IDS internal training - ontologies

Hands-on task 1: authoring an ontology in Protégé

For this task we will build a simple ontology about movies

A. Initialise your new ontology and give it some metadata:

1. Download and install the latest version of Protégé Desktop from https://protege.stanford.edu/products.php#desktop-protege **Note:** there is also a cloud version of Protégé called *WebProtégé* - https://webprotege.stanford.edu/ for collaborative ontology development in the cloud. However, it does not have all the features of Protégé Desktop which we require for this task.
2. Fire up Protégé.
3. Make sure the “Active Ontology” tab is selected. In the “Ontology header” section give your ontology a unique name and version in the “Ontology IRI” and “Ontology Version IRI” textboxes respectively e.g. http://www.maastrichtuniversity.nl/ids/ontologies/movie/movie.owl and http://www.maastrichtuniversity.nl/ids/ontologies/movie/1.0.0
4. Add metadata to your ontology (e.g. a title, description and authors) using the “+” button next to “Annotations”.
5. Save the ontology (click “Save” in the “File” menu). In the dropdown box that pops up choose a syntax e.g. OWL/XML. You can name the file something like “movies.owl”.

B. Add some movie terms to the ontology, define them and notice both intended and unintended inferences:

6. Navigate to the “Entities” tab and make sure the “Classes” sub-tab is selected. This sub-tab displays the taxonomy or hierarchy of classes in your ontology.
7. Create new classes called “Movie”, “Female”, “Person” and “Actress” by selecting the “owl:Thing” class and clicking one of the buttons at the top of the class hierarchy window. The leftmost button creates a child class for the selected class. The middle button creates a sibling class and the last button deletes the selected class.
8. An “Actress” should surely be a subclass of “Person”, right? You can ensure this by dragging and dropping the “Actress” class over the “Person” class. Try to also specify that “Actress” is a subclass of “Female”.
9. How can we define “Actress” further? How about as an entity that plays a role in a movie? But first we need to add a relation term (lets call it “playsARoleIn”) to our

ontology. You can do this on the “Object properties sub-tab of the “Entities” tab. The “Object properties” sub-tab is the analogue of the “Classes” sub-tab for relations. After adding “playsARoleIn”, click on “Actress” in the “Classes” sub-tab. On the right hand side of the window, notice the “Description: Actress” view. Click on the “+” button next to “Equivalent To” in this view. In the resulting window, type in “playsARoleIn some Movie” and click “OK”.

10. Now let’s add some individuals! Go to the “Individuals by class” tab. Select the “Person” class in the “Class hierarchy” view. In the “Instances” view click on the diamond-shaped button to add a new instance of “Person”. Name this person “Julianne_Moore”. In a similar manner, add an instance of “Movie” called “Jurassic_Park”
11. Select “Person” in the “Class hierarchy” view and make sure “Julianne_Moore” is selected in the “Instances” view. On the far right of the “Individuals by class” tab, notice the “Property assertions: Julianne_Moore” view. Click the “+” button next to “Object property assertions”. Enter the object property name “playsARoleIn” in the left textbox and the movie name “Jurassic_Park” in the right textbox. Click “OK”. Now go to the “Reasoner” menu in Protégé, make sure HermiT is selected and click on “Start reasoner”. What did you notice?
12. Now click on “owl:Thing” in the “Class hierarchy” window and an instance of this class called “T_Rex”. Add the information that “T_Rex playsARoleIn Jurassic_Park”. Click on the “Reasoner” menu and click “Synchronize reasoner”. What do you notice now?
13. Save the ontology (click “Save” in the “File” menu).

C. Understanding logical errors that may arise when building ontologies:

14. Navigate back to the “Entities” tab and the “Classes” sub-tab. Add the classes “Theater”, “LumiereTheater”, “PatheTheater” and “Popcorn”. Also, add the relation “serves” in the “Object properties” sub-tab. Finally, add an instance of “LumiereTheater” called “Maastricht_Lumiere”.
15. Now add the following information to the ontology using the “+” button next to “SubClass Of” in the “Description” pane: “LumiereTheater SubClassOf Theater”, “Theater SubClassOf serves some Popcorn” and “LumiereTheater SubClassOf not (serves some Popcorn)”. Click on the “Reasoner” menu and either start or synchronize the reasoner. What do you find?
16. Save the ontology (click “Save” in the “File” menu).

Hands-on task 2: associating data with ontologies

For this task we will annotate data files with ontology terms and allow an existing software application import this data and derive useful new insights from it. The software loads an existing ontology “resources/diseases.owl” which it uses to “understand” the HTML data files it loads from “resources/data/”.

First things first: clone or download the github repo located here:
https://github.com/MaastrichtU-IDS/ontologies-workshop

17. Open UMMedSystem.jar in the root directory (either from command line using “java -jar UMMedSystem.jar” or double click to launch)
18. Click on “Disease Classification” **Note:** classification here does not refer to Machine Learning but rather to a notion of *categorisation.*
19. Select different categories of diseases from the drop-down menu. You might find that no diseases are found in the “Cancer” category.
20. Open “resources/data/breastcancer.html” in your favourite text editor. Each HTML file in “resources/data” contains information about a single disease. Uncomment Lines 11 and 13 in this file and save (overwrite) it. Reopen UMMedSystem.jar and check if there are any cancer diseases identified by the system. The system says Breast cancer is a carcinoma but this information is not in “resources/data/breastcancer.html”. Cool huh?
21. Fire up Protégé and open the file “resources/diseases.owl”. Go to the “DL Query” tab. Type in “:BreastCancer” in the “Query (class expression)” textbox. Click on the “Reasoner” menu and click “Start reasoner”. Click on “Execute” in the “DL Query” tab. On the far right, make sure all checkboxes under “Query for” are ticked. What do you notice? Click on the “?” button next to the “:Carcinoma” class in the “Query results” panel. This is an explanation for the inference that “:Cancer” is a “:Carcinoma” in the ontology.
22. Open “resources/data/heartdisease.html” and “resources/data/lungcancer.html” in your browser. Can you spot which symptoms are common to both diseases? Now close and reopen UMMedSystem.jar and click on the “Disease Comparison” button. Select these two diseases and see which symptoms are common. What do you find?
23. In the file “resources/data/lungcancer.html” uncomment Lines 25 and 27. Close and reopen UMMedSystem.jar and try again to compare the common symptoms it finds. This is the very simple yet powerful act of annotating data with ontology terms!