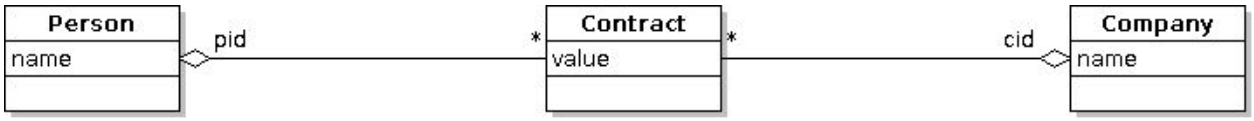
Patrick Dalisay
dalisay.pat@gmail.com
Michael Tirtowidjojo
mtirtowidjojo@gmail.com

P4

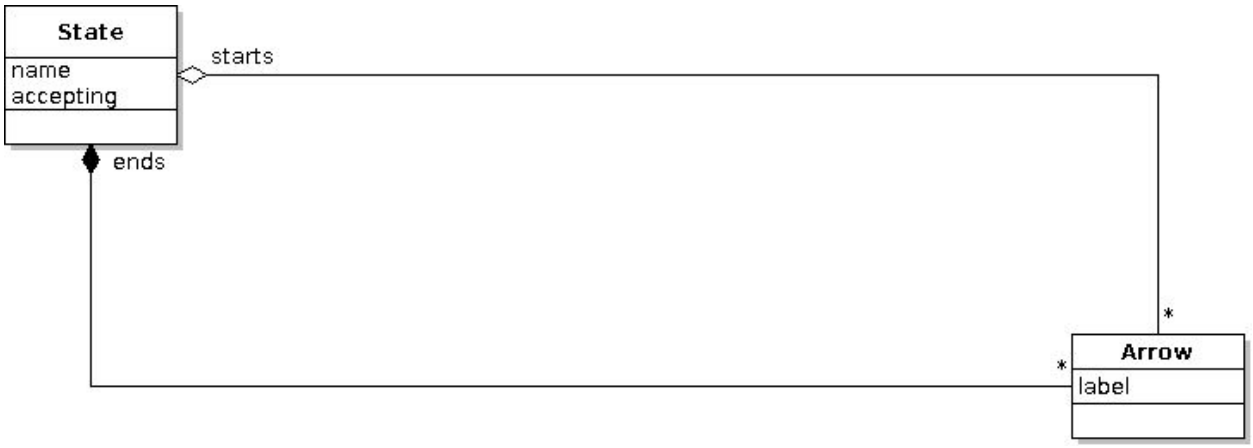
Our project successfully translates a Violet schema to an OO Schema. After reading in the input files, it loops through each tuple in vBox to first figure out if it's an abstract class or not. If it is, we turn in abstract. Then we simply split up the fields by the provided code which splits the string whenever it sees a percent sign. In order to avoid errors, we check to see if the field is empty before adding it to the OO schema. Next, we loop through the tuples again in order to add associations. We ended up joining the vBox and vAssoc tables so we can look at them at the same time. We then checked the arrow types of each association and decided from there what to do which each. If it was a black diamond, we would print out the role first and what it was pointing to. If it was a regular diamond, it would be the same but we add "option" to it, so show that it's an optional field (meaning the field is null). If it was a triangle, we would have to create a new subtable for inheritance. We created a SubTableSchema for each parent class and added the children to its fields. Because the way our code is written, we made a count function that would increment when creating parent subtables just so we don't create subtables for the children classes on accident.

We went ahead and added our own Violet schemas to convert to OO in order to test if it works on other Violet schemas other than the ones provided.



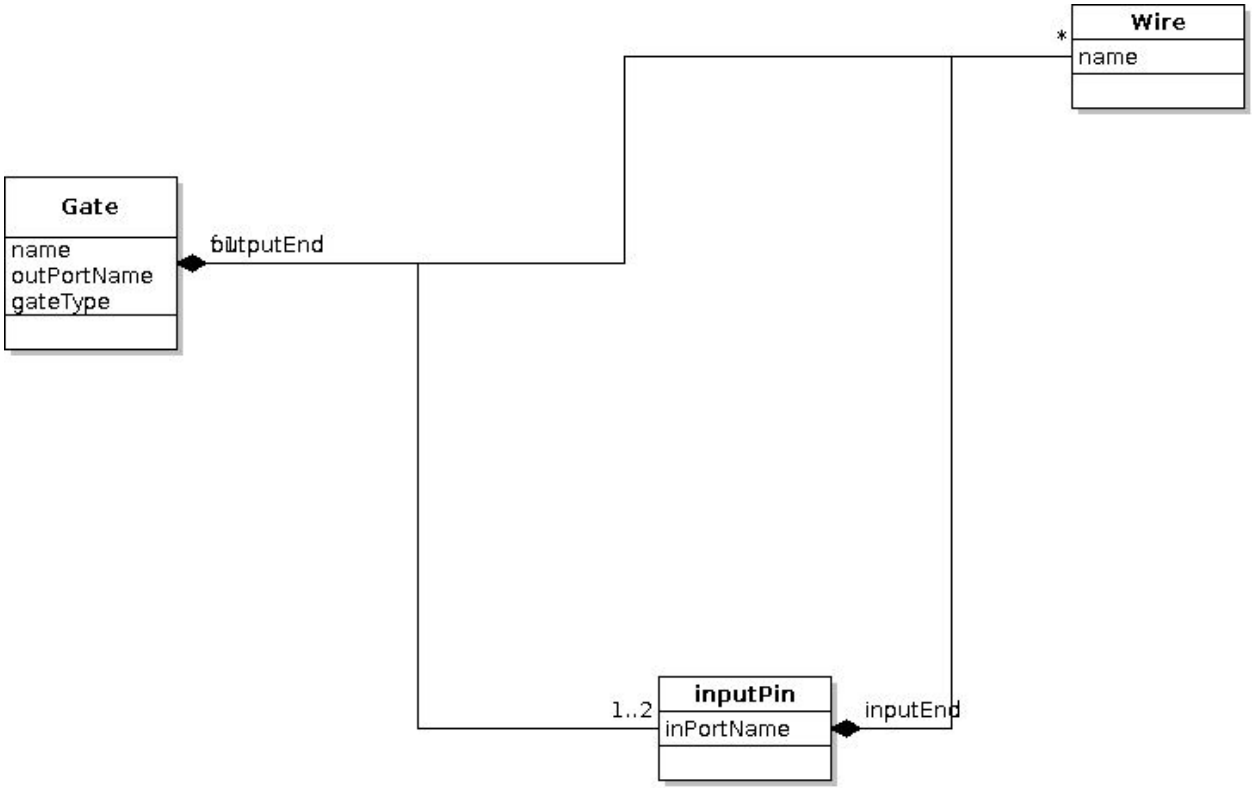
oodb(contract,[Person,Contract,Company]).

table(Person,[id,name]).
table(Contract,[id,value,pid:option Person,cid:option Company]).
table(Company,[id,name]).



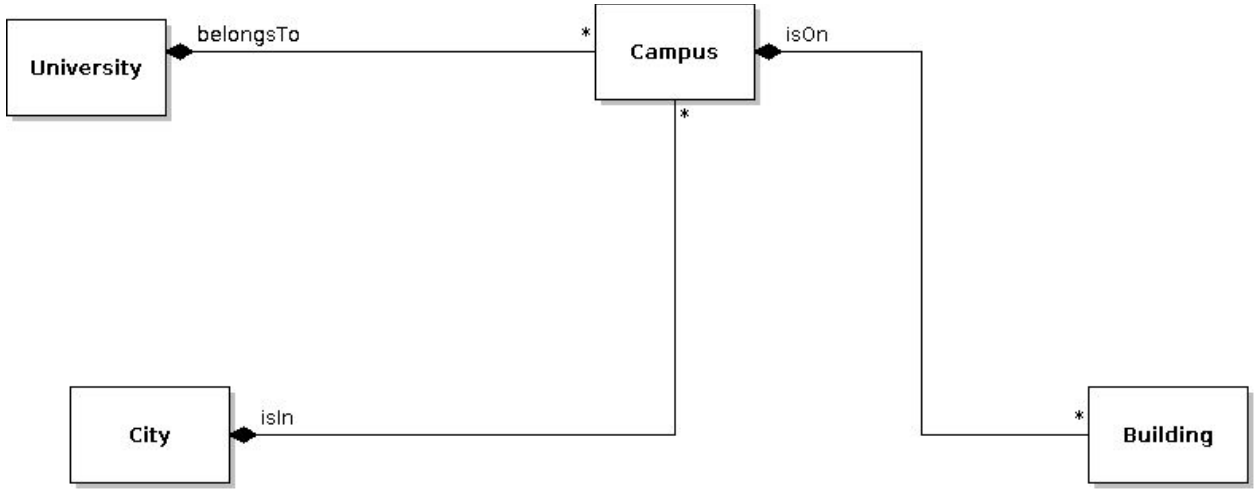
oodb(h1,[State,Arrow]).

table(State,[id,name,accepting]).
table(Arrow,[id,label,ends:State,starts:option State]).



oodb(Logic,[Gate,inputPin,Wire]).

table(Gate,[id,name,outPortName,gateType]).
table(inputPin,[id,inPortName,f1:Gate]).
table(Wire,[id,name,outputEnd:Gate,inputEnd:inputPin]).



oodb(schools,[University,Campus,City,Building]).

table(University,[id]).
table(Campus,[id,belongsTo:University,isIn:City]).
table(City,[id]).
table(Building,[id,isOn:Campus]).