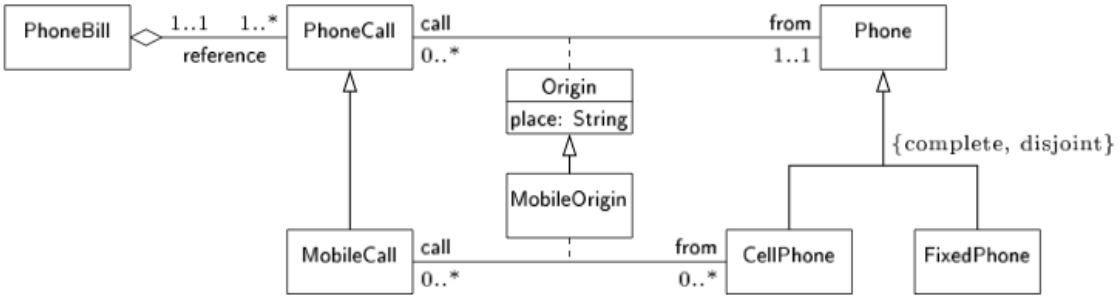
Knowledge Representation and Reasoning

Modelling and Reasoning with UML Class diagrams

1 Converting from UML to Description Logics

Consider the following UML class diagram about different kinds of phones, and phone bills they belong to.

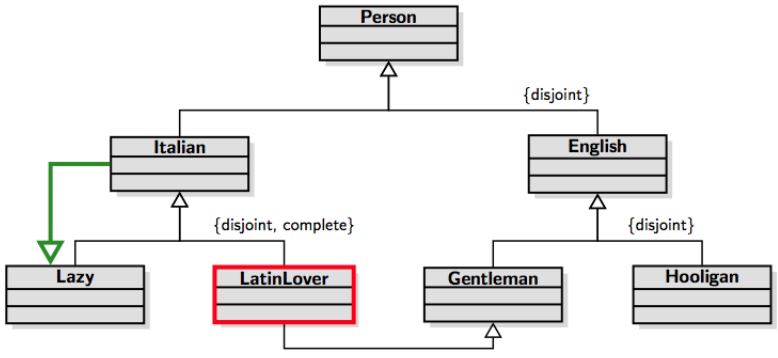


The diagram shows that a MobileCall is a particular kind of PhoneCall and that the Origin of each PhoneCall is one and only one Phone. Additionally, a Phone can be only of two different kinds: a Fixed Phone or a Cell Phone. Mobile calls originate (through the association MobileOrigin) from cell phones. The association MobileOrigin is contained in the binary association Origin: hence MobileOrigin inherits the attribute place of association class Origin. Finally, a PhoneCall is referenced in one and only one PhoneBill, whereas a PhoneBill contains at least one PhoneCall.

1. Convert the UML Diagram into Description Logics.
2. Suppose you add a generalization to the diagram asserting that each CellPhone is a FixedPhone. Which classes become inconsistent (i.e. they cannot be populated) and which pairs of classes become equivalent?

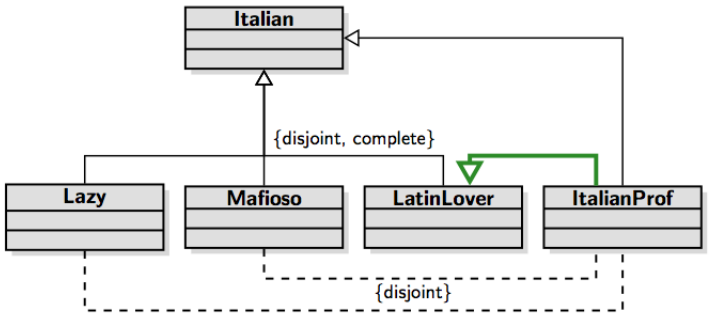
2 Basic Use of Protégé

Consider the Latin Lover Ontology presented in the following figure. Model the ontology with the Protégé tool and, by using the reasoner, try to figure out if there are latin lovers, and whether all Italians are lazy.



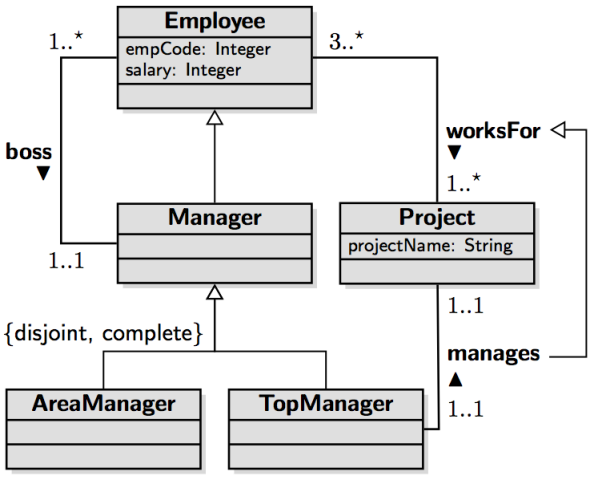
You should follow the instructions (aligned with what you did in the tutorial):

1. Instalation and Configuration
   (a) Install the Protégé tool (can be downloaded at `http://protege.stanford.edu/`)
   (b) Start Protégé and update the reasoners FaCT++ and Pellet by clicking `File` → `Check for Plugins`
   (c) For a better visual representation of ontologies you may download the GraphViz library and configure it appropriately
   (d) Restart Protégé
2. Create the Latin Lover Ontology
   (a) Open Protégé and and name your ontology (latinlover)
   (b) Create the classes in the Entities or Classes tab, starting with Person (you may use the `Add Class` or `Create Sibling` buttons, depending on the cases.
   (c) Add disjoint axioms with the `Disjoint With` + button, by selecting the involved classes
   (d) Add the completeness axiom for the class Italian, by introducing an equivalence axiom or using the option `Add covering axiom` in the `Edit` menu.
3. Use the reasoning services
   (a) Select a reasoner (any will do)
   (b) Choose Start Reasoner
   (c) Check the result in the Entities tab, both in the class hierarchy and in the inferred hierarchy
   (d) The justifications for the inconsistency of some classes can be obtained by pressing the question mark `?` button.
4. Now, do by yourselves the following Mafioso variant of the example, and check the inferred results. (Note: Inferred results are highlighted and are not to be encoded in the ontology explicitly).



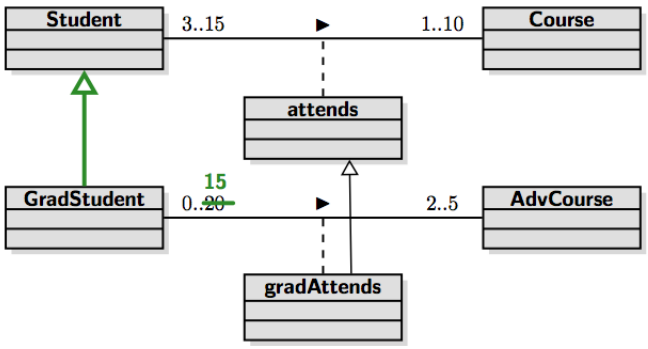
3 Translation of cardinality restrictions and property hierarchies

Encode completely the UML Class diagram below. Note that you need to use data properties to encode attributes.



4 Translating UML association classes into DL

Consider the students example in the next figure. The objective of this exercise is to devise the translation of UML Class diagrams into Description Logics for the case of association classes.



1. Translate the UML class diagram into Description Logic using the features of the Protégé tool. Note again that the major difficulty lies on the translation of the association classes. Use the reification technique studied in the lectures.
2. Use a reasoner to check that `GradStudent` $\sqsubseteq$ `Student`.
3. Create a probe class to verify if it is possible to have an advanced course with more than 15 students enrolled (depending on the probe class, you may experience problems using HermiT, if so, try, e.g., FaCT++ instead).
4. Introduce some individuals and check how the reasoner classifies them.