ACADEMIE DE MONTPELLIER

UNIVERSITE DE MONTPELLIER

II

- SCIENCES ET TECHNIQUES DU LANGUEDOC -

MEMOIRE DE D.E.A.

Description de méta-données de ressources pédagogiques avec OWL

Description of pedagogical resources metadata with OWL

ANNEXES

Encadré par le Pr Danièle Hérin

Anfana TRAORE

03 septembre 2003

TABLE DES MATIERES

TABLE DES MATIERES 2

ANNEXE A : PRESENTATION DU LANGAGE OWL 5

1 INTERET DE OWL 5
1.1 Pourquoi OWL ? *5*
1.2 Usage des trois sous langages de OWL *6*
2 LE LANGAGE DE DESCRIPTION OWL LITE 8
2.1 Classes *8*
2.1.1 Descriptions de classe 9
2.1.1.1 Restrictions de propriété 9
2.1.1.1.1 Contraintes de valeur 10
2.1.1.1.2 Contraintes de cardinalité 11
2.1.1.2 Intersection 13
2.1.2 Axiomes de classe 13
2.1.2.1 Sous classe 14
2.1.2.2 Classe équivalente 15
2.2 Propriétés *16*
2.2.1 Constructeurs de propriété de RDF-S 17
2.2.1.1 rdfs:subPropertyOf 17
2.2.1.2 rdfs:domain 18
2.2.1.3 rdfs:range 18
2.2.2 Relations à d'autres propriétés 18
2.2.2.1 owl:equivalentProperty 18
2.2.2.2 owl:inverseOf 19
2.2.3 Restrictions globales de cardinalités sur les propriétés 19
2.2.3.1 owl:FunctionalProperty 19
2.2.3.2 owl:InverseFunctionalProperty 19
2.2.4 Caractéristiques logiques des propriétés 20
2.2.4.1 owl:TransitiveProperty 20
2.2.4.2 owl:SymmetricProperty 21
2.3 Individus *21*
2.3.1 Axiomes d'individu ("faits") 21
2.3.2 Indentité d'individu 22
2.3.2.1 owl:sameAs and owl:sameIndividualAs 23
2.3.2.2 owl:differentFrom 23
2.3.2.3 owl:AllDifferent 24
2.4 Types de données *24*
2.5 Informations d'en-tête [Harmelen 2003] *27*
2.5.1 Annotations 27
2.5.2 En-tête d'ontologie 28
2.5.3 Importation d'une ontologie 28
2.5.4 Informations de version 28
3 DESCRIPTION INCREMENTALE DE OWL DL ET OWL FULL 30
3.1 Descriptions de classe *30*
3.1.1 Enumération : oneOf 30
3.1.2 Restrictions de propriété 31
3.1.2.1 Contrainte de valeurs : hasValue 31
3.1.2.2 Contraintes de cardinalité : minCardinality, maxCardinality, cardinality 31
2.3.1.2.3 Combinaisons booléennes : intersectionOf, complementOf, unionOf 32
3.2 Axiomes de classe *34*
3.2.1 Classes complexes 34
3.2.2 owl:disjointWith 35

4 CONCLUSION 36

ANNEXE B : DESCRIPTION OWL DU PROTOTYPE 37

ANNEXE C : DESCRIPTION DES META DONNEES AVEC OWL 54

1 DESCRIPTION LOCALE DES CLASSES 54
1.1 Classe Personnel 54
1.2 Classe Secrétaire 54
1.3 Classe Enseignant 54
1.4 Classe Ouvrage de reference 54
1.5 Classe LigneProgramme 55
1.6 Classe Examen 55
1.7 Classe Exercice 55
1.8 Classe Corrigé 56
1.9 Classe FAQ 56
1.10 Classe SupportCours 56
1.11 Classe Ressource 56
1.12 Classe Contenu 56
1.13 Classe LignePlan 57
1.14 Classe Concept 57
1.15 Classe Université 57
1.16 Classe Discipline 57
1.17 Classe Niveau 57
1.18 Classe Cours 58
2 DESCRIPTION GLOBALE 58

ANNEXE D : OWL PLUGIN 69

1 OWL PLUGIN 69
1.1 Panorama 69
1.2 Bref aperçu : le langage d'ontologie du web OWL dans Protégé 69
1.2.1 Named classes 70
1.2.2 Restrictions 70
1.2.3 Expressions complexes de classe 70
1.3 Conception d'ontologie OWL avec Protégé 71
1.3.1 Création d'une nouvelle ontologie OWL 71
1.3.2 Chargement d'un fichier OWL existant 71
1.3.3 Enregistrement d'une ontologie OWL 72
1.3.4 Travailler avec l'interface utilisateur de Protégé 72
1.3.5 Définition de classes nommées simples 73
1.3.6 Définition d'une taxonomie de classe / hiérarchie d'héritage 74
1.3.7 Définition des propriétés / slots 74
1.3.8 Définition des restrictions 77
1.3.9 Edition des expressions OWL 78
1.3.9.1 Syntaxe d'expression 79
1.3.9.2 Editeur d'expression 80
1.3.10 Edition de classes équivalentes, de super classes et de classes disjointes 82
1.3..11 Edition des instances et leurs formulaires 82
1.3.12 Définition d'une nouvelle méta classe 83
1.3.13 Accès à des classifiers et à d'autres reasoners 87
1.4 Limites du Plugin OWL 88
1.5 Résumé du Mapping entre Protégé et OWL 88
1.5.1 Protégé à OWL 88
1.5.2 OWL à Protégé 89

ANNEXE E : PROTOTYPE 91

1 HIERARCHIE DE CLASSES 91
2 DESCRIPTION DE CLASSES 92
2.1 Contenu *92*
2.2 Examen *92*
2.3 Exercice *92*
2.4 Corrige *93*
2.5 FAQ *93*
2.6 Support_Cours *93*
2.7 Personne *94*
2.8 Auteur *94*
2.9 Enseignant *94*
2.10 Secrétaire *95*
2.11 Cours *95*
2.12 Ligne plan *96*
2.13 Ouvrage de référence *96*
2.14 Ligne Programme *97*
2.15 Concept *97*
2.16 Ressource *98*
2.17 Ontologie globale *98*
3 INSTANCES ET LEURS FORMULAIRES 99
3.1 Examen *99*
3.2 Exercice *99*
3.3 Corrige *100*
3.4 Support de Cours *100*
3.5 Auteur *101*
3.6 Enseignant *101*
3.7 Secrétaire *102*
3.8 Concept *102*
3.9 Ouvrage de référence *103*
3.10 Ontologie Globale *103*
3.11 Cours *104*
4 QUELQUES REQUETES 105
4.1 Query1 : Les ressources qui concernent les concepts qui contiennent "database" *105*
4.2 Query2 : Les parties du plan qui traitent du concept "File and system structure" *106*

BIBLIOGRAPHIE **107**

ANNEXE A : PRESENTATION DU LANGAGE OWL

Cette partie décrit le langage d'ontologie du Web OWL. OWL est destiné à être utilisé quand l'information contenue dans les documents doit être traitée par des applications, par opposition aux situations où le contenu doit seulement être présenté. OWL peut être employé pour représenter explicitement la signification des termes dans les vocabulaires et les relations entre ces termes. Cette représentation des termes et de leurs corrélations s'appelle une ontologie. OWL offre plus de facilités pour exprimer la signification et la sémantique que XML, RDF et RDF-S. OWL va ainsi au-delà de ces langages dans sa capacité de représenter le contenu compréhensible par une machine sur le Web. OWL est une révision du langage d'ontologie du Web DAML+OIL intégrant les leçons apprises de la conception et de l'application de DAML+OIL. [McGuinness 2003]

L'objet de cette partie est de présenter les fonctionnalités offertes par OWL (i.e. ce que l'on peut représenter avec OWL) avec des illustrations a l'appui.

1 Intérêt de OWL

1.1 Pourquoi OWL ?

Le Web sémantique est une vision futuriste du Web dans laquelle l'information est donnée avec une signification explicite afin de faciliter le traitement automatique et l'intégration de l'information disponible sur le Web. Le Web sémantique repose à la fois sur la capacité de XML à définir des schémas tagués adaptés et sur la flexibilité de l'approche RDF à représenter les données. Le premier niveau au-dessus de RDF requis pour le Web sémantique est un langage d'ontologie qui peut décrire formellement la signification de la terminologie utilisée dans les documents Web. Si on veut que les machines accomplissent des taches de raisonnement utiles sur ces documents, le langage doit aller au-delà de la sémantique de base de RDF Schéma.

OWL a été conçu pour satisfaire le besoin d'un langage d'ontologie du Web. OWL fait partie de la pile croissante de recommandations du W3C relative au Web sémantique.

- XML fournit une syntaxe de surface pour les documents structurés, mais n'impose aucune contrainte sémantique sur leur signification.

- XML Schéma est un langage pour restreindre la structure des documents XML.

- RDF est un modèle de données pour les objets (ressources) et les relations entre ces objets. Il fournit une sémantique simple pour le modèle de données. Les modèles de données peuvent être représentés en syntaxe XML.

- RDF Schéma est un vocabulaire pour décrire les propriétés et les classes de ressources RDF, avec une sémantique pour les hiérarchies de généralisation de telles propriétés et classes.

- OWL ajoute plus de vocabulaire pour décrire les propriétés et les classes. On peut citer entre autres : les relations entre classes (par exemple la disjonction), les cardinalités (par exemple exactement un), l'égalité, typage plus riche des propriétés, caractéristiques des propriétés (par exemple la symétrie) et les classes énumérées.

1.2 Usage des trois sous langages de OWL

OWL fournit trois sous langages de plus en plus expressifs conçus pour l'usage des communautés spécifiques des utilisateurs et des développeurs.

OWL Lite convient aux utilisateurs qui ont principalement besoin d'une hiérarchie de classification et de contraintes simples. Par exemple, alors qu'il supporte des contraintes de cardinalité, il autorise seulement des valeurs de cardinalité 0 ou 1. Il devrait être plus simple de fournir un outil d'aide pour OWL Lite que ses parents plus expressifs. OWL Lite fournit un chemin rapide de migration pour les thésaurus et d'autres taxonomies.

Tandis que OWL DL convient aux utilisateurs qui veulent le maximum d'expressivité tout en maintenant la complétude (toutes les conclusions sont calculées) et la décidabilité (tous les calculs s'effectuent dans un temps fini). OWL DL inclut tous les constructeurs du langage OWL, mais ils peuvent être utilisés seulement sous certaines restrictions (par exemple, tant qu'une classe peut être sous classe de plusieurs classes, une classe ne peut pas être une

instance d'une autre classe). OWL DL est appelé ainsi en raison de sa correspondance avec les logiques de description, un champ de la recherche qui a étudié les logiques qui constituent la base formelle de OWL.

OWL Full se prête bien aux utilisateurs qui veulent le maximum d'expressivité et la liberté syntaxique de RDF sans aucune garantie sur la conclusion. Par exemple en OWL Full, une classe peut être traitée simultanément comme une collection d'individus et comme un individu (in its own right). OWL Full permet à une ontologie d'augmenter la signification de son vocabulaire (RDF ou OWL) prédéfini. Il est peu probable que n'importe quel logiciel de raisonnement pourra approuver le raisonnement complet pour chaque caractéristique de OWL Full.

Chacun de ces sous langages est une extension de son prédécesseur, à la fois dans ce qui peut être légalement exprimé et dans ce qui peut être bien conclu.

- Toute ontologie légale OWL Lite est une ontologie légale OWL DL.
- Toute ontologie légale OWL DL est une ontologie légale OWL Full.
- Toute conclusion valide OWL Lite est une conclusion valide OWL DL.
- Toute conclusion valide OWL DL est une conclusion valide OWL Full.

Les développeurs d'ontologie adoptant OWL devraient choisir lequel des trois sous langages correspond aux mieux à leurs besoins. Le choix entre OWL Lite et OWL DL dépend de l'ampleur à laquelle les utilisateurs ont besoin de constructeurs plus expressifs fournis par OWL DL et OWL Full. Le choix entre OWL DL et OWL Full dépend principalement de l'ampleur à laquelle les utilisateurs ont besoin des facilités de méta modélisation de RDF Schéma (par exemple définissant des classes de classes, ou attachant des propriétés aux classes). En utilisant OWL Full par rapport à OWL DL, le support de raisonnement est moins prévisible puisque les implémentations complètes de OWL Full n'existent pas actuellement.

OWL Full peut être vu comme une extension de RDF, alors que OWL Lite et OWL DL peuvent être vu comme des extensions d'une vue restreinte de RDF. **Tout document OWL (Lite, DL, Full) est un document RDF**, et tout document RDF est un document OWL Full, mais seulement certains documents RDF seront un document légal OWL Lite ou OWL DL.

La suite de ce chapitre décrit d'abord les caractéristiques de OWL Lite[1], suivi d'une description des caractéristiques qui sont ajoutées dans OWL DL et OWL Full. OWL DL et OWL Full contiennent les mêmes caractéristiques mais OWL Full est plus souple sur la façon dont ces caractéristiques peuvent être combinées.

2 Le langage de description OWL Lite

Les préfixes `rdf:` ou `rdfs:` sont employés lorsque les termes sont déjà présents dans RDF ou RDF Schéma. Sinon les termes sont introduits par OWL. Ainsi le terme `rdfs:subPropertyOf` indique que `subPropertyOf` est déjà dans le vocabulaire de RDF-S. En outre, le terme `class`, plus précisément énoncé comme `owl:class` est un terme introduit par OWL.

2.1 Classes

Les classes fournissent un mécanisme d'abstraction pour regrouper des ressources ayant des caractéristiques communes. Comme les classes RDF, toute classe OWL est associée à un ensemble d'individus appelé l'extension de la classe. Les individus dans l'extension de la classe sont appelés les instances de la classe. Deux classes peuvent avoir la même extension mais être différentes.

Les classes OWL sont décrites à travers les descriptions de classe, qui peuvent être combinées en axiomes de classe. On décrira d'abord les descriptions de classe, puis les axiomes de classe.

[1] OWL Lite uses only some of the OWL language features and has has more limitations on the use of the features than OWL DL or OWL Full. ... are also only allowed between named classes, ... Similarly, restrictions in OWL Lite classes can only be defined in terms of named superclasses (superclasses cannot be arbitrary expressions), and only certain kinds of class restrictions can be used. Equivalence between classes and subclass relationships between classes are also only allowed between named classes, and not between arbitrary class expressions. Similarly, restrictions in OWL Lite use only named classes. OWL Lite also has a limited notion of cardinality - the only cardinalities allowed to be explicitly stated are 0 or 1.

2.1.1 Descriptions de classe

OWL Lite distingue trois types de descriptions de classe :

1. un identificateur de classe (une référence URI)
2. une restriction de propriété
3. l'intersection de descriptions de classe

Le premier type décrit une classe à travers un nom. Les autres types de descriptions de classe définissent une classe anonyme, respectivement une classe dont tous les individus satisfont une restriction particulière de propriété, ou une classe qui satisfait des combinaisons booléennes de descriptions de classe. L'intersection peut être vu comme l'opérateur logique AND.

Le type de description 1 est syntaxiquement représenté comme une instance nommée de `owl:Class`, une sous classe de `rdfs:Class`.

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

Ceci signifie le triplet `"Human rdf:type owl:Class"`.

Dans OWL Lite, `owl:Class` doit être utilisée dans toutes les descriptions de classes. OWL Lite possède un identificateur de classe prédéfinie intitulée la classe `owl:Thing`. L'extension de la classe `owl:Thing` est l'ensemble de tous les individus dans le domaine du discours. Par conséquent, toute classe OWL Lite est une sous classe de `owl:Thing`.

2.1.1.1 Restrictions de propriété

Une restriction de propriété est une sorte de description de classe. Elle définit une classe anonyme, une classe dont tous les individus satisfont la restriction. OWL distingue deux sortes de restrictions : les contraintes de valeur et les contraintes de cardinalité.

Une contrainte de valeur met des contraintes sur le range de valeur d'une propriété. Par exemple, on pourrait référencer des individus dont la valeur de la propriété `adjacentTo` doit être une `Region`, et l'employer dans un axiome, peut être même un axiome de `Region`.

Notons que ceci est très différent de `rdfs:range` qui est appliquée dans toutes les situations dans lesquelles la propriété est utilisée.

Une contrainte de cardinalité met des contraintes sur le nombre de valeurs qu'une propriété peut prendre, dans le contexte de cette description particulière de classe. Par exemple, on peut exprimer que la propriété `hasPlayer` d'une équipe de tennis à une seule valeur.

OWL supporte également un ensemble limité de constructeurs pour définir la cardinalité globale de propriété, à savoir les définitions `owl:FunctionalProperty` et `owl:InverseFunctionalProperty`.

Les restrictions de propriété ont la forme générale suivante :

```
<owl:Restriction>
  <owl:onProperty rdf:resource="(some property)" />
  (précisément une contrainte de valeur ou de cardinalité, voir ci-dessous)
</owl:Restriction>
```

La classe `owl:Restriction` est définit comme une sous classe de `owl:Class`. Les restrictions de propriété peuvent être appliquées à la fois aux propriétés de type de donnée (propriété dont la valeur de range est une donnée littérale) et propriétés d'objet (propriétés dont la valeur du range est un individu).

2.1.1.1.1 Contraintes de valeur

`owl:allValuesFrom`

La contrainte de valeur `owl:allValuesFrom` est une propriété qui lie une classe de restriction à une classe de description ou à un range de donnée. Une restriction contenant une contrainte `owl:allValuesFrom` définit une classe des individus X pour lesquels si le couple (X,Y) est une instance de P (la propriété concernée), alors Y doit être une instance de la description de classe ou une valeur du range de donnée.

Cet exemple décrit tous les individus d'une classe anonyme OWL pour lesquels la propriété `hasParent` a seulement des valeurs de range de la classe `Human`. Notons que cette description ne déclare pas que la propriété a toujours le range de cette classe, mais juste que cela vaut pour les individus de la classe anonyme de restriction.

Dans OWL Lite, la valeur de range de `owl:allValuesFrom` doit être un identificateur de classe (class identifier).

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:allValuesFrom rdf:resource="#Human"  />
</owl:Restriction>
```

Une contrainte `owl:allValuesFrom` est analogue au quantificateur universel, pour chaque instance de la classe ou type de donnée qui est définie, toute valeur de range doit satisfaire la contrainte.

`owl:someValuesFrom`

Une restriction contenant une contrainte `owl:someValuesFrom` décrit tous les individus d'une classe pour lesquels au moins une valeur de la propriété concernée est une instance de la description de la classe ou une valeur de donnée dans le range de donnée. En d'autres termes, elle définit une classe des individus X pour lesquels il y a au moins un Y (une instance de la classe de description ou valeur du range de donnée) telle que le couple (X,Y) est une instance de P. Cela n'exclut pas qu'il ait d'autres instances (X,Y') de P pour lequel Y' n'appartient pas à la description de classe ou à la plage de données.

L'exemple suivant définit une classe d'individus ayant au moins un parent qui est physicien.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:someValuesFrom rdf:resource="#Physician" />
</owl:Restriction>
```

La contrainte `owl:someValuesFrom` est analogue au quantificateur existentiel de la logique des prédicats. Pour chaque instance de la classe qui est définie, il existe au moins une valeur pour P qui satisfait la contrainte. Dans OWL Lite, la valeur du range de `owl:someValuesFrom` doit être un identificateur de classe.

2.1.1.1.2 Contraintes de cardinalité

OWL Lite inclut une forme limitée de restrictions de cardinalité. **Les restrictions de cardinalité de OWL (et OWL Lite) sont des restrictions locales** puisqu'elles sont énoncées sur des propriétés en ce qui concerne une classe particulière. C'est-à-dire les restrictions contraignent la cardinalité de cette propriété sur les instances de cette classe. Les

restrictions de cardinalité de OWL Lite sont limitées parce qu'elles n'autorisent que des cardinalités de valeur 0 ou 1. Elles n'autorisent pas de cardinalité de valeurs arbitraires comme OWL (DL et Full).

owl:minCardinality

Si une cardinalité minimale de 1 est énoncée sur une propriété par rapport à une classe, alors toute instance de cette classe sera reliée par cette propriété à au moins un individu. Cette restriction exprime d'une autre façon que la propriété doit avoir obligatoirement une valeur pour toutes les instances de la classe.

L'exemple suivant décrit une classe des individus ayant au moins un parent.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1
   </owl:minCardinality>
</owl:Restriction>
```

owl:maxCardinality

Si une propriété à une cardinalité maximale de 1 en ce qui concerne une classe, alors toute instance de cette classe sera reliée par cette propriété à tout au plus un individu. Une restriction de cardinalité maximale s'appelle parfois une propriété fonctionnelle ou unique.

Une cardinalité maximale de 1 ne permet pas que conclure sur la cardinalité minimale. On peut déclarer que certaines classes n'ont pas de valeurs pour une propriété particulière.

L'exemple suivant décrit une classe des individus ayant tout au plus un parent.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:maxCardinality rdf:datatype="&xsd;nonNegativeInteger">1
   </owl:maxCardinality>
</owl:Restriction>
```

owl:Cardinality

La cardinalité convient quand il est utile de déclarer qu'une propriété sur une classe à une cardinalité minimale et maximale de 1, ou une cardinalité minimale et maximale de 0.

L'exemple suivant décrit une classe des individus ayant exactement une mère maternelle.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasBirthMother" />
  <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardinality>
</owl:Restriction>
```

2.1.1.2 Intersection

`owl:intersectionOf` est un constructeur de classe employé dans la logique de description. Il peut être vu comme l'opérateur AND sur les classes. La propriété `owl:intersectionOf` lie une classe à une liste de descriptions de classe. Cette propriété décrit une clase dont l'extension contient précisément les individus qui sont membres de l'extension de la classe de toutes les descriptions de classe dans la liste de range.

L'exemple suivant décrit la classe `WhiteBurgundy` comme exactement l'intersection de vins blancs et de Bourgogne.

```
<owl:Class rdf:ID="WhiteBurgundy">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Burgundy" />
    <owl:Class rdf:about="#WhiteWine" />
  </owl:intersectionOf>
</owl:Class>
```

OWL Lite[2] autorise les intersections de classes nommées et de restrictions.

2.1.2 Axiomes de classe

La forme la plus simple d'un axiome de classe est une description de classe de type 1, elle énonce juste l'existence d'une classe, en utilisant `owl:Class` avec un identificateur de classe.

Par exemple, l'axiome de classe suivant déclare la référence de l'URI comme étant une classe OWL.

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

Cet exemple ne donne pas beaucoup d'informations sur la classe `Human`. Les axiomes de classe contiennent typiquement les composants additionnels qui énoncent les caractéristiques nécessaires et/ou suffisantes d'une classe. OWL Lite contient deux constructeurs de langage pour combiner les descriptions dans les axiomes de classe :

[2] OWL lite allows intersection of named classes and restrictions.

- rdfs:subClassOf exprime que l'extension d'une description de classe est un sous ensemble de l'extension d'une autre description de classe.

- rdfs:equivalentClass exprime qu'une description de classe a exactement la même extension de classe qu'une autre description de classe.

Syntaxiquement, ces deux constructeurs de langage sont des propriétés ayant une description de classe comme à la fois domaine et range.

2.1.2.1 Sous classe

Schéma d'axiome : description de classe `rfs:subClassOf` description de classe.

Cet axiome de classe déclare une relation de sous classe entre deux classes OWL qui sont décrites à travers leurs noms (`Opera` et `MusicalWork`). Les relations de sous classe fournissent des conditions nécessaires pour appartenir à une classe. Dans ce cas-ci, pour être un opéra, l'individu doit être également un œuvre musicale.

Notons que dans OWL Lite, la valeur du domaine de `rdfs:subClassOf` doit être un identificateur de classe. La valeur du range doit être un identificateur de classe ou une restriction de propriété.

```
<owl:Class rdf:ID="Opera">
  <rdfs:subClassOf rdf:resource="#MusicalWork" />
</owl:Class>
```

Les axiomes de classe peuvent également utiliser d'autres types de descriptions[3] de classes telles que les restrictions de propriété et les intersections. Pour toute classe, il peut y avoir un certain nombre d'axiomes `subClassOf`. Par exemple on peut ajouter l'axiome suivant à propos de la classe `Opera`.

```
<owl:Class rdf:about="#Opera">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasLibrettist" />
      <owl:minCardinality rdf:datatype="&xsd;nonNegative
       Integer">1</owl:minCardinality>
```

[3] Aux restrictions de propriété et les intersections, s'ajoutent les énumérations, les unions et les compléments. Mais ces derniers ne font partie de OWL Lite.

```
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Cet axiome de classe contient une restriction de propriété. L'exemple énonce que `Opera` est une sous classe d'une classe anonyme OWL (rappelons que **`owl:Restriction` est une sous classe de `owl:Class`**) qui a comme extension l'ensemble de tous les individus ayant au moins une valeur pour la propriété `hasLibrettist`.

2.1.2.2 Classe équivalente

Schéma d'axiome : description de classe `owl:equivalentClass` description de classe.

Le constructeur `owl:equivalentClass` est une propriété qui lie une description de classe à une autre description de classe. Ce schéma d'axiome exprime que deux descriptions de classe ont la même extension (c'est-à-dire que les deux extensions de classe contiennent exactement le même ensemble d'individus). Il permet de créer des classes synonymes.

Dans sa forme la plus simple, l'axiome `equivalentClass` énonce l'équivalence (en termes d'extension de classe) de deux classes nommées. Un exemple :

```
<owl:Class rdf:about="#US_President">
  <equivalentClass rdf:resource="#PrincipalResidentOfWhiteHouse"/>
</owl:Class>
```

L'usage de `owl:equivalentClass` n'implique pas l'égalité de classe. L'égalité de classe signifie que les classes ont la même signification intentionnelle (le même concept). Dans l'exemple ci-dessus, le concept de "Président des Etats Unis"est relié à, mais pas égale au concept de "RésidentPrincipalDeLaMaisonBlache". La vraie égalité de classe[4] ne peut seulement être exprimée qu'avec le constructeur `owl:sameAs` (ou son synonyme `owl:sameIndividualAs`).

Dans OWL Lite, seuls les identificateurs de classe et les restrictions de propriété sont autorisés comme domaine et valeur de range.

[4] Comme cela requiert de traiter les classes comme des individus, l'égalité de classe ne peut seulement être exprimée qu'en OWL Full.

Si on veut mettre à jour un axiome de la forme A `subClassOf` B pour qu'il devienne A `equivalentClassOf` B, alors on ajoute un second axiome `subClassOf` de la forme B `subClassOf` A, qui par définition fait que les deux extensions de classes sont équivalentes. Comme OWL peut être utilisé dans un environnement distribué alors ceci est un puissant moyen.

2.2 Propriétés

OWL distingue deux types de propriétés :

- Les propriétés d'objet ont un range de valeur d'individus de classe et lie ainsi les individus aux individus.

- Les propriétés de type de donnée ont un range de valeur de données, et lie ainsi les individus aux valeurs de données.

Une propriété d'objet est définit à l'aide d'une classe `owl:ObjectProperty`. Une propriété de type de donnée est définit à travers une classe `owl:DatatypeProperty`. `owl:ObjectProperty` et `owl:DatatypeProperty` sont sous classes de la classe RDF `rdf:Property`.

Notons que dans OWL Full, les propriétés d'objet et les propriétés de type de donnée ne sont disjointes. Parce que les valeurs de données peuvent être traitées comme des individus, les propriétés de type donnée sont effectivement sous classes des propriétés d'objet. Dans OWL Full `owl:ObjectProperty` est équivalent à `rdf:Property`. En pratique ceci a principalement des conséquences sur l'usage de `owl:InverseFunctionalProperty`.

Un axiome de propriété définit les caractéristiques d'une propriété. Dans sa forme la plus simple, un axiome de propriété définit tout juste l'existence d'une propriété.

L'exemple suivant définit une propriété dont le range de valeurs doit être des individus.

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

Souvent, les axiomes de propriété définissent les caractéristiques additionnelles de propriétés. OWL supporte les constructeurs suivants :

- Les constructeurs RDF-S : `rdfs:subPropertyOf`, `rdfs:domain`, et `rdfs:range`
- Les relations à d'autres propriétés : `owl:equivalentProperty` et `owl:inverseOf`
- Les contraintes globales de cardinalité : `owl:FunctionalProperty` et `owl:InverseFunctionalProperty`
- Les caractéristiques logiques de propriété : `owl:SymmetricProperty` et `owl:TransitiveProperty`

Dans cette section, on utilise le terme extension de propriété de façon analogue à extension de classe. L'extension d'une propriété est l'ensemble des instances associé à cette propriété. Les instances de propriétés ne sont pas des éléments simples, mais des couples (sujet,objet). En termes de base de données relationnelle, les instances de propriété sont appelées tuples d'une relation binaire (la propriété).

2.2.1 Constructeurs de propriété de RDF-S

2.2.1.1 rdfs:subPropertyOf

Un axiome `rdfs:subPropertyOf` définit qu'une propriété est une sous propriété d'une autre propriété. Formellement, cet axiome signifie que si P1 est une sous propriété de P2, alors l'extension de la propriété P1 (un ensemble de couples) doit être un sous ensemble de l'extension de la propriété de P2 (un ensemble de couples).

L'exemple suivant énonce que toutes les instances (couples) contenues dans l'extension de la propriété `"hasMother"` sont également membres de l'extension de la propriété `"hasParent"`.

```
<owl:ObjectProperty rdf:ID="hasMother">
  <rdfs:subPropertyOf rdf:resource="#hasParent"/>
</owl:ObjectProperty>
```

Les axiomes de sous propriété peuvent être appliqués à la fois aux propriétés de type de donnée et aux propriétés d'objet.

Notons que dans OWL DL, le domaine et la valeur du range d'un axiome de sous propriété doit être à la fois des propriétés de type de donnée ou à la fois des propriétés d'objet.

2.2.1.2 rdfs:domain

`rdfs:domain` est une propriété qui lie une propriété (certaine instance de la classe `rdf:Property`) à une description de classe. Le domaine d'une propriété limite les individus auxquels la propriété peut être appliquée. Si une propriété relie un individu à un autre individu, et la propriété a une classe comme un de ses domaines alors l'individu doit appartenir à la classe.

Les restrictions `rdfs:domain` sont gloables, signifiant qu'elles ne peuvent pas être employées pour un individu pour lequel la classe n'est pas explicitement incluse dans la restriction de domaine. Dans OWL Lite, la plage de valeurs d'un axiome `rdfs:domain` doit être une identificateur de classe.

```
<owl:ObjectProperty rdf:ID="hasChild">
  <rdfs:domain rdf:resource="#Human"/>
  <rdfs:range rdf:resource="#Human"/>
</owl:ObjectProperty>
```

2.2.1.3 rdfs:range

La plage de valeurs d'une propriété limite les individus que la propriété peut prendre comme valeur. Si une propriété relie un individu à un autre individu, et que la propriété à une classe comme plage de valeurs alors l'autre individu doit appartenir à la classe de la plage de valeurs. Dans OWL Lite, la valeur du range d'un axiome `rdfs:range` doit être un identificateur de classe.

2.2.2 Relations à d'autres propriétés

2.2.2.1 owl:equivalentProperty

Le constructeur `owl:equivalentProperty` exprime que deux propriétés ont la même extension de propriété. `owl:equivalentProperty` est une propriété avec comme domaine et range une instance de `rdf:Property`. L'équivalence de propriété est différente de l'égalité de propriété. Les propriétés équivalentes ont les mêmes valeurs (c'est-à-dire, la même extension de propriété), mais peuvent avoir différente signification intentionnelle. L'égalité de propriété s'exprime avec le constructeur `owl:sameAs`.

```
<owl:ObjectProperty rdf:ID="hasLeader">
  <owl:equivalentPropertyOf rdf:resource="#hasHead"/>
</owl:ObjectProperty>
```

2.2.2.2 owl:inverseOf

`owl:inverseOf` est une propriété qui prend les instances de `owl:ObjectProperty` comme domaine et valeur de range. Si la propriété P1 est l'inverse de la propriété P2, alors si X est relié à Y par la propriété P1, alors Y est relié à X par la propriété P2.

```
<owl:ObjectProperty rdf:ID="hasChild">
  <owl:inverseOf rdf:resource="#hasParent"/>
</owl:ObjectProperty>
```

2.2.3 Restrictions globales de cardinalités sur les propriétés

2.2.3.1 owl:FunctionalProperty

Les propriétés peuvent avoir une valeur unique. Si une propriété est une FunctionalProperty, alors elle n'a pas plus d'une valeur pour chaque individu (elle peut ne pas avoir de valeur pour un individu). FunctionalProperty exprime sous forme contractée que la cardinalité minimale d'une propriété est 0 et que sa cardinalité maximale est 1.

```
<owl:ObjectProperty rdf:ID="husband">
  <rdf:type    rdf:resource="&owl;FunctionalProperty" />
  <rdfs:domain rdf:resource="#Woman" />
  <rdfs:range  rdf:resource="#Man" />
</owl:ObjectProperty>
```

L'exemple suivant est équivalent à celui ci-dessus :

```
<owl:ObjectProperty rdf:ID="husband">
  <rdfs:domain rdf:resource="#Woman" />
  <rdfs:range  rdf:resource="#Man" />
</owl:ObjectProperty>

<owl:FunctionalProperty rdf:about="#husband" />
```

2.2.3.2 owl:InverseFunctionalProperty

Si une propriété est `InverseFunctionalProperty` alors l'inverse de la propriété est fonctionnel. Ainsi l'inverse a tout au plus une valeur pour chaque individu. Cette caractéristique a été mentionnée comme une **propriété non ambiguë**.

L'exemple suivant énonce que pour chaque valeur de range (Human) de la propriété biologicalMotherOf, on doit identifier de façon unique la valeur du domaine (Woman). Les propriétés inverse-fonctionnelles ressemblent à la notion de clé en base de données.

```
<owl:InverseFunctionalProperty rdf:ID="biologicalMotherOf">
  <rdfs:domain rdf:resource="#Woman"/>
  <rdfs:range rdf:resource="#Human"/>
</owl:InverseFunctionalProperty>
```

A la différence des propriétés fonctionnelles, les propriétés inverse-fonctionnelles n'ont pas besoin d'ajouter l'axiome de propriété de type de donnée ou de propriété d'objet. Les propriétés inverse-fonctionnelles sont par définition des propriétés d'objet.

Notons que FunctionalProperty et InverseFunctionalProperty spécifient des contraintes globales de cardinalité.

2.2.4 Caractéristiques logiques des propriétés

2.2.4.1 owl:TransitiveProperty

Si une propriété est transitive, alors si le couple **(X,Y)** est une instance de la propriété transitive **P**, et le couple **(Y,Z)** est une instance de **P**, alors le couple **(X,Z)** est aussi une instance de **P**. Si la propriété **Ancêtre** est transitive, et si **Adam** est **Ancêtre** de **David** (i.e. **(Adam,David)** est une instance de la propriété **Ancêtre**) et **David** est **Ancêtre** de **Salamon** ((i.e. **(David,Salomon)** est une instance de la propriété **Ancêtre**), alors on peut déduire que **Adam** est **Ancêtre** de **Salomon** (i.e. **(Adam,Salomon)** est une instance de la propriété **Ancêtre**).

OWL (Lite et DL) imposent que les propriétés transitives (et leurs super propriétés) ne peuvent pas avoir une restriction maxCardinality 1. Sans cette condition OWL (Lite et DL) deviendraient des langages indécidables.

```
<owl:TransitiveProperty rdf:ID="hasAncestor">
  <rdfs:domain rdf:resource="#Human"/>
  <rdfs:range  rdf:resource="#Human"/>
</owl:TransitiveProperty>
```

owl:TransitiveProperty est une sous classe de owl:ObjectProperty. L'exemple suivant est une variante équivalente à l'exemple ci-dessus :

```
<owl:ObjectProperty rdf:ID="hasAncestor">
  <rdf:type rdf:resource="&owl;TransitiveProperty"/>
  <rdfs:domain rdf:resource="#Human"/>
  <rdfs:range  rdf:resource="#Human"/>
</owl:ObjectProperty>
```

OWL DL exige que pour une propriété transitive, aucune contrainte locale ou globale de cardinalité ne doit être déclarée sur la propriété elle-même ou ses sous propriétés, ni sur l'inverse de la propriété ou de ses sous propriétés.

2.2.4.2 owl:SymmetricProperty

Si une propriété P est symétrique, alors si le couple **(X,Y)** est une instance de **P**, alors le couple **(Y,X)** est aussi une instance de **P**. Par exemple **Ami** est une propriété symétrique. Les propriétés symétriques ne peuvent pas avoir des domaines et des ranges arbitraires. Elles doivent avoir des domaines et des ranges identiques pour avoir un sens. `owl:SymmetricProperty` est une sous classe de `owl:ObjectProperty`.

```
<owl:SymmetricProperty rdf:ID="friendOf">
  <rdfs:domain rdf:resource="#Human"/>
  <rdfs:range  rdf:resource="#Human"/>
</owl:SymmetricProperty>
```

2.3 Individus

2.3.1 Axiomes d'individu ("faits")

Les axiomes d'individu (appelés également faits) sont énoncés à propos des individus, indiquant l'adhésion à une classe et énoncés à propos de propriétés appropriées. L'exemple suivant représente une instance de la classe `Opera`.

```
<Opera rdf:ID="Tosca">
  <hasComposer rdf:resource="#Giacomo_Puccini"/>
  <hasLibrettist rdf:resource="#Victorien_Sardou"/>
  <premiereDate rdf:datatype="&xsd;date">1900-01-14</premiereDate>
  <premierePlace rdf:resource="#Roma"/>
  <numberOfActs rdf:datatype="&xsd;positiveInteger">3</numberOfActs>
</Opera>
```

Cet exemple inclut un nombre de faits à propos de l'opéra `Tosca`, composé par `Giacomo Puccini`. L'opéra a un auteur de libretto. La propriété `premiereDate` lie un opéra à un type littéral avec comme type de donnée, le type de donnée XML-S `date`.

Les axiomes d'individus ne portent pas nécessairement sur des individus nommées. Ils peuvent également faire référence à des individus anonymes. Considérons le morceau d'exemple RDF/XML ci-dessous. Il définit un certain nombre de faits sur une instance anonyme de la classe `Measurement`.

```
<Measurement>
  <observedSubject rdf:resource="#JaneDoe"/>
  <observedPhenomenon rdf:resource="#Weight"/>
  <observedValue>
    <Quantity>
      <quantityValue rdf:datatype="&xsd;float">59.5</quantityValue>
      <quantityUnit rdf:resource="#Kilogram"/>
    </Quantity>
  </observedValue>
  <timeStamp rdf:datatype="&xsd;dateTime">2003-01-24T09:00:08+01:00</timeStamp>
</Measurement>
```

2.3.2 Indentité d'individu

Plusieurs langages font l'hypothèse sur l'unicité de noms : différents noms font référence à différentes choses dans le monde. Sur le Web, une telle hypothèse n'est pas possible. Par exemple, la même personne peut être référencée de plusieurs manières différentes (i.e. avec différentes références URI). Pour cette raison, OWL ne fait pas cette hypothèse. A moins qu'un énoncé explicite que deux références se rapportent à un même individu ou à des individus différents, les outils de OWL doivent en principe supposer que les deux cas sont possibles. OWL fournit trois constructeurs concernant l'identité des individus :

- `owl:sameAs` est utilisé pour énoncer que deux références se rapportent à un même individu. Le constructeur `owl:sameIndividualAs` est un synonyme de `owl:sameAs`.

- `owl:differentFrom` est utilisé pour énoncer que deux références se rapportent à des individus différents.

- `owl:AllDifferent` est utilisé pour énoncer qu'une liste d'individus sont tous différents.

2.3.2.1 owl:sameAs and owl:sameIndividualAs

`owl:sameAs` a comme domaine et range un individu. OWL supporte `owl:sameIndividualAs` comme synonyme avec exactement la même signification. Par exemple, on peut dire les deux références URI suivantes se rapportent à la même personne.

```
<owl:Human rdf:about="#William_Jefferson_Clinton">
  <owl:sameAs rdf:resource="#BillClinton"/>
</owl:Human>
```

`owl:sameAs` est souvent employé pour faire des mappings entre ontologies. Il est irréaliste de supposer que tout le monde utilisera le même nom pour référencer les individus. Cela requiert une grande conception qui est contraire à l'esprit du Web.

Dans OWL Full où une classe peut être traitée comme instances de (méta)classes, on peut utiliser le constructeur `owl:sameAs` pour définir l'égalité de classe, indiquant ainsi que deux concepts ont la même signification intentionnelle. Un exemple :

```
<owl:Class rdf:ID="FootballTeam">
  <owl:sameAs rdf:resource="http://sports.org/US#SoccerTeam"/>
</owl:Class>
```

Notons la différence avec l'exemple suivant qui énonce que les deux classes ont la même extension de classe mais ne sont pas nécessairement les mêmes concepts.

```
<footballTeam owl:equivalentClass us:soccerTeam />
```

2.3.2.2 owl:differentFrom

Une propriété `owl:differentFrom` lie un individu à un autre individu. Elle indique que deux références URI se rapportent à des individus différents. Un exemple :

```
<Opera rdf:ID="Don_Giovanni"/>

<Opera rdf:ID="Nozze_di_Figaro">
  <owl:differentFrom rdf:resource="#Don_Giovanni"/>
</Opera>

<Opera rdf:ID="Cosi_fan_tutte">
  <owl:differentFrom rdf:resource="#Don_Giovanni"/>
  <owl:differentFrom rdf:resource="#Nozze_di_Figaro"/>
</Opera>
```

Cet exemple énonce que les trois opéras sont tous différents.

2.3.2.3 owl:AllDifferent

`owl:AllDifferent` est un constructeur spécial de `owl:Class` pour lequel la propriété `owl:distinctMembers` est définie. Cette propriété lie une instance de `owl:AllDifferent` à une liste d'individu. `owl:AllDifferent` est tout particulièrement utile lorsqu'il y a des ensembles d'objets distincts, et lorsque les modélisateurs veulent imposer l'unicité du nom dans ces ensembles d'objets.

```
<owl:AllDifferent>
  <owl:distinctMembers rdf:parseType="Collection">
    <Opera rdf:about="#Don_Giovanni"/>
    <Opera rdf:about="#Nozze_di_Figaro"/>
    <Opera rdf:about="#Cosi_fan_tutte"/>
    <Opera rdf:about="#Tosca"/>
    <Opera rdf:about="#Turandot"/>
    <Opera rdf:about="#Salome"/>
  </owl:distinctMembers>
</owl:AllDifferent>
```

Cet exemple exprime que les six références URI se rapportent à différents opéras.

`owl:distinctMembers` est un constructeur syntaxique spéciale supplémentaire. Il doit être employé avec `owl:AllDifferent` comme valeur de domaine.

2.4 Types de données[5]

OWL utilise les mécanismes de RDF pour les valeurs de données [Bechofer 2003]. Les valeurs de données sont des instances de la classe RDF-Schéma `rdfs:Literal`. Les littéraux peuvent être typés ou non. Les types de données sont des instances de la classe `rdfs:Datatype`. Les types de données XML Schéma recommandés pour les ontologies de OWL sont les suivants :

- Le type de données primitif `xsd:string` et ses dérivés suivants : `xsd:normalizedString`, `xsd:token`, `xsd:language`, `xsd:NMTOKEN`, `xsd:Name`, et `xsd:NCName`.

- Le type de donnée primitif `xsd:boolean`.

[5] **Les types de données énumérés ne font partie de OWL Lite.**

- Le type de donnée numérique primitif `xsd:decimal`, `xsd:float`, et `xsd:double`, et les dérivés du type `xsd:decimal` (`xsd:integer`, `xsd:positiveInteger`, `xsd:nonPositiveInteger`, `xsd:negativeInteger`, `xsd:nonNegativeInteger`, `xsd:long`, `xsd:int`, `xsd:short`, `xsd:byte`, `xsd:unsignedLong`, `xsd:unsignedInt`, `xsd:unsignedShort`, `xsd:unsignedByte`).

- Les types de données primitifs relatifs au temps : `xsd:dateTime`, `xsd:time`, `xsd:date`, `xsd:gYearMonth`, `xsd:gYear`, `xsd:gMonthDay`, `xsd:gDay`, et `xsd:gMonth`.

- Les types de données primitifs `xsd:hexBinary`, `xsd:base64Binary`, et `xsd:anyURI`.

Quand on utilise les types de données, même si une propriété est définie avec un range d'un certain type, RDF/XML exige toujours que le type de données soit indiqué chaque fois que la propriété est employée. Un exemple pourrait être la déclaration d'une propriété que nous avons employé dans l'exemple de la Mesure.

```
<owl:DatatypeProperty rdf:about="#timeStamp">
  <rdfs:domain rdf:resource="#Measurement"/>
  <rdfs:range rdf:resource="&xsd;dateTime"/>
</owl:DatatypeProperty>

<Measurement>
  <timeStamp rdf:datatype="&xsd;dateTime">2003-01-24T09:00:08+01:00</timeStamp>
</Measurement>
```

En plus des types de données de RDF, OWL fournit un constructeur additionnel pour définir un range de valeurs intitulé type de donnée énuméré. Ce format de type de donnée se sert du constructeur `owl:oneOf` qui est également utilisé pour décrire une classe énumérée. Dans le cas d'un type de donnée énuméré, la valeur du domaine de `owl:oneOf` est un nœud vide de classe `owl:DataRange` et la valeur du range est une liste de littéraux. Malheureusement, on ne peut utiliser l'idiome `rdf:parseType="Collection"` pour spécifier la liste de littéraux, parce que RDF exige que la collection soit une liste de nœuds éléments RDF. Par conséquent, on doit spécifier la liste de valeurs de données avec les constructeurs de base de liste `rdf:first`, `rdf:rest` et `rdf:nil`.

L'exemple ci-dessous indique que le range de la propriété tennisGameScore est une liste de valeurs entières {0, 15, 30, 40}.

```
<owl:DatatypeProperty rdf:ID="tennisGameScore">
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf>
        <rdf:List>
           <rdf:first rdf:datatype="&xsd;integer">0</rdf:first>
           <rdf:rest>
             <rdf:List>
               <rdf:first rdf:datatype="&xsd;integer">15</rdf:first>
               <rdf:rest>
                 <rdf:List>
                   <rdf:first rdf:datatype="&xsd;integer">30</rdf:first>
                   <rdf:rest>
                     <rdf:List>
                       <rdf:first rdf:datatype="&xsd;integer">40</rdf:first>
                       <rdf:rest rdf:resource="&rdf;nil" />
                     </rdf:List>
                   </rdf:rest>
                 </rdf:List>
              </rdf:rest>
            </rdf:List>
          </rdf:rest>
        </rdf:List>
      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:DatatypeProperty>
```

Une variation syntaxique plus lisible du même exemple utilisant la notion `nodeID` de RDF est donnée ci-dessous.

```
<owl:DatatypeProperty rdf:ID="tennisGameScore">
  <rdfs:range>
    <owl:DataRange>
      <owl:oneOf>

        <rdf:List>
          <rdf:first rdf:datatype="&xsd;integer">0</rdf:first>
          <rdf:rest rdf:nodeID="l2"/>
        </rdf:List>

        <rdf:List rdf:nodeID="l2">
          <rdf:first rdf:datatype="&xsd;integer">15</rdf:first>
          <rdf:rest rdf:nodeID="l3"/>
        </rdf:List>

        <rdf:List rdf:nodeID="l3">
          <rdf:first rdf:datatype="&xsd;integer">30</rdf:first>
          <rdf:rest rdf:nodeID="l4"/>
        </rdf:List>

        <rdf:List rdf:nodeID="l4">
          <rdf:first rdf:datatype="&xsd;integer">40</rdf:first>
          <rdf:rest rdf:resource="&rdf;nil"/>
```

```
        </rdf:List>

      </owl:oneOf>
    </owl:DataRange>
  </rdfs:range>
</owl:DatatypeProperty>
```

2.5 Informations d'en-tête [Harmelen 2003]

2.5.1 Annotations

OWL Full ne pose aucune contrainte sur les annotations dans une ontologie. OWL DL permet les annotations sur les classes, propriétés, individus et en-têtes d'ontologie, mais seulement dans les conditions suivantes :

- Les propriétés d'annotations doivent avoir un typage explicite de la forme :

  ```
  AnnotationPropertyID rdf:type owl:AnnotationProperty
  ```

- Les propriétés d'annotations ne doivent pas être employées dans les axiomes de propriétés (par exemple, les sous propriétés ne sont pas autorisées).

OWL dispose de cinq annotations prédéfinies, à savoir :

- `owl:versionInfo`
- `rdfs:label`
- `rdfs:comment`
- `rdfs:seeAlso`
- `rdfs:isDefinedBy`

L'exemple suivant montre une utilisation légale d'une propriété d'annotation dans OWL :

```
<owl:AnnotationProperty rdf:about="&dc;creator"/>

<owl:Class rdf:about="MusicalWork">
  <rdfs:label>Musical work</rdf:label>
  <dc:creator>N.N.</dc:creator>
</owl:Class>
```

Cet exemple suppose que `&dc;` et `dc:` pointent respectivement sur l'URI et le namespace de Dublin Core. Ainsi l'emploi des annotations de Dublin Core dans OWL DL requiert un typage explicite. Ceci assure que les annotations sont manipulées de façon correcte sémantiquement par les "reasoners" de OWL DL.

L'emploi de `owl:AnnotationProperty` n'est pas nécessaire dans OWL Full.

2.5.2 En-tête d'ontologie

Un composant d'en-tête est représenté avec une instance de la classe `owl:Ontology`. Une simple en-tête d'ontologie ressemble à ceci :

```
<owl:Ontology rdf:about="">
  <owl:versionInfo>v 1.17 2003/02/26 12:56:51 mdean</owl:versionInfo>
  <rdfs:comment>An example ontology</rdfs:comment>
  <owl:imports rdf:resource="http://www.example.org/foo"/>
</owl:Ontology>
```

Le constructeur `owl:imports` et les constructeurs `owl:priorVersion`, `owl:backwardCompatibleWith` et `owl:inCompatibleWith` sont définis dans le vocabulaire de OWL comme des instances de la classe `owl:OntologyProperty`.

2.5.3 Importation d'une ontologie

`owl:imports` met en référence une autre ontologie OWL contenant des définitions dont la signification est considérée une partie de la signification de l'ontologie d'importation. Chaque référence se compose d'un URI indiquant d'où l'ontologie doit être importée. Techniquement, `owl:imports` est une propriété ayant pour domaine et range la classe `owl:Ontology`.

`owl:imports` est transitif, c'est-à-dire, si l'ontologie A importe l'ontologie B, et B importe C, alors A importe à la fois B et C.

Si l'ontologie A importe B et B importe A, alors A et B sont considérés comme équivalent.

2.5.4 Informations de version

`owl:versionInfo` fournit les informations sur la version. Il peut être utilisé par tous les constructeurs de OWL. Par exemple on peut l'employer dans une classe OWL.

owl:priorVersion contient une référence à une autre ontologie qui identifie l'ontologie spécifiée comme une version antérieure de l'ontologie contenant. Cette information peut être utilisée par un logiciel pour organiser les ontologies par versions. Le constructeur owl:priorVersion est une propriété ayant pour domaine et range la classe owl:Ontology.

owl:backwardCompatibleWith contient une référence à une autre ontologie qui identifie l'ontologie spécifiée comme une version antérieure de l'ontologie contenant, et que l'ontologie contenant est compatible avec l'ontologie spécifiée. En particulier, ceci indique que tous les identificateurs de la précédente version ont les mêmes interprétations dans la nouvelle version. Si owl:backwardCompatibleWith n'est pas déclaré pour deux versions, alors on ne doit pas supposer que les deux versions sont compatibles. owl:backwardCompatibleWith est une propriété intégrée à OWL ayant pour domaine et range la classe owl:Ontology.

owl:incompatibleWith contient une référence à une autre ontologie. Ceci indique que l'ontologie contenant est une version postérieure incompatible de l'ontologie référencée. owl:incompatibleWith est une propriété intégrée à OWL ayant pour domaine et range la classe owl:Ontology.

Deprecation (obsolète) est une caractéristique employée généralement dans le versionnement de logiciels (par exemple avec le langage de programmation Java) pour indiquer qu'une caractéristique particulière est préservée pour la compatibilité vers l'arrière, mais peut être éliminée à l'avenir. Ici un identificateur spécifique est dit être du type owl:DeprecatedClass ou owl:DeprecatedProperty où owl:DeprecatedClass est une sous classe de owl:Class et owl:DeprecatedProperty est une sous classe de rdf:Property. Cette caractéristique permet à des anciennes données et applications de migrer vers une nouvelle version, et ainsi d'accroître le niveau d'adoption de la nouvelle version.

Un exemple de Deprecation est :

```
<owl:Ontology rdf:about="">
  <rdfs:comment>Vehicle Ontology, v. 1.1</rdfs:comment>
  <owl:backwardCompatibleWith rdf:resource="http://www.example.org/vehicle-
   1.0"/>
```

```
  <owl:priorVersion rdf:resource="http://www.example.org/vehicle-1.0"/>
</owl:Ontology>

<owl:DeprecatedClass rdf:ID="Car">
  <rdfs:comment>Automobile is now preferred</rdfs:comment>
</owl:DeprecatedClass>

<owl:Class rdf:ID="Automobile">
  <owl:equivalentClass rdf:resource="#Car"/>
  <!-- note that equivalentClass only means that the classes have the same
   extension, so this DOES NOT lead to the entailment that
   Automobile is of type DeprecatedClass too -->
</owl:Class>

<owl:DeprecatedProperty rdf:ID="hasDriver">
  <rdfs:comment>inverse property drives is now preferred</rdfs:comment>
</owl:DeprecatedProperty>

<owl:ObjectProperty rdf:ID="drives">
  <owl:inverseOf rdf:resource="#hasDriver"/>
</owl:ObjectProperty>
```

3 Description incrémentale de OWL DL et OWL Full

3.1 Descriptions de classe

3.1.1 Enumération : oneOf

L'énumération est un type de description de classe. Les membres de la classe sont exactement l'ensemble des individus énumérés. La liste d'individus est typiquement représentée à l'aide du constructeur `rdf:parseType="Collection"`, qui fournit une courte façon d'écrire un ensemble d'éléments de liste. Par exemple, la syntaxe RDF/XML suivante définit la classe de tous les continents :

```
<owl:Class rdf:ID="Continent">
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Eurasia"/>
    <owl:Thing rdf:about="#Africa"/>
    <owl:Thing rdf:about="#NorthAmerica"/>
    <owl:Thing rdf:about="#SouthAmerica"/>
    <owl:Thing rdf:about="#Australia"/>
    <owl:Thing rdf:about="#Antarctica"/>
  </owl:oneOf>
</owl:Class>
```

La syntaxe RDF/XML `<owl:Thing rdf:about="..."/>` se rapporte à un certain individu (tous les individus sont par définition des instances de `owl:Thing`). . On utilise `owl:Thing` comme un simple cliché pour présenter la référence. Alternativement, on pourrait référencer les éléments de l'ensemble selon leur type spécifique `Continent` comme suit :

```
<owl:Class rdf:ID="Continent">
  <owl:oneOf rdf:parseType="Collection">
```

```
    <Continent rdf:about="#Eurasia"/>
    <Continent rdf:about="#Africa"/>
    <Continent rdf:about="#NorthAmerica"/>
    <Continent rdf:about="#SouthAmerica"/>
    <Continent rdf:about="#Australia"/>
    <Continent rdf:about="#Antarctica"/>
  </owl:oneOf>
</owl:Class>
```

Des constructions plus complexes des individus sont également des éléments valides du constructeur `oneOf` , par exemple :

```
<Continent rdf:about="#Africa">
  <rdf:label>Africa</rdf:label>
</Continent>
```

3.1.2 Restrictions de propriété

3.1.2.1 Contrainte de valeurs : hasValue

La contrainte de valeur `owl:hasValue` est une propriété intégrée de OWL qui lie une classe de restriction à une valeur V. Cette valeur V peut être un individu ou une valeur de donnée. Une restriction contenant une contrainte `owl:hasValue` décrit une classe dont tous les individus ont au moins une valeur sémantiquement égale à V pour la propriété concernée.

L'exemple suivant décrit la classe des individus qui ont pour parent l'individu désigné sous le nom de Anfana.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#hasParent" />
  <owl:hasValue rdf:resource="#Anfana" />
</owl:Restriction>
```

Comme `allValuesFrom` et `someValuesFrom`, `owl:hasValue` est une restriction locale qui permet à `hasParent` de s'appliquer à une classe d'individus.

3.1.2.2 Contraintes de cardinalité : minCardinality, maxCardinality, cardinality

OWL suppose qu'une propriété particulière de toute instance de classe peut avoir un nombre arbitraire (zéro ou plusieurs) de valeurs. Les contraintes de cardinalité peuvent être utilisées pour rendre une propriété obligatoire ou optionnelle.

Alors que OWL Lite restreint les cardinalités minimale et maximale à exactement 1 ou 0, OWL (DL et Full) admet des entiers arbitraires non négatifs. `owl:maxCardinality` spécifie

la borne supérieure et owl:minCardinality spécifie la borne inférieure. Les deux bornes combinées spécifient une plage de valeurs (range).

2.3.1.2.3 Combinaisons booléennes : intersectionOf, complementOf, unionOf

Les trois formes de descriptions de classe dans cette section représentent les constructeurs de classe les plus avancés qui sont utilisés dans la logique de description. Ils peuvent être vu comme les opérateurs AND, OR et NOT.

owl:intersectionOf

La propriété owl:intersectionOf lie une classe à une liste de descriptions de classe. Elle décrit une classe dont l'extension contient précisément les individus qui sont membres de l'extension de classe de toutes les descriptions de classe dans la liste de range.

Les exemples suivants montrent l'utilisation du constructeur owl:intersectionOf.

```
<owl:Class rdf:ID="WhiteWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#White" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

Les classes construites en utilisant les opérateurs d'ensemble sont comme des définitions. Les membres de ces classes sont complètement indiqués par l'opérateur d'ensemble. Le constructeur déclaré ci-dessous énonce que WhiteWine est exactement l'intersection de la classe Wine et de l'ensemble des choses qui ont la couleur blanche. Ceci signifie que si quelque chose a la couleur blanche et est du vin alors c'est une instance de WhiteWine. owl:intersectionOf est un puissant outil pour la catégorisation des individus. Notons que 'rdf:parseType="Collection"' est un élément syntaxique obligatoire.

Ici on définit Bourgogne pour inclure exactement les vins qui ont au moins une relation locatedIn à la région de Bourgogne.

```
<owl:Class rdf:about="#Burgundy">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine" />
```

```
    <owl:Restriction>
      <owl:onProperty rdf:resource="#locatedIn" />
      <owl:hasValue rdf:resource="#BourgogneRegion" />
    </owl:Restriction>
  </owl:intersectionOf>
</owl:Class>
```

En conclusion, la classe `WhiteBurgundy` est exactement l'intersection de vins blanc et de Bourgogne. En conséquence, tous les différents vins qui répondent à ces critères font partie de l'extension de la classe `WhiteBurgundy`.

```
<owl:Class rdf:ID="WhiteBurgundy">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Burgundy" />
    <owl:Class rdf:about="#WhiteWine" />
  </owl:intersectionOf>
</owl:Class>
```

owl:unionOf

La propriété `owl:unionOf` lie une classe à une liste de descriptions de classe. Elle décrit une classe anonyme dont l'extension contient les individus qui sont dans au moins une des extensions des descriptions de classe de la liste de range.

L'exemple suivant montre l'emploi du constructeur `unionOf` :

```
<owl:Class rdf:ID="Fruit">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

La classe `Fruit` inclut à la fois l'extension de `SweetFruit` et l'extension de `NonSweetFruit`.

L'exemple ci-dessous montre que les instances de `Fruit` sont un sous ensemble de l'intersection de `SweetFruit` et `NonSweetFruit`, qui est bien entendu l'ensemble vide.

```
<owl:Class rdf:ID="Fruit">
  <rdfs:subClassOf rdf:resource="#SweetFruit" />
  <rdfs:subClassOf rdf:resource="#NonSweetFruit" />
</owl:Class>
```

owl:complementOf

La propriété owl:complementOf lie une classe à précisément une description de classe. Elle décrit une classe dont l'extension contient exactement les individus qui n'appartiennent pas à l'extension de la classe de range. owl:complementOf est analogue à la négation logique, mais restreint aux individus seulement.

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

  <owl:Class rdf:ID="NonConsumableThing">
    <owl:complementOf rdf:resource="#ConsumableThing" />
  </owl:Class>
```

La classe NonConsumableThing inclut comme membres tous les individus qui n'appartiennent pas à l'extension de ConsumableThing. Cet ensemble contient tous les vins, régions, etc.. C'est une différence réglée entre owl:Thing et ConsumableThing. Par conséquent un usage typique de complementOf est de le combiner avec d'autres opérateurs d'ensemble.

```
<owl:Class rdf:ID="NonFrenchWine">
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Wine"/>
    <owl:Class>
      <owl:complementOf>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#locatedIn" />
          <owl:hasValue rdf:resource="#FrenchRegion" />
        </owl:Restriction>
      </owl:complementOf>
    </owl:Class>
  </owl:intersectionOf>
</owl:Class>
```

Cet exemple définit la classe NonFrenchWine comme l'intersection de Wine et de l'ensemble de toutes les choses non situées en France.

3.2 Axiomes de classe

3.2.1 Classes complexes

Dans plusieurs constructions, OWL Lite restreint la syntaxe à de simples noms de classe (par exemple subclassOf ou equivalentClass). OWL (DL et Full) étend cette restriction pour permettre arbitrairement les descriptions de classes complexes, composées de classes énumérées, de restrictions de propriétés et de combinaisons booléennes. OWL (DL et Full) inclut également une classe spéciale appelée Nothing qui est la classe qui n'a aucune instance.

Contrairement à OWL Lite et OWL DL, **OWL Full permet d'utiliser les classes comme des instances.**

3.2.2 owl:disjointWith

Schéma d'axiome : description de classe `owl:disjointWith` description de classe.

Le constructeur `owl:disjointWith` est une propriété intégrée de OWL ayant une description de classe comme domaine et plage de valeurs. Il exprime que les extensions de classe de deux descriptions de classe n'ont aucun individu en commun.

Par exemple les classes **Homme** et **Femme** peuvent être disjointes. On peut déduire une incohérence si un individu est à la fois instance de ces deux classes. De façon analogue, si **X** est une instance **Homme**, alors **X** ne peut pas être une instance de **Femme**.

```
<owl:Class rdf:about="Homme">
  <owl:disjointWith rdf:resource="#Femme"/>
</owl:Class>
```

L'exemple de la pâte (`Pasta`) démontre une multiple disjonctions de classes. Il affirme seulement que les pâtes sont disjointes de toutes les autres classes. Il n'affirme pas que la viande (`Meat`) et le `Fruit` dont disjoints.

```
<owl:Class rdf:ID="Pasta">
  <rdfs:subClassOf rdf:resource="#EdibleThing"/>
  <owl:disjointWith rdf:resource="#Meat"/>
  <owl:disjointWith rdf:resource="#Fowl"/>
  <owl:disjointWith rdf:resource="#Seafood"/>
  <owl:disjointWith rdf:resource="#Dessert"/>
  <owl:disjointWith rdf:resource="#Fruit"/>
</owl:Class>
```

Afin d'affirmer que les classes sont deux à deux disjoints, on faire une assertion `owl:disjointWith` pour chaque paire.

```
<owl:Class rdf:ID="SweetFruit">
  <rdfs:subClassOf rdf:resource="#EdibleThing" />
</owl:Class>

<owl:Class rdf:ID="NonSweetFruit">
  <rdfs:subClassOf rdf:resource="#EdibleThing" />
  <owl:disjointWith rdf:resource="#SweetFruit" />
</owl:Class>

<owl:Class rdf:ID="Fruit">
```

```
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#SweetFruit" />
    <owl:Class rdf:about="#NonSweetFruit" />
  </owl:unionOf>
</owl:Class>
```

On a défini ci-dessus Fruit comme exactement l'union de SweetFruit et NonSweetFruit. Et ces deux sous classes divisent Fruit en deux sous classes distinctes puisqu'elles sont disjointes.

4 Conclusion

Cette partie a donné une vue d'ensemble du langage d'ontologie du Web OWL en fournissant une brève introduction sur la nécessité d'un langage d'ontologie du Web. Il a aussi présenté les trois sous langages de OWL à savoir OWL Lite, OWL DL et OWL Full avec une synthèse des caractéristiques de chacun de ces langages.

ANNEXE B : DESCRIPTION OWL DU PROTOTYPE

Code généré avec Protégé v2.0 alpha

```
<rdf:RDF
    xmlns:j.0="http://owl.protege.stanford.edu#"
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:j.0="http://protege.stanford.edu/plugins/owl/protege#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
  <owl:Ontology>
    <owl:versionInfo
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    ></owl:versionInfo>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >TRAORE Anfana
Description de meta donnees de ressources pedagogiques</rdfs:comment>
  </owl:Ontology>
  <owl:Class rdf:ID="Auteur">
    <rdfs:subClassOf rdf:resource="#Personne"
       rdf:type="http://www.w3.org/2002/07/owl#Class"
       j.0:abstract="true"/>
  </owl:Class>
  <owl:Class rdf:ID="Examen">
    <rdfs:subClassOf rdf:resource="#Contenu"
       rdf:type="http://www.w3.org/2002/07/owl#Class"
       j.0:abstract="true"/>
    <owl:disjointWith>
      <owl:Class rdf:about="#Exercice"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#Corrige"/>
    </owl:disjointWith>
    <owl:disjointWith>
      <owl:Class rdf:about="#FAQ"/>
    </owl:disjointWith>
    <rdfs:subClassOf rdf:resource="#Ressource"
       rdf:type="http://www.w3.org/2002/07/owl#Class"
       j.0:abstract="true"/>
  </owl:Class>
  <owl:Class rdf:ID="Cours"/>
  <owl:Class rdf:ID="Discipline"/>
  <owl:Class>
    <owl:unionOf rdf:parseType="Collection">
      <owl:Class rdf:about="#Enseignant"/>
      <owl:Class rdf:about="#Secretaire"/>
    </owl:unionOf>
  </owl:Class>
  <owl:Class rdf:ID="Prerequis">
    <rdfs:subClassOf rdf:resource="#Informations_sur_le_cours"
       rdf:type="http://www.w3.org/2002/07/owl#Class"
       j.0:abstract="true"/>
```

```
</owl:Class>
<owl:Class rdf:ID="Ligne_plan"/>
<owl:Class rdf:nodeID="A0">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Enseignant"/>
    <owl:Class rdf:about="#Secretaire"/>
    <owl:Class rdf:about="#Prerequis"/>
  </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="Ouvrage_de_reference"/>
<owl:Class rdf:nodeID="A1">
  <owl:unionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Support_Cours"/>
    <owl:Class rdf:about="#Examen"/>
    <owl:Class rdf:about="#Exercice"/>
    <owl:Class rdf:about="#Corrige"/>
    <owl:Class rdf:about="#FAQ"/>
  </owl:unionOf>
</owl:Class>
<owl:Class rdf:ID="Concept"/>
<owl:Class rdf:ID="Ligne_programme"/>
<owl:Class rdf:ID="Support_Cours">
  <rdfs:subClassOf rdf:resource="#Contenu"/>
  <owl:disjointWith rdf:resource="#Examen"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Exercice"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#Corrige"/>
  </owl:disjointWith>
  <owl:disjointWith>
    <owl:Class rdf:about="#FAQ"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Ressource"/>
</owl:Class>
<owl:Class rdf:ID="Corrige">
  <rdfs:subClassOf rdf:resource="#Contenu"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#FAQ"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Ressource"/>
</owl:Class>
<owl:Class rdf:ID="Exercice">
  <rdfs:subClassOf rdf:resource="#Contenu"/>
  <owl:disjointWith rdf:resource="#Corrige"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#FAQ"/>
  </owl:disjointWith>
  <rdfs:subClassOf rdf:resource="#Ressource"/>
</owl:Class>
<owl:Class rdf:ID="Secretaire">
  <rdfs:subClassOf rdf:resource="#Personne"/>
  <owl:disjointWith>
    <owl:Class rdf:about="#Enseignant"/>
  </owl:disjointWith>
  <owl:disjointWith rdf:resource="#Auteur"/>
```

```
  <rdfs:subClassOf rdf:resource="#Informations_sur_le_cours"/>
</owl:Class>
<owl:Class rdf:ID="FAQ">
  <rdfs:subClassOf rdf:resource="#Contenu"/>
  <rdfs:subClassOf rdf:resource="#Ressource"/>
</owl:Class>
<owl:Class rdf:ID="Ontologie_globale"/>
<owl:Class rdf:ID="Enseignant">
  <rdfs:subClassOf rdf:resource="#Personne"/>
  <rdfs:subClassOf rdf:resource="#Informations_sur_le_cours"/>
</owl:Class>
<owl:Class rdf:ID="Universite"/>
<owl:ObjectProperty rdf:ID="ressources_ligne_plan">
  <rdfs:domain rdf:resource="#Ligne_plan"/>
  <rdfs:range rdf:resource="#Ressource"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="concepts_ligne_plan">
  <rdfs:domain rdf:resource="#Ligne_plan"/>
  <rdfs:range rdf:resource="#Concept"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ressources_ligne_programme">
  <rdfs:domain rdf:resource="#Ligne_programme"/>
  <rdfs:range rdf:resource="#Ressource"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="exercices_du_support_cours">
  <rdfs:domain rdf:resource="#Support_Cours"/>
  <rdfs:range rdf:resource="#Exercice"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="programme">
  <rdfs:domain rdf:resource="#Cours"/>
  <rdfs:range rdf:resource="#Ligne_programme"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="auteurs_ouvrage">
  <rdfs:domain rdf:resource="#Ouvrage_de_reference"/>
  <rdfs:range rdf:resource="#Auteur"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ressources_concept">
  <rdfs:domain rdf:resource="#Concept"/>
  <rdfs:range rdf:resource="#Ressource"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="plan_cours">
  <rdfs:domain rdf:resource="#Cours"/>
  <rdfs:range rdf:resource="#Ligne_plan"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="informations_sur_le_cours">
  <rdfs:domain rdf:resource="#Cours"/>
  <rdfs:range rdf:nodeID="A0"/>
  <rdfs:range rdf:resource="#Informations_sur_le_cours"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="contenu_cours">
  <rdfs:domain rdf:resource="#Cours"/>
  <rdfs:range rdf:nodeID="A1"/>
  <rdfs:range rdf:resource="#Contenu"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="se_compose_de_concepts">
  <rdfs:domain rdf:resource="#Ontologie_globale"/>
```

```
    <rdfs:range rdf:resource="#Concept"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="examens_du_support_cours">
    <rdfs:domain rdf:resource="#Support_Cours"/>
    <rdfs:range rdf:resource="#Examen"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="ouvrage_de_reference">
    <rdfs:domain rdf:resource="#Cours"/>
    <rdfs:range rdf:resource="#Ouvrage_de_reference"/>
  </owl:ObjectProperty>
  <owl:DatatypeProperty rdf:ID="telephone"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Personne"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="niveau"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Cours"/>
    <rdfs:comment rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >1. la licence comporte six semestres
2. le master comporte quatre semestres
3. le doctorat comporte six semestres</rdfs:comment>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="semestre"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Cours"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="emploi_du_temps">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Cours"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="notations_examen">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Cours"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="prerequis">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Prerequis"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="titre_chapitre"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Ligne_programme"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="adresse_postale"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Enseignant"/>
  </owl:DatatypeProperty>
  <owl:DatatypeProperty rdf:ID="id_cours"
     rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="#Cours"/>
  </owl:DatatypeProperty>
```

```
<owl:DatatypeProperty rdf:ID="date_edition"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ouvrage_de_reference"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="date_ligne_programme"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ligne_programme"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="titre_ouvrage"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ouvrage_de_reference"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="format"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Contenu"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="libelle_concept"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Concept"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="nom_prenoms"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="edition_ouvrage"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ouvrage_de_reference"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="id_contenu"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Contenu"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="e-mail"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Personne"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="numero_chapitre"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ligne_programme"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="annonces">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="informations_sur_exercices">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
```

```
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="themes">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="charte">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="site_web"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="communication_equipe"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Cours"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="id_ligne_plan"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ligne_plan"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="libelle_ligne_plan"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Ligne_plan"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="bureau"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Secretaire"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="id_universite"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Universite"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="id_discipline"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Discipline"/>
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="heure_bureau"
   rdf:type="http://www.w3.org/2002/07/owl#FunctionalProperty">
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
  <rdfs:domain rdf:resource="#Enseignant"/>
</owl:DatatypeProperty>
<owl:FunctionalProperty rdf:ID="auteur_ressource"
   rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
  <rdfs:domain rdf:resource="#Ressource"/>
  <rdfs:range rdf:resource="#Enseignant"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="discipline"
```

```
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Cours"/>
   <rdfs:range rdf:resource="#Discipline"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="pere_concept"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Concept"/>
   <rdfs:range rdf:resource="#Concept"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="pere_ligne_plan"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Ligne_plan"/>
   <rdfs:range rdf:resource="#Ligne_plan"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="contrainte_de_precedence"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Concept"/>
   <rdfs:range rdf:resource="#Concept"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="fils_gauche_ligne_plan"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Ligne_plan"/>
   <rdfs:range rdf:resource="#Ligne_plan"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="suivant_ligne_programme"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Ligne_programme"/>
   <rdfs:range rdf:resource="#Ligne_programme"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="corrige_exercice"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Exercice"/>
   <rdfs:range rdf:resource="#Corrige"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="universite"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Cours"/>
   <rdfs:range rdf:resource="#Universite"/>
</owl:FunctionalProperty>
<owl:FunctionalProperty rdf:ID="corrie_examen"
    rdf:type="http://www.w3.org/2002/07/owl#ObjectProperty">
   <rdfs:domain rdf:resource="#Examen"/>
   <rdfs:range rdf:resource="#Corrige"/>
</owl:FunctionalProperty>
<j.0:Auteur rdf:ID="Instance_42"
    j.0:nom_prenoms="Widom"/>
<j.0:Support_Cours rdf:ID="Instance_57"
    j.0:id_contenu="Support_Cours3"
    j.0:format=".PPT">
   <j.0:auteur_ressource>
     <j.0:Enseignant rdf:ID="Instance_36"
        j.0:e-mail="cooperb@cs.stanford.edu"
        j.0:telephone="650-723-1923">
       <j.0:nom_prenoms>Brian Cooper</j.0:nom_prenoms>
       <j.0:heure_bureau>Monday : 11-12 am</j.0:heure_bureau>
     </j.0:Enseignant>
```

```
  </j.0:auteur_ressource>
  <j.0:examens_du_support_cours>
    <j.0:Examen rdf:ID="Instance_61"
       j.0:id_contenu="Examen1"
       j.0:format=".PPT">
      <j.0:auteur_ressource rdf:resource="#Instance_36"/>
      <j.0:corrie_examen rdf:resource="#Instance_71"
         rdf:type="#Corrige"
         j.0:id_contenu="Corrige5"
         j.0:format=".PPT"/>
    </j.0:Examen>
  </j.0:examens_du_support_cours>
</j.0:Support_Cours>
<j.0:Exercice rdf:ID="Instance_64"
   j.0:id_contenu="Exercice2"
   j.0:format=".HTML">
  <j.0:auteur_ressource>
    <j.0:Enseignant rdf:ID="Instance_37"
       j.0:e-mail="rsram@cs.stanford.edu"
       j.0:telephone="650-723-6805">
      <j.0:nom_prenoms>Sriram Raghavan</j.0:nom_prenoms>
      <j.0:heure_bureau>Wednesday : 11-12 am</j.0:heure_bureau>
    </j.0:Enseignant>
  </j.0:auteur_ressource>
  <j.0:corrige_exercice rdf:resource="#Instance_68"
     rdf:type="#Corrige"
     j.0:id_contenu="Corrige2"
     j.0:format=".PPT"/>
</j.0:Exercice>
<j.0:Auteur rdf:ID="Instance_25">
  <j.0:nom_prenoms>garcia molina</j.0:nom_prenoms>
</j.0:Auteur>
<j.0:Support_Cours rdf:ID="Instance_58"
   j.0:id_contenu="Support_Cours4"
   j.0:format="AUTRES">
  <j.0:auteur_ressource rdf:resource="#Instance_37"/>
  <j.0:examens_du_support_cours>
    <j.0:Examen rdf:ID="Instance_62"
       j.0:id_contenu="Examen2"
       j.0:format=".PPT">
      <j.0:auteur_ressource rdf:resource="#Instance_37"/>
    </j.0:Examen>
  </j.0:examens_du_support_cours>
</j.0:Support_Cours>
<rdf:Description rdf:ID="Instance_13">
  <rdf:type rdf:nodeID="A1"/>
</rdf:Description>
<j.0:Ligne_plan rdf:ID="Instance_72"
   j.0:libelle_ligne_plan="Introduction">
  <j.0:id_ligne_plan>Chapitre 1</j.0:id_ligne_plan>
  <j.0:ressources_ligne_plan>
    <j.0:Support_Cours rdf:ID="Instance_55"
       j.0:id_contenu="Support_Cours1"
       j.0:format=".PPT">
      <j.0:auteur_ressource rdf:resource="#Instance_36"/>
      <j.0:examens_du_support_cours rdf:resource="#Instance_61"/>
```

```
        <j.0:exercices_du_support_cours>
          <j.0:Exercice rdf:ID="Instance_63"
             j.0:id_contenu="Exercice1"
             j.0:format=".HTML">
            <j.0:auteur_ressource rdf:resource="#Instance_37"/>
            <j.0:corrige_exercice rdf:resource="#Instance_67"
               rdf:type="#Corrige"
               j.0:id_contenu="Corrige1"
               j.0:format=".PPT"/>
          </j.0:Exercice>
        </j.0:exercices_du_support_cours>
        <j.0:exercices_du_support_cours rdf:resource="#Instance_64"/>
      </j.0:Support_Cours>
    </j.0:ressources_ligne_plan>
    <j.0:concepts_ligne_plan>
      <j.0:Concept rdf:ID="Instance_85"
         j.0:libelle_concept="INTRODUCTION">
        <j.0:ressources_concept rdf:resource="#Instance_55"/>
        <j.0:pere_concept>
          <j.0:Concept rdf:ID="Instance_84">
            <j.0:libelle_concept>DATABASE SYSTEM
PRINCIPLES</j.0:libelle_concept>
            <j.0:ressources_concept rdf:resource="#Instance_55"/>
            <j.0:ressources_concept>
              <j.0:Support_Cours rdf:ID="Instance_56"
                 j.0:id_contenu="Support_Cours2"
                 j.0:format=".PPT">
                <j.0:auteur_ressource rdf:resource="#Instance_37"/>
                <j.0:examens_du_support_cours
rdf:resource="#Instance_61"/>
                <j.0:exercices_du_support_cours>
                  <j.0:Exercice rdf:ID="Instance_65"
                     j.0:id_contenu="Exercice3"
                     j.0:format=".PDF">
                    <j.0:auteur_ressource rdf:resource="#Instance_36"/>
                    <j.0:corrige_exercice rdf:resource="#Instance_69"
                       rdf:type="#Corrige"
                       j.0:id_contenu="Corrige3"
                       j.0:format=".HTML"/>
                  </j.0:Exercice>
                </j.0:exercices_du_support_cours>
              </j.0:Support_Cours>
            </j.0:ressources_concept>
            <j.0:ressources_concept rdf:resource="#Instance_57"/>
            <j.0:ressources_concept rdf:resource="#Instance_58"/>
            <j.0:ressources_concept>
              <j.0:Support_Cours rdf:ID="Instance_59"
                 j.0:id_contenu="Support_Cours5"
                 j.0:format="AUTRES">
                <j.0:auteur_ressource rdf:resource="#Instance_37"/>
                <j.0:examens_du_support_cours
rdf:resource="#Instance_62"/>
              </j.0:Support_Cours>
            </j.0:ressources_concept>
            <j.0:ressources_concept>
              <j.0:Support_Cours rdf:ID="Instance_60"
```

```
                j.0:id_contenu="Support_Cours6"
                j.0:format="AUTRES">
               <j.0:auteur_ressource rdf:resource="#Instance_37"/>
               <j.0:examens_du_support_cours
rdf:resource="#Instance_62"/>
             </j.0:Support_Cours>
           </j.0:ressources_concept>
           <j.0:pere_concept>
             <j.0:Concept rdf:ID="Instance_102">
               <j.0:libelle_concept>COMPUTER
SCIENCE</j.0:libelle_concept>
             </j.0:Concept>
           </j.0:pere_concept>
         </j.0:Concept>
       </j.0:pere_concept>
       <j.0:contrainte_de_precedence>
         <j.0:Concept rdf:ID="Instance_86"
            j.0:libelle_concept="HARDWARE">
           <j.0:pere_concept rdf:resource="#Instance_84"/>
           <j.0:ressources_concept rdf:resource="#Instance_55"/>
         </j.0:Concept>
       </j.0:contrainte_de_precedence>
     </j.0:Concept>
   </j.0:concepts_ligne_plan>
 </j.0:Ligne_plan>
 <j.0:Ligne_programme rdf:ID="Instance_44"
    j.0:date_ligne_programme="25/06/2003">
   <j.0:numero_chapitre>1, 2</j.0:numero_chapitre>
   <j.0:titre_chapitre>Introduction, Hardware</j.0:titre_chapitre>
   <j.0:suivant_ligne_programme>
     <j.0:Ligne_programme rdf:ID="Instance_45"
        j.0:date_ligne_programme="30/06/2003"
        j.0:numero_chapitre="3">
       <j.0:titre_chapitre>File and system
structure</j.0:titre_chapitre>
       <j.0:suivant_ligne_programme>
         <j.0:Ligne_programme rdf:ID="Instance_46"
            j.0:date_ligne_programme="02/07/2003"
            j.0:numero_chapitre="4">
           <j.0:titre_chapitre>Indexing and
Hashing</j.0:titre_chapitre>
           <j.0:suivant_ligne_programme>
             <j.0:Cours rdf:ID="Instance_47"
                rdf:type="#Ligne_programme"
                j.0:date_ligne_programme="07/07/2003"
                j.0:numero_chapitre="5">
               <j.0:titre_chapitre>Indexing and
hashing</j.0:titre_chapitre>
               <j.0:suivant_ligne_programme>
                 <j.0:Ligne_programme rdf:ID="Instance_49"
                    j.0:date_ligne_programme="09/07/2003"
                    j.0:numero_chapitre="6">
                   <j.0:titre_chapitre>Query
processing</j.0:titre_chapitre>
                   <j.0:suivant_ligne_programme>
                     <j.0:Ligne_programme rdf:ID="Instance_50"
```

```
                          j.0:date_ligne_programme="14/07/2003"
                          j.0:numero_chapitre="7">
                         <j.0:titre_chapitre>Query
processing</j.0:titre_chapitre>
                         <j.0:suivant_ligne_programme>
                           <j.0:Ligne_programme rdf:ID="Instance_51"
                              j.0:date_ligne_programme="16/07/2003">
                             <j.0:titre_chapitre>Midterm
exam</j.0:titre_chapitre>
                             <j.0:suivant_ligne_programme>
                               <j.0:Ligne_programme rdf:ID="Instance_52"
                                  j.0:date_ligne_programme="21/07/2003"
                                  j.0:numero_chapitre="8">
                                 <j.0:titre_chapitre>Midterm answers,
Crash recovery</j.0:titre_chapitre>
                                 <j.0:suivant_ligne_programme>
                                   <j.0:Ligne_programme
rdf:ID="Instance_53"

j.0:date_ligne_programme="23/07/2003"
                                      j.0:numero_chapitre="9">
                                    <j.0:titre_chapitre>Crash recovery,
Concurrency control</j.0:titre_chapitre>
                                    <j.0:suivant_ligne_programme>
                                      <j.0:Ligne_programme
rdf:ID="Instance_54"

j.0:date_ligne_programme="28/07/2003"
                                         j.0:numero_chapitre="9">
                                       <j.0:titre_chapitre>Concurrency
control</j.0:titre_chapitre>
                                      </j.0:Ligne_programme>
                                    </j.0:suivant_ligne_programme>
                                  </j.0:Ligne_programme>
                                </j.0:suivant_ligne_programme>
                              </j.0:Ligne_programme>
                            </j.0:suivant_ligne_programme>
                          </j.0:Ligne_programme>
                        </j.0:suivant_ligne_programme>
                        <j.0:ressources_ligne_programme
rdf:resource="#Instance_59"/>
                    </j.0:Ligne_programme>
                  </j.0:suivant_ligne_programme>
                  <j.0:ressources_ligne_programme
rdf:resource="#Instance_58"/>
                </j.0:Cours>
              </j.0:suivant_ligne_programme>
              <j.0:ressources_ligne_programme
rdf:resource="#Instance_57"/>
            </j.0:Ligne_programme>
          </j.0:suivant_ligne_programme>
          <j.0:ressources_ligne_programme rdf:resource="#Instance_56"/>
        </j.0:Ligne_programme>
      </j.0:suivant_ligne_programme>
```

```
    <j.0:ressources_ligne_programme rdf:resource="#Instance_55"/>
  </j.0:Ligne_programme>
  <rdf:Description rdf:ID="Instance_11">
    <rdf:type rdf:nodeID="A0"/>
  </rdf:Description>
  <j.0:Concept rdf:ID="Instance_88"
     j.0:libelle_concept="FILE">
    <j.0:pere_concept>
      <j.0:Concept rdf:ID="Instance_87">
        <j.0:libelle_concept>FILE AND SYSTEM
STRUCTURE</j.0:libelle_concept>
        <j.0:pere_concept rdf:resource="#Instance_84"/>
        <j.0:ressources_concept rdf:resource="#Instance_56"/>
      </j.0:Concept>
    </j.0:pere_concept>
    <j.0:ressources_concept rdf:resource="#Instance_56"/>
  </j.0:Concept>
  <j.0:Cours rdf:ID="Instance_10"/>
  <j.0:Ligne_plan rdf:ID="Instance_74">
    <j.0:id_ligne_plan>Chapitre 3</j.0:id_ligne_plan>
    <j.0:libelle_ligne_plan>File and system
structure</j.0:libelle_ligne_plan>
    <j.0:ressources_ligne_plan rdf:resource="#Instance_56"/>
    <j.0:concepts_ligne_plan rdf:resource="#Instance_87"/>
    <j.0:concepts_ligne_plan rdf:resource="#Instance_88"/>
    <j.0:concepts_ligne_plan>
      <j.0:Concept rdf:ID="Instance_89">
        <j.0:libelle_concept>SYSTEM STRUCTURE</j.0:libelle_concept>
        <j.0:pere_concept rdf:resource="#Instance_87"/>
        <j.0:ressources_concept rdf:resource="#Instance_56"/>
      </j.0:Concept>
    </j.0:concepts_ligne_plan>
  </j.0:Ligne_plan>
  <j.0:Cours rdf:ID="Instance_29"/>
  <rdf:Description rdf:ID="Instance_7">
    <rdf:type rdf:nodeID="A0"/>
  </rdf:Description>
  <j.0:Auteur rdf:ID="Instance_41"
     j.0:nom_prenoms="Ullman"/>
  <j.0:Cours rdf:ID="Instance_27"/>
  <j.0:Ligne_programme rdf:ID="Instance_26"/>
  <j.0:Universite rdf:ID="Instance_98"/>
  <rdf:Description rdf:ID="Instance_48">
    <rdf:type rdf:nodeID="A0"/>
  </rdf:Description>
  <j.0:Concept rdf:ID="Instance_92"
     j.0:libelle_concept="HASHING">
    <j.0:pere_concept>
      <j.0:Concept rdf:ID="Instance_90">
        <j.0:libelle_concept>INDEXING AND HASHING</j.0:libelle_concept>
        <j.0:pere_concept rdf:resource="#Instance_84"/>
        <j.0:ressources_concept rdf:resource="#Instance_57"/>
      </j.0:Concept>
    </j.0:pere_concept>
    <j.0:ressources_concept rdf:resource="#Instance_57"/>
  </j.0:Concept>
```

```
  <j.0:Concept rdf:ID="Instance_93">
    <j.0:libelle_concept>QUERY PROCESSING</j.0:libelle_concept>
    <j.0:pere_concept rdf:resource="#Instance_84"/>
    <j.0:ressources_concept rdf:resource="#Instance_58"/>
    <j.0:ressources_concept rdf:resource="#Instance_59"/>
  </j.0:Concept>
  <rdf:Description rdf:ID="Instance_6">
    <rdf:type rdf:nodeID="A0"/>
  </rdf:Description>
  <j.0:Ouvrage_de_reference rdf:ID="Instance_22"/>
  <j.0:Ouvrage_de_reference rdf:ID="Instance_23"
     j.0:titre_ouvrage="database">
    <j.0:auteurs_ouvrage rdf:resource="#Instance_24"
       rdf:type="#Auteur"
       j.0:nom_prenoms="widom"/>
    <j.0:auteurs_ouvrage rdf:resource="#Instance_25"/>
  </j.0:Ouvrage_de_reference>
  <j.0:Cours rdf:ID="Instance_5"
     j.0:niveau="DOCTORAT"
     j.0:semestre="SEMESTRE_4"/>
  <j.0:Ligne_programme rdf:ID="Instance_19"/>
  <j.0:Ligne_programme rdf:ID="Instance_18"/>
  <j.0:Ouvrage_de_reference rdf:ID="Instance_43">
    <j.0:titre_ouvrage>DATABASE SYSTEM, THE COMPLETE
BOOK</j.0:titre_ouvrage>
    <j.0:auteurs_ouvrage rdf:resource="#Instance_40"
       rdf:type="#Auteur"
       j.0:nom_prenoms="Garcia-Molina"/>
    <j.0:auteurs_ouvrage rdf:resource="#Instance_41"/>
    <j.0:auteurs_ouvrage rdf:resource="#Instance_42"/>
  </j.0:Ouvrage_de_reference>
  <j.0:Cours rdf:ID="Instance_21"/>
  <j.0:Cours rdf:ID="Instance_14"/>
  <j.0:Examen rdf:ID="Instance_4"/>
  <j.0:Cours rdf:ID="Instance_35"
     j.0:niveau="MASTER"
     j.0:semestre="SEMESTRE_1"
     j.0:site_web="http://www.stanford.edu/class/cs245"
     j.0:communication_equipe="cs245-staff@cs.stanford.edu">
    <j.0:id_cours>CS - 245 Database System Principles</j.0:id_cours>
    <j.0:annonces>Sections 10.4-10.7 were initially listed as optional
reading.</j.0:annonces>
    <j.0:annonces>Assignment 5 is available from the handouts
page</j.0:annonces>
    <j.0:annonces>Clarification to Homework 4 : For problem 6, you can
assume that S1 and S2 have the same actions, though in a possibly
different order.</j.0:annonces>
    <j.0:annonces>Midterm solutions posted.</j.0:annonces>
    <j.0:themes>File organisation and access, buffer management,
performance analysis, and storage management.</j.0:themes>
    <j.0:themes>Database system architecture, query optimization,
transaction management, recovery, concurrency control.</j.0:themes>
    <j.0:themes>Reliability, protection, and integrity.</j.0:themes>
    <j.0:themes>Design and management issues.</j.0:themes>
    <j.0:informations_sur_le_cours>
      <j.0:Secretaire rdf:ID="Instance_38"
```

```
          j.0:e-mail="siroker@cs.stanford.edu"
          j.0:telephone="650-723-0872">
        <j.0:nom_prenoms>Marianne Siroker</j.0:nom_prenoms>
        <j.0:bureau>Gates 435</j.0:bureau>
      </j.0:Secretaire>
    </j.0:informations_sur_le_cours>
    <j.0:emploi_du_temps>Monday 1:15-3:05 pm</j.0:emploi_du_temps>
    <j.0:emploi_du_temps>Wednesday 1:15-3:05 pm</j.0:emploi_du_temps>
    <j.0:ouvrage_de_reference>
      <j.0:Ouvrage_de_reference rdf:ID="Instance_39">
        <j.0:titre_ouvrage>DATABASE SYSTEM
IMPLEMENTATION</j.0:titre_ouvrage>
        <j.0:auteurs_ouvrage rdf:resource="#Instance_40"/>
        <j.0:auteurs_ouvrage rdf:resource="#Instance_41"/>
        <j.0:auteurs_ouvrage rdf:resource="#Instance_42"/>
      </j.0:Ouvrage_de_reference>
    </j.0:ouvrage_de_reference>
    <j.0:ouvrage_de_reference rdf:resource="#Instance_43"/>
    <j.0:informations_sur_exercices>Five written homework
assignments.</j.0:informations_sur_exercices>
    <j.0:informations_sur_exercices>No
programming.</j.0:informations_sur_exercices>
    <j.0:informations_sur_exercices>Also readings in
Textbook.</j.0:informations_sur_exercices>
    <j.0:notations_examen>Homeworks : 20%</j.0:notations_examen>
    <j.0:notations_examen>Midterm : 30%</j.0:notations_examen>
    <j.0:notations_examen>Final : 50%</j.0:notations_examen>
    <j.0:programme rdf:resource="#Instance_44"/>
    <j.0:programme rdf:resource="#Instance_45"/>
    <j.0:programme rdf:resource="#Instance_46"/>
    <j.0:programme rdf:resource="#Instance_47"/>
    <j.0:programme rdf:resource="#Instance_49"/>
    <j.0:programme rdf:resource="#Instance_50"/>
    <j.0:programme rdf:resource="#Instance_51"/>
    <j.0:programme rdf:resource="#Instance_52"/>
    <j.0:programme rdf:resource="#Instance_53"/>
    <j.0:programme rdf:resource="#Instance_54"/>
    <j.0:contenu_cours rdf:resource="#Instance_55"/>
    <j.0:contenu_cours rdf:resource="#Instance_56"/>
    <j.0:contenu_cours rdf:resource="#Instance_57"/>
    <j.0:plan_cours rdf:resource="#Instance_72"/>
    <j.0:plan_cours>
      <j.0:Ligne_plan rdf:ID="Instance_73"
         j.0:libelle_ligne_plan="Hardware">
        <j.0:id_ligne_plan>Chapitre 2</j.0:id_ligne_plan>
        <j.0:ressources_ligne_plan rdf:resource="#Instance_55"/>
        <j.0:concepts_ligne_plan rdf:resource="#Instance_86"/>
      </j.0:Ligne_plan>
    </j.0:plan_cours>
    <j.0:plan_cours rdf:resource="#Instance_74"/>
    <j.0:plan_cours>
      <j.0:Ligne_plan rdf:ID="Instance_75">
        <j.0:id_ligne_plan>Chapitre 4</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Indexing and
hashing</j.0:libelle_ligne_plan>
        <j.0:ressources_ligne_plan rdf:resource="#Instance_57"/>
```

```
        <j.0:concepts_ligne_plan rdf:resource="#Instance_90"/>
        <j.0:concepts_ligne_plan>
          <j.0:Concept rdf:ID="Instance_91"
             j.0:libelle_concept="INDEXING">
            <j.0:pere_concept rdf:resource="#Instance_90"/>
            <j.0:ressources_concept rdf:resource="#Instance_57"/>
          </j.0:Concept>
        </j.0:concepts_ligne_plan>
      </j.0:Ligne_plan>
    </j.0:plan_cours>
    <j.0:plan_cours>
      <j.0:Support_Cours rdf:ID="Instance_76"
         j.0:format=".PPT"
         rdf:type="#Ligne_plan">
        <j.0:id_ligne_plan>Chapitre 5</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Indexing and
hashing</j.0:libelle_ligne_plan>
        <j.0:ressources_ligne_plan rdf:resource="#Instance_58"/>
        <j.0:concepts_ligne_plan rdf:resource="#Instance_90"/>
        <j.0:concepts_ligne_plan rdf:resource="#Instance_92"/>
      </j.0:Support_Cours>
    </j.0:plan_cours>
    <j.0:plan_cours>
      <j.0:Cours rdf:ID="Instance_77"
         rdf:type="#Ligne_plan">
        <j.0:informations_sur_le_cours>
          <rdf:Description rdf:ID="Instance_78">
            <rdf:type rdf:nodeID="A0"/>
          </rdf:Description>
        </j.0:informations_sur_le_cours>
        <j.0:contenu_cours>
          <rdf:Description rdf:ID="Instance_79">
            <rdf:type rdf:nodeID="A1"/>
          </rdf:Description>
        </j.0:contenu_cours>
        <j.0:id_ligne_plan>Chapitre 6</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Query
processing</j.0:libelle_ligne_plan>
        <j.0:ressources_ligne_plan rdf:resource="#Instance_59"/>
        <j.0:concepts_ligne_plan rdf:resource="#Instance_93"/>
      </j.0:Cours>
    </j.0:plan_cours>
    <j.0:plan_cours>
      <j.0:Cours rdf:ID="Instance_80"
         rdf:type="#Ligne_plan">
        <j.0:id_ligne_plan>Chapitre 7</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Query
processing</j.0:libelle_ligne_plan>
        <j.0:ressources_ligne_plan rdf:resource="#Instance_60"/>
        <j.0:concepts_ligne_plan rdf:resource="#Instance_93"/>
      </j.0:Cours>
    </j.0:plan_cours>
    <j.0:plan_cours>
      <j.0:Ligne_plan rdf:ID="Instance_81">
        <j.0:id_ligne_plan>Chapitre 8</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Crash recovery</j.0:libelle_ligne_plan>
```

```
        <j.0:concepts_ligne_plan>
          <j.0:Concept rdf:ID="Instance_94">
            <j.0:libelle_concept>CRASH RECOVERY</j.0:libelle_concept>
            <j.0:pere_concept rdf:resource="#Instance_84"/>
          </j.0:Concept>
        </j.0:concepts_ligne_plan>
      </j.0:Ligne_plan>
    </j.0:plan_cours>
    <j.0:plan_cours>
      <j.0:Ligne_plan rdf:ID="Instance_82">
        <j.0:id_ligne_plan>Chapitre 9</j.0:id_ligne_plan>
        <j.0:libelle_ligne_plan>Concurrency
control</j.0:libelle_ligne_plan>
        <j.0:concepts_ligne_plan>
          <j.0:Concept rdf:ID="Instance_95">
            <j.0:libelle_concept>CONCURRENCY
CONTROL</j.0:libelle_concept>
            <j.0:pere_concept rdf:resource="#Instance_84"/>
          </j.0:Concept>
        </j.0:concepts_ligne_plan>
      </j.0:Ligne_plan>
    </j.0:plan_cours>
    <j.0:contenu_cours rdf:resource="#Instance_58"/>
    <j.0:contenu_cours rdf:resource="#Instance_59"/>
    <j.0:contenu_cours rdf:resource="#Instance_60"/>
    <j.0:universite>
      <j.0:Universite rdf:ID="Instance_97">
        <j.0:id_universite>STANFORD UNIVERSITY</j.0:id_universite>
      </j.0:Universite>
    </j.0:universite>
    <j.0:discipline>
      <j.0:Discipline rdf:ID="Instance_96">
        <j.0:id_discipline>COMPUTER SCIENCE</j.0:id_discipline>
      </j.0:Discipline>
    </j.0:discipline>
    <j.0:informations_sur_le_cours rdf:resource="#Instance_36"/>
    <j.0:informations_sur_le_cours rdf:resource="#Instance_37"/>
  </j.0:Cours>
  <j.0:Cours rdf:ID="Instance_20"/>
  <j.0:Ligne_plan rdf:ID="Instance_83"/>
  <j.0:Ouvrage_de_reference rdf:ID="Instance_16"/>
  <j.0:Ontologie_globale rdf:ID="Instance_101"/>
  <rdf:List rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#nil"/>
  <j.0:Prerequis rdf:ID="Instance_28"/>
  <j.0:Examen rdf:ID="Instance_15"/>
  <j.0:Cours rdf:ID="Instance_34"/>
  <j.0:Corrige rdf:ID="Instance_70"
     j.0:id_contenu="Corrige4"
     j.0:format="AUTRES"/>
  <j.0:Exercice rdf:ID="Instance_66"
     j.0:id_contenu="Exercice4"
     j.0:format=".PDF">
    <j.0:auteur_ressource rdf:resource="#Instance_36"/>
  </j.0:Exercice>
  <j.0:Cours rdf:ID="Instance_31"/>
  <j.0:Auteur rdf:ID="Instance_17"/>
```

```
  <j.0:Ontologie_globale rdf:ID="Instance_103"/>
  <rdf:Description rdf:ID="Instance_12">
    <rdf:type rdf:nodeID="A0"/>
  </rdf:Description>
</rdf:RDF>
```

ANNEXE C : DESCRIPTION DES META DONNEES AVEC OWL

1 Description locale des classes

1.1 Classe Personnel

```
<owl:Class rdf:ID="Personnel" />
<owl:DatatypeProperty rdf:ID="Nom&Prénom">
    <rdfs:domain rdf:resource="#Personnel" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="e-mail">
    <rdfs:domain rdf:resource="#Personnel" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Téléphone">
    <rdfs:domain rdf:resource="#Personnel" />
    <rdfs:range rdf:resource="&xsd;PositiveInteger" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Bureau">
    <rdfs:domain rdf:resource="#Personnel" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

1.2 Classe Secrétaire

```
<owl:Class rdf:ID="Secretaire" />
    <rdfs:SubClassOf rdf:resource="#Personnel" />
</owl:Class>
```

1.3 Classe Enseignant

```
<owl:Class rdf:ID="Enseignant" />
    <rdfs:SubClassOf rdf:resource="#Personnel" />
</owl:Class>
<owl:DatatypeProperty rdf:ID="AdressePostale ">
    <rdfs:domain rdf:resource="#Enseignant" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="HeureBureau">
    <rdfs:domain rdf:resource="#Enseignant" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

1.4 Classe Ouvrage de reference

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

```
</owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="Auteurs">
    <rdfs:domain rdf:resource="# OuvrageDeReference " />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="Edition">
    <rdfs:domain rdf:resource="# OuvrageDeReference " />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="DateDeParution">
    <rdfs:domain rdf:resource="# OuvrageDeReference " />
    <rdfs:range rdf:resource="&xsd;date" />
  </owl:DatatypeProperty>
```

1.5 Classe LigneProgramme

```
  <owl:Class rdf:ID="LigneProgramme" />
- <owl:DatatypeProperty rdf:ID="Date">
    <rdfs:domain rdf:resource="#LigneProgramme" />
    <rdfs:range rdf:resource="&xsd;date" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="NuméroDeChapitre">
    <rdfs:domain rdf:resource="#LigneProgramme" />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="TitreDeChapitre">
    <rdfs:domain rdf:resource="#LigneProgramme" />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
```

1.6 Classe Examen

```
  <owl:Class rdf:ID="Examen" />
- <owl:DatatypeProperty rdf:ID="IDExamen ">
    <rdfs:domain rdf:resource="#Examen" />
    <rdfs:range rdf:resource="&xsd;anyURI" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="Format">
    <rdfs:domain rdf:resource="#Examen" />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
```

1.7 Classe Exercice

```
  <owl:Class rdf:ID="Exercice" />
- <owl:DatatypeProperty rdf:ID="IDExercice">
    <rdfs:domain rdf:resource="#Exercice" />
    <rdfs:range rdf:resource="&xsd;anyURI" />
  </owl:DatatypeProperty>
- <owl:DatatypeProperty rdf:ID="Format">
    <rdfs:domain rdf:resource="#Exercice" />
    <rdfs:range rdf:resource="&xsd;string" />
  </owl:DatatypeProperty>
```

1.8 Classe Corrigé

```
<owl:Class rdf:ID="Corrige" />
<owl:DatatypeProperty rdf:ID="IDCorrige ">
    <rdfs:domain rdf:resource="#Corrige" />
    <rdfs:range rdf:resource="&xsd;anyURI" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Format">
    <rdfs:domain rdf:resource="#Corrige" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

1.9 Classe FAQ

```
<owl:Class rdf:ID="FAQ" />
<owl:DatatypeProperty rdf:ID="IDFAQ">
    <rdfs:domain rdf:resource="#FAQ" />
    <rdfs:range rdf:resource="&xsd;anyURI" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Format">
    <rdfs:domain rdf:resource="#FAQ" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

1.10 Classe SupportCours

```
<owl:Class rdf:ID="SupportCours" />
<owl:DatatypeProperty rdf:ID="IDSupportCours">
    <rdfs:domain rdf:resource="#SupportCours " />
    <rdfs:range rdf:resource="&xsd;anyURI" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Format">
    <rdfs:domain rdf:resource="#SupportCours " />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

1.11 Classe Ressource

```
<owl:Class rdf:ID="Ressource">
  <owl:oneOf rdf:parseType="Collection">
      <owl:Thing rdf:about="#SupportCours" />
      <owl:Thing rdf:about="#Exercice" />
      <owl:Thing rdf:about="#Corrige" />
      <owl:Thing rdf:about="#Examen" />
      <owl:Thing rdf:about="#FAQ" />
  </owl:oneOf>
</owl:Class>
```

1.12 Classe Contenu

```
<owl:Class rdf:ID="Contenu">
    <rdfs:comment>Commentaire</rdfs:comment>
  <owl:unionOf rdf:parseType="Collection">
```

```
        <owl:Class rdf:about="#SupportCours" />
        <owl:Class rdf:about="#Exercice" />
        <owl:Class rdf:about="#Corrige" />
        <owl:Class rdf:about="#Examen" />
        <owl:Class rdf:about="#FAQ" />
      </owl:unionOf>
    </owl:Class>
```

1.13 Classe LignePlan

```
    <owl:Class rdf:ID="LignePlan" />
  - <owl:DatatypeProperty rdf:ID="IDLignePlan">
      <rdfs:domain rdf:resource="#LignePlan" />
      <rdfs:range rdf:resource="&xsd;string" />
    </owl:DatatypeProperty>
  - <owl:DatatypeProperty rdf:ID="Libelle">
      <rdfs:domain rdf:resource="#LignePlan" />
      <rdfs:range rdf:resource="&xsd;string" />
    </owl:DatatypeProperty>
```

1.14 Classe Concept

```
    <owl:Class rdf:ID="Concept" />
  - <owl:DatatypeProperty rdf:ID="Libelle">
      <rdfs:domain rdf:resource="#Concept" />
      <rdfs:range rdf:resource="&xsd;string" />
    </owl:DatatypeProperty>
```

1.15 Classe Université

```
    <owl:Class rdf:ID="Universite" />
  - <owl:DatatypeProperty rdf:ID="IDUniversite">
      <rdfs:domain rdf:resource="#Universite" />
      <rdfs:range rdf:resource="&xsd;string" />
    </owl:DatatypeProperty>
```

1.16 Classe Discipline

```
    <owl:Class rdf:ID="Discipline" />
  - <owl:DatatypeProperty rdf:ID="IDDiscipline">
      <rdfs:domain rdf:resource="#Discipline" />
      <rdfs:range rdf:resource="&xsd;string" />
    </owl:DatatypeProperty>
```

1.17 Classe Niveau

```
  - <owl:Class rdf:ID="Niveau">
    - <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Licence" />
        <owl:Thing rdf:about="#Master" />
        <owl:Thing rdf:about="#Doctorat" />
      </owl:oneOf>
```

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

1.18 Classe Cours

```
<owl:Class rdf:ID="Cours" />
<owl:DatatypeProperty rdf:ID="IDCours">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Themes">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Annonces">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Prerequis">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="EmploiDuTemps">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="InfosSurLesExercices">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="NotationsExamen">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="SiteWeb">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;anyURI" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="CommunicationEquipe">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Charte">
    <rdfs:domain rdf:resource="#Cours" />
    <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
```

2 Description globale

```
<owl:Class rdf:ID="Cours">
    <rdfs:subClassOf>
        <owl:Restriction>
            <owl:onProperty rdf:resource="#AppartientUniversite" />
```

```
        <owl:cardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
            lity>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#AppartientUniversite" />
        <owl:allValuesFrom rdf:resource="#Universite" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#AppartientDiscipline" />
        <owl:cardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
            lity>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#AppartientDiscipline" />
        <owl:allValuesFrom rdf:resource="#Discipline" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#AppartientNiveau" />
        <owl:cardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
            lity>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#AppartientNiveau" />
        <owl:allValuesFrom rdf:resource="#Niveau" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourEnseignants" />
        <owl:minCardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
            dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourEnseignants" />
        <owl:allValuesFrom rdf:resource="#Enseignant" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
```

```
- <owl:Restriction>
    <owl:onProperty rdf:resource="#APourSecretaire" />
    <owl:cardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
      lity>
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourSecretaire" />
    <owl:allValuesFrom rdf:resource="#Secretaire" />
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourOuvrageDeRefrence"
      />
    <owl:minCardinality
      rdf:datatype="&xsd;nonNegativeInteger">0</owl:minCar
      dinality>
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourOuvrageDeRefrence"
      />
    <owl:allValuesFrom rdf:resource="#OuvrageDeRefrence" />
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourProgramme" />
    <owl:minCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
      dinality>
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourProgramme" />
    <owl:allValuesFrom rdf:resource="#LigneProgramme" />
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourContenu" />
    <owl:minCardinality
      rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
      dinality>
  </owl:Restriction>
</rdfs:subClassOf>
- <rdfs:subClassOf>
  - <owl:Restriction>
    <owl:onProperty rdf:resource="#APourContenu" />
```

```
            <owl:allValuesFrom rdf:resource="#Contenu" />
        </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
            <owl:onProperty rdf:resource="#APourPlan" />
            <owl:minCardinality
                rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
                dinality>
        </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
            <owl:onProperty rdf:resource="#APourPlan" />
            <owl:allValuesFrom rdf:resource="#LignePlan" />
        </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>


- <owl:Class rdf:ID="Niveau">
  - <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Licence" />
        <owl:Thing rdf:about="#Master" />
        < owl:Thing rdf:about="#Doctorat" />
    </owl:oneOf>
</owl:Class>


<owl:Class rdf:about="Licence" />
    <owl:disjointWith rdf:resource="#Master" />
    <owl:disjointWith rdf:resource="#Doctorat" />
</owl:Class>


<owl:Class rdf:about=" Master" />
    <owl:disjointWith rdf:resource="#Doctorat" />
</owl:Class>


- <owl:Class rdf:ID="Licence">
  - <rdfs:subClassOf>
    - <owl:Restriction>
            <owl:onProperty rdf:resource="#APourSemestre" />
            <owl:cardinality
                rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
                lity>
        </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
            <owl:onProperty rdf:resource="#APourSemestre" />
        - <owl:allValuesFrom>
            - <owl:Class>
                - <owl:oneOf rdf:parseType="Collection">
```

```
                    <owl:Thing rdf:about="#Semestre1" />
                    <owl:Thing rdf:about="#Semestre2" />
                    <owl:Thing rdf:about="#Semestre3" />
                    <owl:Thing rdf:about="#Semestre4" />
                    <owl:Thing rdf:about="#Semestre5" />
                    <owl:Thing rdf:about="#Semestre6" />
                </owl:oneOf>
            </owl:Class>
        </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

- <owl:Class rdf:ID="Master">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSemestre" />
        <owl:cardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
            lity>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSemestre" />
      - <owl:allValuesFrom>
        - <owl:Class>
          - <owl:oneOf rdf:parseType="Collection">
              <owl:Thing rdf:about="#Semestre1" />
              <owl:Thing rdf:about="#Semestre2" />
              <owl:Thing rdf:about="#Semestre3" />
              <owl:Thing rdf:about="#Semestre4" />
            </owl:oneOf>
          </owl:Class>
        </owl:allValuesFrom>
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

- <owl:Class rdf:ID="Doctorat">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSemestre" />
        <owl:cardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
            lity>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSemestre" />
      - <owl:allValuesFrom>
        - <owl:Class>
```

```
            - <owl:oneOf rdf:parseType="Collection">
                <owl:Thing rdf:about="#Semestre1" />
                <owl:Thing rdf:about="#Semestre2" />
                <owl:Thing rdf:about="#Semestre3" />
                <owl:Thing rdf:about="#Semestre4" />
                <owl:Thing rdf:about="#Semestre5" />
                <owl:Thing rdf:about="#Semestre6" />
              </owl:oneOf>
            </owl:Class>
          </owl:allValuesFrom>
        </owl:Restriction>
      </rdfs:subClassOf>
    </owl:Class>

- <owl:Class rdf:ID="LigneProgramme">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSuivant" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourSuivant" />
        <owl:allValuesFrom rdf:resource="#LigneProgramme" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourRessource" />
        <owl:minCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
          dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourRessource" />
        <owl:allValuesFrom rdf:resource="#Ressource" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

- <owl:Class rdf:ID="Ressource">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourAuteur" />
        <owl:cardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:cardina
          lity>
      </owl:Restriction>
```

```
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourAuteur" />
        <owl:allValuesFrom rdf:resource="#Enseignant" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

**********Description de la hiérarchie**********
- <owl:Class rdf:ID="Concept">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourPere" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourPere" />
        <owl:allValuesFrom rdf:resource="#Concept" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ContrainteDePrecedence" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ContrainteDePrecedence" />
        <owl:allValuesFrom rdf:resource="#Concept" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ConcerneRessources" />
        <owl:minCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
          dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ConcerneRessources" />
        <owl:allValuesFrom rdf:resource="#Ressource" />
      </owl:Restriction>
    </rdfs:subClassOf>
```

```
</owl:Class>

**********Plan du cours**********
- <owl:Class rdf:ID="OntologieGlobale">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#SeComposeDe" />
        <owl:minCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
          dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#SeComposeDe" />
        <owl:allValuesFrom rdf:resource="#Concept" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

- <owl:Class rdf:ID="LignePlan">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#EstReferenceePar" />
        <owl:minCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
          dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#EstReferenceePar" />
        <owl:allValuesFrom rdf:resource="#Ressource" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourPere" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourPere" />
        <owl:allValuesFrom rdf:resource="#LignePlan" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourFilsGauche" />
```

```
            <owl:maxCardinality
                rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
                rdinality>
        </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourFilsGauche" />
        <owl:allValuesFrom rdf:resource="#LignePlan" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ConcerneConcepts" />
        <owl:minCardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
            dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#ConcerneConcepts" />
        <owl:allValuesFrom rdf:resource="#Concept" />
      </owl:Restriction>
    </rdfs:subClassOf>
</owl:Class>


- <owl:Class rdf:ID="Ressource">
  - <owl:oneOf rdf:parseType="Collection">
      <owl:Thing rdf:about="#SupportCours" />
      <owl:Thing rdf:about="#Exercice" />
      <owl:Thing rdf:about="#Corrige" />
      <owl:Thing rdf:about="#Examen" />
      <owl:Thing rdf:about="#FAQ" />
    </owl:oneOf>
</owl:Class>


- <owl:Class rdf:ID="SupportCours ">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourExamen" />
        <owl:minCardinality
            rdf:datatype="&xsd;nonNegativeInteger">1</owl:minCar
            dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourExamen" />
        <owl:allValuesFrom rdf:resource="#Examen" />
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
```

```
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourExercice" />
        <owl:minCardinality
          rdf:datatype="&xsd;nonNegativeInteger">0</owl:minCar
          dinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourExercice" />
        <owl:allValuesFrom rdf:resource="#Exercice" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

- <owl:Class rdf:ID="Exercice">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourCorrige" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourCorrige" />
        <owl:allValuesFrom rdf:resource="#Corrige" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

- <owl:Class rdf:ID="Examen">
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourCorrige" />
        <owl:maxCardinality
          rdf:datatype="&xsd;nonNegativeInteger">1</owl:maxCa
          rdinality>
      </owl:Restriction>
    </rdfs:subClassOf>
  - <rdfs:subClassOf>
    - <owl:Restriction>
        <owl:onProperty rdf:resource="#APourCorrige" />
        <owl:allValuesFrom rdf:resource="#Corrige" />
      </owl:Restriction>
    </rdfs:subClassOf>
  </owl:Class>

  <owl:Class rdf:ID="Secretaire" />
    <rdfs:SubClassOf rdf:resource="#Personnel" />
    <owl:disjointWith rdf:resource="#Enseignant" />
```

```
</owl:Class>

<owl:Class rdf:ID="Enseignant" />
   <rdfs:SubClassOf rdf:resource="#Personnel" />
</owl:Class>
<owl:DatatypeProperty rdf:ID="AdressePostale ">
   <rdfs:domain rdf:resource="#Enseignant" />
   <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="HeureBureau">
   <rdfs:domain rdf:resource="#Enseignant" />
   <rdfs:range rdf:resource="&xsd;string" />
</owl:DatatypeProperty>

<owl:Class rdf:about="SupportCours" />
   <owl:disjointWith rdf:resource="#Exercice" />
   <owl:disjointWith rdf:resource="#Corrige" />
   <owl:disjointWith rdf:resource="#Examen" />
   <owl:disjointWith rdf:resource="#FAQ" />
</owl:Class>

<owl:Class rdf:about="Exercice" />
   <owl:disjointWith rdf:resource="#Corrige" />
   <owl:disjointWith rdf:resource="#Examen" />
   <owl:disjointWith rdf:resource="#FAQ" />
</owl:Class>

<owl:Class rdf:about="Corrige" />
   <owl:disjointWith rdf:resource="#Examen" />
   <owl:disjointWith rdf:resource="#FAQ" />
</owl:Class>

<owl:Class rdf:about="Examen" />
   <owl:disjointWith rdf:resource="#FAQ" />
</owl:Class>
```

ANNEXE D : OWL PLUGIN

1 OWL Plugin

1.1 Panorama

Le Plugin OWL est une extension de l'outil de modélisation de connaissances et d'édition d'ontologie Protégé. Développé par Stanford Medical Informatics (Informatiques Médicales de Stanford), il permet d'éditer des ontologies dans le langage d'ontologie du web (OWL), le prochain langage standard du web sémantique du W3C. En particulier, on peut utiliser le plugin OWL pour :

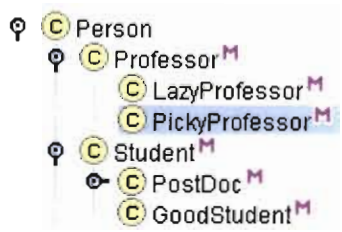
- Charger et enregistrer les ontologies en divers formats de fichier OWL (RDF, N3, N-TRIPLE),
- Editer les classes nommées OWL, les restrictions et les descriptions de classes arbitraires,
- Editer les propriétés et les individus,
- Accéder aux composants d'inférence de logiques de description tels que les classifiers (partiellement implémentés),
- Utiliser n'importe quel plugin existant pour Protégé, tels que les composants de visualisation et de raisonnement,
- Intégrer les modules arbitraires qui sont basés sur la bibliothèque OWL Jena de HP Lab,
- Intégrer nos propres extensions construites sur Protégé ou Jena.

1.2 Bref aperçu : le langage d'ontologie du web OWL dans Protégé

Une ontologie est un modèle conceptuel d'un domaine. Les ontologies sont construites pour divers buts tels que la modélisation de connaissances, les agents intelligents et le web

sémantique. Les langages d'ontologie sont employés pour spécifier une ontologie. La plupart des langages d'ontologie fournissent des moyens pour spécifier les classes (concepts), les relations, les attributs, les contraintes et les instances. OWL est un très puissant nouveau langage d'ontologie du web développé par le Word Wide Web Consortium (W3C) en support à leur vision du web sémantique. Les autres langages d'ontologie sont OKBC (le langage originellement supporté par Protégé et (dans une moindre mesure) UML. Les éléments de modélisation de base de OWL et leur représentation dans Protégé sont traités dans les sections suivantes.

1.2.1 Named classes

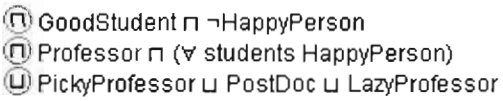


Les classes nommées représentent les concepts du domaines. Les classes peuvent avoir des propriétés pour décrire les attributs de la classe et leurs relations avec d'autres classes. Les classes peuvent avoir des individus (instances). Les classes peuvent être organisées en hiérarchie d'héritage (sous-classe/super classe), et il est possible d'énoncer que deux classes sont équivalentes ou disjointes. Les classes peuvent également être définies aux moyens de restrictions sur leurs attributs.

1.2.2 Restrictions

Les restrictions représentent les contraintes sur les valeurs valides d'une certaine propriété. Par exemple, la première restriction énonce que tous les students doivent être PostDoc, la seconde énonce qu'il doit avoir au moins un student. Les restrictions définissent une classe anonyme, i.e la classe de tous les individus qui ont au moins un student.

1.2.3 Expressions complexes de classe



Les classes peuvent être définies par combinaison logique d'autres classes. Par exemple, on peut définir l'intersection de tous les GoodStudents et le complément de HappyPersons, ou la classe de tous les Professors ayant seulement des HappyPersons comme students. On peut utiliser ces expressions pour définir des super classes, des classes équivalentes et des classes disjointes.

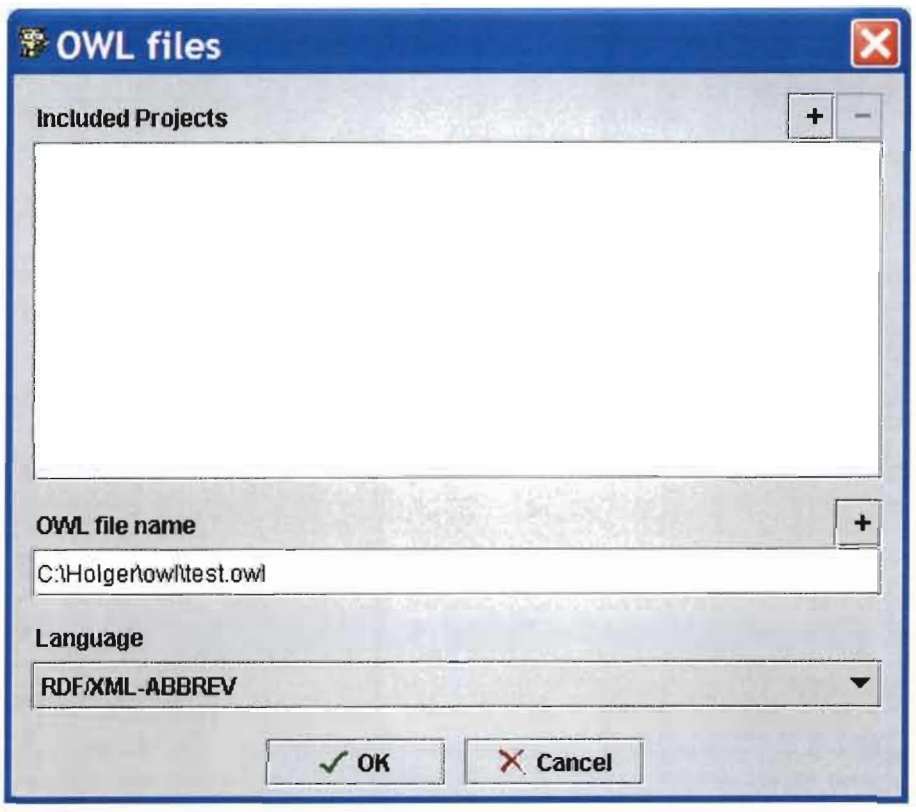
1.3 Conception d'ontologie OWL avec Protégé

1.3.1 Création d'une nouvelle ontologie OWL

Au démarrage du système, le dialogue d'entrée demande si on veut exécuter un projet existant ou créer une ontologie vide OWL. Sélectionner "OWL files" sous "New project"pour créer une ontologie vide OWL. Optionnellement, on peut utiliser l'item "New..." à partir du menu "Project". L'ontologie tout juste créée contiendra quelques classes systèmes (telles que :THING et :META-CLASS). Ces classes définissent le méta modèle de OWL en termes de classes Protégé.

1.3.2 Chargement d'un fichier OWL existant

Si on veut charger un fichier OWL existant (en formats RDF, N3 ou N-TRIPLE) dans Protégé alors on utilise la fonction "Import..." à partir du menu "Project". Sélectionner OWL File comme type de fichier. Ceci donnera le dialogue suivant.

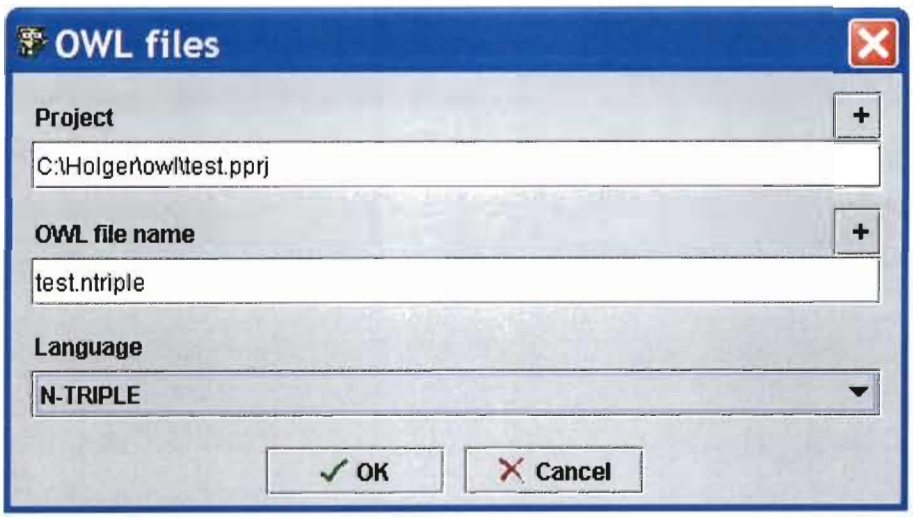


Utiliser le bouton + à côté du champ texte "OWL file name" pour sélectionner le fichier OWL qu'on souhaite importer. Utiliser le combo box "Language" ci-dessous pour

sélectionner le format du fichier OWL. La plupart des fichiers OWL sont enregistrés dans le langage standard RDF/XML-ABBREV.

1.3.3 Enregistrement d'une ontologie OWL

Si on enregistre un projet OWL, le système créera deux fichiers : un fichier Protégé (pprj) et un fichier OWL. Utiliser "Save as..." dans le menu Project pour choisir les noms de fichiers et le langage.



Si on veut enregistrer un projet existant Protégé développé avec un autre backend (i.e. le format natif CLIPS ou UML), on utilise le menu Save in Format..." et on sélectionne OWL comme format. Cela génèrera un fichier OWL, bien qu'il pourrait y avoir quelques problèmes selon les dispositifs utilisés. Par exemple, l'ontologie Newspaper distribuée avec les dispositifs spécifiques de Protégé tel que la classe :RELATION qui n'a pas encore été implémentée pour OWL.

1.3.4 Travailler avec l'interface utilisateur de Protégé

L'interface utilisateur de Protégé est divisé en plusieurs onglets. Dans la configuration par défaut, Protégé fournit les onglets suivants :

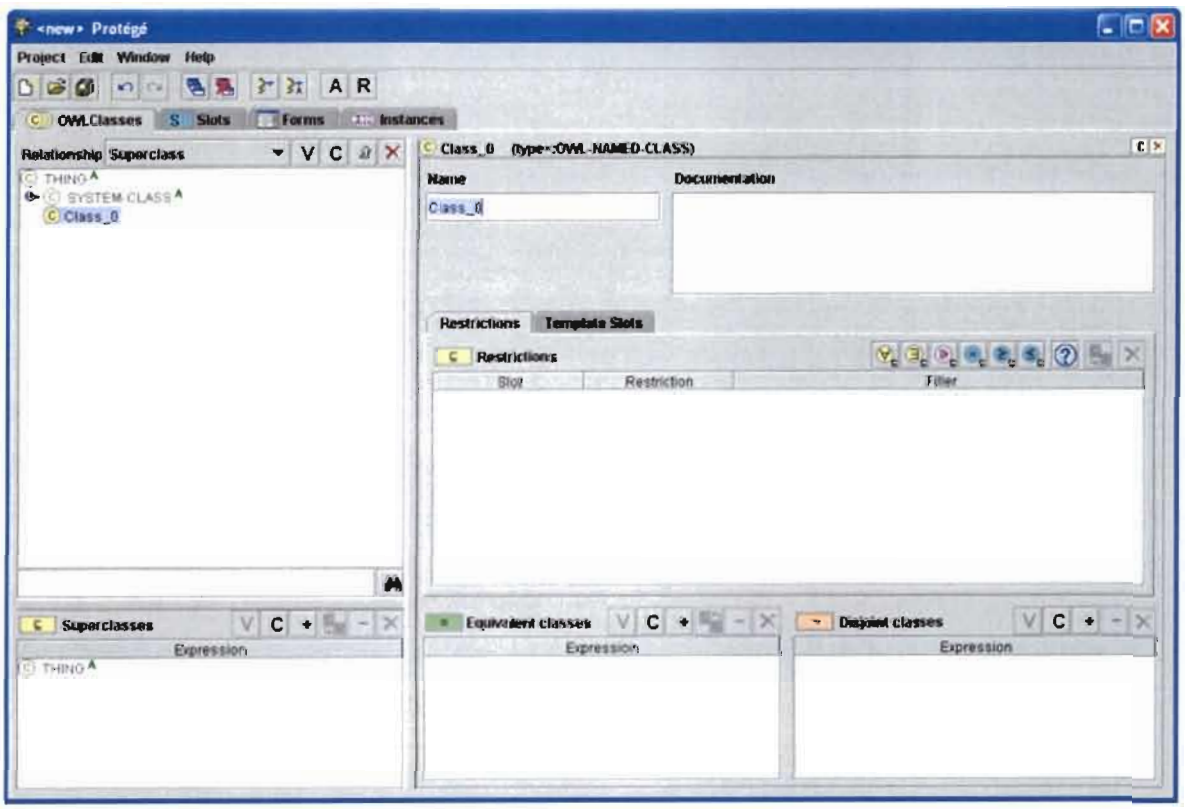
- OWL Classes : la page principale utilisée pour définir les classes, les restrictions, les relations, etc.

- Slots : fournissent une vue des propriétés
- Forms : permet de personnaliser les formulaires utilisées pour acquérir les individus
- Instances : fournissent un accès aux individus dans l'ontologie

On peut télécharger des plugins additionnels à partir du site web de Protégé et ensuite afficher d'autres onglets en utilisant le menu "Configure...".

1.3.5 Définition de classes nommées simples

Dans l'onglet OWL Classes, on peut parcourir la taxonomie de classe (hiérarchie d'héritage) dans l'arbre dans le panel droit. Sélectionner :THING et cliquer sur le bouton C pour créer une nouvelle classe OWL. Cette nouvelle classe aura initialement un nom par défaut comme Class_0 mais on peut la renommer.



Les attributs de la nouvelle classe peuvent être édités dans le formulaire à droite. Ici on peut spécifier le nom de la classe (les noms doivent commencer par une lettre majuscule et ne doivent pas contenir d'espaces ou de caractères spéciaux) et peut être, rentrer de la documentation (commentaires). On peut également définir l'hiérarchie de classe, restrictions, classes équivalentes et classes disjointes (voir ci-dessous).

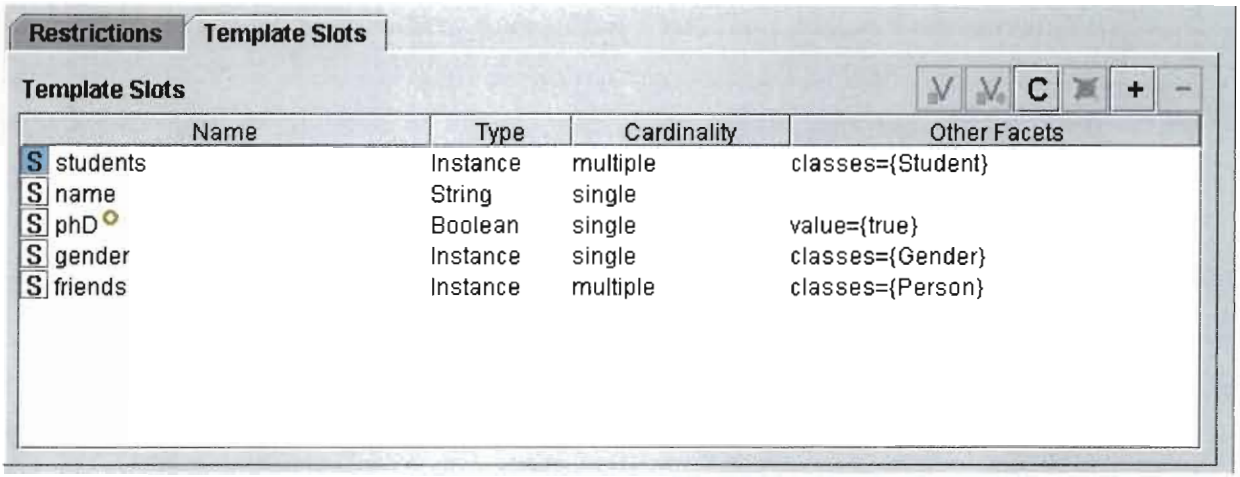
1.3.6 Définition d'une taxonomie de classe / hiérarchie d'héritage

L'arbre à gauche de l'onglet OWL Classes affiche les classes nommées dans l'ontologie. Au-dessous de celui-ci, on peut voir un panel qui énumère les super classes de la classe sélectionnée. Si on veut ajouter une sous classe d'une classe existante, on sélectionne la classe existante et on clique sur le bouton C. On peut changer la hiérarchie d'héritage en faisant cliquer-glisser de noms de classe. Par ailleurs, on peut modifier la structure à partir du panel Superclasses, par exemple en ajoutant des classes existantes avec le bouton +.

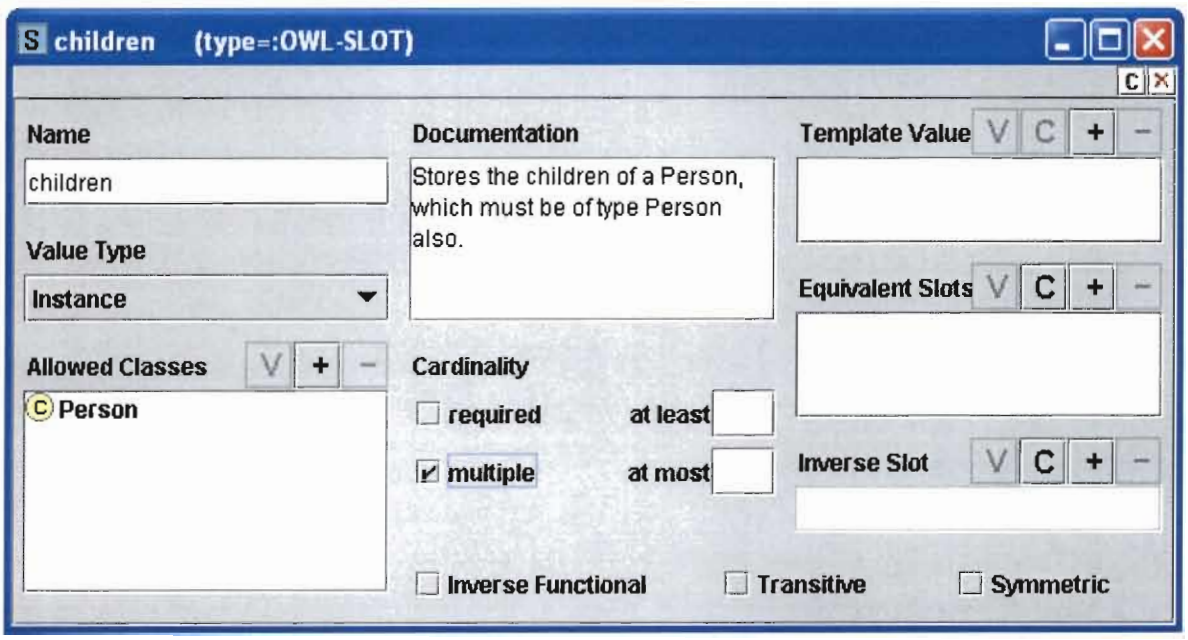
Toutes les classes utilisateurs définies dans le plugin OWL sont basées sur de nouveaux méta modèles de classes tels que : OWL-NAMED-CLASS. Pour le moment, il est conseillé de s'abstenir d'employer d'autres méta classes (ne pas employer "Create subclass from metaclass..." pour le moment.

1.3.7 Définition des propriétés / slots

En OWL, les propriétés sont employées pour définir les attributs des individus d'une classe donnée. Dans Protégé, les propriétés sont traditionnellement appelées "slots". Si on veut définir les propriétés de la classe actuellement éditée, on bascule sur l'onglet "Template Slots" dans l'éditeur de classes.



Chaque ligne de cette table représente une propriété d'une classe donnée. Les propriétés / slots avec un S blanc à leur gauche sont héritées d'une super classe. Les propriétés avec un S bleu sont introduites par la classe (dans la terminologie OWL, elles ont la classe comme domaine). On utilise le bouton C pour ajouter une propriété à la classe. Ceci ouvrira un formulaire slot comme ce qui suit.



Ce formulaire peut être également affiché plus tard si on veut changer un slot. Soit on fait un double -clique sur un slot dans la table template slots, on utilise le bouton V pour le slot sélectionné, ou on va sur la table slots dans la fenêtre principale de Protégé pour afficher et

éditer tous les slots existants. Il y a deux façons pour éditer un slot. On peut soit ouvrir globalement le slot au niveau supérieur ou localement pour une classe donnée. Ceci signifie qu'on peut définir le comportement global par défaut d'un slot (i.e., en général le slot children a le type de valeur Person), mais on peut le surcharger pour une certaine classe (i.e. pour la classe ParentOfDaughters, le slot children a le type de valeur FemalePerson). Cette distinction est rendue explicite par les boutons V au-dessus de la table des slots.

Name: Spécifie le nom de la propriété / slot. Il doit commencer par une lettre minuscule et ne doit pas comporter d'espace ni de caractères spéciaux. Il y a différentes conventions de nommage des propriétés. La communauté des logiques de description utilise souvent des noms comme hasChildren ou hasSpouse, alors que dans Protégé, on omet "has" et on utilise des noms comme "children" and "spouse". Holger Knublauch recommande le dernier format parce qu'il est plus court et les rend les expressions plus lisible.

Value type: Le type de valeur (en OWL: le range) est soit primitif (Boolean, Integer, Float, String) ou non-primitif (Instance). Protégé supporte les types de valeur Symbol and Class, mais ils ne devraient pas être utilisés dans la version actuelle. Si on choisit Instance comme type de valeur, on doit utiliser le bouton + pour ajouter les classes qui représentent le range de la propriété.

Template Value: Est seulement significatif lorsqu'on édite le slot au niveau classe. Ici il représente la valeur d'une restriction hasValue (voir ci-dessous).

Equivalent Slots: Spécifie les slots qui sont équivalents. L'équivalence de slot peut être utilisée pour énoncer que des concepts de différentes ontologies (included files) ont la même signification.

Inverse Slot: Spécifie le slot inverse. Actuellement, seul un inverse peut être défini pour chaque slot. Si deux slots sont inverses, alors leurs valeurs sont synchronisés. Par exemple, on peut définir que le slot inverse de "children" est "parent". Si on ajoute un child à une personne alors ce child aura la personne comme parent.

Inverse Functional: Si un slot est inverse functional alors toutes ces valeurs sont globalement unique[6].

Transitive: Par exemple, la propriété subRegionOf entre regions est transitive.

Symmetric: Un exemple de propriété symétrique est la relation friendOf:

Il faut noter l'absence de widgets pour des dispositifs tells que les valeurs par défaut, les contraintes et des valeurs numériques minimum et maximum. Ceux-ci n'ont pas de correspondances directes en OWL et seront implémentés plus tard.

1.3.8 Définition des restrictions

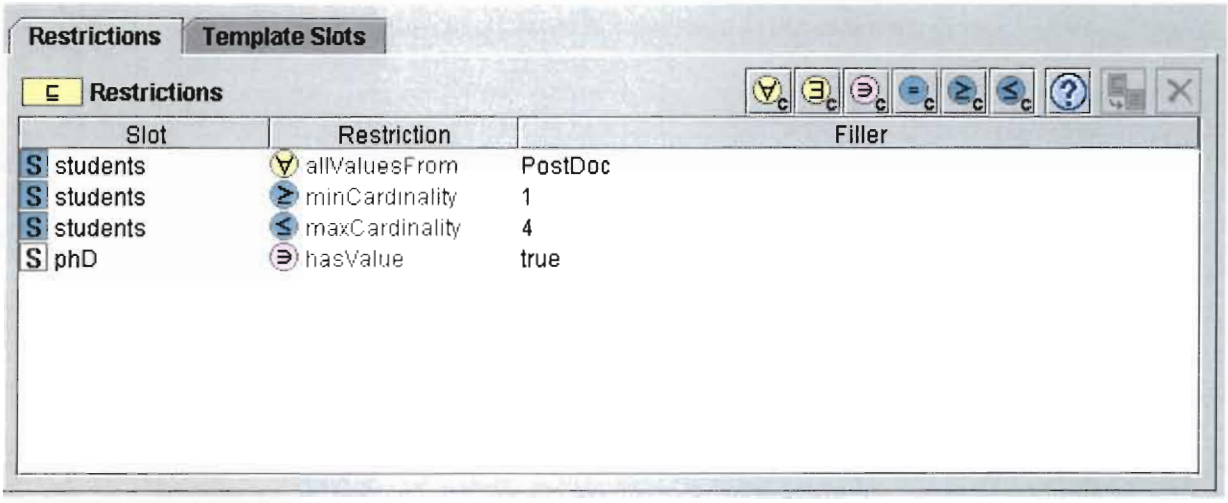
Les restrictions contraignent les valeurs d'une propriété sur une classe donnée. Par exemple, on peut utiliser une restriction pour exprimer que toutes FemalePersons ont "female" comme leur gender. On peut exprimer également que chaque membre de la classe Company doit avoir au moins un employee. OWL offre six types de restrictions.

- allValuesFrom: Spécifie que toutes les valeurs d'une propriété doivent avoir un certain type
- someValuesFrom: Spécifie qu'au moins une valeur d'une propriété doit avoir un certain type
- hasValue: Spécifie qu'une propriété doit avoir une certaine valeur
- minCardinality: Spécifie qu'une propriété doit avoir au moins X valeurs
- maxCardinality: Spécifie qu'une propriété doit avoir au plus X valeurs
- cardinality: Spécifie à la fois les cardinalités minimum et maximum

Techniquement, une restriction sur une classe est une super classe anonyme de la classe . Dans la terminologie Protégé, les restrictions sont la technologie OWL pour définir les

[6] Examples include social security numbers or primary keys in data bases

surcharges de slot. Ceci signifie qu'on peut soit utiliser la forme de slot décrite ci-dessus, ou soit l'onglet Restrictions dans l'éditeur de classe pour éditer les restrictions :



Cette table montre chaque restriction dans une seule ligne. La table a les colonnes suivantes :

Slot: Donne le nom du slot qui est restreint, et si la restriction est héritée d'une super classe (S blanc).

Restriction: Montre le type de restriction, i.e. "allValuesFrom", "someValuesFrom", "minCardinality", "maxCardinality", "cardinality", or "hasValue".

Filler: Montre le filler qui est soit un type de donnée, une description de classe, ou une valeur de cardinalité.

Les restrictions peuvent être ajoutées et supprimées avec les boutons usuels dans le coin supérieur droit du widget. Il y a six boutons pour ajouter des restrictions, reflétant les six types de restriction OWL

1.3.9 Edition des expressions OWL

L'une des principales différences entre le traditionnel modèle de connaissances Protégé et OWL est que OWL permet de spécifier des expressions arbitraires de classe au lieu de classes nommées. Ces expressions peuvent combiner logiquement des énoncés de classes et restrictions, en utilisant des opérateurs tels que intersectionOf (and), unionOf (or) et

complementOf (not). Plus particulièrement, on peut définir une classe par énumération de ses individus. De sorte à éditer ces expressions de classe dans le Plugin OWL de Protégé, des widgets et onglets personnalisables ont été développés. On peut éditer ces expressions pour des restrictions, super classes, classes équivalentes et classes disjointes. Plus tard, on pourra éditer des domaines et ranges de propriété arbitraire.

1.3.9.1 Syntaxe d'expression

Une question cruciale pour le nouveau GUI est comment afficher les expressions de classe OWL en Protégé. La syntaxe formel OWL est basée sur RDF qui est difficilement lisible pour des utilisateurs moyens. Il y a une syntaxe abstraite de OWL qui est beaucoup plus simple mais trop verbeux. C'est pourquoi la syntaxe compacte suivante a été définie. Elle est basée sur les symboles traditionnels de la communauté de logiques de description.

OWL Element	Symbol	Key	Example
allValuesFrom	∀	*	∀ children Male
someValuesFrom	∃	?	∃ children Lawyer
hasValue	∋	$	rich ∋ true
cardinality	=	=	children = 3
minCardinality	≥	>	children ≥ 3
maxCardinality	≤	<	children ≤ 3
complementOf	¬	!	¬ Parent
intersectionOf	⊓	&	Human ⊓ Male
unionOf	⊔	|	Doctor ⊔ Lawyer
enumeration	{...}	{ }	{male female}

Cette syntaxe est utilisée pour exprimer et éditer des expressions complexes de classe. Son principal avantage est sa simplicité (compactness). La plupart des expressions tiennent facilement dans une simple ligne de texte, de sorte qu'elles puissent être affichées dans une table conventionnelle ou de vues de liste. Un inconvénient potentiel est que la syntaxe compacte se fonde sur les caractères spéciaux, de sorte que les utilisateurs doivent apprendre des clefs spéciales pour entrer des expressions (voir ci-dessous). La table ci-dessus peut être montrée dans Protégé dans une fenêtre supplémentaire à l'aide d'un nouveau bouton d'aide.

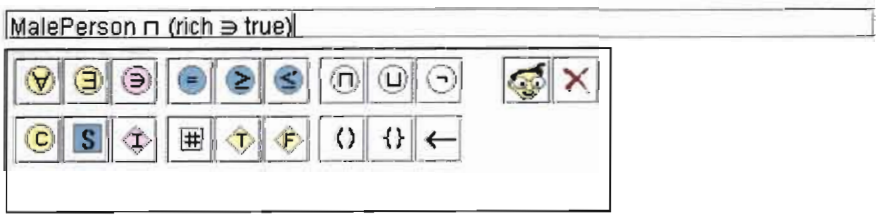
Le processus de construction du support OWL pour Protégé a commencé avec la syntaxe compacte. Selon le feedback utilisateur, les versions postérieures peuvent fournir des langages

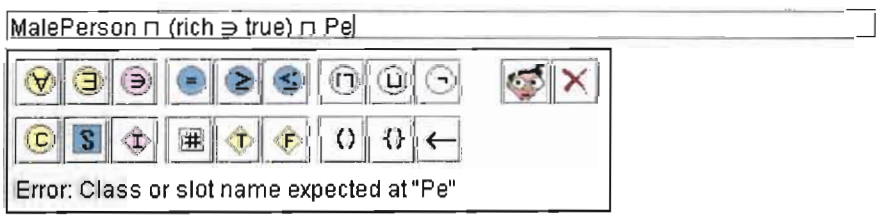
additionnels, tels que la syntaxe abstraite ou d'autres formats facilement lisibles. Celles-ci ont pu être activées par une option globale. Cependant, on se concentre sur une syntaxe simple d'abord, parce qu'elle est cruciale pour avoir un système exécutable avec la fonctionnalité essentielle disponible aussitôt que possible. On espère également pouvoir réutiliser des éditeurs d'expression tiers et d'autres widgets plus tard.

1.3.9.2 Editeur d'expression

Le Plugin OWL fournit un éditeur d'expression personnalisé pour des expressions de syntaxe compactes. Cet éditeur est utilisé partout où des expressions complexes de classe peuvent être écrites. Dans la plupart des widgets, on peut double-cliquer sur une expression pour lancer l'éditeur d'expression. L'éditeur affichera une fenêtre popup au-dessous de l'expression. Cette fenêtre fournit des raccourcis aux symboles spéciaux et affiche les messages d'erreur. Elle fournit les dispositifs suivants :

Vérification de syntaxe. La syntaxe de l'expression est vérifiée pendant la saisie. Si l'expression est correcte, le bouton "Protégé nerd" sourit. Si l'expression contient une erreur, le nerd aura un visage rouge (comme montré ci-dessous) et on ne peut pas assigner la valeur. Au lieu de cela, quand on frappe entrez, un message d'erreur est montré au fond de la fenêtre popup.

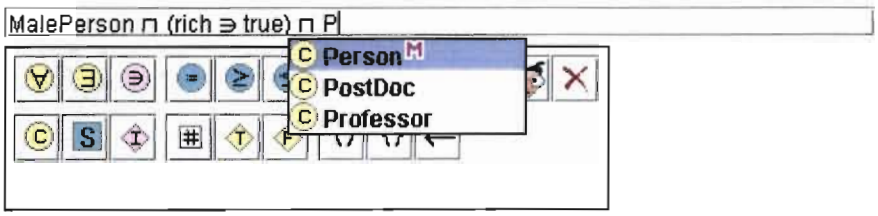




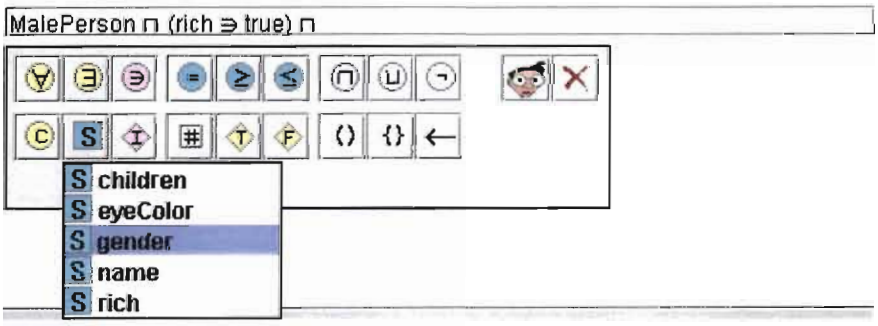
Remplacement de mot-clé. Puisque la syntaxe OWL se sert de symboles qui ne sont pas disponibles sur les claviers normaux, l'éditeur d'expression fournit des moyens pour écrire ces

symboles. D'abord, il y a un mapping direct entre les clés et les symboles. Cela signifie que si on veut entrer le symbole "and", on peut frapper & et le caractère spécial apparaîtra. En outre, certains mots-clés sont remplacés pendant la frappe. Par exemple, "allValuesFrom", "all", "only" sont tous remplacés par le symbole du quantificateur all.

Word completion. Si on tape les premières lettres d'un nom de classe, nom de propriété ou d'instance, on peut taper CTRL+Space pour compléter automatiquement le mot. S'il y a une seule extension possible d'un nom partiel, alors le système insérera automatiquement le reste du nom. Par exemple, on a besoin d'entrer "Per" et CTRL+Space et le système le complètera en "Person". S'il y a des noms multiples qui matchent le contexte courant, alors une liste déroulant est affichée au-dessous du curseur. L'utilisateur peut soit continuer la frappe ou sélectionner un nom de la liste.



Full mouse support. L'interface utilisateur est conçu pour supporter à la fois clavier et souris utilisateurs, i.e. tous les dispositifs de OWL peuvent être édités avec la souris uniquement. Par exemple, l'ajout d'un nom de classe dans des expressions OWL peut être réalisé en utilisant le bouton de classe dans la fenêtre popup de symbole, comme ci-dessous. Les éléments spéciaux du langage tels que les types de données (#), true (T) et false (F) sont également supportés aussi bien que l'espacement (flèche).



1.3.10 Edition de classes équivalentes, de super classes et de classes disjointes

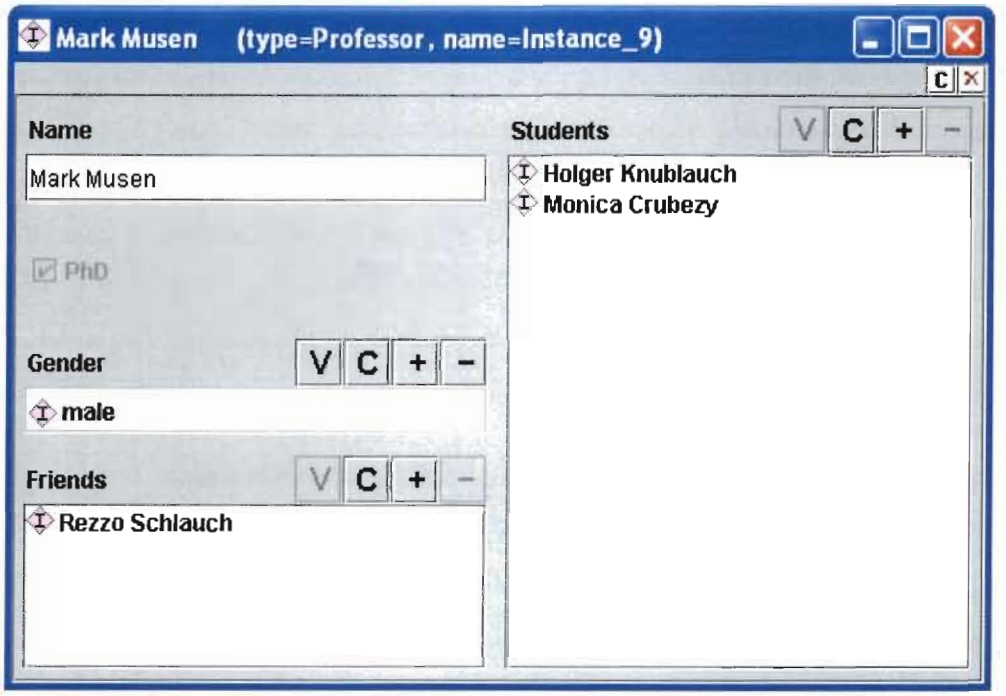
L'onglet OWL Classes fournit trois widgets additionnels comme montré dans le grand écran ci-dessus. Chacun de ces derniers abrite fondamentalement une liste dans laquelle chaque ligne représente une description de classe. On peut éditer ces descriptions à travers l'éditeur d'expression comme décrit ci-dessus. On peut également déplacer les définitions de classe entre les axiomes equivalentClass et subClassOf avec un simple clic de souris (ou glisser-relâcher plus tard). Notons que si on active une classe équivalente anonyme, elle pourrait apparaître dans la liste de super classes ou de la table de restriction, si c'est une restriction.

Les widgets fournissent également les boutons + et – pour ajouter et supprimer des classes nommées existantes. Par ailleurs, il y a les boutons C et X pour créer ou supprimer des expressions de classe anonyme. Les classes anonymes ont un bouton de suppression différent parce que leur cycle de vie dépend de la classe éditée, i.e. elles ne peuvent pas exister sans être assignées à une classe.

Notons que les utilisateurs ont uniquement des formulaires pour éditer des classes nommées OWL dans Protégé. S'ils souhaitent éditer des expressions de classe nommée (énumérations, et intersections) ils ont besoin de créer une classe nommée et d'entrer la définition de la classe comme une classe équivalente. Si on ouvre une ontologie OWL qui contient des expressions de classes complexes nommées, alors celles-ci sont converties en une classe nommée qui a l'expression comme classe équivalente anonyme. On ne devrait pas essayer de manipuler les classes anonymes en utilisant d'autres formulaires de Protégé.

1.3..11 Edition des instances et leurs formulaires

L'une des principales forces de Protégé est sa capacité à éditer des instances. Les instances sont les objets spécifiques d'une classe donnée, par exemple la Personne spécifique "Hans". En utilisant Protégé, on a tout juste besoin de définir les classes et leurs slots et le système génère automatiquement les formulaires de sorte qu'on puisse éditer les instances. Un exemple de formulaire de la classe Professor ressemble à ce qui suit.



Le système crée les widgets graphiques appropriés pour chacun des slots dans la classe de l'instance. Ces formulaires peuvent être consultés à partir de l'onglet Instances ou de divers autres endroits. On peut changer la présentation par défaut de ces formulaires dans Protégé en utilisant l'onglet Forms. Ici on peut réarranger la disposition des widgets ou remplacer certains widgets par d'autres, y compris les composants adaptables écrits en Java. (Se référer au tutorial général de Protégé pour plus d'information sur ces questions)

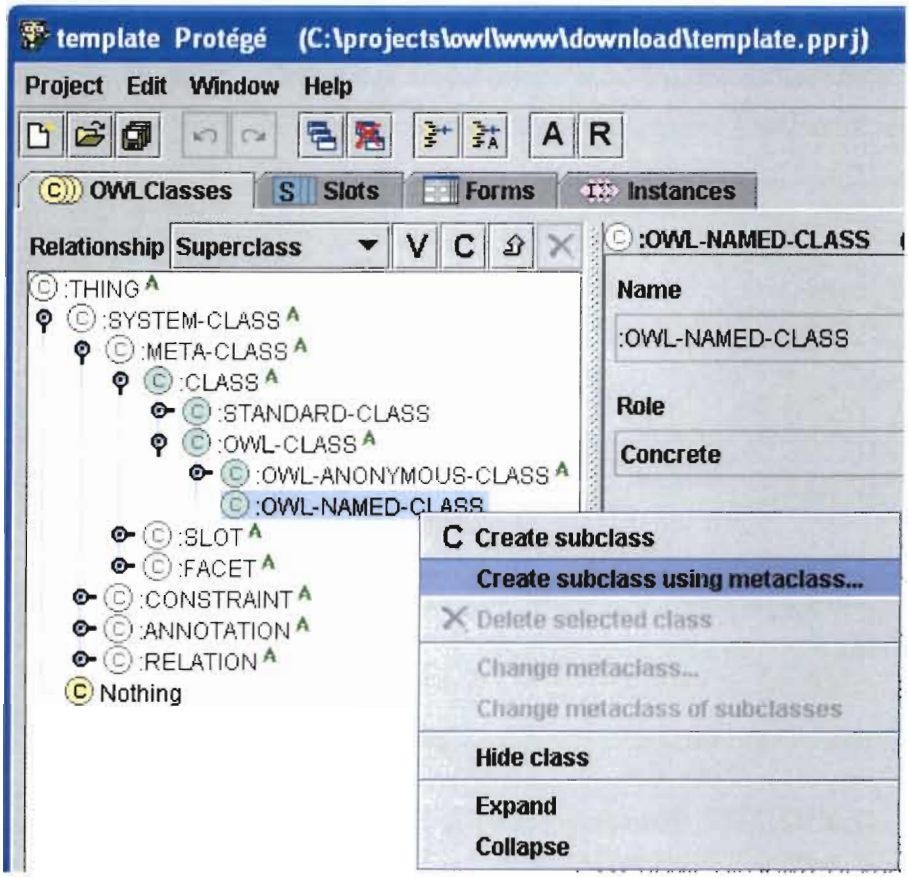
Si on veut changer le nom d'une instance affiché par défaut (de sorte qu'il ne s'affiche pas "Instance_42" mais comme "Hans Pansen") alors on va à l'onglet Forms pour sélectionner la classe des instances et choisir le "Form Browser Key" dans la liste des slots disponibles.

1.3.12 Définition d'une nouvelle méta classe

Une méta classe est une classe qui a le type "class" comme super classe. Les instances d'une méta classe sont également des classes. Protégé autorise la définition de méta classes propres pour introduire des slots additionnels dans des classes. Contrairement au template slots normal, ces slots sont utilisés au niveau classe et définissent les propriétés des classes qui sont instances de la méta classe. Etant donné que la puissance de OWL réside dans la définition de classes, ceci est un dispositif important du Plugin OWL.

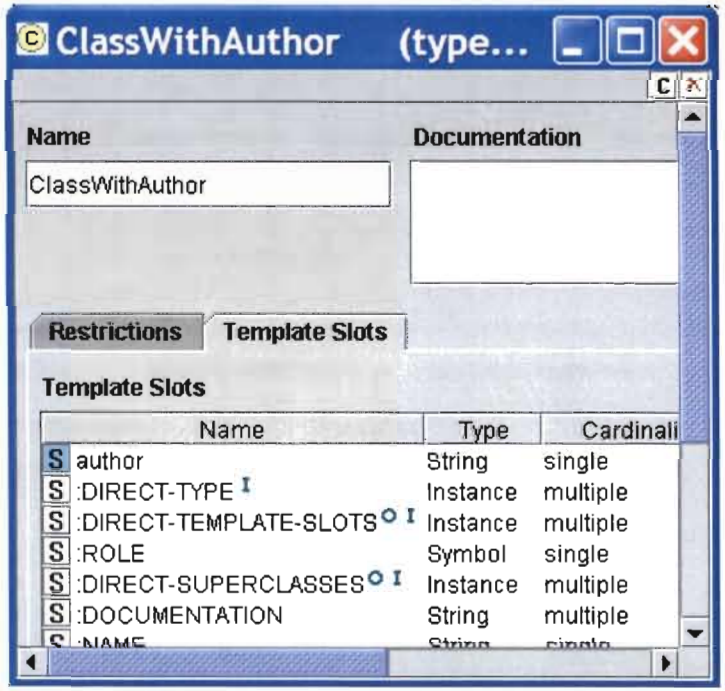
Il y a deux façons de simuler le support désiré de méta classe dans le Plugin OWL. La précédente approche serait d'utiliser les méta classes intégrées dans OWL. Ceci signifie que les ontologies deviennent automatiquement des ontologies OWL Full, et comme résultat aucun classifier pourrait travailler avec ces méta classes. C'est pourquoi la décision de conception de maintenir tout en OWL DL et d'utiliser AnnotationProperties pour capturer les méta données additionnelles. Pour faire cela, on a fourni une méta ontologie qui définit quelques propriétés et classes OWL utilisés par Protégé. Ces propriétés sont habituellement ignorées par d'autres outils, mais pourraient être également exploitées. D'ailleurs, cette méta donnée représente également des attributs de classe spécifiques de Protégé comme si une classe est abstraite ou non.

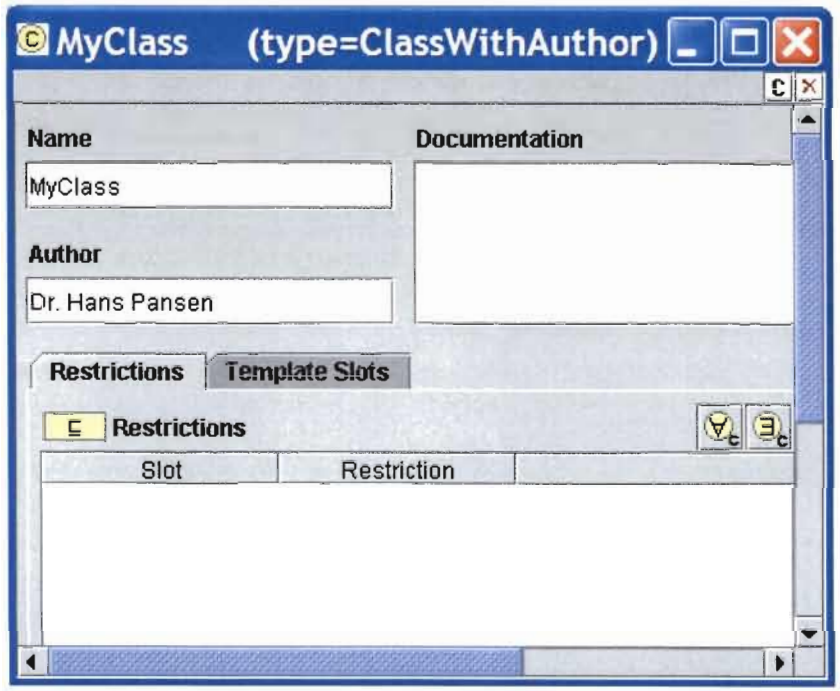
Si on veut définir nos propres méta classes, on peut créer des sous classes de la classe :OWL-NAMED-CLASS, qui se trouve dans la hiérarchie de méta classes systèmes. Cette nouvelle méta classe doit également avoir la classe :OWL-NAMED-CLASS comme type, ainsi on doit utiliser "Create subclass using metaclass..." comme ci-dessous.



On peut plus tard instancier ces nouvelles méta classes avec le popup menu "Create subclass using metaclass...". Dans la version courante, tous les concepts de méta modélisation devraient être manipulés avec prudence.

La partie gauche de la figure ci-dessous montre une méta classe qui a un template slot additionnel "author". La partie droite de la figure montre une instance de cette méta classe, où les utilisateurs ont spécifié une certaine valeur pour le slot author. Le système déclare automatiquement les propriétés comme AnnotationProperties quand elles sont assignées à une méta classe (dans ce cas, le slot author est représenté par AnnotationProperty). De cette façon, on assure qu'aucune des classes définies par l'utilisateur n'entre dans OWL inutilement. Ceci signifie également que n'importe quelle valeur assignée sera ignorée par les classifiers ou d'autres composants.





On peut appliquer la même technique pour définir les sous classes de la méta classe slot :OWL-SLOT et ensuite créer de nouveaux slots comme instances des nouvelles méta classes slot. Toutefois, ceci n'est pas entièrement supporté encore (il y a quelques dispositifs importants non encore implémentés dans Protege 2.0).

1.3.13 Accès à des classifiers et à d'autres reasoners

Les reasoners sont des composants qui peuvent dériver de nouvelles "connaissances" à partir d'une ontologie existante. Les riches sémantiques de OWL supportent divers types de raisonnement, en particulier la classification. Un classificateur (classifier) prend une hiérarchie de classe et la change pour refléter les contraintes et d'autres définitions de classe dans l'ontologie.

La classification supportée actuellement est basée sur l'API OWL de Sean Bechhofer. Cependant, on fournit uniquement un très simple interface pour y accéder. Fondamentalement, on peut utiliser le bouton "Classify" au-dessus de l'arborescence de classes dans l'onglet OWL Classes pour exécuter les classificateurs. Avant qu'on puisse l'utiliser, on doit s'assurer qu'un classificateur externe avec un interface DIG-compliant s'exécute et est accessible à travers notre localhost au port 8080. Ceci se produit habituellement quand on démarre le Server Racer avant Protégé. Quand on tape le bouton classify, le système invoquera alors Racer (selon la taille et les propriétés structurelles de notre ontologie, ceci pourrait mettre du temps). Quand Racer a terminé, il affiche un petit rapport pour indiquer le nombre de nouvelles relations de classe identifiées.

L'exécution du classificateur peut changer directement la hiérarchie d'héritage. Si on veut seulement tester le classificateur et ne pas tenir compte des changements, on peut utiliser les boutons Archive (A) et Revert (R) dans la barre d'outils. Un cycle typique de classification pourrait être vu comme :

- Taper A pour enregistrer la hiérarchie courante
- Taper Classify pour exécuter le classificateur
- Bouillir (Boil) un ou deux tasses de thé si on a une grande ontologie à classifier
- Vérifier si la hiérarchie nouvellement créée répond à nos espérances ou non
- Taper R pour retourner à la version créée dans l'étape 1.

Cette interface utilisateur de base et certainement insuffisante sera étendue avec le support intelligent pour comparer les deux versions de notre ontologie. En particulier on a l'intention d'employer le Plugin Prompt pour visualiser les changements, de sorte d'obtenir deux vues de notre hiérarchie de classes qui met en exergue les différences. On peut déjà simuler cette vue en enregistrant les deux versions en deux différents fichiers (par exemple, my-ontology-pre.pprj et my-ontology-classified.pprj) et en les ouvrant avec Prompt.

L'implémentation du prototype actuelle se fonde sur une version pré-alpha du API OWL de Sean. Ce prototype peut ne pas pouvoir analyser toutes les ontologies OWL.

1.4 Limites du Plugin OWL

Protégé est un éditeur d'ontologie open source. Il fournit un interface graphique utilisateur convivial et une architecture ouverte pour plugins. Beaucoup de plugins de langages tels que UML et XML, et plusieurs outils d'affichage de Protege existent.

Le plugin OWL permet de charger et d'enregistrer des fichiers OWL et d'utiliser des widgets graphiques adaptables pour éditer des ontologies de logique de description. C'est tout juste une pré-version avec des dispositifs incomplets. Cette version est construite sur les pré-versions de Protege 2.0 alpha et du API OWL Jena 2.0 de HP Labs.

La version actuelle est limitée. Il n'y a aucun accès aux classificateurs ou aux "reasoners" de OWL encore. L'enregistrement de grandes ontologies Protege pourraient engendrer des problèmes de mémoire. L'importation des ontologies n'est pas implémentée (actuellement, seulement un fichier avec un seul namespace est autorisé). Tous les éléments du langage OWL ne peuvent pas être édités directement (par exemple owl:AllDifferent).

1.5 Résumé du Mapping entre Protégé et OWL

Les tableaux suivants résument le mapping entre le modèle objet de Protégé (plus les composants qui l'opèrent) et OWL.

1.5.1 Protégé à OWL

Protégé Facet	**OWL Global Concept**	**OWL Local Concept** (Override)
:CONSTRAINTS	-	(not available in Protégé)
:DIRECT-SUPERSLOTS	rdfs:subPropertyOf	(not available in Protégé)

:DEFAULTS	-	-
:DOCUMENTATION	rdfs:comment	-
:INVERSE	owl:inverseOf	(not available in Protégé)
:MAXIMUM-CARDINALITY	restrictions on owl:Thing, owl:functionalProperty ("1" only)	owl:maxCardinality
:MINIMUM-CARDINALITY	restrictions on owl:Thing	owl:minCardinality
:NUMERIC-MAXIMUM	-	-
:NUMERIC-MINIMUM	-	-
:VALUE-TYPE / allowed values/classes	rdfs:range	owl:AllValuesFrom

1.5.2 OWL à Protégé

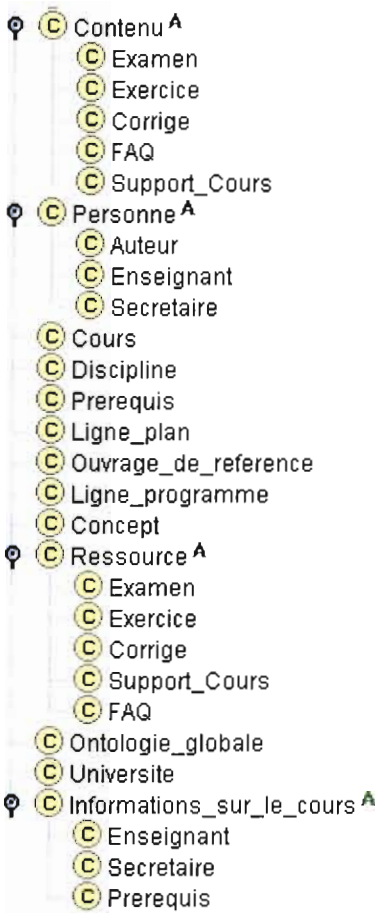
Le tableau suivant résume le mapping de tous les éléments du langage OWL dans les éléments d'ontologie de Protégé. *Level* est soit Ontologie (Ont), Classe (C), Propriété (P) or Individu (I). *Scope* est Locale pour indiquer que ce OWL représente une restriction de propriété qui sera implémentée comme une facette slot privilégiée dans Protégé. Les extensions au méta modèle de Protégé sont écrites en caractères gras.

OWL Language Element	**Level**	**Scope**	**Protégé Element**
owl:AllDifferent	Ont		**OWL-API only or customized individual sets in KnowledgeBase**
owl:allValuesFrom	P	Local	**:OWL-ALL-RESTRICTION** metaclass with override on :VALUE-TYPE / allowed values
owl:cardinality	P	Local	**:OWL-CARDI-RESTRICTION** metaclass with override on both min and max cardinalities
owl:Class	C		**:OWL-CLASS / :OWL-NAMED-CLASS**
owl:complementOf	C		**:OWL-COMPLEMENT-CLASS** metaclass with complementary class as operand
rdfs:Datatype	P		On-the-fly mapping to Protégé value types (ValueType.STRING for "difficult" types)
owl:DatatypeProperty	P		:SLOT-VALUE-TYPE one of BOOLEAN, FLOAT, INTEGER, STRING, SYMBOL
owl:differentFrom	I		**OWL-API only or customized individual sets in KnowledgeBase**
owl:disjointWith	C		**:OWL-DISJOINT-CLASSES** slot
rdfs:domain	P		:DIRECT-TEMPLATE-SLOTS
owl:equivalentClass	C		:DIRECT-SUPERCLASSES with both classes being a subclass of each other
owl:FunctionalProperty	P		:MAXIMUM-CARDINALITY of 1

owl:hasValue	P	Local	**:OWL-HAS-VALUE-RESTRICTION** metaclass with override on :VALUES
owl:intersectionOf	C		**:OWL-INTERSECTION-CLASS** metaclass with intersection classes as operands
owl:InverseFunctionalProperty	P		**:OWL-INVERSE-FUNCTIONAL** facet in :OWL-SLOT metaclass
owl:inverseOf	P		:INVERSE
owl:maxCardinality	P	Local	**:OWL-MAXCARDI-RESTRICTION** metaclass with override on max cardinality
owl:minCardinality	P	Local	**:OWL-MINCARDI-RESTRICTION** metaclass with override on min cardinality
owl:Nothing	Ont		- (?)
owl:ObjectProperty	P		:SLOT-VALUE-TYPE with a value type INSTANCE
owl:oneOf	P		:SLOT-VALUE-TYPE Symbol or **:OWL-ENUMERATION-CLASS** metaclass
owl:onProperty	P	Local	Subclass of **:OWL-RESTRICTION** with property as the only template slot
owl:Ontology	Ont		**OWL-API only or customized metadata representation**
rdf:Property	P		**:OWL-SLOT**
rdfs:range	P		:VALUE-TYPE / allowed classes
owl:Restriction	P	Local	Superclass of **:OWL-RESTRICTION** type with slot overriding (but only narrowing!)
owl:sameAs / owl:sameIndividualAs	I		**OWL-API only or customized individual sets in KnowledgeBase**
owl:equivalentProperty	P		**:OWL-EQUIVALENT-SLOTS**
owl:someValuesFrom	P	Local	**:OWL-SOME-RESTRICTION** metaclass with override on **:OWL-SOME**

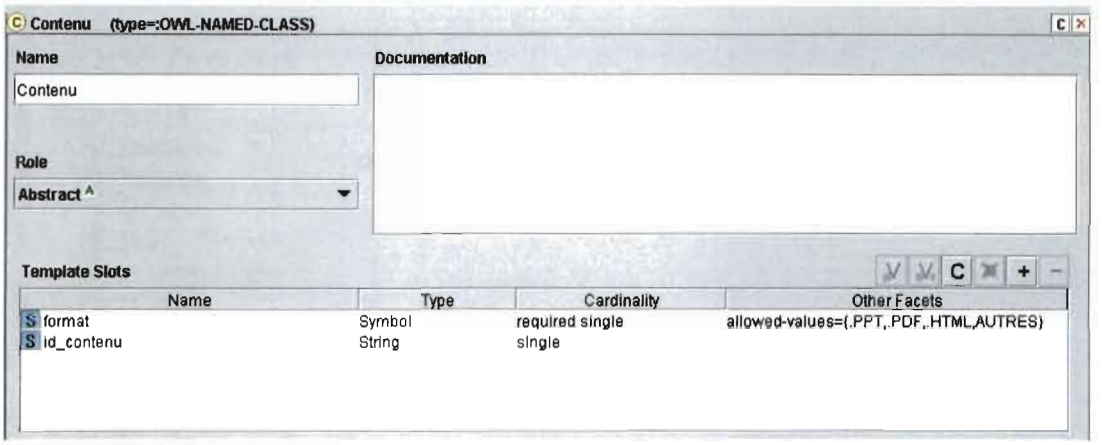
ANNEXE E : PROTOTYPE

1 Hiérarchie de classes



2 Description de classes

2.1 Contenu


Contenu (type=:OWL-NAMED-CLASS)

Name: Contenu

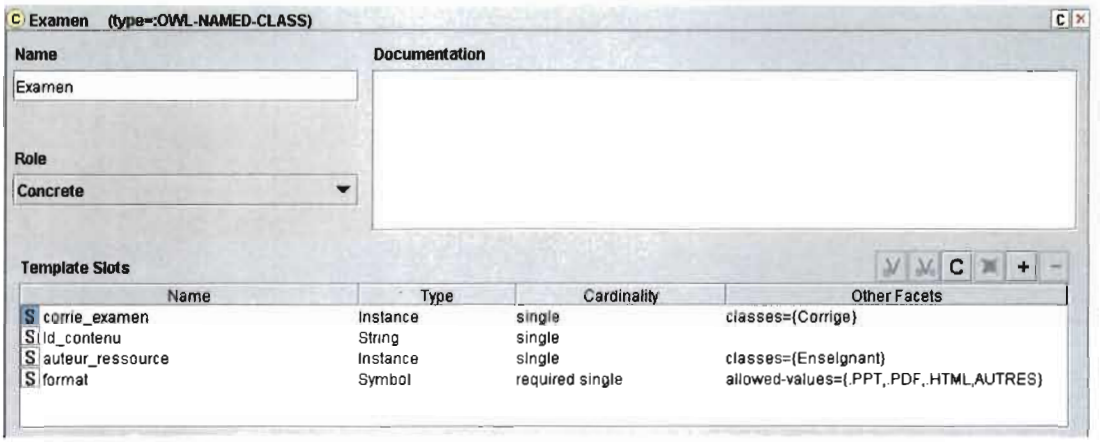
Documentation:

Role: Abstract

Template Slots

Name	Type	Cardinality	Other Facets
format	Symbol	required single	allowed-values={.PPT,.PDF,.HTML,AUTRES}
id_contenu	String	single	

2.2 Examen


Examen (type=:OWL-NAMED-CLASS)

Name: Examen

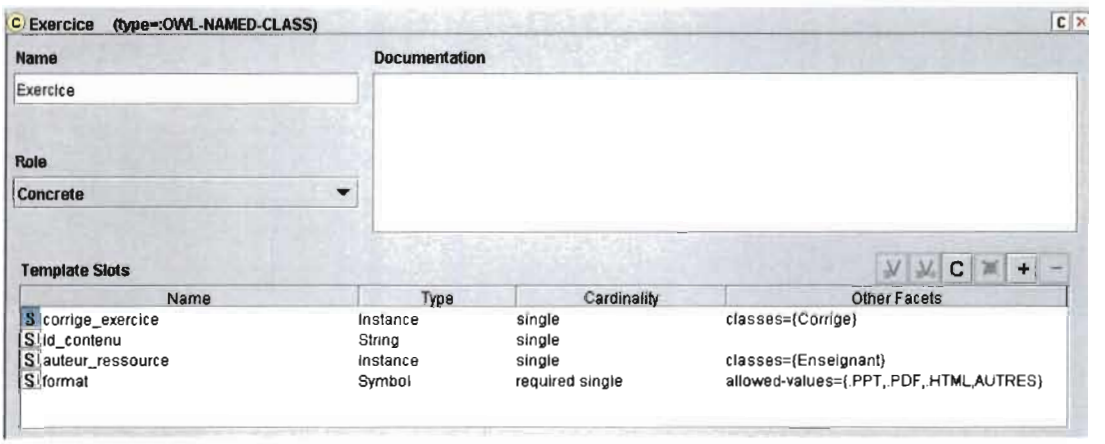
Documentation:

Role: Concrete

Template Slots

Name	Type	Cardinality	Other Facets
corrie_examen	Instance	single	classes={Corrige}
id_contenu	String	single	
auteur_ressource	Instance	single	classes={Enseignant}
format	Symbol	required single	allowed-values={.PPT,.PDF,.HTML,AUTRES}

2.3 Exercice


Exercice (type=:OWL-NAMED-CLASS)

Name: Exercice

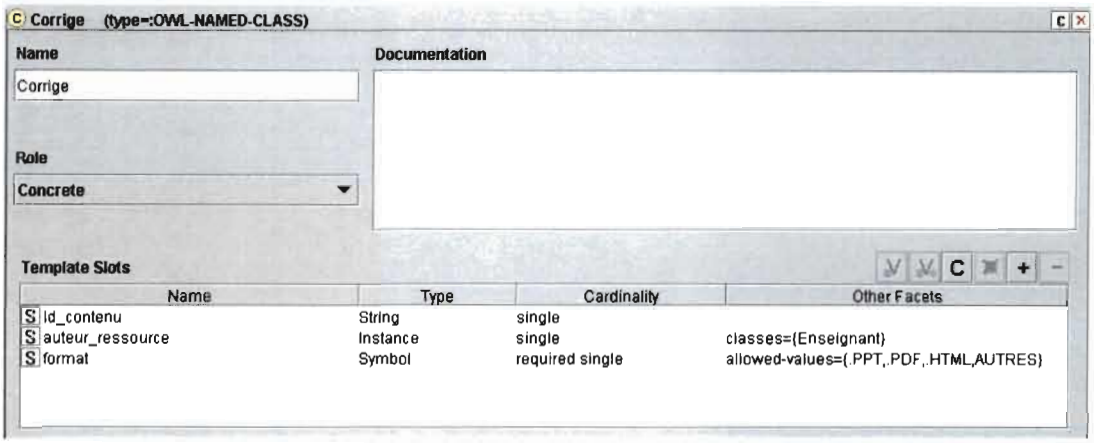
Documentation:

Role: Concrete

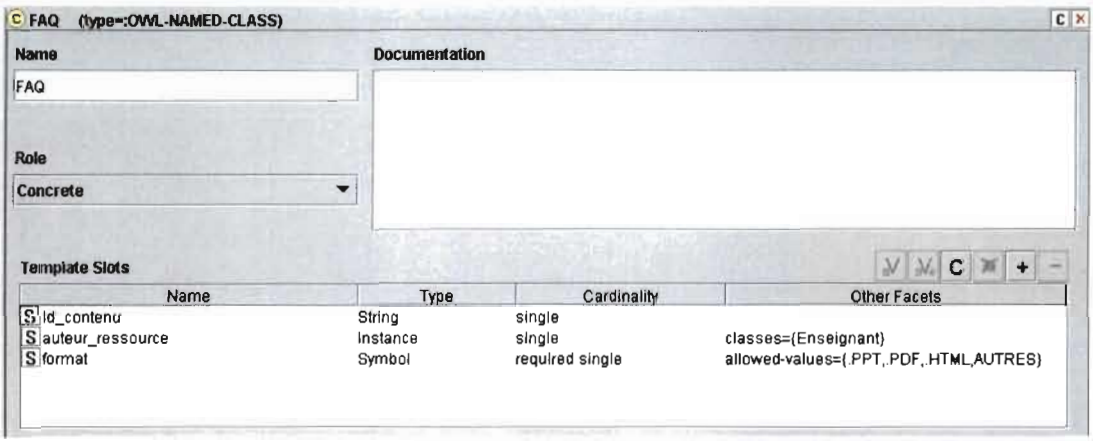
Template Slots

Name	Type	Cardinality	Other Facets
corrige_exercice	Instance	single	classes={Corrige}
id_contenu	String	single	
auteur_ressource	Instance	single	classes={Enseignant}
format	Symbol	required single	allowed-values={.PPT,.PDF,.HTML,AUTRES}

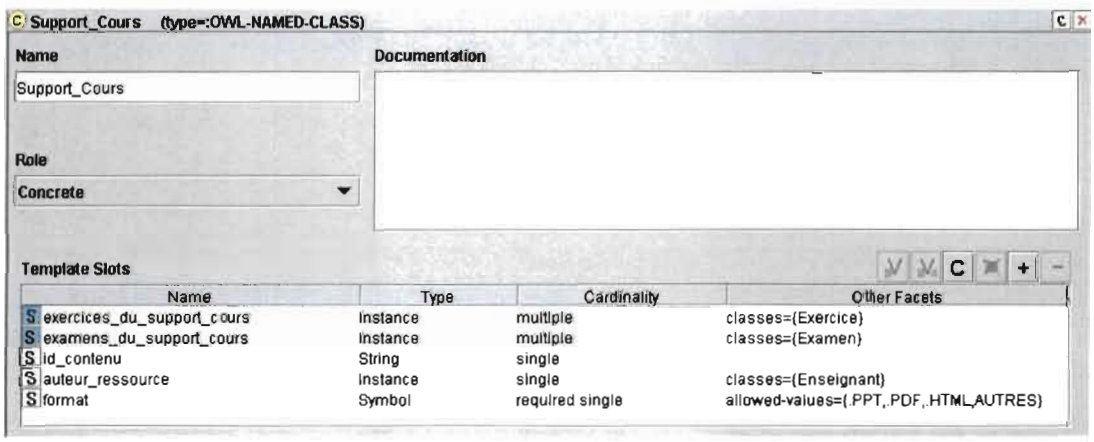
2.4 Corrige



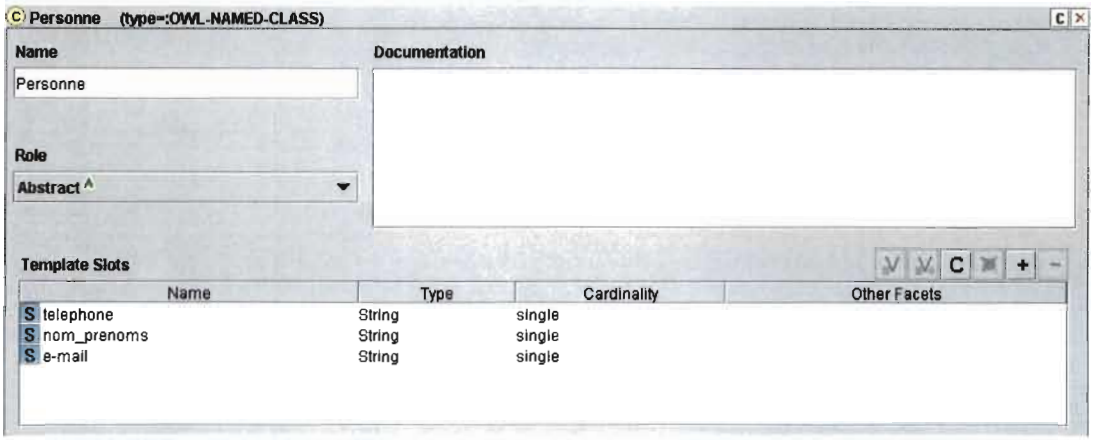
2.5 FAQ



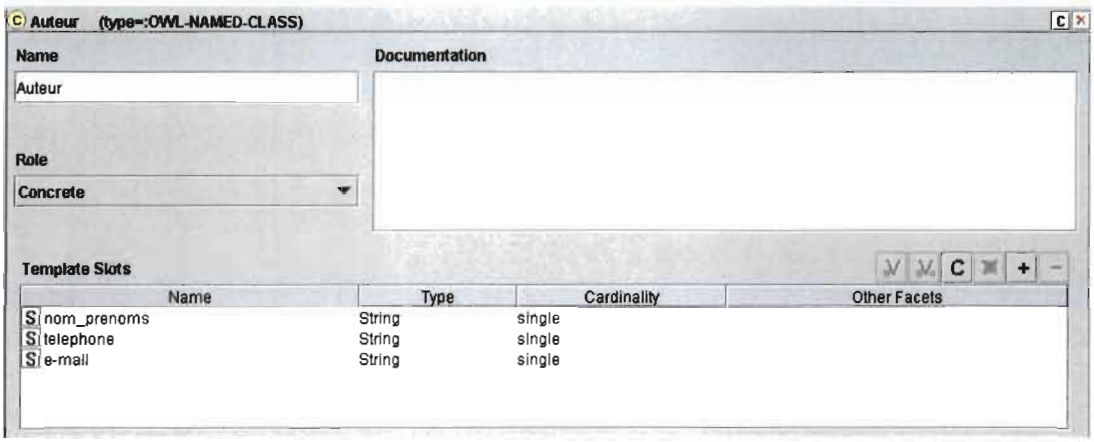
2.6 Support_Cours



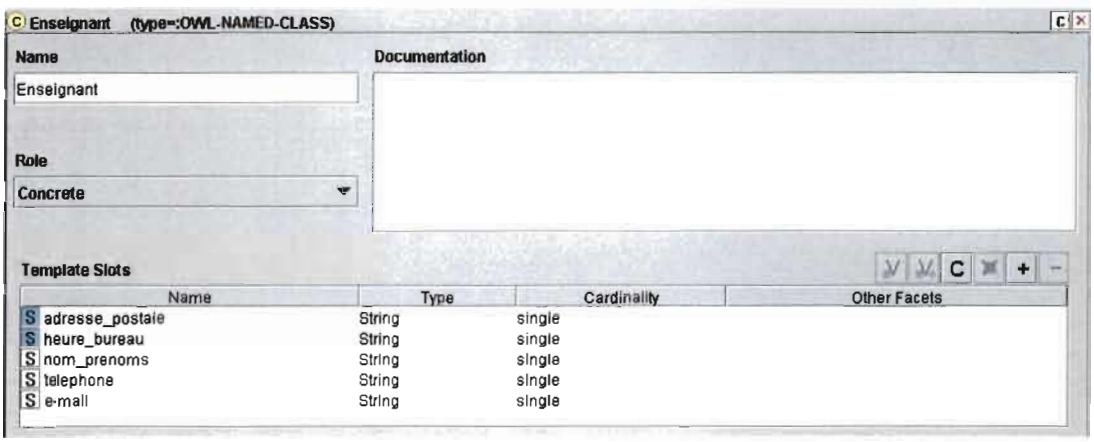
2.7 Personne



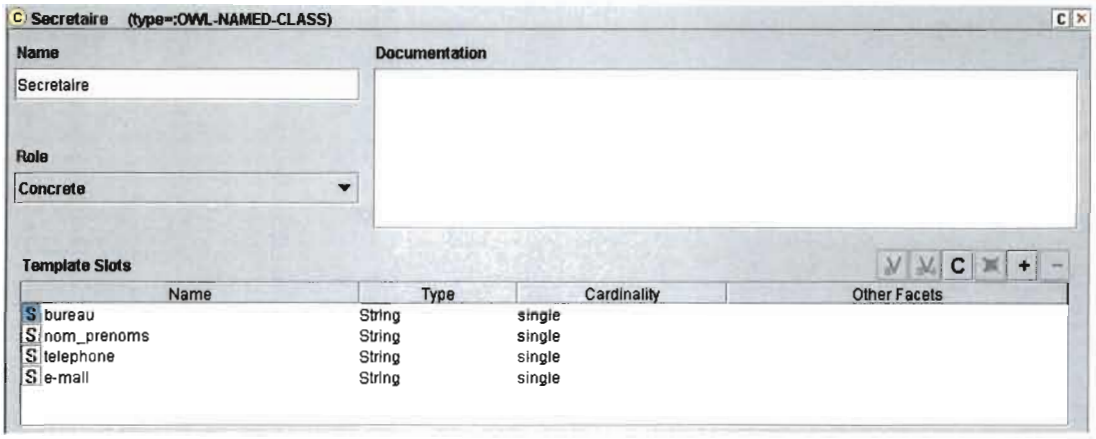
2.8 Auteur



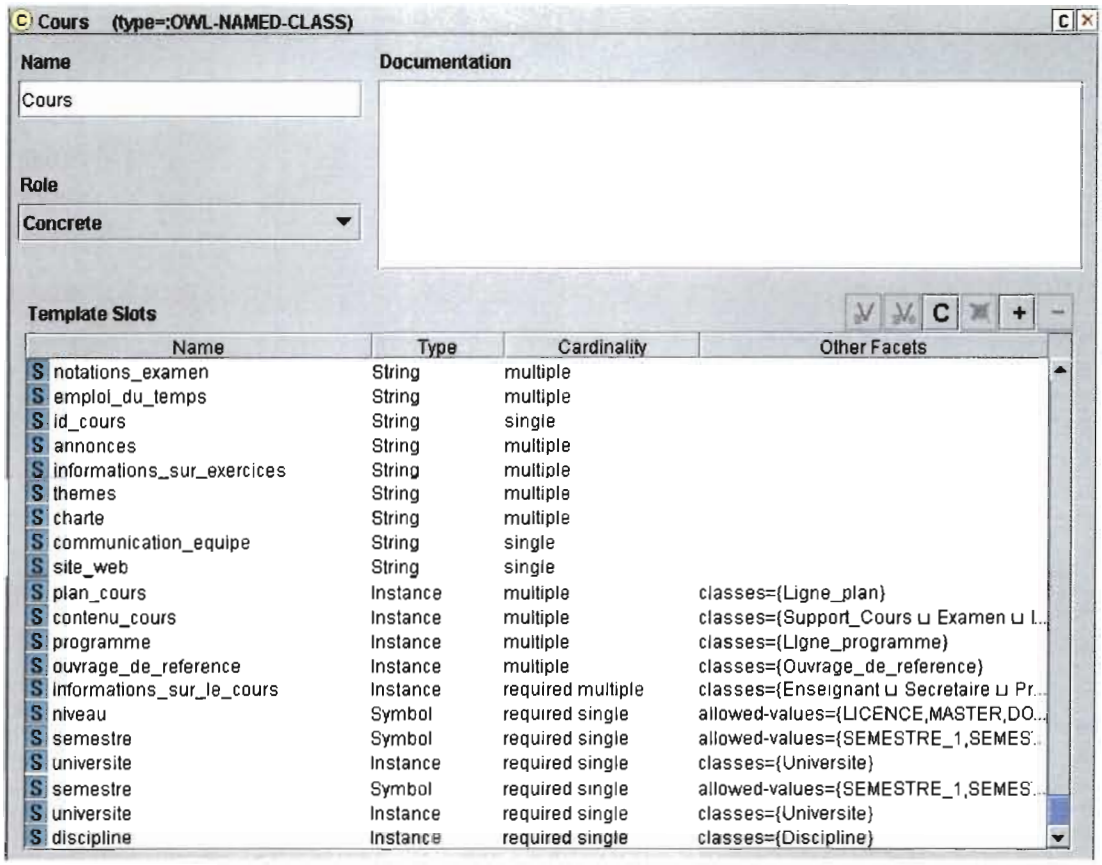
2.9 Enseignant



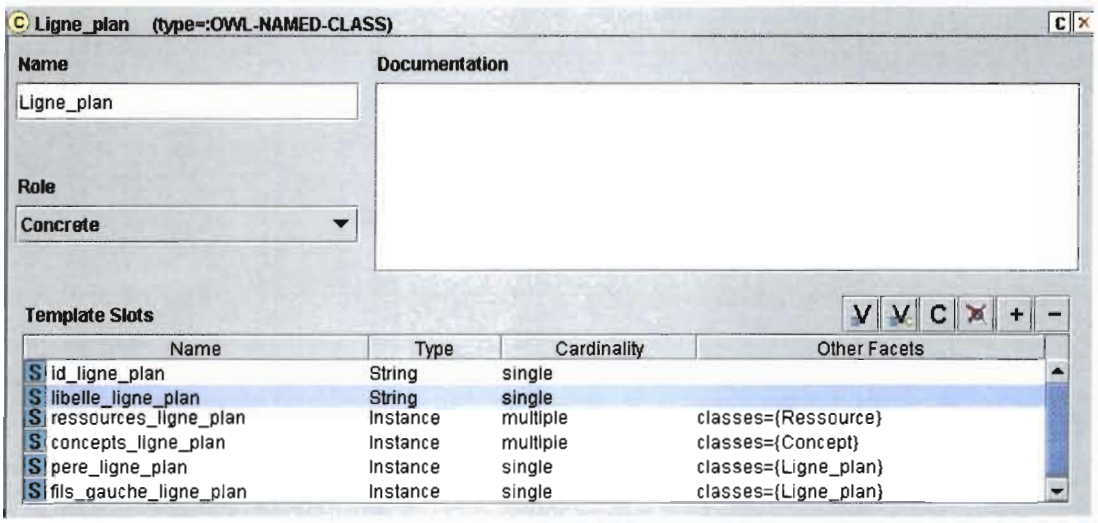
2.10 Secrétaire



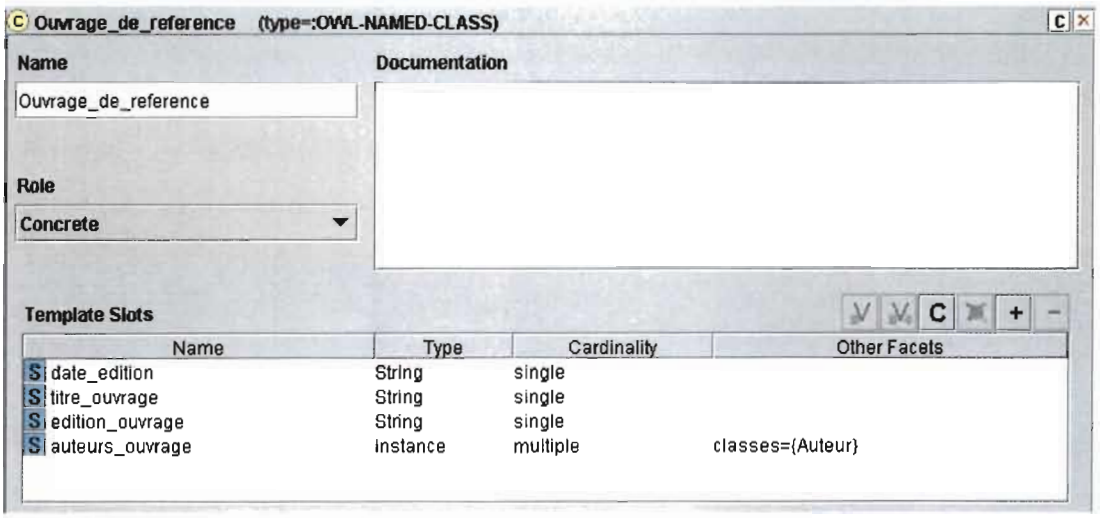
2.11 Cours



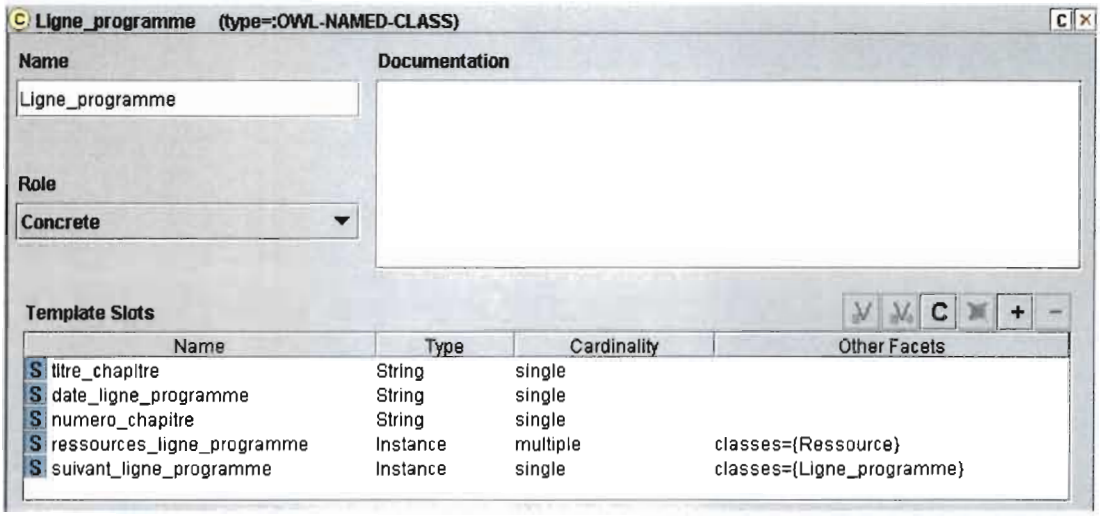
2.12 Ligne plan



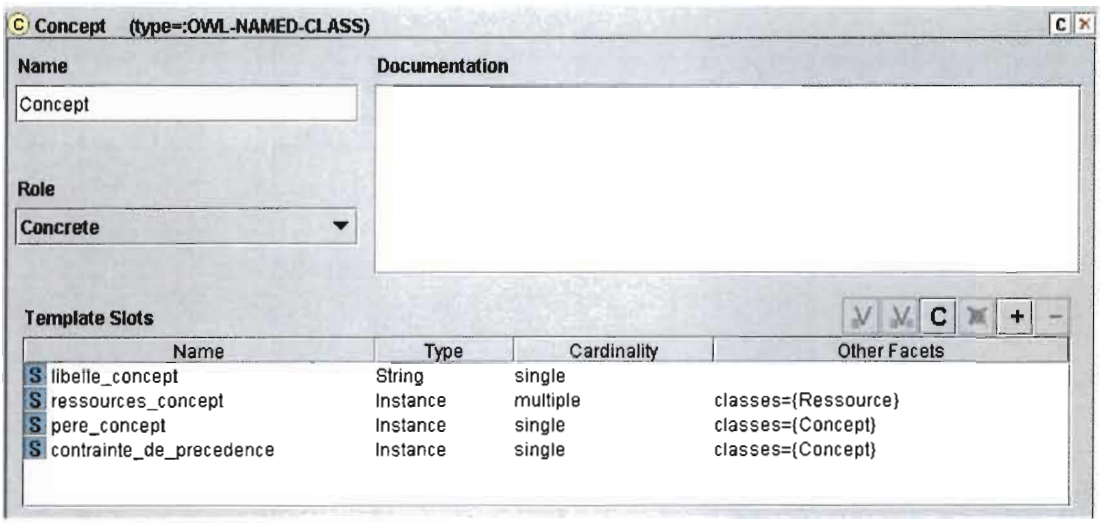
2.13 Ouvrage de référence



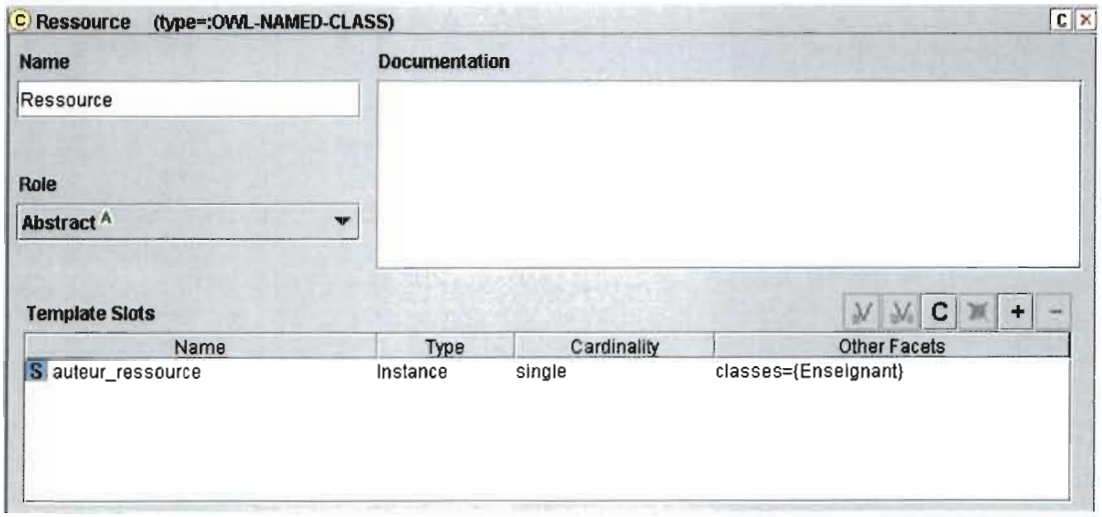
2.14 Ligne Programme



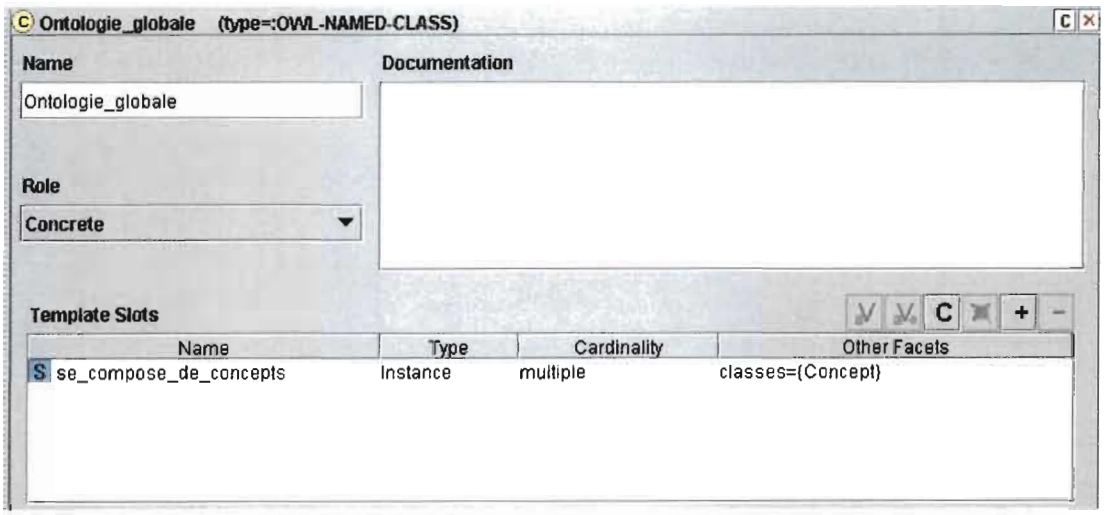
2.15 Concept



2.16 Ressource

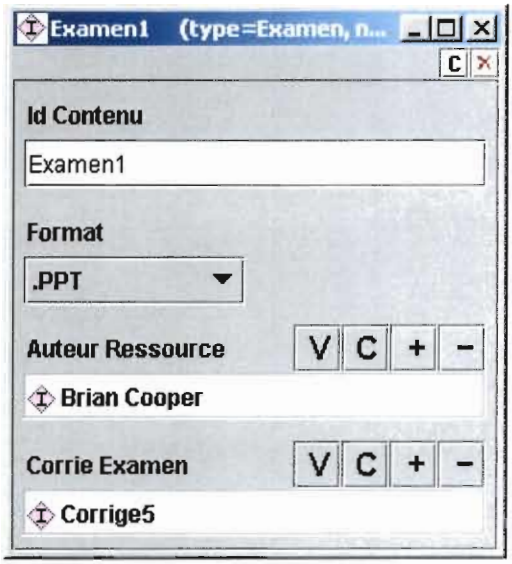


2.17 Ontologie globale



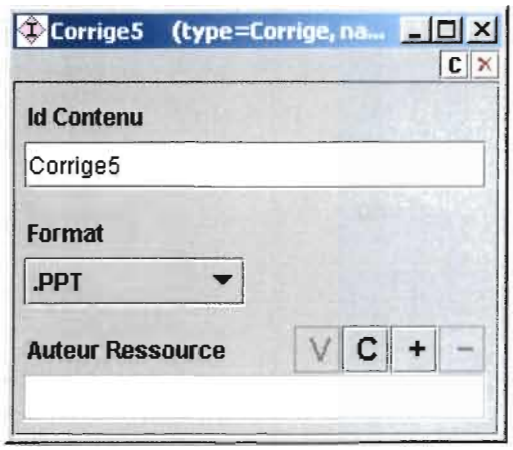
3 Instances et leurs formulaires

3.1 Examen

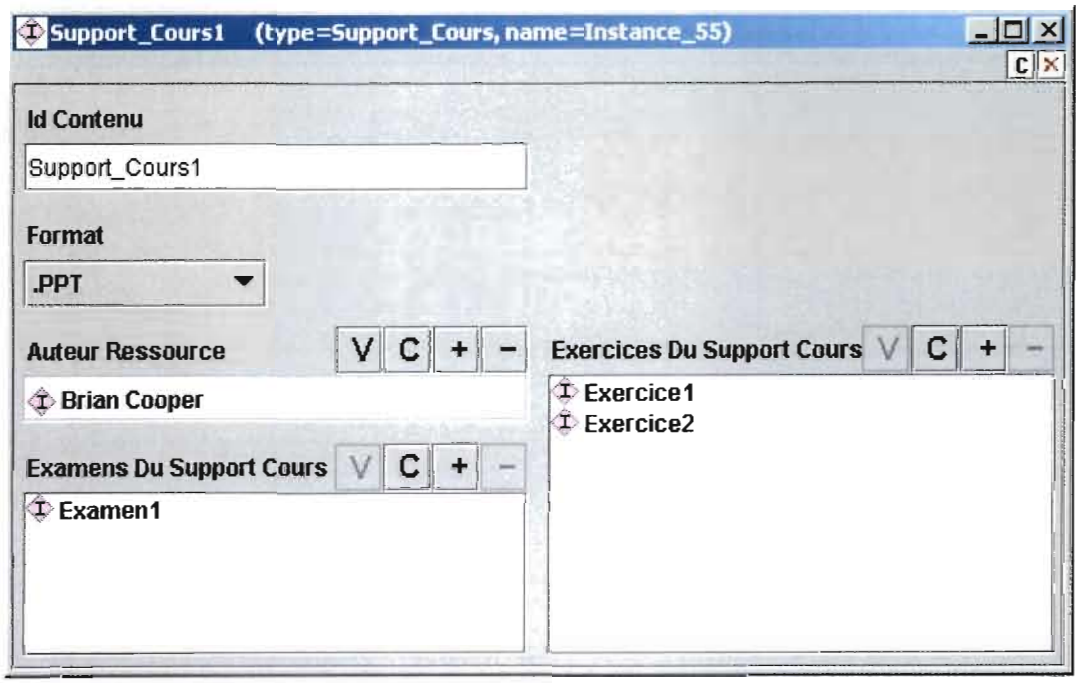


3.2 Exercice

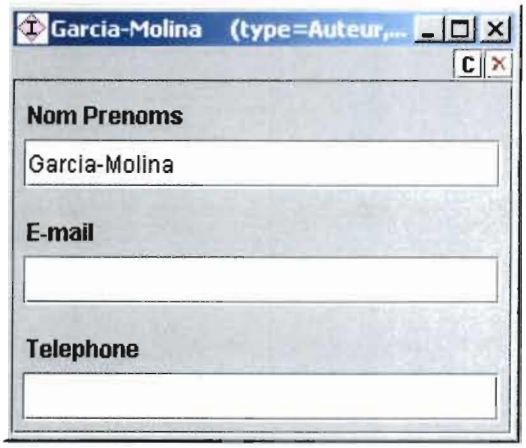
3.3 Corrige



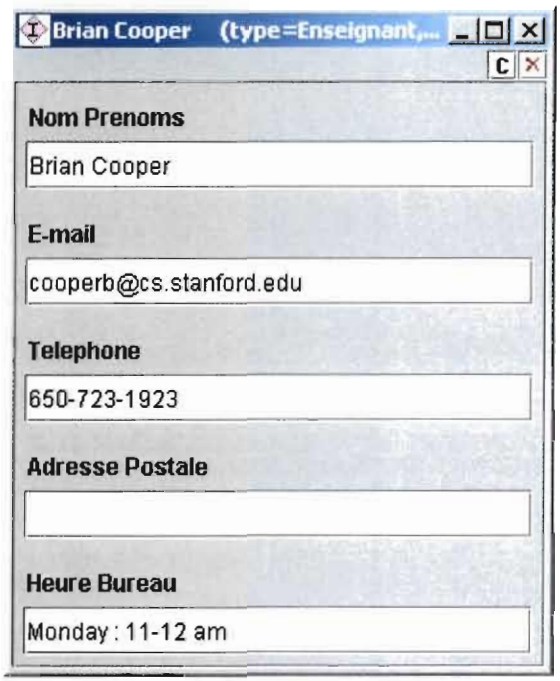
3.4 Support de Cours



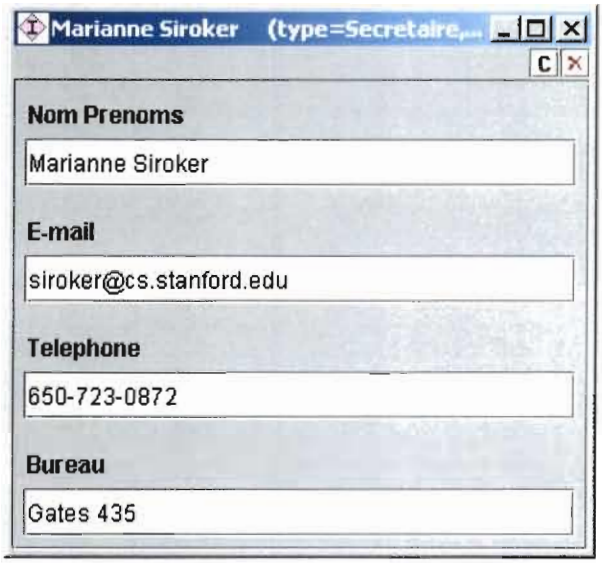
3.5 Auteur



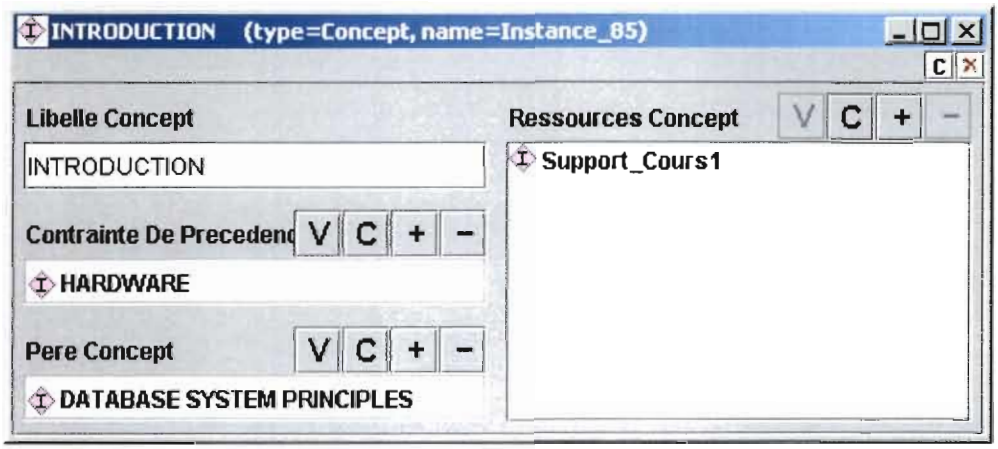
3.6 Enseignant



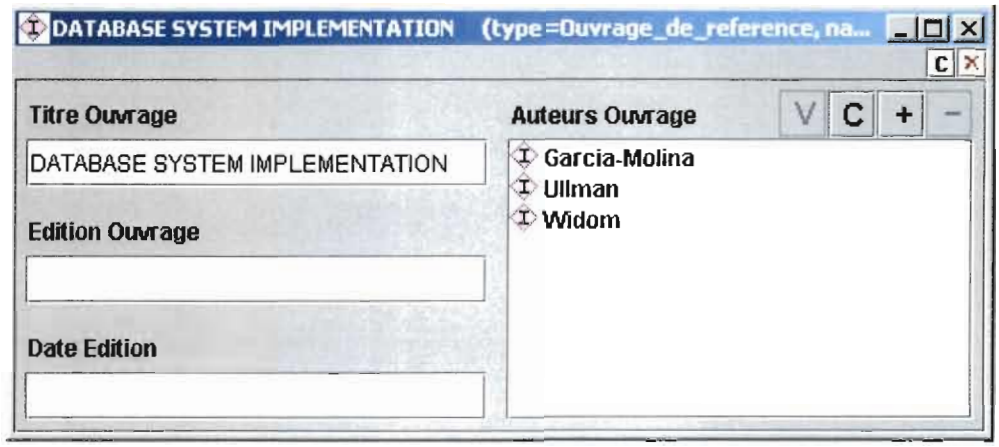
3.7 Secrétaire



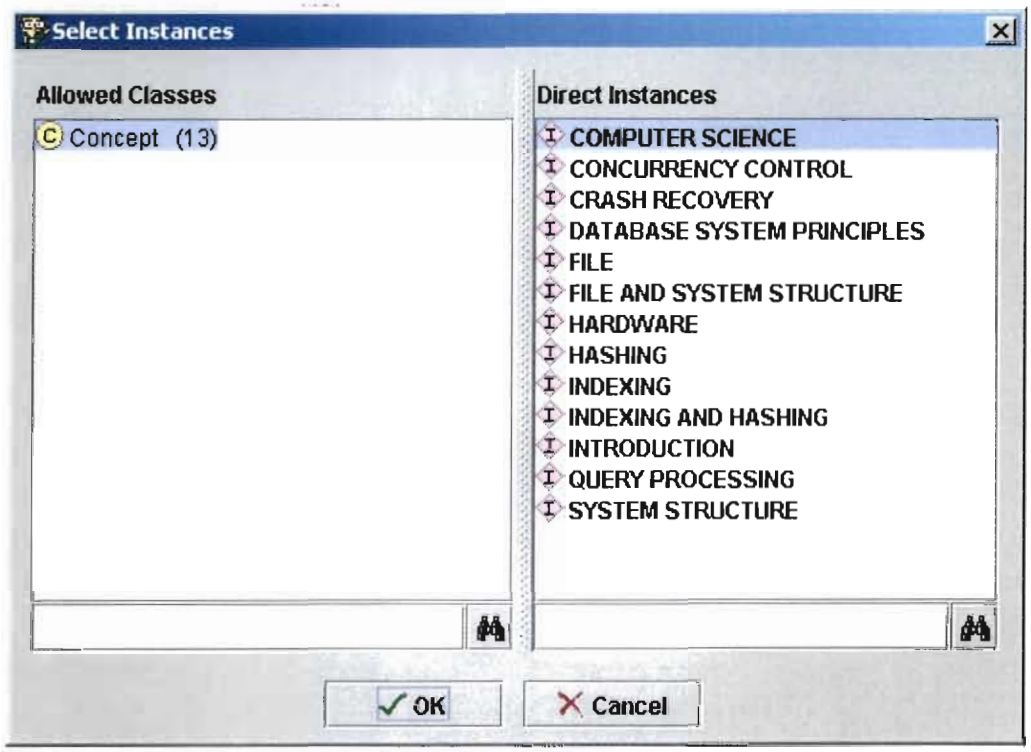
3.8 Concept



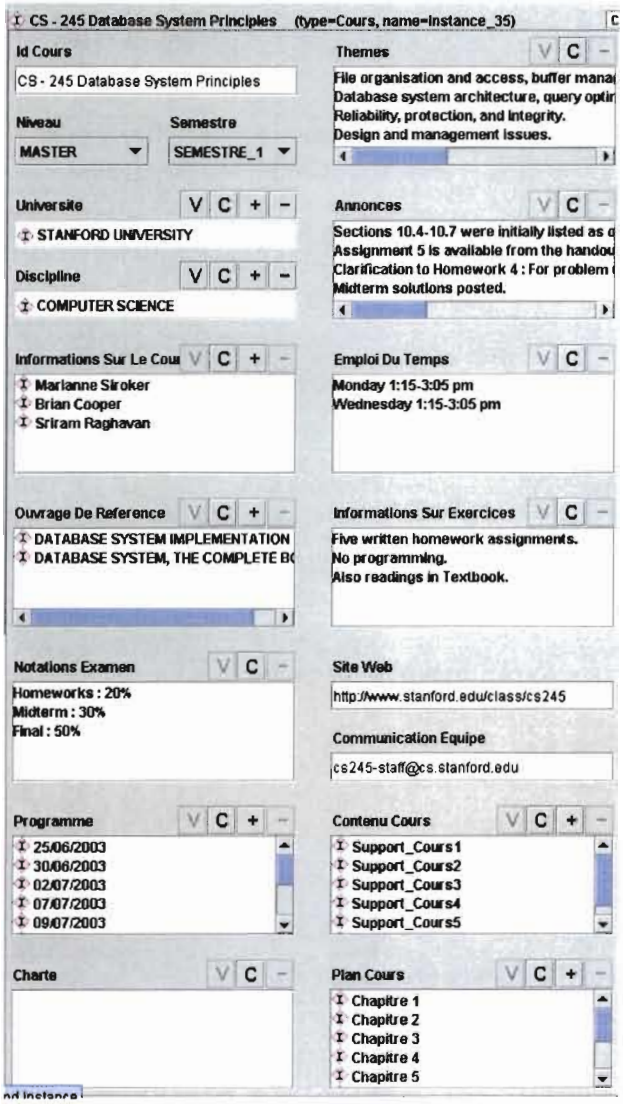
3.9 Ouvrage de référence



3.10 Ontologie Globale

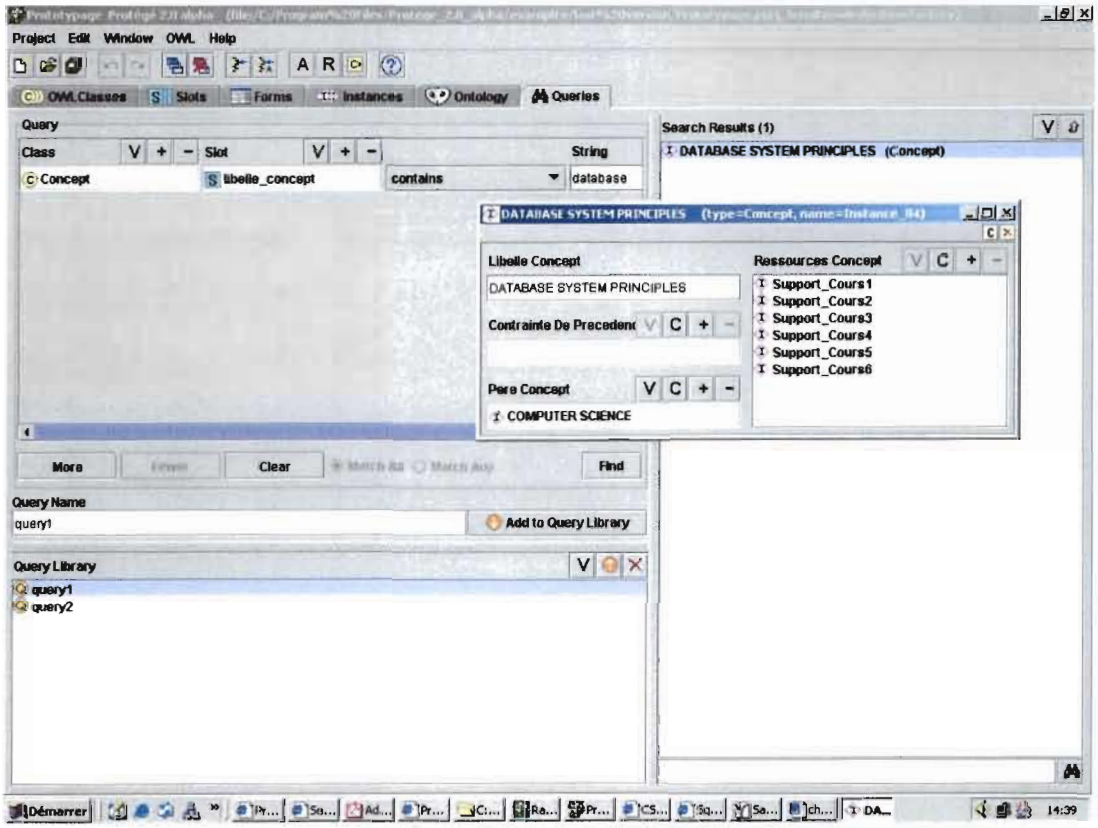


3.11 Cours

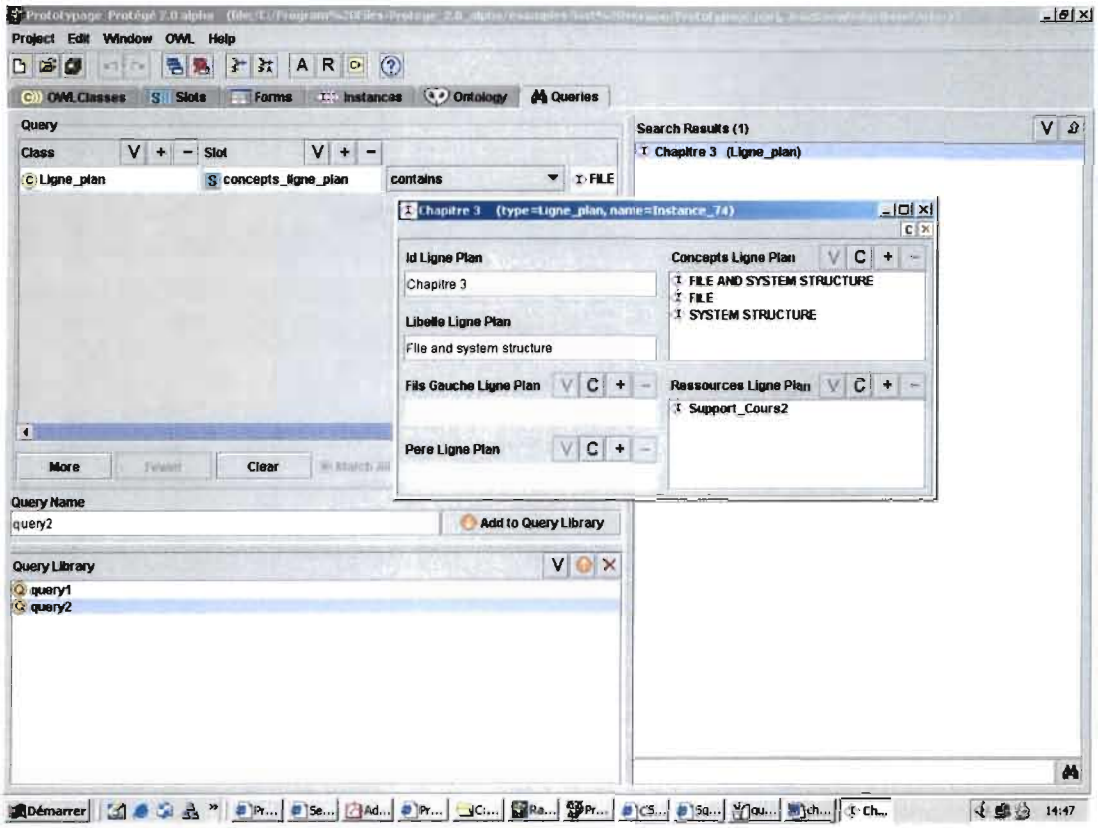

CS - 245 Database System Principles (type=Cours, name=Instance_35)
Id Cours
CS - 245 Database System Principles
Themes
File organisation and access, buffer mana
Database system architecture, query optir
Reliability, protection, and integrity.
Design and management issues.
Niveau
MASTER
Semestre
SEMESTRE_1
Universite
STANFORD UNIVERSITY
Annonces
Sections 10.4-10.7 were initially listed as
Assignment 5 is available from the handou
Clarification to Homework 4 : For problem
Midterm solutions posted.
Discipline
COMPUTER SCIENCE
Informations Sur Le Cou
Marianne Siroker
Brian Cooper
Sriram Raghavan
Emploi Du Temps
Monday 1:15-3:05 pm
Wednesday 1:15-3:05 pm
Ouvrage De Reference
DATABASE SYSTEM IMPLEMENTATION
DATABASE SYSTEM, THE COMPLETE B
Informations Sur Exercices
Five written homework assignments.
No programming.
Also readings in Textbook.
Notations Examen
Homeworks : 20%
Midterm : 30%
Final : 50%
Site Web
http://www.stanford.edu/class/cs245
Communication Equipe
cs245-staff@cs.stanford.edu
Programme
25/06/2003
30/06/2003
02/07/2003
07/07/2003
09/07/2003
Contenu Cours
Support_Cours1
Support_Cours2
Support_Cours3
Support_Cours4
Support_Cours5
Charte
Plan Cours
Chapitre 1
Chapitre 2
Chapitre 3
Chapitre 4
Chapitre 5

4 Quelques requêtes

4.1 Query1 : Les ressources qui concernent les concepts qui contiennent "database"



4.2 Query2 : Les parties du plan qui traitent du concept "File and system structure"



BIBLIOGRAPHIE

[Bechofer & al. 2003] Sean Bechofer, Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, Lynn Andrea Stein, *OWL Web Ontology Language Reference,* Editors Mike Dean, Guus Schreiber, Editor's Draft document – no status at this time June 19 2003. http://www.daml.org/2002/06/webont/owl-ref-proposed#datatype

[Brase & al. 2003a] Jan Brase, Wolfgang Nejdl, *Ontology and Metadata for eLearning,* Published in Handbook on ontology, Springer Verlag, 2003.

[Brase & al. 2003b] Jan Brase, Mark Painter, Wolfgang Nejdl, *Completing LOM – How additional axioms increase the utility of learning object Metadata,* Published in the 3rd IEEE International Conference on Advanced Learning Technologies ICALT 2003.

[Dublin] Dublin Core Metadata Initiative, http://dublincore.org

[Harmelen & al. 2003] Frank van Harmelen, Jim Hendler, Ian Horrocks, Deborah L. McGuinness, Peter F. Patel-Schneider, Lynn Andrea Stein, *OWL Web Ontology Language Reference,* Editors Mike Dean, Guus Schreiber, W3C Working Draft March 31, 2003. http://www.w3.org/TR/owl-ref/#Header

[Kettani 1998] Nasser Kettani, Dominique Mignet, Pascal Paré, Camille Rosenthal-Sabroux, *De Merise à UML,* Eyrolles, 1998

[Knublauch 2003] Holger Knublauch, Stanford University, Protégé OWL Plugin – User's Guide, http://protege.stanford.edu/plugins/owl last updated 16 juillet 2003

[LOM 2002] IEEE Learning Technology Standard Committee (LTSC), IEEE P1484.12 Learning Objects Metadata Working Group, http://ltsc.ieee.org/wg12

[McGuinness 2003] Deborah L. McGuinness (Knowledge Systems Laboratory, Stanford University), Frank van Harmelen (Vrije Universiteit, Amsterdam), *OWL Web Ontology*

Language, Overview, W3C Working Draft March 31, 2003. http://www.w3.org/TR/owl-features/

[Muller 2003] Pierre Alain Muller, *Modélisation Objet avec UML,* http://magda.elibel.tm.fr/refs/UML/didacticiel.pdf Version 2.0

[Muller 2000] Pierre Alain Muller, Nathalie Gaertner, *Modélisation Objet avec UML,* Eyrolles, 2000

[Touitou 2003] Jonathan Touitou, *Conception de cours à partir de ressources pédagogiques sur le web,* Mémoire de DEA, 20 juin 2003