

# RDFS, OWL

Cours « Document et Web Semantique »

Nicolas Delestre

# Plan...

---

- 1 Ontologie
- 2 Spécification des types en RDF
- 3 Les ontologies légères : RDF Schéma
- 4 OWL
- 5 Introduction au raisonnement
- 6 Protégé
- 7 Conclusion

## Constat

- Les métadonnées vont pouvoir aider les moteurs de recherche à sélectionner des ressources
- Mais :
  - Elles ne permettent pas d'identifier le contexte
  - Leur pouvoir d'expression ne permet d'inférer (déduire de nouvelles connaissances à partir de connaissances)

## Définition (1)

« Ensemble des objets reconnus comme existant dans le domaine. Construire une ontologie c'est aussi décider de la manière d'être et d'exister des objets. » [CBT04]

## Définition (2)

« Une ontologie implique ou comprend une certaine vue du monde par rapport à un domaine donné. Cette vue est souvent conçue comme un ensemble de concepts - e.g. entités, attributs, processus -, leurs définitions et leurs interprétations. On appelle cela une conceptualisation.

[...]

Une ontologie peut prendre différentes formes mais elle inclura nécessairement un vocabulaire de termes et une spécification de leur signification

[...]

Une ontologie est une spécification rendant partiellement compte d'une conceptualisation. » [CBT04]

# Construction d'une ontologie

## Étapes de construction d'une ontologie [CBT04]

- 1 Analyse du corpus  
L'analyse de documents du domaine permet d'identifier les termes et relations utilisés
- 2 Normalisation sémantique  
Passage des termes aux concepts du domaine
- 3 Engagement ontologique  
Relier les concepts du domaine à des concepts plus globaux plus générique (par exemple un « Médecin » est une « Personne »)
- 4 Opérationnalisation  
Représentation de l'ontologie à l'aide d'un langage

# Ontologie et thésaurus, classification, etc.

## Ne pas confondre une ontologie avec

- Terminologie  
« ensemble de termes particulier à un domaine »
- Thésaurus  
« ensemble de termes normalisés fondé sur une structuration hiérarchique » [CBT04]  
ex : *MESH*
- Classification  
« distribuer par classes ou par catégories »  
ex : *Classification Dewey*

## Classe (rappel) 1 / 2

- Une ontologie permet donc d'identifier des classes et des propriétés décrivant les concepts et leurs relations
- En RDF il est possible d'associer un URI à une ou plusieurs classes :
  - en utilisant la propriété `rdf:type`
  - en utilisant le nom de la classe pour la balise décrivant la ressource en remplacement `rdf:Description` (sérialisation XML)  
*C'est ce que l'on avait vu avec les ensembles, séquences, etc. au cours précédent*
  - en utilisant la propriété `a` (sérialisation N3)

## Classe (rappel) 2 / 2

## Exemple (notation N3)

```

@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix teaching: <https://purl.oclc.org/teaching#>.
@prefix asi_person: <https://asi.insa-rouen.fr/Personnes/>.
@prefix asi_ec: <https://asi.insa-rouen.fr/ECs/>.

asi_person:nicolasMalandain
  rdf:type teaching:Teacher ;
  foaf:name "Nicolas Malandain"^^xsd:string ;
  teaching:school "INSA de Rouen"^^xsd:string ;
  teaching:teach asi_ec:programmationAvancee .

asi_ec:programmationAvancee
  a teaching:Course ;
  teaching:courseTitle "Programmation avancée"@fr ;
  teaching:courseTitle "Object-Oriented Programming (Java)"@en .

```



# RDF Schema

## RDFS

- RDFS est le langage de description des vocabulaires RDF
- Un vocabulaire RDF permet de définir les classes et les propriétés d'un schéma RDF
- Un vocabulaire RDF est décrit en RDF :
  - Un URI représente une classe s'il a la propriété `rdf:type` avec la valeur `rdfs:Class` (la métaclasse des classes)
    - la propriété `rdfs:subClassOf` permet de définir des héritages
  - Un URI représente une propriété s'il a la propriété `rdf:type` avec la valeur `rdf:Property` (la métaclasse des propriétés)
    - la propriété `rdfs:domain` permet de définir le *domain* (si non défini alors toute classe est valide)
    - la propriété `rdfs:range` permet de définir le *range* (si non spécifié alors toute classe ou littéral est valide)
    - la propriété `rdfs:subPropertyOf` permet de définir des héritages
  - On peut utiliser les propriétés : `rdfs:label`, `rdfs:comment`, `rdfs:isDefinedBy`, `rdfs:seeAlso`

## RDF Schema par l'exemple 1 / 2

## RDFS ASI (notation N3)

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix teaching: <https://purl.oclc.org/teaching#>.

teaching:Teacher
  rdf:type rdfs:Class ;
  rdfs:subClassOf foaf:Person .

teaching:school
  rdf:type rdf:Property ;
  rdfs:domain teaching:Teacher ;
  rdfs:range xsd:string .

teaching:teach
  rdf:type rdf:Property ;
  rdfs:domain teaching:Teacher ;
  rdfs:range teaching:Course .

teaching:Course a rdfs:Class .

teaching:courseTitle
  a rdf:Property ;
  rdfs:domain teaching:Course ;
  rdfs:range xsd:string .
```

## RDF Schema par l'exemple 2 / 2

## RDFS ASI (notation RDF/XML)

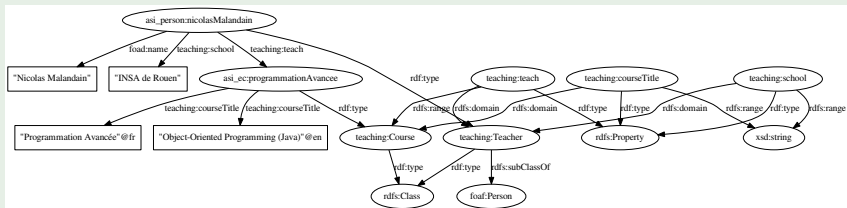
```

<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  <rdf:Description rdf:about="https://purl.oclc.org/teaching#Teacher">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
  </rdf:Description>
  <rdf:Description rdf:about="https://purl.oclc.org/teaching#school">
    <rdfs:domain rdf:resource="https://purl.oclc.org/teaching#Teacher"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
  <rdf:Description rdf:about="https://purl.oclc.org/teaching#courseTitle">
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
    <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
    <rdfs:domain rdf:resource="https://purl.oclc.org/teaching#Course"/>
  </rdf:Description>
  <rdf:Description rdf:about="https://purl.oclc.org/teaching#Course">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  </rdf:Description>
  <rdf:Description rdf:about="https://purl.oclc.org/teaching#teach">
    <rdfs:domain rdf:resource="https://purl.oclc.org/teaching#Teacher"/>
    <rdfs:range rdf:resource="https://purl.oclc.org/teaching#Course"/>
    <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  </rdf:Description>
</rdf:RDF>

```

# Remarques

- Du fait que les instances et les vocabulaires soient créés indépendamment, il se peut qu'ils ne soient pas « valides »
- Les instances et les vocabulaires sont au même niveau, ils sont tous les deux représentés par des triplets RDF



# Différences entre RDF et modélisation/programmation orienté-objet

## POO (Cf. [Gen08])

- Une classe contient des attributs (typés)
- Toutes les instances de la classe ont une valeur pour ces attributs
- Si deux classes ont un attribut de même nom, ces attributs n'ont rien en commun : les types peuvent être différents, aucun lien entre les deux

## RDF

- Une ressource peut être (ou pas) « instance » de plusieurs classes
- Les classes et les propriétés sont définies séparément
- Une ressource peut avoir (ou pas) une valeur pour les propriétés
- Une instance peut ne pas être « valide »

## RDFS définit en RDFS :-) 1 / 6

## Classe rdfs:Resource

- Tout ce qui est décrit en RDF est une ressource instance de la classe rdfs:Resource (ou d'une de ses sous-classes)
- Toute classe est sous classe de rdfs:Resource
- rdfs:Resource est une instance de rdfs:Class

```
<rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Resource">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#"/>
  <rdfs:label>Resource</rdfs:label>
  <rdfs:comment>The class resource, everything.</rdfs:comment>
</rdfs:Class>
```

```
rdfs:Resource a rdfs:Class;
  rdfs:comment "The class resource, everything.";
  rdfs:isDefinedBy rdfs:;
  rdfs:label "Resource".
```

## RDFS définit en RDFS :-) 2 / 6

## Métaclasse rdfs:Class

- C'est la classe de toutes les classes
- Elle est instance d'elle même

```
<rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Class">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#" />
  <rdfs:label>Class</rdfs:label>
  <rdfs:comment>The class of classes.</rdfs:comment>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource" />
</rdfs:Class>
```

```
rdfs:Class a rdfs:Class;
  rdfs:comment "The class of classes
  .";
  rdfs:isDefinedBy rdfs:;
  rdfs:label "Class";
  rdfs:subClassOf rdfs:Resource.
```

```
a a rdf:Property;
  rdfs:comment "The subject is an
  instance of a class.";
  rdfs:domain rdfs:Resource;
  rdfs:isDefinedBy rdf:;
  rdfs:label "type";
  rdfs:range rdfs:Class.
```

## RDFS définit en RDFS :-) 3 / 6

Classe `rdfs:Literal`

- C'est la classe de tous littéraux
- C'est une sous classe `rdfs:Resource` et instance de `rdfs:Class`

```
<rdfs:Class rdf:about="http://www.w3.org/2000/01/rdf-schema#Literal">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#" />
  <rdfs:label>Literal</rdfs:label>
  <rdfs:comment>The class of literal values, eg. textual strings and integers.</
    rdfs:comment>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource" />
</rdfs:Class>
```

- Il y a aussi :
  - La métaclasse `rdfs:Datatype` qui est la classe des types de données. C'est une instance et sous classe de `rdfs:Class`
  - La classe `rdfs:XMLLiteral` qui est la classe de tous les littéraux exprimés en XML. C'est une sous classe de `rdfs:Literal` et instance de `rdfs:Datatype`



## RDFS définit en RDFS :-) 4 / 6

## La classe rdfs:Property

- C'est la classe de toutes les propriétés
- C'est une instance de rdfs:Class

```
<rdfs:Class rdf:about="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#"/>
  <rdfs:label>Property</rdfs:label>
  <rdfs:comment>The class of RDF properties.</rdfs:comment>
  <rdfs:subClassOf rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
</rdfs:Class>
```

- Lorsque l'on instancie cette rdfs:Property, on peut renseigner :
  - rdfs:domain : La source de la propriété
  - rdfs:range : La valeur de cette propriété
- Si non renseigné, alors tout est accepté

## RDFS définit en RDFS :-) 5 / 6

## Exemples de création de propriétés

```
<rdf:Property rdf:about="http://www.w3.org/2000/01/rdf-schema#subClassOf">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#" />
  <rdfs:label>subClassOf</rdfs:label>
  <rdfs:comment>The subject is a subclass of a class.</rdfs:comment>
  <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class" />
  <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class" />
</rdf:Property>
```

```
<rdf:Property rdf:about="http://www.w3.org/2000/01/rdf-schema#comment">
  <rdfs:isDefinedBy rdf:resource="http://www.w3.org/2000/01/rdf-schema#" />
  <rdfs:label>comment</rdfs:label>
  <rdfs:comment>A description of the subject resource.</rdfs:comment>
  <rdfs:domain rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource" />
  <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal" />
</rdf:Property>
```

## RDFS définit en RDFS :-) 6 / 6

## Quelques propriétés

Property name	domain	range
rdf:type	rdfs:Resource	rdfs:Class
rdfs:subClassOf	rdfs:Class	rdfs:Class
rdfs:subPropertyOf	rdf:Property	rdf:Property
rdfs:domain	rdf:Property	rdfs:Class
rdfs:range	rdf:Property	rdfs:Class
rdfs:label	rdfs:Resource	rdfs:Literal
rdfs:comment	rdfs:Resource	rdfs:Literal
rdfs:seeAlso	rdfs:Resource	rdfs:Resource
rdfs:isDefinedBy	rdfs:Resource	rdfs:Resource
rdf:value	rdfs:Resource	rdfs:Resource
rdf:subject	rdf:Statement	rdfs:Resource

# Un exemple de RDFS : Le Dublin Core

```

<rdf:Property rdf:about="http://purl.org/dc/elements/1.1/title">
  <rdfs:label xml:lang="en-US">Title</rdfs:label>
  <rdfs:comment xml:lang="en-US">A name given to the resource.</rdfs:comment>
  <rdfs:isDefinedBy rdf:resource="http://purl.org/dc/elements/1.1/">
  <dcterms:issued>1999-07-02</dcterms:issued>
  <dcterms:modified>2008-01-14</dcterms:modified>
  <rdf:type rdf:resource="http://www.w3.org/1999/02/22-rdf-syntax-ns#Property"/>
  <dcterms:hasVersion rdf:resource="http://dublincore.org/usage/terms/history/#title-006"/>
  <skos:note xml:lang="en-US">A second property with the same name as this property has been declared in the dcterms: namespace (http://purl.org/dc/terms/). See the Introduction to the document "DCMI Metadata Terms" (http://dublincore.org/documents/dcmi-terms/) for an explanation.</skos:note>
</rdf:Property>

```

## OWL

- « Peut être vu comme une extension du RDFS » : pouvoir d'expression plus grand, par exemple :
  - préciser la cardinalité des propriétés (min, max et exact)
  - caractériser les propriétés (transitive, réflexive, symétrique, fonctionnelle, etc.)
  - définir des classes à partir d'autres classes (vision ensembliste : intersection, union) ou par extension
- Plus le pouvoir d'expression est grand, plus la vérification de la « validité » des instances est difficile
- Plusieurs langages avec un pouvoir d'expressivité différent
  - OWL 1 : OWL Lite  $\subset$  OWL DL  $\subset$  OWL Full
  - OWL 2 : OWL EL (polynomial), QL (proche BDR), RL (règles d'inférences en temps polynomial), DL (la plus expressive)

## Caractéristiques générales

- Comme en RDF, une ressource/individu est instance d'une classe avec la possibilité de mettre les individus en relation
- Réutilisation des propriétés `rdfs:subClassof`, `rdfs:Property`, `rdfs:subProperty`, `rdfs:range` et `rdfs:domain`

## Création et Caractéristiques des classes

- Création d'une nouvelle classe à l'aide de `owl:Class`
- Possibilité de définir une classe comme étant :
  - l'union de deux classes : `owl:unionOf`
  - l'union disjointes de plusieurs classes : `owl:disjointUnionOf`
  - l'intersection de deux classes : `owl:intersectionOf`
  - l'énumération des instances d'une classe : `owl:oneOf`
  - le complément d'une classe : `owl:complementOf`
  - la disjonction de plusieurs classes : `owl:AllDisjointClasses` (spécification par instanciation)

### Création d'une propriété

- Propriété dont le *range* est une ressource : owl:ObjectProperty
- Propriété dont le *range* est un type simple : owl:DatatypeProperty
- Propriété permettant de décrire la ressource owl:AnnotationProperty

## Caractérisation des propriétés

- owl:inverseOf permet d'exprimer : si  $P(x, y)$  alors  $P'(y, x)$
- owl:TransitiveProperty permet d'exprimer : si  $P(x, y)$  et  $P(y, z)$  alors  $P(x, z)$
- owl:SymmetricProperty (owl:AssymmetricProperty) permet d'exprimer : si  $P(x, y)$  alors  $P(y, x)$
- owl:propertyDisjointWith permet d'exprimer que deux propriétés ne peuvent pas être utilisées en même temps
- owl:ReflexiveProperty (owl:IrreflexiveProperty) permet d'exprimer : si  $P$  est reflexive alors si  $P(x, x)$
- owl:FunctionalProperty permet d'exprimer : si  $P(x, y)$  et  $P(x, z)$  alors  $y = z$
- owl:InverseFunctionalProperty permet d'exprimer : si  $P(x, z)$  et  $P(y, z)$  alors  $x = y$
- owl:hasKey permet d'exprimer : si  $P(x, z_1)$  et  $P(y, z_2)$  et  $z_1$  et  $z_2$  ont la même valeur alors  $x = y$
- owl:propertyChainAxiom permet d'exprimer : si  $P_1(x_1, x_2)$  et  $P_2(x_2, x_3)$  et ... et  $P_n(x_n, x_{n+1})$  alors  $Q(x_1, x_{n+1})$



## Equivalence

- Classe : owl:equivalentClass
- Propriété : owl:equivalentProperty
- Individu : owl:SameIndividual
- Ressource : owl:sameAs

## Différence

- Individus : owl:DifferentIndividuals
- Ressource : owl:differentFrom

## Restriction

- owl:Restriction permet de créer des classes anonymes
- On l'utilise avec rdfs:subClassOf ou owl:equivalentClass

## Restriction des cibles d'une propriété

- owl:allValuesFrom : si  $C$  est associé à  $P$  alors si  $P(x, y)$  alors  $\forall y, y \in C$
- owl:someValuesFrom : si  $C$  est associé à  $P$  alors si  $P(x, y)$  alors  $\exists y, y \in C$
- owl:hasValue

## Restriction de la cardinalité d'une propriété

- owl:minCardinality
- owl:maxCardinality
- owl:qualifiedCardinality

## Classe

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

<owl:Class rdf:ID="Region" />

<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="#food;PotableLiquid" />
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasMaker" />
      <owl:cardinality rdf:datatype="#xsd;nonNegativeInteger">1</owl:cardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
  ...

  <rdfs:label xml:lang="en">wine</rdfs:label>
  <rdfs:label xml:lang="fr">vin</rdfs:label>
</owl:Class>
```

La propriété *hasMaker* définie de manière générale est restreinte pour la classe *Wine* à être de cardinalité 1

## Propriété symétrique

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

## Propriété transitive

```
<owl:ObjectProperty rdf:ID="locatedIn">  
  <rdf:type rdf:resource="&owl;TransitiveProperty" />  
  <rdfs:domain rdf:resource="http://www.w3.org/2002/07/owl#Thing" />  
  <rdfs:range rdf:resource="#Region" />  
</owl:ObjectProperty>
```

```
<Region rdf:ID="MedocRegion">  
  <locatedIn rdf:resource="#BordeauxRegion" />  
</Region>
```

```
<Region rdf:ID="BordeauxRegion">  
  <locatedIn rdf:resource="#FrenchRegion" />  
</Region>
```

## Création par intersection

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

## Restriction des propriétés

```
<owl:Class rdf:about="#WhiteLoire">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#madeFromGrape" />
      <owl:allValuesFrom>
        <owl:Class>
          <owl:oneOf rdf:parseType="Collection">
            <owl:Thing rdf:about="#CheninBlancGrape" />
            <owl:Thing rdf:about="#PinotBlancGrape" />
            <owl:Thing rdf:about="#SauvignonBlancGrape" />
          </owl:oneOf>
        </owl:Class>
      </owl:allValuesFrom>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

# Raisonneur 1 / 4

## Qu'est ce ?

- Un raisonneur est un programme qui permet d'inférer de nouveaux faits et de vérifier la cohérence

## Exemple : définition des classes et des propriétés

```
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dom: <https://purl.oclc.org/a#> .

dom:A
  rdf:type rdfs:Class .

dom:B
  rdf:type rdfs:Class .

dom:pAB
  rdf:type owl:ObjectProperty;
  rdfs:domain dom:A ;
  rdfs:range dom:B .

dom:pAA
  rdf:type owl:ObjectProperty , owl:TransitiveProperty ;
  rdfs:domain dom:A ;
  rdfs:range dom:A .
```

## Exemple : les instances

```
dom:iA1
  rdf:type dom:A ;
  dom:pAA dom:iA2 .

dom:iA2
  rdf:type dom:A ;
  dom:pAA dom:iA3 .

dom:iA3
  rdf:type dom:A .

dom:iB1
  rdf:type dom:B ;
  dom:pAB dom:iB2 .

dom:iB2
  rdf:type dom:B .
```

## Raisonneur 3 / 4

## Résultats du raisonneur (Hermit)

```
dom:pAA
  rdf:type owl:ObjectProperty ,owl:TransitiveProperty .

dom:A
  rdf:type owl:Class .

dom:B
  rdf:type owl:Class .

dom:iA1
  rdf:type owl:NamedIndividual , dom:A ;
  dom:pAA dom:iA2 , dom:iA3 .

dom:iA2
  rdf:type owl:NamedIndividual , dom:A ;
  dom:pAA dom:iA3 .

dom:iA3
  rdf:type owl:NamedIndividual , dom:A .

dom:iB1
  rdf:type owl:NamedIndividual , dom:A , dom:B ;
  dom:pAB dom:iB2 .

dom:iB2
  rdf:type owl:NamedIndividual , dom:B .
```



## Comment ?

- Grâce à la programmation logique :

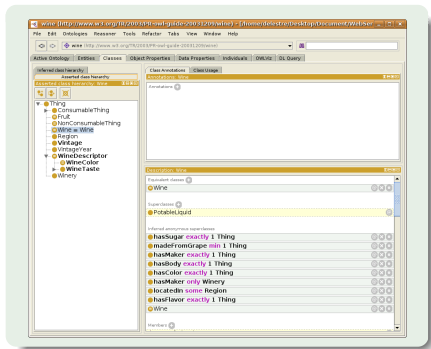
*« La programmation logique est une forme de programmation qui définit les applications à l'aide d'un ensemble de faits élémentaires les concernant et de règles de logique leur associant des conséquences plus ou moins directes. Ces faits et ces règles sont exploités par un démonstrateur de théorème ou moteur d'inférence, en réaction à une question ou requête. » (Wikipédia 2015)*

- Il existe plusieurs théories logiques : logique des propositions, logique des prédicats, logique modale, etc.
- Plus leur pouvoir d'expression est grand plus la complexité des algorithmes d'inférence est grande
- Une ontologie OWL peut être représentée en logiques des descriptions (peut être vue comme une restriction de logique des prédicats)

## Protégé : un Éditeur d'ontologie 1 / 2

## Protégé

- Logiciel open source multi-plateforme (java) : <http://protege.stanford.edu/>
- Permet de créer dans les classes, attributs, objets d'une ontologie
- Permet d'interroger l'ontologie :
  - SPARQL pour la famille 3.x (sans la gestion des relations transitives)
  - DL Query pour la famille 4.x
- Permet d'exporter sous plusieurs format : RDF(S), OWL ou XML Schema



Voir aussi

<http://owl.cs.manchester.ac.uk/browser/manage/>

DL Query (cf. [http://www.co-ode.org/resources/reference/manchester\\_syntax/](http://www.co-ode.org/resources/reference/manchester_syntax/))

- *The Manchester OWL Syntax*
- Extrait de la documentation :

OWL	DL Symbol	M OWL S	Example
someValuesFrom	$\exists$	some	hasChild some Man
allValuesFrom	$\forall$	only	hasSibling only Woman
hasValue	$\in$	value	hasCountryOfOrigin value Englan
minCardinality	$\geq$	min	hasChild min 3
cardinality	$=$	exactly	hasChild exactly 3
maxCardinality	$\leq$	max	hasChild max 3

## Exemples sur la base des vins

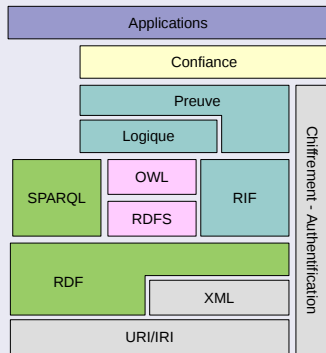
- Obtenir des vins blancs : `Wine and hasColor value White`
- Obtenir des vins blancs français : `Wine and hasColor value White and locatedIn value FrenchRegion`
- Quel repas peut on manger avec du vin blanc ? `MealCourse and hasDrink some (Wine and hasColor value White)`

# Conclusion

## Ce que nous n'avons pas vu dans ce cours

- Comment documenter et gérer la vie d'une ontologie ?
- Qu'est ce la démonstration logique ?
- Quelles règles utilisent les moteurs d'inférences ?
- Comment les ontologies peuvent publier des règles sur leur domaine (RuleML, RIF, SWRL, etc.) ?

## Pile du web sémantique



# Références

---

- [CBT04] J. Charlet, B. Bachimont, and R. Troncy.  
*Le Web Sémantique*, chapter Ontologie pour le Web sémantique.  
Cépaduès - Éditions, 2004.
- [Gen08] D. Genest.  
Cours sur le web sémantique, 2008.  
<http://www.info.univ-angers.fr/pub/genest/enseignement/index.html>.
- [GFZC15] Fabien Gandon, Catherine Faron-Zucker, and Olivier Corby.  
Web sémantique et web de données.  
[https://www.france-universite-numerique-mooc.fr/courses/inria/41002/Trimestre\\_1\\_2015/info](https://www.france-universite-numerique-mooc.fr/courses/inria/41002/Trimestre_1_2015/info), Mars 2015.