Université Abdelmalek Essaadi
Faculté Polydisciplinaire De Tetouan
LP Administrateur des Applications des Entreprises

L'Ontologie (Informatique)

Réalisé par :

- ***ASKANE Younes***
- ***EL-OUCHI Younes***
- ***EL MESSBAHI Salima***
- ***EL JADIDI ALAOUI Sadik***

Encadré par :

- ***Prof. ALI Younes***
- ***Prof. Mouna HAMDANE***

Année Universitaire 2009/201

Sommaire

Introduction 3

1. Historique et Définition d'ontologie 5

1.1 La Notion d'Ontologie 5

1.2 Historique de L'Ontologie (informatique) 7

1.3 L'objectif d'ontologie 7

2. Web Sémantique 9

2.1 Définition et historique : 9

2.2 Principe général de web sémantique : 10

A. Le RDF 11

B. OWL 12

C. Logique 13

2.4 Exemples pratique 15

A. Manipulation OWL 15

Définition d'un individu 21

Les propriétés 22

Caractéristiques des proprieties 25

Exemple d'ontologie OWL DL - description d'une population 26

Présentation de la population sur laquelle porte l'ontologie 26

Ecriture des classes 28

Ecriture des proprieties 29

Assertion de faits caractéristisant la population 32

B. Protege 35

Conclusion 40

Introduction

Depuis environ une quinzaine d'années maintenant, le concept d'ontologie imprègne les discussions et les pratiques dans les domaines de l'informatique et de l'ingénierie des connaissances (IC). Les pratiques, disons-nous, car on ne se limite plus à parler de l'ontologie mais d'ontologies, désignant par cette double nuance grammaticale des artefacts concrets en lieu et place d'une composante essentielle et millénaire (en dépit d'une appellation tardive) de la philosophie.

A l'instar de tout objet technique, ceux-ci remplissent des usages privilégiés.

En l'occurrence la représentation des connaissances, proches en cela de la démarche philosophique en ce qu'ils opèrent, à leur tour, un inventaire des différentes dimensions de la réalité en mobilisant pour ce faire des formalismes issus des langages informatiques (le Web sémantique s'appuie en particulier sur le langage OWL - Web Ontology Language - lui-même dérivé de la logique des descriptions).

Parallèlement, on observe la poursuite du projet initié par l'intelligence artificielle (AI) d'opérationnaliser le raisonnement humain en ajointant les modélisations des régions de la connaissance à des logiciels effectuant des raisonnements inférentiels – à ceci près, toutefois, que l'intelligence, notamment dans la perspective du Web sémantique, se situe désormais davantage dans les données elles-mêmes que dans les moteurs d'inférences. D'où, précisément, la vigueur actuelle des problématiques ontologiques : si l'intelligence imprègne les données, encore faut-il les mettre en forme de manière idoine ; telle est la tâche qui attend l'ingénierie des connaissances en ce début de XXIe siècle.

La tâche de la philosophie, quant à elle, outre de contribuer à l'effort de clarification nécessaire au recensement et à la cartographie des régions du « ce qu'il y a », sera d'apprécier dans quelle mesure, son projet essentiel désormais portée par ces démarches nouvelles, celles-ci interrogent en retour les conditions de son exercice.

1. Historique et Définition d'ontologie

1.1 La Notion d'Ontologie

En philosophie, l'**ontologie** (du grec ὤν, ὄντος, participe présent du verbe être) est l'étude de l'être en tant qu'être, c'est-à-dire l'étude des propriétés générales de ce qui existe. L'Ontologie est une branche fondamentale de la Métaphysique qui s'intéresse à la notion d'existence, aux catégories fondamentales de l'existant et étudie les propriétés les plus générales de l'être. L'Ontologie a donc un rapport direct avec notre conception de la réalité. L'ontologie qui interroge nos conceptualisations du monde et l'ingénieur qui conçoit ses représentations logicielles ne sont-ils pas plus proches qu'il n'y semblait initialement?

Par analogie, le terme est repris en informatique et en science de l'information, où une **ontologie** est l'ensemble structuré des termes et concepts représentant le sens d'un champ d'informations, que ce soit par les métadonnées d'un espace de noms, ou les éléments d'un domaine de connaissances. L'ontologie constitue en soi un modèle de données représentatif d'un ensemble de concepts dans un domaine, ainsi que les relations entre ces concepts. Elle est employée pour raisonner à propos des objets du domaine concerné.

Les concepts sont organisés dans un graphe dont les relations peuvent être :

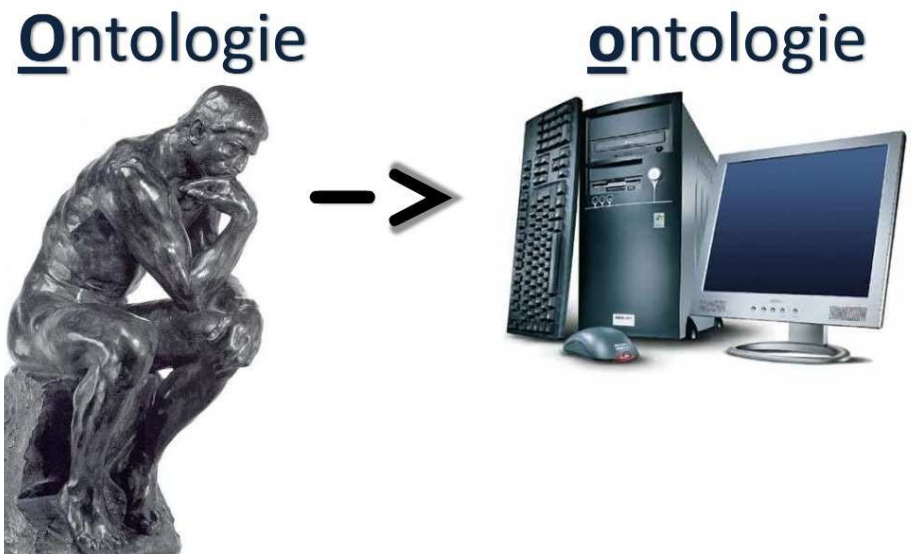
- des relations sémantiques,
- des relations de subsomption (inclusion).

Si la notion d'ontologie informatique et l'exemple de la programmation orientée objets ont cette ressemblance, c'est qu'ils ont un ancêtre commun: les systèmes de l'intelligence artificielle symbolique. Les débuts de cette branche de l'intelligence artificielle se confondent avec les débuts de l'informatique car dès ses prémices, l'informatique a perpétué le rêve d'automate de simuler et dépasser l'intelligence humaine avec des systèmes artificiels.

Si la branche de l'intelligence artificielle à laquelle nous nous intéressons ici est qualifiée de symbolique c'est parce qu'elle repose sur des représentations formelles des connaissances sous la forme de symboles que le système peut stocker et manipuler (ex: langages et opération logiques, structures et opérations de graphes). Contrairement à d'autres approches, ces représentations sont à la fois compréhensibles par les humains et manipulables par les systèmes en appliquant des règles de manipulations définies sur les symboles de ces représentations et dont l'interprétation simule, par exemple, un raisonnement.

1.2 Historique de L'Ontologie (informatique)

"Ontologie" est un mot que l'informatique a emprunté à la philosophie au début des années 90.



Depuis la Création du **Web en 1990**, par Tim Berners Lee,Il était exclusivement destiné à partager des informations sous forme de pages html, affichables par un logiciel «navigateur web», et généralement destinées à être lues par un utilisateur humain.

Ainsi L'arrivée de **XML en 1998**, a donné un cadre à la structuration des connaissances, rendant ainsi possible la création de nouveaux langages web destinés non plus à un rendu graphique à l'écran pour un utilisateur humain, mais à un réel partage et à une manipulation des savoirs.

En 2004 W3C publie RDF et OWL deux technologies clés du Web sémantique :*Ces* deux technologies RDF et OWL, fournissent un cadre de travail pour la gestion des ressources, l'intégration, le partage et la réutilisation des données sur le Web.

1.3 L'objectif d'ontologie

L'objectif premier d'une ontologie est de modéliser un ensemble de connaissances dans un domaine donné, qui peut être réel ou imaginaire.

Les ontologies sont employées dans **l'intelligence artificielle**, le Web sémantique, le **génie logiciel**, l'**informatique biomédicale** et l'**architecture de l'information** comme une forme

de représentation de la connaissance au sujet d'un monde ou d'une certaine partie de ce monde. Les ontologies décrivent généralement :

- Individus : les objets de base,
- Classes : ensembles, collections, ou types d'objets,
- Attributs : propriétés, fonctionnalités, caractéristiques ou paramètres que les objets peuvent posséder et partager,
- Relations : les liens que les objets peuvent avoir entre eux,
- Événements : changements subis par des attributs ou des relations.

EN général le but des « ontologies » serait :

- l'intelligence artificielle (« des machines comprennent des humains »),
- l'interopérabilité (« des machines comprennent des machines »),
- de faire sens pour l'usager (« des humains comprennent des humains via des machines »).

2. Web Sémantique

2.1 Définition et historique :

Le **Web sémantique** désigne un ensemble de technologies visant à rendre le contenu des *ressources* du *World Wide Web* accessible et utilisable par les programmes et agents logiciels, grâce à un système de *métadonnées* formelles, utilisant notamment la famille de langages développés par le *W3C*.

La notion de métadonnées utilisables par les machines fut proposée assez tôt dans l'histoire du Web, dès 1994 par son inventeur Tim Berners-Lee, lors de la conférence WWW 94 où fut annoncée la création du **W3C**. Ces métadonnées formelles sont alors présentées comme une nécessaire représentation utilisable par les machines de l'information contenue dans les documents, par exemple le fait qu'une personne X est employée par une organisation Y.

Le développement de cette idée aboutit à la publication en 1999 de la première version de **RDF** (Resource Description Framework), langage qui définit un cadre général pour la standardisation des métadonnées des ressources Web.

Sur la base de RDF se sont ensuite développés des vocabulaires spécifiques destinés à des applications particulières, comme FOAF destiné à décrire les relations entre personnes, puis des langages destinés à structurer ces vocabulaires, comme **RDFS** et le langage d'ontologie **OWL**, publiés dans leur forme finale en février 2004.

Au cours de cette évolution, la notion de *ressource* s'étend de son sens original de « document publié sur le Web » à des sens plus généraux et plus abstraits. Dans les langages d'ontologie ou le langage SKOS, en cours de développement, les ressources décrites sont des concepts comme des classes, des propriétés, ou des concepts utilisés pour l'indexation. À ce titre, les langages et technologies du Web sémantique sont parfois présentés comme des outils de représentation des connaissances adaptés à l'environnement Web, permettant de transformer automatiquement les données en information, et les informations en savoir.

2.2 Principe général de web sémantique :

Le Web sémantique est entièrement fondé sur le **Web** et ne remet pas en cause ce dernier. Le Web sémantique s'appuie donc sur la fonction primaire du **Web** « classique » : un moyen de publier et consulter des **documents**. Mais les documents traités par le Web sémantique contiennent non pas des textes en **langage naturel** (français, espagnol, chinois, etc.) mais des informations formalisées pour être traitées automatiquement. Ces documents sont générés, traités, échangés par des **logiciels**. Ces logiciels permettent souvent, sans connaissance informatique, de :

- générer des **données** sémantiques à partir de la saisie d'information par les utilisateurs ;
- agréger des données sémantiques afin d'être publiées ou traitées ;
- publier des données sémantiques avec une mise en forme personnalisée ou spécialisée ;
- échanger automatiquement des données en fonction de leurs relations sémantiques ;
- générer des données sémantiques automatiquement, sans saisie humaine, à partir de règles d'inférences.

2.3 Introduction aux technologies du web sémantique

Le partage de la connaissance reste toujours un problème important à traiter dans les organisations. Des études en Web sémantique et les interfaces homme machine fournissent quelques techniques pour guider l'accès à l'information, ces techniques ne sont pas suffisantes pour donner un accès contextuel aux besoins de l'utilisateur. En fait, avec les techniques du Web sémantiques [Berners-Lee, et al, 01] l'accès à l'information par l'utilisateur est guidé par l'ontologie du domaine. Cependant ce type d'ontologie est établi comme une définition consensuelle de concepts du domaine qui peut être considéré comme une référence des concepts utilisés dans un domaine donné. Généralement l'utilisateur a sa propre représentation du domaine qui est plus au moins proche de l'ontologie définie. Notons également, que la structure de cette ontologie est généralement décrite par un ingénieur de connaissance qui introduit des biais d'interprétation.

L'activité de l'utilisateur change à travers le temps ainsi que ses besoins d'accès à l'information. Plusieurs travaux en interface homme machine [Jing et autres, 02] étudient des techniques pour capturer l'activité de l'utilisateur afin d'offrir une recherche d'information contextuelle. Des études en ergonomie montrent que nous avons besoin de connaître les objectifs de l'activité et son environnement avant d'observer cette activité, ce qui n'est pas facile si la traçabilité des informations est réalisée d'une façon générique indépendamment de la tâche de l'utilisateur. La recherche d'information peut être basée sur les relations entre ces concepts, qui peuvent offrir un accès contextuel à l'information. Nous exploitons les graphes conceptuels pour représenter le réseau sémantique et nous utilisons le moteur de recherche d'information Corese [Corby et al, 02] qui se base sur des fonctions de spécialisation, de généralisation et de projection pour offrir une recherche d'information en profondeur.

On présentera donc, dans une première partie, les langages XML et RDF, ainsi que les principes fondamentaux du Web sémantique. La deuxième partie, qui constitue l'introduction

à OWL proprement dite, s'appuie largement sur les Recommandations du W3C. Enfin, une dernière partie propose quelques exemples de logiciels en lien direct avec OWL et le Web sémantique.

A. Le RDF

RDF (*Resources Description Framework*) est un langage servant à décrire des ressources. Son élément constitutif de base est la formulation d'un triplet se composant d'une entité (appelée ressource en terminologie web), d'une propriété et d'une valeur (qui peut être une autre ressource). La formulation) est essentiellement la définition d'un fait P (a,b) où P est une propriété binaire, et (a,b) sont des ressources. Dans la perspective du web sémantique, RDF définit une couche située au-dessus de XML. De ce fait, RDF a été doté d'une syntaxe XML.
RDF est indépendant du domaine, en d'autres mots, il ne pose aucun présupposé quant à un domaine spécifique. C'est donc à l'utilisateur que revient le rôle de définir sa propre terminologie dans un langage schéma appelé RDF Schema (RDFS). Constitutivement,

RDFS est un langage d'ontologies primitif (ou naturel) proposant les caractéristiques suivantes:

- Organisation des objets en classes (professeur, membre du personnel, étudiant, cours, cours pour étudiants) et propriétés binaires (enseigne, étudie, travaille).
- Sous-classes (tous les professeurs sont des membres du personnel) et relations des sous-propriétés (tout chef de département fait partie de ce département).
- Restrictions de domaine (seul le personnel académique peut enseigner) et d'étendue (une personne ne peut qu'enseigner) au niveau des propriétés. RDF et RDFS fournissent les langages de base pour le web sémantique.

B. OWL

La puissance d'expression de RDF et de RDFS est volontairement très limitée: RDF est (en gros) limité à des attributs binaires et RDFS est (toujours en gros) limité aux hiérarchies de sous-classes et de sous-propriétés, avec restrictions de domaine et d'entendue des propriétés.

Il existe cependant plusieurs cas particuliers d'utilisation du web sémantique qui nécessitent une plus grande expressivité.

Ce type d'extensions comprend:

- la *disjonction* (par ex. une personne ne peut être à la fois professeur et membre du personnel administratif).
- les *combinaisons booléennes des classes* (par ex. le personnel est l'ensemble du corps académique, du personnel administratif et du personnel d'assistance technique).
- les *restrictions de cardinalité* (par ex. un service ne peut avoir qu'un seul chef).
- les *caractéristiques spéciales des propriétés* (par ex. "supérieur de" est transitif, "enseigne" et " reçoit des cours de " sont des propriétés inverses)
- l'*étendue locale des propriétés ("range")* définit la portée d'une propriété, par exemple "mange" pour toutes les classes. Dans certains cas, on peut souhaiter réduire la portée en fonction de la classe. Par exemple, les vaches ne mangent que de l'herbe tandis que d'autres animaux mangent également de la viande.

OWL a été élaboré comme nouveau langage d'ontologies standard pour le web. Il repose sur RDFS et tente de trouver un équilibre entre puissance d'expression et support

logique efficace. La logique (le raisonnement) est un facteur important parce qu'elle permet de

A. vérifier la cohérence d'une ontologie et des connaissances,
B. vérifier la présence de relations non voulues entre classes
C. classer automatiquement les instances en classes.

C. Logique

La création formelle du langage OWL est une partie de la représentation et du raisonnement des connaissances appelée logique descriptive. Cette création est riche de promesses; l'approche est différente pour la représentation et le raisonnement sur la base de règles. Ses principaux avantages sont:

- Les moteurs de règles existent et sont très puissants.
- Les règles sont bien connues et s'utilisent en informatique générale. Elles sont faciles à apprendre.

Les systèmes de règles peuvent être envisagés comme une extension ou une alternative à OWL. La première idée est de poursuivre les recherches actuelles en visant à intégrer les règles et la logique descriptive tout en maintenant un support logique assez efficace. Une idée plus récente étudie l'utilisation de RDF/S conjointement aux règles comme base d'un autre langage ontologique web.

Outre les systèmes classiques à base de règles, il est intéressant d'analyser ceux capables de prendre en compte des conclusions contradictoires. Ces systèmes sont utiles pour la modélisation des données ayant hérités de défauts et des règles comportant des exceptions. Ils sont aussi très pratiques pour l'intégration des connaissances où des incohérences peuvent évidemment se produire lorsqu'on assemble des connaissances de sources différentes.

Une ontologie décrit les concepts et les rapports qui sont importants dans un domaine particulier, fournissant un vocabulaire pour ce domaine aussi bien que des spécifications automatisées de la signification des termes utilisés dans le vocabulaire.

Les Ontologies traitent de taxonomies et de classifications, schémas de base de données, et de théories entièrement axiomatisées. Ces dernières années, des ontologies ont été introduites dans beaucoup de communautés scientifiques de manière à partager, réutiliser et traiter la connaissance d'un domaine spécifique. Les Ontologies sont devenues vitales pour beaucoup d'applications telles que des portails de la connaissance scientifique, des systèmes de gestion d'information et d'intégration, du commerce électronique, et de services de Web sémantique. L'absence d'une véritable solution générique au niveau de la norme MPEG 7 de l'ISO à constitué un handicap majeur! Cerise sur le gâteau, les scientifiques de certaines universités ont développé des technologies nouvelles qui sont de véritables moteurs d'ontologie qui vont grandement simplifier la vie de tous ceux qui rêvent de produire des outils en OWL, RDF, … Un moteur d'ontologies est un processus applicatif qui possède une dimension sémantique (définition et mise en relation de concepts); une dimension logique (validation de la création de relations entre concepts ou déduction de relations non explicites entre concepts) et enfin une dimension usage avec la construction d'un vocabulaire («outil», «utilisateur», «traitement»), la construction d'une syntaxe (sujet, verbe, complément) et d'une grammaire: «l'utilisateur» «accorde» «l'outil» pour réaliser «un traitement ».

La finalité du moteur d'ontologie est de:

- réduire la distance entre le langage de la machine (logique) et le langage de l'utilisateur (lisible) en intercalant entre l'une et l'autre un langage compréhensible par l'un et l'autre;
- réduire la distance entre des langages pratiqués dans différentes cultures (métier, région, temps, etc.) en intercalant entre l'une et l'autre un langage interprétable par l'un et l'autre;
- améliorer la qualité, l'efficacité et l'efficience des échanges entre les acteurs d'un projet via un socle sémantique commun;

Un moteur d'ontologies est un processus applicatif qui possède:

Une dimension sémantique: Définir des concepts Mettre en relation des concepts Une dimension logique: Valider la création de relations entre concepts Déduire des relations non explicites entre concepts Une dimension usage: Construction d'un vocabulaire («outil», «utilisateur», «traitement») Construction d'une syntaxe (sujet, verbe, complément) Construction d'une grammaire: «l'utilisateur» «accorde» «l'outil» pour réaliser «un traitement».

2.4 Exemples pratique

A. Manipulation OWL

La structure d'une ontologie OWL.

Cette partie indique les principaux éléments constituant une ontology OWL. Avant de poursuivre, il est bien nécessaire de comprendre qu'OWL n'a pas été conçu dans un esprit fermé permettant de borner les frontières d'une ontologie. Au contraire, la conception d'OWL a pris en compte la nature distribuée du web sémantique et, de ce fait, a intégré la possibilité d'étendre des ontologies existantes, ou d'employer diverses ontologies existantes pour compléter la définition d'une nouvelle ontologie.

Espaces de nommage

Afin de pouvoir employer des termes dans une ontologie, il est nécessaire d'indiquer avec précision de quels vocabulaires ces termes proviennent. C'est la raison pour laquelle, comme tout autre document XML, une ontologie commence par une déclaration d'espace de nom

(parfois appelée « de nommage ») contenue dans une balise rdf:RDF. Supposons que nous souhaitons écrire une ontologie sur une population de personnes ou, d'une manière plus générale, sur l'humanite. Voici la déclaration d'espace de nom qui pourrait être employée :

```
<rdf:RDF
xmlns = "http://domain.tld/path/humanite#"
xmlns:humanite= "http://domain.tld/path/humanite#"
```

```
xmlns:base = "http://domain.tld/path/humanite#"
xmlns:vivant = "http://otherdomain.tld/otherpath/vivant#"
xmlns:owl = "http://www.w3.org/2002/07/owl#"
xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd = "http://www.w3.org/2001/XMLSchema#">
```

Les deux premières décalarations identifient l'espace de nommage propre à l'ontologie que nous sommes en train d'écrire. La première déclaration d'espace de nom indique à quelle ontologie se rapporter en cas d'utilisation de noms sans préfixe dans la suite de l'ontologie. La troisième déclaration identifie l'URI de base de l'ontologie courante.

La quatrième déclaration signifie simplement que, au cours de la rédaction de l'ontologie humanité, on va employer des concepts développés dans une ontologie vivant, qui décrit ce qu'est un être vivant. Les quatre dernières déclaration introduisent le vocabulaire d'OWL et les objets définis dans l'espace de nommage de RDF, du schéma RDF et des types de données du Schéma XML. Afin de simplifier l'écriture des URI dans la déclaration d'espace de nom et, surtout, dans les valeurs des attributs de l'ontologie, il est conseillé de définir des abbréviations au moyen d'entités de type de document :

```
<!DOCTYPE rdf:RDF [
<!ENTITY humanite "http://domain.tld/path/humanite#" >
<!ENTITY vivant "http://otherdomain.tld/otherpath/vivant#" >
]>
```

Ainsi, la déclaration d'espace de nom initiale devient :

```
<rdf:RDF
xmlns = "&humanite;"
```

```
xmlns:humanite= "&humanite;"
xmlns:base = "&humanite;"
xmlns:vivant = "&vivant;"
xmlns:owl = "http://www.w3.org/2002/07/owl#"
xmlns:rdf = "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:rdfs = "http://www.w3.org/2000/01/rdf-schema#"
xmlns:xsd = "http://www.w3.org/2001/XMLSchema#">
```

En-têtes d'une ontologie

Tout comme il existe une section d'en-tête <head>..</head> en haut de tout document XHTML bien formé, on peut écrire, à la suite de la déclaration d'espaces de nom, un en-tête décrivant le contenu de l'ontologie courante. C'est la balise owl:Ontology qui permet d'indiquer ces informations :

```
<owl:Ontology rdf:about="">
<rdfs:comment>Ontologie décrivant l'humanité</rdfs:comment>
<owl:imports
rdf:resource="http://otherdomain.tld/otherpath/vivant"/>
<rdfs:label>Ontologie sur l'humanité</rdfs:label>
...
```

Le contenu et la signification des différents éléments de cette section d'en-tête sont parfaitement décrits dans « OWL Web Ontology Language Reference » [W3C 2004k], et il est recommandé de s'y reporter pour plus de détails.

Eléments du langage

Cette partie ne va pas reprendre toutes les finesses de OWL Lite, OWL DL et OWL Full, mais uniquement les plus importantes. Pour des

explications exhaustives du rôle de chacun des éléments composant le vocabulaire d'OWL, il est recommandé de se reporter à la Recommandation du W3C « OWL Web Ontology Language Reference » [W3C 2004k].

Les classes

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

Déclaration de classe

La déclaration d'une classe se fait par le biais du mécanisme de « description de classe », qui se présente sous diverses formes. Une classe peut ainsi se déclarer de six manières différentes :

- l'indicateur de classe : La description de la classe se fait, dans ce cas, directement par le nommage de cette classe. Une classe « humain » se déclare de la manière suivante :

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

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

- l'énumération des individus composant la classe :

Ce type de descrition se fait en énumérant les instances de la

classe, à l'aide de la propriété owl:oneOf :

```
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
<owl:Thing rdf:about="#Damien" />
<owl:Thing rdf:about="#Olivier" />
<owl:Thing rdf:about="#Philippe" />
<owl:Thing rdf:about="#Xavier" />
<owl:Thing rdf:about="#Yves" />
</owl:oneOf>
</owl:Class>
```

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

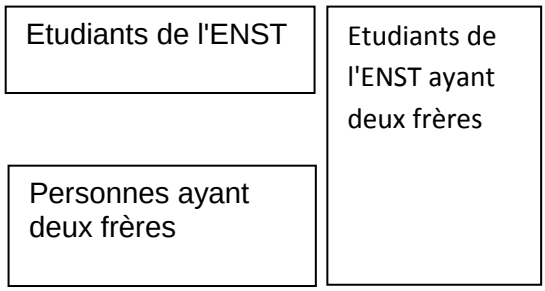
- La restriction de propriétés :

La description par restriction de propriété permet de définir une classe anonyme composée de toutes les instances de owl:Thing qui satisfont une ou plusieurs propriétés. Ces contraintes peuvent être de deux types : contrainte de valeur ou contrainte de cardinalité. Une contrainte de valeur s'exerce sur la valeur d'une certaine propriété de l'individu (par exemple, pour un individu de la classe Humain, sexe = Homme), tandis qu'une contrainte de cardinalité porte sur le nombre de valeurs que peut prendre une propriété (par exemple, pour un individu de la classe Humain, aPourFrere est une propriété qui peut ne pas avoir de valeur, ou avoir plusieurs valeurs, suivant le nombre de frères de l'individu. La contrainte de cardinalité portant sur aPourFrere restreindra donc la classe décrite aux individus pour lesquels la propriété aPourFrere apparaît un certain nombre de fois).

Il existe naturellement différents opérateurs de comparaison pour établir les contraintes. Pour plus de détails, se reporter à « OWL Web Ontology Language Reference » [W3C 2004k].

- Enfin, les descriptions par intersection, union ou complémentaire permettent de décrire une classe par, comme leur nom l'indique, l'intersection, l'union ou le complémentaire d'autres classes déjà définies, ou dont la définition se fait au sein même de la défition de la classe courante :

```
<owl:Class>
  <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#etudiantsENST" />
        <owl:Restriction>
        <owl:onProperty rdf:resource="#aPourFrere" />
  <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
</owl:cardinality>
```



Héritage

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

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

<owl:Class rdf:ID="Homme">
<rdfs:subClassOf rdf:resource="#Humain" />
</owl:Class>

<owl:Class rdf:ID="Femme">
<rdfs:subClassOf rdf:resource="#Humain" />
```

```
</owl:Class>
```

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

Les instances de classe

Définition d'un individu

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

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

```
<Humain rdf:ID="Pierre">
<aPourPere rdf:resource="#Jacques" />
<aPourFrere rdf:resource="#Paul" />
</Humain>
```

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

```
<Humain>
<aPourPere rdf:resource="#Jacques" />
<aPourFrere rdf:resource="#Paul" />
```

```
</Humain>
```

Ce fait décrit, dans ce cas, l'existence d'un Humain dont le père se nomme « Jacques »

et qui a un frère nommé « Paul ».

- les faits concernant l'identité des individus

Une difficulté qui peut éventuellement apparaître dans le nommage des individus concerne la non-unicité éventuelle des noms attribués aux individus. Par exemple, un même individu pourrait être désigné de plusieurs façons différentes.

C'est la raison pour laquelle OWL propose un mécanisme permettant de lever cette ambiguité, à l'aide des propriétés owl:sameAs, owl:diffrentFrom et owl:allDifferent. L'exemple suivant permet de déclarer que les noms « Louis_XIV » et « Le_Roi_Soleil » désignent la même personne :

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

Affecter une propriété à un individu

Une fois que l'on sait écrire une classe en OWL, la création d'une ontologie se fait par l'écriture d'instances de ces objets, et la description des relations qui lient ces instances.

Les propriétés

Maintenant que l'on sait écrire des classes OWL, il ne manque plus que la capacité à exprimer des faits au sujet de ces classes et de leurs instances. C'est ce que permettent de faire les propriétés OWL. OWL fait la distinction entre deux types de propriétés :

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

Une propriété d'objet est une instance de la classe owl:ObjectProperty, une propriété de type de donnée étant une instance de la classe owl:DatatypeProperty. Ces deux classes sont elles-même sous-classes de la classe RDF rdf:Property. La définition des caractéristiques d'une propriété se fait à l'aide d'un axiome de propriété qui, dans sa forme minimale, ne fait qu'affirmer l'existence de la propriété :

```
<owl:ObjectProperty rdf:ID="aPourParent" />
```

Cependant, il est possible de définir beaucoup d'autres caractéristiques d'une propriété dans un axiome de propriété.

Définition d'une propriété

Afin de spécifier une propriété, il existe différentes manières de restreindre la relation qu'elle symbolise. Par exemple, si on considère que l'existence d'une propriété pour un individu donné de l'ontologie constitue une fonction faisant correspondre à cet individu un autre individu ou une valeur de donnée, alors on peut préciser le domaine et l'image de la propriété. Une propriété peut également être définie comme la spécialisation d'une autre propriété.

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

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

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

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

Dans ce cas, anneDeNaissance fait correspondre aux instances de la classe de

dateDeNaissance des entiers positifs.

On peut également employer un mécanisme de hiérarchie entre les propriétés, exactement comme il existe un mécanisme d'héritage sur les classes :

```
<owl:Class rdf:ID="Humain" />
<owl:ObjectProperty rdf:ID="estDeLaFamilleDe">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Humain" />
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="aPourFrere">
<rdfs:subPropertyOf rdf:resource="#estDeLaFamilleDe" />
<rdfs:range rdf:resource="#Humain" />
...
```

```
</owl:ObjectProperty>
</owl:Class>
```

La propriété aPourFrere est une sous-propriété de estDeLaFamilleDe, ce qui signifie que toute entité ayant une propriété aPourFrere d'une certaine valeur a aussi une propriété estDeLaFamilleDe de même valeur.

Caractéristiques des proprieties

En plus de ce mécanisme d'héritage et de restriction du domaine et de l'image d'une propriété, il existe divers moyens d'attacher des caractéristiques aux propriétés, ce qui permet d'affiner grandement la qualité des raisonnements liés à cette propriété. Parmi les caractéristiques de propriétés principales, on trouve la transitivité, la symétrie, la fonctionnalité, l'inverse, etc. L'ajout d'une caractéristique à une propriété de l'ontologie se fait par l'emploi de la balise OWL type :

```
<owl:ObjectProperty rdf:ID="aPourFrere">
<rdf:type rdf:resource="&owl;SymmetricProperty" />
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Humain" />
</owl:ObjectProperty>
```

Propriété non symétrique

```
<Humain rdf:ID="Pierre">
<aPourFrere rdf:resource="#Paul" />
<aPourPere rdf:resource="#Jacques" />
</Humain>
```

Propriété symétrique

```
<Humain rdf:ID="Pierre">
<aPourFrere rdf:resource="#Paul" />
<aPourPere rdf:resource="#Jacques" />
</Humain>
```

L'Humain Pierre a pour frère Paul, de même que (symétrie) l'Humain

Paul a pour frère Pierre. Par contre, si Pierre a pour père Jacques, l'inverse n'est pas vrai (aPourPere n'est pas symétrique).

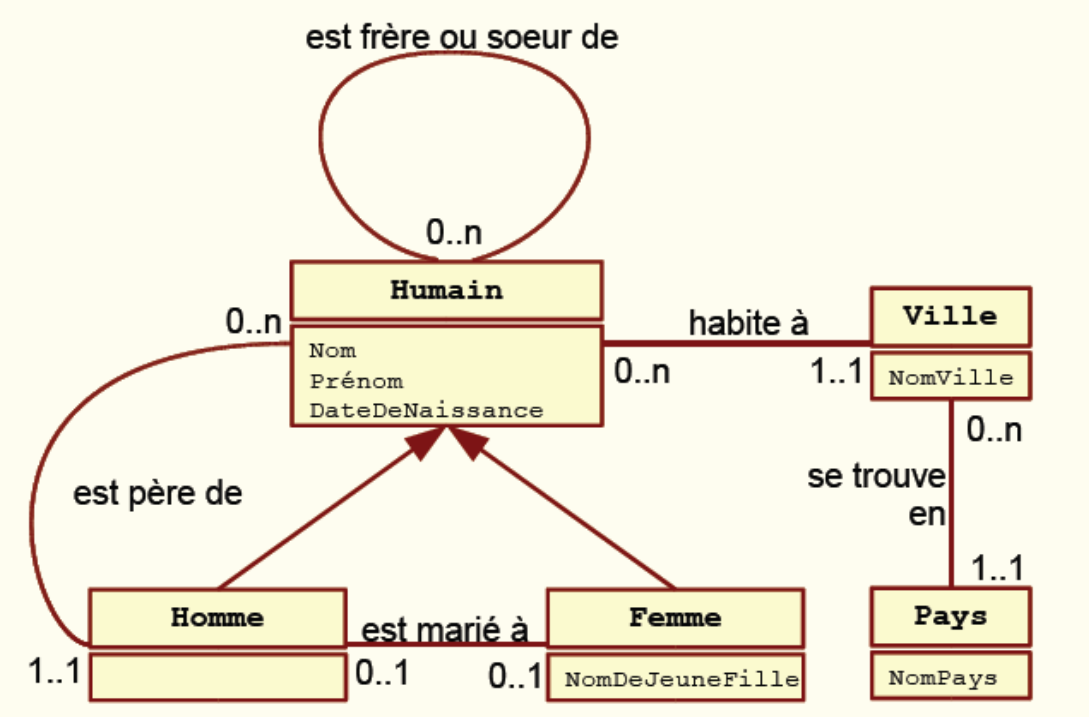
Exemple d'ontologie OWL DL - description d'une population

L'ontologie OWL présentée dans ce paragraphe décrit un groupe de personnes et leurs relations de parenté, ainsi que leur lieu d'habitation. Cette ontologie ne fera pas l'objet de manipulations (édition, validation, raisonnement automatique, etc.) dans cette partie. Pour cela, il est préférable de se référer au chapitre « Outils disponibles ».

Présentation de la population sur laquelle porte l'ontologie

Le diagramme de classes UML de la figure « Classes composant la population à décrire »

indique les classes qui composent la population dont on souhaite écrire une ontologie OWL :



La population que l'on souhaite décrire est composée d'humains, divisés en deux sous-classes Homme et Femme, et habitant dans une certaine ville. Un humain peut avoir un lien de fraternité avec un autre humain. Un homme peut être père d'un autre humain, et un homme et une femme peuvent être mariés.

Le schéma suivant indique plus précisémment les individus qui existent dans l'ontologie que nous allons écrire :

Le monde que l'on va décrire comporte trois instances de la classe « Homme » (en vert), une instance de la classe « Femme » (en rouge), ainsi que deux instances de « Ville » (en bleu) et de « Pays » (en mauve).

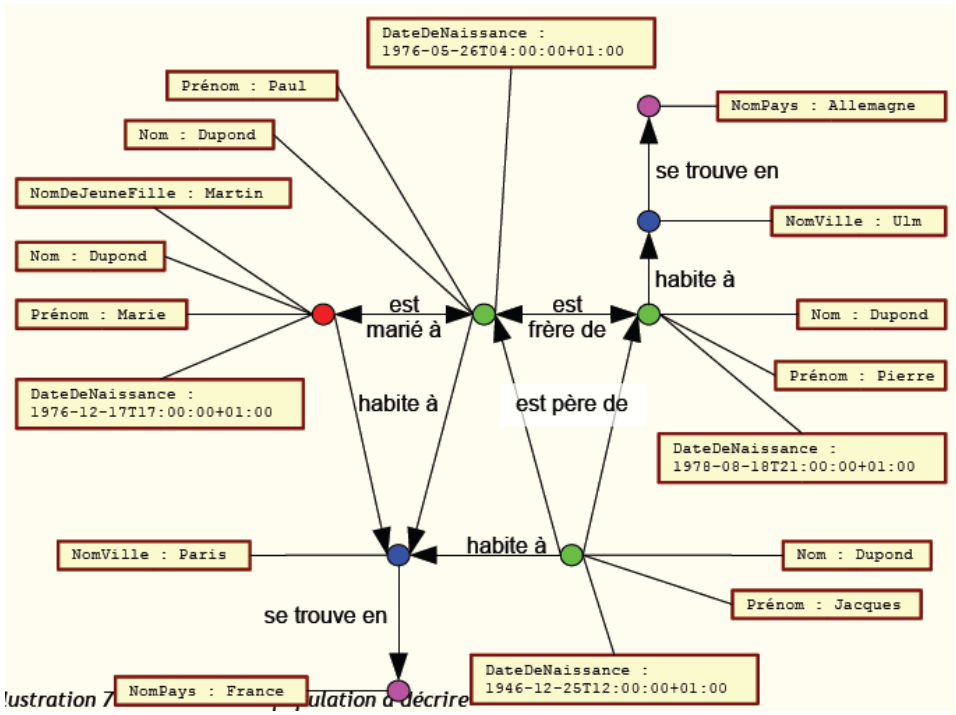


Illustration 7 – Population à décrire

Ecriture des classes

La première étape de l'écriture de l'ontologie OWL représentant cette population consiste à écrire les classes du monde. Il est possible de définir avec plus ou moins de précision la nature des éléments qui composent le monde que l'on décrit, en créant plus ou moins de classes. Par exemple, il serait possible d'écrire une classe « date » dont seraient instances toutes les dates de l'ontologie. Cela pourrait avoir un interêt dans le cadre d'une population plus vaste, pour rechercher, par exemple, les personnes nées un même jour.

On peut se poser la question du type de déclaration de classe à utiliser : par indicateur de classe ou par énumération ? L'énumération comporte, dans le cas de cette ontologie, l'inconvénient de brider l'extensibilité du monde. En conséquence, on choisit de déclarer Humain par le biais d'un indicateur de classe :

```
<!-- Défintion des classes -->
<owl:Class rdf:ID="Humain" />

<owl:Class rdf:ID="Homme">
<rdfs:subClassOf rdf:resource="#Humain" />
</owl:Class>

<owl:Class rdf:ID="Femme">
<rdfs:subClassOf rdf:resource="#Humain" />
</owl:Class>

<owl:Class rdf:ID="Ville" />
<owl:Class rdf:ID="Pays" />
```

Cette première définition des classes de l'ontologie est satisfaisante, mais elle est néanmoins perfectible. En effet, on peut ajouter des contraintes, notamment sur les classes Humain et Ville :

- un humain a toujours un père.

- Une ville se trouve forcément dans un pays. D'autres contraintes sont imaginables, notamment les contraintes d'unicité des noms, dates de naissance, etc. Ces contraintes sont consultables dans le code complet de l'ontologie OWL, consultable en Annexe de ce document. Voici la nouvelle écriture, plus détaillée, des deux classes « Humain » et « Ville » :

```
<owl:Class rdf:ID="Humain">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#aPourPere" />
<owl:cardinality
rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:ID="Ville">
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#seTrouveEn" />
<owl:cardinality
rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
</owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

Ecriture des proprieties

L'écriture des propriétés est l'étape qui va permettre de détailler la population que l'on veut décrire. Ecrivons tout d'abord les propriétés d'objet : habiteA, aPourPere, aUnLienDeFraternite,

estMarieA et seTrouveEn :

```
<!-- Propriétés d'objet -->
<owl:ObjectProperty rdf:ID="habiteA">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Ville" />
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="aPourPere">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Homme" />
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="aUnLienDeFraternite">
<rdf:type rdf:resource="&owl;SymmetricProperty" />
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Humain" />
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="estMarieA">
<rdf:type rdf:resource="&owl;SymmetricProperty" />
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="#Humain" />
</owl:ObjectProperty>

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

Les propriétés aUnLienDeFraternite et estMarieA sont définies comme des propriétés symétriques : si une telle relation lie un individu A à un individu B, alors la même relation lie également l'individu B à l'individu A.

Passons maintenant aux propriétés de type de donnée. Ce sont ces propriétés qui, concrètement, permettent d'affecter des propriétés quantifiées aux instances des classes (par exemple, un nom, un prénom, une date de naissance, etc.) :

```
<!-- Propriétés de type de donnée -->
<owl:DatatypeProperty rdf:ID="nom">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="prenom">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="nomDeJeuneFille">
<rdfs:domain rdf:resource="#Femme" />
<rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="dateDeNaissance">
<rdfs:domain rdf:resource="#Humain" />
<rdfs:range rdf:resource="&xsd;date" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="nomVille">
<rdfs:domain rdf:resource="#Ville" />
<rdfs:range rdf:resource="&xsd;string" />
```

```
</owl:DatatypeProperty>

<owl:DatatypeProperty rdf:ID="nomPays">
<rdfs:domain rdf:resource="#Pays" />
<rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

L'image de ces propriétés est définie selon XML Schema (auquel fait référence la chaine « &xsd; »), ce qui permet de donner un type aux propriétés.

Assertion de faits caractéristisant la population

La dernière étape concerne l'assertion des faits caractérisant la population, présentés sur la figure « Individus de la population à décrire ». Il s'agit donc non seulement de l'instanciation des individus de la population, mais également de leur description par l'énonciation de leurs propriétés :

```
<!-- Humains -->
<Homme rdf:ID="unknown">
<aPourPere rdf:resource="#unknown" />
</Homme>

<Homme rdf:ID="Pierre">
<nom>Dupond</nom>
<prenom>Pierre</prenom>
<dateDeNaissance>1978-08-18</dateDeNaissance>
<aUnLienDeFraternite rdf:resource="#Paul" />
<aPourPere rdf:resource="#Jacques" />
<habiteA rdf:resource="#Ulm" />
</Homme>
```

```
<Homme rdf:ID="Paul">
<nom>Dupond</nom>
<prenom>Paul</prenom>
<dateDeNaissance>1976-05-26</dateDeNaissance>
<estMarieA rdf:resource="#Marie" />
<aPourPere rdf:resource="#Jacques" />
<habiteA rdf:resource="#Paris" />
</Homme>

<Homme rdf:ID="Jacques">
<nom>Dupond</nom>
<prenom>Jacques</prenom>
<dateDeNaissance>1946-12-25</dateDeNaissance>
<aPourPere rdf:resource="#unknown" />
<habiteA rdf:resource="#Paris" />

</Homme>

<Femme rdf:ID="Marie">
<nom>Dupond</nom>
<prenom>Marie</prenom>
<dateDeNaissance>1976-12-17</dateDeNaissance>
<nomDeJeuneFille>Martin</nomDeJeuneFille>
<aPourPere rdf:resource="#unknown" />
<habiteA rdf:resource="#Paris" />
</Femme>

<!-- Pays -->
<Pays rdf:ID="Allemagne">
<nomPays>Allemagne</nomPays>
</Pays>
<Pays rdf:ID="France">
```

```
<nomPays>France</nomPays>
</Pays>


<!-- Villes -->
<Ville rdf:ID="Paris">
<nomVille>Paris</nomVille>
<seTrouveEn rdf:resource="#France" />
</Ville>


<Ville rdf:ID="Ulm">
<nomVille>Ulm</nomVille>
<seTrouveEn rdf:resource="#Allemagne" />
</Ville>
```

On peut observer l'existence d'un individu « unknown », dont la présence est nécessitée par la contrainte de parenté qui caractérise un humain. Sans cet individu, « Jacques et « Marie », dont le père n'apparait pas dans la population à décrire, poseraient un problème d'inconsistance vis-à-vis de la définition de la classe « Humain ».
Cette ontologie exemple est maintenant écrite. La partie suivante, qui traite des outils existants, utilisera cette ontologie comme référence.

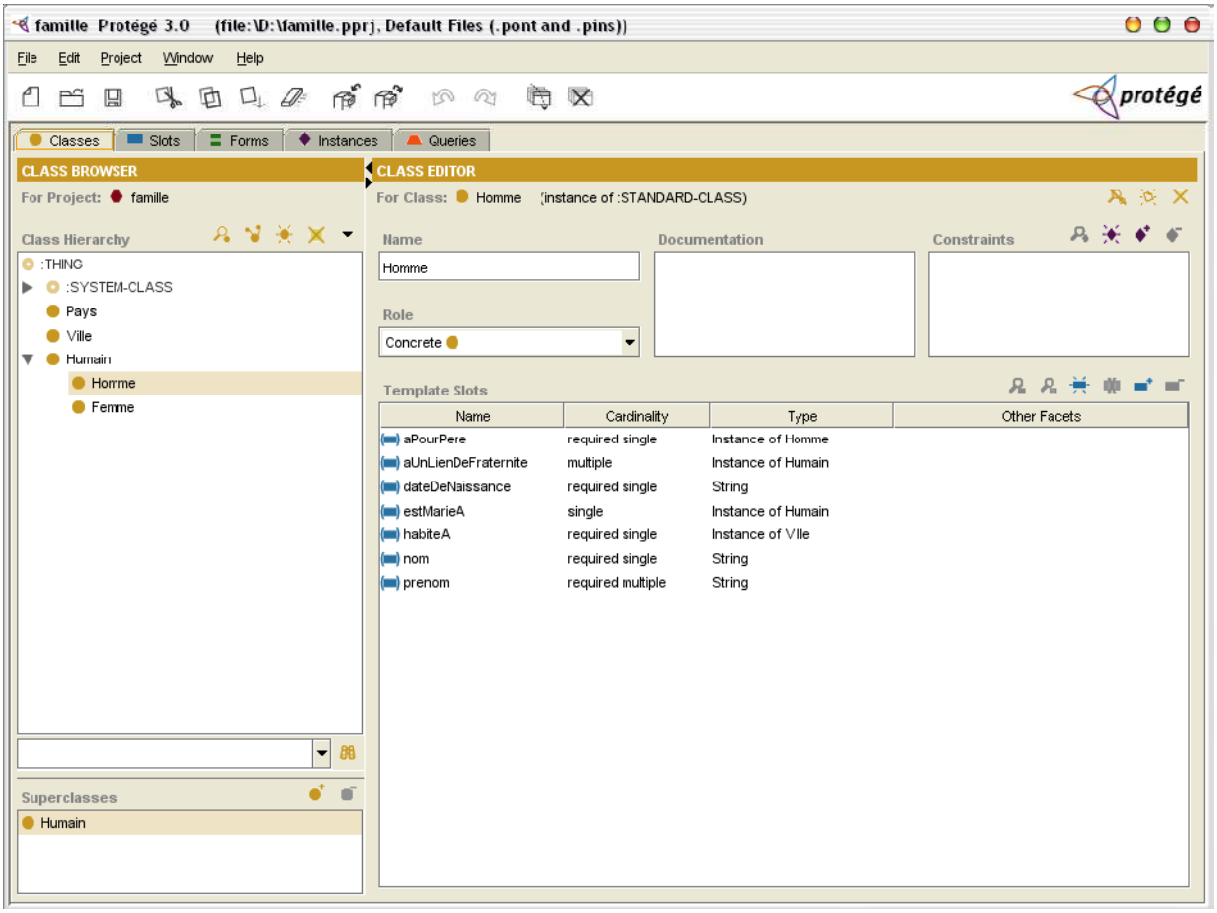
B. Protege

«Protege» est une plate-forme ouverte développée par l'université de Stanford et qui fournit à une communauté d'utilisateur une série d'outils logiciels pour construire des modèles de domaine et des applications basées sur la connaissance des ontologies. En son sein, «Protege» met en application un ensemble riche de structures de «connaissance-modélisation et actions» qui soutiennent la création, la visualisation, et la manipulation des ontologies dans divers formats de représentation. «Protege» peut être adapté aux besoins d'un client pour fournir l'appui logistique à la création de modèles de la connaissance de saisie de données. «Protege», basé sur Java, est extensible, et fournit un environnement prêt à l'emploi qui en fait une base flexible pour le prototypage et le développement d'applications rapides. De plus, «Protege» peut être utilisé en mode Plug-in avec une interface de programmation API pour construire des outils basés sur la connaissance et le développement d'applications.

La plate-forme Protégé propose deux manières principales de modeler des ontologies:

- L'éditeur «Protege - Frames» permet à des utilisateurs d'établir et populer des ontologies basées sur le protocole ouvert de connectivité de base de connaissance (OKBC). Dans ce modèle, une ontologie se compose d'un ensemble de classes organisées dans une hiérarchie pour représenter les concepts fondateurs d'un domaine, un ensemble de connecteurs associés aux classes pour décrire leurs propriétés et rapports, et un ensemble d'exemples de ces classes – différents exemplaires des concepts qui tiennent des valeurs spécifiques pour leurs propriétés.
- L'éditeur de «Protege-OWL» permet à des utilisateurs d'établir des ontologies pour le Web sémantique, en particulier dans la spécification du W3C (OWL).

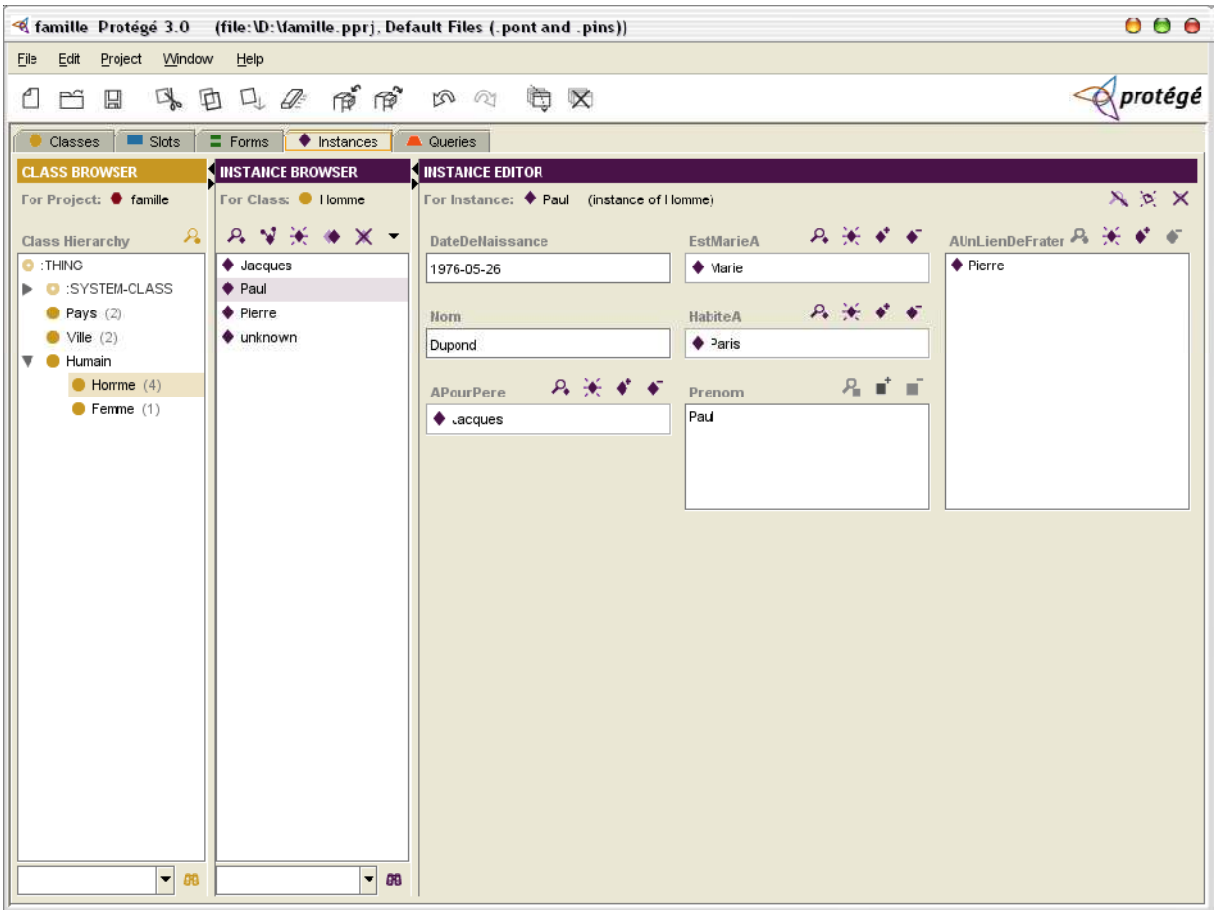
Définition des classes et des propriétés



L'interface de Protégé est assez simple, l'ensemble des fonctionnalités de l'éditeur étant regroupé en cinq onglets. Le premier onglet présente les classes de l'ontologie. Il est possible de créer, modifier, supprimer une classe, et de lui attacher des propriétés. Ces propriétés peuvent elles-même être caractérisées. La capture d'écran de la figure 8 : Capture de l'écran principal de Protégé, onglet «classes » montre l'exemple de l'ontologie développée dans la partie précédente. Cette ontologie, écrite à la main, a été importée dans Protégé à l'aide de son plugin OWL, et peut maintenant être modifiée de façon bien plus confortable.

Gestion des instances de classe et de leurs propriétés

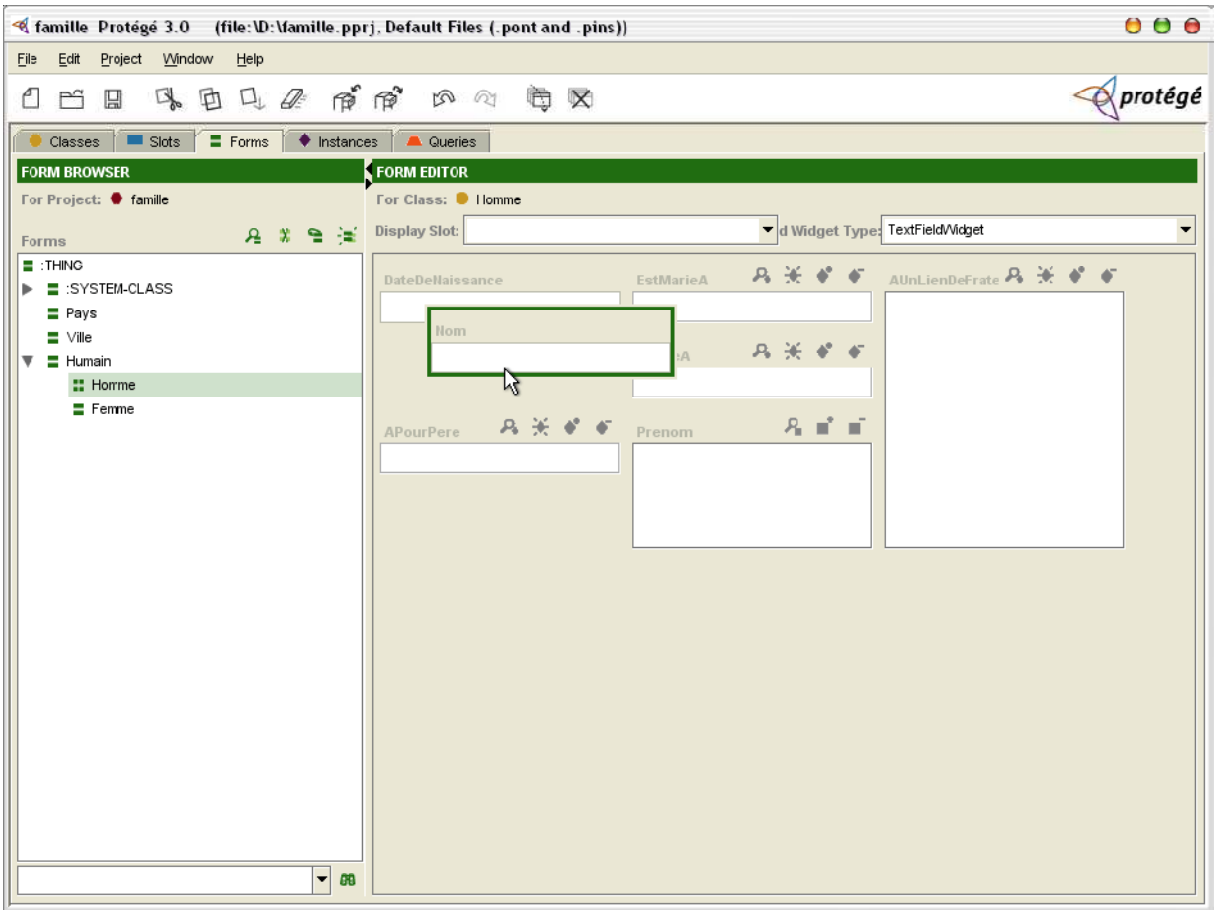
L'onglet « Instances » permet de créer des instances et de leur affecter des propriétés, conformément à la définition des classes et des propriétés effectuée dans l'onglet « Classes » :



Sur l'écran présenté, il est par exemple possible d'éditer les informations concernant l'individu « Paul ». Il est important de comprendre que, dans l'Instance Browser, un individu est désigné par son identifiant (son « rdf:ID »), et non par son prénom. En ce sens, le nommage des individus de l'ontologie écrite en exemple est sans doute un peu ambigu, bien que tout à fait correct. Pour des raisons pédagogiques, il aurait été plus indiqué de nommer ces instances « homme1 », « homme2 », etc.

Flexibilité de l'interface

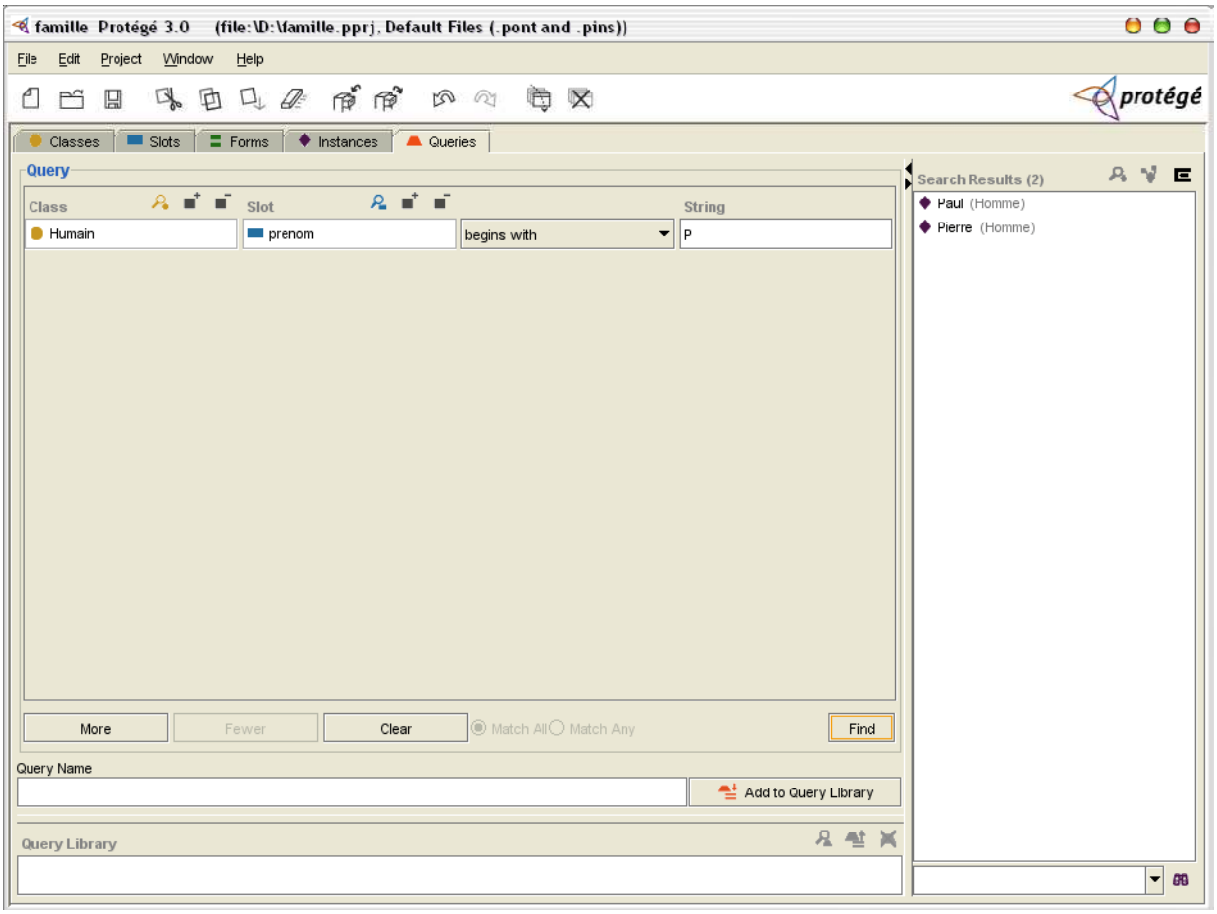
Un des atouts majeurs de Protégé est la flexibilité des formulaires de saisie de l'onglet « Instances ». En effet, ce formulaire dynamique s'adapte en fonction des classes décrites dans le premier onglet du logiciel. Par ailleurs, l'onglet « Forms » permet de modifier l'organisation des formulaires d'instances, par un simple Drag and Drop :



Possibilité d'effectuer des requêtes

Enfin, une fonctionnalité intéressante de Protégé concerne la possibilité d'effectuer des requêtes sur l'ontologie en cours d'édition. La capture d'écran de la figure « Illustration 11 : Gestionnaire de requêtes de Protégé » présente une requête. En l'occurence, il s'agit de chercher toutes les instances de la classe « Humain » dont le « prenom » commence par «P».

Le résultat de cette requête est évidemment l'ensemble {Pierre, Paul}. Le gestionnaire de requêtes de Protégé, bien que pratique, reste cependant limité en fonctionnalités.


famille Protégé 3.0 (file:\D:\famille.pprj, Default Files (.pont and .pins))
File Edit Project Window Help
protégé
Classes Slots Forms Instances Queries
Query
Class
Slot
String
Humain
prenom
begins with
P
Search Results (2)
Paul (Homme)
Pierre (Homme)
More
Fewer
Clear
Match All
Match Any
Find
Query Name
Add to Query Library
Query Library

Conclusion

Le Web semble enfin comprendre son objectif initial : le partage rapide de savoirs précis. Pas uniquement leur présentation anarchique mais, plutôt, leur mise à disposition dans un format non ambigu, compréhensible par tous et extensible. Créés dans un objectif de partage des connaissances en réseau, les dernires langages du Web sémantique que sont RDF et OWL sont en vérité les premières pierres d'une nouvelle forme de Web : le Web sémantique, le Web de la connaissance. Facilitant l'appropriation des savoirs en les libérant de la couche présentationnelle qui les enfermait jusqu'alors, le Web sémantique contribue, en ce sens, à rejoindre les objectifs initiaux de Tim Berners Lee. Malgré leur jeunesse relative, RDF et OWL sont déjà source d'un réel enthousiasme. Naturellement objet de l'attention des universitaires, RDF, OWL, et les autres développements du Web sémantique semblent également convaincre certaines grosses entreprises. L'exemple de Jena, soutenu par Hewlett Packard, est flagrant. D'une manière plus générale, la richesse logicielle qui entoure les deux dernières créations du W3C ne trompe pas : le Web de demain sera sémantique.