

Rapport TP3
Applications en logique des descriptions
Représentation des Connaissances et Raisonnement 1

BOUROUINA Rania
181831052716

CHIBANE Ilies
181831072041

Résumé

Dans ce TP, nous allons manipuler des données ontologique en logique des descriptions en optant pour le raisonneur **Pellet** avec deux options d'outils :

- 1 - La librairie Python **OwlReady**.
- 2 - Le logiciel **WebProtégé**.

Nous commençons par la présentation de notre TBOX et ABOX

Tbox : Entités atomiques et composées

Cette box est composée d'entités atomiques ayant des concepts et des roles. Les premiers représentant des classes et les deuxièmes leurs méthodes. Par ailleurs, les entités composées sont des sous ensembles des concepts.

2.1 Tbox : Entités atomiques

2.1.a Concepts

Personne
Aliment
University

2.1.b Roles

Mange
Mange_par
Enseigne
Enseigne_par
PartieDe

2.2 Entités composées

Faculty : Sous ensemble de University
Departement : Sous ensemble de Faculty
Etudiant :Sous ensemble de Personne
Enseignant : Sous ensemble de Personne
Malbouffe : Sous ensemble de Aliment

2.3 Abox : Instances

Personne(Khellaf)
Pesonne(Mohamed)

Tool 1 : Python Owl Ready Library

3.0.a Code Source

```
1  ###
2  from owlready2 import *
3
4  onto = get_ontology("http://testxyz.org/onto.owl") #create ontology using iri
5
6  with onto: #defining our ontology
7
8      ##Defining concepts##
9      class Personne(Thing):
10         pass
```

```

11
12 class Aliment(Thing):
13     pass
14
15 class University(Thing):
16     pass
17
18 AllDisjoint([Personne, Aliment, University]) #pour dire qu'un individu ne peut pas etre une
19 personne et un aliment
20
21 ##Defining roles##
22 class mange(Personne >> Thing):
23     pass
24
25 class enseigne(Personne >> Thing): #persone est thing
26     pass
27
28 class enseigne_par(ObjectProperty):
29     inverse_property = enseigne
30
31 class mange_par(ObjectProperty):
32     inverse_property = mange
33
34 class PartieDe(Thing >> Thing):
35     pass
36
37 ##Defining composed entities##
38 class Faculty(Thing):
39     equivalent_to = [Thing & PartieDe.some(University)]
40
41 class Departement(Thing):
42     equivalent_to = [Thing & PartieDe.some(Faculty)]
43
44 class Enseignant(Personne):
45     equivalent_to = [Personne & enseigne.only(Personne)]
46
47 class Etudiant(Thing):
48     equivalent_to = [Personne & enseigne_par.only(Enseignant)]
49
50 ##defining instances ABOX##
51
52 class Mohamed(Thing):
53     equivalent_to = [Personne & mange.only(Aliment)]
54
55 class Khellaf(Personne):
56     equivalent_to = [Enseignant & mange.some(Aliment) & enseigne.only(Etudiant)]
57
58 class MalBouffe(Thing):
59     equivalent_to = [Aliment & (mange_par.some(Personne))]
60
61 AllDisjoint([Etudiant, Enseignant])
62 AllDisjoint([Khellaf, Mohamed])
63 AllDisjoint([MalBouffe, Departement, Faculty, University])
64
65 sync_reasoner_pellet(infer_property_values=True)
66 onto.save(file = "tp_rc1.owl", format = "rdxml")
67
68
69 ###
70 with onto:
71
72     USTHB = onto.University()
73
74     Ranya = onto.Etudiant()
75
76     Chocolat = onto.Aliment()
77
78     Moulai = onto.Personne()
79
80
81     SI = Thing() #department

```

```

82 INFO = Thing() #faculty
83
84 INFO.PartieDe.append(USTHB)
85 SI.PartieDe.append(INFO)
86
87 Ranya.mange = [Chocolat]
88 Moulai.enseigne = [Ranya]
89
90 sync_reasoner_pellet(infer_property_values=True)
91 onto.save(file = "tp_rc2.owl", format = "rdxml")
92
93 # %%

```

Voici les résultats déduits par le raisonneur :

Tool 2 : WebProtégé

Voici l'interface de l'outil Web Protégé :

```

* Owlready2 * Pellet took 1.63551926612854 seconds
* Owlready * Reparenting onto.MalBouffe: {owl.Thing} => {onto.Aliment}
* Owlready * Reparenting onto.Khellaf: {onto.Personne} => {onto.Enseignant}
* Owlready * Reparenting onto.Etudiant: {owl.Thing} => {onto.Personne}
* Owlready * (NB: only changes on entities loaded in Python are shown, other changes are done but not listed)

```

FIGURE 1 – Résultats déduits sur les instances

Ici, le raisonneur a déduit que :

- La Mallbouffe est un Aliment.
- Khellaf est Enseignant(e).
- Etudiant est une Personne.

NB : Seulement les instances changées sont mentionnées par le raisonneur.

```

* Owlready * Reparenting onto.aliment1: {onto.Aliment} => {onto.MalBouffe}
* Owlready * Reparenting onto.thing1: {owl.Thing} => {onto.Departement}
* Owlready * Reparenting onto.thing2: {owl.Thing} => {onto.Faculty}
* Owlready * Reparenting onto.personne1: {onto.Personne} => {onto.Enseignant}
* Owlready * (NB: only changes on entities loaded in Python are shown, other changes are done but not listed)

```

FIGURE 2 – Résultats du raisonneur

Ici, le raisonneur a déduit que :

- Aliment1 (Chocolat) est une Malbouffe.
- Thing1 (Objet) est un departement.
- Thing2 (Objet) est une Faculté.
- Personne1 est un(e) enseignant(e).

Tool 2 : WebProtégé

Voici l'interface de l'outil Web Protégé :

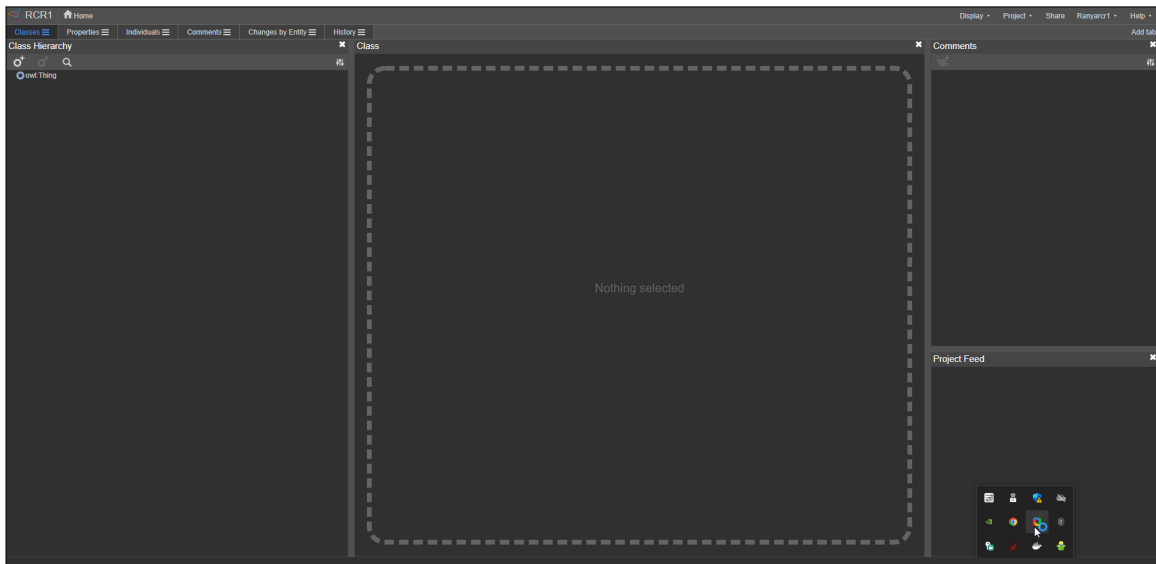


FIGURE 3 – Interface de WebProtégé

Voici donc notre description :

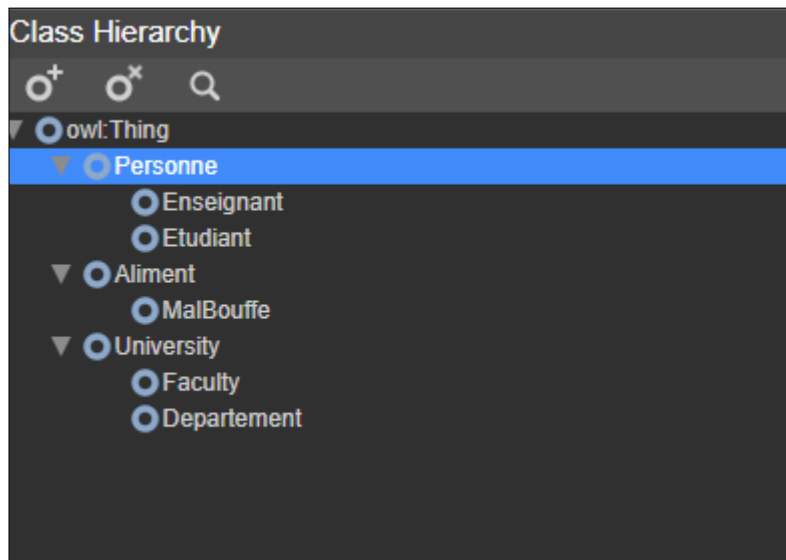


FIGURE 4 – Concepts et entités composées

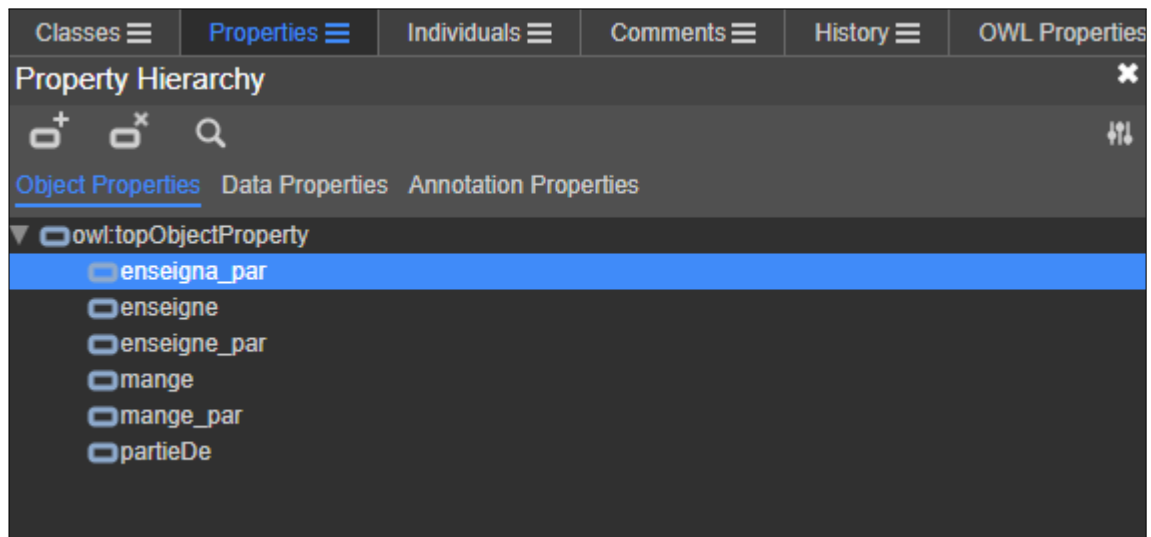


FIGURE 5 – Roles

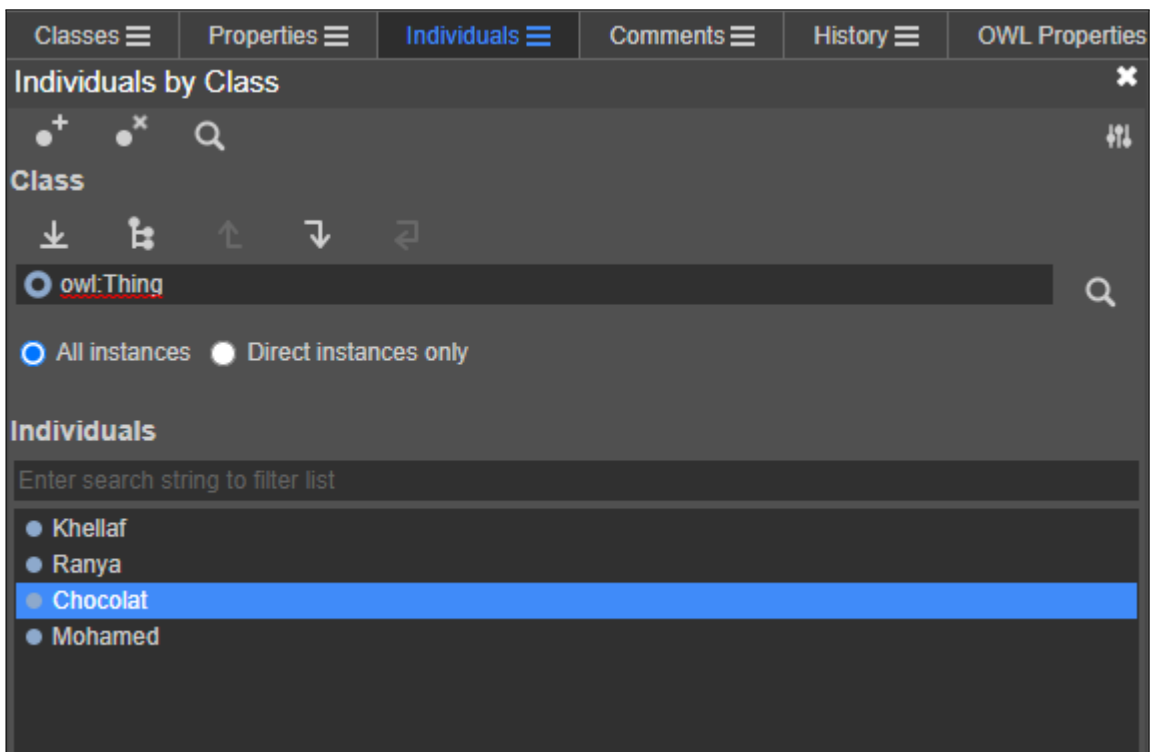





FIGURE 6 – Instances


Individual: Chocolat






IRI

http://webprotege.stanford.edu/RDTeSJcXe3AcDiSNjOn3RIm

Annotations


rdfs:label


Chocolat


Enter property


Enter value

Types

Enter a class name

Relationships


mange_par


Mohamed

Enter property

Enter value

Same As

Enter an individual name

FIGURE 7 – Exemple details d'un individu

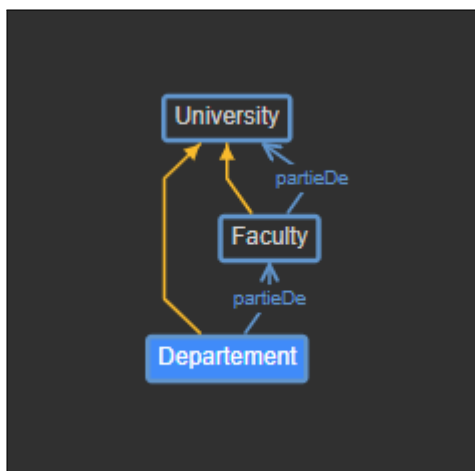


FIGURE 8 – Schéma de departement

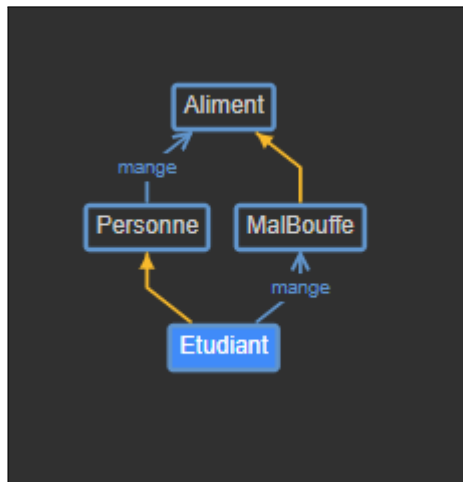


FIGURE 9 – Schéma de étudiant

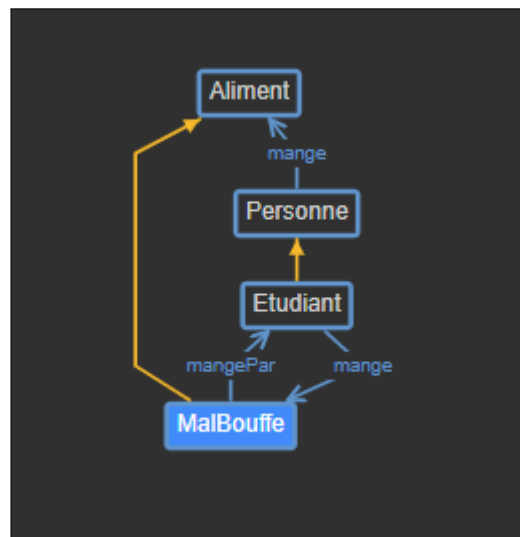


FIGURE 10 – Schéma de MalBouffe

Individual: Ranya

IRI
http://webprotege.stanford.edu/R9byeWKSjb8Zmwk9dyuiAC

Annotations

↔ rdfs:label Ranya lang ✕

Enter property Enter value lang

Types

○ Etudiant ✕

Enter a class name

Relationships

☐ mange ○ MalBouffe ✕

Enter property Enter value lang

Same As

● Mohamed ✕

Enter an individual name

FIGURE 11 – Dédution pour Ranya

Individual: Khellaf

IRI
http://webprotege.stanford.edu/R9NsdKsPYcVoFoVmETx7vRu

Annotations

↔ rdfs:label Khellaf lang ✕

Enter property Enter value lang

Types

○ Personne ✕

Enter a class name

Relationships

☐ enseigne ● Ranya ✕

☐ mange ○ Aliment ✕

Enter property Enter value lang

Same As

Enter an individual name

FIGURE 12 – Dédution pour Khellaf

5.1 Conclusion

Nous avons remarqué que les deux outils sont différent à utiliser. La librairie Python donne un peu plus de flexibilité à l'utilisateur le laissant créer n'importe quelle hiérarchie de classes et relations. Cependant, ceci peut induire à une logique non consistante. Dans ce cas là, l'outil protégé est le choix optimale pour avoir une logique bien définie.