TP: OWL & Protégé

Exploring Protégé with the Pizza Ontology

This series of exercises is based upon the pizza tutorial, developed by the University of Manchester.

- 1. Load the pizza ontology file (from the course website).
- 2. Browse class and property tabs.
- 3. Compare the axioms for Margherita and VegetarianPizza.
 - Write the definition of VegetarianPizza as a DL axiom.
 - Likewise, translate the subclass statements for Margherita into DL inclusions.
 - Is a Margherita pizza a VegetarianPizza? Why / why not?
- 4. Compare the axioms for InterestingPizza and Mushroom.
 - Is a Mushroom pizza an InterestingPizza? Why / why not?
- 5. Examine the property hasIngredient.
 - What is the domain and range of this property?
 - What are its subproperties?
 - What is the inverse property of hasIngredient?
 - What property characteristics does hasIngredient have?
- 6. Examine the definitions of the following three classes: VegetarianPizza, VegetarianPizzaEquivalent1, and VegetarianPizzaEquivalent2.
 - Are any of these classes equivalent? Why or why not?
- 7. Start the reasoner (with Hermit selected as the reasoner).
 - Browse the inferred class hierarchy. Observe what new information is now present (indicated by light yellow background).
 - Check whether you correctly predicted whether Margherita is a VegetarianPizza, and whether Mushroom is an InterestingPizza.
 - Check which of the classes VegetarianPizza, VegetarianPizzaEquivalent1, and VegetarianPizzaEquivalent2 are equivalent.
- 8. Pick a pizza which is classified as an InterestingPizza, use the explanation facility (button '?') to understand why.
- 9. Locate the two unsatisfiable classes (these are listed under owl:Nothing and marked in red).
 - For each unsatisfiable class, determine what caused the problem. Try to solve this first by inspecting the axioms, and otherwise, use the explanation facility.
 - Modify the ontology as needed so that there are no unsatisfiable classes (to launch the reasoner again, you can use "Synchronize reasoner" in the Reasoner menu).

- 10. Add a new pizza with the name and definition you choose.
 - Add annotations (using comment annotation property) to provide a textual descriptions of your pizza, in both English and French.
 - Try to determine whether your pizza is a Meaty / Interesting / Cheesey Pizza, then run the reasoner to check.
- 11. Add a data property hasPrice.
 - Mark hasPrice as functional and with range integer.
 - State that every pizza has a positive price using the following class expression: hasPrice some xsd:integer[> 0]
 - State that FourSeasons has a price of 19.
 - State that Margherita has a price of 10.
 - Define new classes PremiumPizza and BasicPizza, defined as those pizzas having a price of >= 15 and < 15, respectively.
 - Run the reasoner to check that FourSeasons is a PremiumPizza, Margherita is a BasicPizza, and PremiumPizza is disjoint with BasicPizza.

Ontology Construction

Now that you've mastered the basics, you can explore how to build your own ontology on a new topic. (Alternatively, you can extend the pizza ontology, e.g. to cover a new category of foods (pasta, sandwiches, ...).

- 1. Pick a domain to model, and make a list of important terms.
- 2. Decide which terms should be classes, which should be properties.
- 3. Organize classes into hierarchy (recall that 'C subClassOf D' means that every individual in C must belong to D).
- 4. Enter your class hierarchy in Protégé.
 - Also add any appropriate disjointness statements.
- 5. Add the properties into Protégé.
 - Specify any relationships that hold between them (inverse? subproperty?) and any relevant characteristics (functional? transitive?).
 - Specify the domain and/or range, whenever appropriate.
 - Run the reasoner to make sure there are no problems.
- 6. Return to your class hierarchy and consider what further axioms should be added to capture the meaning of the classes.
 - Add your axioms into Protégé, regularly running the reasoner to check for problems.
- 7. What individuals and assertions make sense in your domain?
 - Add a few individuals and assertions.
 - Run the reasoner to see what you can infer about the named individuals.