Lab: Ontology and Ontology Development in OWL

After this lab you should get a good understanding of ontology, OWL and ontology development, and the ontology development tool, Protégé.

Reading Materials

- Lecture slides
- Guarino, N., Oberle, D., & Staab, S. (2009). What is an ontology?. Handbook on ontologies, 1-17.
- OWL 2 Web Ontology Language Primier.

Submission

You should submit a document containing the answers to the questions in task 1, 2 and 4. For Task 5, please submit the document with the outcomes from steps 1 and 2, document the steps of the conceptualization process, and the file of the OWL ontology. **Note:** Any form of plagiarism, including using AI to generate answers, will result in failing the lab.

1. Ontology Definition

Explain this definition of ontology: *An ontology is an explicit, formal specification of a shared conceptualization.*

2. Web Ontology Language (OWL)

Describe the OWL 2 DL and OWL 2 Full, and the profiles OWL 2 EL, OWL 2 QL, and OWL 2 RL.

3. Using Protégé

Download Protégé. *Note*: no need to do the registration. Choose "No, thanks. I'm already registered".
How to use Protégé, please check the document Protégé 5 documentation, especially the introduction and guides on the tabs and views in the tool.

1. **“An ontology is a formal, explicit specification of a shared conceptualization”**

The definition of 'explicit specification' refers to that all elements of an ontology are explicitly defined. The 'formal' specification means that the ontology is described in a language with a formal syntax and semantics, allowing for machine interpretable ontology descriptions. 'Shared conceptulization' implies that an ontology is represented with a collected view of knowledge that has been agreed on by a group of people.

2. Each of these originate from the second edition of Web Ontology Language that became the standard in 2009.

OWL 2 RL, OWL 2 QL, OWL 2 EL $\subseteq$ OWL 2 DL $\subseteq$ OWL 2 Full

OWL 2 DL: Description logic used Web Ontology Language that provides well-defnied semantics used in cases where high expressiveness is of great importance. It also allows for complex reasoning.

OWL 2 Full: Contains the syntax provided in OWL 2 DL + features from RDFS

These dialects of OWL 2 are all a bit more restrictive when it comes to expressiveness compared to OWL 2 DL but are less complex when it comes to reasoning which makes them faster.

OWL 2 EL (Existential Language): Has great expressive power and is well suited for ontologies with large amounts of classes and properties. Basic reasoning can be made in polynomial time relative to the size of the ontology.

OWL 2 QL (Query Language): Is best fitted for applications using larger amounts of instance data and query answering is of higher priority. It has limited expressive power.

OWL 2 RL (Rule Language): Is mainly used in applications who require scalable reasoning without significant loss of expressive power. RL reasoning systems can be used with rule-based reasoning engines.

4. The Family Ontology

In this task you will get familiar with the ontology editor, Protégé, and also ontology in OWL, by looking at a family ontology, which the sample ontology used in OWL 2 Web Ontology Language Primer.
Download the *families.owl* in Canvas. Go to the tab `Entities`, take some time to browse the classes, object properties, datatype properties and individuals, understand the ontology and solve the exercises below:

1) Find the classes HappyPerson, JohnsChildren, NarcisticPerson, Orphan and Grandfather in the ontology, give the descriptions of the classes using DL syntax and natural language.

2) Find the classe Teenager in the ontology, give the description of the class in natural language.

3) Find the object properties hasFather, hasGrandparent and hasUncle, give the descriptions of the object properties using DL syntax and natural language.

4) Find the individual John, give its description using natural language.

5) Choose the `HermiT` reasoner, then click on `Start Reasoner` in the Reasoner menu. Once the reasoning is done, browse through the classes, properties, and individuals in the ontology to observe the changes (highlighted in yellow). Select one change each from a class, an object property, and an individual. For each, describe what the change is and explain why it was generated by the reasoning process.

5. An Ontology for Educational Resources in European Universities

In this task you will develop an ontology from scratch. You can imagine that the ontology will be used to support the systems that integrate the data from different European universities about their educational resources. The system will serve students to look for courses and programs and/or create recommendations on courses/programs based on students' preferences.

1) Write a text describing the typical information that should be managed by the systems. In this step you define the purpose and scope of the ontology.

2) You can reuse the knowledge base created in the task 4 of the DL lab.

3) Prepare a list of competency questions (CQs), additional restrictions, and potential reasoning requirements that could be elicited from there. In this step you define the

classes HappyPerson, JohnsChildren, NarcisticPerson, Orphan and Grandfather

4.

1) $HappyPerson \equiv \exists hasChild.HappyPerson \sqcap$

A happy person is someone who has at least one child and all of their children are happy people.

$JohnsChildren \equiv Person \sqcap \exists hasParent.John$

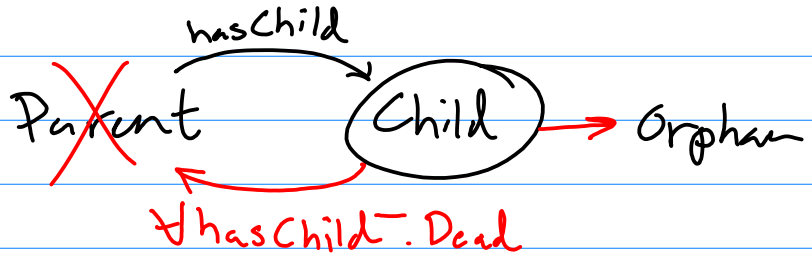
Is a person how has a parent named John.

$NarcisticPerson \equiv Person \sqcap \exists loves.Self$

NarcisticPerson is someone who loves him/her self.

$Orphan \equiv \forall hasChild^{-}.Dead$

An orphan is someone who no longer has any parents. (Inverse relation)



$GrandFather \sqsubseteq Parent \sqcap Man$

$GrandFather \equiv Man \sqcap \exists hasChild.Parent$

A GrandFather is someone who is a man and is a parent to a parent

2) A Teenager is a person with an age grater than 12 and less or equal to 19.

3) $hasFather \sqsubseteq hasParent$

$hasFather \equiv \forall hasChild^{-}.Man$

The hasFather relationship is the inverse relation between a child and a parent who is a man.

$hasGrandparent \equiv hasParent \circ hasParent$

The hasGrandparent relationship is when someone has a parent who has a parent.

$hasUncle \equiv hasFather \circ hasBrother$

The hasUncle relationship is when someone has a father who has a brother.

4) John is a father

Three of John's children are parents

John has five children

At most four of his children can be parents.

At least two of his children can be parents.

John's full name is John Brown

John has a wife named Mary

John is 51 years old

John is not Bill

5) MyBirthdayGuests was initially just defined as {Bill, John, Mary}, describing birthday guests but without any direct connections to the individuals. After the reasoner had executed they were all assigned as a birthday guest.

The hasRelative property had no relations in the beginning but after running the reasoner all reflexive properties holds true for every individual. This resulted in John and Mary being relatives to John Brown and Mary Brown respectively which then also lead to JohnBrown and MaryBrown being invited as birthday guests.

5.

1) Given that the requested ontology should consider data from different European universities and their educational resources, we can derive the following knowledge from each university:
2) - name,
- location (country),
- faculties,
- programs,
- courses,
- number of students,
- academic positions,
(and much more).

3) Not sure how to formulate CQs. Should the knowledge base only be partially specified?

requirements on the ontology. *Note*: For this lab, it is reasonable to prepare around 10 CQs. Of course, in real-world scenario, you will need have more CQs to specify the requirements on the ontology.

4) Work as an ontologist and develop the ontology:

- Follow the steps during the conceptualization: enumerate terms, define classes, defined class hierarchy, define properties (both object properties and datatype properties), define axioms and add a few instances.
- You should keep it in mind that the ontology is built using OWL 2 DL. Be careful about what can be or cannot be expressed in OWL 2 DL.
- As always in a typical modeling, there is no one correct solution.
- Create the ontology in OWL using Protégé:
  - The IRI of the ontology does not matter, use any "example" IRI you can think of or the one suggested by the tool.
  - Reasoning over your ontology to check its satisfiability and consistency.
  - Choose any syntax when you save the ontology.
  - You do not need to comment your ontology or the elements inside it, although that would normally be a good ontology development practice.