Document et Web Semantique - TP OWL avec Protégé

L'objectif de ce TP est de créer à l'aide du logiciel Protégé un schéma ontologique décrivant la famille [1]. Nous instancierons un exemple afin d'identifier les inférences possibles.

1 Protégé

Protégé (`http://protege.stanford.edu/`) est un éditeur d'ontologie permettant entre autres de déclarer :

— des classes OWL ;
— des propriétés OWL (owl:DatatypeProperty et owl:ObjectProperty) ;
— des individus ;

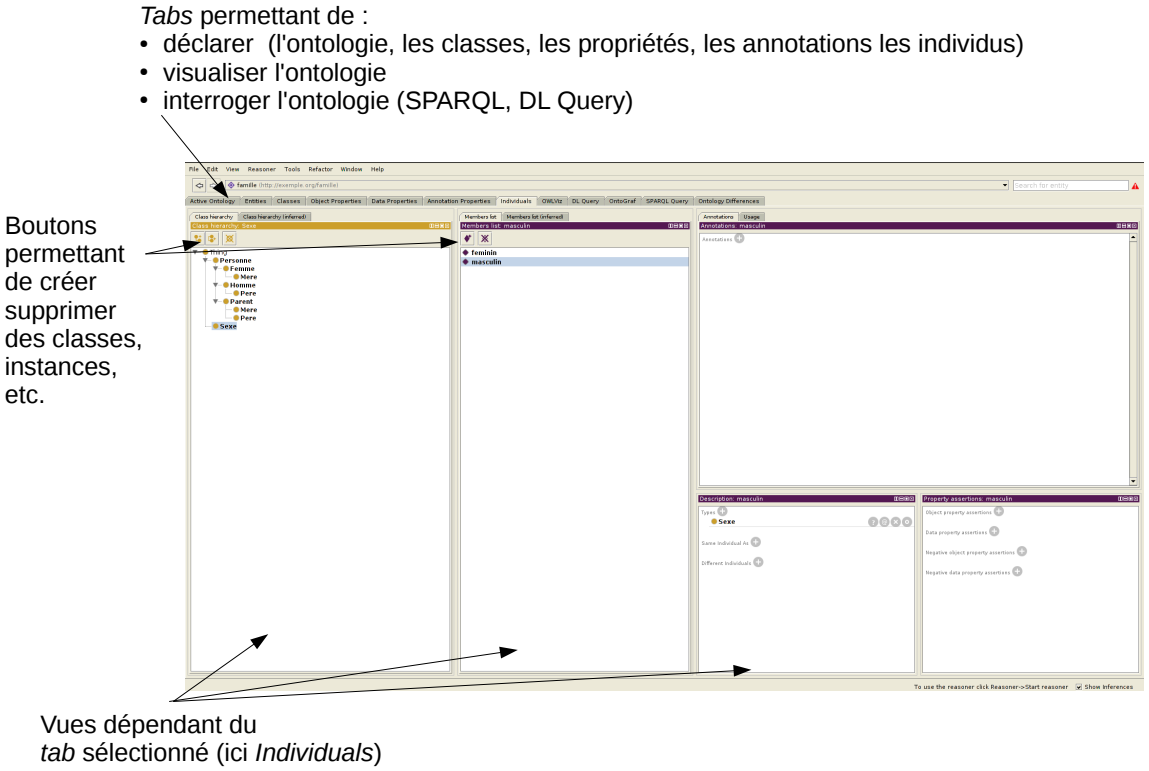

FIGURE 1 – Interface de l'éditeur Protégé

1. Cette ontologie ne reflète pas la société, c'est un exemple jouet :-)

Il permet aussi d'utiliser des raisonneurs (suivant les versions de Protégé : Fact++, Hermit, Pellet, etc.) pour inférer des nouveaux faits. Enfin il propose des requéteurs (SPARQL, DL Query), un visualiseur, l'export de documentation, etc.

Comme l'indique la figure 1, la plupart des déclarations se spécifient en sélectionnant des éléments graphiques (bouton, case à cocher, etc.) mais certaines (par exemple lors de la création de classe anonyme) nécessitent d'utiliser la syntaxe de Manchester. Le tableau 1 rappelle les principaux mots clés proposés par cette syntaxe.

OWL	Manchester	Exemple
someValuesFrom	some	`enfant some Homme`
allValuesFrom	only	`soeur only Femme`
hasValue	value	`paysDOrigine value angleterre`
minCardinality	min	`enfant min 3`
cardinality	exactly	`enfant exactly 3`
maxCardinality	max	`enfant max 3`

TABLE 1 – Principaux mots clés de la syntaxe Manchester

Ainsi pour exprimer qu'un homme est une personne de genre masculin, il faut créer une classe Homme :

— sous classe d'une classe `Personne` (utilisation de l'interface graphique) ;
— sous classe d'une classe anonyme qui limite la relation `genre` à la valeur `masculin`.

2 Création de l'ontologie

L'URI de l'ontologie que nous allons créer est `http://exemple.org/famille`.

2.1 Personne et Genre

1. Créer la classe `Personne` équivalente à la classe `Person` de l'ontologie FOAF.
2. Créer la classe `Genre` avec deux instances non identiques : `masculin` et `feminin`.
3. Créer la propriété `nom` qui permet de décrire le nom du personne.
4. Créer la propriété `genre` qui permet de fixer le genre d'une personne (attention cette propriété a certaine(s) caractéristique(s)).

2.2 Homme, Femme et leurs instances

1. Créer la classe `Homme` comme étant une personne de genre masculin.
2. Créer la classe `Femme` comme étant une personne de genre féminin (les femmes ne peuvent pas être des hommes et inversement)
3. Créer les individus (tous différents les uns des autres) :
   — Homme : jacques, louis, pierre, gerard, jean, patrick
   — Femme : monique, germaine, yvette, therese, muriel, sandrine, astride

2.3 Parent, Mère, Père, etc.

1. Créer la propriété `enfant` et ses sous-propriétés `fille` et `fils`.
2. Créer la propriété `parent` comme étant l'inverse de la propriété `enfant` et ses sous-propriétés `mere` et `pere`.
3. Créer la classe `Parent` comme étant une personne ayant au moins un enfant.
4. Créer la classe `Mere` comme étant à la fois une femme et un parent.
5. Créer la classe `Pere` comme étant à la fois un homme et un parent.
6. Créer la propriété `frereSoeur` qui est symétrique et transitive et ses sous-propriétés `soeur` et `frere`.
7. Créer la propriété `grandParent` à partir de la propriété `parent`.
8. Créer les relations entre les instances à l'image de la figure 2 en utilisant uniquement les propriétés `fille`, `fils`, `soeur` et `frere`

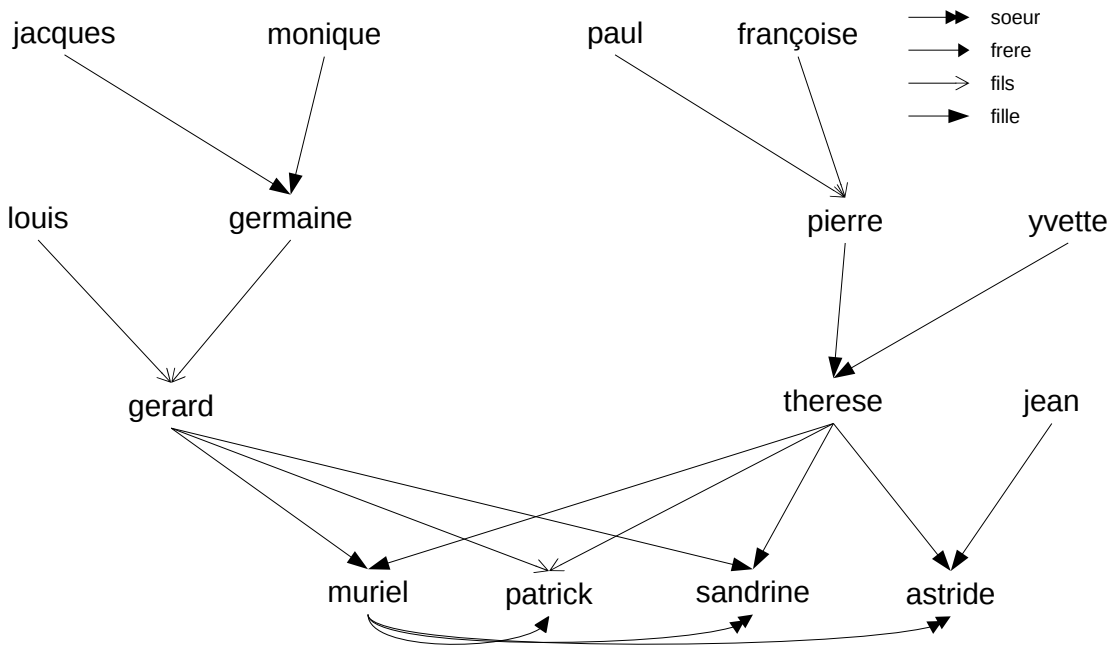

FIGURE 2 – Les liens entre individus

3 Requêtages et inférence

3.1 Requêtages sans inférence

1. Donner la requête SPARQL permettant de lister l'ensemble des hommes.
2. Pourquoi la requête SPARQL suivante ne retourne aucune instance ?

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX fam: <http://exemple.org/famille/>
SELECT ?p WHERE {
```

```
?p rdf:type fam:Personne
}
```

3. Proposer une requête SPARQL permettant d'obtenir toutes les personnes.

3.2 Requêtages avec inférence

1. Lancer le moteur d'inférence. Quelle inférence le moteur ne semble pas pouvoir trouver ?
2. Proposer une requête DL (DL query) pour trouver les soeurs de patrick.
3. Créer la propriété `petitEnfant`.
4. Proposer une requête DL pour trouver les grands parents de sandrine.