Web sémantique et ontologies

Yannick Prié
Département Informatique – Faculté des Sciences et Technologies
Université Claude Bernard Lyon 1
2010-2011

Objectifs de ce module

- Comprendre ce qu'est le web sémantique / web de données
- Comprendre ce qu'est RDF
- Comprendre ce qu'est un vocabulaire / une ontologie
- Concevoir une ontologie et l'utiliser

Moyens

- Cours magistral
- TP découverte RDFa, Protege, RDFS et OWL
- TP conception d'ontologie

Plan

- Web sémantique / web de données
- RDF
- RDF schéma
- OWL

D'abord

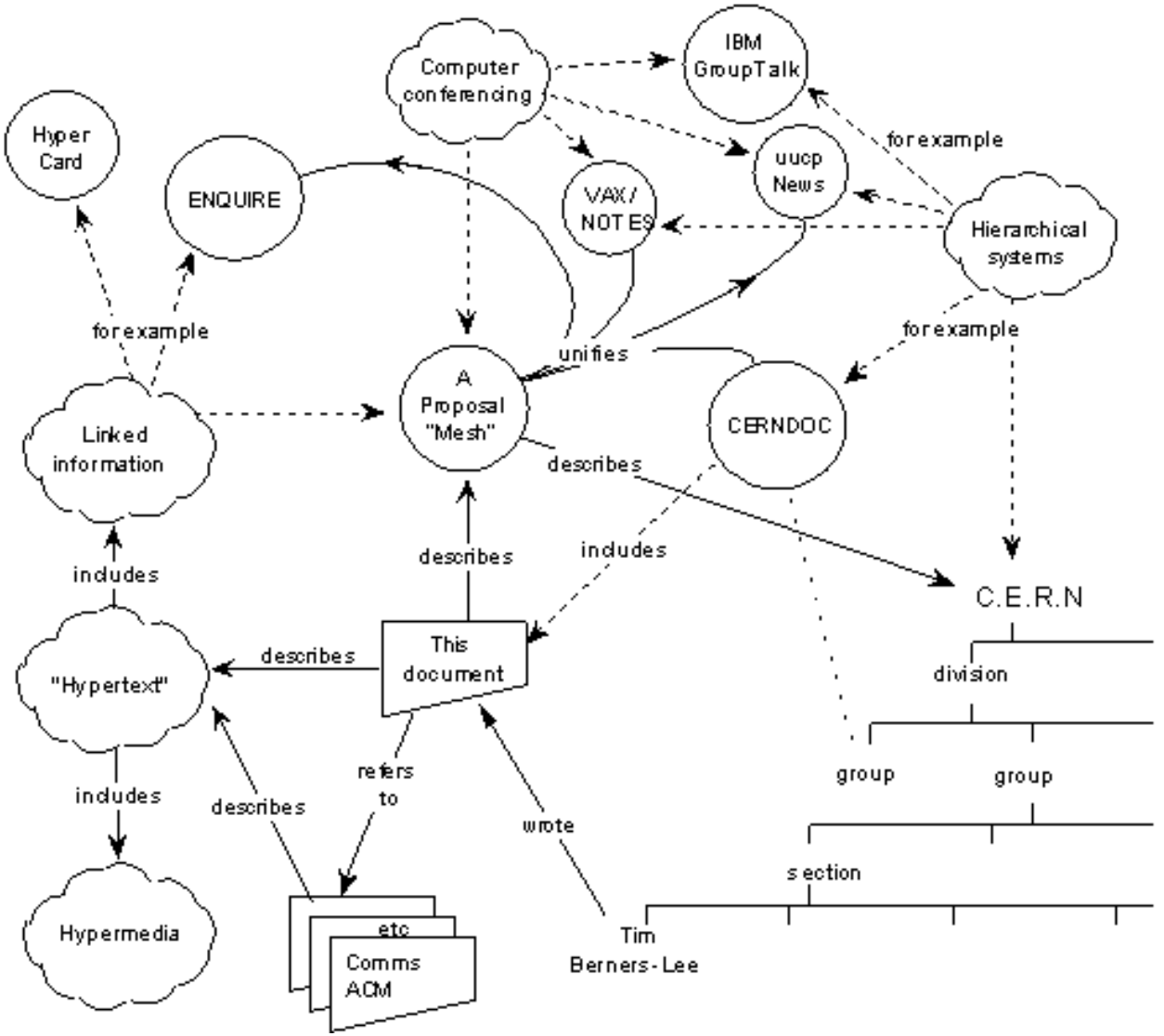
- IC et connaissances 

Plan

- **Web sémantique / web de données**
- RDF
- RDF schéma
- OWL

http://www.w3.org/History/1989/proposal.html

Proposition originale pour le web



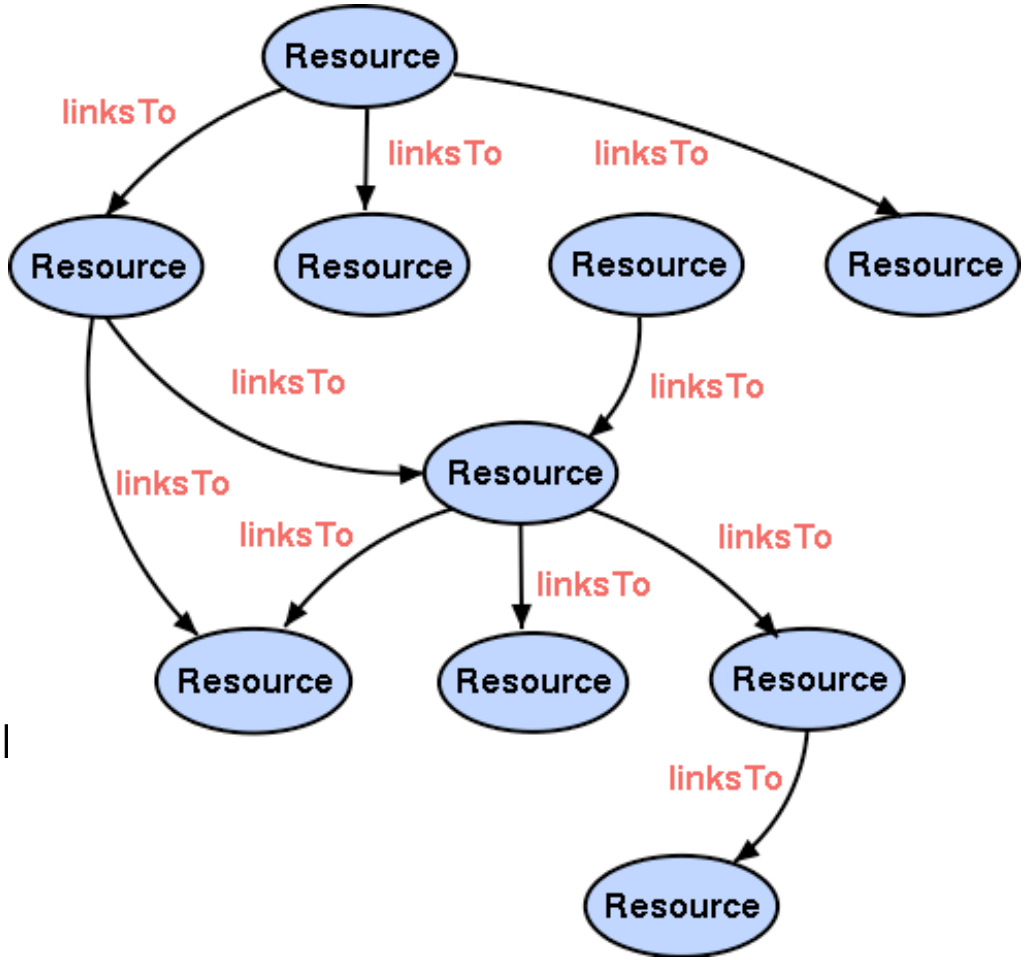
Web « classique » (1/2)

- Essentiellement
  - des pages web
  - décrites en (X)HTML
    - structures documentaires orientées description des documents
  - disponibles à des URLs sur le Web
  - avec des liens entre les pages web
    - tutu
- Il n'y a pas de beaucoup de choses pour les machines
  - contenus textuels des pages
    - → indexation / recherche d'informations
  - structures documentaires peu significatives
    - → parties, paragraphes, images
  - très peu de métadonnées
  - un seul type de lien : « la page X est en relation avec la page Y »

Web « classique » (2/2)

http://www.w3.org/2004/Talks/0120-semweb-umich/

- Ressources
  - identifiées par des URI's
  - non typées
- Liens
  - href, src…
  - limités, non-descriptifs
- Humain
  - Un monde excitant – la sémantique des ressources est cependant récupérée des contenus
- Machine
  - Très peu d'information disponibl – la signification des liens ne devient évidente que depuis le contexte autour des ancres



Tim Berners-Lee, 1st World Wide Web Conference, Geneva, May 1994

To a computer, the Web is a flat, boring world, devoid of meaning. This is a pity, as in fact documents on the Web describe real objects and imaginary concepts, and give particular relationships between them. For example, a document might describe a person. The title document to a house describes a house and also the ownership relation with a person…. Adding semantics to the Web involves two things: allowing documents which have information in machine-readable forms, and allowing links to be created with relationship values. Only when we have this extra level of semantics will we be able to use computer power to help us exploit the information to a greater extent than our own reading.

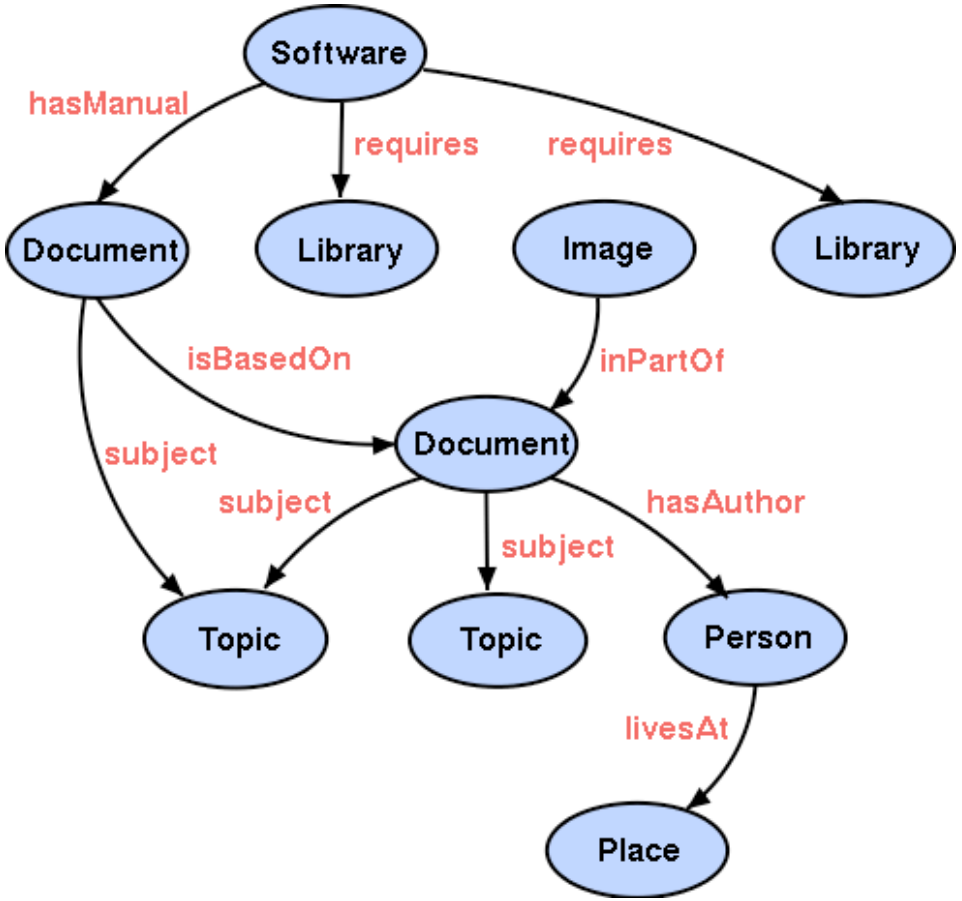
Tim Berners-Lee, 1st World Wide Web Conference, Geneva, May 1994

To a computer, the Web is a flat, boring world, devoid of meaning. This is a pity, as in fact documents on the Web describe real objects and imaginary concepts, and give particular relationships between them. For example, a document might describe a person. The title document to a house describes a house and also the ownership relation with a person… ***Adding semantics to the Web involves two things: allowing documents which have information in machine-readable forms, and allowing links to be created with relationship values. Only when we have this extra level of semantics will we be able to use computer power to help us exploit the information to a greater extent than our own reading.***

Web sémantique

http://www.w3.org/2004/Talks/0120-semweb-umich/

- Resources
  - Identifiées globalement par des URI's ou à portée locale (Blank) Extensible Relational
  - Links: identifiés par des URIs. Extensible Relational
- Humain
  - Un monde encore plus riche, une expérience utilisateur plus excitante (!)
- Machine
  - Plus d'information traitable automatiquement
- Humains et machines
  - Travaillent, apprennent et échangent plus facilement



Web sémantique : définition

- De multiples définitions, à peu près cohérentes
- http://www.uen.org/core/edtech/glossary.shtml
  - *The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries. It is a collaborative effort led by W3C with participation from a large number of researchers and industrial partners. It is based on the Resource Description Framework (RDF), which integrates a variety of applications using XML for syntax and URIs for naming. A concept proposed by World Wide Web inventor Tim Berners-Lee*.

Plan

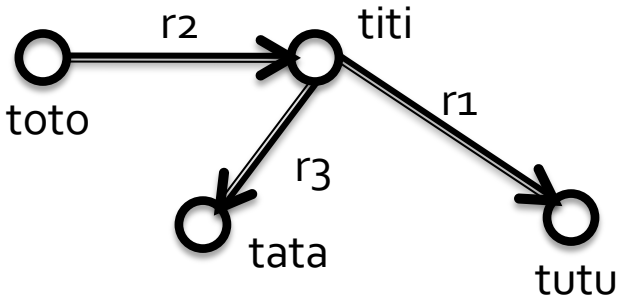
- Web sémantique / web de données
- **RDF**
- RDF schéma
- OWL

Rappel sur les URI

- Uniform Resource Identifier
- Séquence de caractères qui identifie une ressource abstraite ou physique, telle que définie dans la RFC3986.
- Permet d'identifier n'importe quoi de façon unique et non ambiguë
  - pages web, personnes, documents, livres, centres d'intérêts, bâtiments, etc.
- Remarque
  - on peut avoir une URI qui représente une personne, sans que cette URI soit une URL ou un document à propos de cette personne
    - http://liris.cnrs.fr/yannick.prie - ma page
    - http://yannickprie.net/me - moi
      - URI déréférençable sur HTTP →qui sert de pointeur vers une ressource
  - tout est question de convention

RDF : carte d'identité

- Ressource Description Framework
- Recommandation W3C depuis 2009
- Modèle de données pour représenter des graphes orientés étiquetés avec des URI
  - orientés : les relations sont orientées 
  - labellisés : les nœuds et les relations ont des étiquettes



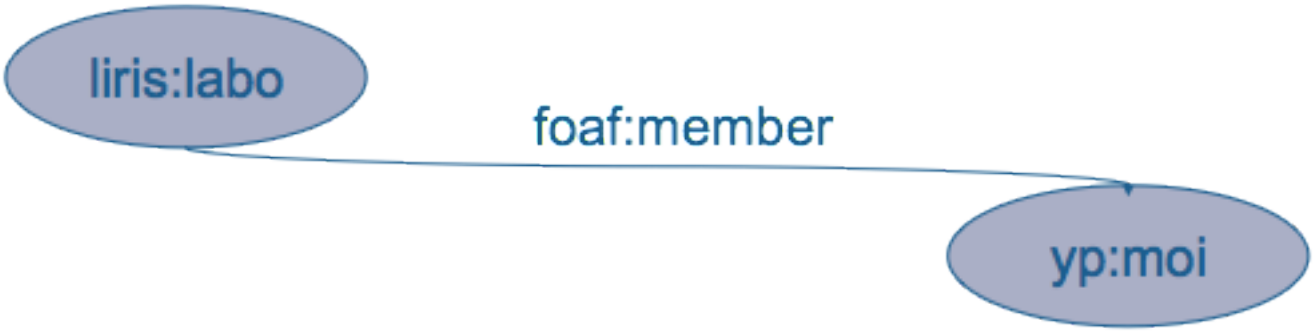
RDF : notion de triplet

- <sujet> <prédicat> <objet>
- Trois types de termes
  - URI
  - Nœuds vierges (blank)
  - Littéraux
    - chaînes de caractères, nombres, dates, etc. (cf. XML schema)

RDF : triplet

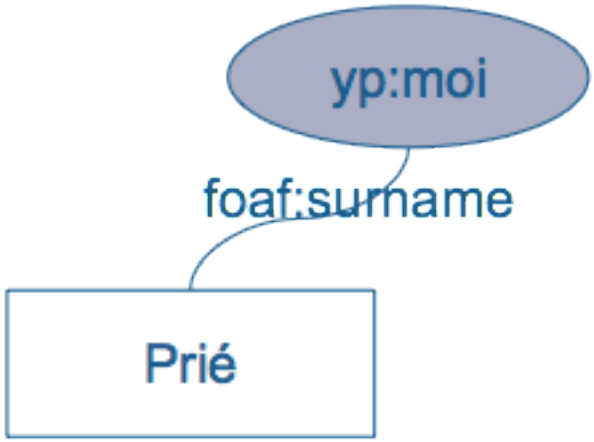

http://liris.cnrs.fr/labo
http://xmlns.com/foaf/0.1/member
http://liris.cnrs.fr/yannick.prie/moi

RDF : espaces de noms



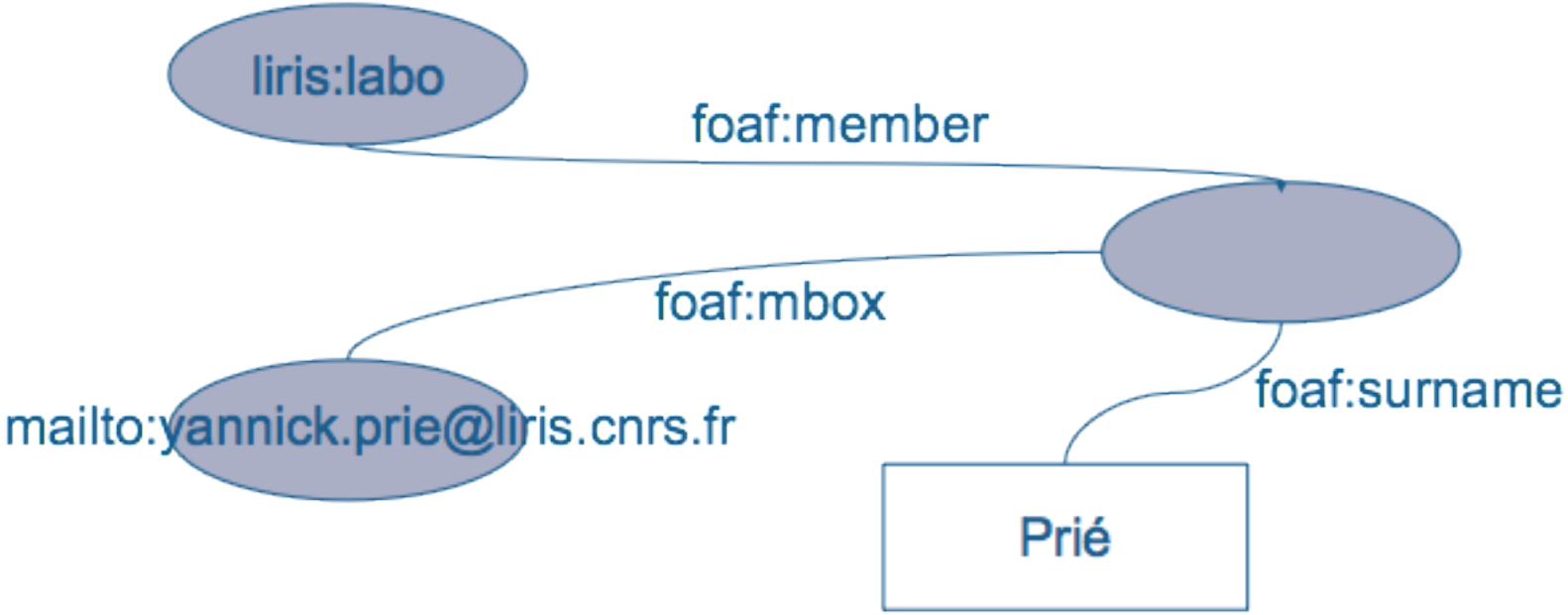
foaf: http://xmlns.com/foaf/0.1/
liris: http://liris.cnrs.fr/
yp: http://liris.cnrs.fr/yannick.prie/

RDF : littéral

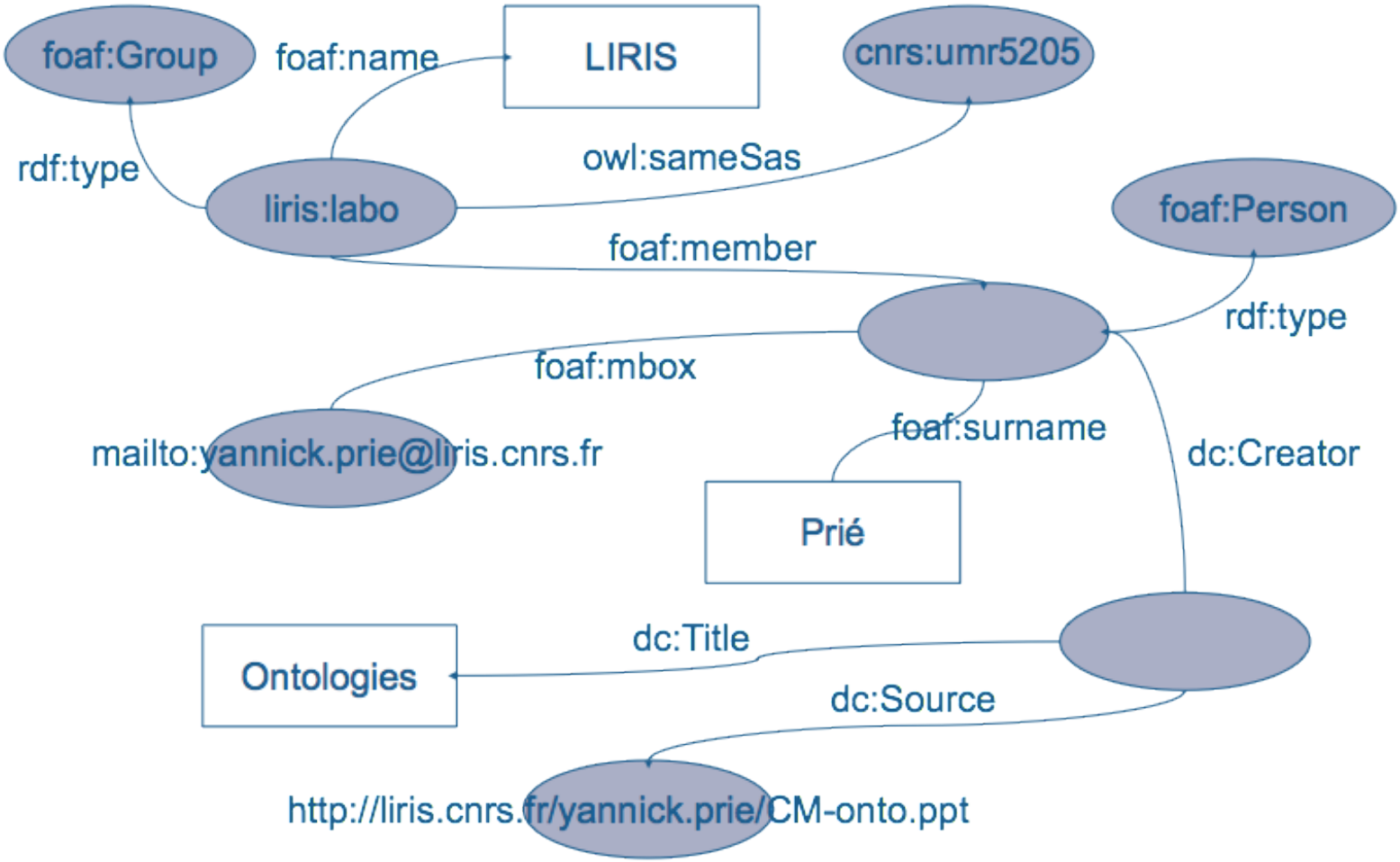


RDF : nœud vierge

RDF : exemple


liris:labo
foaf:member
foaf:mbox
mailto:yannick.prie@liris.cnrs.fr
foaf:surname
Prié

RDF : exemple plus complet



Sérialisation RDF : RDF/XML

- Verbeux, difficile à lire, le plus utilisé
- http://www.w3.org/TR/rdf-syntax-grammar/

```
<?xml version="1.0"?>
<rdf:RDF    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:dc="http://purl.org/dc/elements/1.1/"
            xmlns:ex="http://example.org/stuff/1.0/">
<rdf:Description   rdf:about="http://www.w3.org/TR/rdf-syntax-grammar"
                   dc:title="RDF/XML Syntax Specification (Revised)">
    <ex:editor>
        <rdf:Description ex:fullName="Dave Beckett">
            <ex:homePage rdf:resource="http://purl.org/net/dajobe/" />
        </rdf:Description>
    </ex:editor>
</rdf:Description>
</rdf:RDF>
```

Sérialisation RDF : N3/Turtle

- Plus facile à lire pour les humains
- http://www.w3.org/TeamSubmission/turtle/

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix dc: <http://purl.org/dc/elements/1.1/> .
@prefix ex: <http://example.org/stuff/1.0/> .

<http://www.w3.org/TR/rdf-syntax-grammar>
    dc:title "RDF/XML Syntax Specification (Revised)" ;
    ex:editor [
        ex:fullname "Dave Beckett";
        ex:homePage <http://purl.org/net/dajobe/>
    ] .
```

Sérialisation RDF : RDFa

- Le RDF est inséré dans du XHTML
  - Une page web lisible par les humains et les machines
- http://www.w3.org/TR/rdfa-syntax/

- Quelques attributs
  - `about` : URI spécifiant la ressource décrite par les métadonnées ; en son absence, il s'agit de l'élément en cours
  - `href`, `src`, `resource` : spécifiant l'objet d'une propriété
  - `rel`, `rev` : spécifiant une relation avec une autre ressource
  - `property` : spécifiant une propriété pour le contenu d'un élément, également décrit avec content
  - `typeof` : attribut optionnel spécifiant le type RDF d'un sujet
- Exemple
  - Voir TP1 !

Langage de requête

- SPARQL
- http://www.w3.org/TR/rdf-sparql-query/

```
PREFIX foaf:    <http://xmlns.com/foaf/0.1/>
SELECT ?name ?mbox
WHERE  {
           ?x foaf:name ?name .
           ?x foaf:mbox ?mbox .
       }
```

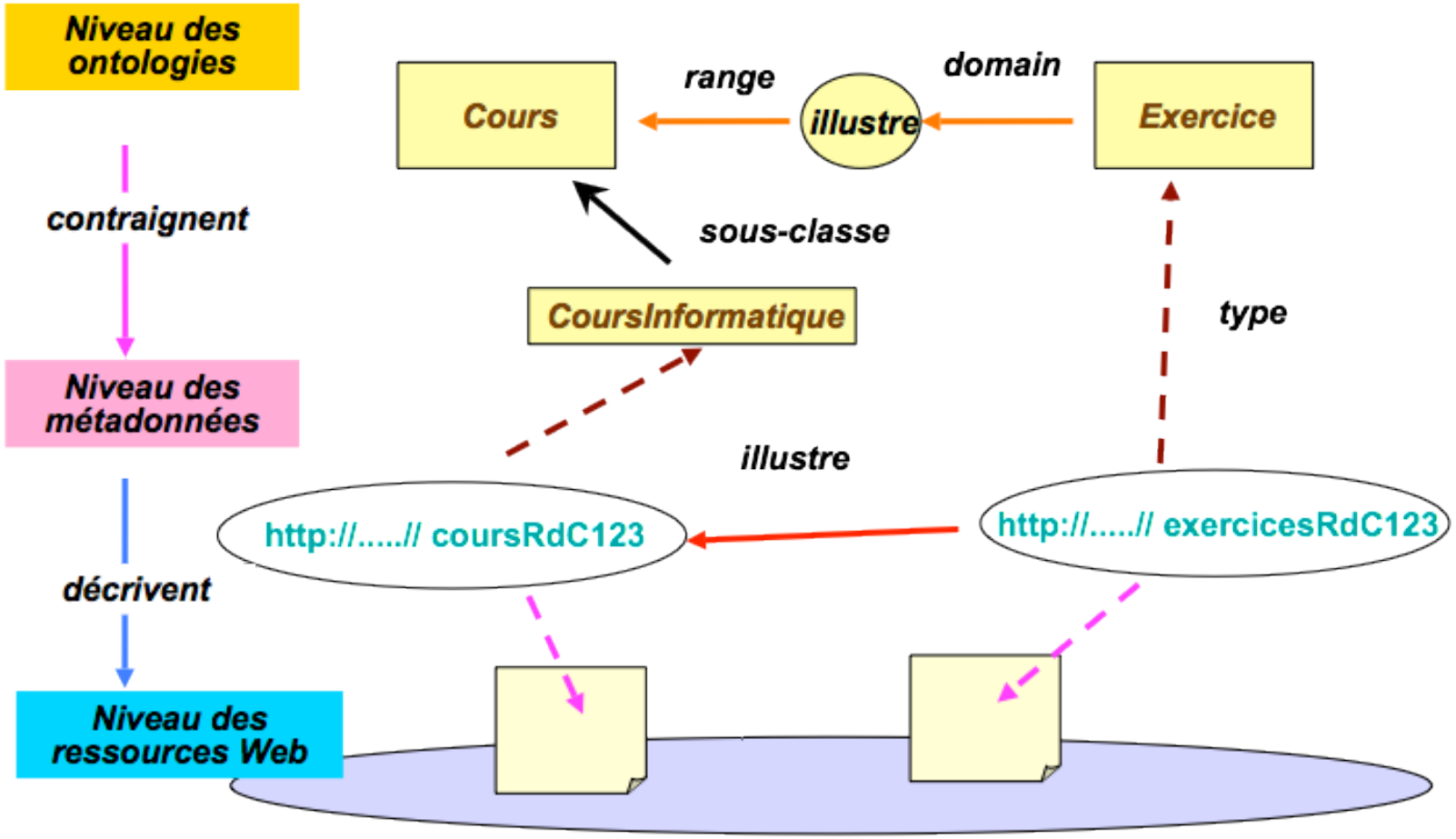
- Fournit un mode d'accès uniforme aux graphes RDF

Schémas : vocabulaires, ontologies

- On sait exprimer n'importe quel graphe avec RDF, que manque-t-il ?
  - pouvoir spécifier des manières de décrire des graphes RDF
  - pouvoir spécifier la signification des graphes RDF
  - pouvoir partager ces significations
- Idée
  - donner plus d'informations abstraites et sémantiques sur les nœuds dans les graphes RDF
  - typer les nœuds RDF
    - leur donner des catégories
  - typer les relations RDF
    - comme des propriétés des noeuds
      - attributs
      - relations

Niveau du vocabulaire – schéma
Niveau des assertions

(d'après Philippe Laublet)



Dans la suite

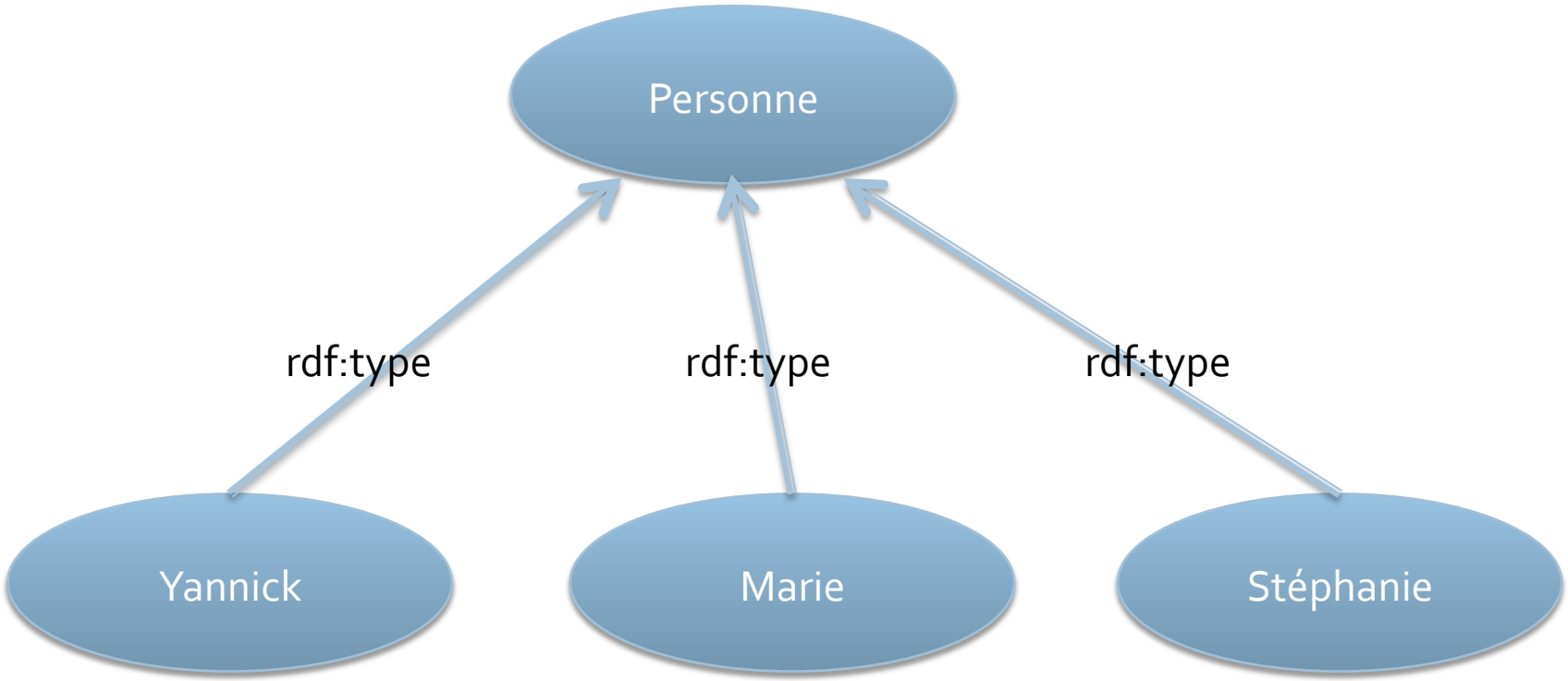
- RDF-schema :
  - pour des vocabulaires basiques
  - pour spécifier des hiérarchies de types et de propriétés
- OWL (Ontology Web Language)
  - pour des vocabulaires plus riches – logiques de description
  - pour spécifier des hiérarchies de concepts, des relations, des axiomes, etc.
- Remarque

  - au sens large du terme, une ontologie est une « spécification partagée d'une conceptualisation » (Gruber)
  - on peut faire des ontologies en RDFS ou OWL

Plan

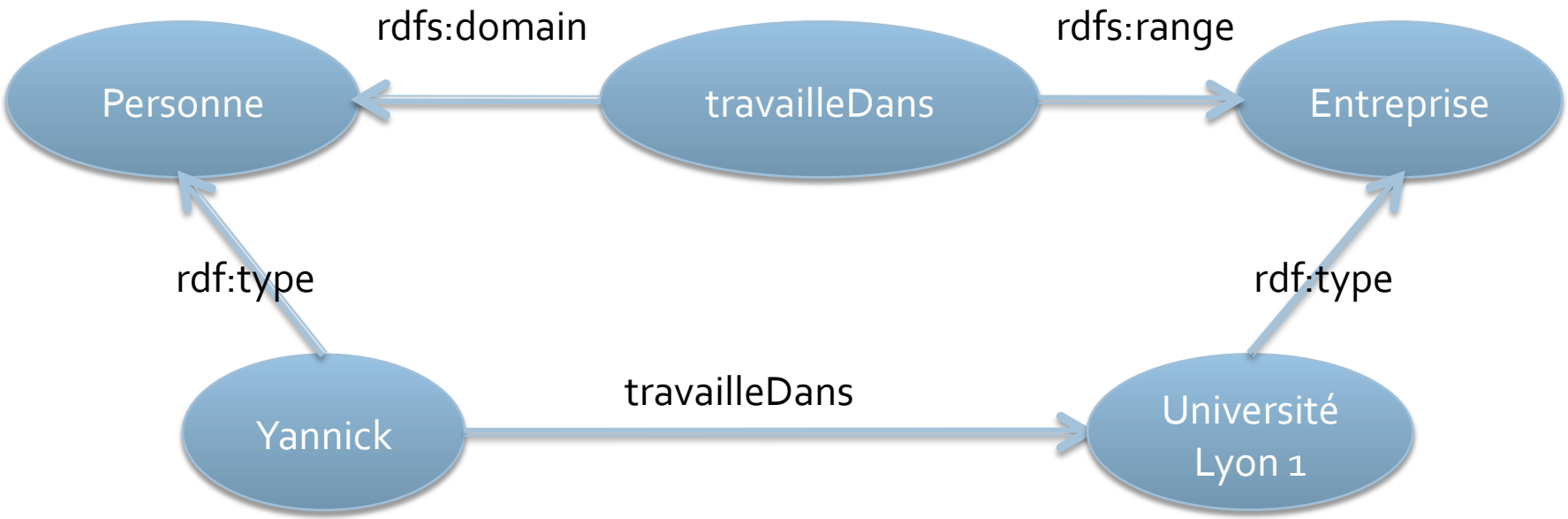
- Web sémantique / web de données
- RDF
- **RDF schéma**
- OWL

Classes et instances

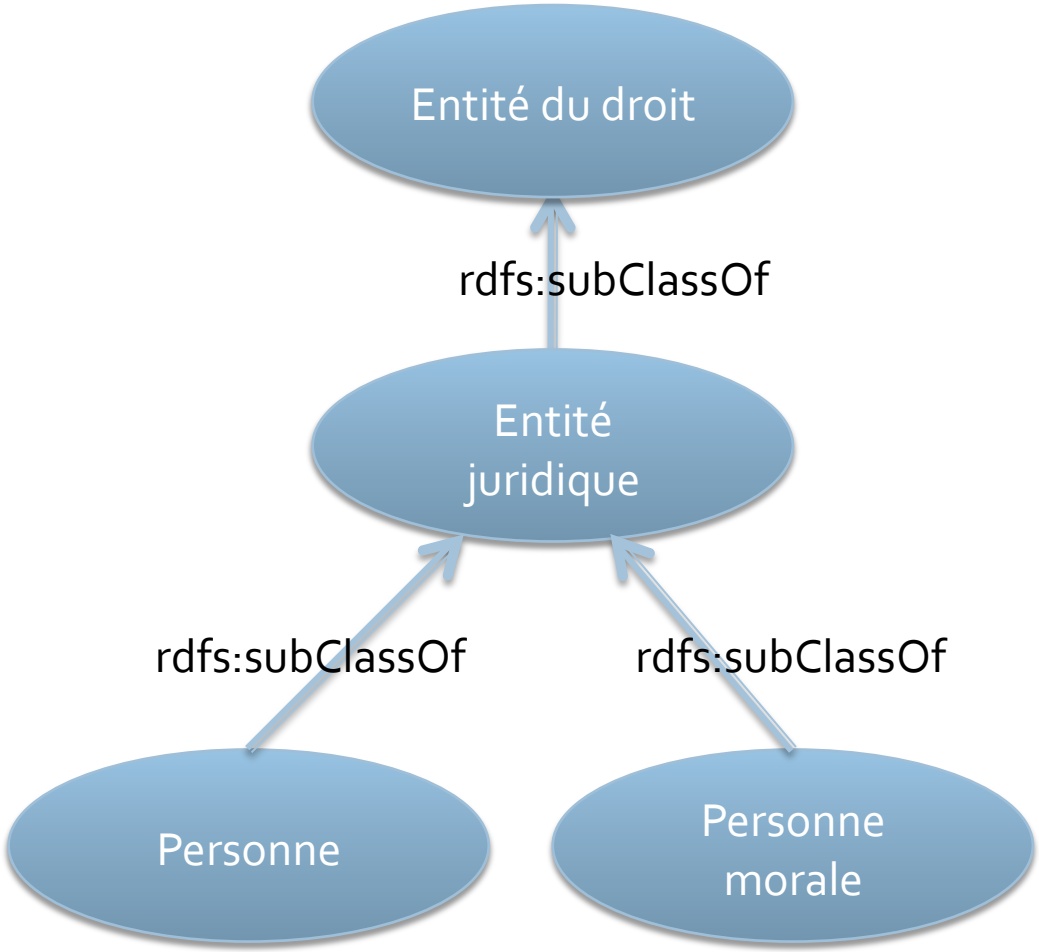


Propriété

- Une propriété RDFs à un domaine (*domain*) et une portée(*range)*

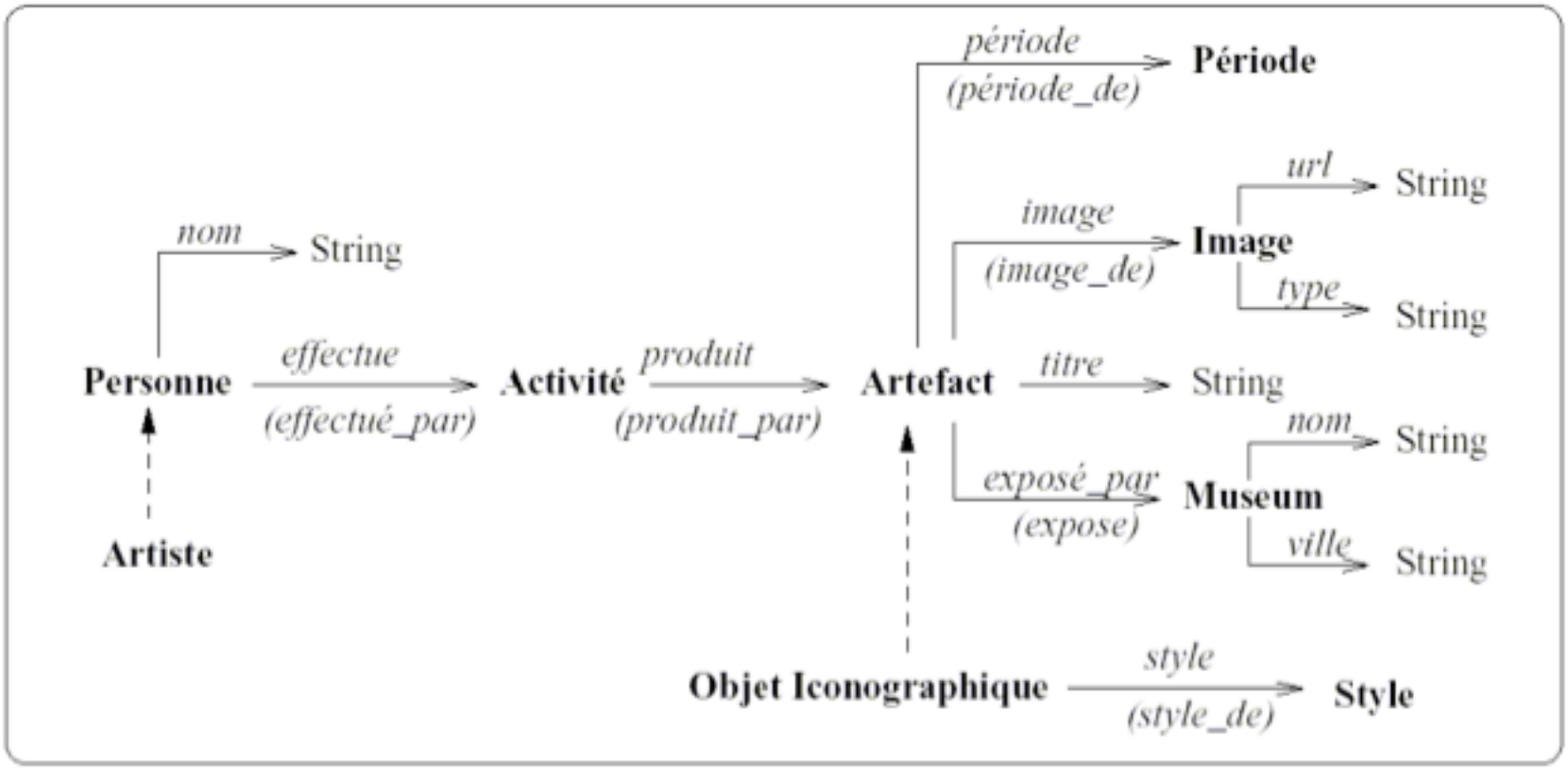


Hiérarchie – subClassOf



- Interprétation
  - Toute instance d'une classe est aussi instance de ses super-classes
  - *Yannick* est une instance de *Entité juridique*
- Relations équivalente :
  - Généralisation (inv. Spécialisation)
- Relations transitives
  - *Personne morale* est une sous classe de *Entité du droit*

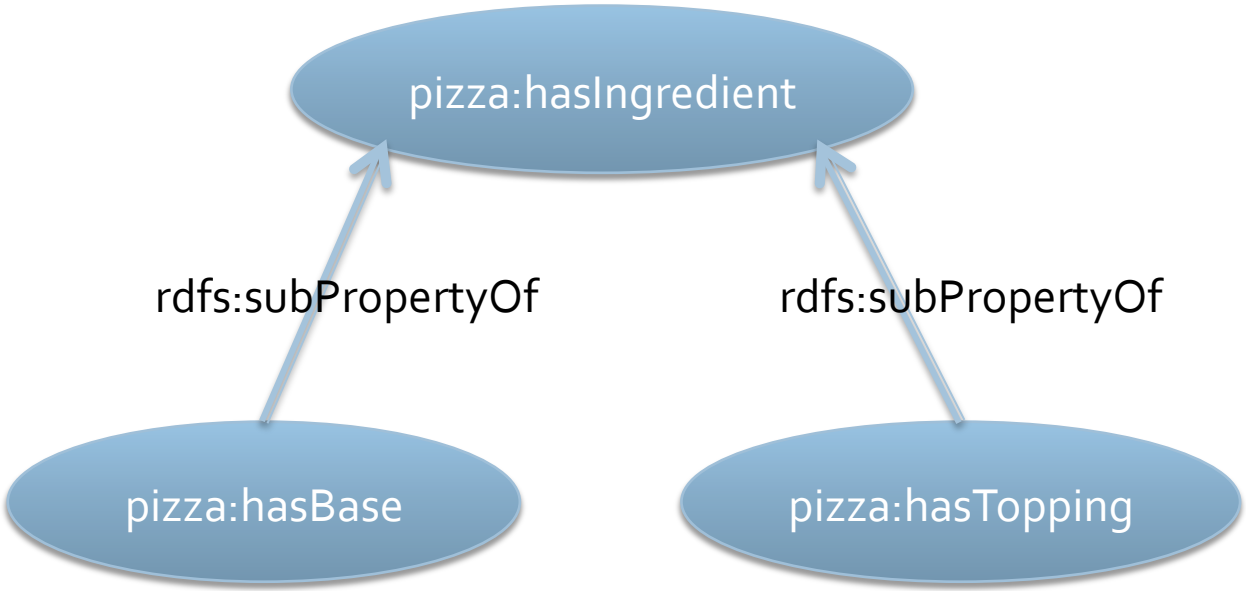
RDF-Schema : hiérarchie de classes



D'après Bernd Amman

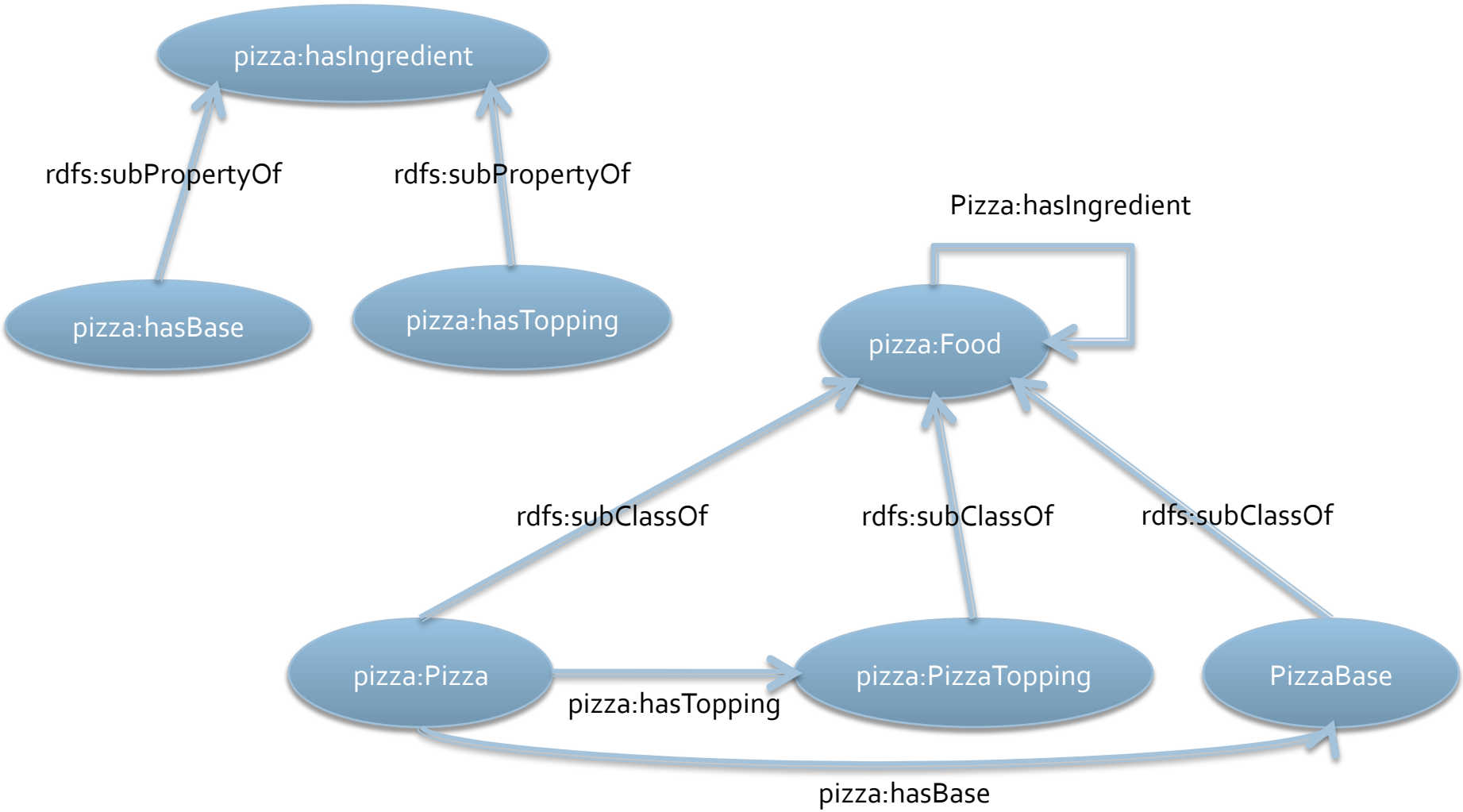
RDF-Schema : hiérarchie de propriétés

http://www.co-ode.org/ontologies/pizza/2007/02/12/pizza.owl

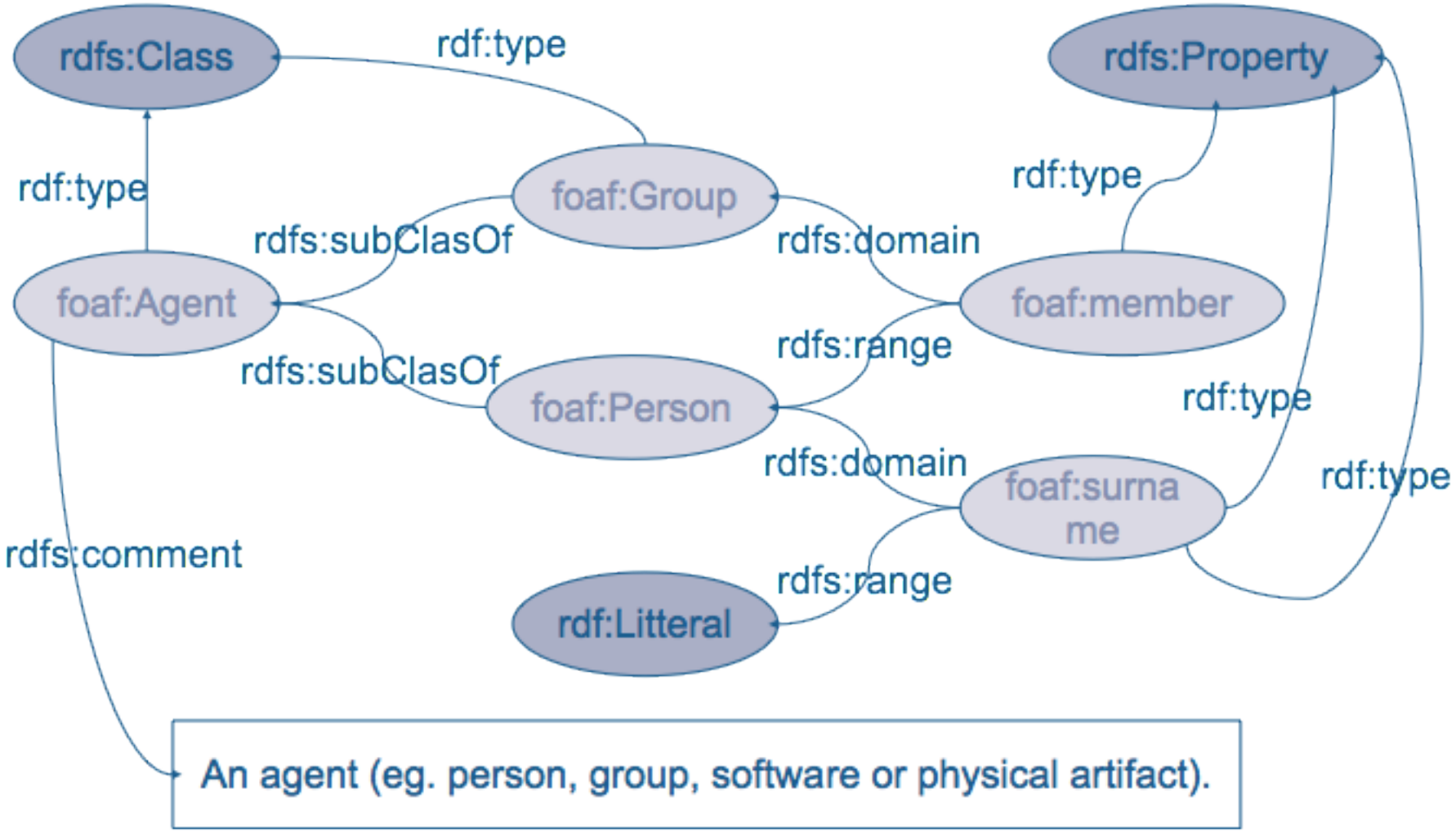


RDF-Schema : hiérarchie de propriétés

http://www.co-ode.org/ontologies/pizza/2007/02/12/pizza.owl



Exemple schema FOAF



RDF Schema : inférences

- Inférence en général
  - déduction automatique à partir d'une structure informatique, effectuée sur la base de la syntaxe du langage utilisé, mais dont les résultats doivent correspondre à une sémantique formelle précise

RDF-Schema : inférence de spécialisation

- Inférence principale liée à une hiérarchie, en suivant la sémantique classique de la spécialisation : toute instance de la sous-classe est instance de la super-classe
- Si je cherche toutes les *Entités de droit*, alors j'ai comme réponse *Yannick*, *Marie*, *Stéphanie* même si je n'ai pas explicitement demandé des *Personne*
- Si je cherche des choses qui ont des ingrédients (*hasIngredient*), je trouverai des *Pizza*

Exemples RDF-schema

- FOAF : Friend of a Friend
- DOAC : Description of a carreer
- DOAP : Description of a project
- Dublin Core
- etc.

Plan

- Web sémantique / web de données
- RDF
- RDF schéma
- **OWL**

OWL

- Basé sur une les logiques de description
  - = classe de langages de représentation de connaissances
- Ontology Web Language

OWL

- Changement de vocabulaire
  - Classes → concepts
  - Instance → individus
- On garde les hiérarchies de concepts et de propriétés
- On garde range et domain pour les propriétés
- On ajoute des possibilités
  - plus grande expressivité du langage
  - inférences plus riches, mais…
  - plus grande complexité pour faire des inférences
    - en terme de calculs informatiques

OWL : caractériser des propriétés

- Déclarer une propriété comme transitive
  - Si A p B et B p C alors A p C
- Déclarer une propriété comme réflexive
  - A p B équivalent à B p A
- Déclarer une propriété comme inverse d'une autre
  - A p B équivalent à B q A si q = inv(p)
- etc.

OWL : caractériser des concepts (1/2)

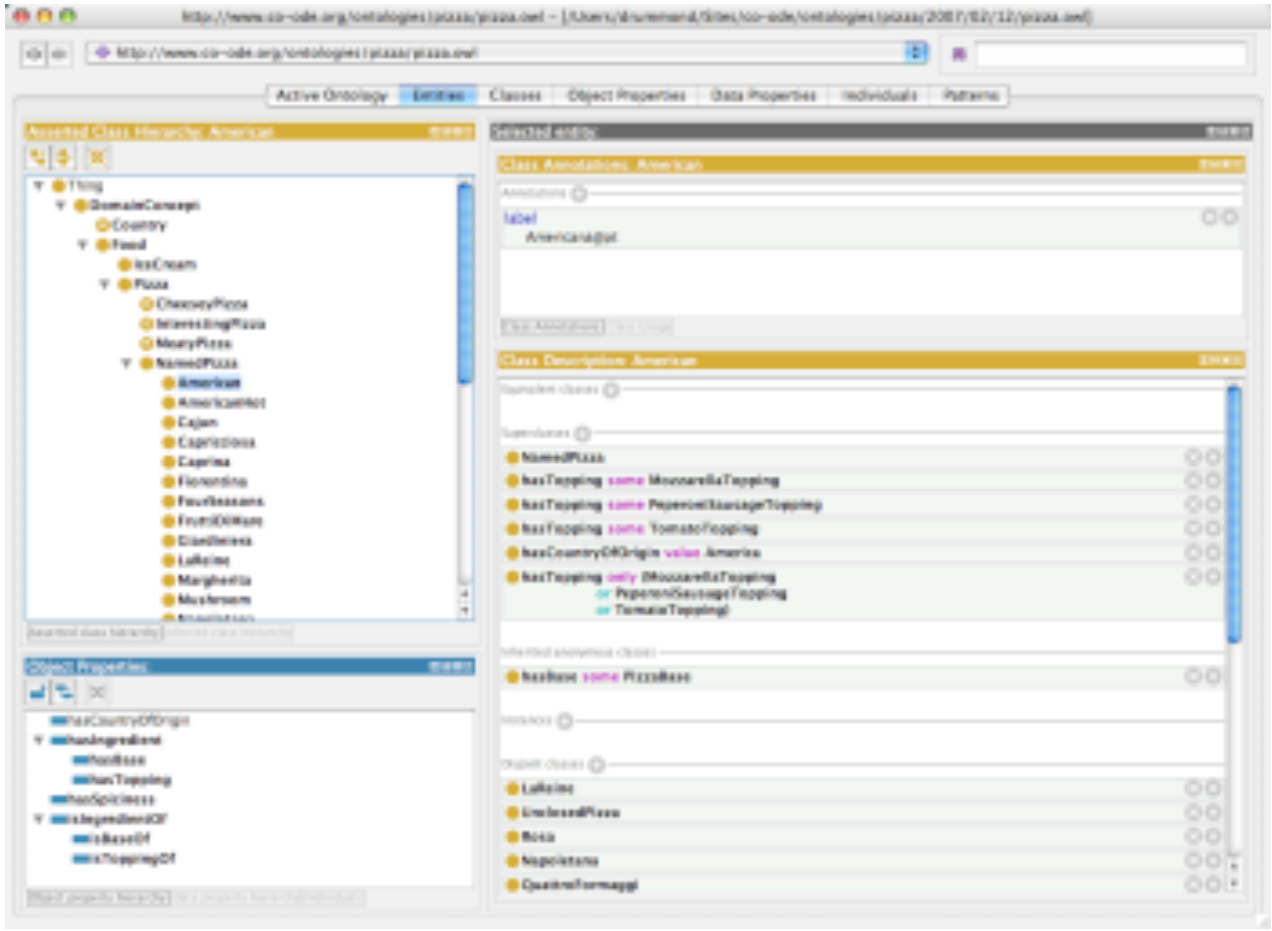
- Ne pas juste définir un concept comme un sous-concept d'un autre (rdfs:subClassOf)
- Mais le caractériser par rapport à d'autres concepts ou à des propriétés
  - Déclarer qu'un concept est disjoint d'un autre
    - A disjoint de B ssi aucun individu ne peut être instance de A et de B à la fois
  - Déclarer qu'un concept est le complémentaire d'un autre
    - Le monde des individus se divise en deux, les instances de A, et les instances de B
  - Déclarer qu'un concept est l'intersection de deux autres
    - Les individus instance de ce concept sont ceux qui sont instance de A et de B

OWL : caractériser des concepts (2/2)

- Déclarer un concept en listant les propriétés qu'il doit avoir
  - Les choses qui ont un époux (une propriété aPour Epoux)
  - Les choses qui sont des Voiture (sous-classe) et qui ont au moins 3 roues
  - etc.

Outil pour gérer des ontologies

- Protege : outil le plus répandu
- Concepts
- Propriétés
- Individus
- Requêtes
- Raisonneur
  - Fact++

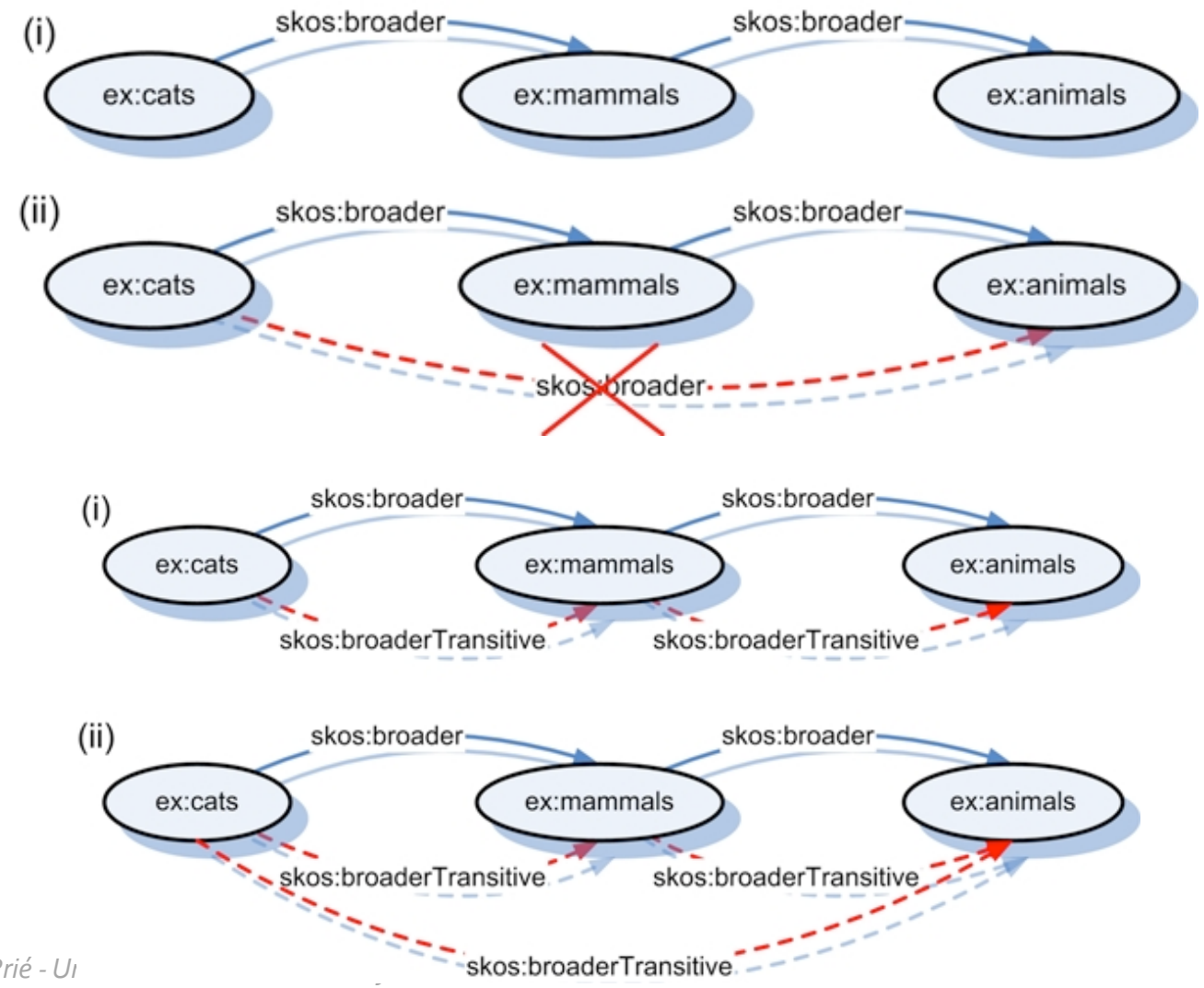


http://protege.stanford.edu

Exemples OWL

- Wine, Pizza, Travel
  - ontologies jouet accessibles dans Protege
- SKOS : Simple Knowledge Organisation System
  - http://www.w3.org/TR/2009/REC-skos-reference-20090818/
  - Pour représenter des thésaurii existants et les porter sur le web sémantique
- Nombreux autres exemples

SKOS : remarque sur la transitivité



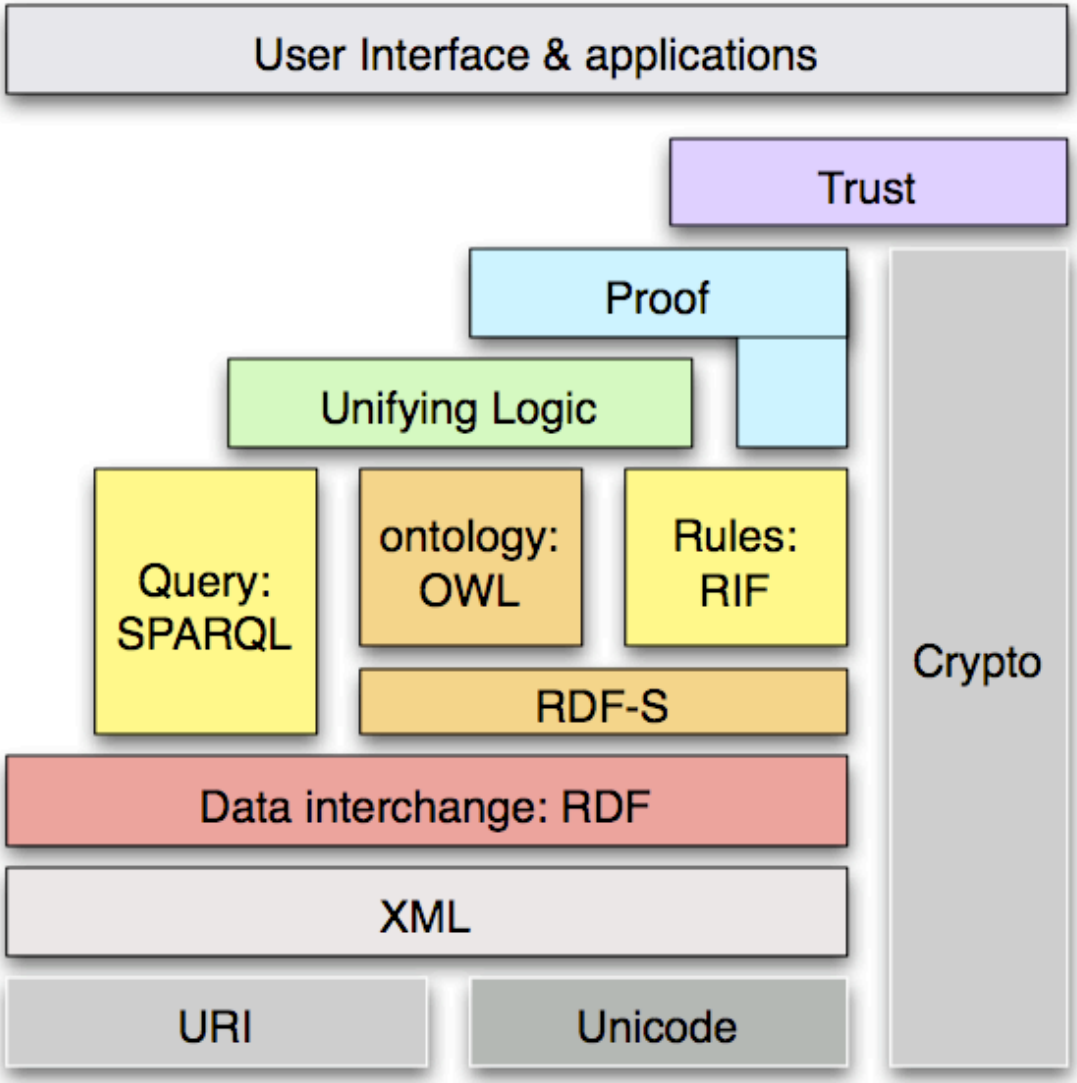
Bonus

- Conception d'ontologies 

Plan

- Web sémantique / web de données
- RDF
- RDF schéma
- OWL
- **Web de données**

La couche des langages

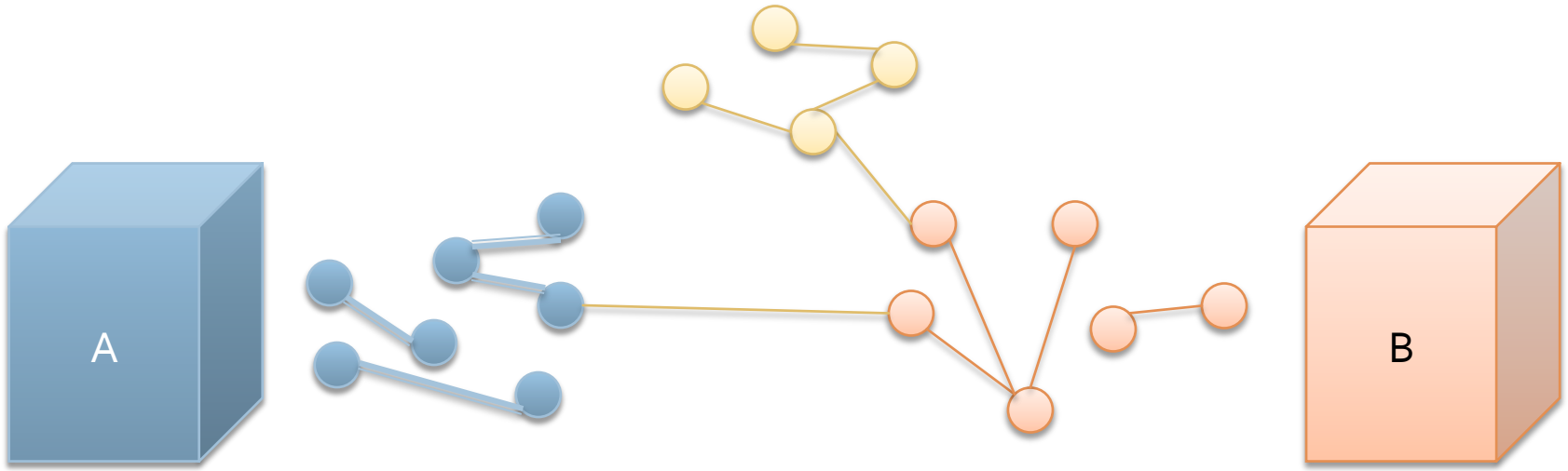


Où on en est

- Plus de 10 ans de travail du W3C
  - technologies qui arrivent à maturité
  - applications industrielles
- Les acteurs basculent petit à petit
  - applications web construites « à base de données sémantiques »
    - bases internes en RDF
    - bases extérieures
  - exposition de données propres sur le web de données
    - Service d'accès à ses données en RDF
  - exposition de données en RDFa
  - etc.

Web of data : principes

- Exposition des données sous forme RDF
- Vision unifiée des données de A, de B, dans le cadre de C
- Construction d'un service par C sur ce graphe unique



Exemple LOD : DBPedia

- Les données *structurées* de wikipedia disponibles en RDF
- Exemple d'utilisation
  - *All soccer players, who played as goalkeeper for a club that has a stadium with more than 40.000 seats and who are born in a country with more than 10 million inhabitants*

SPARQL Explorer for http://dbpedia.org/sparql

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT DISTINCT ?page  {
?s foaf:page ?page.
?s rdf:type <http://dbpedia.org/ontology/SoccerPlayer> .
?s dbpedia2:position <http://dbpedia.org/resource/Goalkeeper> .
?s <http://dbpedia.org/ontology/club> ?club .
?club <http://dbpedia.org/ontology/capacity> ?cap .
?s <http://dbpedia.org/ontology/birthPlace> ?place .
?place ?population ?pop
Filter (xsd:int(?cap) >40000 ) .
Filter (xsd:int(?pop) >10000000 ) .
```

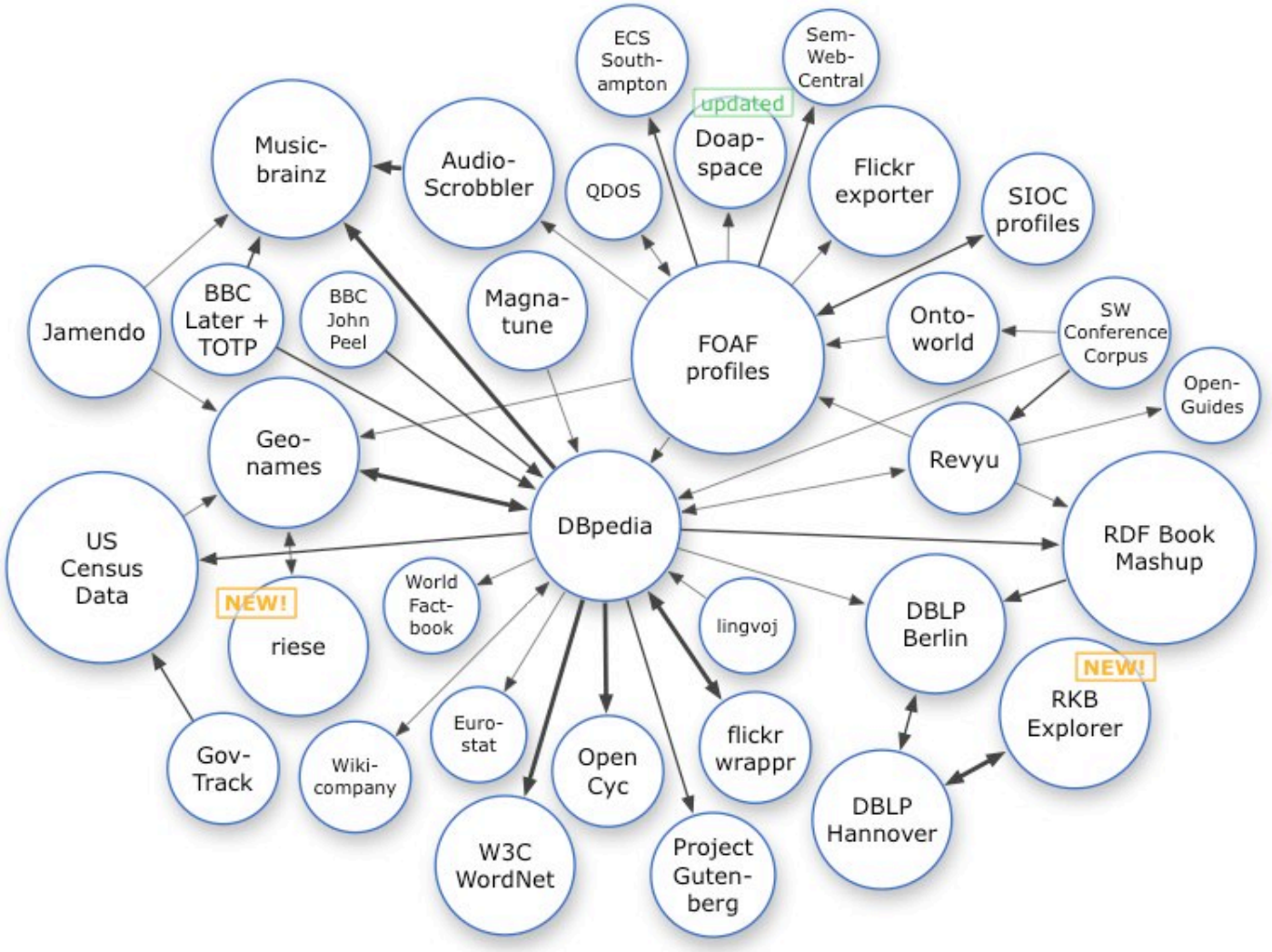
Results: Browse Go! Reset

SPARQL results:

page
<http://en.wikipedia.org/wiki/%C3%89dson_Bastos>
<http://en.wikipedia.org/wiki/Olivier_Renard>
<http://en.wikipedia.org/wiki/Francisco_Ramos>
<http://en.wikipedia.org/wiki/Leonardo_Zamora>
<http://en.wikipedia.org/wiki/Paulo_Garc%C3%A9s>
<http://en.wikipedia.org/wiki/Ra%C3%BAl_Olivares>
<http://en.wikipedia.org/wiki/Richard_Leyton>
<http://en.wikipedia.org/wiki/Joseph-Antoine_Bell>
<http://en.wikipedia.org/wiki/Cantarele>
<http://en.wikipedia.org/wiki/Matthew_Nash>
<http://en.wikipedia.org/wiki/Danny_Milosevic>
<http://en.wikipedia.org/wiki/Ivan_Necevski>

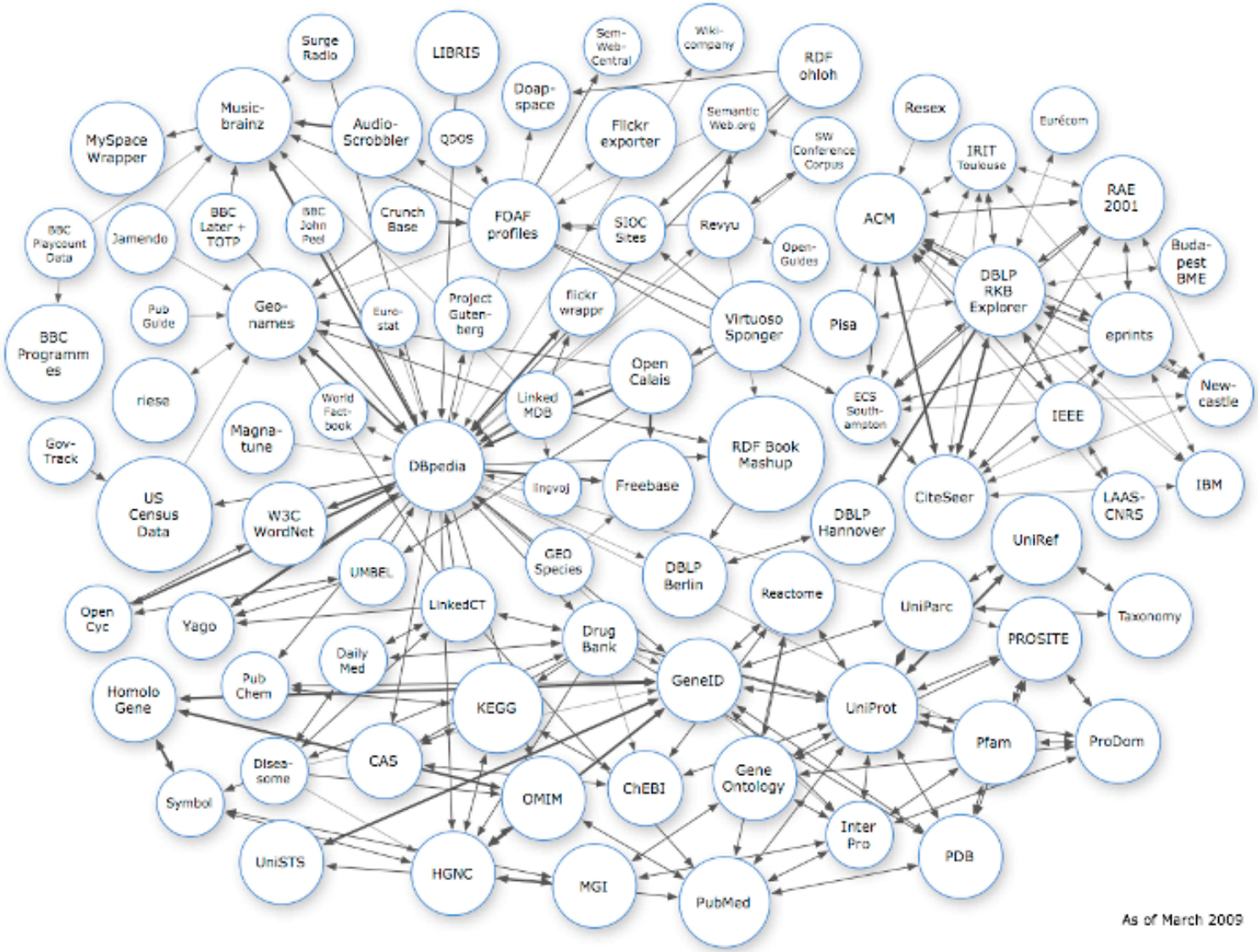
Web of data - Linked Open Data

(2008)



Web of data - Linked Open Data

(mars 2009)



Dans la suite

- TP1 : découverte de RDFa, Protege, OWL
- TP2-3 : conception d'une ontologie pour décrire des documents

Remerciements

- Alexandre Passant
- Eric Miller
- Pierre-Antoine Champin

Annexe : SPARQL

Exemples de requêtes sur BPedia

Adapté d'après une présentation d'Alexandre Passant
http://www.slideshare.net/terraces/the-social-semantic-web-and-linked-data-presentation

DBPedia

- http://dbpedia.org
- Représentation RDF de données extraites de Wikipedia
- Plus de 2 millions de concepts (personnes, lieux, etc.)

Les URI Dbpedia

- URI de resource URI
  - http://dbpedia.org/resource/Semantic_Web
  - Redirige vers sa représentation HTML ou RDF en fonction du client
- Document HTML
  - http://dbpedia.org/page/Semantic_Web
- Données RDF
  - http://dbpedia.org/data/Semantic_Web

DBPedia : Lyon

About: Lyon

An Entity of Type: 0, in Data Space: dbpedia.org

Lyon (prononcé) est une commune française située dans le quart s
Saône. C'est le chef-lieu du département du Rhône et de la région
les Lyonnais. Lyon est en situation de carrefour géographique, au r
Rhône (qui s'étend de Lyon à Marseille).

Property	Value
dbpedia-owl:abstract	▪ Lyon er den tredjestørste by i 4.415.000 (2007) indbyggere, floderne Saône og Rhône, og gallerne som en tilflugtsborg (efter Julius Cæsars erobring a var fordelagtig ved en korsvej Da burgunderne havde lidt ne De gjorde byen til hovedstad i Frankerriget. Efter Karl den St Lyon. Dette område blev sene højmiddelalderen. Efter 100-å franske konges herredømme, vinregioner, Beaujolais og Côt ligger i Lyon. Klubben har vun ▪ Datei:Loudspeaker. svg Lyon [Rhône im Osten Frankreichs. ▪ Lyon on kaupunki itäisessä Ra mukaanluettuina yli 1,7 miljoor Rhônen departementin, Rhône pääkaupunki. Kaupungin kesk Lyonissa on kolme yliopistoa.

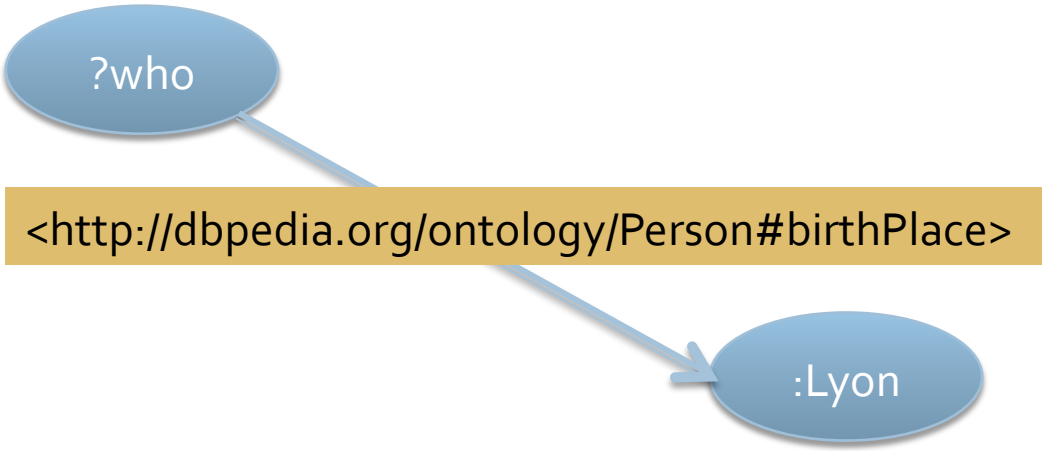
dbpedia-owl:mayor	▪ dbpedia:Gérard_Collomb
dbpedia-owl:populationTotal	▪ 472305 (xsd:integer)
dbpedia-owl:region	▪ dbpedia:Rhône-Alpes
dbpedia-owl:subdivisions	▪ 9 (xsd:integer)
dbpedia-owl:thumbnail	▪ http://upload.wikimedia.org/wikipedia/commons/thumb/f/f6/Flag_of_Lyon.png/
dbpprop:areaKm	▪ 47.950000 (xsd:double)
dbpprop:arrondissement	▪ Lyon
dbpprop:canton	▪ chief town of 14 cantons
dbpprop:caption	▪ Basilique de Fourvière and the Hôtel de Ville de Lyon
dbpprop:cityMotto	▪ Avant, avant, Lion le melhor.
dbpprop:coatOfArmsLegend	▪ City coat of arms
dbpprop:commonName	▪ Lyon
dbpprop:criteria	▪ ii, iv
dbpprop:department	▪ Rhône
dbpprop:elevationMaxM	▪ 305 (xsd:integer)
dbpprop:elevationMinM	▪ 162 (xsd:integer)
dbpprop:flagLegend	▪ City flag
dbpprop:hasPhotoCollection	▪ http://www4.wiwiss.fu-berlin.de/flickrwrappr/photos/Lyon
dbpprop:id	▪ 872 (xsd:integer)
dbpprop:imageCoatOfArms	▪ Blason_Lyon.png
dbpprop:imageCoatOfArmsSize	▪ 115px
dbpprop:imageFlag	▪ Flag of Lyon.png
dbpprop:imageFlagSize	▪ 115px
dbpprop:intercomDetails	▪ dbpedia:Urban_Community_of_Lyon
dbpprop:latitude	▪ 45.759723 (xsd:double)
dbpprop:link	▪ http://whc.unesco.org/en/list/872
dbpprop:longitude	▪ 4.842223 (xsd:double)
dbpprop:mayor	▪ dbpedia:Gérard_Collomb

Requêtes SPARQL sur DBPEdia

- http://dbpedia.org/sparql
- http://dbpedia.org/snorql/

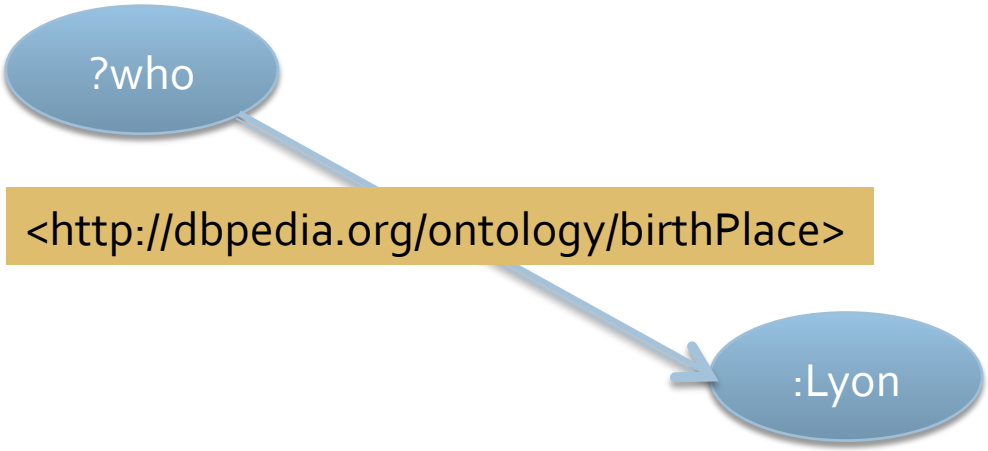
Les gens nés à Lyon

- Pattern de triplet simple
- Utilisation de la propriété
  - <http://dbpedia.org/ontology/Person#birthPlace>



Les gens nés à Lyon

- Pattern de triplet simple
- Utilisation de la propriété
  - <http://dbpedia.org/ontology/birthPlace>



Requête

```
SELECT ?who
WHERE {
    ?who <http://dbpedia.org/ontology/birthPlace> :Lyon.
}
```

Réponse

SPARQL Explorer for http://dbpedia.org/sparql

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT ?who
WHERE {
        ?who <http://dbpedia.org/ontology/birthPlace> :Lyon.
}
```

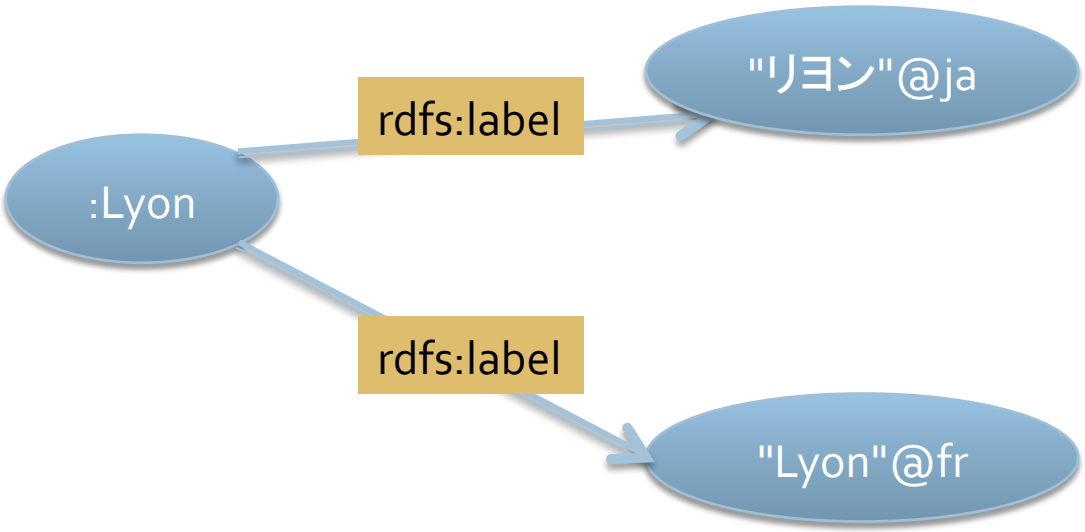
Results: Browse Go! Reset

SPARQL results:

who
:Albert_Jacquard
:Allan_Kardec
:Antoine_Laurent_de_Jussieu
:Bruno_N%27Gotty
:Christine_Pascal
:Claude_Martin
:Fleury_Di_Nallo
:Louis_Sclavis
:Lucille_Young
:Philip_V_of_France
:Jean-Baptiste_Arban

Nom japonais de Lyon

- Utilisation de la clause FILTER by LANG
  - FILTER (lang(?x) = "ja")

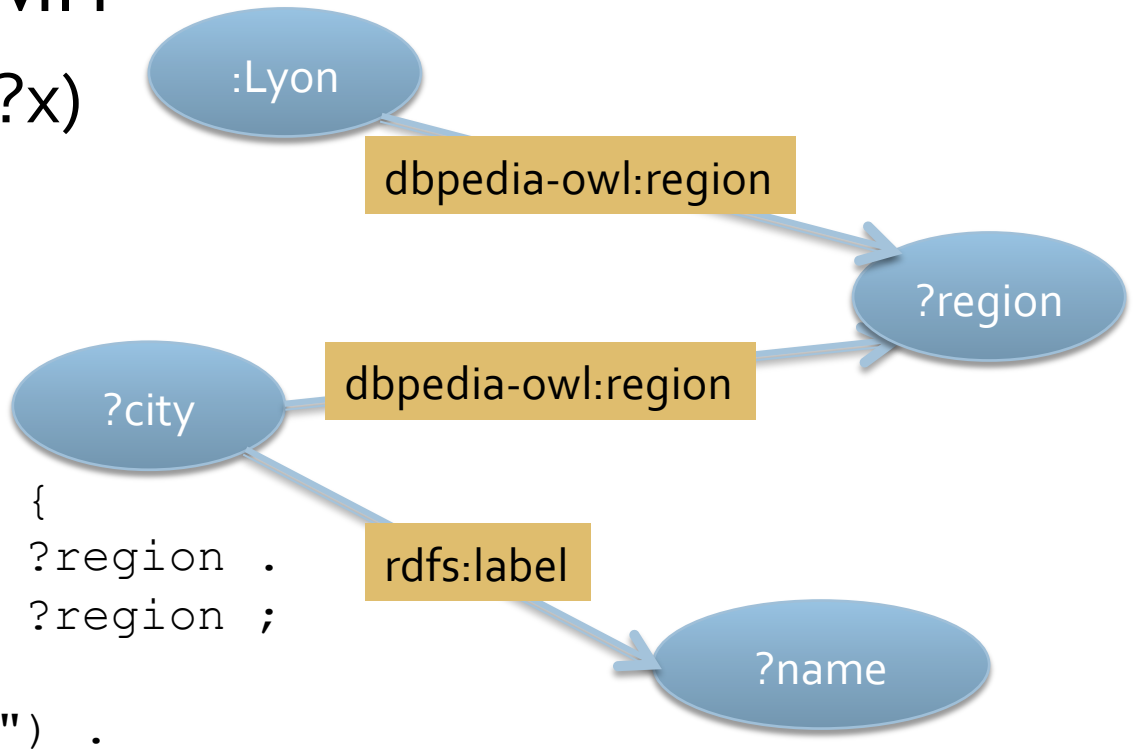


Requête

```
SELECT ?name
WHERE {
    :Lyon rdfs:label ?name .
    FILTER (lang(?name) = "ja")
}
```

Les 10 premières villes de la même région

- Combinaison de patterns de triplets
- ORDER BY et LIMIT
  - ORDER BY ASC(?x)
  - LIMIT X



Requête

```
SELECT ?city ?name WHERE {
:Lyon dbpedia-owl:region ?region .
?city dbpedia-owl:region ?region ;
      rdfs:label ?name .
FILTER(lang(?name) = "fr") .
} ORDER BY ASC (?name) LIMIT 10
```

Réponse

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT ?city ?name WHERE {
:Lyon dbpedia-owl:region ?region .
?city dbpedia-owl:region ?region ;
      rdfs:label ?name .
FILTER(lang(?name) = "fr") .
} ORDER BY ASC (?name) LIMIT 10
```

Results: Browse Go! Reset

SPARQL results:

city	name
:Abondance%2C_Haute-Savoie	"Abondance (Haute-Savoie)"@fr
:Abo%C3%ABn	"Aboën"@fr
:Accons	"Accons"@fr
:Affoux	"Affoux"@fr
:Agnin	"Agnin"@fr
:Aiguebelette-le-Lac	"Aiguebelette-le-Lac"@fr
:Aigueblanche	"Aigueblanche"@fr
:Aigueperse%2C_Rh%C3%B4ne	"Aigueperse (Rhône)"@fr
:Ailhon	"Ailhon"@fr
:Ailleux	"Ailleux"@fr

Les villes de la même région situées à l'Est

- FILTER par type et comparaison de coordonnées
- PREFIX geo: http://www.w3.org/2003/01/geo/wgs84_pos#

```
SELECT DISTINCT ?place ?long ?llong
WHERE {
:Lyon geo:long ?llong ;
      dbpedia-owl:region ?region .
?place dbpedia-owl:region ?region ;
      geo:long ?long .
FILTER (?long > ?llong) .
}
```

Réponse

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
```

```
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT DISTINCT ?place ?long ?llong WHERE {
:Lyon geo:long ?llong ;
      dbpedia-owl:region ?region .
?place dbpedia-owl:region ?region ;
      geo:long ?long .
FILTER (?long > ?llong) .
}
```

Results: Browse Go! Reset

SPARQL results:

place	long	llong
:Albens	5.94583333333	4.842223
:Alixan	5.02833333333	4.842223
:All%C3%A8ves	6.08111111111	4.842223
:Amancy	6.33	4.842223
:Ambel%2C_Is%C3%A8re	5.93111111111	4.842223
:Anjou%2C_Is%C3%A8re	4.88222222222	4.842223
:Annecy	6.133	4.842223
:Annoisin-Chatelans	5.29416666667	4.842223
:Anthon%2C_Is%C3%A8re	5.17055555556	4.842223
:Assieu	4.86861111111	4.842223
:Auberives-en-Royans	5.30194444444	4.842223

Les personnes nées à Lyon et le cas échéant leur lieu de décès

- Utilisation de OPTIONAL

- ```
  SELECT ?who, ?dplace
  WHERE {
     ?who dbpedia-owl:birthPlace :Lyon.
          OPTIONAL { ?who dbpedia-
  owl:deathPlace ?dplace .
        }
  }
  ```
  ```

Réponse

SPARQL:

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
PREFIX : <http://dbpedia.org/resource/>
PREFIX dbpedia2: <http://dbpedia.org/property/>
PREFIX dbpedia: <http://dbpedia.org/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT ?who, ?dplace
WHERE {
        ?who dbpedia-owl:birthPlace :Lyon.
        OPTIONAL { ?who dbpedia-owl:deathPlace ?dplace .
      }
}
```

Results: Browse Go! Reset

SPARQL results:

who	dplace
:Albert_Jacquard	-
:Allan_Kardec	:Paris
:Antoine_Laurent_de_Jussieu	-
:Bruno_N%27Gotty	-
:Christine_Pascal	:Garches
:Claude_Martin	:Lucknow
:Fleury_Di_Nallo	-
:Louis_Sclavis	-
:Lucille_Young	:Los_Angeles%2C_California
:Philip_V_of_France	:Longchamp_%28abbey%29
:Jean-Baptiste_Arban	:Paris

Annexe : complément ontologies

Composants d'une ontologie

2010-2011 - Yannick Prié - Université Claude Bernard Lyon 1

Composants d'une ontologie : concepts / classes

- Un concept représente un ensemble d'objets et leurs propriétés communes.
- Décrit par un terme
  - eg. Voiture, Vache, Violon
- Un concept peut avoir
  - une définition intensionnelle : CNS pour appartenir au concept
    - eg. Véhicule de transport automobile conçu et aménagé pour le transport d'un petit nombre de personnes
  - une définition extensionnelle : description exhaustive de tout ce qui obéit à la définition
    - eg. la liste de toutes les voitures du monde
- A lier à une approche triadique du concept (Aristote, Frege, etc.)
  - terme, intension, extension : mot, sens, objets

Composants d'une ontologie : individus

- Instances de concepts
- Les éléments décrits par les classes
- Exemples :
  - la voiture de Jean
  - Blanchette, Marguerite, Roussette

Composants d'une ontologie : propriétés

- On différencie en général
  - attributs : propriétés simples
    - eg. age, nom, nombre-de-roues, etc.
  - rôles, relations sémantiques : association entre concepts
    - eg. parent-de, partie-de, proche-de, sous, contient, connecté à, etc.
- On peut y associer des facettes
  - valeurs possibles d'attributs
    - eg. 0 < age < 150
  - restrictions de rôles
    - eg. pas plus de 2 parents

Composants d'une ontologie relations de spécialisation

- Relation binaire entre un concept général et un concept plus spécifique.
- Relation inverse = généralisation
- Expression d'une inclusion ensembliste en terme sémantique
- Noms variés
  - is-a, est un, est une sorte de
- Induit une hiérarchie de spécialisation, ou une taxinomie
- Remarque
  - les propriétés peuvent aussi être organisées en hiérarchie de spécialisation

Taxinomie des pains

Pain

Pain spécial

Pain complet

Pain aux lardons

Pain blanc

Pain de seigle

Pain au son

Pain aux noix et aux raisins

Pain régional

Pain auvergnat

Couronne

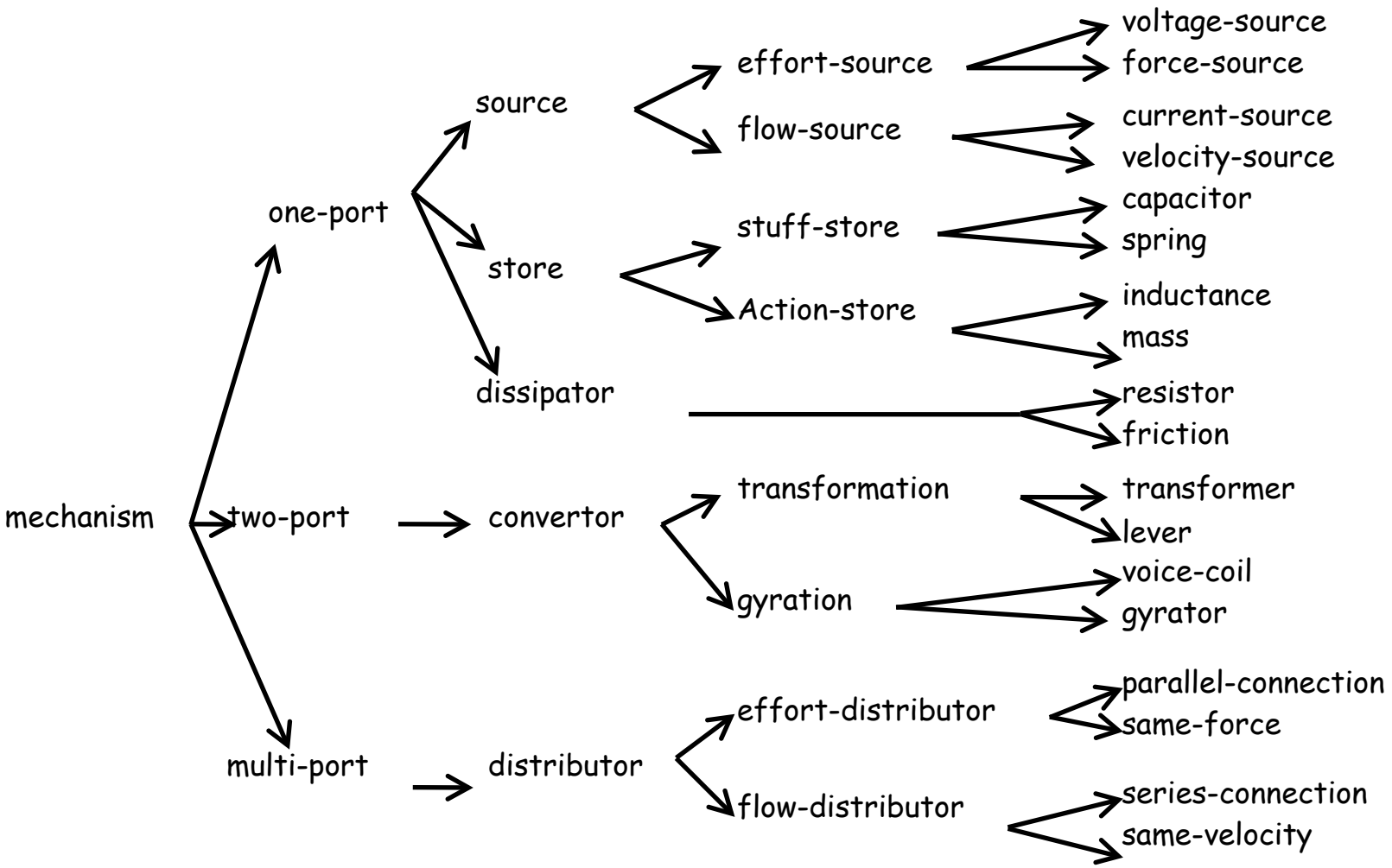
Pain marguerite

Lyonnaise

Pain vaudois

Taxinomie de mécanismes physiques

(Borst)



Connectivity Type Effort/flow Domain (e.g. el, me)

Taxinomies et ontologies

- Taxinomie/Taxonomie : classification d'éléments (Petit Robert)
- Origines
  - arbre de Porphyre, classification des êtres vivants et des fossiles, etc.
- Naturelle pour l'homme qui fonctionne souvent par abstraction et association
- Structure à la base de deux inférences élémentaires que nous faisons tous les jours
  - l'identification : capacité à reconnaître la classe d'un objet à partir de ses caractéristiques
  - la spécialisation : capacité à prendre en compte des catégories de niveaux de précision variables

Composants d'une ontologie : restrictions, règles, axiome

- Pour exprimer ce qui ne peut l'être comme concept ou propriété
- Restriction : chose qui doit être vraie pour que ce qui est exprimé soit valable
  - cf. restrictions de rôles
- Règle : affirmation sous la forme antécédent > conséquent décrivant des inférences possibles
  - "Louis XIV" est le même individu que "Roi soleil"
  - une voiture rare est chère
  - uncle(x, y) ← brother(x, z) ∧ father(z, y)
- Axiome : assertion générale sur les fondements de l'ontologie
  - partOf est transitive
  - parent-de est l'inverse de enfant-de

Annexe : logiques de description

Rappel : Interprétation et modèle

- Etant donné un langage formel
- Une interprétation $\mathcal{I}$ est définie par
  - un domaine d'interprétation $\Delta^{\mathcal{I}}$ (ensemble non vide)
  - une fonction d'interprétation $\cdot^{\mathcal{I}}$ qui associe les structures syntaxiques du langage au domaine
- Un énoncé $\mathcal{E}$ du langage est vérifié par $\mathcal{I}$ si ses structures vérifient des propriétés définies par la sémantique du langage
  - on note : $\models_{\mathcal{I}} \mathcal{E}$
  - $\mathcal{I}$ est appelée un modèle de $\mathcal{E}$

Logiques de description

- Famille de langages de représentation des connaissances, offrant de bons compromis entre
  - lisibilité et formalisation
    - réseaux sémantiques, frames
    - communication homme – machine
  - expressivité et décidabilité/complexité
    - $\subsetneq$ logique du premier ordre
    - communication machine – machine
  - Bons candidats pour représenter des ontologies

Principes de base (1)

- Les termes du langage sont séparés en familles disjointes
  - Les concepts (ou classes)interprétés comme des sous-ensembles de $\Delta^{\mathfrak{I}}$
  - exemples : *Personne*, *Étudiant*, *Document*
- Les rôles (ou propriétés)
  - interprétés comme des relations binaires sur $\Delta^{\mathfrak{I}}$
  - exemples : *père*, *auteur*

Principes de base (2)

- Dans certaines LD, d'autres familles de termes sont utilisées :
  - Les individus
    - interprétés comme des éléments de Δ
    - exemples : pa_champin, cette_présentation
  - Les valeurs concrètes (rare)
    - interprétés comme des éléments de ΔD
      - ΔD Domaine concret
    - Exemple : 42, 3.1415, "hello world"

Différentes LD

- Les diverses LD se différencient par :
  - Les termes complexes que l'on peut construire à partir de termes de bases
    - aussi appelés concepts définis vs concepts primitifs
    - exemple : « toute personne qui est l'auteur d'un document »
  - Les axiomes que l'on peut exprimer (« phrases »)
    - exemple : « un étudiant est toute personne qui suit un cours »
  - Les mécanismes d'inférence qu'elles offrent

Concepts complexes (1)

- constructeurs ensemblistes
  - $C \sqcap D \quad \underrightarrow{\mathcal{I}} \quad C^{\mathcal{I}} \cap D^{\mathcal{I}}$
  - $C \sqcup D \quad \underrightarrow{\mathcal{I}} \quad C^{\mathcal{I}} \cup D^{\mathcal{I}}$
  - $\neg C \quad \underrightarrow{\mathcal{I}} \quad \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$
- quantificateurs
  - $\exists\ R.C \quad \underrightarrow{\mathcal{I}} \quad \{ x \mid \exists\ (x,y) \in R^{\mathcal{I}}, y \in C^{\mathcal{I}} \}$
  - $\forall\ R.C \quad \underrightarrow{\mathcal{I}} \quad \{ x \mid \forall\ (x,y) \in R^{\mathcal{I}}, y \in C^{\mathcal{I}} \}$
- restrictions de cardinalité
  - $\leq n\ R \quad \underrightarrow{\mathcal{I}} \quad \{ x \mid \text{card}\{(x,y) \in R^{\mathcal{I}}\} \leq n \}$
  - $\geq n\ R \quad \underrightarrow{\mathcal{I}} \quad \{ x \mid \text{card}\{(x,y) \in R^{\mathcal{I}}\} \geq n \}$
  - $= n\ R \quad \underrightarrow{\mathcal{I}} \quad \{ x \mid \text{card}\{(x,y) \in R^{\mathcal{I}}\} = n \}$

Concepts complexes (2)

- restrictions de cardinalité qualifiées
  - $\leq n\ R.C \underrightarrow{\mathcal{I}} \{ x \mid card\{(x,y) \in R^{\mathcal{I}} \text{ et } y \in C^{\mathcal{I}}\} \leq n \}$
  - $\geq n\ R.C \underrightarrow{\mathcal{I}} \{ x \mid card\{(x,y) \in R^{\mathcal{I}} \text{ et } y \in C^{\mathcal{I}} \} \geq n \}$
  - $= n\ R.C \underrightarrow{\mathcal{I}} \{ x \mid card\{(x,y) \in R^{\mathcal{I}} \text{ et } y \in C^{\mathcal{I}} \} = n \}$
- extension
  - $\{ i, j, k... \} \underrightarrow{\mathcal{I}} \{ i^{\mathcal{I}}, j^{\mathcal{I}}, k^{\mathcal{I}}, ... \}$

Exemples de concepts complexes

- ¬ (Étudiant ⊔ Salarié)
- Groupe ⊓ ∃ membre . Étudiant
  - Peut contenir un groupe sans aucun membre ?
- Groupe ⊓ ∀ membre . Étudiant
  - Peut contenir un groupe sans aucun membre ?
- Groupe ⊓ (≥ 10 membre)
- (= 1 auteur)
- (= 1 auteur.Étudiant)
- { john, paul, george, ringo }

Rôles complexes (certaines LD)

- constructeurs ensemblistes
  - $R \sqcap S \quad \underrightarrow{\mathcal{I}} \quad R^{\mathcal{I}} \cap S^{\mathcal{I}}$
  - $R \sqcup S \quad \underrightarrow{\mathcal{I}} \quad R^{\mathcal{I}} \cup S^{\mathcal{I}}$
  - $\neg R \quad \underrightarrow{\mathcal{I}} \quad \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \setminus R^{\mathcal{I}}$
- composition
  - $R \circ S \quad \underrightarrow{\mathcal{I}} \quad \{ (x,y) \mid \exists z, (x,z) \in R^{\mathcal{I}} \text{ et } (z,y) \in S^{\mathcal{I}} \}$
- inverse
  - $R^- \quad \underrightarrow{\mathcal{I}} \quad \{ (x,y) \mid (y,z) \in R^{\mathcal{I}} \}$
- fermeture transitive
  - $R^* \quad \underrightarrow{\mathcal{I}} \quad \{ (x,y) \mid (x,y) \in (R^{\mathcal{I}})^* \}$

Exemples de rôles complexes

- père ⊔ *mère*
- *collègue* ⊓ *ami*
- *père*°*mère*
- *parent* ⁻
- *parent* *

Axiomes

- Définition de concept atomique
  - $A \sqsubseteq C \quad \underrightarrow{ssi} \quad A^{\mathcal{I}} \subseteq C^{\mathcal{I}}$ (subsomption, CN)
    - Etudiant $\sqsubseteq$ Personne
  - $A \equiv C \quad \underrightarrow{ssi} \quad A^{\mathcal{I}} = C^{\mathcal{I}}$ (équivalence, CNS)
    - Etudiant $\equiv$ Student
- General Inclusion Axiom ($C \sqsubseteq D$, $C \equiv D$)
  - entre concept complexes, limitations
    - Etudiant $\sqsubseteq$ Personne $\sqcap \forall$ statut . Majeur

Autres axiomes

- Hiérarchie de rôles ($R \sqsubseteq S$, $R \equiv S$)
  - auteur $\sqsubseteq$ contributeur
- Rôle transitif, rôle réflexif
  - ancêtre : transitif
- ...

A-Box

- Dans les LD qui supportent les individus :
- T-Box (axiomes sur les classes et les rôles)
  - niveau terminologique
- A-Box (axiomes sur les individus)
  - niveau assertion
  - $i:C \xrightarrow{\mathcal{I}} i^{\mathcal{I}} \in C^{\mathcal{I}}$
  - $i,j:R \xrightarrow{\mathcal{I}} (i^{\mathcal{I}}, j^{\mathcal{I}}) \subseteq R^{\mathcal{I}}$
  - $i = j \xrightarrow{\mathcal{I}} i^{\mathcal{I}} = j^{\mathcal{I}}$
  - $i \neq j \xrightarrow{\mathcal{I}} i^{\mathcal{I}} \neq j$
- NB : pas de distinction dans les LD qui supportent l'extension
  - $i:C \Leftrightarrow \{ i \} \sqsubseteq C$, $i,j:R \Leftrightarrow \{ i \} \sqsubseteq \exists R.\{ j \}$, etc.

Inférences dans les LD

- Inférences de base
  - satisfiabilité : $\exists \mathcal{I}, C^{\mathcal{I}} \neq \emptyset$
    - un concept peut-il avoir des instances ?
  - subsomption : $\forall \mathcal{I}, C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$
    - un concept en subsume-t'il un autre ?
  - classification
    - où placer un concept dans la hiérarchie ?
- Inférence sur les individus
  - identification ou test à l'instanciation
    - quel est le concept le plus spécifique décrivant un individu ?
- Raisonnement sur une base de connaissance
  - T-Box seule ou T-Box + A-Box

Inférence dans les LD

- On peut tout ramener à un problème de satisfiabilité, donc au test d'existence d'un modèle pour une expression
  - C est subsumé par D ⇔ C ⊓ ¬D est insatisfiable
  - C et D sont équivalents ⇔ C ⊓ ¬D et D ⊓ ¬C sont insatisfiables
  - C et D sont disjoints ⇔ C ⊓ D est insatisfiable
  - a est une instance de C ⇔ ABox ∪ {a : ¬C} est insatisfiable
- La complexité dépend de la logique de description choisie, qui dépend des constructeurs utilisés
  - http://www.cs.man.ac.uk/~ezolin/dl/

LD – Méthode des tableaux

- Un tableau est une représentation d'un ensemble de modèles
  - arbre fini étiqueté
  - chaque branche mémorise une série d'évaluations possibles pour les énoncés testés
- L'application de règles de transformation (adaptation de la méthode des tableaux en logique des propositions) garantit qu'on explorera tous les modèles possibles
- Principe : raisonnement par réfutation
  - on suppose qu'une instance x existe
  - on déduit tout ce qu'on peut sur cette instance
    - si on arrive à une contradiction ou clash : $x_i \in C_i \cap \neg C_i$, il n'y a pas de modèle
    - si on arrive à une branche complète : on a un modèle

LD – Méthode des tableaux (1)

- Soit une T-box
  - Homme $\sqsubseteq \neg$ Femme
  - Personne $\sqsubseteq \exists$ père Homme $\sqcap$
    $\exists$ mère Femme $\sqcap$
    $\forall$ père Homme $\sqcap$
    parent ≤ 2
  - père $\sqsubseteq$ parent
  - mère $\sqsubseteq$ parent
- Question
- Personne $\sqcap$ père ≥ 2 est il satisfiable ?

LD – Méthode des tableaux (2)

- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2 est il satisfiable ?

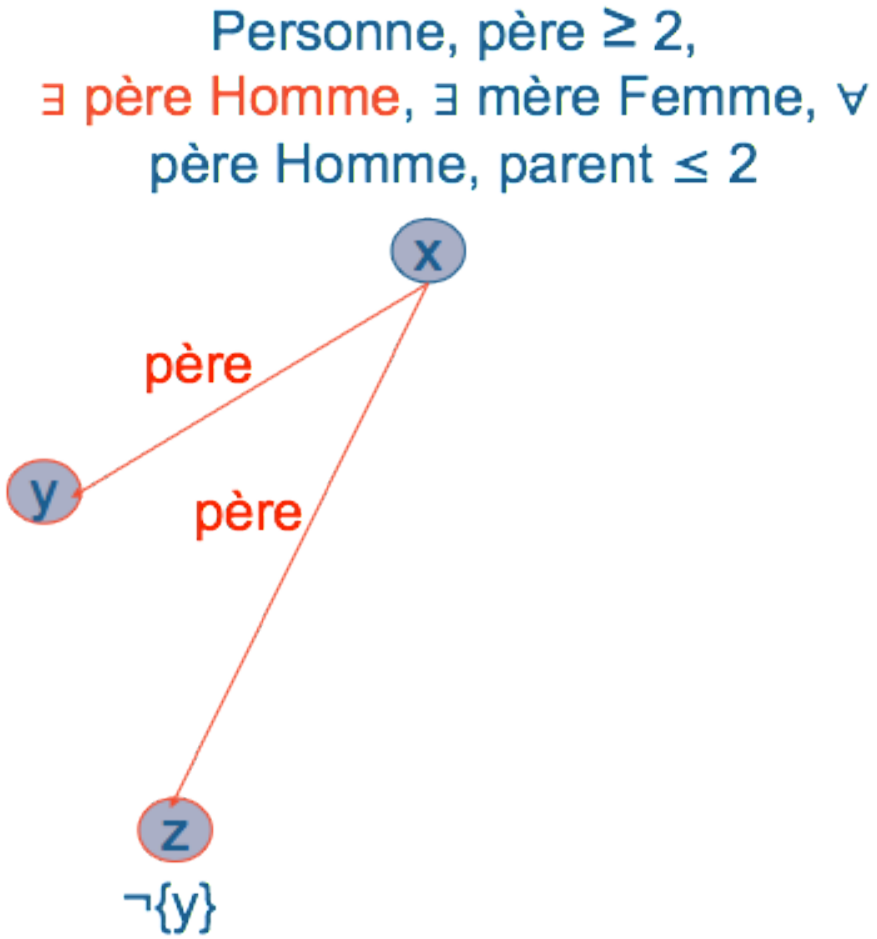
LD – Méthode des tableaux (3)

Personne, père ≥ 2,
∃ père Homme, ∃ mère Femme, ∀
père Homme, parent ≤ 2

- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

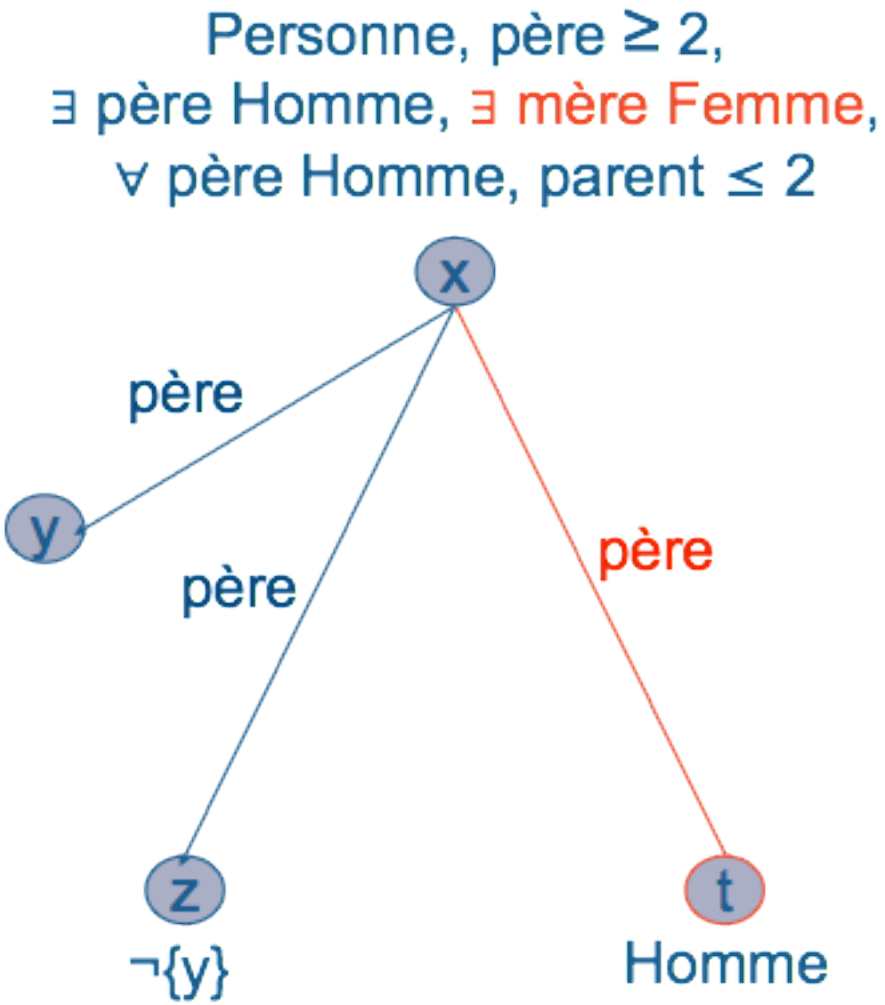
LD – Méthode des tableaux (4)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

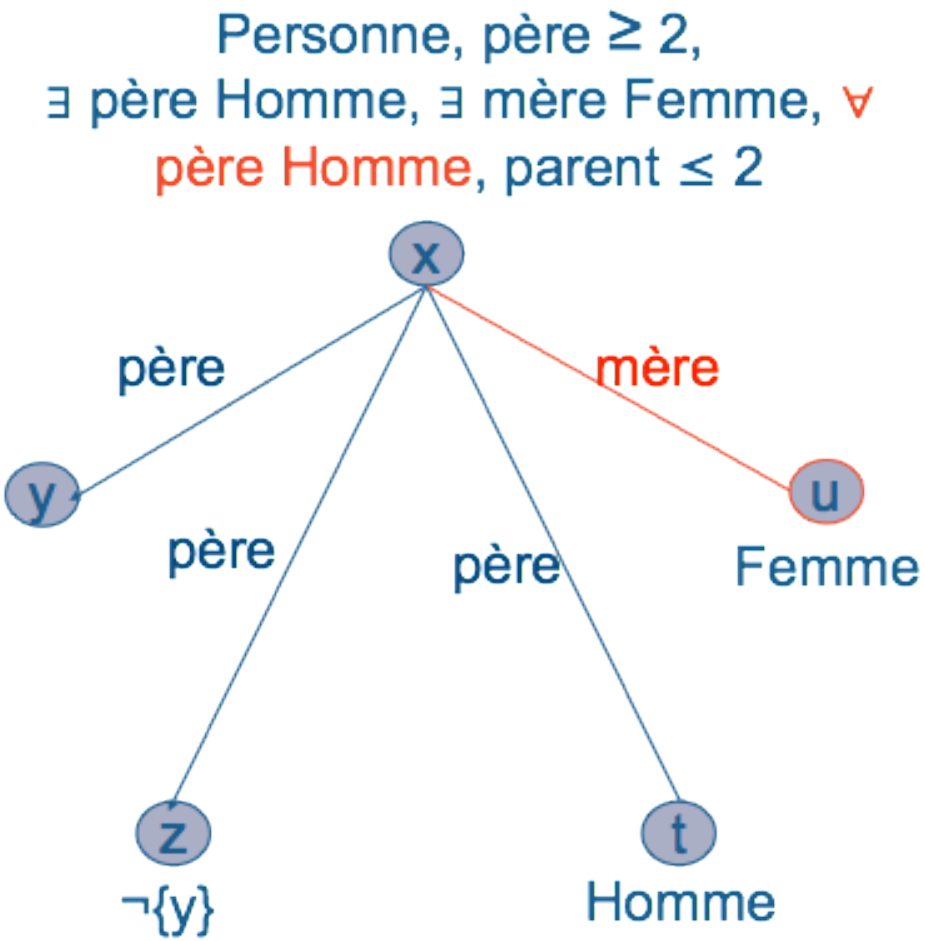
LD – Méthode des tableaux (5)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

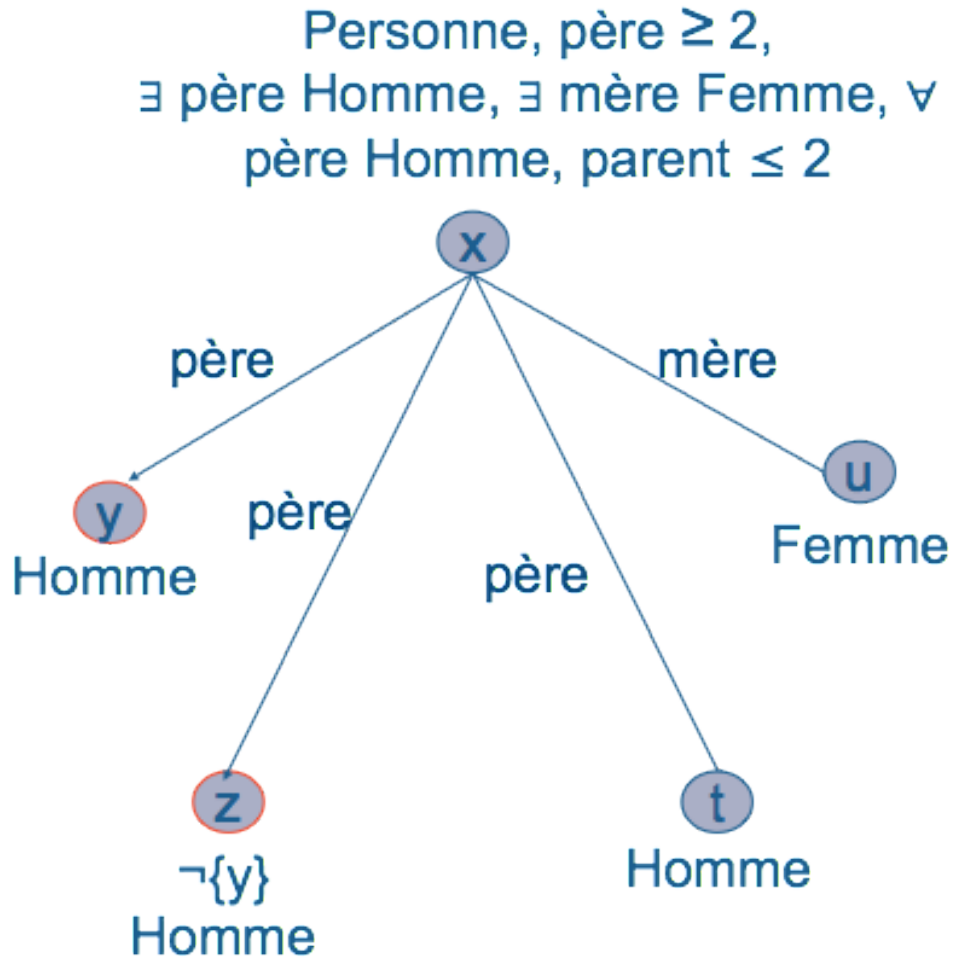
LD – Méthode des tableaux (6)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2 est il satisfiable ?

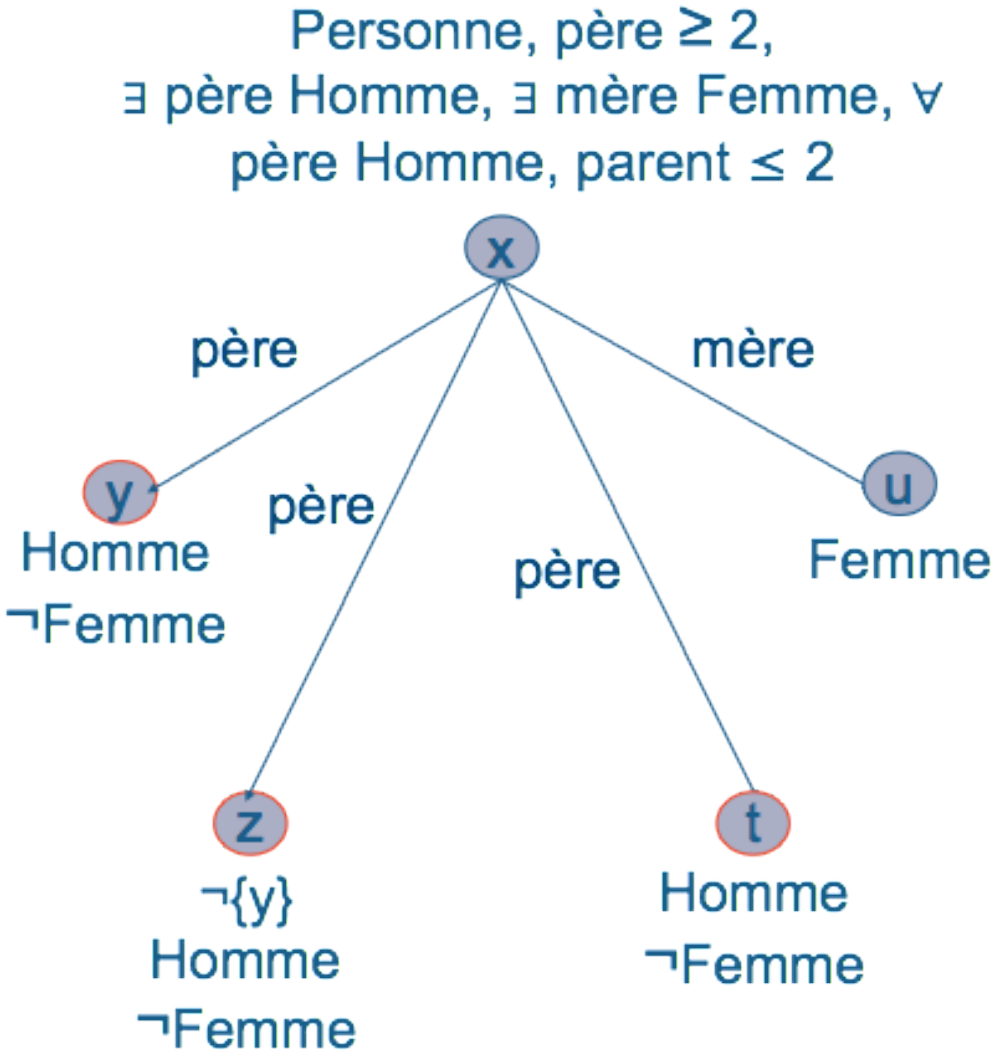
LD – Méthode des tableaux (7)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

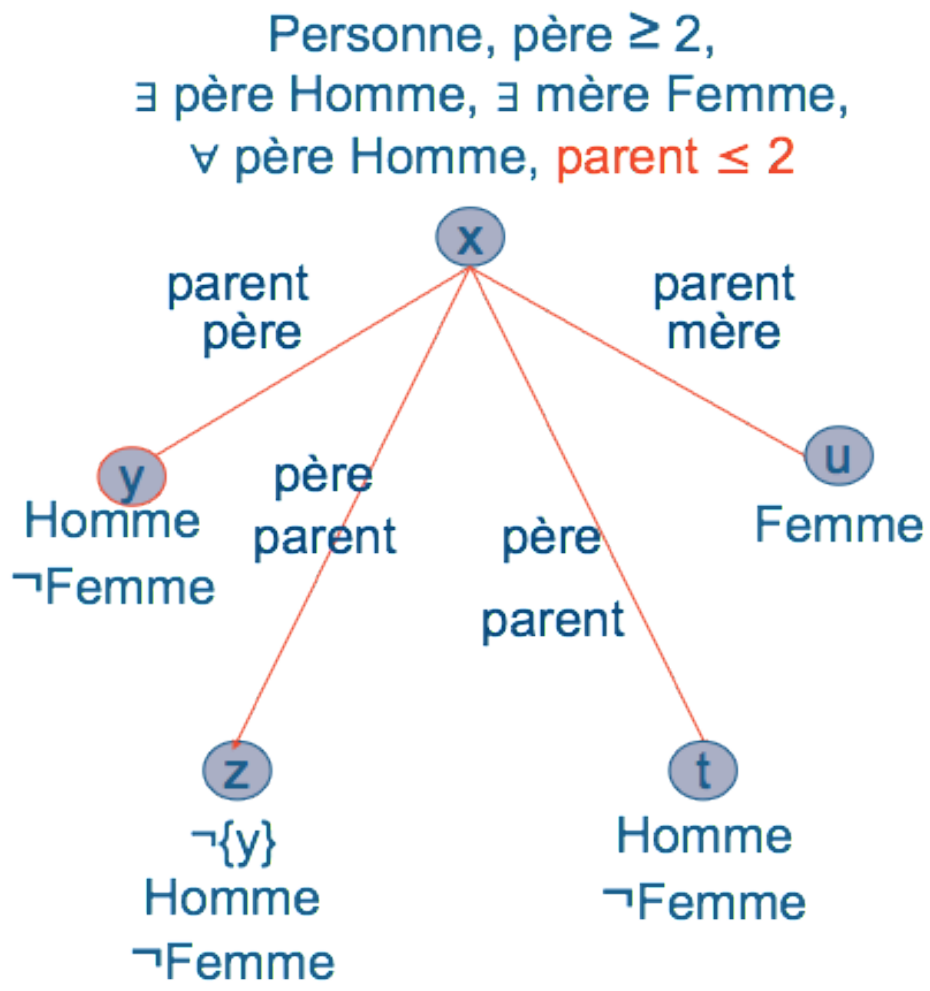
LD – Méthode des tableaux (8)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

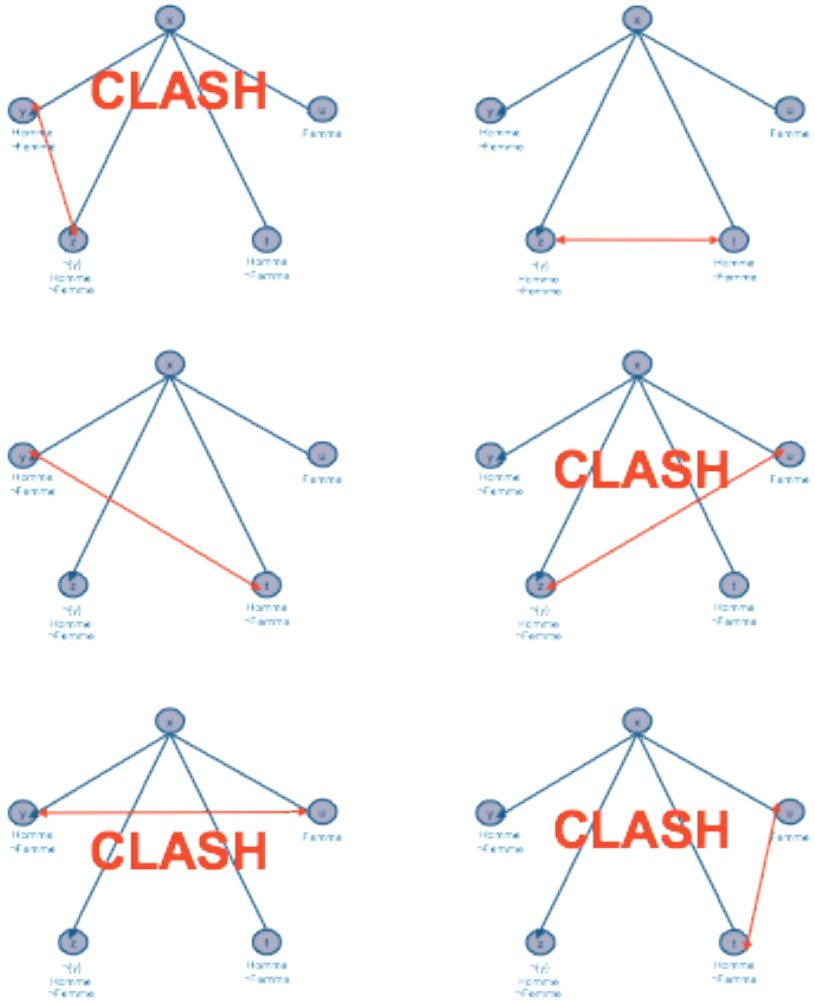
LD – Méthode des tableaux (9)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

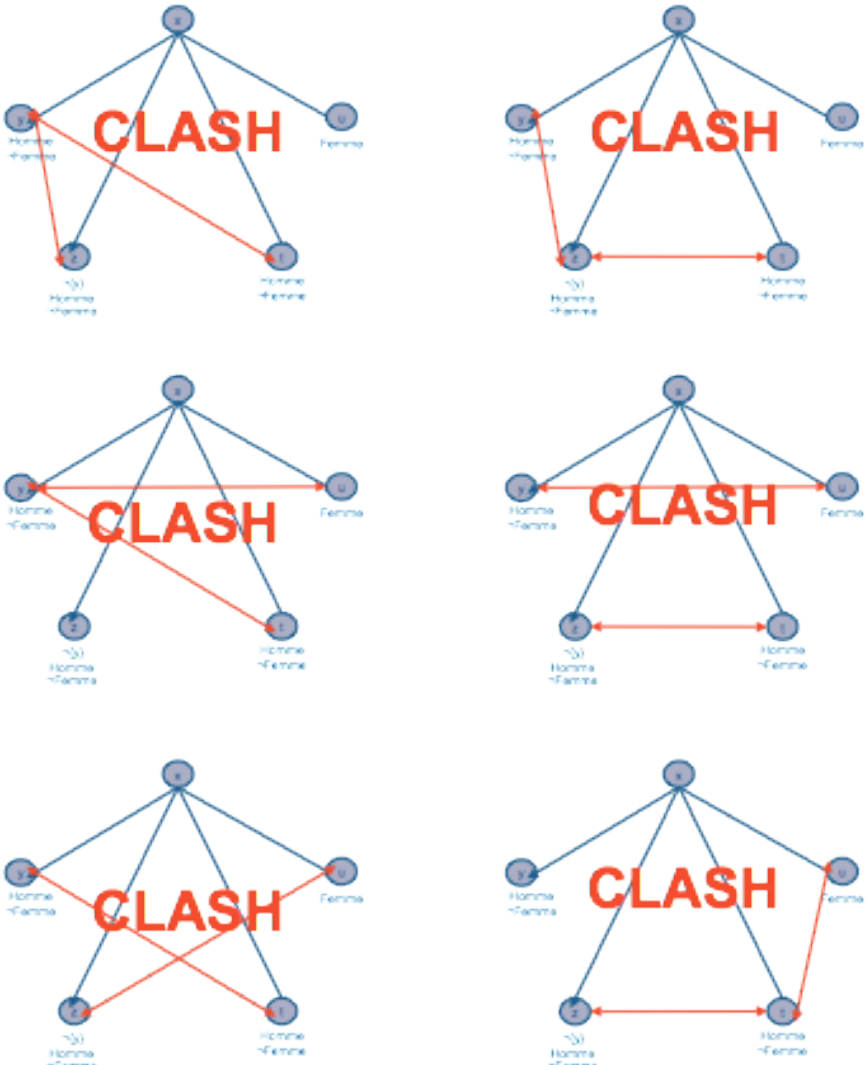
LD – Méthode des tableaux (10)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  est il satisfiable ?

LD – Méthode des tableaux (11)



- Homme ⊑ ¬ Femme
- Personne ⊑
  ∃ père Homme ⊓
  ∃ mère Femme ⊓
  ∀ père Homme ⊓
  parent ≤ 2
- père ⊑ parent
- mère ⊑ parent

- Personne ⊓ père ≥ 2
  n'est pas satisfiable ?

Familles de logiques de description

- Différentes familles se distinguent par les opérateurs qu'elles offrent
- Les opérateurs sont indiqués par des lettres
  - S : LD basique (ALC) plus rôles transitifs (e.g., ancestor R+)
  - H : hiérarchie de rôles (e.g., hasDaughter v hasChild)
  - O : description extentionnelle des classes (e.g., {Italy, France})
  - I : rôles inverses (e.g., isChildOf ≡ hasChild–)
  - N : restriction de nombre (e.g., >2 hasChild)
- DL de base + hiérachie de rôles + nominaux + inverse + NR = SHOIN
  - base de OWL-DL
  - SHOIN est très expressive, et encore décidable
    - possibilité de définir des raisonneurs efficaces

Raisonneurs sur les LD

- Cerebra Engine
  - is a commercial C++-based reasoner. It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, tree-conjunctive query answering using a XQuery-like syntax). It supports the OWL-API and comes with numerous other features.
- FaCT++
  - is a free (GPL/LGPL) open-source C++-based reasoner for SROIQ with simple datatypes (i.e., for OWL 2). It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval). It supports the OWL-API, the lisp-API and the DIG interface.
- KAON2
  - is a free (free for non-commercial usage) Java reasoner for SHIQ extended with the DL-safe fragment of SWRL. It implements a resolution-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, conjunctive query answering). It comes with its own, Java-based interface, and supports the DIG interface.

Raisonneurs sur les LD

- Pellet
- is a free open-source Java-based reasoner for SROIQ with simple datatypes (i.e., for OWL 1.1). It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, conjunctive query answering). It supports the OWL-API, the DIG interface, and the Jena interface and comes with numerous other features.
- RacerPro
- is a commercial (free trials and research licenses are available) lisp-based reasoner for SHIQ with simple datatypes (i.e., for OWL-DL with qualified number restrictions, but without nominals). It implements a tableau-based decision procedure for general TBoxes (subsumption, satisfiability, classification) and ABoxes (retrieval, nRQL query answering). It supports the OWL-API and the DIG interface and comes with numerous other features.
- Liste à jour sur http://www.cs.man.ac.uk/~sattler/reasoners.html

DIG

- DL Implementation Group
- http://dig.cs.manchester.ac.uk/
- The DIG Interface provides a specification of an interface for description logic reasoners. It is intended to be a lightweight mechanism providing access to reasoning functionality. There are many things that a reasoning service may be expected to provide that the DIG interface does not provide — it is expected that a DIG reasoner will be one component within a larger architecture.