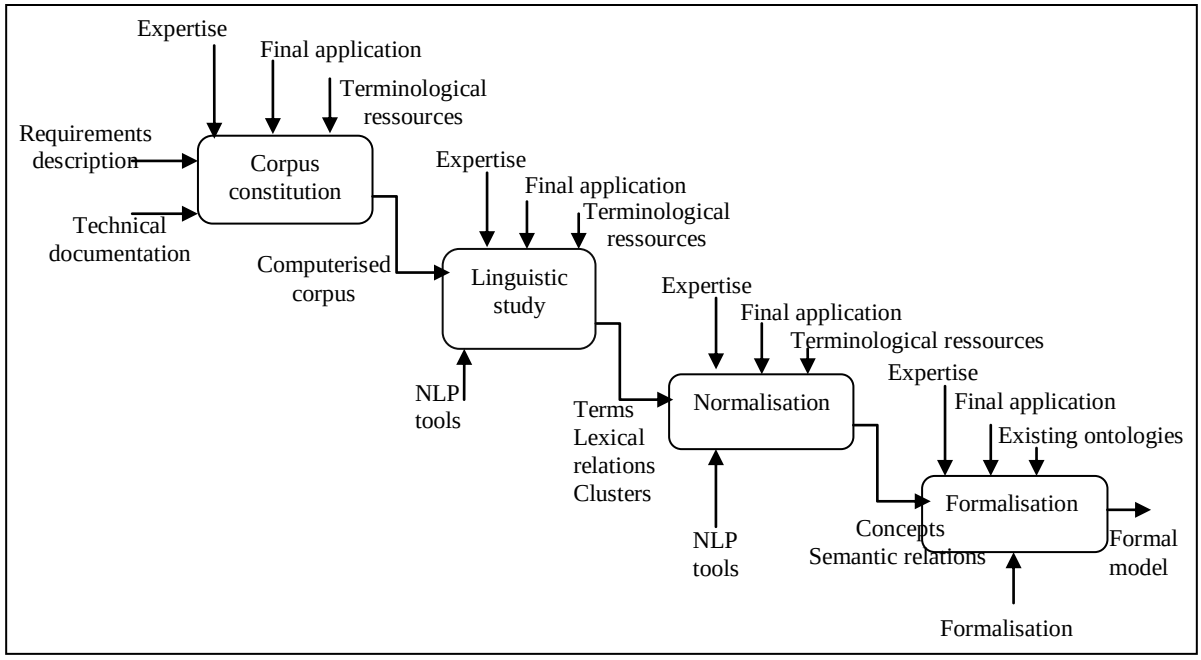

Figure 2.10 Les étapes de la méthode TERMINAE.

Le choix des outils de traitement de langue est laissé à l'utilisateur. Les auteurs de la méthode ARCHONTE par exemple, proposent d'utiliser l'outil Syntex-Upery.

2.6 Principes d'ingénierie ontologique

Malgré qu'il existe un ensemble de principes et critères de conception qui ont fait leurs preuves dans le développement des ontologies, ils sont souvent oubliés lors de la construction de ces dernières. Parmi celle-ci nous présentons celles qui se sont avérées efficaces dans l'ingénierie

ontologique :

- Clarté/Objectivité (Gruber, 1993b);
- Complétude/ Perfection (Gruber, 1993b);
- Cohérence (Gruber, 1993b);
- Extensibilité monotone maximale (Gruber, 1993b);
- Biais de codage minimal (Gruber, 1993b) ;
- Engagements ontologiques minimaux (Gruber, 1993b) ;
- Distinction ontologique (Borgo, *et al.*, 1996) ;
- Modularité (Bernaras, *et al.*, 1996);
- Distance sémantique minimale (Arpírez, *et al.*, 1998) ;
- Normaliser les noms (Arpírez, *et al.*, 1998) ;
- Diversification des hiérarchies (Arpírez, *et al.*, 1998).

Selon Gruber, les principes d'ingénierie ontologique fournissent des critères objectifs qui permettent de guider et d'évaluer l'ingénierie ontologique.

2.7 Mécanismes de représentation des connaissances

Représenter des connaissances propres à un domaine consiste à décrire et à coder les éléments de ce domaine pour qu'une machine puisse les manipuler afin de raisonner (Kayser, 1997). Il existe un certain nombre de formalismes de représentation de connaissances ; ceux qui ont été les plus utilisés pour représenter les ontologies sont :

- Les frames ;
- Les graphes conceptuels ;
- Et les logiques de descriptions.

2.7.1 Frames

Le modèle des Frames (introduit dés les années 70 par Minsky) est un classique de l'Intelligence Artificielle et a été initialement proposé comme langage de représentation d'ontologies par T. Gruber (Gruber, 1993a). 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) (Sowa, 1984).

2.7.2 Graphes conceptuels

Le modèle des Graphes Conceptuels (GC) est un formalisme de représentation de connaissances de type **réseau sémantique**, fondé sur la définition de concepts et de relations entre concepts. Il a été introduit par *John F. SOWA* en 1984. Ce formalisme emploie la représentation graphique comme méthode pour modéliser les connaissances.

Le modèle des graphes conceptuels se décompose en deux parties (Kayser, 1997):

-Une partie terminologique

Dédiée au vocabulaire conceptuel des connaissances à représenter (le support), c'est-à-dire les types de concepts, les types de relations et les instances des types de concepts. Le terme concept est utilisé dans ce formalisme pour désigner les instances des types de concepts, qui correspondent à la notion classique de concept. Cette partie correspond à la représentation du modèle conceptuel mais intègre également des connaissances sur la hiérarchisation des types de concepts et de relations.

-Une partie assertionnelle

Dédiée à la représentation des assertions du domaine de connaissances étudié. Les faits sont représentés sous la forme de graphes particuliers (les graphes conceptuels). Ces derniers sont construits à partir du vocabulaire conceptuel du niveau terminologique.

La description de la partie suivante est essentiellement tirée de (Fürst, 2004).

a) Le niveau terminologique (le support)

Un support comprend des ensembles *des types de concepts* Tc et *des types de relations* Tr ainsi qu'un *ensemble des marqueurs individuels M*.

-Les types de concepts Tc

Un type de concepts encapsule les caractéristiques communes à plusieurs concepts. Les concepts sont des instances de leur type de concepts. Une relation de spécialisation, notée $\leq_c$, est définie sur Tc et correspond sémantiquement à la notion « sorte de » : soient deux types t_c et $t'c$ de Tc, si $tc \leq_c t'c$ alors $t'c$ est dit sur-type de tc et tc est dit sous-type de $t'c$. Par exemple, le type de concepts Chat peut être vu comme un sous-type du type de concepts *Animal*.

Deux types de concepts particuliers sont distingués :

- Le type de concepts **Universel**, noté $\top$, qui encapsule tous les autres types. Il est donc sur-type de tous les types de concepts de Tc ;
- Le type de concepts **Absurde**, noté $\bot$, qui représente le type de concepts plus spécifique que tous les autres types de concepts. $\bot$ ne possède pas d'instance, est sous-type de tous ceux de Tc.

Un exemple de hiérarchie de types de concepts est présenté dans la figure 2.11 où Les arêtes représentent la relation $\leq_c$, les types de concepts les plus généraux étant placés au-dessus des types plus spécifiques.

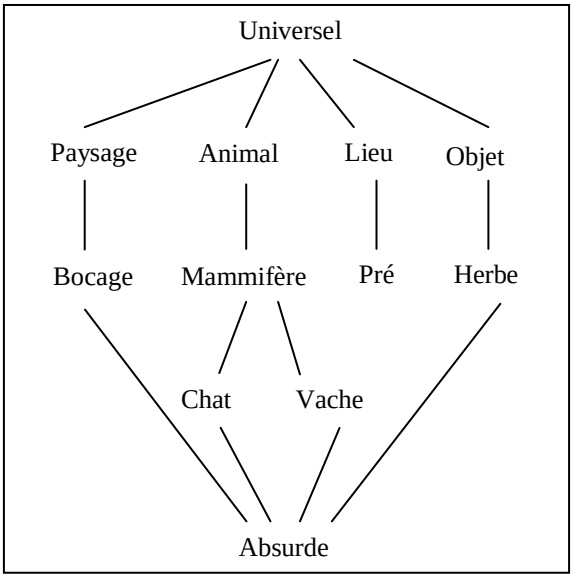

Figure 2.11 Exemple d'une hiérarchie de types de concepts (Fürst, 2004).

-Les types de relations

L'ensemble des **types de relations** est noté Tr. Ces types de relations représentent les relations pouvant exister entre les différents concepts, c'est pourquoi à chaque type de relations est associée une signature qui spécifie les types de concepts définis dans Tc. Formellement, on définit une application $\sigma : Tr \rightarrow T_c^n$ qui, à chaque type de relations, associe sa signature, c'est-à-dire le n-uplet de types de concepts les plus généraux pouvant être liés par ce type de relations. L'arité de la signature est également l'arité du type de relations. Par exemple, le type de relations broute est d'arité 2 et sa signature est (Animal; Herbe). Les types de relations sont également hiérarchisés par une relation de spécialisation notée $\leq r$. Deux types de relations d'arités différentes ne sont cependant pas comparables par $\leq r$.

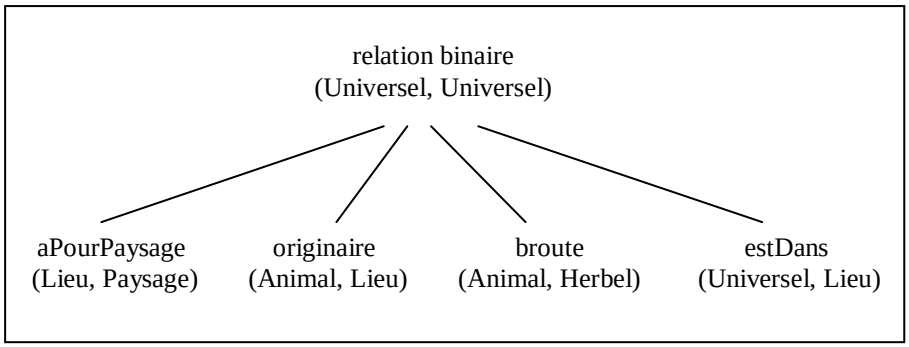

Figure 2.12 Exemple d'une hiérarchie de types de relations (Fürst, 2004).

La figure 2.12 présente un exemple de hiérarchie de relations, avec leurs signatures, basé sur la hiérarchie de types de concepts de la figure 2.11.

-Les marqueurs individuels

L'ensemble des **marqueurs individuels** est noté *M*. Ces marqueurs permettent d'identifier les concepts, c'est-à-dire les instances des types de concepts. Par exemple, la vache Marguerite sera identifiée par le marqueur Marguerite associé au type de concepts *Vache*. Sur l'ensemble M est définie une application $\tau : M \rightarrow Tc$ qui à chaque marqueur associe le type le plus spécifique de l'instance qu'il désigne. Le type $\perp$ n'a bien évidemment aucun antécédent par τ .

Donc :

Le **support** S = (Tc; Tr; M). Les 3 ensembles Tc, Tr et M étant disjoints et les ensembles Tc et Tr étant partiellement ordonnés.

b) Le niveau assertionnel (le graphe conceptuel)

Au niveau assertionnel, les faits sont représentés en utilisant le vocabulaire décrit dans le support. Ces connaissances sont présentées sous forme de graphes d'un type particulier appelés **graphes conceptuels**. Un graphe conceptuel est un multi-graphe fini, non-orienté et biparti, composé:

- d'un ensemble de *sommets concepts* (représentés par des rectangles) où chaque sommet concept possède un type de concepts et un marqueur individuel ;
- d'un ensemble de *sommets relations* qui expriment la nature des liens entre les concepts (représentés par des ellipses) ;
- d'un ensemble d'*arêtes* qui lient les sommets relations aux sommets concepts ;
- *et* d'une *étiquette* associée à chaque sommet et à chaque arête (un couple *(type de concept, marqueur)* pour un sommet concept, un type de relation pour un sommet relation, le numéro d'ordre dans le voisinage d'un sommet relation donné pour une arête).

Un graphe conceptuel est forcément défini sur un support donné.

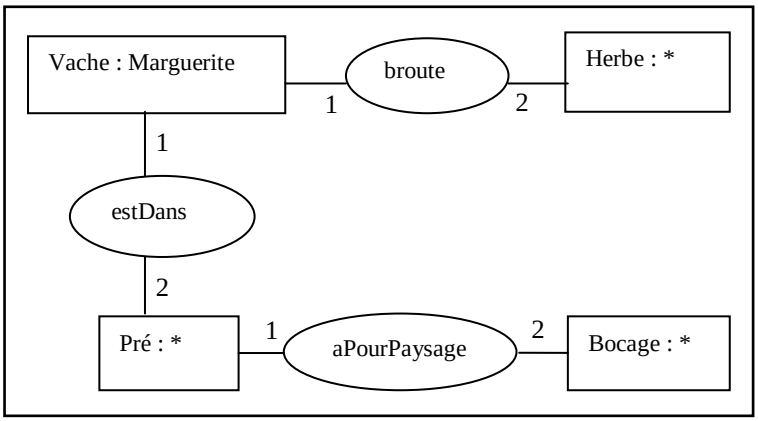

Figure 2.13 Exemple d'un graphe conceptuel (Fürst, 2004).

La figure 2.13 illustre un exemple d'un graphe conceptuel. Ce dernier peut être interprété comme : *« le mammifère de type vache Marguerite broute de l'herbe dans un pré d'une région de bocage»*.

c) Sémantique logique des graphes conceptuels

A tout *type* de concept ou de relation, on associe un *prédicat* de même nom : prédicat unaire pour les types de concepts, pour les types de relation prédicat ayant la même arité que le type. A tout *marqueur individuel*, on associe une *constante* de même nom.

L'interprétation logique d'un support *S* est un ensemble de formules Φ*(S)* traduisant les liens *sorte de* entre types de concepts et entre types de relations. Ainsi :

(1) Pour tous t1 et t2 de TC (ensemble des types de concepts), tels que t1 couvre t2 dans TC (c'est-à-dire t2 < *t*1 *e*t il n'existe pas de type t3, tel que t2 < *t*3 < *t*1), on considère la formule :

$\forall\ x\ t2(x) \rightarrow t1(x)$, qui correspond à l'interprétation des liens *Sorte de* entre types de concepts (2) Pour tous *tr*1 et *tr*2 de *TRp* (*p* étant l'arité) tels que *tr*1 couvre *tr*2 dans *TR* (ensemble des types de relations), on considère la formule : $\forall\ x1\ \ldots\ \forall\ xp\ tr2(x1\ \ldots\ xp) \rightarrow tr1(x1\ \ldots\ xp)$, qui correspond à l'interprétation des liens *Sorte de* entre types de relations.

A tout graphe conceptuel *G*, on associe une formule de la logique du premier ordre, notée Φ(*G*). A tout sommet concept, on associe un terme, qui est une variable si c'est un sommet générique, et une constante est associée à son marqueur individuel sinon. L'ensemble des variables est en bijection avec l'ensemble des sommets concepts génériques. L'ensemble des constantes est en bijection avec l'ensemble des marqueurs individuels (ainsi, à des sommets ayant même marqueur individuel est associée la même constante). Ensuite, on construit Φ(*G*) de la façon suivante :

A un sommet concept *c*, on associe l'atome *tc(idc)*, où *tc* est le prédicat associé au type de *c*, et *idc* est le terme associé à *c*. A un sommet relation *r*, on associe l'atome *tr(idc*1 *... idcp)*, où *tr* est le prédicat associé au type de *r*, *p* est l'arité de ce prédicat, et *idci* désigne le terme associé au *ième* voisin de *r* dans *G*. On obtient Φ*(G)* en faisant la conjonction de ces atomes, puis en fermant existentiellement la formule.

d) Mécanisme de raisonnement

La **projection** est le mécanisme de base dans les graphes conceptuels. Elle permet de déterminer si un graphe est plus spécialisé, ou plus général, qu'un autre. La projection d'un graphe G_1 dans un graphe G_2, tous deux associés à un support S est la donnée de deux applications F_r, de l'ensemble des sommets relations de G_1 dans l'ensemble des sommets relations de G_2 et F_c, de l'ensemble des sommets concepts de G_1 dans l'ensemble des sommets concepts de G_2.

Il existe un certain nombre d'extensions qui ont été proposées dans plusieurs travaux afin

d'accroître les possibilités de représentation et l'expressivité du modèle de base des GCs.

2.7.3 Logiques de descriptions

Les logiques de descriptions LDs (Description Logic DL en anglais), appelées parfois logiques terminologiques, sont des langages formels conçus pour décrire et raisonner sur les connaissances d'un domaine. Elles ont été introduites par Brachman en 1979, par la suite elles ont connues de nombreux développements. Elles sont issues de la logique des prédicats, des frames et des réseaux sémantiques (des correspondances existent entre ces logiques et ces formalismes).

Ils existent de nombreuses logiques de descriptions qui se distinguent par la richesse des constructeurs qu'elles proposent à la représentation des connaissances souhaitées. La logique nommée *AL* (Attributive Language) qui a été introduite par Schmib et Smolka en 1991 (Schmidt-Schauß & Smolka, 1991) est l'une parmi elles. Cette logique est minimale, dans le sens où elle est la moins expressive (c.-à-d. propose le moins de constructeurs). Nous décrivons par la suite la syntaxe est la sémantique de la logique AL.

a) La syntaxe d'*AL*

La grammaire d'AL est donnée par (tableau 2.1):

C, D $\rightarrow$ A (concept atomique) | T (Le concept universel) | $\perp$ (Le concept le plus spécifique) | $\neg$A (la négation atomique) | C $\cap$ D (l'intersection) | $\exists$R. T (Restriction existentielle limitée) | $\forall$ R.C (restriction universelle complète)

Tableau 2.1 Syntaxe de la logique *AL*.

Où : A est concept atomique, C, D sont des concepts composés et R est un rôle atomique (AL ne permet pas la spécification de rôles à l'aide de constructeurs : rôles composés).

-Le constructeur $\neg$A : est utilisé pour évoquer la négation qui ne peut être appliquée qu'à un concept atomique, c'est-à-dire les individus pour une interprétation qui n'appartiennent pas au concept atomique A.

-Le constructeur C $\cap$ D : permet de faire la conjonction de deux concepts composés, ce qui représente l'ensemble des individus appartenant à la fois au concept C et au concept D pour une interprétation.

-Le quantificateur existentiel $\exists$R. T : désigne l'ensemble des individus, membres du domaine du rôle R pour une interprétation donnée.

-Le quantificateur universel $\forall$ R.C : désigne l'ensemble des individus du domaine du rôle R qui sont

en relation, par le biais de R, qu'avec les individus du concept C, pour une interprétation donnée.

Les concepts et les rôles (c'est-à-dire des relations entre concepts) atomiques constituent les entités élémentaires d'une T-box (voir le tableau 2.3). Ces derniers peuvent être combinés au moyen de constructeurs pour former des concepts et des rôles composés. Par exemple, le concept composé Male∩Femelle est le résultat de l'utilisation du constructeur $\cap$ sur les concepts atomiques Male et Femelle. L'interprétation du concept ainsi composé est « l'ensemble des individus qui appartiennent à la fois au concept Male et au concept Femelle ».

b) La sémantique formelle d'*AL*

La sémantique d'une LD est donnée au moyen d'une interprétation ^{I} qui est un couple (Δ^I , ^{I}) où :

Δ^I : est le domaine d'interprétation. C'est un ensemble non vide d'individus.

^{I} : est la fonction d'interprétation qui fait correspondre à un concept atomique A un ensemble A^I tel que $A^I \subseteq \Delta^I$ et à chaque rôle atomique R une relation binaire $R^I \subseteq \Delta^I \times \Delta^I$.

La sémantique de AL définit plus haut est donnée par la table suivante (tableau 2.2):

$T^I = \Delta^I$
$\bot^I = \Phi$
$(\neg A)^I = \Delta^I \setminus A^I$
$(C \cap D)^I = C^I \cap D^I$
$(\forall R.C)^I = \{a \in \Delta^I \mid \forall b$, if $(a,b) \in R^I$ then $b \in C^I\}$
$(\exists R.T)^I = \{a \in \Delta^I \mid \exists b . (a,b) \in R^I\}$

Tableau 2.2 Sémantique de la logique *AL*.

La logique de description *AL* peut être enrichie par les constructeurs suivants:

- *O* : qui permet la description de concepts par l'énumération d'individus nommés ;
- *U* : désigne l'union de concepts ;
- *E* : la quantification existentielle complète ;
- *C* : la négation complète ;
- *I* : les rôles inverses ;
- *H* : l'inclusion entre rôles ;
- Les constructeurs *F, Q* et *N* sont trois variantes de la contrainte de cardinalité sur le rôle.

La figure 2.14 donne une vue d'ensemble de constructeurs enrichissant *AL* avec leur syntaxe et leur sémantique.

$[\mathcal{O}]$	$\{a_1, a_2, ..., a_n\}$	$\{a_1^{\mathcal{I}}, a_2^{\mathcal{I}}, ..., a_n^{\mathcal{I}}\}$
$[\mathcal{U}]$	$C \sqcup D$	$C^{\mathcal{I}} \cup D^{\mathcal{I}}$
$[\mathcal{E}]$	$\exists R.C$	$\{a \in \triangle^{\mathcal{I}} \mid \{\exists b.(a,b) \in R^{\mathcal{I}}\} \wedge b \in C^{\mathcal{I}}\}$
$[\mathcal{C}]$	$\neg C$	$\triangle^{\mathcal{I}} \backslash C^{\mathcal{I}}$
$[\mathcal{I}]$	R_1^{-1}	$\{(y,x) \mid (x,y) \in R_1^{\mathcal{I}}\}$
$[\mathcal{H}]$	$R_1 \sqsubseteq R_2$	$R_1^{\mathcal{I}} \subseteq R_2^{\mathcal{I}}$
$[\mathcal{F}]$	$= 1R$	$\{x \in \triangle^{\mathcal{I}} \mid \mid \{y \in \triangle^{\mathcal{I}} \mid (x,y) \in R^{\mathcal{I}}\}\mid = 1\}$
	$\geq 2R$	$\{x \in \triangle^{\mathcal{I}} \mid \mid \{y \in \triangle^{\mathcal{I}} \mid (x,y) \in R^{\mathcal{I}}\}\mid \geq 2\}$
$[\mathcal{N}]$	$\geq nR$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}}\}\mid \geq n\}$
	$\leq nR$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}}\}\mid \leq n\}$
	$= nR$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}}\}\mid = n\}$
$[\mathcal{Q}]$	$\geq nR.C$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\mid \geq n\}$
	$\leq nR.C$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\mid \leq n\}$
	$= nR.C$	$\{a,b \in \triangle^{\mathcal{I}} \mid \mid (a,b) \in R^{\mathcal{I}} \wedge b \in C^{\mathcal{I}}\}\mid = n\}$

Figure 2.14 Exemple de constructeurs de rôles et concepts pour étendre *AL* (Fournier-Viger, 2005). La première colonne contient la lettre qui désigne le constructeur, la deuxième sa syntaxe d'utilisation et la dernière sa sémantique.

Il est important de noter qu'il existe une autre façon d'étendre une LD. La spécification d'un ensemble de rôles transitifs N_{R+}, constitue une extension par ajout de contraintes sur l'interprétation des rôles (désignée par la lettre R^+), qui permet l'expression de rôles transitifs. Une dernière extension, symbolisée par la lettre (*D*), ajoute le support des types primitifs.

La nomenclature des LD dicte que pour chaque constructeur ajouté, il faut agglutiner la lettre correspondante au nom de la logique originale (Fournier-Viger, 2005). Par conséquent, La différence entre *AL* et *ALC*, vient du constructeur de négation complète (*C*). Ainsi, la logique *AL*, enrichie de l'union (*U*) et de la quantification existentielle complète (*E*), se nomme *ALUE*. Il faut noter que l'appellation *ALC* équivaut à *ALUE* (l'union et la quantification existentielle complète s'expriment par la négation complète et inversement car : C∪D⇔ ¬ (¬C∩¬D) et ∃R.C⇔ ¬∀R.¬C). La lettre *S* désigne la logique *ALC* additionnée de R^+.

c) La T-Box et la A-Box

La modélisation des connaissances d'un domaine à l'aide des LDs comporte deux niveaux, la T-box et la A-box (Baader, *et al.*, 2003):

- Le niveau terminologique T-box : décrit les connaissances générales d'un domaine et contient les déclarations des primitives conceptuelles organisées en concepts et relations. Ces déclarations décrivent les propriétés des concepts et des relations et constituent donc une définition intentionnelle des connaissances ;
- Le niveau assertionnel A-box : décrit les connaissances factuelles d'un domaine et représente une configuration précise. Il contient les déclarations d'individus, instances des

concepts qui ont été définis dans la T-box. Plusieurs A-box peuvent être associées à une même T-box ; chacune représente une configuration constituée d'individus, et utilise les concepts et rôles de la T-box pour l'exprimer.

Un exemple d'une base de connaissances est fourni au tableau 2.3. Le côté gauche présente un exemple de T-box dans laquelle les noms commençant par une lettre majuscule, comme Humain, Animal, Femelle ou Male, désignent des concepts et ceux commençant par une minuscule, comme relationParentEnfant, désignent des rôles.

TBox	*ABox*
$Femelle \subseteq T \cap \neg\ Mâle$ $Mâle \subseteq T \cap \neg\ Femelle$ $Animal \equiv Mâle \cup Femelle$ $Humain \subseteq Animal$ $Femme \equiv Humain \cap Femelle$ $Homme \equiv Humain \cap \neg\ Femelle$ $Mère \equiv Femme \cap \exists\ relationParentEnfant$ $Père \equiv Homme \cap \exists\ relationParentEnfant$ $MèreSansFille \equiv Mère \cap$ $\forall\ relationParentEnfant. \neg\ Femme$ $RelationParentEnfant \subseteq T_R$	*Humain(Anne)* *Femelle(Anne)* *Femme(Sophie)* *Humain(Robert)* $\neg$*Femelle(Robert)* *Homme(David)* *relationParentEnfant(Sophie,Anne)* *relationParentEnfant(Robert,David)*

Tableau 2.3 Base de connaissances composée d'une T-box et d'une A-box (Fournier-Viger, 2005).

d) L'inférence dans les logiques de description

Elle s'effectue au niveau terminologique ou assertionnel. Quatre principaux problèmes se présentent pour chacun de ces deux niveaux que le moteur d'inférence essai de les résoudre.

-L'inférence au niveau terminologique : quatre principaux problèmes d'inférence se présentent au niveau terminologique :

- Satisfiabilité : Trouver tous les concepts insatisfiables.
- Subsomption : Calculer la hiérarchie de subsomption ou taxonomie des concepts.
- Trouver les concepts équivalents.
- Trouver les concepts disjoints.

-L'inférence au niveau assertionnel : le niveau assertionnel comprend quatre principaux problèmes d'inférence :

- Cohérence de la ABox : les assertions définies restent cohérentes avec la TBox.
- Vérification d'instance : vérifier que chaque instance respecte la définition de son concept.
- Vérification de rôle : vérifier si les rôles de l'instance sont correctement utilisés.

2.8 Langages de représentation de l'ontologie

Une des principales décisions à prendre dans le procédé de développement d'ontologies consiste à choisir le langage dans lequel l'ontologie sera exprimée et utilisée.

Au début des années 1990, des langages fondés sur des techniques de représentation de l'Intelligence Artificielle (IA) ont été conçus pour la spécification des ontologies. Ils sont basés principalement, selon (Corcho, *et al.*, 2003), sur les formalismes de représentation de connaissances suivants :

- La logique du premier ordre tel que ***KIF*** ;
- Les frames combinés avec la logique du premier ordre tel qu'Ontolingua, OCML et ***Flogic*** ;
- La logique de description telle que ***Loom***.

Ces langages sont considérés comme étant traditionnels (appelés aussi classiques) par rapport à ceux présentés dans la section suivante.

Le boom d'Internet à mené à la création des langages d'implémentation des ontologies exploitant les caractéristiques du Web. Ils sont connus sous le nom des langages de balisage (*markup languages*) ou des langages d'ontologie dans le contexte du Web sémantique (web-based ontology language). Certains d'entre eux sont basés sur la syntaxe de XML tels que : ***XOL*** (Ontology Exchange Language), ***SHOE*** (Simple HTML Ontology Extension) qui a été précédemment basé sur le HTML, ***RDF*** (Resource Description Framework) et ***RDF Schéma*** qui est une extension de RDF. Les deux derniers sont des langages développés par des groupes de travail du World Wide Web Consortium (W3C). La combinaison de RDF et RDF *Schema* est connue sous le nom de ***RDF(S)***. Par la suite, trois autres langages ont été développés comme une extension de RDF(S) qui sont basés sur la logique de description : ***OIL*** (Ontology Inference Layer), ***DAML+OIL*** et ***OWL*** qui est le successeur de DAML+OIL. OWL est fractionné en trois langages distincts : ***OWL LITE***, ***OWL DL*** et ***OWL FULL***.

La figure 2.15 présente les langages de spécification d'ontologie qui sont récemment développés et les rapports principaux entre eux sous forme d'une pyramide des langages du Web sémantique.

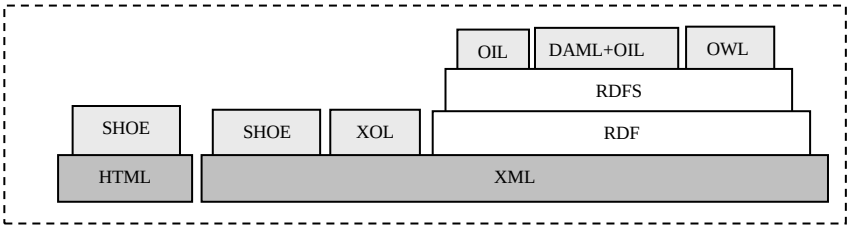

Figure 2.15 La pyramide des langages d'ontologies basés Web (Corcho, *et al.*, 2003).

2.8.1 RDF (Lassila & McGuinness, 2001)

RDF[6] (Ressource Description Framework) développé et recommandé par le W3C, permet de décrire les ressources du web sémantique qui sont l'élément de base de RDF. Chaque ressource est pourvue d'un identifiant URI (Uniform Resource Identifier).

Tout document RDF est composé d'un ensemble de triplets (sujet, prédicat, objet) ou encore (ressource, propriété, valeur). Un ensemble de tels triplets est appelé un graphe RDF. Ceci peut être illustré par un diagramme composé de nœuds et d'arcs orientés, dans lequel chaque triplet est représenté par un lien nœud-arc-nœud (d'où le terme de "graphe").
A ce modèle est associée une syntaxe écrite en XML et basée sur les triplets :

- Ressource (Sujet) : une entité d'informations pouvant être référencée par un identificateur. Cet identificateur doit être une URI.
- Propriété (prédicat) : l'attribut ou la relation utilisée (e) pour décrire une ressource.
- Valeur (objet) : la valeur d'une propriété associée à une ressource spécifique.

2.8.2 RDFS

Afin de renforcer ce langage, RDF Schéma[7] a été construit par W3C comme extension de RDF comportant des primitives basées sur des frames. RDF Schéma permet notamment de déclarer les propriétés des ressources ainsi que le type des ressources. La combinaison de RDF et RDF Schéma est connue sous le nom RDF(S). Bien que relativement limités dans la mesure où ils ne sont pas très expressifs, les langages RDF(S) peuvent cependant spécifier des concepts, des taxonomies et des relations binaires.

2.8.3 DAML-OIL

DAML-OIL a été proposé par le W3C pour représenter des méta-données et des ontologies. DAML a été transformé en DAML+OIL en intégrant certaines propriétés d'OIL. Il repose sur RDF et RDF schéma et fournit en plus des primitives plus riches issues de la logique de description.

2.8.4 OWL

OWL[8] (Web Ontology Language) est le standard actuellement recommandé par W3C pour représenter les ontologies. C'est une extension du vocabulaire de RDF(S). Il est dérivé du langage d'ontologie DAML+OIL.

[6] http://www.w3.org/RDF/
[7] http://www.w3.org/TR/rdf-schema/
[8] http://www.w3.org/OWL/

OWL a été fractionné en fait en trois sous-langages distincts offrants une complexité croissante et destinés à des communautés différentes d'utilisateurs. Chacun est une extension par rapport à son prédécesseur plus simple. Ils sont caractérisés par l'expressivité croissante (d'OWL Lite à OWL FULL) et la calculabilité des inférences décroissantes (d'OWL Lite à OWL FULL).

- **OWL LITE:** est d'expressivité faible par apport aux autres sous langages, mais qui reste quand même suffisant pour des utilisateurs qui ont principalement besoin d'une hiérarchie de classification et de contraintes simples. Par exemple, malgré qu'il permet d'exprimer des contraintes de cardinalité, il limite leurs valeurs à 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. Ainsi, le langage offre une calculabilité maximale (temps de calcul acceptables), ce qui compense sa faible expressivité.
- **OWL DL:** nommé DL car il correspond à la logique descriptive. Il est d'expressivité maximale sans perte de calculabilité. Il convient aux utilisateurs qui veulent une expressivité maximale tout en maintenant la complétude de calcul (toutes les inférences sont garanties calculables) et la décidabilité (tous les calculs s'effectuent dans un temps fini). OWL DL inclut tous les constructeurs du langage OWL, mais ils sont utilisables seulement sous certaines restrictions pour garantir la décidabilité des calculs. Par exemple, lorsqu'une classe est sous classe de plusieurs classes, elle ne peut pas être instance d'une autre classe.
- **OWL FULL:** offre un maximum d'expressivité. Il a l'avantage de la compatibilité complète avec RDF/RDFS, mais aucune garantie concernant la calculabilité (la complétude et la décidabilité des calculs liés à l'ontologie) n'est offerte par ce langage.

2.9 Outils de développement des ontologies

Les outils de développement d'ontologies qui existent sur le marché aujourd'hui sont divers et variés à bien des égards. Cet état de choses suscite beaucoup d'interrogations lorsque vient le moment d'en choisir un pour construire une nouvelle ontologie (Gómez-Pérez, *et al.*, 2004): L'outil offre-t-il une assistance au développement ? L'outil dispose-t-il d'un moteur d'inférence ? Quels langages d'ontologies l'outil supporte-t-il ? L'outil permet-il d'importer/exporter des ontologies ? L'outil offre t-il un support à la réutilisation d'ontologies existantes ? L'outil permet-il de documenter les ontologies construites ? L'outil offre-t-il un support graphique à la construction des ontologies ? Les réponses à toutes ces questions pourraient aider à prendre une décision dans le choix de l'un ou l'autre outil.

Dans cette section, nous allons présenter quelques outils d'ingénierie ontologique. Ils permettent à l'utilisateur de créer des ontologies de manière indépendante des langages de représentation et de prendre en charge la phase d'opérationnalisation de l'ontologie en l'exportant dans des langages informatisés standards.

2.9.1 PROTÉGÉ (Noy, *et al.*, 2000)

PROTÉGÉ[9] a été développé par le Stanford Medical Informatics de l'Université de Stanford. Protégé est une plate-forme Open Source autonome, qui fournit un environnement graphique permettant l'édition, la visualisation et le contrôle (vérification des contraintes) d'ontologies. Le modèle de représentation de connaissances de PROTÉGÉ, est issu du modèle des frames. Ce dernier contient des classes (pour modéliser les concepts), des slots (pour modéliser les attributs des concepts) et des facettes (pour définir les valeurs des propriétés et des contraintes sur ces valeurs), ainsi que des instances des classes. PROTÉGÉ introduit la notion de métaclasse, dont les instances sont des classes. L'interface très complète ainsi que l'architecture logicielle extensible permettant l'insertion de plusieurs plug-ins offrant de nouvelles fonctionnalités, notamment des pluggins pour gérer les représentations sous forme graphique, par exemple OWLViz et la prise en charge de nouveaux langages.

Toutes ces caractéristiques ont participé à son succès et le rendent l'éditeur d'ontologie jouissant de la plus grande renommée à l'heure actuelle, servant de référence pour une importante communauté d'utilisateurs.

2.9.2 OILEd (Bechhofer, *et al*., 2001)

L'éditeur OILEd**[10]** a été développé en 1991 sous la responsabilité de l'université de Manchester pour éditer des ontologies dans les langages de représentation OIL, puis DAML+OIL. Il est orienté vers la représentation en logique de description expressive et, à ce titre, fournit tous les éléments d'interface permettant de spécifier des hiérarchies de concepts et de rôles, les restrictions sur les rôles et les instances.

Il peut être connecté à un raisonneur de logique des descriptions tel que FaCT et RACER, capable de tester la satisfiabilité des ontologies construites ou d'expliciter de nouvelles relations de subsomption entre concepts complexes.

[9] PROTÉGÉ est disponible à l'adresse suivante : http://protege.stanford.edu/.
[10] http://oiled.man.ac.uk/

2.9.3 WebODE (Arpírez, *et al*., 2003)

WebODE[11] a été développé par le groupe Ontological Engineering du département d'Intelligence artificielle de la faculté d'Informatique de l'université polytechnique de Madrid. Un éditeur qui assurait le support de METHONTOLOGY, la méthodologie proposée par ce laboratoire. WebODE est composé de plusieurs modules : un éditeur d'ontologie qui intègre la plupart des services nécessaires à la construction d'ontologies (édition, navigation, comparaison, fusion, raisonnement…), un système de gestion des connaissances à base ontologique, un outil pour annoter les ressources du web et un éditeur de services pour le Web sémantique.

2.9.4 ONTOEDIT (Sure, *et al*., 2002)

ONTOEDIT est un environnement d'ingénierie ontologique mis au point par l'institut AIFB de l'université de Karlsruhe et qui est maintenant commercialisé par la société Ontoprise GmbH. Cet outil est fondé sur un processus de développement d'ontologies suivant les différentes étapes de la méthode de construction ON-TO-KNOWLEDGE et met à disposition de l'utilisateur plusieurs vues graphiques correspondant aux différentes phases de conception de l'ontologie.

2.10 Moteurs d'inférences

La plupart des moteurs d'inférences existants sont conçus pour raisonner sur les logiques de descriptions, mais acceptent en entrée des fichiers OWL. Une fois l'ontologie chargée, ces moteurs effectuent les inférences sur la TBox et la ABox.

Moteur	RACER	Pellet	FaCT	FaCT++
DL	*SHIQ*	*SHIN(D)* *SHON*	*SHIQ* *SHF*	*SHIF*
Implantation	C++	Java	Common Lisp	C++
Inférence	TBox/ABox	TBox/ABox	TBox	TBox
API Java	Oui	Natif	Oui	Oui
OWL	OWL DL	OWL DL	OWL DL	OWL Lite
Décidabilité	Oui (OWL Lite)	Oui (OWL Lite)	Oui	Oui

Moteur	Surnia	Hoolet	F-OWL	
DL	Logique predicats	Logique predicats	*SHIQ*	
Implantation	Python	Java	Java	
Inférence	TBox/ABox	TBox/ABox	TBox/ABox	
API Java	Non	Oui	Oui	
OWL	OWL Full	OWL DL	OWL Full	
Décidabilité	Non	Non	Non	

Tableau 2.4 Une comparaison des principaux moteurs d'inférence pour les LDs expressives (Fournier-Viger, 2005).

[11] http://webode.dia.fi.upm.es/WebODEWeb/index.html

Pellet et Racer sont à l'heure actuelle les deux seuls moteurs d'inférence, permettant le raisonnement sur la ABox et la TBox et exploitent des ontologies possédant un niveau d'expressivité en logique de description et acceptent en entrée des fichiers OWL. Le tableau 2.4, présenté ci-dessus, représente une comparaison des principaux moteurs d'inférences pour les LD.

2.11 Quelques ontologies médicales

Comme dans la plupart des domaines de recherche. Les chercheurs dans le domaine médical visent à représenter, partager et réutiliser leurs connaissances. Par conséquent plusieurs terminologies et ontologies ont été proposées et construites à des besoins précis et divers. On va faire un panorama de certaines d'elles.

2.11.1 CIM

L'appellation complète de la Classification internationale des maladies est « Classification statistique Internationale des Maladies et des problèmes de santé connexes » (en anglais : International Statistical Classification of Diseases and Related Health Problems). La désignation usuelle abrégée de « Classification Internationale des Maladies » est à l'origine du sigle couramment utilisé pour la désigner : « la CIM » (en anglais : ICD). La CIM permet le codage des maladies, des traumatismes et de l'ensemble des motifs de recours aux services de santé. Elle est publiée par l'Organisation Mondiale de la Santé. Elle bénéficie d'une remise à niveau régulière, la version la plus récente étant la 10ème révision CIM-10. Il s'agit d'une classification monoaxiale avec 21 chapitres principaux dont 17 concernent des maladies et 4 concernent les signes et résultats anormaux, les causes de traumatismes, d'empoisonnement ou de morbidité, l'état de santé et les facteurs de recours aux soins.

2.11.2 UMLS (Bodenreider & Burgun, 2005)

UMLS (Unified Medical Language System) est un vaste projet élaboré par le NLM (National Library of Medecine) aux Etats-Unis. Il est proposé depuis 1986 de mettre au point un langage médical unifié pour aider les professionnels de santé et les chercheurs d'accéder aux informations biomédicales collectées à partir d'une variété de sources (plus de 100 sources dans la version 2004) tel que les vocabulaires, les classifications et les terminologies (MESH, SNOMED,…etc.). Ce langage repose sur :

- Un méta-thésaurus : si un terme apparue dans plusieurs ressources terminologiques qui l'inclut, un concept est créé dans l'UMLS avec un nom du terme préféré associé.

- Un réseau sémantique développé indépendamment de ressources terminologiques intégrées dans le méta-thésaurus. Il sert de base comme une top level ontology pour le domaine biomédical.

2.11.3 GALEN (Rector, *et al.*, 1995) (Bodenreider & Burgun, 2005)

GALEN (General Architecture for Language, Encyclopedia and Nomenclature) est un projet européen, développé par l'organisation Open Galen depuis 1992-1999. Il vise à mettre en place un serveur de terminologie médicale partageable et réutilisable. Il est centré sur une ontologie, Common Reference Model : CRM, représentée dans un langage de représentation propre à GALEN appelé GRAIL (Galen Representation And Integration Language). Cette ontologie représente des concepts médicaux des domaines dans lesquels le projet s'est développé, indépendamment de toute application. La version de GALEN top-level ontology (Décembre 2002) comprend environ 25,000 concepts.

2.11.4 MENELAS (Zweigenbaum, 1994) (Bodenreider & Burgun, 2005)

MENELAS est un projet européen, avait pour but de proposer une approche d'accès aux dossiers médicaux rédigés en différents langues naturels. Une ontologie couvrant le domaine des maladies coronariennes a été développée dans le cadre d'une application pilote. Cette ontologie a été construite à partir de plusieurs sources incluant l'analyse des corpus, les interviews avec les spécialistes et la réutilisation des ressources terminologiques existantes. Elle comporte plus de 1800 concepts et 300 relations.

2.11.5 FMA (Rosse & Mejino, 2003)

Le Foundational Model of Anatomy FMA est une ontologie de référence pour le domaine de l'anatomie. C'est une représentation de toutes les entités anatomiques et les relations nécessaires pour la modélisation symbolique de la structure phénotypique du corps humain dans une forme qui soit compréhensible par l'homme et qui soit également traitable par une machine. Le FMA est mis à la disposition d'utilisateurs qui peuvent récupérer des parties de la modélisation pour les intégrer dans leur propre ontologie.

2.11.6 NAUTILUS (Dieng-Kuntz, *et al*., 2006)

Est une ontologie médicale construite, au sein du projet 'ligne de vie', à partir d'une base de données à travers un algorithme de translation proposé par les membres du projet. Son but est d'annoter les documents afin d'améliorer la recherche des documents partageables est accessibles par les membres du réseau de soins. Elle est représentée dans un langage de représentation de connaissance standard : RDF(S).

2.11.7 OntoPneumo (Baneyx, 2007)

Est une ontologie médicale qui a été développée dans le domaine de la pneumologie pour faciliter, d'une part, l'aide au codage médicoéconomique des pathologies et, d'autre part, la représentation des connaissances relatives au patient, dans ce domaine de spécialité. Elle sert de pivot dans un outil de codage médical et médico-économique.

Bien que, les ressources présentées au dessus ne sont pas tous considérées comme étant des ontologies, ce n'est pas un problème parce que chacune d'elles est construite pour répondre à des objectifs bien précis au départ.

2.12 Conclusion

A travers ce que nous avons présenté dans ce chapitre, il en ressort que la notion d'ontologie constitue une approche très efficace pour représenter les connaissances. Tout au long de ce chapitre, nous avons essayé d'éclaircir la notion d'ontologie en présentant certaines définitions. Ensuite, nous avons exposé les composantes d'une ontologie, les différentes classifications ainsi que les principaux formalismes de représentation de connaissances à savoir les frames, les graphes conceptuels et les logiques de descriptions. En outre, nous avons décri plusieurs méthodologies de construction des ontologies. Certes, toutes les méthodologies initient le processus de construction par l'identification, puis l'organisation et la structuration des concepts et des relations à représenter mais aucune n'ait à l'heure actuelle réussie à s'imposer comme standard. En dépit de cela, de nombreux critères et principes permettant de guider la construction d'ontologies ont été proposés. Après avoir effectué un panorama de langages d'implémentation et d'éditeurs d'ontologies nous pouvons dire que le langage OWL est le plus répandu et que l'outil PROTÉGÉ s'impose comme référence.

Le chapitre suivant présente notre contribution au problème posé par ce mémoire. Nous décrirons alors le processus de développement d'ontologies qui sera utilisé par la suite afin de bâtir l'ontologie du domaine de la prise en charge des patients à domicile, qui est l'objectif de ce mémoire.

CHAPITRE III

DÉVELOPPEMENT DE L'ONTOLOGIE OntoPAD

3.1 Introduction

Le présent chapitre présente notre contribution à la problématique posée dans ce mémoire, à savoir la construction d'une ontologie pour la prise en charge des patients à domicile, baptisée «OntoPAD». En premier lieu, nous décrirons le processus que nous avons suivi pour la construction de l'ontologie. Ce processus est composé de cinq- phases : spécification des besoins, conceptualisation, formalisation, opérationnalisation et évaluation. En effet, ces différentes phases sont inspirées à partir de certaines méthodologies étudiées dans l'état de l'art. Ces dernières sont celles permettant la construction d'ontologie à partir de zéro, plus particulièrement la méthodologie d'Uschold et King, le processus de Hemam et Boufaida, METHONTOLOGY et ON-TO-KNOWLEDGE. D'ailleurs, la majorité de ces phases sont partagées dans ces cadres méthodologiques même avec des noms différents. Par ailleurs, nous avons tenu-compte des conseils et principes d'ingénierie ontologique (IO) largement acceptés et évalués par les experts du domaine d'IO. En deuxième lieu, nous montrerons comment ce processus a été utilisé pour la conception de l'OntoPAD. Nous verrons alors, comment l'ontologie a été conceptualisée, en utilisant un ensemble de représentations intermédiaires, semi-formelles de METHONTOLOGY. Le résultat de cette phase est une ontologie conceptuelle. Pour la formaliser, nous avons fait recours à la logique de description *SHOIN(D)* qui possède une sémantique claire et procure des services d'inférences et des mécanismes de raisonnement puissants. Par la suite, nous décrirons comment nous avons utilisé l'outil PROTÉGÉ pour le codage d'OntoPAD en langage OWL-DL. Enfin, nous illustrerons la phase de vérification par le biais du système d'inférence RACER.

3.2 Le processus suivi pour la construction de l'OntoPAD

Bien qu'il n'existe pas une méthode ou méthodologie standard pour bâtir une ontologie, et que chaque équipe utilise la méthode qu'elle propose, nous avons utilisé un processus basé sur certains travaux existants.

Tout d'abord, Nous appliquons tout au long du processus que nous avons adopté pour le développement de notre ontologie les conseils de départ suivants proposés par (Noy & McGuiness, 2001):

- Il n'y a pas qu'une seule façon correcte pour modéliser un domaine, il y a toujours des alternatives viables.
- Le développement d'une ontologie est nécessairement un processus itératif.
- Les concepts dans une ontologie doivent être très proches des objets (physiques ou logiques) et des relations dans le domaine d'intérêt. Fort probablement, ils sont des noms (objets) et des verbes (relations) dans les phases qui décrivent le domaine.

3.2.1 Description des principes d'ingénierie ontologique et leur application

Nous appliquons les principes de construction des ontologies, permettant de guider l'ontologiste à construire et à évaluer l'ontologie, dans la réalisation des différentes phases du processus de construction d'ontologies mentionnées au-dessous (voir la section 3.2.2):

- **Clarté/Objectivité** (Gruber, 1993b) : L'ontologie doit fournir la signification des termes définis en fournissant des définitions <<objectives>> ainsi qu'une documentation en langage naturel.
- **Complétude/ Perfection** (Gruber, 1993b) : Une définition complète (exprimée par des conditions nécessaires et suffisantes) est préférable à une définition partielle (définie seulement par des conditions nécessaires).
- **Cohérence** (Gruber, 1993b) : Une ontologie cohérente doit permettre des inférences conformes à ces définitions. Les axiomes doivent être consistants.
- **Extensibilité monotone maximale** (Gruber, 1993b) : De nouveaux termes généraux et spécialisés devraient être inclus dans l'ontologie d'une façon qui n'exige pas la révision des définitions existantes.
- **Biais de codage minimal** (Gruber, 1993b) **:** l'ontologie doit être conceptualisée indépendamment de tout langage d'implémentation. Il faut minimiser les choix de représentations faits pour permettre le partage des connaissances, contenues dans l'ontologie, entre différentes applications utilisant des langages de représentation différents.
- **Engagements ontologiques minimaux** (Gruber, 1993b) : l'ontologie doit faire aussi peu d'hypothèses (réclamations) que possible sur le monde qu'elle modélise. elle doit contenir un vocabulaire partagé mais ne doit pas être une base de connaissances comportant des connaissances supplémentaires sur le monde à modéliser. Autrement dit, elle doit choisir de représenter un point de vue en particulier sur le domaine qui l'intéresse.
- **Distinction ontologique** (Borgo, *et al.*, 1996) : les classes dans une ontologie devraient être disjointes. Le critère utilisé pour isoler le noyau de propriétés considérées comme

invariables pour une instance d'une classe est appelé le critère d*'Identité.* Ce critère permet de spécialiser les concepts tant qu'il est possible.

- **Modularité** (Bernaras, *et al.*, 1996) **:** il faut minimiser les couplages entre les modules.
- **Distance sémantique minimale** (Arpírez, *et al.*, 1998) : Il faut minimiser la distance entre les concepts enfants de même parents. Les concepts similaires sont groupés et représentés comme des sous-concepts d'un concept, et devraient être définis en utilisant les mêmes primitives, considérant que les concepts qui sont moins similaires sont représentés plus loin dans la hiérarchie.
- **Normaliser les noms** (Arpírez, *et al.*, 1998) : Il faut normaliser les termes chaque fois que c'est possible.
- **Diversification des hiérarchies** (Arpírez, *et al.*, 1998) : Ce qui permet d'augmenter la puissance fournie par les mécanismes d'héritage multiple. Si suffisamment de connaissances sont représentées dans l'ontologie et que suffisamment de différents critères de classifications sont utilisés, il est plus facile d'ajouter de nouveaux concepts (puisqu'ils peuvent être facilement spécifiés à partir des concepts et des critères de classifications préexistants) et de les faire hériter de propriétés de différents points de vue.

Selon notre vue, ces principes sont appliqués de la façon suivante (tableau 3.1):

Spécification	Conceptualisation	Formalisation
-Engagements ontologiques minimaux	-Clarté/Objectivité -Complétude/ Perfection -Cohérence -Biais de codage minimal -Distinction ontologique -Modularité minimale -Distance sémantique minimale -Normaliser les noms -Diversification des hiérarchies	-Cohérence -Extensibilité monotone maximale

Tableau 3.1 Application des principes d'IO.

3.2.2 Les phases du processus de construction de l'ontologie

Il en ressort que les phases qui permettent la construction d'une ontologie opérationnelle partant des connaissances brutes d'un domaine sont :

- Spécification ;
- Conceptualisation ;
- Formalisation ;

- Opérationnalisation ;
- Évaluation.

3.2.2.1 Spécification

Pour commencer le développement de l'ontologie, une première importante étape doit être effectuée. Elle consiste à établir un document informel de spécification de besoins écrit dans un langage naturel. Nous décrivons dans ce document (Staab, *et al.*, 2001), (Fernández-López, *et al.*, 1997), (Uschold & King, 1995) (Hemam & Boufaida, 2004):

- Le domaine de connaissance qui sera représenté par l'ontologie;
- L'objectif de l'ontologie à créer pour le domaine considéré ;
- Les utilisateurs futurs de l'ontologie ;
- Les sources d'informations desquelles les connaissances seront obtenues. Ils sont de nature différentes et variées, par exemple : les interviews avec les experts du domaine, les documents techniques (publications scientifiques, livres), les observations, les ontologies existantes qui peuvent être réutilisés…etc. ;
- La portée de l'ontologie : déterminer la liste des termes candidats du domaine à travers l'analyse des sources d'informations relatif au domaine.

3.2.2.2 Conceptualisation

Sans doute, l'étape de conceptualisation est la plus importante dans le processus de développement de l'ontologie. Elle mérite une attention particulière car elle détermine le reste de la construction de l'ontologie.

Cette phase est inspirée de la méthodologie METHONTOLOGY (Fernández-López, *et al.*, 1997) parce qu'elle est spécifiée de manière très détaillée. Elle consiste à organiser et à structurer, à partir des sources d'informations, les connaissances du domaine en utilisant un ensemble de représentations intermédiaires, semis formelles, sous forme de tableaux et graphes, indépendamment du mécanisme de formalisation utilisé pour représenter l'ontologie. A la fin nous obtenons une ontologie conceptuelle (model conceptuel).

Les principales tâches suivantes sont réalisées :

- **Tâche1 :** construire le glossaire des termes en identifiant tous les termes inclut dans l'ontologie, leurs définitions en langage naturel, leurs synonymes et leurs acronymes ;
- **Tâche2 :** construire la taxonomie de concepts. Elle est réalisée à travers un choix d'une stratégie d'identification de concepts. Trois stratégies d'identification des concepts ont été proposées par (Uschold & Grüninger, 1996):
    - Approche ascendante (bottom-up strategy) : les concepts les plus

spécifiques sont identifiés, par la suite, ils sont généralisés en concepts plus abstraits.

- ♦ Approche descendante (top-down strategy): les concepts les plus abstraits sont identifiés, par la suite, ils sont spécialisés en plus spécifiques.
- ♦ Approche centrifuge (middle-out strategy): les concepts les plus importants sont identifiés (centraux), par la suite, ils sont généralisés et spécialisés comme il est nécessaire.

En fait, aucune de ces trois stratégies n'est fondamentalement meilleure que les autres. L'approche à adopter dépend fortement du point de vue personnel du concepteur d'ontologie sur le domaine. L'approche combinée est souvent, la plus facile à utiliser pour la plupart des développeurs d'ontologies. Leur justification est que les concepts « du milieu », qui ont une position intermédiaire dans l'ontologie, ont tendance à être les concepts les plus descriptifs du domaine.

- ➢ **Tâche3 :** construire le diagramme de relations binaires où les relations entre les concepts de l'ontologie sont identifiés;
- ➢ **Tâche4 :** construire le dictionnaire de concepts qui inclut principalement les instances de chaque concept, leurs attributs et leurs relations;
- ➢ **Tâche5 :** décrire en détail dans une table de relations binaires les relations apparues dans le diagramme de relation binaires;
- ➢ **Tâche6 :** décrire les attributs apparus dans le dictionnaire de concepts en spécifiant leur contraintes dans une table d'attributs ;
- ➢ **Tâche7 :** spécifier les axiomes sur les concepts dans une table d'axiomes logiques ;
- ➢ **Tâche8 :** décrire les instances des concepts dans une table d'instances ;

3.2.2.3 Formalisation

Le processus de construction d'ontologie se poursuit par une étape de formalisation. Elle consiste à formaliser l'ontologie conceptuelle obtenue à l'étape précédente afin de faciliter sa représentation ultérieure dans un langage formel et opérationnel. Nous utilisons comme formalisme de représentation de connaissance la logique de description en s'appuyant sur la syntaxe de type *SHOIN(D)*. Ce choix est motivé par le fait que *SHOIN(D)* est une logique de description très expressive, qui offre un certain nombre de constructeurs permettant de représenter des concepts (aussi appelés *classes*) d'un domaine d'application et les propriétés (aussi appelées *rôles*) qui représentent des relations binaires pouvant être établies entre les instances de ces classes. Ainsi, elle est la logique sur laquelle se base le langage OWL-DL qui est utilisé par la

suite pour la codification de l'ontologie. Le résultat de cette étape est une ontologie formelle (model formel).

3.2.2.3.1 Les constructeurs de la logique *SHOIN(D)*

La logique propose les constructeurs : S désigne la logique ALC additionnée de R+ (ensemble de rôles transitifs), H permet d'exprimer l'inclusion des rôles, O qui permet la description de concepts par énumération d'individus nommés, I permet la description des rôles inverses, N permet d'exprimer des contraintes de cardinalités sur les rôles et une dernière extension, symbolisée par la lettre (D), ajoute le support de types de données (voir le tableau 3.2).

Δ^I : est un ensemble arbitraire non vide d'individus ;

Δ^I_D: est un second domaine d'interprétation disjoint avec Δ^I, représente l'ensemble des valeurs de types de données prédéfinis tels que les entiers, les chaînes de caractères, etc.

D: est un type de données tel que $D^D \subseteq \Delta^I_D$;

^{I} : est une fonction d'interprétation que fait correspondre à un concept *C* un sous ensemble $C^I \subseteq \Delta^I$, à une relation *R* un sous ensemble de $R^I \subseteq \Delta^I * \Delta^I$, à un attribut *U* un sous ensemble $U^I \subseteq \Delta^I * \Delta^I_D$, à une instance *o* : $o^I \in \Delta^I$ et à un littérale *t* : $t^I \in \Delta^I_D$.

Nom du constructeur	Syntaxe	Sémantique
Top (Thing)	T	$T^I = \Delta^I$
Buttom (Nothing)	$\perp$	$\perp^I = \Phi$
Le concept atomique (primitif)	A	$A^I \subseteq \Delta^I$
Le nom de classe	C_1	$C_1^I \subseteq \Delta^I$
L'intersection (conjonction)	$C_1 \cap C_2$	$(C_1 \cap C_2)^I = C_1^I \cap C_2^I$
L'union (disjonction)	$C_1 \cup C_2$	$(C_1 \cup C_2)^I = C_1^I \cup C_2^I$
La négation	$\neg C_1$	$(\neg C_1)^I = \Delta^I \setminus C_1^I$
Énumération	$\{o_1, o_2, ..., o_N\}$	$\{o_1, o_2, ..., o_N\}^I = \{o_1^I, o_2^I ..., o_N^I\}$
Le quantificateur existentiel	$\exists\ R_1.C_1$	$(\exists R_1.C_1)^I = \{x \mid \exists y.\ (x,y) \in R_1^I \text{ and } y \in C_1^I\}$
Le quantificateur universel	$\forall R_1.C_1$	$(\forall R_1.C_1)^I = \{x \mid \forall x.\ (x,y) \in R_1^I \text{ then } y \in C_1^I\}$
Restriction à une valeur	$\ni R_1.o_1$	$(\ni R_1.o_1)^I = \{x \mid (x, o_1^I) \in R_1^I\}$
Contraintes de cardinalité non quantifié Minimum/Maximum sur une relation	$(\geq n\ R_1)$ $(\leq n\ R_1)$ $(= n\ R_1)$	$(\geq n\ R_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in R^I{}_1\} \geq n\}$ $(\leq n\ R_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in R^I{}_1\} \leq n\}$ $(= n\ R_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in R^I{}_1\} = n\}$
Le quantificateur existentiel	$\exists\ U_1.D$	$(\exists U_1.D)^I = \{x \mid \exists y.\ (x,y) \in U_1^I \text{ and } y \in D^D\}$
Le quantificateur universel	$\forall\ U_1.D$	$(\forall U_1.D)^I = \{x \mid \forall x.\ (x,y) \in U_1^I \text{ then } y \in D^D\}$

Contraintes de cardinalité non quantifié Minimum/Maximum sur un attribut	$(\geq n\ U_1)$ $(\leq n\ U_1)$ $(= n\ U_1)$	$(\geq n\ U_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in U_1^I\} \geq n\}$ $(\leq n\ U_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in U_1^I\} \leq n\}$ $(= n\ U_1)^I = \{x \mid \text{card}\{y \mid (x,y) \in U_1^I\} = n\}$
Le rôle inverse	R_1^-	$(R_1^-)^I = (R_1^I)^-$

Tableau 3.2 Syntaxe et sémantique de *SHOIN(D)* inspirée de (Horrocks & Patel-Schneider, 2003).

Les concepts sont soit atomiques soit des expressions s'appuyant sur des constructeurs de classe. *SHOIN(D)* permet de définir les connaissances suivant deux niveaux de description : le niveau terminologique (TBox) décrit les concepts et les rôles entre les concepts d'un domaine alors que le niveau assertionnel (ABox), décrit les assertions sur les individus.

3.2.2.3.2 La partie terminologique (T-BOX)

La T-BOX, ou base terminologique, est un ensemble fini d'axiomes d'inclusions (équivalences) de classes, d'axiomes d'inclusions (équivalence) de rôles et d'axiomes de transitivité. Ils sont utilisés pour décrire (axiomes d'inclusions) ou définir (axiomes d'équivalences) les classes relatives à notre domaine, en utilisant les constructeurs fournis par les LDs.

Les axiomes de classes consistent en deux classes séparées par le symbole d'inclusion $\subseteq$ (axiome d'inclusion ou encore GCI : General Concept Inclusion) ou par le symbole d'équivalence $\equiv$ (équivalent à $\subseteq$ dans les deux directions $\subseteq$ et $\supseteq$) (voir le tableau 3.3). Même chose pour les axiomes de rôles, ils consistent en deux noms de rôles séparés par le symbole $\subseteq$ ou $\equiv$ (voir le tableau 3.3) sachant que l'apparence de ces deux rôles dans le même axiome réfère soit à deux attributs soit à deux relations. On peut trouver aussi l'axiome de transitivité d'une relation trans.
Les axiomes que nous avons utilisés sont de la forme :

Nom de l'axiome	Syntaxe de l'axiome	Sémantique de l'axiome
inclusion de concepts	$C_1 \subseteq C_2$	$C_1^I \subseteq C_2^I$
équivalence de concepts	$C_1 \equiv C_2$	$C_1^I \equiv C_2^I$
inclusion de relations	$R_1 \subseteq R_2$	$R_1^I \subseteq R_2^I$
équivalence de relations	$R_1 \equiv R_2$	$R_1^I \equiv R_2^I$
transitivité d'une relation	$Trans(R_1)$	$R_1^I = (R_1^I)^+$

Tableau 3.3 Syntaxe des axiomes avec leurs sémantiques (Horrocks & Patel-Schneider, 2003).

3.2.2.3.3 La partie Assertionnelle (A-BOX)

La ABox, ou base assertionnel, contient un ensemble d'assertions sur les individus. Ils sont soit des appartenances à une classe soit des valeurs de rôles (relations et attributs) (voir le tableau 3.4).

Les assertions d'individus qu'on va utiliser sont de la forme :

Syntaxe de l'assertion	Sémantique de l'assertion
$C_1\ (o_1)$	$o_1^I \in C_1^I$
$R_1\ (o_1, o_2)$	$(o_1, o_2) \in R_1$
$U_1\ (o_1, t)$	$(o_1, t) \in U_1$

Tableau 3.4 Syntaxe des assertions avec leurs sémantiques (Horrocks & Patel-Schneider, 2003).

Malgré qu'il est possible de définir un axiome d'inclusion de rôles pour les attributs. Plusieurs axiomes pour les relations tels que l'axiome de rôle transitif et le constructeur de rôle inverse n'ont pas leur équivalent pour les attributs.

3.2.2.4 Opérationnalisation

L'ontologie obtenue dans l'étape précédente est formelle. Le but de cette étape consiste à encoder l'ontologie formelle en un langage d'implémentation d'ontologies pour la rendre opérationnelle. Elle est réalisée en faisant recours à un outil d'édition d'ontologies. Cette étape achève donc le processus de construction de l'ontologie.

3.2.2.5 Évaluation

C'est une étape qui peut être effectué en parallèle avec les trois étapes précédentes (conceptualisation, formalisation et opérationnalisation). Elle est effectuée au niveau conceptuel qu'au niveau formel et opérationnel :

- Évaluation de l'ontologie conceptuelle : rappelons que les principes d'ingénierie ontologique sont utilisés afin de guider l'ontologiste à construire et à évaluer l'ontologie. Par conséquent, les principes respectés dans l'étape de conceptualisation sont eux même utilisés pour l'évaluation de l'ontologie conceptuelle.
- Évaluation de l'ontologie formelle et opérationnelle : elle est réalisée automatiquement au moyen de l'utilisation d'un raisonneur capable de calculer les inférences afin de supporter le processus de construction et d'améliorer la qualité de l'ontologie.

Par ailleurs, il existe une liste de critères de vérification et de validation de l'ontologie, proposé par les auteurs de la méthode METHONTOLOGY, en vue de vérifier l'inexistence des erreurs détectables au niveau d'une ontologie conceptuelle. Selon (Gómez-Pérez, *et al.*, 2004), la majorité des erreurs possibles effectuées par l'ontologiste lors de la modélisation sont faites au niveau des taxonomies et elles sont classées dans les catégories suivantes (Gómez-Pérez, *et al.*, 2004):

1) Erreurs d'inconsistance

a) Erreurs de circularité

Elles se produisent quand une classe est définie comme spécialisation ou généralisation d'elle-même (voir figure 3.1). Selon le nombre de relations impliquées, ces erreurs peuvent être classées comme : erreurs de circularité à distance zéro (une classe avec elle-même), à distance 1 et à distance n.

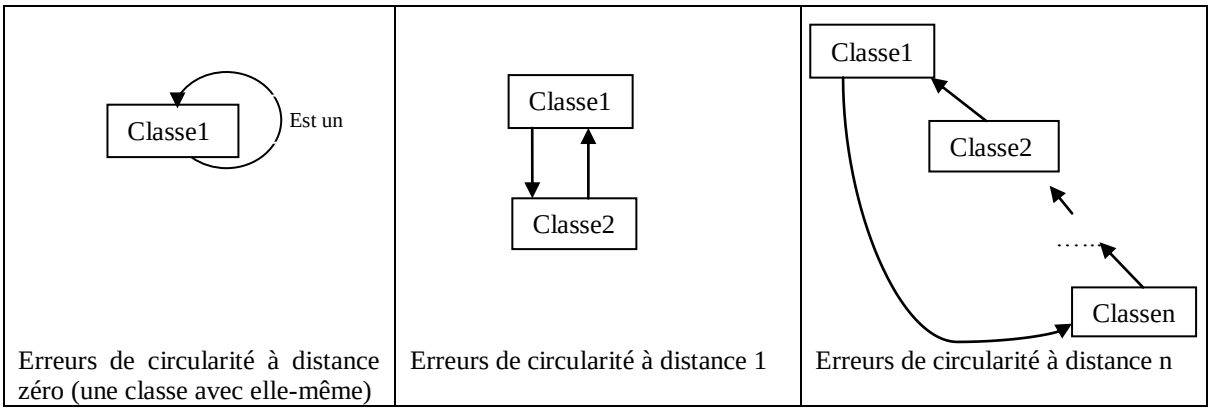

Figure 3.1 Exemple d'erreurs circulaires dans la taxonomie.

b) Erreurs sémantiques

Elles se produisent habituellement parce que l'ingénieur ontologique fait une classification sémantique incorrecte, c.-à-d., il classifie un concept comme sous-classe d'une classe d'un concept auquel il n'appartient pas vraiment. Le même problème peut arriver lors de la classification d'instances.

c) Erreurs de partition

La classification des concepts peut être définie de façon disjointe (une décomposition disjointe), complète (décomposition exhaustive) ou disjointe et complète (partition). Les erreurs suivantes sont définies (voire figure3.2):

(a) *Classes communes à une décomposition disjointe ou à une partition*. Elles apparaissent quand il y a une décomposition disjointe ou une partition *class_p1, … class_pn* définie dans une classe *class_A*, et que une ou plusieurs classes *class_B1, …, class_Bk* sont des sous-classes de plus d'une *class_pi*.

(b) *Instances communes à une décomposition disjointe ou à une partition*. Elles apparaissent quand une ou plusieurs instances appartiennent à plus d'une classe d'une décomposition disjointe ou d'une partition.

(c) *Instances externes dans une décomposition exhaustive ou une partition*. Elles apparaissent

quand on définit une décomposition complète ou une partition d'une classe *class_A* dans un ensemble de classes *class_p1, … class_pn*, et qu'il y a une ou plusieurs instances de *class_A* qui n'appartiennent à aucune classe *class_pi*.

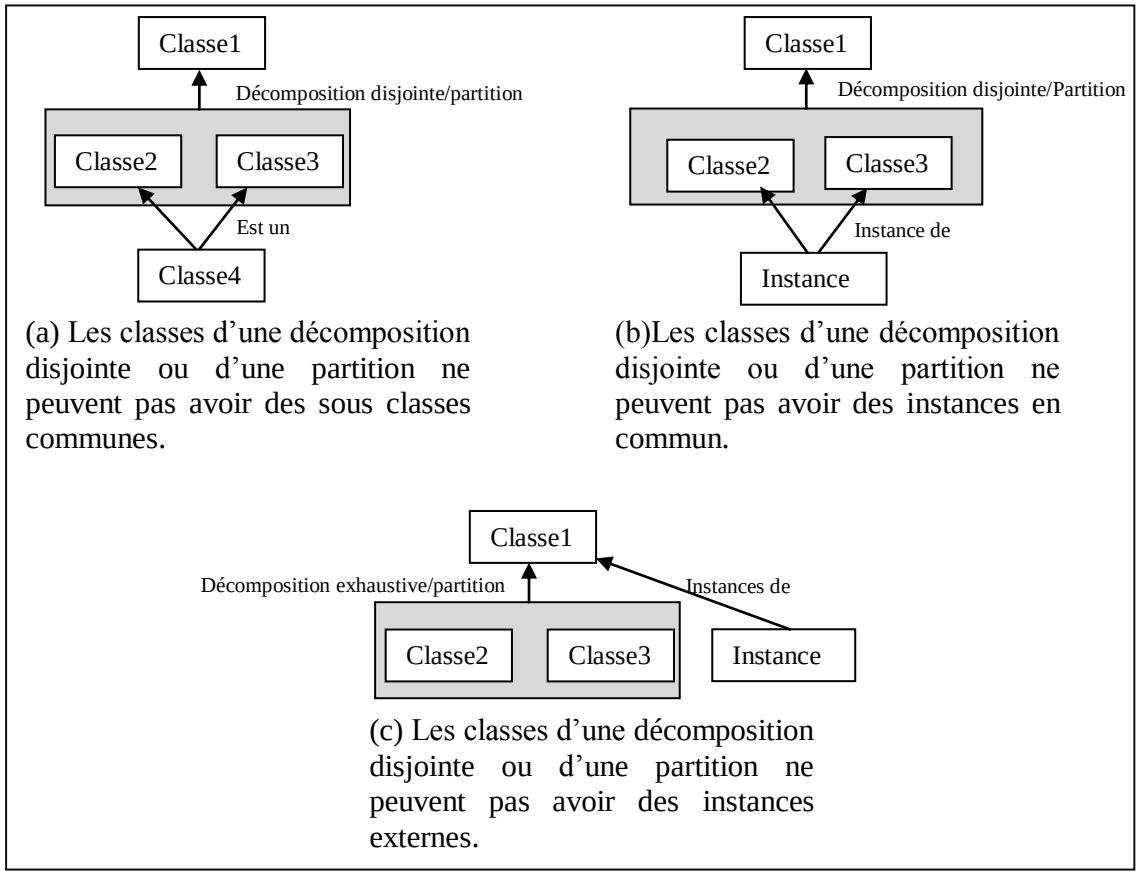

Figure 3.2 Exemple d'erreurs de partition dans l'ontologie.

2) Erreurs d'incomplétude

a) Classification de concepts incomplète : généralement, cette erreur se produit chaque fois que les concepts sont classifiés sans qu'ils n'aient tous été pris en compte, c.-à-d. quand des concepts existants dans le domaine ont été négligés.

b) Erreurs de partition : Les classifications de concepts peuvent être définies d'une façon disjointe (décomposition disjointe), complète (décomposition complète), ou disjointe et complète (partition). Deux types d'erreurs sont définis : (a) l'ingénieur ontologique définit un ensemble de sous-classe d'une classe donnée, mais omet de modéliser dans la taxinomie que les sous-classes sont disjointes alors qu'elles devraient l'être ; (b) L'ingénieur ontologique définit une partition de classe pour un ensemble de sous-classes qui ne sont pas complètement classées et devraient l'être.

3) Erreurs de redondance

Elles se produisent au niveau de la taxonomie quand on redéfinit des expressions qui ont déjà été

explicitement définies ou qui peuvent être déduites à partir d'autres définitions.

a) **Redondance de la relation « est un » :** elle apparait entre les classes qui ont plus d'une relation « est un ». on peut distinguer les répétitions directe et indirecte. Une répétition directe existe où deux ou plusieurs relation « est un » entre les mêmes classes sources et destinataires sont définies. Une répétition indirecte existe où deux ou plusieurs relations « est un » entre une classe et ses superclasses indirectes sont définies.
b) **Redondance de la relation « instance de » :** comme dans le premiers cas, on peut distinguer entre les répétitions directe et indirecte.
c) **Définitions formelles identique pour certaines classes :** des erreurs apparaissent quand différent classes ont la même définition formelle quoiqu' ils ont des différents noms.
d) **Définitions formelles identique pour certaines instances :** des erreurs apparaissent quand différent instances ont la même définition quoiqu' ils ont des différents noms.

Nous présenterons dans la section suivante comment nous avons appliqué ce processus dans le développement de l'OntoPAD.

3.3 Construction d'une ontologie pour la prise en charge des patients à domicile

Dans cette section, nous construisons OntoPAD. Selon les critères de classification étudiés dans l'état de l'art (chapitre II, section 2.3), OntoPAD est une ontologie : **lourde** (avec contrainte logique), **de domaine** de la prise en charge des patients à domicile, d'une **granularité ni assez large ni assez fine** et **formelle**. Nous suivons le processus décrit précédemment. Rappelons que nous ne pouvons pas dissocier les différentes phases de ce dernier car il s'agit d'un processus non linéaire, plusieurs allers-retours ont été fait lors du développement. Ainsi, il faut noter qu'il n'est pas possible de savoir dés le début, que les termes collectés sont suffisants pour répondre à l'objectif pour lequel l'ontologie a été construite, de nouveaux termes sont ajoutés lorsqu'il est nécessaire, de même d'autres sont retirés lorsqu'ils sont jugés inutiles jusqu'à l'obtention des termes pertinents du domaine. Le raffinement incrémental tient compte des erreurs détectées dans chaque phase.

3.3.1 Spécification

Une ontologie ne peut être construite qu'après la phase de spécification. Il s'agit d'établir un document informel de spécification de besoins. Au niveau de ce document, nous décrivons l'ontologie à construire à travers les cinq aspects suivants (figure 3.3):

Domaine de connaissance

Domaine de la prise en charge à domicile des patients.

Objectif

Permettre aux différents intervenants de la prise en charge à domicile d'avoir un vocabulaire conceptuel commun qui leur permet de partager de façon collaboratif les connaissances médicales et de communiquer facilement.

Utilisateurs

Patients, Professionnels de santé médicale, Professionnels de santé paramédicale, Professionnels psychosociales…etc.

Sources d'informations

Documents techniques en rapport avec le domaine de la PAD et l'encyclopédie Médicale.

Portée de l'ontologie

Patient, ProfessionnelDeSante, Medecin, Medicament, nom, ExamenPhysique, StructurePAD, Document, etc.

Figure 3.3 Document de spécifications de besoin d'OntoPAD.

3.3.2 Conceptualisation

Dans cette étape on distingue les principales tâches suivantes :

3.3.2.1 Construction du glossaire de termes

Construire un glossaire de termes est la première tâche à effectuer dans l'étape de conceptualisation. Il recueille et décrit tous les termes (concepts, instances, attributs, relations entre les concepts, etc.) qui sont utiles et potentiellement utilisables dans le domaine que nous allons représenter leurs descriptions détaillées et non ambiguës en langage naturel. Le tableau 3.5 illustre le glossaire de termes d'OntoPAD. Les termes recensés sont ceux en commun avec le domaine médicale et ceux propre au domaine considéré.

Nom	Description
Personne	Est un humain qui intervient dans la prise en charge à domicile d'un patient.
Entourage	Il s'agit de la famille, proche famille, collègues, amis et voisin...etc.
Patient	est une personne souffrant d'une maladie pour laquelle elle est prise en charge à son domicile. Elle subit plusieurs activités et il y a plusieurs documents qui la concernent.
ProfessionnelDeSante	Toute personne travaillant dans le cadre du système de santé. Il peut être un professionnel de santé médical ou un professionnel de santé paramédical ou un professionnel de santé psychosocial.
ProfessionnelDeSanteMedical	Ils sont au nombre de deux : médecin et pharmacien.
Medecin	Il est un professionnel de santé. Il peut exercer à l'hôpital ou avoir une

	activité libérale. Il peut être médecin prescripteur ou médecin coordonateur.
MedecinTraitant	est un médecin chargé de diagnostiquer la maladie de ses patients en effectuant des examens et de la traiter.
MedecinPrescipteur	est le médecin qui prescrit l'admission d'une prise en charge à domicile d'un patient ainsi que la sortie. Il peut être médecin hospitalier ou médecin de famille.
MedecinHospitalier	Est un médecin qui travaille dans un hôpital. s'engage à suivre le patient au niveau hospitalier et le ré-hospitaliser.
MedecinDeFamille	Est un médecin traitant exerçant dans un cabinet médical.
MedecinCoordonateur	Est un médecin intégré à l'équipe de la structure de la PAD, non prescripteur. Son rôle est d'assurer le bon fonctionnement médical de la structure, la collaboration au sein de l'équipe, de veiller à la bonne transmission des informations médicales à l'accomplissement des soins. Il participe à l'évaluation des soins, il assure la liaison et la coordination entre le médecin traitant, le médecin hospitalier et les autres intervenants. Il évalue toute demande d'admission ou de sortie.
MedecinSpecialiste	Est un médecin traitant qui a une spécialité.
ProfessionnelDeSanteParamedical	exerce des professions de la santé qui ne sont pas exercées par un médecin et un pharmacien
ProfessionnelDeSoins	Exerce des activités de soins infirmiers.
Infirmiere	est un professionnel de santé paramédical dont le rôle est de délivrer des soins infirmiers au patient sur prescription médicale ainsi que des soins d'hygiène et de prévention.
InfirmiereCoordinatrice	Elle travaille en partenariat avec l'équipe médicale et paramédicale. Elle assure le lien entre l'hôpital et l'équipe de soins.
AideSoignante	L'aide-soignante travaille en étroite collaboration et sous la responsabilité et l'encadrement de l'infirmière. Elle dispense aussi bien des soins d'hygiène et de prévention et contribue à la réalisation des actes de la vie quotidienne.
ProfessionnelDeReeducationEtReadaptation	Exercent des activités de rééducation et réadaptation.
ProfessionnelPsychosocial	Ils sont aux nombres de deux : assistante sociale et psychologue.
Psychologue	Il accompagne et soutient le patient et son entourage selon les besoins.
AssistanteSociale	Coopère étroitement avec la structure PAD. Au travers d'une enquête, elle essaie de déterminer avec le concours de la famille les problèmes sociaux et financiers ainsi que les besoins du malade. Cette évaluation sociale est préalable à l'admission. Elle aide le patient et sa famille à surmonter leurs difficultés et à développer leurs capacités propres en vue d'améliorer leurs conditions de vie sur les plans social, économique ou culturel.
DateHeure	Représente la date et l'heure exacte.
Maladie	La maladie est une altération des fonctions ou de la santé d'un organisme. se traduisant par des symptômes et des signes. Elle concerne un ou plusieurs patients.

Médicament	Substance ou composition présentant des propriétés curatives ou préventives à l'égard des maladies. Il est prescrit sur prescription médicale.
Document	Est défini comme un ensemble de papiers qui peuvent être transférés au niveau de la structure de PAD.
Demande	est un document rédigé par le médecin prescripteur et évaluée par le médecin coordonateur. Elle peut être une demande d'admission d'un patient ou une demande de sortie.
DemandeAdmissionPAD	est une demande.
DemandeSortiePAD	est une demande.
DocumentInterventionPourTraitement	est un document rédigé par le médecin traitant.
PrescriptionMedicamenteuse	Est un document indiquant les différents médicaments du plan thérapeutique proposé par le médecin traitant.
Activite	effectuée par une personne sur un patient.
Examen	Peut être un examen clinique ou complémentaire
ExamenClinique	Comprend inspection, palpation, percussion et auscultation des différentes parties du corps et de certains organes. Il est effectué par le médecin traitant pour la recherche des signes et des symptômes.
ExamenParaclinique	Examen ou technique complémentaire à l'examen clinique pour lequel le médecin peut confirmer son diagnostic et de suivre l'évolution d'une maladie sous traitement. Les techniques complémentaires comprennent notamment : les examens de laboratoire, les techniques d'imagerie médicale.
ActiviteDeReeducationEtReadaptation	sont des activités effectuées par les professionnels de rééducation et réadaptation.
SoinsInfirmiers	sont des activités effectuées par des infirmiers
SoinsTechnique	Sont des soins infirmiers
Nursing	sont des soins d'hygiène et de prévention effectuée par l'infirmière et/ou l'aide soignante.
StructurePAD	est une équipe composée de plusieurs personnes.
Specialite	représente les différentes spécialités en médecine.
(……)	(…...)

Tableau 3.5 Glossaire de termes.

3.3.2.2 Classification des concepts en hiérarchies de concepts

Cette tâche consiste à définir les relations taxonomiques entre les concepts définis dans le glossaire de termes.

En souvenir de l'existence de trois stratégies d'identification de concepts, nous avons réellement combiné les trois stratégies pour aboutir aux hiérarchies de la figure 3.4.

En outre, METHONTOLOLY propose d'utiliser quatre relations taxonomiques qui sont sous-

classe-de (*Subclass-Of*), décomposition-disjointe (*Disjoint-Decomposition*), décomposition-exhaustive (*Exhaustive-Decomposition*), et partition (*Partition*).

- Un concept C_1 est sous classe d'un autre concept C_2 si et seulement si chaque instance de C_1 est aussi une instance de C_2.
- Une décomposition disjointe d'un concept C est un ensemble de sous classes de C qui n'ont pas des instances en commun et ne couvrent pas C c.à.d. il peut exister des instances du concept C qui sont des instances d'aucun concept de la décomposition.
- Une décomposition exhaustive d'un concept C est un ensemble de sous classes de C qui couvrent C et peuvent avoir des instances en commun et des sous classes. Autrement dit, il ne peut pas exister des instances du concept C qui ne sont pas des instances d'au moins un concept de la décomposition.
- Une partition d'un concept C est l'ensemble de sous classes de C qui ne partagent pas des instances en commun et couvrent C

La figure 3.4 illustré les hiérarchies de concepts.

3.3.2.3 Construction de diagramme de relations binaires

Une relation binaire permet de relier deux concepts entre eux (un concept source et un concept cible). « *Si R est une relation entre deux concepts C1 et C2 alors pour tout couple d'instances des concepts C1 et C2, il existe une relation de type R qui lie deux instances de C1 et C2* ».

Cette tâche permet de représenter d'une manière graphique les différentes relations qui existent entre les divers concepts que se soit de même ou de différentes hiérarchies (voir figure 3.5).

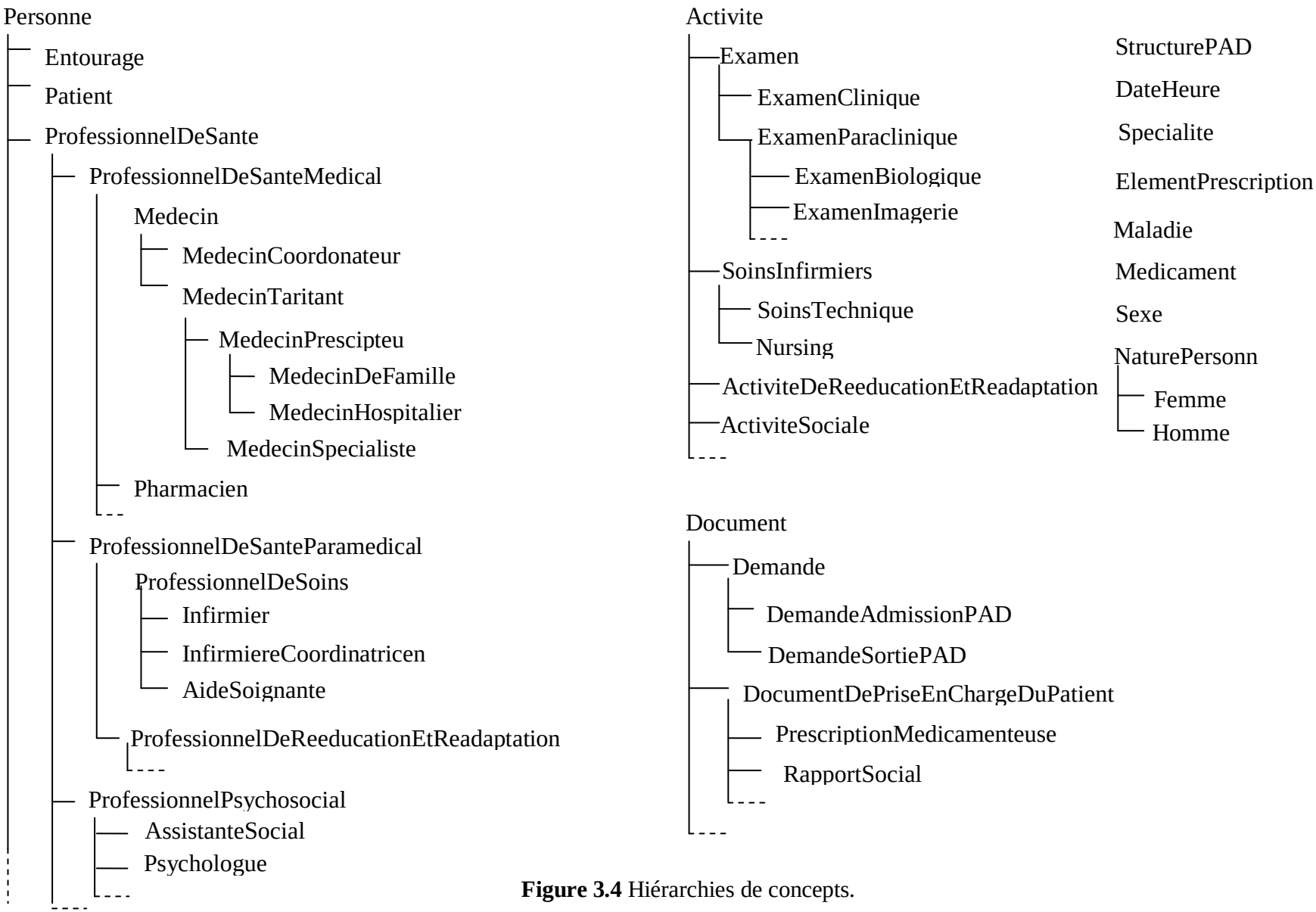

Figure 3.4 Hiérarchies de concepts.

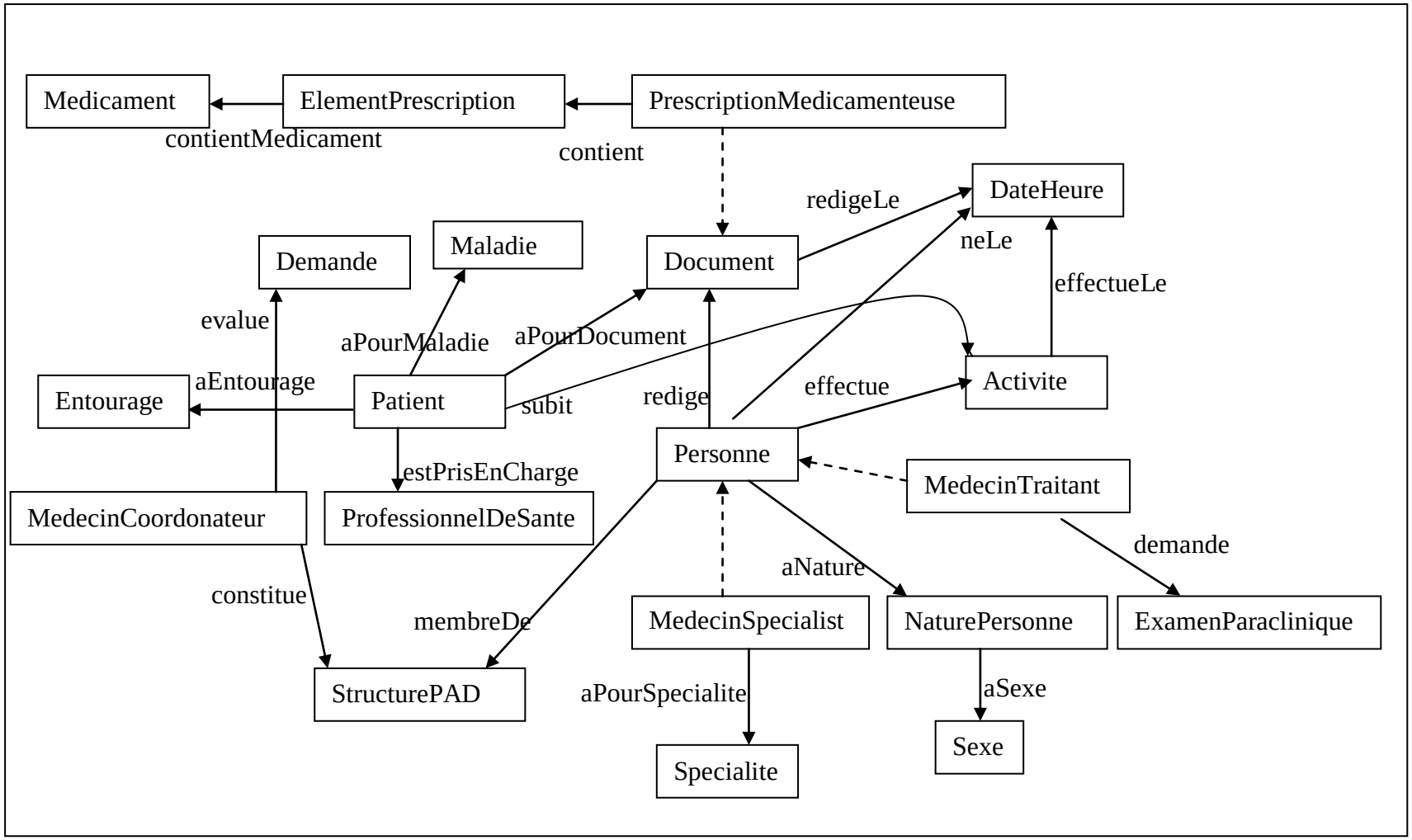

Figure 3.5 Diagramme de relations binaires.

3.3.2.4 Construction d'un dictionnaire de concepts

Une fois la taxonomie de concepts et le diagramme de relations binaires sont effectués, il faut donc spécifier les propriétés qui décrivent chaque concept de la hiérarchie dans un dictionnaire de concepts. Un dictionnaire de concepts contient tous les concepts du domaine, leurs synonymes, leurs acronymes, leurs attributs et leurs relations. Les relations spécifiées pour chaque concept sont celle où leur domaine est le concept lui-même (voir Tableau 3.6).

Nom de concept	Attributs	Les relations	Synonymes
personne	-adresse -age -e_mail -lieu_de_naissance -nom -numero_de_carte_nationnale -numero_de_securite_sociale -numero_de_telephone -prenom -situation_familiale	-aNature -effectue -membreDe -neLe -redige	Humain
Patient	-profession	-aEntourage -aPourDocument -aPourMaladie -subit estPrisEnChargePar	Malade
ProfessionnelDeSante			ActeurDeSante
Professionnel DeSanteMedical			ActeurDeSanteMedical
Medecin		-demande	
MedecinPrescripteur			
MédecinTraitant			
MedecinHospitalier			MedecinDeHopital
MedecinFamille			MedecinDeVille, MedecinLibéral, MedecinexerçantATitreLibéral
MEdecinCoordonnateur		constiue evalue	
MedecinSpecialiste		aPourSpecialite	
ProfessionnelDeSanteParaMedical			
ProfessionnelDeSoins			
InfirmiereCoordinatrice			
Infirmiere			
AideSoignante			
ProfessionnelDeReeducationEtReadaptat			

ion			
ProfessionnelPsychosocial			
psychologue			
AssistanteSociale			
Activite		effectueLe estEffectuePar estSubitPar	
Examen			
ExamenParaclinique		estDemandePar	
ReeducationeReadaptation			
Document		estDocumentDe estRedigePar redigeLe	
Demande	avis	evaluePar	
DemandeAdmissionPAD			
DemandeSortie PAD			
StructurePAD		composeDe constituePar	
Maladie		estMaldieDe	Pathologie
DateHeure	date_heure	dateActivite dateDocument dateNaissanceDe	
Specialite		estSpecialiteDe	
(…..)	(….)	(….)	(….)

Tableau 3.6 Dictionnaire de concepts.

3.3.2.5 Construction de la table des relations binaires

Le but de cette tâche consiste à construire une table de relations binaires décrites en détaille. Pour chaque relation utilisé dans le diagramme des relations binaires, nous définissons le nom de la relation, le nom des concepts sources et cibles, le nom de la relation inverse et les cardinalités source et cible (voir Tableau 3.7).

Nom de la relation	Concept source	Concept cible	Card source	Card cible	Relation inverse
estDocumentDe	Document	Patient	1..n	1..1	aPourDocument
apourSpecialite	MedecinSpecialiste	Specialite	1..n	1..1	estSpecialiteDe
appliqueSur	Activite	Patient	1..n	0..1	subit
composeDe	StructurePAD	Personne	0..n	1..n	membreDe

dateActivite	DateHeure	Activite	1..1	0..n	effectueLe
dateNaissance	DateHeure	Personne	0..n	0..1	neeLe
dateRedaction	DateHeure	Document	1..1	0..n	redigeLe
effectue	Personne	Activite	0..1	0..n	effectuePar
evalue	MedecinCoordonateur	Demande	1..1	1..n	evaluePar
redige	Personne	Document	1..1	0..n	redigePar
(…..)					

Tableau 3.7 Table des relations binaires.

3.3.2.6 Construction de la table d'attributs

La table des attributs (voir le tableau 3.8) comporte une description détaillée des attributs inclus dans le dictionnaire de concepts, et l'ensemble de contraintes et de restrictions sur ces valeurs.

Nom attribut	Type de valeur	Card (max/min)	Valeur par défaut	Domaine de valeurs
numero_de_carte_nationnale	string	1..1	-	-
nom	string	1..1	-	-
prenom	string	1..n	-	-
e_mail	string	0..n	-	-
lieu_de_naissance	string	1..1	-	-
age	int	1..1	-	-
adresse	string	1..1	-	-
situation_familiale	string	1..1	-	marie, celibataire, veuf, divorce
numero_de_securite_sociale	string	0..1	-	-
numero_de_telephone	string	0..n	-	-
profession	string	0..1	-	-
Position professionnelle	string	1..1	-	(travailleur, retraité, sans emploi)
avis	string	1..1		(accepté ,refusé)
date_heure	datetime	0..1	-	-
nom_maladie	string	1..1	-	-
Nom_medicament	string	1..1	-	-
(…..)				

Tableau 3.8 Table des attributs.

3.3.2.7 Construction de la table des axiomes

Dans cette étape, nous définissons les concepts au moyen d'expressions logiques. Pour chaque axiome, il faut spécifier la description de l'axiome en langage naturel, l'expression logique qui décrit formellement l'axiome en logique du premier ordre, les concepts, les relations et les

variables utilisées (voir le tableau 3.9).

Nom du concept	Description	Expression logique	Concepts	Relations	variables
Personne	Chaque personne est soit un patient, soit un professionnel de sante, soit entourage.	∀xPersonne(X) ⇒Patient(X)∨ ProfessionnelDeSante(X) ∨ Entourage	Personne, Patient, professionnelDeSante Entourage		x
Professionnel DeSante	peut être un professionnel de sante médicale, paramédicale ou psychosociale	∀ x Personne(x) ⇒ ProfessionnelDeSanteMedical(x) ∨ ProfessionnelDeSanteParamedical (x) ∨ professionnelPsychosociale	ProfessionnelDeSante Medical, ProfessionnelDeSante Paramedical, ProfessionnelPsychosociale	-	x
	Professionnel de sante est disjoint avec patient	∀x ProfessionnelDeSante (x) ⇒ not Patient(x)	ProfessionnelDeSante , Patient		x
ExamenCliniq ue	Un examen clinique est effectué par le médecin traitant	∀x ExamenClinique(x) ⇒∃y MedecinTraitant(y) ∧ effectue (y,x)	ExamenClinique, MedecinTraitant	effectue	x, y
(…)	(…)	(…)	(…)	(…)	(..)

Tableau 3.9 Table des axiomes.

3.3.2.8 Construction de la table des instances

La table des instances décrit les instances connues; qui sont déjà identifiées dans le dictionnaire de concepts. Pour chaque instance, il faut spécifier le nom de l'instance, le nom du concept où elle appartienne, ses attributs et les valeurs qui lui y sont associés. Le tableau 3.10 ci-après illustre quelques instances créées.

Nom de l'instance	Nom du concept	Attributs	Valeurs
Masculin Feminin	Sexe		
Nephrologie Cardiologie Neurologie	Specialite		
Hypertension Diabete	Maladie		
(..)		(…)	(..)

Tableau 3.10 Table des instances.

3.3.3 Formalisation

Comme cité auparavant, dans cette étape, nous utilisons le formalisme de la logique de description pour formaliser le modèle conceptuel que nous avons obtenu dans l'étape de conceptualisation. Le résultat est une base de connaissances en logique de description *SHOIN(D)* composé de deux parties T-BOX (voir le tableau 3.11) et A-BOX (voir le tableau 3.12).

3.3.3.1 Construction de TBox

//---les axiomes suivants permettent de décrire et définir les classes de l'ontologie ---//

Personne⊆T∩(∃nom.String)∩(∃prenom.String)∩(∃lieu_de_naissance.String)∩(∃adresse.String)∩(∃age.String)∩(∃numero_de_securite_sociale.String)∩(∃numero_de_telephone.String)∩(∃situation_familiale.String)∩(=1neeLe)∩(≥0effectue)∩(=1aNature)∩(≥0redige)∩(≥0membreDe)

Patient⊆Personne(∃aPourDocument.Document)∩(∃subit.Activité)∩(≥0aEntourage)∩(∃aPourMaladie.Maladie) ∩(∃estPrisEnChargePar.ProfessionnelDeSante)∩(=1membreDe)

Entourage⊆Personne∩(∃type_entourage.String)∩(∃estEntourageDe.Patient)

ProfessionnelDeSante⊆Personne∩(≥0prendEnCharge)

ProfessionnelDeSanteMedical⊆ ProfessionnelDeSante

Medecin⊆ ProfessionnelDeSanteMedical

MedecinCoordonateur⊆ Medecin∩(∃constiue.StructurePAD) ∩(∃evalue.Demande)

MedecinTraitant⊆ Medecin ∩(≥0demande)

MedecinPrescipteur⊆MedecinTraitant∩(MedecinHospitalier∪MedecinDeFamille)∩(∀redige.Demande)∩(∃redige.Demande)

MedecinDeFamille⊆MedecinPrescipteur

MedecinHospitalier⊆MedecinPrescipteur

MedecinSpecialiste⊆(=1 aPourSpecialite)

ProfessionnelDeSanteParamedical⊆ProfessionnelDeSante

ProfessionnelDeReeducationEtReadaptation⊆ProfessionnelDeSanteParamedical)∩(∀effectue.ActiviteDeReeducationEtReadaptation)∩(∃effectue.ActiviteDeReeducationEtReadaptation)

ProfessionnelDeSoins⊆ProfessionnelDeSanteParamedical

AideSoignante⊆ ProfessionnelDeSoins∩(∀effectue.Nursing) ∩(∃effectue.Nursing)

Infirmiere⊆ ProfessionnelDeSoins∩(∀effectue.SoinsInfirmiers) ∩(∃effectue.SoinsInfirmiers)

InfirmiereCoordinatrice⊆ProfessionnelDeSoins

ProfessionnelPsychosocial⊆ProfessionnelDeSante

AssistanteSociale⊆ProfessionnelPsychosocial∩(∀effectue.ActiviteSociale)(∀redige.RapportSoci

al) ∩(∃redige.RapportSocial)

Psychologue⊆ProfessionnelPsychosocial

NaturePersonne≡ T∩(Femme ∪Homme)

Femme≡ (ɔ aSexe.Feminin)

Homme≡ (ɔ aSexe.Masculin)

Activite⊆T∩(∃note.string)∩ (=1effectueLe)∩(≥effectuePar) ∩ (=1appliqueSur)

ActiviteDeReeducationEtReadaptation⊆Activite∩(∀effectuePar.ProfessionnelDeReeducationEtReadaptation) ∩(∃effectuePar.ProfessionnelDeReeducationEtReadaptation)

ActiviteSociale⊆ (∀effectuePar.AssistanteSociale) ∩ (∀effectuePar.AssistanteSociale)

Examen⊆ Activite

ExamenClinique⊆ Examen

ExamenParaclinique⊆Examen∩(∃estDemandePar.MedecinTraitant)

SoinsInfirmiers⊆ Activite

Nursing⊆ SoinsInfirmiers

SoinsTechnique⊆ SoinsInfirmiers

Document ⊆ T∩ (∃redigeLe.DateHeure) ∩(∃ redigePar. personne) ∩ (∃concerne. patient)

Demande⊆ (∃avis.string)∩(∃revaluePar.MedecinCoordonateur)∩ (=1redigePar)

(∀redigePar.MedecinPrescripteur)

StructurePAD⊆ (∃composeDe.Personne) ∩(∃constituePar. MedecinCoordonateur)

Sexe≡{masculin,feminin}

PrescriptionMedicamenteuse⊆DocumentDePriseEnChargeDuPatient∩(>=1contien)∩(∃estRedigePar.MedecinTraitant)∩(∀estRedigePar.MedecinTraitant)

Specialite⊆ T∩(∀estSpecialiteDe.MedecinSpecialiste)

(…..)

//---les axiomes suivants permettent d'exprimer les classes disjoints---//

(Examen∩SoinsInfirmiers∩ ActiviteDeReeducationEtReadaptation)⊆ ⊥

(ExamenClinique∩ExamenParaclinique) ⊆ ⊥

(Femme∩Homme)⊆ ⊥

(Entourage∩Patient∩ProfessionnelDeSante) ⊆ ⊥

(ProfessionnelDeSanteMedical∩ProfessionnelDeSanteParamedical∩ProfessionnelPsychosocial) ⊆⊥

(Activite∩DateHeure∩Document∩ElementPrescription∩Maladie∩Medicament∩NaturePersonne∩Personne∩Sexe∩Specialite∩StructurePAD) ⊆ ⊥

(…...)
//---les axiomes suivants permettent de définir le domaine, le co-domaine et les caractéristiques des relations---//
∃aPourDocument.T⊆Patient
T⊆∀ aPourDocument.Document
aPourDocument≡estDocumentDe^{-}
T⊆ ≤1 estDocumentDe
∃contient.T ⊆PrescriptionMedicamenteuse
T⊆∀contient.ElementPrescription
contient ≡faitPartie^{-}
T⊆ ≤1 faitPartie
(……..)
//---les axiomes suivants permettent de définir le domaine et le co-domaine d'un exemple d'un attribut qui est nom. La même chose est réalisée sur toutes les attributs de chaque concept---//
∃nom.T⊆Personne
T⊆∀nom.string
∃ age.T⊆Personne
T⊆∀ age.int
(…….)

Tableau 3.11 Axiomes de classes et de rôles.

3.3.3.2 Construction d'ABox

Sexe(Masculin)
Sexe(Feminin)
Specialite(Neurologie)
Specialite(Cardiologie)
Maladie(Hypertension)
Maladie(Diabete)
(……)

Tableau 3.12 Assertions d'individus.

Jusqu'à présent, nous avons attribué un modèle formel à l'ontologie OntoPAD dans cette section. Dans la section suivante, nous détaillons son implémentation en utilisant les outils qui ont été développés pour cet objectif.

3.3.4 Opérationnalisation

L'ontologie que nous avons obtenue à ce stade est une ontologie formelle, il nous reste à opérationnaliser cette ontologie pour qu'elle soit concrètement manipulable dans un système informatique. Par conséquent, elle doit être spécifiée dans un langage de représentation de connaissance doté de capacités d'inférences. Pour ce faire, nous utilisons le logiciel PROTÉGÉ-OWL 3.3.1.

3.3.4.1 Choix d'un langage de spécification

Notre choix a été orienté vers OWL qui est le langage standard de représentation et de spécification de l'ontologie, comparé à ses prédécesseurs RDFS et DAML-OIL qui sont insuffisants pour codifier l'ontologie de ce mémoire en terme de fonctionnalités sémantiques. En plus, le codage d'une ontologie sous forme OWL présente l'avantage de la rendre réutilisable.

En effet, notre ontologie est codée exactement en 'OWL DL' pour les raisons suivantes :

- OWL Lite est d'expressivité minimale et ne permet d'exprimer que des hiérarchies et de contraintes simples de cardinalité 0 ou 1, tandis qu'OWL DL permet d'exprimer des cardinalités multiples.
- OWL Full offre un plus haut niveau d'expressivité que demande l'ontologie de ce projet, tandis qu'OWL DL offre un niveau d'expressivité assez suffisant tout en maintenant la complétude de calculs (toutes les conclusions sont calculées) et la décidabilité (tous les calculs sont effectués en un temps fini).

3.3.4.2 Edition de l'ontologie et Génération du Code OWL sous PROTÉGÉ

La construction de l'ontologie est une tâche complexe, même pour implémenter une petite ontologie, celle-ci va prendre plusieurs lignes de code et nécessite un grand effort et du temps. Notamment, si cette ontologie est codée directement par le développeur en langage d'ontologie sans faire recours à aucun outil. Pour cela, PROTÉGÉ est conçu pour libérer l'ontologiste de cette complexité et générer automatiquement la structure de l'ontologie créée en OWL-DL. Une présentation de l'outil PROTEGE, ainsi que les étapes d'édition de l'ontologie OntoPAD seront fournis en annexe A. De même, Le code OWL d'OntoPAD (non instanciée) complet généré sera donné à l'annexe B.

La figure 3.6 montre alors la hiérarchie de l'ontologie construite manuellement, identique à celle de la phase de conceptualisation.

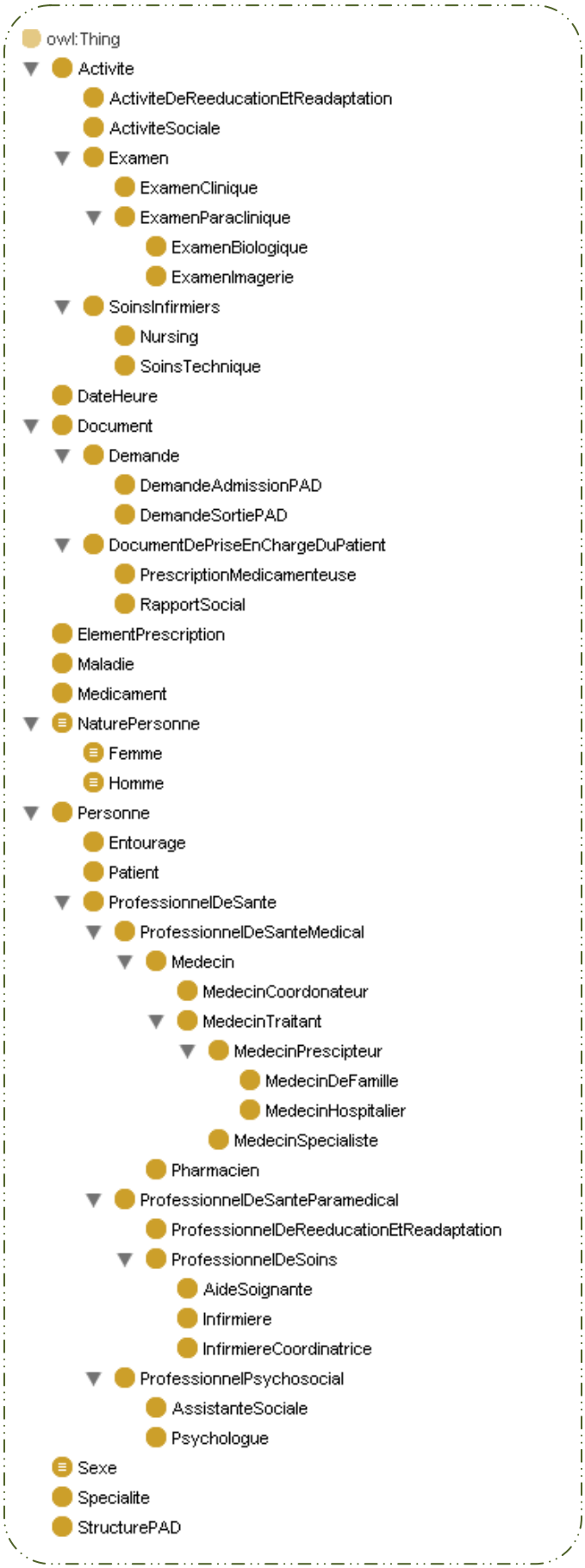

Figure 3.6 Hiérarchie de l'OntoPAD sous PROTÉGÉ**.**

Nous donnons dans la figure 3.7 un aperçu de l'ontologie OntoPAD édité sous PROTÉGÉ et dans la figure 3.8 un extrait du code OWL généré par PROTÉGÉ.

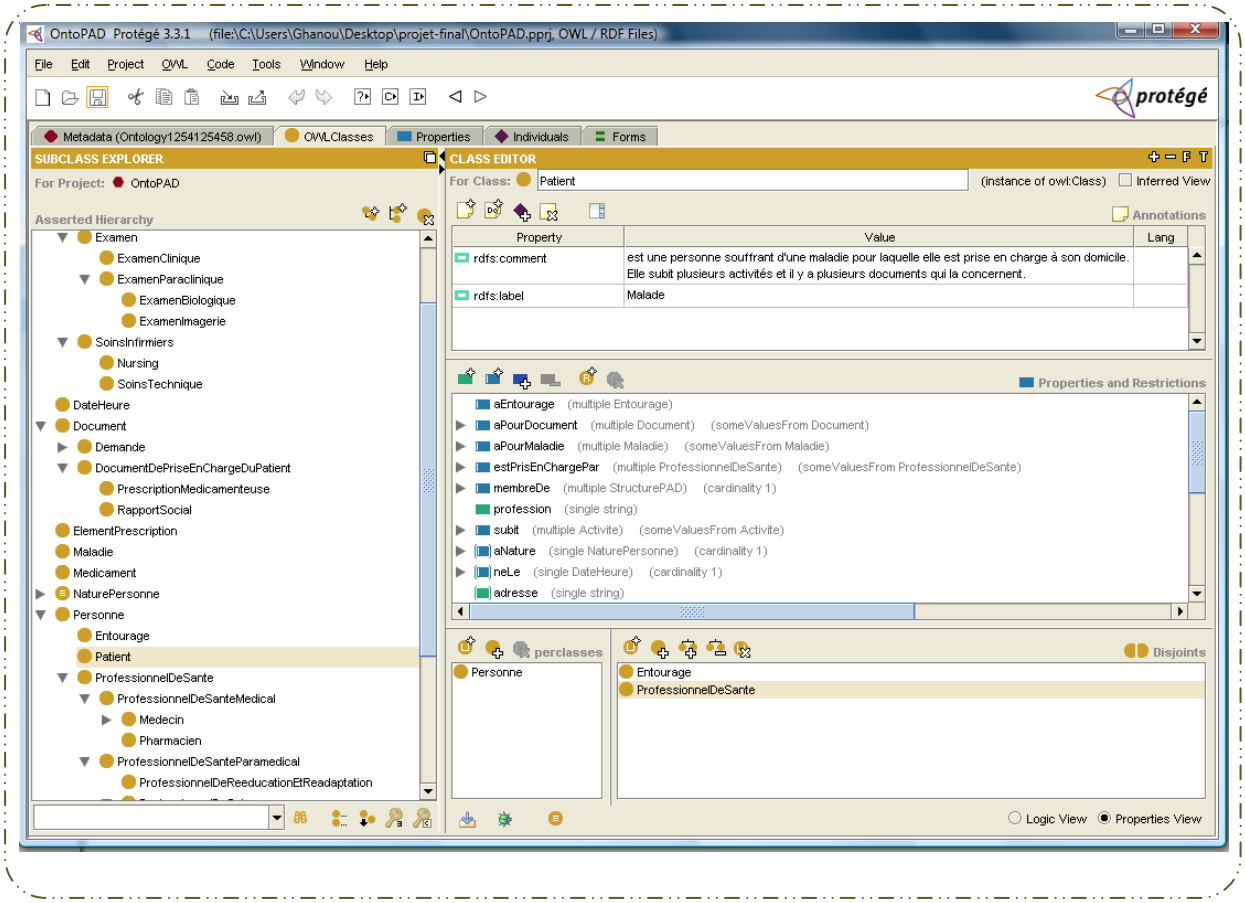

Figure 3.7 OntoPAD édité sous l'outil PROTÉGÉ.

```
…
<owl:DatatypeProperty rdf:ID="adresse">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="age">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="&xsd;int"/>
    </owl:DatatypeProperty>
    <owl:ObjectProperty rdf:ID="date_naissance">
        <rdfs:domain rdf:resource="#DateHeure"/>
        <rdfs:range rdf:resource="#Personne"/>
        <owl:inverseOf rdf:resource="#nee_le"/>
    </owl:ObjectProperty>
    <owl:Class rdf:ID="DateHeure">
        <owl:disjointWith rdf:resource="#Personne"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:ID="dateheure">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#DateHeure"/>
        <rdfs:range rdf:resource="&xsd;dateTime"/>
    </owl:DatatypeProperty>
    <owl:DatatypeProperty rdf:ID="lieu_de_naissance">
```

```
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
    <owl:ObjectProperty rdf:ID="nee_le">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="#DateHeure"/>
        <owl:inverseOf rdf:resource="#date_naissance"/>
    </owl:ObjectProperty>
    <owl:DatatypeProperty rdf:ID="nom">
        <rdf:type rdf:resource="&owl;FunctionalProperty"/>
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
       <owl:Class rdf:ID="Personne">
        <owl:disjointWith rdf:resource="#DateHeure"/>
    </owl:Class>
    <owl:DatatypeProperty rdf:ID="prenom">
        <rdfs:domain rdf:resource="#Personne"/>
        <rdfs:range rdf:resource="&xsd;string"/>
    </owl:DatatypeProperty>
……
```

Figure 3.8 Fragment du code OWL généré par PROTÉGÉ.

3.3.5 Évaluation de l'ontologie

Nous utilisons le moteur d'inférence Racer (la configuration du RACER est détaillée en annexeB) pour tester l'OntoPAD. Il est conçue pour raisonner sur les logiques de descriptions et accepte en entrée un fichier OWL. Une fois l'ontologie chargée, il effectue les inférences sur la TBox et la ABox.

Les principaux services offerts par RACER sont : le test de consistance (satisfiabilité, cohérence) et le test de classification (subsomption).

3.3.5.1 Test de consistance

Le test de consistance fournit par RACER est effectué en se basant sur la description des classes (conditions). Il permet de s'assurer qu'aucune définition d'une classe est contradictoire avec une autre (l'inexistence des classes contradictoires) c à d vérifier que pour chaque classe, il faut qu'il existe au moins un individu membre de cette classe. Une classe est considérée comme étant inconsistante si elle ne peut avoir aucune instance.

Pour effectuer ce test, il faut aller au menu « OWL/check consistency», ou bien en cliquant sur le bouton 'check consistency' de la barre d'outils Protégé-OWL. Ce même test peut être appliqué sur chaque concept à part, en cliquant par le bouton droit de la sourie sur un concept et en choisissant l'option « check concept consistancy ».

Le résultat de ce test, comme le montre la figure 3.9, indique que toutes les classes sont consistantes.

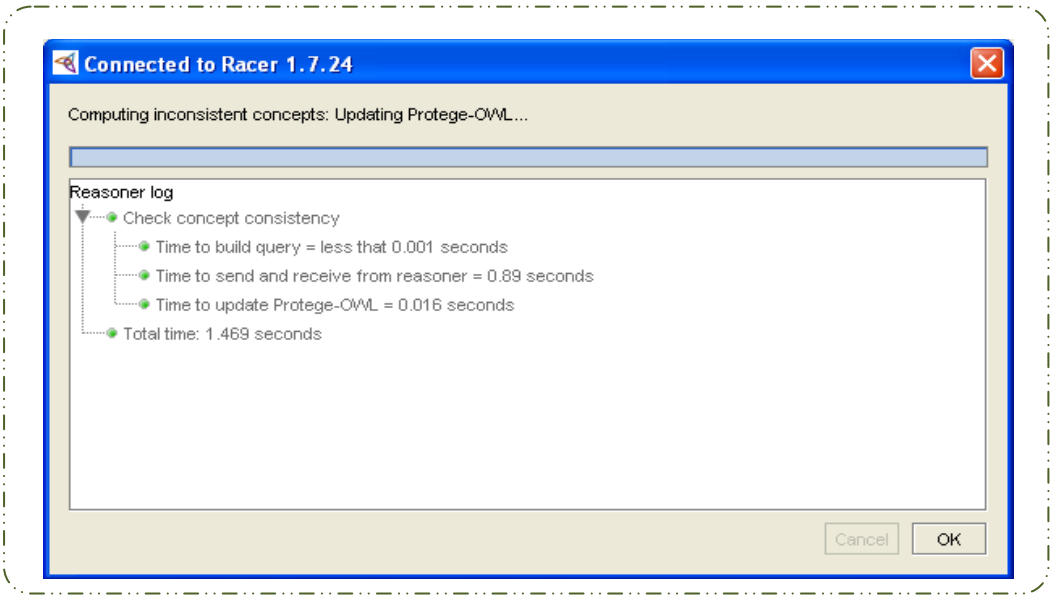

Figure 3.9 Résultats du test de consistance.

3.3.5.2 Test de classification

Le test de classification (subsomption) permet de tester si une classe est une sous classe d'une autre classe ou non. Lorsque ce test est invoqué, le test de consistance est d'abord effectué pour toutes les classes de l'ontologie parce que les classes inconsistantes ne peuvent pas être classées correctement. Une fois le test de classification est effectué sur la hiérarchie des classes contenant les expressions logiques, il est possible pour le classifieur d'inférer une nouvelle hiérarchie « inferred ontology class hierarchy » qui est une hiérarchie où les classes sont classifiées selon la relation superclasse/sousclasse.

Pour effectuer ce test (classification automatique de l'ontologie), il faut aller au menu « OWL/ Classify taxonomy» ou bien en cliquant sur le bouton 'Classify taxonomy' de la barre d'outil PROTÉGÉ-OWL. Le résultat de ce test est affiché graphiquement par PROTÉGÉ-OWL. Il consiste en une 'inferred hierarchy' calculé à partir de l'hiérarchie construite manuellement par l'ontologiste. La différence entre les deux hiérarchies est soulignée par le système si elle existe (Si une classe est reclassifiée c.à.d. que ses superclasses sont changées, elle est affichée en couleur bleu dans la hiérarchie calculée).

La nouvelle hiérarchie est visualisée, telle qu'illustrée à la figure 3.10, juste à coté de 'asserted hierarchy'. Nous remarquons qu'aucune suggestion n'est produite par le raisonneur et que 'asserted hierarchy' et 'inferred hierarchy' sont identiques. Cela est dû à notre ontologie qui n'est pas large.

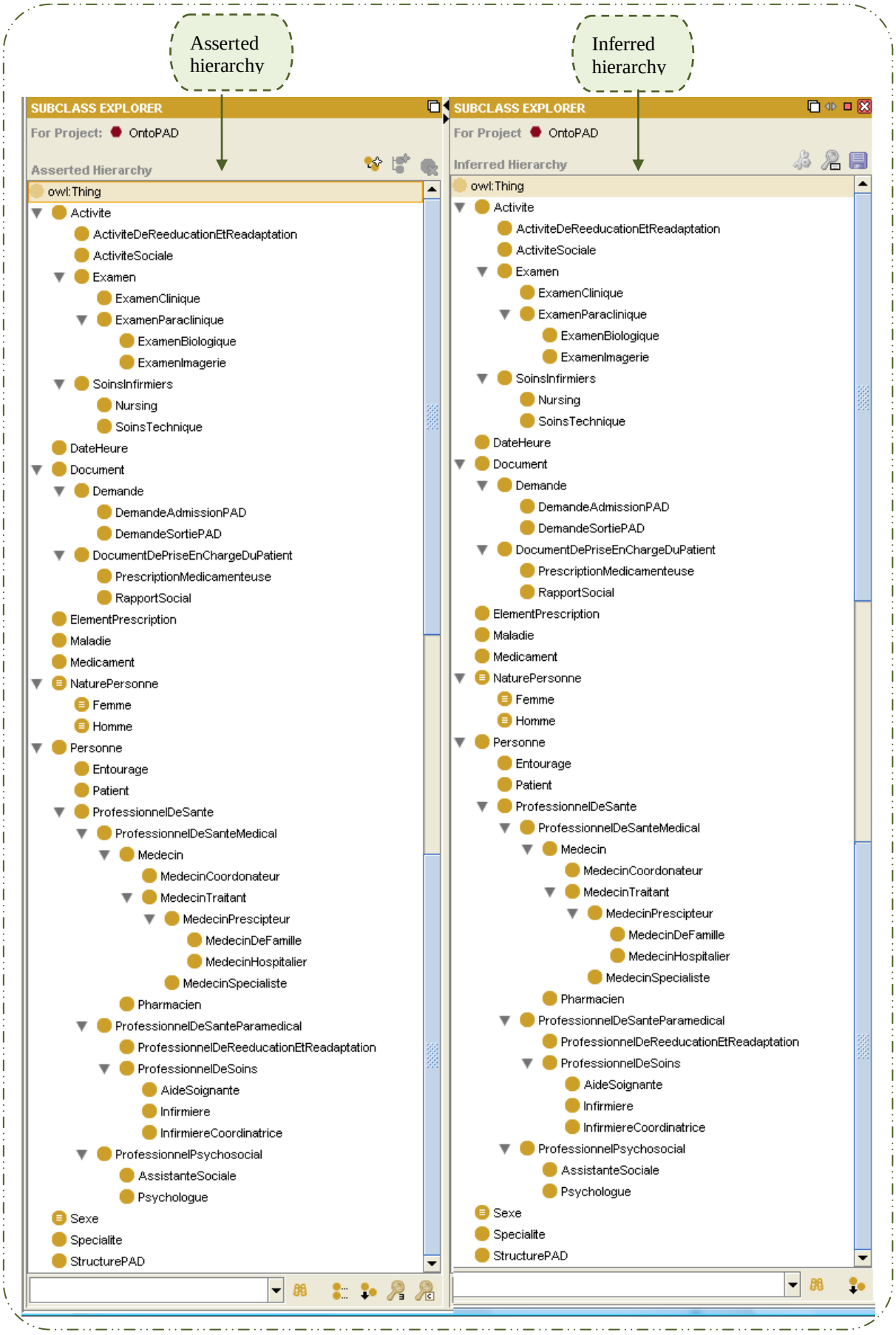

Figure 3.10 Résultats du test de classification de l'ontologie.

Indépendamment de RACER, PROTÉGÉ-OWL fournit un mécanisme pour exécuter une liste de tests configurables sur l'ontologie que nous sommes en train d'éditer. Ces tests sont des petits programmes JAVA prédéfinis (liste disponible à travers le menu « OWL », test settings), qui peuvent être étendus par les programmeurs. Ils permettent principalement de vérifier des conditions spécifiées dans l'ontologie. A titre d'exemple, la propriété de disjonction des classes sœurs (les classes qui ont la même superclasse) et l'absence des métaclasses qui rend l'ontologie en OWL-FULL doivent être vérifiées.

Pour accéder à ces différents tests, il faut aller au menu « OWL/preferences». Dans la fenêtre qui s'affiche, nous choisissons l'onglet 'tests' comme le montre la figure 3.11. Pour exécuter ces tests, il suffit d'aller au menu OWL et sélectionner RunOntologyTests.

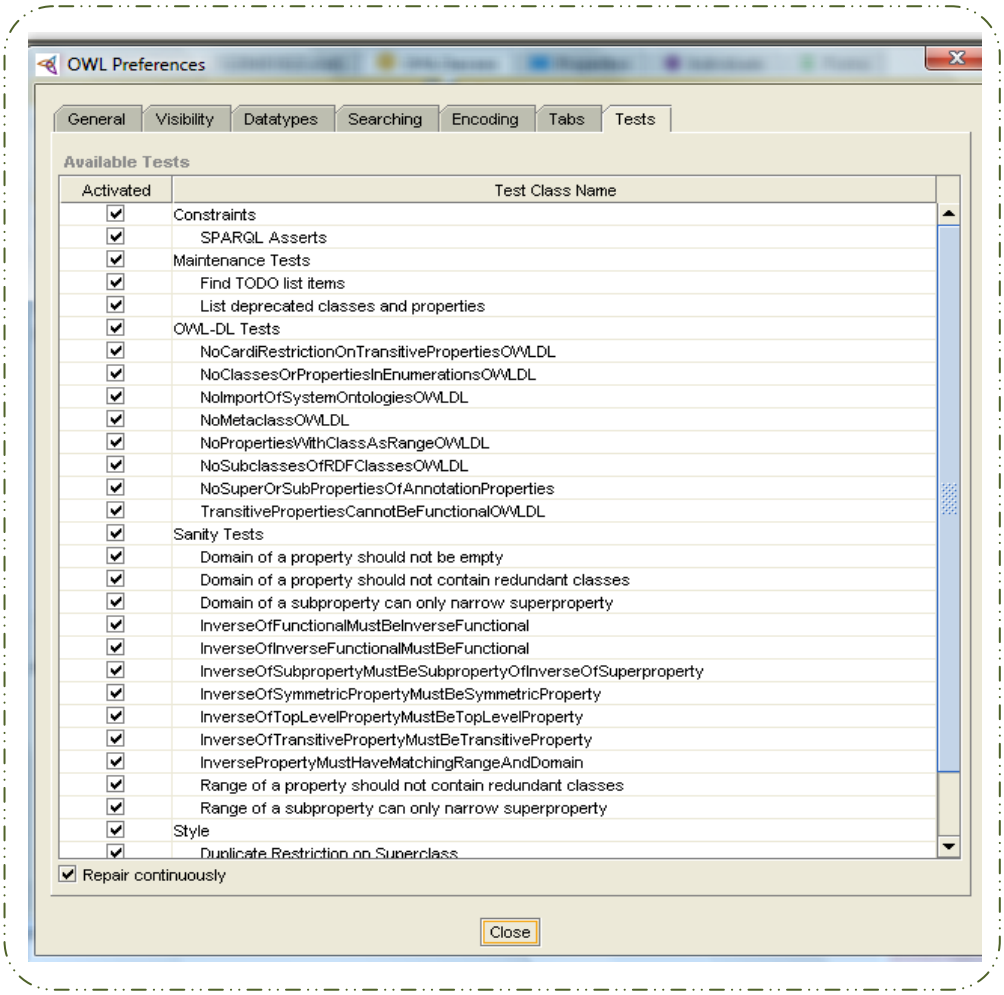

Figure 3.11 Extrait des tests qui peuvent être vérifiés par PROTÉGÉ.

Si une propriété dans l'ontologie enfreint n'importe qu'elle condition prédéfinie, un cadre contenant les colonnes suivantes est affiché : le type du résultat du test (erreur, warning, etc.), la source d'erreur (peut être une classe ou une propriété) et le résultat du test (un message qui décrit le résultat obtenu). L'éditeur fournit ainsi un bouton 'repair' qui permet de réparer la source de la violation automatiquement.

La figure 3.12 montre un genre d'erreur que nous avons effectué lors de l'édition de l'ontologie. Il s'agit de la violation de la propriété de disjonction entre les sous classes directes de la classe universelle (Thing). Cette erreur peut être réparée en utilisant le bouton 'repair' qui va ajouter la propriété de disjonction oubliée.

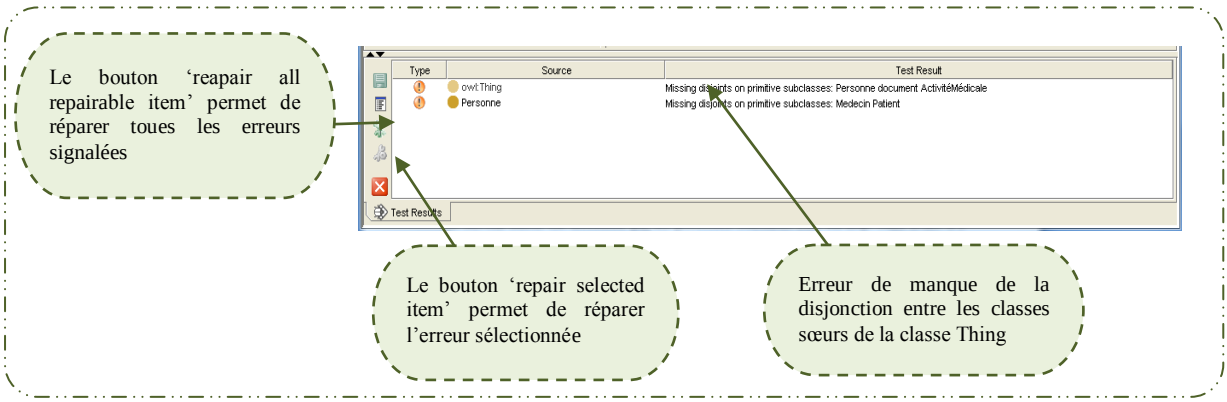

Figure 3.12 Résultats des tests de PROTÉGÉ.

3.4 Conclusion

À travers tous ce que nous avons réalisé dans ce chapitre, nous pouvons dire que le processus suivi pour la construction d'OntoPAD nous a permis de réussir finalement à construire une ontologie de domaine, lourde, de granularité ni assez fine ni assez large et formelle. L'atout majeur d'OntoPAD consiste à sa cohérence ainsi que sa consistance montrées par les tests effectués sur elle. De ce fait, elle est prête à une future évaluation au sein de n'importe quelle application en rapport avec le domaine qu'elle modélise.

CONCLUSION GÉNÉRALE ET PERSPECTIVES

1 Bilan

Notre contribution consista en la construction d'une ontologie de domaine de la PAD. Pour ce faire, nous avons eu recours à un processus basé sur certains travaux intéressants, trouvés dans la littérature, notamment la méthodologie METHONTOLOGY. Bien évidemment, nous étions guidés dans notre travail par plusieurs principes largement acceptés par la communauté des ontologistes. Une fois l'ontologie conceptuelle mise au propre, nous avons passé à sa formalisation en nous appuyant sur une logique très expressive *SHOIN(D)* et son opérationnalisation avec l'outil PROTÉGÉ qui nous a permis de générer automatiquement le code OWL-DL d'OntoPAD. Enfin, des tests de consistance et de classification ont été appliqués sur l'ontologie opérationnelle par le biais du raisonneur RACER. Cela a donné naissance à une ressource ontologique cohérente prête à une évaluation et une exploitation par les différents acteurs de la PAD de même qu'au sein d'un système informatique.

Le travail sur l'ontologie de ce mémoire nous a permis de marquer que :

- *La construction des ontologies est une tâche ardue* : le processus que nous avons suivi pour la construction de l'ontologie 'OntoPAD' n'était point linéaire. Plusieurs versions ont été construites avant la convergence à une première version plus ou moins complète.
- *L'ingénierie ontologique est pluridisciplinaire à moins que l'ingénieur ontologique soit également expert du domaine* : nous nous sommes confrontés à plusieurs difficultés dans l'extraction des connaissances du domaine de la PAD vu le manque de traces des différents acteurs dans une organisation réelle, d'une part. D'autre part, le secteur médical algérien n'a pas largement adopté la PAD. Conséquemment, les professionnels de santé supposés experts du domaine manquaient d'une conception claire d'une telle organisation de soins.
- *La phase de conceptualisation, cœur de la méthode METHONTOLOGY, sur laquelle nous nous sommes basés pour la conceptualisation de l'OntoPAD présente quelques limites :* malgré qu'elle soit très détaillée en termes de représentations intermédiaires construites, il reste que nous ne savons toujours pas précisément comment obtenir le contenu de ces structures en particulier les taxonomies de concepts. Pour enfreindre ce type de limites, nous avons utilisé un ensemble de critères qui nous ont guidés lors de la conception et de l'évaluation de la qualité de l'ontologie.

- *le cycle de vie de l'ontologie n'est pas suivi complètement :* ce que nous avons pu réaliser réellement est le développement de l'ontologie. Rappelons que le processus de construction est intégré au cycle de vie de l'ontologie. Ce qui nous reste à faire est de compléter les phases manquantes de ce dernier qui sont en effet nos perspectives.

2 Perspectives

Finalement, nous envisageons comme perspectives du travail réalisé dans ce mémoire:

- L'évaluation d'OntoPAD à la limite par les experts du domaine. Leur point de vue concernant le contenu de l'ontologie a un impact très important pour vérifier sa complétude. Par la suite, la mettre au point au sein d'une application concrète de la prise en charge des patients à domicile. Il est fort probable que l'ontologie soit révisée, raffinée et complétée.
- Il est peu probable qu'OntoPAD soit suffisante pour représenter toutes les connaissances de ce domaine qui évolue sans cesse. Il s'agit donc d'étendre cette ontologie et suivre son évolution lors de la configuration des services qui doivent s'adapter à l'évolution des besoins des utilisateurs, ou encore de nouveaux services constamment en cours de développement, de nouveaux acteurs qui peuvent intervenir ainsi que de nouvelles tâches réalisées auprès du patient,.. Etc.
- Compléter le processus suivi pour bâtir l'ontologie par l'ajouter d'une phase d'acquisition de connaissances semi-automatique.
- Créer un outil de navigation et de visualisation de l'ontologie.

RÉFÉRENCES BIBLIOGRAPHIQUES

ANAES. (2003). Agence nationale d'accrédiation et d'évaluation en santé. Complément au manuel d'acrédiation: l'hospitalisation à domicile. Direction de l'accrédiation.

ANAES. (2004). Agence nationale d'accrédiation et dévaluation en santé. Méthode d'élaboration d'une démarche de soins type à domicile pour une population définie de personnes en situation de dépendance. Service communication.

ARH. (2006). Schéma Régional d'Organisation Sanitaire.Hospitalisation à domicile. Guyane.

Arpírez, J. C., Corcho, O., Fernández-López, M., & Gómez-Pérez, A. (2003). *WebODE in a nutshell.* AI Magazine. To be published in 2003.

Arpírez, J. C., Gómez-Pérez, A., Lozano, A., & Pinto, H. S. (1998). (ONTO)2Agent: An ontology-based WWW broker to select ontologies. Dans A. Gómez-Pérez, & R. V. Benjamins (Éd.), *ECAI'98 Workshop on Applications of Ontologies and Problem-Solving Methods*, (pp. 16–24). Brighton, United Kingdom.

Aussenac-Gilles, N., Biébow, B., & Szulman, N. (2000). Revisiting Ontology Design: a method based on corpus analysis. In R. Dieng, & O. Corby (Ed.), *12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00)* (pp. 172-188). Berlin, Germany: Springer-Verlag.

BAADER, F., MCGUINESS, D., NARDI, D., & PATEL-SCHNE, P. (2003). *The Description Logic Handbook : Theory, Implementation and Applications.* Cambridge University Press.

BACHIMONT, B. (2000). Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en ingénierie des connaissances. In R. TEULIER, J. CHARLET, & P. TCHOUNIKINE (Ed.), *Coordinateurs, Ingénierie des connaissances.* Paris : L'Harmattan.

BACHIMONT, B., ISAAC, A., & TRONCY, R. (2002). Semantic commitment for designing ontologies :A proposal. *13th International Conference on Knowledge Engineering and Knowledge Management (EKAW)* (pp. 114-121). Springer.

Baneyx, A. (2007). *Construire Une Ontologie De La Pneumologie : Aspects Théoriques, Modèles Et Expérimentations.* Thèse de doctorat, UNIVERSITÉ PIERRE ET MARIE CURIE - PARIS 6, LABORATOIRE INSERM UMR_S 872 – SANTÉ PUBLIQUE ET INFORMATIQUE MÉDICALE, France.

Bechhofer, S., Horrocks, I., Goble, C., & Stevens, R. (2001). OilEd: a reasonable ontology

editor for the Semantic Web. Dans F. Baader, G. Brewka, & T. Eiter (Éd.), *Joint German/Austrian conference on Artificial Intelligence (KI'01)* (pp. 396–408). (Lecture Notes in Artificial Intelligence LNAI 2174) Springer-Verlag.

Bernaras, A., Laresgoiti, I., & Corera, J. (1996). Building and reusing ontologies for electrical network applications. In W. Wahlster (Ed.), *European Conference on Artificial Intelligence (ECAI'96)* (pp. 298–302). Chichester, United Kingdom: John Wiley and Sons.

Blazquez, M., Fernandez, M., Garcia-Pinar, J., & Gomez-Perez, A. (1998). Building Ontologies at the Knowledge Level using the Ontology Design Environment. *In Proceedings of the Banff Workshop on Knwoledge Acquisition for Knwoledge-based Systems.*

Bodenreider, O., & Burgun, A. (2005). BIOMEDICAL ONTOLOGIES. Dans H. Chen, S. Fuller, W. R. Hersh, & C. Friedman (Éd.), *Medical informatics: Advances in knowledge management and data mining in biomedicine.* Springer-Verlag (in press).

Borgo, S., Guarino, N., & Masolo, C. (1996). Stratified Ontologies: the case of physical bjects. *ECAI96. Workshop on Ontological Engineering* , 5-15.

Borst, W. N. (1997). Construction of Engineering Ontologies. Centre for Telematica and Information Technology, University of Tweenty, Enschede, The Netherlands.

Bricon-Souf, N., Anceaux, F., Bennanib, N., Dufresne, E., & Watbled, L. (2005). A distributed coordination platform for home care: analysis, framework and prototype. *International Journal of Medical Informatics , 74*, 809—825.

Corcho, O., Fernandez-Lopez, M., & Gomez-Perez, A. (2003). Methodologies, tools and languages for building ontologies. Where is their meeting point?. *Data & Knowledge Engineering. 46*, pp. 41–64. Elsevier.

Dieng, R., Corby, O., Gandon, F., Giboin, A., GOLEBIOWSKA, J., MATTA, N., et al. (2001). *Méthodes et outils pour la gestion des connaissances : une approche pluridisciplinaire du knowledge management.* Dunod Edition Informatiques Séries Systèmes d'Information (2ième édition).

Dieng-Kuntz, R., Minier, D., Ružicka, M., Corby, F., Corby, O., & Ala, L. (2006). Building and using a medical ontology for knowledge management and cooperative work in a health care network. *Computers in Biology and Medicine , 36*, 871–892.

DirectionRégionaleDuServiceMédical, URCAMdesPaysDeLaLoire, & CRAMdesPaysDeLaLoire. (2003). *Etude médico-administrative des structures administratives d'hospitalisation à domicile dans les pays de la Loire.*

Fernández-López, M., Gómez-Pérez, A., & Juristo, N. (1997). METHONTOLOGY: From Ontological Art Towards Ontological Engineering. *Spring Symposium on Ontological Engineering of AAAI* , 33–40.

Fournier-Viger, P. (2005). *Un modèle de représentation des connaissances à trois niveaux de sémantique pour les systèmes tutoriels intelligents.* Master's thesis, Université de Sherbrooke, Sherbrooke, Canada.

FÜRST, F. (2004). *Contribution à l'ingénierie des ontologies : une méthode et un outil d'opérationalisation.* Thèse de doctorat, LINA (Laboratoire d'Informatique de Nantes Atlantique),Université de Nantes.

FÜRST, F. (2002). *L'ingénierie ontologique.* Rapport de recherche n 02-07, Institut de Recherche en Informatique, université de Nantes.

GANDON, F. (2006). Ontologies informatiques. *Interstices, Journal en ligne de l'INRIA* .

Garf, B. (1996). *Lexique de philosophie.* Paris: Edition du Seuil.

Gómez-Pérez, A., Fernández-López, M., & Corcho, O. (2004). *Ontological Engineering (with examples from the areas of Knowledge Management, e-Commerce and the Semantic Web).* Springer.

Gruber, T. R. (1993a). A translation approach to portable ontology specification. *Knowledge Acquisition , 5* (2), 199–220.

Gruber, T. R. (1993b). Toward principles for the design of ontologies used for knowledge sharing. In N. Guarino, & R. Poli (Ed.), *International Workshop on Formal Ontology in Conceptual Analysis and Knowledge Representation.*

Grüninger, M., & Fox, M. (1995). Methodology for the design and evaluation of ontologies. In D. Skuce (Ed.), *IJCAI95 Workshop on Basic Ontological Issues in Knowledge Sharing*, (pp. 6.1–6.10).

Guarino, N. (1998). Formal Ontology in Information Systems. In N. Guarino (Ed.), *1st International Conference on Formal Ontology in Information Systems (FOIS'98)* (pp. 3-15). IOS Press.

Guarino, N., & Giaretta, P. (1995). Ontologies and Knowledge Bases: Towards a Terminological Clarification. In N. Mars (Ed.), *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing (KBKS'95)* (pp. 25–32). IOS Press.

Hemam, M., & Boufaida, Z. (October 2004). An Ontology Development Process for the Semantic Web. *EKAW'04, 14th International Conference on Knowledge Engineering and*

Knowledge Management Whittlebury Hall, (pp. 5-8). Northamptonshire, UK.

Horrocks, I., & Patel-Schneider, P. F. (2003). Reducing OWL Entailment to Description Logic Satisfiability.

Isern, D., Moreno, A., Pedone, G., & Varga, L. Z. (2007). An Intelligent Platform to Provide Home Care Services.

ITABASHI, G., CHIBA, M., TAKAHASHI, K., & KATO, Y. (2005). A Support System for Home Care Service Based on Multi-agent System.

KACTUS. (1996). The KACTUS Booklet version 1.0. *Esprit Project 8145 KACTUS* .

KAYSER, D. (1997). *La représentation des connaissances.* Paris, France : Hermes.

Koutkias, V. G., Chouvarda, I., & Maglaveras, N. (2005). A Multiagent System Enhancing Home-Care Health Services for Chronic Disease Management. *IEEE TRANSACTIONS ON INFORMATION TECHNOLOGY IN BIOMEDICINE , 9* (4), 528-537.

Lassila, O., & McGuinness, D. (2001). *The Role of Frame-Based Representation on the Semantic Web.* Technical Report KSL-01-02, Knowledge Systems Laboratory.Stanford University, Stanford, California.

Mizoguchi, R., Vanwelkenhuysen, J., & Ikeda, M. (1995). Task Ontology for reuse of problem solving knowledge. In N. Mars (Ed.), *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing (KBKS'95)* (pp. 46-57). IOS Press.

Neches, R; Fikes, R E; Finin, T; Gruber, T R; Senator, T; Swartout, W R. (1991). Enabling technology for knowledge sharing. *AI Magazine , 12* (3), 36–56.

Noy, N. F., Fergerson, R. W., & Musen, M. A. (2000). The knowledge model of Protege-2000:Combining interoperability and flexibility. In R. Dieng, & O. Corby (Ed.), *12th International Conference in Knowledge Engineering and Knowledge Management (EKAW'00)* (pp. 17–32). (Lecture Notes in Artificial Intelligence LNAI 1937) Springer-Verlag.

Noy, N., & McGuiness, D. (2001). *Ontology Development 101 – A guide to creating your first ontology*. KSL Technical Report, Stanford University, Stanford, CA, USA.

Psyché, V., Mendes, O., & Bourdeau, J. (2003). Apport de l'ingénierie ontologique aux environnements de formation à distance.

Rector, A., Solomon, W., Nowlan, W., & Rush, T. (1995). A Terminology Server for Medical Language and Medical Information Systems. *Methods of Information in Medicine 34* , 147–157.

ROSSE, C., & MEJINO, J. (2003). A reference ontology for bioinformatics : The foundational model of anatomy. *Journal of Biomedical Informatics* .

Schmidt-Schauß, M., & Smolka, G. (1991). Attributive concept descriptions with complements. *Artificial Intelligence , 48* (1), 1–26.

SOWA, J. (1984). *Conceptual Structures : Information Processing in Mind and Machine.* London :Addison-Wesley.

Staab, S., Schnurr, H. P., Studer, R., & Sure, Y. (2001). Knowledge Processes and Ontologies. *IEEE Intelligent Systems , 16* (1), 26–34.

Studer, R., Benjamins, V. R., & Fensel, D. (1998). Knowledge Engineering: Principles and Methods. *IEEE Transactions on Data and Knowledge Engineering , 25* (1-2), 161-197.

Sure, Y., Erdmann, M., Angele, J., Staab, S., & Studer, R. (2002). OntoEdit: Collaborative Ontology Engineering for the Semantic Web. Dans I. Horrocks, & J. A. Hendler (Éd.), *First International Semantic Web Conference (ISWC'02)* (pp. 221–235). (Lecture Notes in Computer Science LNCS 2342) Springer-Verlag.

Swartout, B., Ramesh, P., Knight, K., & Russ, T. (1997). Toward Distributed Use of Large-Scale Ontologies. In A. Farquhar, M. Gruninger, A. Gómez-Pérez, & M. Uschold (Ed.), *AAAI'97 Spring Symposium on Ontological Engineering*, (pp. 138–148). Stanford University, California.

Uschold, M., & Grüninger, M. (1996). Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review , 11* (2), 93–155.

Uschold, M., & King, M. (1995). Towards a Methodology for Building Ontologies. In D. Skuce (Ed.), *IJCAI'95 Workshop on Basic Ontological Issues in Knowledge Sharing*, (pp. 6.1-6.10). Montreal, Canada.

Van Heijst, G., Schreiber, A. T., & Wielinga, B. J. (1997). Using explicit ontologies in KBS development. *International Journal of Human-Computer Studies , 45*, 183–292.

Zarour, K., & Zarour, N. (2007). *Vers un système d'information coopératif pour la prise en charge des soins à domicile: une architecture basé agent.* Memoire de magister, Université Mentouri de Constantine, Faculté des sciences de l'ingenieur, département informatique, laboratoire LIRE.

Zweigenbaum, P. (1994). Menelas - an Access System for Medical Records Using Natural language. *Computer Methods and Programs in Biomedicine , 45* (1-2), 117-120.

ANNEXE A

LES ÉTAPES DE L'ÉDITION D'UNE ONTOLOGIE SOUS L'ÉDITEUR PROTÉGÉ

A.1 Présentation de l'éditeur PROTÉGÉ

L'outil PROTÉGÉ a été développé par le Stanford Medical Informatics au sein de l'Université de Stanford. Le modèle de représentation de connaissances de PROTÉGÉ est issu du paradigme des frames et contient des classes (pour modéliser les concepts), des slots (pour modéliser les propriétés des concepts) et des facettes (pour définir les valeurs des propriétés et des contraintes sur ces valeurs), ainsi que des instances des classes. PROTÉGÉ introduit la notion de méta-classes, dont les instances sont des classes.

L'architecture de PROTÉGÉ consiste en deux parties : 'le model' et 'la vue'. Le model est le mécanisme de représentation interne pour les ontologies et les composants de la vue fournit une interface utilisateur pour l'affichage et la manipulation du model sous-jacent.
L'interface très complète ainsi que l'architecture logicielle ouverte et extensible permettant l'insertion de plug-ins, ont grandement participé au succès de PROTÉGÉ. Ces plug-ins pouvant apporter de nouvelles fonctionnalités entre autre OWLViz qui permet de gérer des représentations sous forme graphique, PROMPT qui est dédié à la gestion de plusieurs ontologies, en particulier à leur fusion et un plugin dédié à OWL.

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.
Il est également possible de faire fonctionner des raisonneurs, comme RACER (Renamed ABox and Concept Expression Reasoner) pour le langage OWL-DL par exemple, pour vérifier la cohérence et la consistance de la structure ontologique.

Le plugin PROTÉGÉ OWL est une extension de PROTÉGÉ qui supporte le développement des ontologies en OWL. Il permet de :

- Sauvegarder des ontologies en plusieurs formats : RDF et OWL ;

- Editer et visualiser les classes et les propriétés des ontologies OWL ;
- Définir des expressions logiques des classes ;
- Exécuter des raisonneurs.

A.2 Conventions de nomination

Définir des conventions à suivre lorsqu'on nomme les différentes entités de l'ontologie (classes, propriétés et instances) et y adhérer, non seulement rend l'ontologie plus lisible et compréhensible, mais aide également à éviter les quelques erreurs les plus fréquentes de modélisation (Noy & McGuiness, 2001). Plusieurs alternatives existent pour la normalisation mais il n'y a pas de raison particulière pour privilégier l'une ou l'autre de ces alternatives.

Les caractéristiques suivantes d'éditeur d'ontologie affectent le choix de la convention de nomination (Noy & McGuiness, 2001):

- Le système a-t-il le même espace de nomination pour les classes, attributs et instances ? C'est-à-dire, permet–il d'avoir une classe et un attribut ayant le même nom?
- Le système est-il sensible à la casse ? C'est-à-dire, traite-t-il de la même façon les noms selon qu'ils sont entrés en majuscules ou en minuscules ?
- Quels délimiteurs le système autorise-t-il pour les noms ? C'est-à-dire, les noms peuvent-ils contenir des espaces, des virgules, des astérisques, etc. ?

L'éditeur d'ontologie «PROTÉGÉ OWL 3.3.1 » avec lequel l'ontologie de ce projet a été éditée, maintient un espace de nommage unique pour les classes, les propriétés et les instances. Il est sensible à la casse. Il ne permet l'apparition des espaces dans les noms et que les délimiteurs autorisés sont bien « _ » et «-». C'est pour ces considérations que les expressions linguistiques, représentant les différentes entités de l'ontologie, déjà présentées dans l'ontologie conceptuelle et formelle ont été légèrement modifiées, lors de leurs éditions sous PROTÉGÉ.

Comme PROTÉGÉ 3.3.1 est sensible à la casse et n'autorise pas l'utilisation des espaces dans les noms et pour améliorer considérablement la lisibilité de l'ontologie, nous utilisons systématiquement ; ce qu'il est d'usage ; des lettres initiales majuscules pour les noms des classes et des minuscules pour les noms des propriétés. Ainsi, lorsque le nom d'un concept contient plus d'un mot, nous avons collé les mots les uns aux autres, sans aucun délimiteur, en employant une lettre initiale capitale pour chacun des mots.

A.3 Les étapes de l'édition d'une ontologie sous PROTÉGÉ-OWL

Avec l'ontologie OntoPAD, on va démontrer comment nous avons utilisé PROTÉGÉ-OWL pour éditer une ontologie OWL.

A.3.1 Création d'un nouveau projet

Lors du premier démarrage de PROTÉGÉ sous Windows, une boite de dialogue '*Welcome to PROTEGE'* s'ouvre. Afin de créer un nouveau projet OWL, nous devons cliquer sur le bouton de création d'un nouveau projet « New Project...». Cette action génère une autre boite de dialogue 'Create New Project' contenant différent type du projet à choisir. Pour ce qui est de notre cas, il s'agit d'un fichier OWL ' OWL Files (.owl or .rdf) '. Une fois que ce choix est validé, il faut préciser le langage avec lequel sera éditée l'ontologie, il s'agit d'OWL DL. Par la suite, l'interface de l'outil s'affiche (voir Figure A.1) permettant d'éditer, de visualiser et d'enregistrer des ontologies en OWL DL.

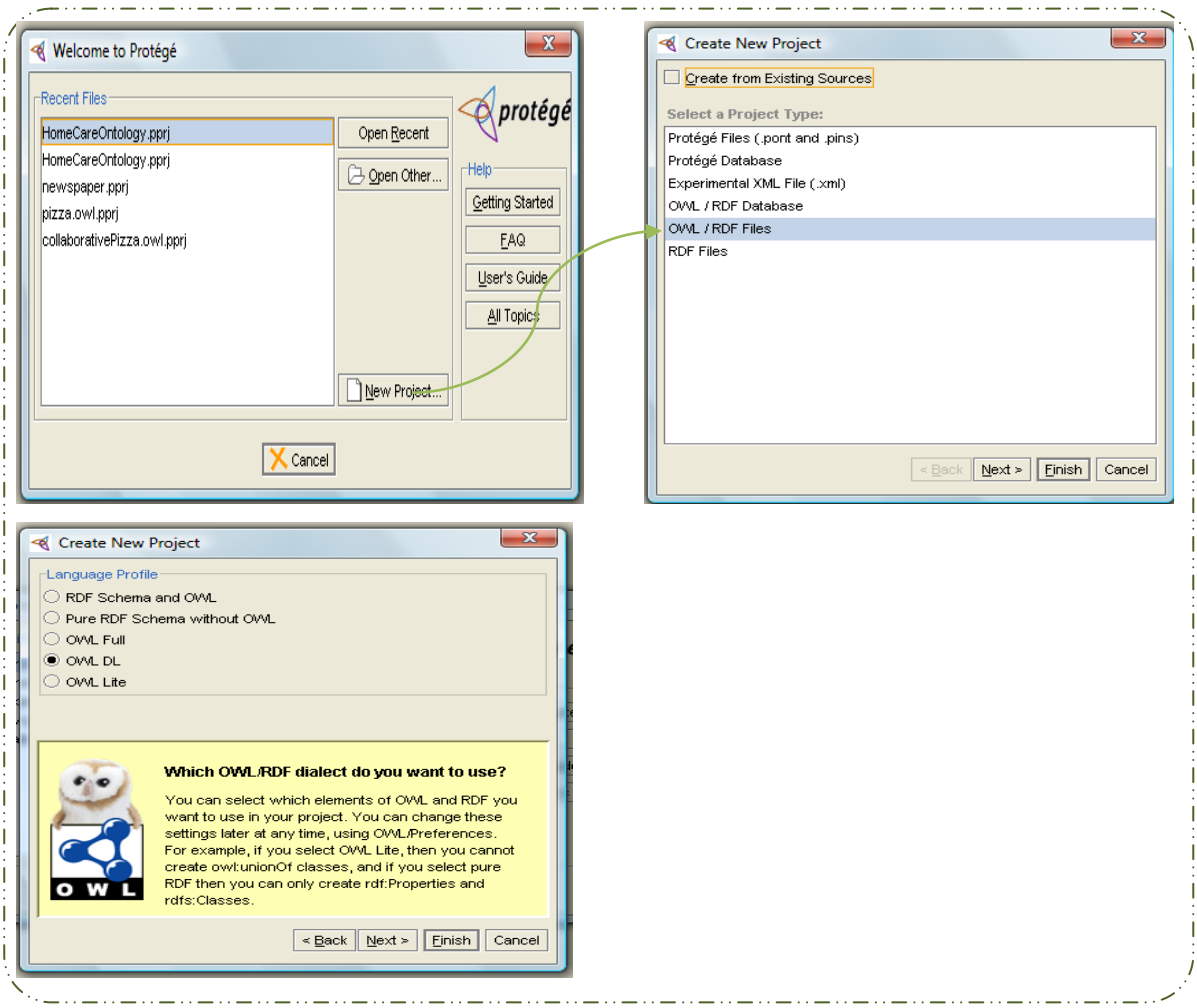

Figure A.1 Création d'un nouveau projet.

Comme illustré dans la figure A.2, l'interface utilisateur de PROTÉGÉ-OWL plugin fournit un ensemble d'onglets. Les trois onglets les plus importants que nous allons utilisés par la suite pour l'édition des composants de l'ontologie sont :

- L'onglet *OWL Classes* : affiche l'hiérarchie de classes de l'ontologie. Il permet aux développeurs de créer et d'éditer les classes et affiche le résultat de la classification.
- L'onglet P*roperties* : est utilisé pour créer et éditer les propriétés de l'ontologie.
- L'onglet I*ndividuals* : est utilisé pour créer et éditer les instances.

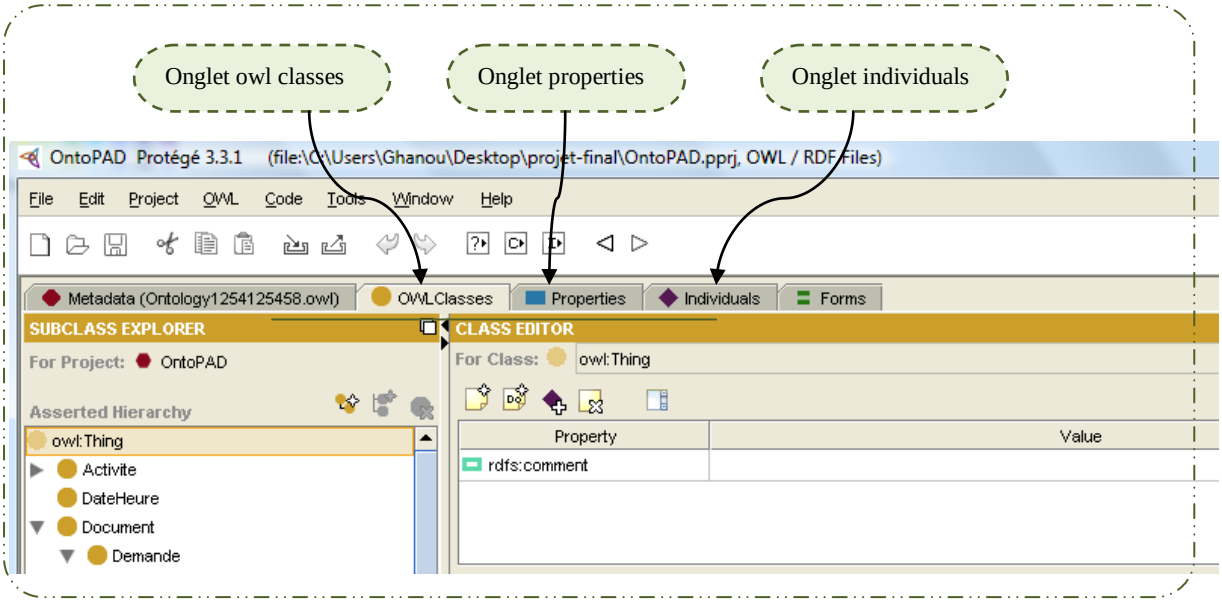

Figure A.2 Interface de PROTÉGÉ OWL.

A.3.2 Création des classes et la hiérarchie des classes

Nous avons commencé par éditer toutes les classes de l'ontologie spécifiées dans l'étape de conceptualisation en utilisant l'onglet *OWL Classes*.

Une classe universelle (owl : thing) est utilisée comme racine pour cette hiérarchie, et la création des sous-classes se fait par le choix de la classe mère, suivi par un simple clique sur le bouton de création des sous-classes de la classe sélectionnée (voir figure A.3).

Dans cet onglet, les classes disposant des sous-classes apparaissant sur l'onglet sont précédées par le signe (▼), alors que les classes disposant des sous-classes qui n'apparaissent pas sur l'onglet sont précédées par le signe (▶). Ces sous-classes peuvent être montrées par un simple clique sur ce signe. Les classes qui n'ont pas ce signe ne disposent pas de sous classes.

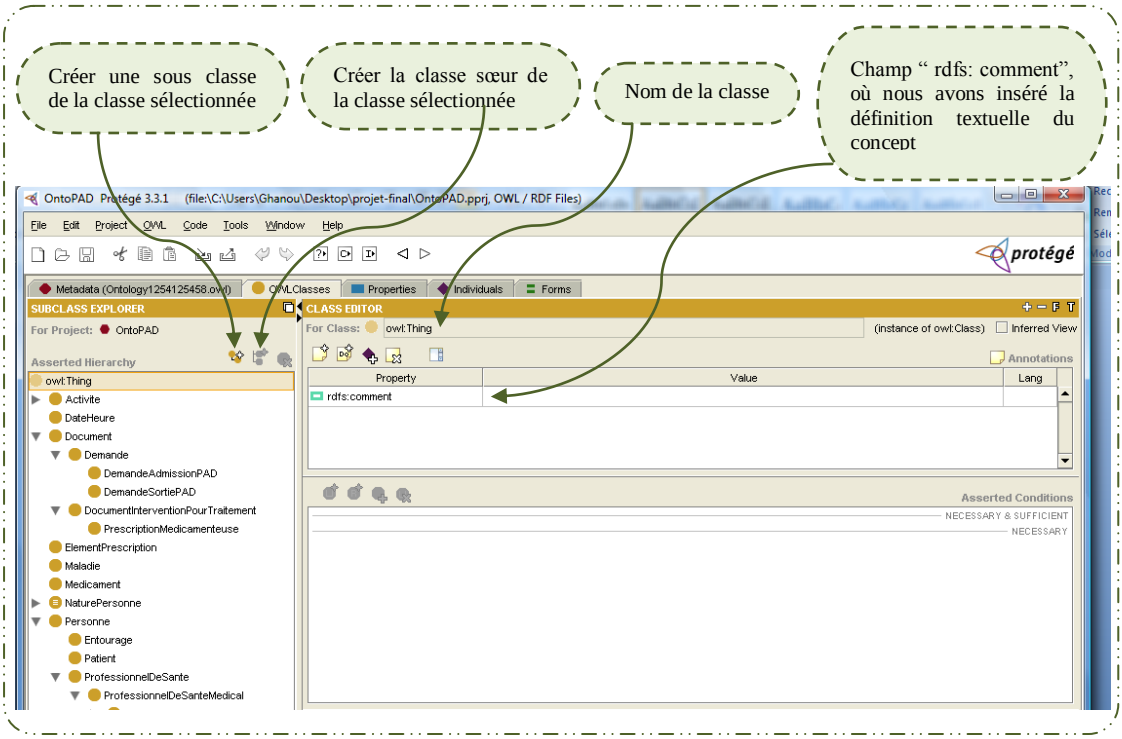

Figure A.3 Édition des concepts.

A.3.3 Création des propriétés

Après avoir construit les classes, nous créons maintenant les propriétés (sont appelées rôles en logique de description) pour chacune d'elles en utilisant l'onglet *Properties*. Il existe deux types de propriétés, les attributs *'Datatype properties'* et les relations *'Object properties'*. Les propriétés d'une classe sont les propriétés héritées de sa superclasse, plus ses propres propriétés privées. On peut enrichir la sémantique des propriétés lors de la création de ces derniers, en leur spécifiant les caractéristiques suivantes :

- **Functional property :** si une propriété est fonctional pour un individu donné, il peut exister au plus un individu qui est relié à l'autre individu par cette relation.
- **Inverse property :** si une propriété P a sa propriété inverse, et P relie un individu A à un individu B, alors la propriété inverse relie l'individu B à l'individu A.
- **Inverse functional property :** si une propriété est inverse functional, ca veut dire que la propriété inverse de cette propriété est functionel.
- **Transitive property :** si une propriété P est transitive et P relie l'individu A à l'individu B et l'individu B à l'individu C, alors on peut déduire que A est relié à C par P.
- **Symetric property :** si une propriété P est symetrique, et P relie l'individu A à l'individu B alors, B est relié à A par P

La figure A.4 montre comment on peut créer ces propriétés.

A.3.3.1 Création des attributs (dataTypeProperty)

Un attribut permet de relier des individus à des valeurs de données. Une valeur étant un élément d'un des types de données prédéfinis de XML Schema (float, string, etc.).

Pour créer un attribut d'une classe, il faut cliquer sur le bouton de création d'une nouvelle propriété *'dataTypeProperty'*  . Cette action génère une fenêtre comme la montre la figure A.5; où on doit spécifier le nom de l'attribut, le domaine, le type XML associé, les valeurs permises s'ils ont connues ainsi que le type de valeurs de l'attribut (la seule option qui est affiché pour un attribut est functional).

A.3.3.2 Création des relations (ObjectProperty)

Une relation permet de relier des individus à d'autres individus. Pour créer une relation entre deux classes, il faut cliquer sur le bouton de création d'une propriété '*object Property'*  . Cette action permet de générer une nouvelle fenêtre (voir la figure A.6) différente de celle des attributs où on spécifie le nom de la relation, le domaine, le co-domaine, la relation inverse si elle existe ainsi que les caractéristiques de la relation (functional, symétrique, transitive,…). Iles peuvent être hiérarchisées en utilisant l'onglet (Properties).

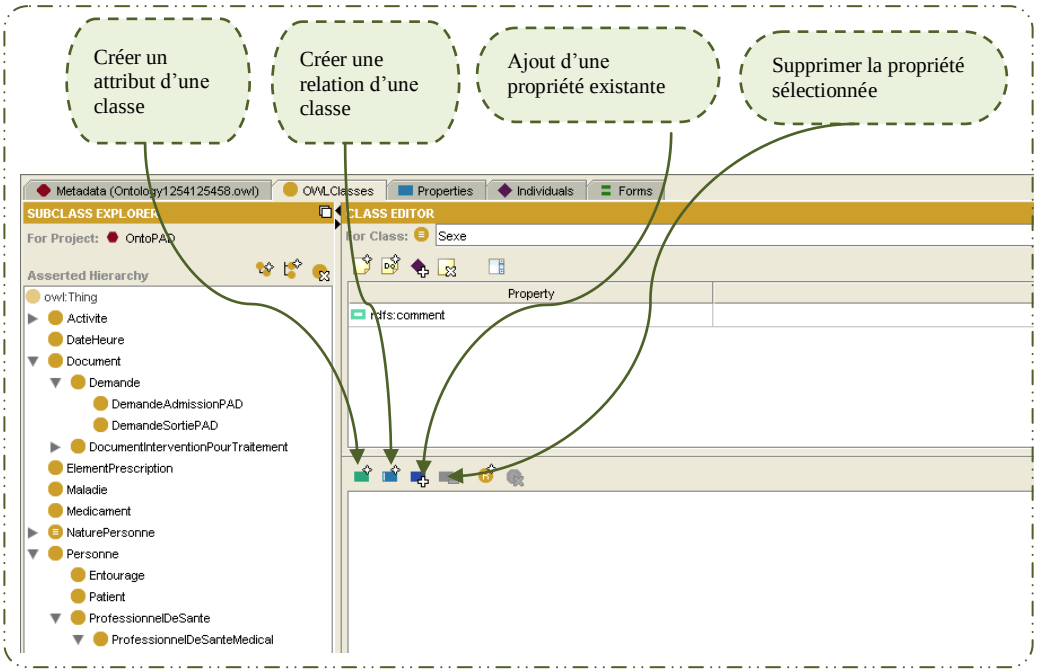

Figure A.4 Création de propriétés pour une classe**.**

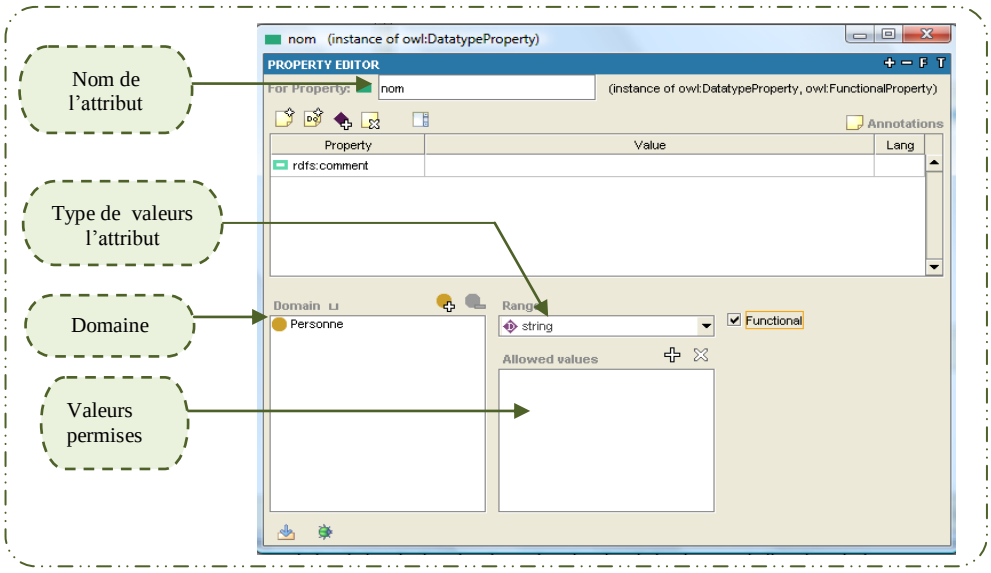

Figure A.5 Création d'un attribut.

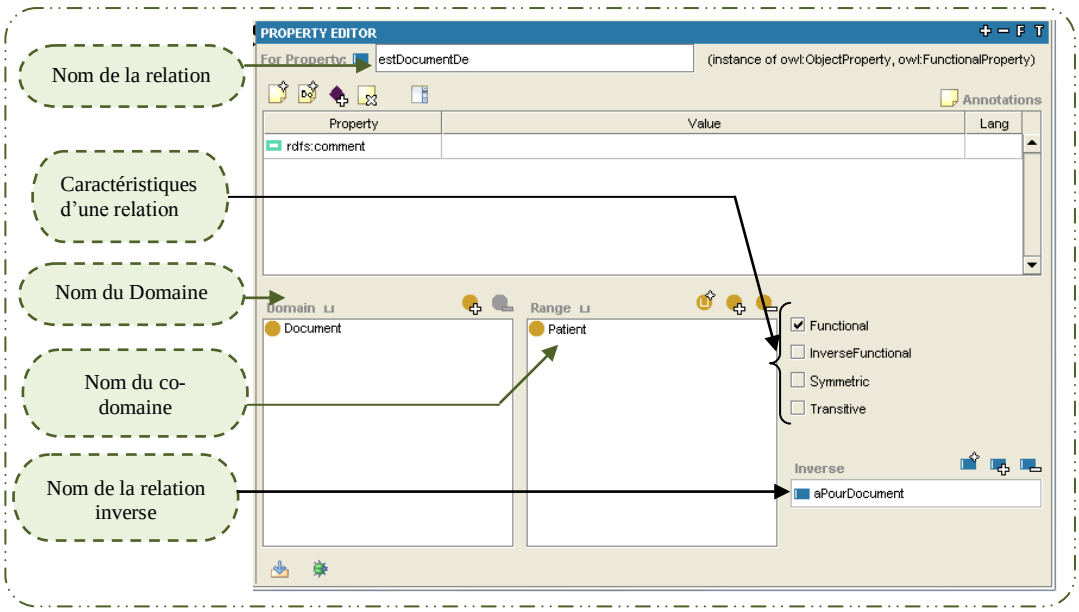

Figure A.6 Création d'une relation entre deux classes.

A.3.4 Restrictions sur les propriétés

Les restrictions sont utilisées pour décrire ou définir une classe c.-à-d. restreindre les individus appartenant à une classe. En OWL DL, il existe deux types de restrictions de propriétés :

Les contraintes de valeurs (values contraints) : $\forall$, $\exists$, $\ni$.

Les contraintes de cardinalités (cardinality contraints) : >, <, =.

Pour créer une restriction sur une propriété d'une classe en utilisant PROTÉGÉ, il suffit de sélectionner la classe qu'on veut décrit, puis cliquez au dessus du bouton  , la fenêtre montré dans la figure A.7 s'affichera.

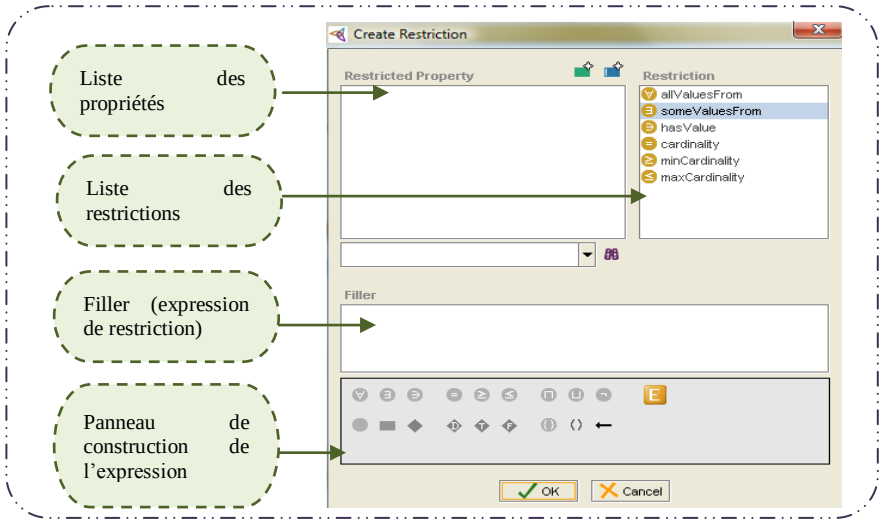

Figure A.7 Création d'une restriction**.**

Une fois la fenêtre s'affiche, vous sélectionnez la propriété sur laquelle on va faire la restriction à partir de la liste de propriétés, puis choisissez le type de la restriction (*Cardinality*, *minCardinality*, *maxCardinality*, *hasValue*, *allValuesFrom, someValuesFrome*) de la liste de restriction, et enfin spécifier le filler pour la restriction dans la zone 'filler' ou bien utiliser le panneau de construction de l'expression.

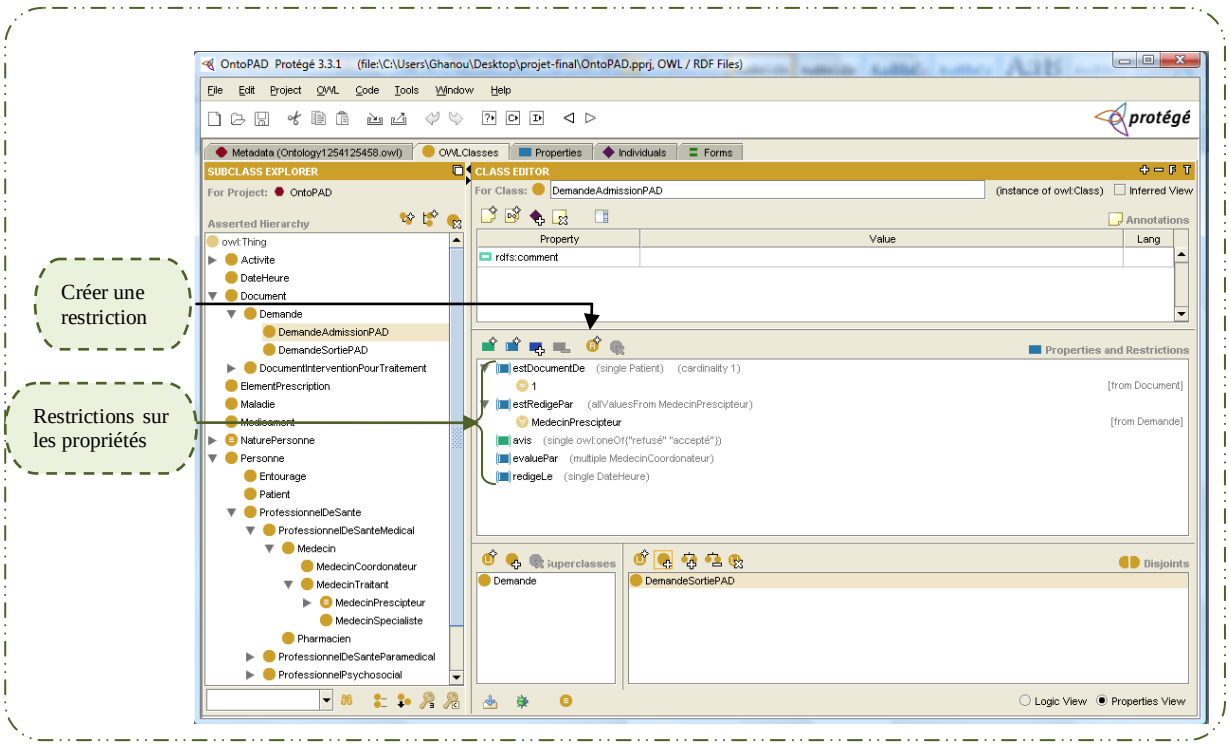

Figure A.8 Les restrictions créées sur les propriétés d'une classe spécifiée.

A.3.5 Création des instances

Une dernière tâche était la saisie des instances des classes de l'hiérarchie dans les formulaires que nous a produit Protégé sous l'onglet *Individuals*. Pour chaque instance d'une classe sélectionnée (la classe à instancier), on doit spécifier le nom de l'instance ainsi que les valeurs des propriétés (les valeurs des attributs et/ou les noms des instances avec lesquelles cette instance est reliée par une relation) comme illustré dans la figure A.9.

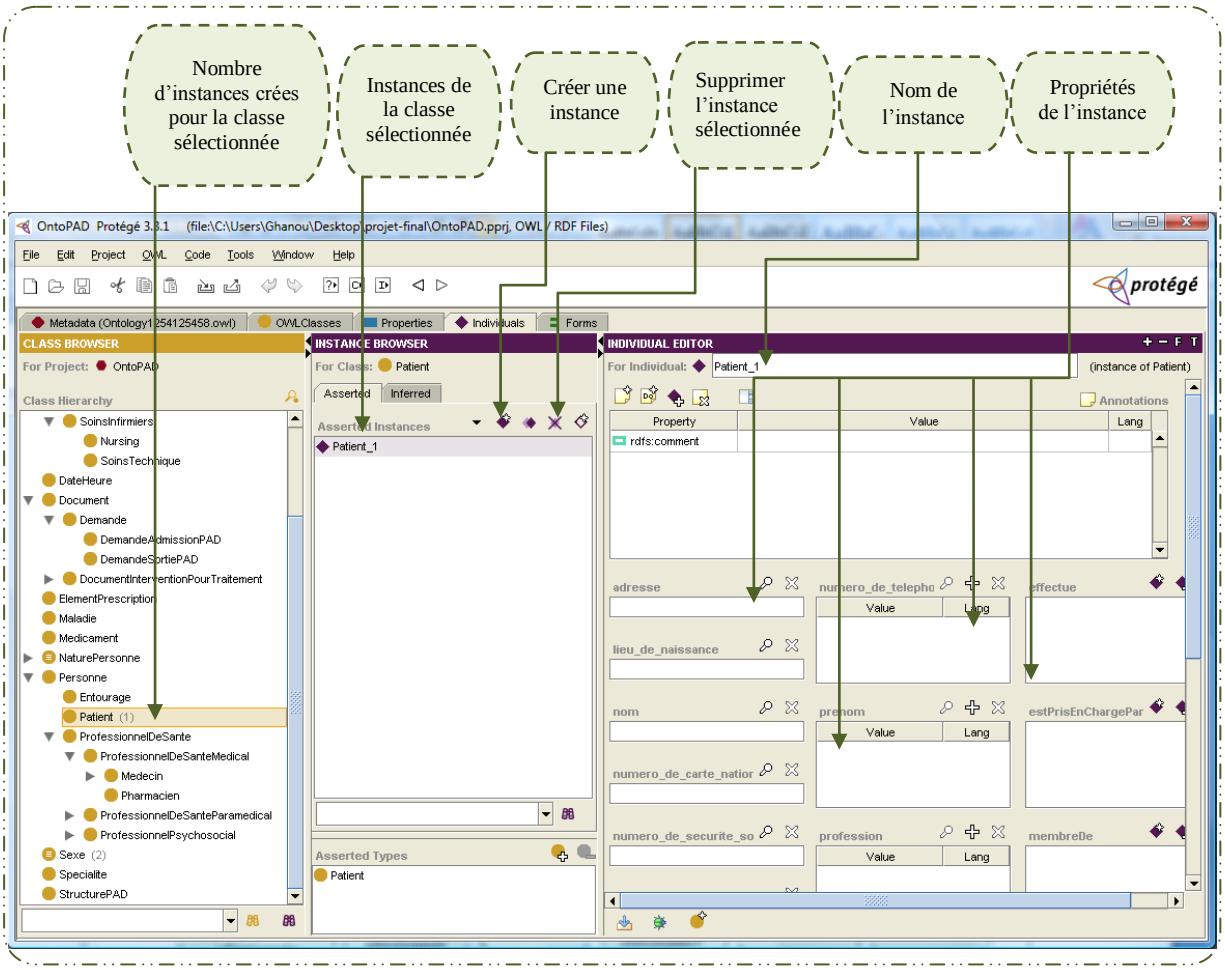

Figure A.9 Création des instances.

ANNEXE B

CONFIGURATION DE L'OUTIL RACER AVEC PROTÉGÉ

RACER (Renamed Abox And Concept Expression Reasoner) est un fichier exécutable disponible sous la forme d'un serveur pour Linux et Windows. Il peut être lancé directement par un invité de commande ou par un double click sur l'icône du programme.

Le système RACER est accessible par les protocoles standards TCP ou HTTP. Mais avant d'y accéder, il faut d'abord configurer la connexion au serveur qui l'héberge.

Pour ce qui est de notre cas, nous allons tester l'ontologie localement. Pour ce faire, il faut connecter RACER à PROTÉGÉ en suivrons les étapes suivantes :

- Lancer l'outil protégé ;
- Activer le menu *OWL* de PROTÉGÉ ;
- Sélectionner l'option *Préférences ;*
- Dans la fenêtre de dialogue qui s'affiche (*OWL préférences*, voir la figure B.1), s'assurer que l'url du raisonneur est réglé sur *'http://localhost:8080'* ou *'http://127.0.0.1:8080'* qui est habituellement la valeur par défaut ;
- Après réalisation des changements nécessaire, valider en pressant sur le bouton 'Close' ;

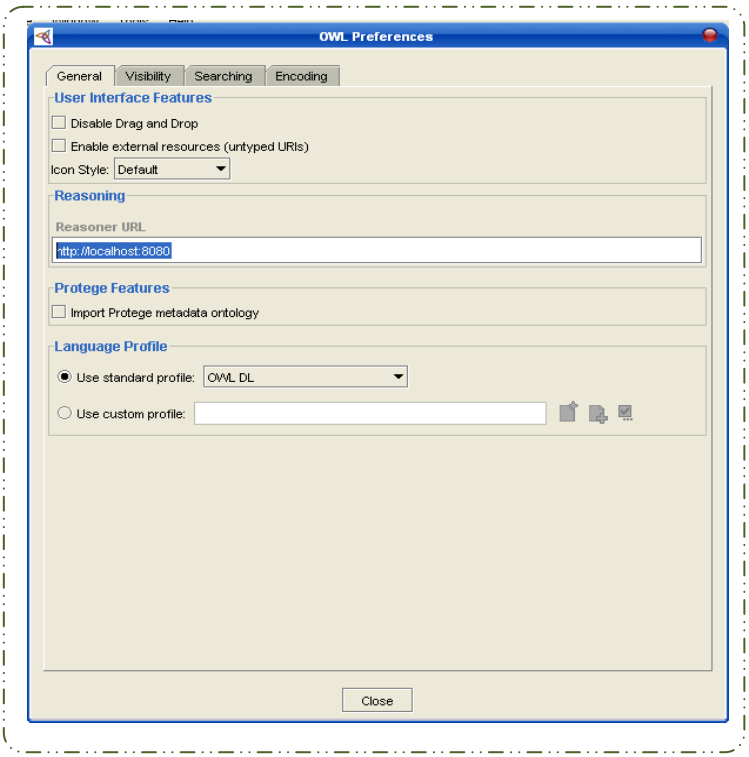

Figure B.1 La fenêtre de Préférences OWL.

- Télécharger et exécuter RACER localement sous windows, les services HTTP et TCP vont être activés sur le port 8080 de la machine local (localhost), comme illustré dans la figure B.2.

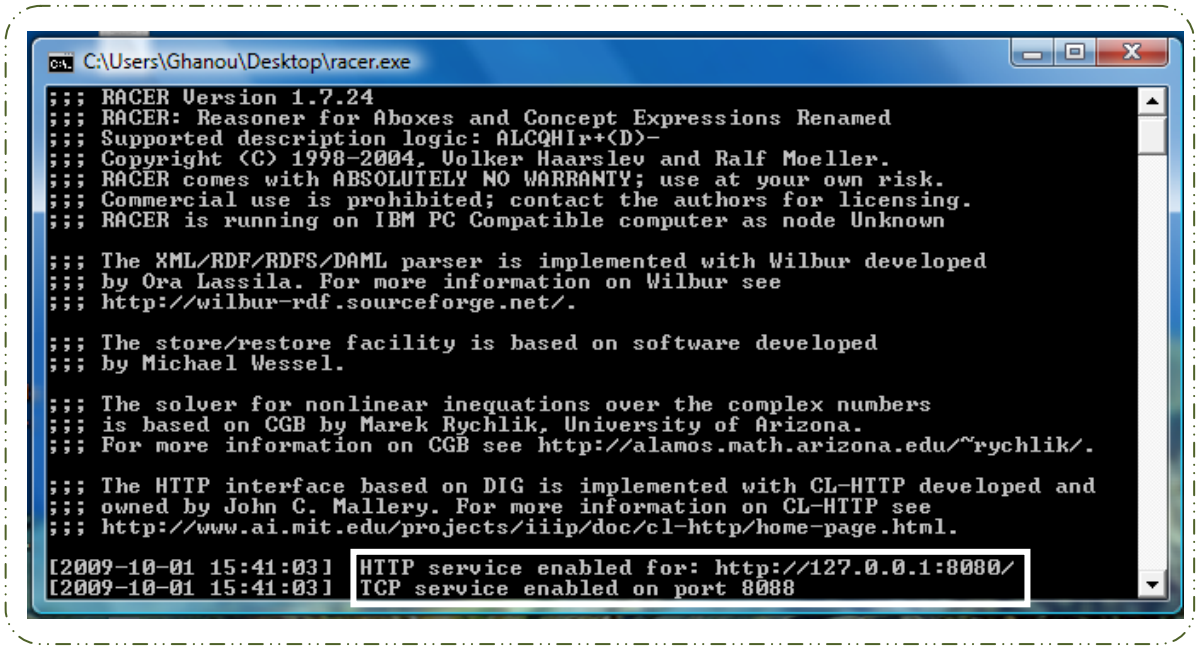

Figure B.2 L'exécutable de RACER sous Windows.

ANNEXE C

LE CODE OWL CORRESPONDANT A L'ONTOLOGIE OntoPAD

```
<?xml version="1.0"?>
<rdf:RDF
    xmlns="http://www.owl-ontologies.com/Ontology1254125458.owl#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:p1="http://www.owl-ontologies.com/assert.owl#"
  xml:base="http://www.owl-ontologies.com/Ontology1254125458.owl">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:ID="ProfessionnelDeSanteParamedical">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >exerce des professions de la santé qui ne sont pas exercées par un médecin et  un
pharmacien</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="ProfessionnelDeSante"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="ProfessionnelDeSoins">
    <rdfs:subClassOf rdf:resource="#ProfessionnelDeSanteParamedical"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Exerce des activités de soins infirmiers.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:about="#ProfessionnelDeSante">
    <owl:disjointWith>
      <owl:Class rdf:ID="Patient"/>
    </owl:disjointWith>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Toute personne travaillant dans le cadre du système de santé. Il peut être un professionnel de
santé médical ou un professionnel de santé paramédical ou un professionnel de santé
psychosocial.</rdfs:comment>
    <rdfs:subClassOf>
      <owl:Class rdf:ID="Personne"/>
    </rdfs:subClassOf>
    <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >ActeurDeSante</rdfs:label>
  </owl:Class>
  <owl:Class rdf:ID="Activite">
    <owl:disjointWith>
      <owl:Class rdf:ID="Document"/>
    </owl:disjointWith>
    <owl:disjointWith>
```

```
    <owl:Class rdf:ID="Maladie"/>
  </owl:disjointWith>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="appliqueSur"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="effectueLe"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <owl:disjointWith>
    <owl:Class rdf:about="#Personne"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >effectuée par une personne sur un patient.</rdfs:comment>
  <owl:disjointWith>
    <owl:Class rdf:ID="StructurePAD"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:ID="DateHeure"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:ID="Specialite"/>
  </owl:disjointWith>
</owl:Class>
<owl:Class rdf:ID="ProfessionnelDeReeducationEtReadaptation">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Exercent des activités de rééducation et réadaptation.</rdfs:comment>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:ID="effectue"/>
      </owl:onProperty>
      <owl:someValuesFrom>
        <owl:Class rdf:ID="ActiviteDeReeducationEtReadaptation"/>
      </owl:someValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
```

```
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:InverseFunctionalProperty rdf:about="#effectue"/>
    </owl:onProperty>
    <owl:allValuesFrom>
     <owl:Class rdf:about="#ActiviteDeReeducationEtReadaptation"/>
    </owl:allValuesFrom>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#ProfessionnelDeSanteParamedical"/>
</owl:Class>
<owl:Class rdf:ID="SoinsTechnique">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:FunctionalProperty rdf:ID="estEffectuePar"/>
    </owl:onProperty>
    <owl:allValuesFrom>
     <owl:Class rdf:ID="Infirmiere"/>
    </owl:allValuesFrom>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Class rdf:ID="SoinsInfirmiers"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="DocumentDePriseEnChargeDuPatient">
  <rdfs:subClassOf>
   <owl:Class rdf:about="#Document"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="MedecinTraitant">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="Medecin"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="Pharmacien">
  <rdfs:subClassOf>
   <owl:Class rdf:ID="ProfessionnelDeSanteMedical"/>
  </rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:about="#Personne">
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Humain</rdfs:label>
  <owl:disjointWith>
   <owl:Class rdf:about="#Specialite"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Activite"/>
```

```
  <owl:disjointWith>
    <owl:Class rdf:about="#StructurePAD"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Maladie"/>
  </owl:disjointWith>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Est un humain qui intervient dans la prise en charge à domicile d'un patient</rdfs:comment>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#DateHeure"/>
  </owl:disjointWith>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="aNature"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="neLe"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <owl:disjointWith>
    <owl:Class rdf:about="#Document"/>
  </owl:disjointWith>
</owl:Class>
<owl:Class rdf:about="#ActiviteDeReeducationEtReadaptation">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#ProfessionnelDeReeducationEtReadaptation"/>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:about="#estEffectuePar"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Activite"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:someValuesFrom rdf:resource="#ProfessionnelDeReeducationEtReadaptation"/>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:about="#estEffectuePar"/>
```

```
      </owl:onProperty>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >sont des activités effectuées par les professionnels de rééducation et
réadaptation.</rdfs:comment>
   <owl:disjointWith>
    <owl:Class rdf:ID="Examen"/>
   </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:ID="Medicament"/>
  <owl:Class rdf:ID="DemandeSortiePAD">
   <owl:disjointWith>
    <owl:Class rdf:ID="DemandeAdmissionPAD"/>
   </owl:disjointWith>
   <rdfs:subClassOf>
    <owl:Class rdf:ID="Demande"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Patient">
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:someValuesFrom rdf:resource="#Activite"/>
     <owl:onProperty>
      <owl:InverseFunctionalProperty rdf:ID="subit"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Malade</rdfs:label>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:cardinality>
     <owl:onProperty>
      <owl:ObjectProperty rdf:ID="membreDe"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf rdf:resource="#Personne"/>
   <owl:disjointWith>
    <owl:Class rdf:ID="Entourage"/>
   </owl:disjointWith>
   <owl:disjointWith rdf:resource="#ProfessionnelDeSante"/>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:InverseFunctionalProperty rdf:ID="aPourDocument"/>
     </owl:onProperty>
```

```
      <owl:someValuesFrom>
       <owl:Class rdf:about="#Document"/>
      </owl:someValuesFrom>
     </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
     <owl:Restriction>
      <owl:onProperty>
       <owl:ObjectProperty rdf:ID="estPrisEnChargePar"/>
      </owl:onProperty>
      <owl:someValuesFrom rdf:resource="#ProfessionnelDeSante"/>
     </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:subClassOf>
     <owl:Restriction>
      <owl:someValuesFrom>
       <owl:Class rdf:about="#Maladie"/>
      </owl:someValuesFrom>
      <owl:onProperty>
       <owl:ObjectProperty rdf:ID="aPourMaladie"/>
      </owl:onProperty>
     </owl:Restriction>
    </rdfs:subClassOf>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >est une personne souffrant d'une maladie pour laquelle elle est prise en charge à son domicile.
Elle subit plusieurs activités et il y a plusieurs documents qui la concernent.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="ExamenImagerie">
    <owl:disjointWith>
     <owl:Class rdf:ID="ExamenBiologique"/>
    </owl:disjointWith>
    <rdfs:subClassOf>
     <owl:Class rdf:ID="ExamenParaclinique"/>
    </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="MedecinCoordonateur">
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Est un médecin intégré à l'équipe de la structure de la PAD, non prescripteur. Son rôle est
d'assurer le bon fonctionnement médical de la structure, la collaboration au sein de l'équipe, de
veiller à la bonne transmission des informations médicales à l'accomplissement des soins. Il participe
à l'évaluation des soins, il assure la liaison et la coordination entre le médecin traitant, le médecin
hospitalier et les autres intervenants. Il évalue toute demande d'une admission ou
sortie.</rdfs:comment>
    <rdfs:subClassOf>
     <owl:Restriction>
      <owl:someValuesFrom>
       <owl:Class rdf:about="#Demande"/>
      </owl:someValuesFrom>
      <owl:onProperty>
```

```
        <owl:ObjectProperty rdf:ID="evalue"/>
       </owl:onProperty>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty>
         <owl:InverseFunctionalProperty rdf:ID="constiue"/>
       </owl:onProperty>
       <owl:someValuesFrom>
         <owl:Class rdf:about="#StructurePAD"/>
       </owl:someValuesFrom>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
     <owl:Class rdf:about="#Medecin"/>
   </rdfs:subClassOf>
 </owl:Class>
 <owl:Class rdf:about="#DemandeAdmissionPAD">
   <rdfs:subClassOf>
     <owl:Class rdf:about="#Demande"/>
   </rdfs:subClassOf>
   <owl:disjointWith rdf:resource="#DemandeSortiePAD"/>
 </owl:Class>
 <owl:Class rdf:about="#Document">
   <owl:disjointWith>
     <owl:Class rdf:about="#Maladie"/>
   </owl:disjointWith>
   <owl:disjointWith>
     <owl:Class rdf:about="#StructurePAD"/>
   </owl:disjointWith>
   <rdfs:subClassOf rdf:resource="http://www.w3.org/2002/07/owl#Thing"/>
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty>
         <owl:FunctionalProperty rdf:ID="estDocumentDe"/>
       </owl:onProperty>
       <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
       >1</owl:cardinality>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Est défini comme un ensemble de papiers qui peuvent être transférés au niveau de la structure de
PAD.</rdfs:comment>
   <owl:disjointWith>
     <owl:Class rdf:about="#DateHeure"/>
   </owl:disjointWith>
   <owl:disjointWith rdf:resource="#Personne"/>
   <rdfs:subClassOf>
```

```
    <owl:Restriction>
     <owl:onProperty>
      <owl:FunctionalProperty rdf:ID="estRedigePar"/>
     </owl:onProperty>
     <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >1</owl:cardinality>
    </owl:Restriction>
   </rdfs:subClassOf>
   <owl:disjointWith rdf:resource="#Activite"/>
   <owl:disjointWith>
    <owl:Class rdf:about="#Specialite"/>
   </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:ID="ExamenClinique">
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Comprend inspection, palpation, percussion et auscultation des différentes parties du corps et de
certains organes. Il  est effectué par le médecin traitant pour la recherche des signes et des
symptômes.</rdfs:comment>
   <owl:disjointWith>
    <owl:Class rdf:about="#ExamenParaclinique"/>
   </owl:disjointWith>
   <rdfs:subClassOf>
    <owl:Class rdf:about="#Examen"/>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:allValuesFrom rdf:resource="#MedecinTraitant"/>
     <owl:onProperty>
      <owl:FunctionalProperty rdf:about="#estEffectuePar"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:ID="Psychologue">
   <rdfs:subClassOf>
    <owl:Class rdf:ID="ProfessionnelPsychosocial"/>
   </rdfs:subClassOf>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Il accompagne et soutient le patient et son entourage selon les besoins.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="Femme">
   <rdfs:subClassOf>
    <owl:Class rdf:ID="NaturePersonne"/>
   </rdfs:subClassOf>
   <owl:disjointWith>
    <owl:Class rdf:ID="Homme"/>
   </owl:disjointWith>
   <owl:equivalentClass>
    <owl:Restriction>
```

```
      <owl:hasValue>
        <Sexe rdf:ID="Feminin"/>
      </owl:hasValue>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:ID="aSexe"/>
      </owl:onProperty>
    </owl:Restriction>
  </owl:equivalentClass>
 </owl:Class>
 <owl:Class rdf:about="#Infirmiere">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:about="#effectue"/>
      </owl:onProperty>
      <owl:someValuesFrom rdf:resource="#SoinsTechnique"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:about="#effectue"/>
      </owl:onProperty>
      <owl:allValuesFrom>
        <owl:Class rdf:about="#SoinsInfirmiers"/>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#ProfessionnelDeSoins"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >est un professionnel de santé paramédical dont le métier est de délivrer des soins infirmiers au
patient sur prescription médicale ainsi que des soins d'hygiène et de prévention.</rdfs:comment>
 </owl:Class>
 <owl:Class rdf:ID="MedecinSpecialiste">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="aPourSpecialite"/>
      </owl:onProperty>
      <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#MedecinTraitant"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Est un médecin traitant qui a une spécialité.</rdfs:comment>
 </owl:Class>
 <owl:Class rdf:about="#Specialite">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

```
   >représente les différentes spécialités en médecine.</rdfs:comment>
   <owl:disjointWith rdf:resource="#Activite"/>
   <owl:disjointWith>
     <owl:Class rdf:about="#DateHeure"/>
   </owl:disjointWith>
   <owl:disjointWith rdf:resource="#Document"/>
   <owl:disjointWith rdf:resource="#Personne"/>
   <owl:disjointWith>
     <owl:Class rdf:about="#StructurePAD"/>
   </owl:disjointWith>
   <owl:disjointWith>
     <owl:Class rdf:about="#Maladie"/>
   </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:ID="ActiviteSociale">
   <rdfs:subClassOf>
     <owl:Restriction>
       <owl:onProperty>
         <owl:FunctionalProperty rdf:about="#estEffectuePar"/>
       </owl:onProperty>
       <owl:allValuesFrom>
         <owl:Class rdf:ID="AssistanteSociale"/>
       </owl:allValuesFrom>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf rdf:resource="#Activite"/>
  </owl:Class>
  <owl:Class rdf:ID="MedecinHospitalier">
   <rdfs:subClassOf>
     <owl:Class rdf:ID="MedecinPrescipteur"/>
   </rdfs:subClassOf>
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >MedecinDeHopital</rdfs:label>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Est un médecin qui travaille dans un hôpital. s'engage à suivre le patient au niveau hospitalier et
le ré-hospitaliser.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:about="#StructurePAD">
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >est une équipe composée de plusieurs personnes.</rdfs:comment>
   <owl:disjointWith rdf:resource="#Activite"/>
   <owl:disjointWith>
     <owl:Class rdf:about="#DateHeure"/>
   </owl:disjointWith>
   <owl:disjointWith rdf:resource="#Document"/>
   <owl:disjointWith rdf:resource="#Personne"/>
   <owl:disjointWith rdf:resource="#Specialite"/>
   <owl:disjointWith>
     <owl:Class rdf:about="#Maladie"/>
```

```
   </owl:disjointWith>
  </owl:Class>
  <owl:Class rdf:about="#AssistanteSociale">
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Coopère étroitement avec la structure PAD. Au travers d'une enquête, elle essaie de déterminer
avec le concours de la famille les problèmes sociaux et financiers ainsi que les besoins du malade.
Cette évaluation sociale est préalable à l'admission. Elle aide le patient et sa famille à surmonter leurs
difficultés et à développer leurs capacités propres en vue d'améliorer leurs conditions de vie sur les
plans social, économique ou culturel.</rdfs:comment>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:onProperty>
      <owl:InverseFunctionalProperty rdf:ID="redige"/>
     </owl:onProperty>
     <owl:allValuesFrom>
      <owl:Class rdf:ID="RapportSocial"/>
     </owl:allValuesFrom>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:someValuesFrom>
      <owl:Class rdf:about="#RapportSocial"/>
     </owl:someValuesFrom>
     <owl:onProperty>
      <owl:InverseFunctionalProperty rdf:about="#redige"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:allValuesFrom rdf:resource="#ActiviteSociale"/>
     <owl:onProperty>
      <owl:InverseFunctionalProperty rdf:about="#effectue"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Class rdf:about="#ProfessionnelPsychosocial"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Maladie">
   <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Pathologie</rdfs:label>
   <owl:disjointWith rdf:resource="#Activite"/>
   <owl:disjointWith>
    <owl:Class rdf:about="#DateHeure"/>
   </owl:disjointWith>
   <owl:disjointWith rdf:resource="#Document"/>
```

```
  <owl:disjointWith rdf:resource="#Personne"/>
  <owl:disjointWith rdf:resource="#Specialite"/>
  <owl:disjointWith rdf:resource="#StructurePAD"/>
</owl:Class>
<owl:Class rdf:about="#DateHeure">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Représente la date et l'heure exacte.</rdfs:comment>
  <owl:disjointWith rdf:resource="#Activite"/>
  <owl:disjointWith rdf:resource="#Document"/>
  <owl:disjointWith rdf:resource="#Personne"/>
  <owl:disjointWith rdf:resource="#Specialite"/>
  <owl:disjointWith rdf:resource="#StructurePAD"/>
  <owl:disjointWith rdf:resource="#Maladie"/>
</owl:Class>
<owl:Class rdf:about="#Homme">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#NaturePersonne"/>
        <owl:Restriction>
          <owl:hasValue>
            <Sexe rdf:ID="Masculin"/>
          </owl:hasValue>
          <owl:onProperty>
            <owl:InverseFunctionalProperty rdf:about="#aSexe"/>
          </owl:onProperty>
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
  <owl:disjointWith rdf:resource="#Femme"/>
</owl:Class>
<owl:Class rdf:about="#Entourage">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:ID="estEntourageDe"/>
      </owl:onProperty>
      <owl:someValuesFrom rdf:resource="#Patient"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Personne"/>
  <owl:disjointWith rdf:resource="#Patient"/>
</owl:Class>
<owl:Class rdf:about="#SoinsInfirmiers">
  <rdfs:subClassOf rdf:resource="#Activite"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >sont des activités effectuées par des infirmiers</rdfs:comment>
  <owl:disjointWith>
```

```
    <owl:Class rdf:about="#Examen"/>
  </owl:disjointWith>
</owl:Class>
<owl:Class rdf:ID="PrescriptionMedicamenteuse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:InverseFunctionalProperty rdf:ID="contient"/>
      </owl:onProperty>
      <owl:minCardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
      >1</owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:allValuesFrom rdf:resource="#MedecinTraitant"/>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:about="#estRedigePar"/>
      </owl:onProperty>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#DocumentDePriseEnChargeDuPatient"/>
</owl:Class>
<owl:Class rdf:about="#RapportSocial">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty>
        <owl:FunctionalProperty rdf:about="#estRedigePar"/>
      </owl:onProperty>
      <owl:allValuesFrom rdf:resource="#AssistanteSociale"/>
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#DocumentDePriseEnChargeDuPatient"/>
</owl:Class>
<owl:Class rdf:ID="Sexe">
  <owl:equivalentClass>
    <owl:Class>
      <owl:oneOf rdf:parseType="Collection">
        <Sexe rdf:about="#Feminin"/>
        <Sexe rdf:about="#Masculin"/>
      </owl:oneOf>
    </owl:Class>
  </owl:equivalentClass>
</owl:Class>
<owl:Class rdf:about="#ProfessionnelPsychosocial">
  <rdfs:subClassOf rdf:resource="#ProfessionnelDeSante"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Ils sont aux nombres de deux : assistante sociale et psychologue.</rdfs:comment>
</owl:Class>
```

```
 <owl:Class rdf:about="#MedecinPrescipteur">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >est le médecin qui prescrit l'admission d'une prise en charge à domicile d'un patient ainsi que la sortie. Il peut être médecin hospitalier ou médecin de famille.</rdfs:comment>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:InverseFunctionalProperty rdf:about="#redige"/>
    </owl:onProperty>
    <owl:someValuesFrom>
     <owl:Class rdf:about="#Demande"/>
    </owl:someValuesFrom>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:InverseFunctionalProperty rdf:about="#redige"/>
    </owl:onProperty>
    <owl:allValuesFrom>
     <owl:Class rdf:about="#Demande"/>
    </owl:allValuesFrom>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#MedecinHospitalier"/>
     <owl:Class rdf:ID="MedecinDeFamille"/>
    </owl:unionOf>
   </owl:Class>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#MedecinTraitant"/>
 </owl:Class>
 <owl:Class rdf:ID="AideSoignante">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:someValuesFrom>
     <owl:Class rdf:ID="Nursing"/>
    </owl:someValuesFrom>
    <owl:onProperty>
     <owl:InverseFunctionalProperty rdf:about="#effectue"/>
    </owl:onProperty>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:allValuesFrom>
     <owl:Class rdf:about="#Nursing"/>
```

```
      </owl:allValuesFrom>
      <owl:onProperty>
       <owl:InverseFunctionalProperty rdf:about="#effectue"/>
      </owl:onProperty>
     </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf rdf:resource="#ProfessionnelDeSoins"/>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >L'aide-soignante travaille en étroite collaboration et sous la responsabilité et l'encadrement de
l'infirmière. Elle dispense aussi bien des soins d'hygiène et de prévention et contribue à la réalisation
des actes de la vie quotidienne.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="InfirmiereCoordinatrice">
   <rdfs:subClassOf rdf:resource="#ProfessionnelDeSoins"/>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Elle travaille en partenariat avec l'équipe médicale et paramédicale. Elle assure le lien entre
l'hôpital et l'équipe de soins.</rdfs:comment>
  </owl:Class>
  <owl:Class rdf:ID="ElementPrescription"/>
  <owl:Class rdf:about="#ExamenParaclinique">
   <owl:disjointWith rdf:resource="#ExamenClinique"/>
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >Examen ou technique complémentaire à l'examen clinique pour lequel le médecin peut confirmer
son diagnostic et de suivre l'évolution d'une maladie sous traitement. Les techniques
complémentaires comprennent notamment : les examens de laboratoire, les techniques d'imagerie
médicale.</rdfs:comment>
   <rdfs:subClassOf>
    <owl:Restriction>
     <owl:cardinality rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
     >0</owl:cardinality>
     <owl:onProperty>
      <owl:FunctionalProperty rdf:about="#estEffectuePar"/>
     </owl:onProperty>
    </owl:Restriction>
   </rdfs:subClassOf>
   <rdfs:subClassOf>
    <owl:Class rdf:about="#Examen"/>
   </rdfs:subClassOf>
  </owl:Class>
  <owl:Class rdf:about="#Nursing">
   <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
   >sont des soins d'hygiènes et de préventions effectuées par l'infirmière et/ou l'aide
soignante.</rdfs:comment>
   <rdfs:subClassOf rdf:resource="#SoinsInfirmiers"/>
  </owl:Class>
  <owl:Class rdf:about="#ExamenBiologique">
   <rdfs:subClassOf rdf:resource="#ExamenParaclinique"/>
   <owl:disjointWith rdf:resource="#ExamenImagerie"/>
  </owl:Class>
```

```
 <owl:Class rdf:about="#ProfessionnelDeSanteMedical">
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Ils sont au nombre de deux : médecin et  pharmacien.</rdfs:comment>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >ActeurDeSanteMedical</rdfs:label>
  <rdfs:subClassOf rdf:resource="#ProfessionnelDeSante"/>
 </owl:Class>
 <owl:Class rdf:about="#Examen">
  <rdfs:subClassOf rdf:resource="#Activite"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Peut être un examen clinique ou complémentaire</rdfs:comment>
  <owl:disjointWith rdf:resource="#ActiviteDeReeducationEtReadaptation"/>
  <owl:disjointWith rdf:resource="#SoinsInfirmiers"/>
 </owl:Class>
 <owl:Class rdf:about="#NaturePersonne">
  <owl:equivalentClass>
   <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
     <owl:Class rdf:about="#Femme"/>
     <owl:Class rdf:about="#Homme"/>
    </owl:unionOf>
   </owl:Class>
  </owl:equivalentClass>
 </owl:Class>
 <owl:Class rdf:about="#Demande">
  <rdfs:subClassOf>
   <owl:Restriction>
    <owl:onProperty>
     <owl:FunctionalProperty rdf:about="#estRedigePar"/>
    </owl:onProperty>
    <owl:allValuesFrom rdf:resource="#MedecinPrescipteur"/>
   </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf rdf:resource="#Document"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >est un document rédigé par le médecin prescripteur et évaluée par le médecin coordonateur. Elle
peut être une demande d'admission d'un patient ou une demande de sortie.</rdfs:comment>
 </owl:Class>
 <owl:Class rdf:about="#Medecin">
  <rdfs:subClassOf rdf:resource="#ProfessionnelDeSanteMedical"/>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Il est un professionnel de santé. Il peut exercer à l'hôpital ou avoir une activité libérale. Il peut
être médecin prescripteur ou médecin coordonateur.</rdfs:comment>
 </owl:Class>
 <owl:Class rdf:about="#MedecinDeFamille">
  <rdfs:subClassOf rdf:resource="#MedecinPrescipteur"/>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >MedecinDeVille</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
```

```
  >MdecinLiberal</rdfs:label>
  <rdfs:label rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >MedecinExercantATitreLiberal</rdfs:label>
  <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
  >Est un médecin traitant responsable d'un cabinet médical.</rdfs:comment>
</owl:Class>
<owl:ObjectProperty rdf:ID="prendEnCharge">
  <rdfs:range rdf:resource="#Patient"/>
  <rdfs:domain rdf:resource="#ProfessionnelDeSante"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:about="#estPrisEnChargePar"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#estPrisEnChargePar">
  <rdfs:range rdf:resource="#ProfessionnelDeSante"/>
  <rdfs:domain rdf:resource="#Patient"/>
  <owl:inverseOf rdf:resource="#prendEnCharge"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#evalue">
  <rdfs:range rdf:resource="#Demande"/>
  <rdfs:domain rdf:resource="#MedecinCoordonateur"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="evaluePar"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="dateNaissanceDe">
  <rdfs:domain rdf:resource="#DateHeure"/>
  <owl:inverseOf>
    <owl:FunctionalProperty rdf:about="#neLe"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#Personne"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="figure">
  <rdfs:range rdf:resource="#ElementPrescription"/>
  <owl:inverseOf>
    <owl:FunctionalProperty rdf:ID="contientMedicament"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Medicament"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#aPourMaladie">
  <rdfs:range rdf:resource="#Maladie"/>
  <rdfs:domain rdf:resource="#Patient"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="estMaldieDe"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#membreDe">
  <rdfs:range rdf:resource="#StructurePAD"/>
  <owl:inverseOf>
```

```
    <owl:ObjectProperty rdf:ID="composeDe"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="estDemandePar">
  <rdfs:domain rdf:resource="#ExamenParaclinique"/>
  <rdfs:range rdf:resource="#MedecinTraitant"/>
  <owl:inverseOf>
    <owl:ObjectProperty rdf:ID="demande"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="estSpecialiteDe">
  <rdfs:range rdf:resource="#MedecinSpecialiste"/>
  <rdfs:domain rdf:resource="#Specialite"/>
  <owl:inverseOf>
    <owl:FunctionalProperty rdf:about="#aPourSpecialite"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="estNatureDe">
  <owl:inverseOf>
    <owl:FunctionalProperty rdf:about="#aNature"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#NaturePersonne"/>
  <rdfs:range rdf:resource="#Personne"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#evaluePar">
  <rdfs:domain rdf:resource="#Demande"/>
  <owl:inverseOf rdf:resource="#evalue"/>
  <rdfs:range rdf:resource="#MedecinCoordonateur"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#estMaldieDe">
  <rdfs:range rdf:resource="#Patient"/>
  <rdfs:domain rdf:resource="#Maladie"/>
  <owl:inverseOf rdf:resource="#aPourMaladie"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="dateDocument">
  <rdfs:range rdf:resource="#Document"/>
  <rdfs:domain rdf:resource="#DateHeure"/>
  <owl:inverseOf>
    <owl:FunctionalProperty rdf:ID="redigeLe"/>
  </owl:inverseOf>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#composeDe">
  <owl:inverseOf rdf:resource="#membreDe"/>
  <rdfs:range rdf:resource="#Personne"/>
  <rdfs:domain rdf:resource="#StructurePAD"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:about="#demande">
  <owl:inverseOf rdf:resource="#estDemandePar"/>
```

```
  <rdfs:domain rdf:resource="#MedecinTraitant"/>
  <rdfs:range rdf:resource="#ExamenParaclinique"/>
</owl:ObjectProperty>
<owl:DatatypeProperty rdf:ID="numero_de_telephone">
  <rdfs:domain rdf:resource="#Personne"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="effet_secondaire">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Medicament"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="prenom">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="note">
  <rdfs:domain rdf:resource="#Activite"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="contre_indication">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Medicament"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="effet_indesirable">
  <rdfs:domain rdf:resource="#Medicament"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="e_mail">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="indication">
  <rdfs:domain rdf:resource="#Medicament"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:DatatypeProperty>
<owl:FunctionalProperty rdf:ID="duree">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#ElementPrescription"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="type_document">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#aNature">
  <owl:inverseOf rdf:resource="#estNatureDe"/>
  <rdfs:range rdf:resource="#NaturePersonne"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:domain rdf:resource="#Personne"/>
```

```
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="nom_maladie">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="situation_familiale">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:domain rdf:resource="#Personne"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf rdf:parseType="Resource">
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >celibataire</rdf:first>
        <rdf:rest rdf:parseType="Resource">
          <rdf:rest rdf:parseType="Resource">
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >divorcé</rdf:first>
            <rdf:rest rdf:parseType="Resource">
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >veuf</rdf:first>
            </rdf:rest>
          </rdf:rest>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >marié</rdf:first>
        </rdf:rest>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#neLe">
  <rdfs:domain rdf:resource="#Personne"/>
  <rdfs:range rdf:resource="#DateHeure"/>
  <owl:inverseOf rdf:resource="#dateNaissanceDe"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#effectueLe">
  <rdfs:domain rdf:resource="#Activite"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:ID="dateActivite"/>
  </owl:inverseOf>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#DateHeure"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="mode_administration">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#ElementPrescription"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
</owl:FunctionalProperty>
```

```
<owl:FunctionalProperty rdf:ID="constituePar">
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#constiue"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#StructurePAD"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#MedecinCoordonateur"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="adresse">
  <rdfs:domain rdf:resource="#Personne"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#redigeLe">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <owl:inverseOf rdf:resource="#dateDocument"/>
  <rdfs:domain rdf:resource="#Document"/>
  <rdfs:range rdf:resource="#DateHeure"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="profession">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Patient"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#aPourSpecialite">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#Specialite"/>
  <rdfs:domain rdf:resource="#MedecinSpecialiste"/>
  <owl:inverseOf rdf:resource="#estSpecialiteDe"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="type_entourage">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:domain rdf:resource="#Entourage"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf rdf:parseType="Resource">
        <rdf:rest rdf:parseType="Resource">
          <rdf:rest rdf:parseType="Resource">
            <rdf:rest rdf:parseType="Resource">
              <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
              >collégue</rdf:first>
              <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
            </rdf:rest>
            <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >ami</rdf:first>
          </rdf:rest>
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >proche famille</rdf:first>
        </rdf:rest>
```

```
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >famille</rdf:first>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="numero_de_securite_sociale">
  <rdfs:domain rdf:resource="#Personne"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="nom">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:domain rdf:resource="#Personne"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="description">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="avis">
  <rdfs:domain rdf:resource="#Demande"/>
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf rdf:parseType="Resource">
        <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >refusé</rdf:first>
        <rdf:rest rdf:parseType="Resource">
          <rdf:first rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
          >accepté</rdf:first>
          <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
        </rdf:rest>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="age">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#estEffectuePar">
  <rdfs:domain rdf:resource="#Activite"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#Personne"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#effectue"/>
  </owl:inverseOf>
```

```
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="estSexeDe">
  <rdfs:range rdf:resource="#NaturePersonne"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#aSexe"/>
  </owl:inverseOf>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="faitPartie">
  <rdfs:domain rdf:resource="#ElementPrescription"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#contient"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#PrescriptionMedicamenteuse"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="posologie">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:domain rdf:resource="#ElementPrescription"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#estRedigePar">
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#redige"/>
  </owl:inverseOf>
  <rdfs:range rdf:resource="#Personne"/>
  <rdfs:domain rdf:resource="#Document"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#estEntourageDe">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:ID="aEntourage"/>
  </owl:inverseOf>
  <rdfs:domain rdf:resource="#Entourage"/>
  <rdfs:range rdf:resource="#Patient"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="nom_medicament">
  <rdfs:domain rdf:resource="#Medicament"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#appliqueSur">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#Patient"/>
  <rdfs:domain rdf:resource="#Activite"/>
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#subit"/>
  </owl:inverseOf>
```

```
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#estDocumentDe">
  <owl:inverseOf>
    <owl:InverseFunctionalProperty rdf:about="#aPourDocument"/>
  </owl:inverseOf>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:domain rdf:resource="#Document"/>
  <rdfs:range rdf:resource="#Patient"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="classe_therapeutique">
  <rdfs:domain rdf:resource="#Medicament"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="numero_de_carte_nationnale">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Personne"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="lieu_de_naissance">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="date_heure">
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#DatatypeProperty"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
  <rdfs:domain rdf:resource="#DateHeure"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:about="#contientMedicament">
  <rdfs:domain rdf:resource="#ElementPrescription"/>
  <owl:inverseOf rdf:resource="#figure"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:range rdf:resource="#Medicament"/>
</owl:FunctionalProperty>
<owl:InverseFunctionalProperty rdf:about="#aPourDocument">
  <owl:inverseOf rdf:resource="#estDocumentDe"/>
  <rdfs:range rdf:resource="#Document"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  <rdfs:domain rdf:resource="#Patient"/>
</owl:InverseFunctionalProperty>
<owl:InverseFunctionalProperty rdf:about="#contient">
  <owl:inverseOf rdf:resource="#faitPartie"/>
  <rdfs:range rdf:resource="#ElementPrescription"/>
  <rdfs:domain rdf:resource="#PrescriptionMedicamenteuse"/>
  <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
</owl:InverseFunctionalProperty>
<owl:InverseFunctionalProperty rdf:about="#effectue">
  <rdfs:domain rdf:resource="#Personne"/>
```

```
    <owl:inverseOf rdf:resource="#estEffectuePar"/>
    <rdfs:range rdf:resource="#Activite"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#constiue">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#MedecinCoordonateur"/>
    <rdfs:range rdf:resource="#StructurePAD"/>
    <owl:inverseOf rdf:resource="#constituePar"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#subit">
    <rdfs:domain rdf:resource="#Patient"/>
    <owl:inverseOf rdf:resource="#appliqueSur"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#Activite"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#aSexe">
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#FunctionalProperty"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#NaturePersonne"/>
    <owl:inverseOf rdf:resource="#estSexeDe"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#aEntourage">
    <owl:inverseOf rdf:resource="#estEntourageDe"/>
    <rdfs:domain rdf:resource="#Patient"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#Entourage"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#dateActivite">
    <owl:inverseOf rdf:resource="#effectueLe"/>
    <rdfs:domain rdf:resource="#DateHeure"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:range rdf:resource="#Activite"/>
  </owl:InverseFunctionalProperty>
  <owl:InverseFunctionalProperty rdf:about="#redige">
    <owl:inverseOf rdf:resource="#estRedigePar"/>
    <rdf:type rdf:resource="http://www.w3.org/2002/07/owl#ObjectProperty"/>
    <rdfs:domain rdf:resource="#Personne"/>
    <rdfs:range rdf:resource="#Document"/>
  </owl:InverseFunctionalProperty>
  <RapportSocial rdf:ID="RapportSocial_13"/>
  <ActiviteSociale rdf:ID="ActiviteSociale_6"/>
  <AssistanteSociale rdf:ID="AssistanteSociale_5"/>
</rdf:RDF>

<!-- Created with Protege (with OWL Plugin 3.3.1, Build 430)  http://protege.stanford.edu -->
```