5. Unit: [Lab Course] OwlAPI & Protégé

A group of farmers join together to sell their vegetables, dairy products and meat directly to customers. Since the prices are higher than in the supermarket, the farmers do not only want to be qualitatively better, but also to give information about special nutritional restrictions, such as vegetarian, vegan, halal, kosher and various food intolerances.

Since these rules are complicated and food must not be misclassified, an ontology should be made and a reasoner should automatically classify the food. It can be assumed that producers know what is in their products and what kind of animals it came from, but they do not know what food restrictions it satisfies. E.g. it can be assumed that a farmer knows that his hens hatch with feathers, but not whether eggs kosher or not.

Further you can ignore any information with (ritual) for the sake of simplification, even it might be the most important part in those cases, it does not contribute to the learning goal and makes things a lot more complicated since it is not directly related with the food but its processing.

Exercise 5.1 (Protégé - Ontology generation) Write an ontology that describes the rules for the food restriction from below.

- **Halal** is everything, except it is forbidden. Following (simplified and shortened) restrictions hold for halal:
  - **Halal Animals** are vegetarians and are not explicit forbidden like pigs (irrelevant in this cases but for your information: mice, rats, insects (except locusts) and all predators are also forbidden)
  - **Meat** is allowed form animals which are considered halal. In addition, slaughter must meet special ritual conditions (ritual).
  - **Milk and Eggs** are allowed if the animal is considered halal.
  - **Alcohol** is forbidden.
  - All **Plants** are halal.
- **Kosher** following (simplified and shortened) restrictions hold for kosher:
  - **Kosher Animals** are not predators, if they live on the ground, animals must be cloven hoofed (Paarhufer) and ruminants (Wiederkäuer). If they can fly they have to hatch with feathers and are precocial (Nestflüchter).
  - **Meat** is allowed form animals which are considered kosher. In addition, slaughter must meet special ritual conditions (ritual).
  - **Milk and Eggs** are allowed if the animal is kosher, in case of cheese the rennet must be from a kosher slaughtered animal (ritual).
  - **Alcohol** is allowed as long as all ingredients are kosher and in case of wine was supervised during production by jews (ritual) and as long as it's not used for religious services of other religions (ritual).
  - All **Plants** are kosher.
- **Food intolerances & restrictions**
  - **Vegetarian Food** everything which does not include meat or fish.
  - **Vegan Food** everything which is not from or produced by an animal.
  - **Lactose** is contained in milk, quark, cream, cream cheese and partly in chocolate
  - **Gluten** is found in grains.

Exercise 5.2 (OWLAPI - Adding new Classes) Use OWLAPI to write a program to add animals, plants and products as described below. The program should have the following methods:

- addLifeform([name], [List of Classes]) - to add a new Animal or Plant
- addBasicProduct([name],[property],[LifeForm]) - to add a new Product which is directly produced by/from a life form.
- addCompoundProdut([name],[ListOfParts]) - to add a new Product which is created of multiple other products.

Then add all products, plants and animals with their properties and classes from below, but only based on the knowledge mentioned as known to the farmer.

Hint: You may have to generate or change classes as well as individuals

The following information is known to the farmers and can be used to generate individuals: **Animals:**

- **cows**
- **chickens**
- **sheep**
- **goats**
- **rabbits**
- **ostriches**
- **turkeys**
- **pigs**
- **horse**
- **bees**
- **pigeons**

Plants:

- **wheat**
- **barley**
- **rye**
- **carrot**
- **potato**
- **salad**
- **pepper**
- **pumpkin**
- **grape**
- **hop**

Following Products are available

- **Meat** from all animals
- **Sausage** made form pork (pigs), beef (cows) or horse
- **Gelatin** made from pig skin and bones
- **Milk** from cows, sheep or goats
- **Rennet** (Lab) from cows
- **Cream Cheese** from cows or goats
- **Mature Cheese** from cows or sheep
- **Quark** from cow
- **Egg** from chicken, turkey, pigeon or ostrich
- **Red Wine** is made from grapes (but also contains small pieces of the filtration medium)
- **Beer** is made from barley and hops
- **Bread** is made form wheat flour and rye flour
- **Cheese** is made from milk and rennet
- **Flour** made from wheat, barley or rye
- **Vegetables** in form of tomato, potato, carrot, salad, pepper or pumpkin
- **Honey** from bees

All farmers know that:

- **cloven hoofe** animals are **cows, goats, pigs and sheep**
- **ruminant** animals are **cows, goats, horses and sheep**
- There are no products from **predators**

- **Wine** and **beer** contain alcohol
- Birds are the only kept animals which lay eggs
- All their kept birds are **precocial** birds and are **born with feathers** except **pigeons**
- The only kept animals which are not earthbound are chickens, pigeons, turkeys and bees.
- **Gelantin** is used during the filtration of **wine**.

Exercise 5.3 (Hermit) Now use the HermitAPI to validate the ontology and try to determine for each Product, which food restriction may hold for it and which do not. Very likely you have to make changes in the ontology, if so remember that the Hermit API can aid you with reports.

Write a method which accepts the name of a product and returns all its classes.

Hint: Cheese often contains animal rennet. Animal rennet is a mixture of enzymes obtained from the stomach of milk drinking mammals. Therefore the cheese in this case is not vegetarian by definition.

Last but not least some quick reminders:

- Use multiple inheritance! A lot!
- Make disjoint things disjoint
- To the best of our knowledge, there no way in OWL to define DisjointUnion over all subclasses without mentioning each subclass explicit. Do it programmatically.
- To derive classes which are defined as not/only having/being something you have to close the world! Especially for classes which are not allowed to have a property ranging over certain classes it must be proven that their could not be any other unknown property which might range over one of those classes.
- Individuals may be the same, even if they have different IRIs, at least if you did not said otherwise.
- A class can not "have" a property, but properties can have domain and range over classes.