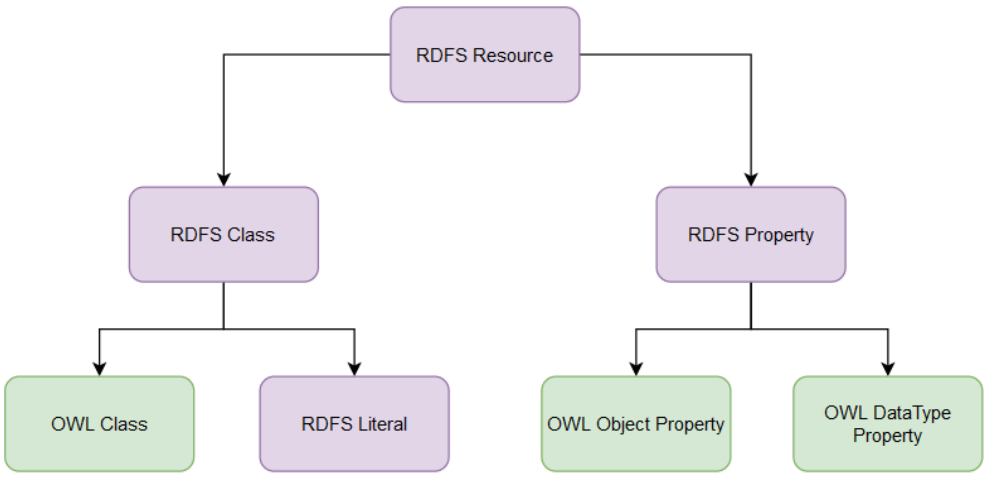
Домашно бр. 3
OWL

А) Домашна задача

1. Што е онтологија, а што јазик за опис на онтологија?

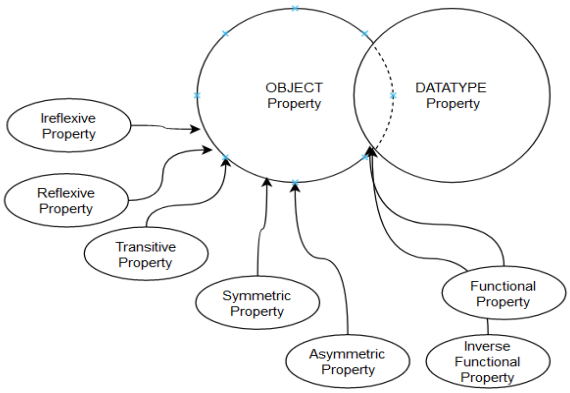
Онтологија е формална, експлицитна спецификација на споделен концептуализам.
Јазик за опис на онтологија е средство за да се искажат, дефинираат, манипулираат онтологиите.
Споредбено со релациони бази на податоци: ЕР дијаграм е шемата за опис на базата, но SQL е јазикот со кој пребаруваме, манипулираме, креираме податоци.

2. Нацртајте го односот / компатибилноста на OWL класи и својства со RDF/RDFS.



3. Наведете ги различните типови својства кои постојат во OWL.

Секое својство во ОВЛ мора да припаѓа во една од двете Property класи (ObjectProperty or DataTypeProperty).
Дополнително, својството може да припаѓа во следните класи:
Transitive, Reflexive, Irreflexive, Symmetric, Asymmetric, Functional, Inverse-Functional Property.



4. **Напишете примери за дефинирање класа и релација во OWL. Нека класата има суперкласа, а релацијата нека биде инверзна на друга релација. Користете Turtle синтакса.**

```
:Person    rdf:type            owl:Class .

:Student   rdf:type            owl:Class ;
           rdfs:subClassOf     :Person .

:Writes      rdf:type          owl:ObjectProperty .
:WrittenBy   rdf:type          owl:ObjectProperty ;
             owl:inverseOf     :Writes .
```

5. **Како се дефинираат ограничувања кај класите во OWL? Напишете еден пример за дефинирање ограничувања кај една OWL класа.**

   Постојат различни ограничувања кај класите во ОВЛ. Од генерализирачки до рестриктивни кои се креираат врз однос на постоечките својства на класата.
   За нивно дефинирање ги креираме како подкласи на рестрикциите кои се најавени како тип owl:Restriction.

   Пример за дефинирање ограничување на класата GoodStudent да припаѓаат само студентите чии оцени (сите!) припаѓаат во GoodGrades.

```
:GoodStudent    rdf:type            owl:Class ;
                rdfs:subClassOf     [ rdf:type owl:Restriction ;
                                      owl:onProperty :hasGrades ;
                                      owl:allValuesFrom :GoodGrades
                                    ].
```

Б) Практична задача

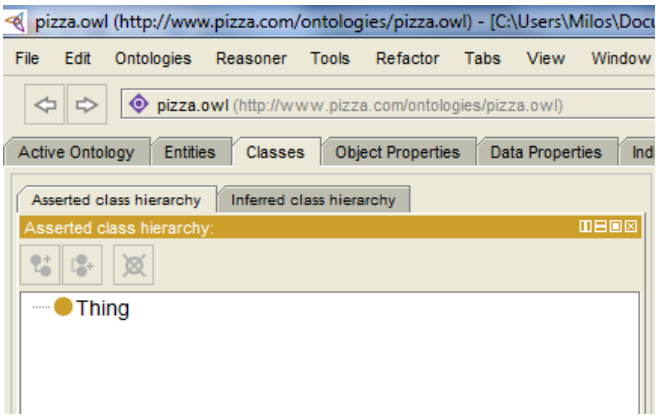
Вовед

Protégé е бесплатна, open-source платформа за креирање модели на домени и апликации со бази на знаење, базирани на онтологии. Protégé нуди поддршка за развој на секаков вид онтологии: колекции од хиерархиски поврзани термини, класификации, шеми на бази на податоци, итн.

1. Стартувајте го Protégé. Креирајте нова OWL онтологија:
   a. Ontology URI: http://www.pizza.com/ontologies/pizza.owl
   b. Локација: MyDocuments/WBS/pizza.owl

I. Дефинирање на класите во онтологијата

2. Отворете го табот Classes. Празното дрво на класи содржи само една класа, наречена owl:Thing, која е суперкласа на сите класи (Слика 1). Креирајте подкласи Pizza, PizzaTopping и PizzaBase. Сите се подкласи на owl:Thing.



Слика 1

3. Дефинирајте ги трите класи Pizza, PizzaTopping и PizzaBase како дисјунктни (disjoint). За оваа цел селектирајте една од класите и до десниот дел од интерфејсот изберете "Disjoint classes", со клик на знакот „+". За секоја од класите дефинирајте ги останатите две како дисјунктни со неа.

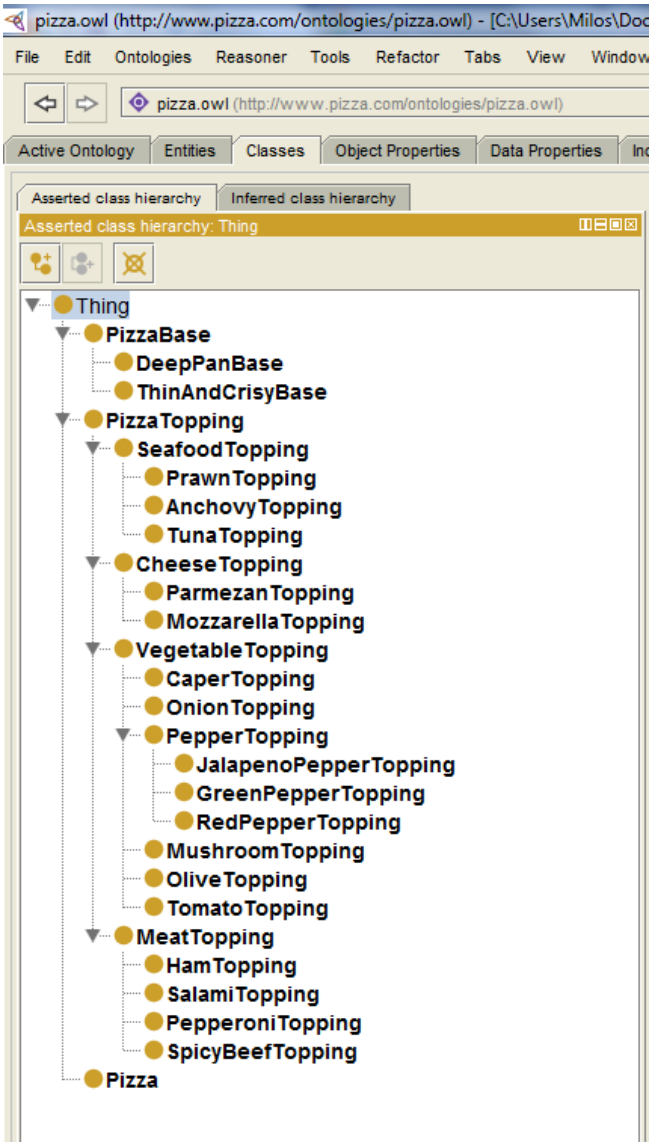
   Напомена: за побрзо дефинирање на дисјунктни класи (особено кога нивниот број е голем) направете multiple селекција на сите останати класи. На тој начин сите ќе станат дисјунктни.

4. Креирајте ги класите ThinAndCrisyBase и DeepPanBase како подкласи на PizzaBase. Дефинирајте ги овие две класи како дисјунктни.

5. Креирајте ги подкласите MeatTopping, VegetableTopping, CheeseTopping и SeafoodTopping како подкласи на PizzaTopping. Дефинирајте ги овие класи како дисјунктни.

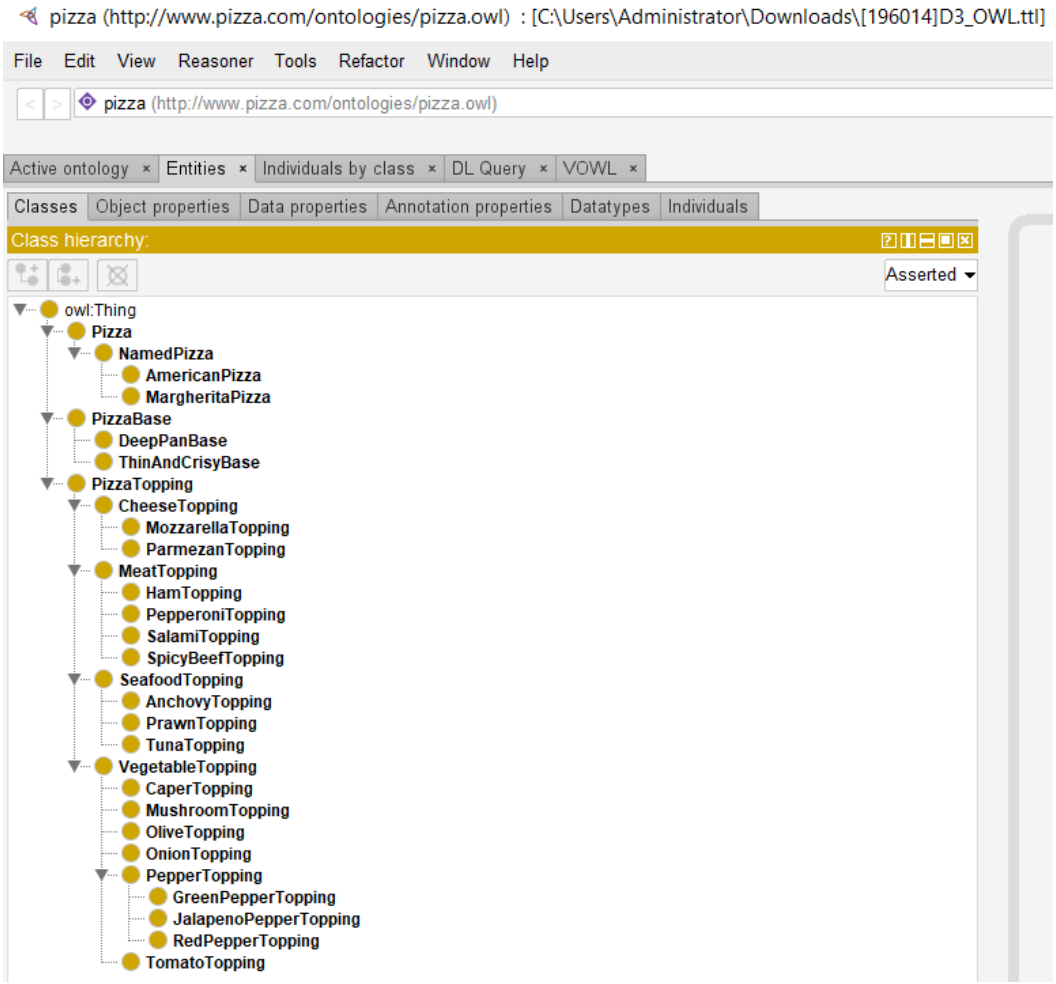
6. Во рамките на класата MeatTopping додадете ги следниве дисјунктни подкласи: SpicyBeefTopping, PepperoniTopping, SalamiTopping и HamTopping.

7. Во рамките на класата VegetableTopping додадете ги следниве дисјунктни подкласи: TomatoTopping, OliveTopping, MushroomTopping, PepperTopping, OnionTopping и CaperTopping.
8. Во рамките на класата PepperTopping додадете ги следниве дисјунктни подкласи: RedPepperTopping, GreenPepperTopping и JalapenoPepperTopping.
9. Во рамките на класата CheeseTopping додадете ги следниве дисјунктни подкласи: MozzarellaTopping и ParmezanTopping.
10. Во рамките на класата SeafoodTopping додадете ги следниве дисјунктни подкласи: TunaTopping, AnchovyTopping и .
11. После овие точки, вашата онтологија треба да изгледа како на Слика 2. Доколку имате грешки, вратете се назад на соодветниот чекор и отстранете ги.



Слика 2

Приказ изглед на онтологијата:



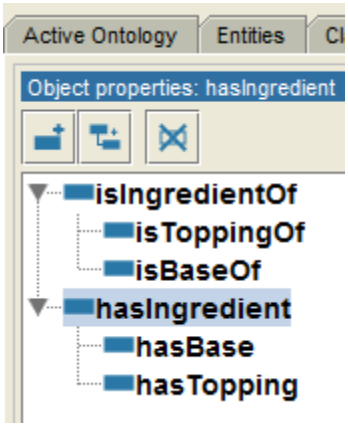
II. Дефинирање на релациите (својствата) во онтологијата

12. Префрлете се во табот Object Properties. Кликнете на копчето Add Property и додадете објектно својство со име hasIngredient.
13. Селектирајте го својството hasIngredient. Додадете му ги подсвојствата hasTopping и hasBase.
14. Креирајте ново објектно својство, isIngredientOf. Во десниот дел од интерфејсот додадете информација дека ова својство е инверзно на својството hasIngredient. За таа цел искористете го копчето „+“ до опцијата Inverse properties. Со ова дефинираме дека својствата hasIngredient и isIngredientOf се инверзни меѓу себе.
15. Креирајте ги својствата isBaseOf и isToppingOf како подсвојства на isIngredientOf. Дефинирајте ги како инверзни својства на hasBase и hasTopping, соодветно.
16. Селектирајте го објектното својство hasIngredient. Дефинирајте го како транзитивно својство, преку селектирање на check box-от даден во Characteristics (во средина на интерфејсот).
17. Дефинирајте го и својството isIngredientOf како транзитивно.
18. Дефинирајте го својството hasBase како функционално својство. Што значи одредено својство да биде функционално?
19. Специфицирајте ги доменот и опсегот на својството hasTopping. Доменот ги одредуваше класите кои може да ги имаат овие релации, а опсегот ги одредување класите кои можат да бидат вредности на овие релации.

    Кликнете на „+“ кај Domains (intersection) во делот Description (во десниот дел од интерфејсот). Во прозорецот кој ќе ви се појави, изберете го табот “Asserted class hierarchy”. Оттаму изберете ја класата Pizza како домен на својството hasTopping.

    Аналогно на овие постапки, дефинирајте ја класата PizzaTopping како опсег (range) на својството hasTopping.

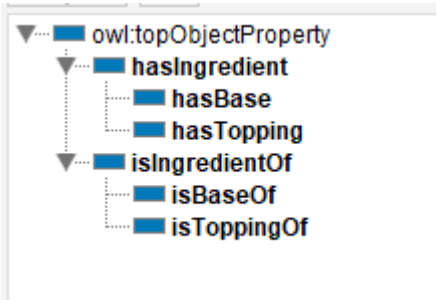
    Ова означува дека својството hasTopping се однесува на инстанци од класата Pizza (доменот), а како вредност може да има инстанци од класата PizzaTopping (опсегот).
20. Специфицирајте ги доменот и опсегот на својството isToppingOf. Поради тоа што својството е инверзно со hasTopping, доменот и опсегот на isToppingOf имаат обратни вредности од вредностите кај hasTopping својството. Тоа значи дека доменот на isToppingOf е PizzaTopping, додека пак опсегот е Pizza.
21. Специфицирајте ги доменот и опсегот на својството hasBase и на неговото инверзно својство isBaseOf. Домен на hasBase е Pizza, а опсег е PizzaBase. Кај isBaseOf доменот и опсегот се обратни, односно PizzaBase е доменот, а Pizza е опсегот.



Слика 3

22. Доколку правилно сте ги креирале релациите (својствата) во онтологијата, би требало да добиете изглед како на Слика 3.

Приказ изглед на релациите:



III. Дефинирање ограничувања во онтологијата

23. Дефинирањето на ограничувања за класите во една онтологија се прави со дефинирање на анонимни суперкласи. Инстанците од класата ќе мора да ги почитуваат овие ограничувања, за да бидат деца и на суперкласата.

    Вратете се назад на табот Classes и за класата Pizza дефинирајте ограничување дека нејзината релација hasBase треба да има барем една вредност (some) од PizzaBase. За да го направите ова, мора да додадете нова анонимна суперкласа за класата Pizza. Кликнете на „+“ веднаш до Superclasses. Од прозорецот одберете го табот “Object restriction creator”, во кој ќе дефинирате дека својството hasBase има тип на рестрикција Some (existential) кон класата PizzaBase. Кликнете OK.

    На овој начин, со помош на анонимна суперкласа дефинирвте ограничувања за класата Pizza.

24. Во класата Pizza дефинирајте подкласа NamedPizza. Во NamedPizza дефинирајте подкласа MargheritaPizza. Додајте го следниот коментар кај MargheritaPizza: „A pizza that only has Mozarella and Tomato toppings“. Додавањето на коментар се прави со својството comment, во делот Annotations +.

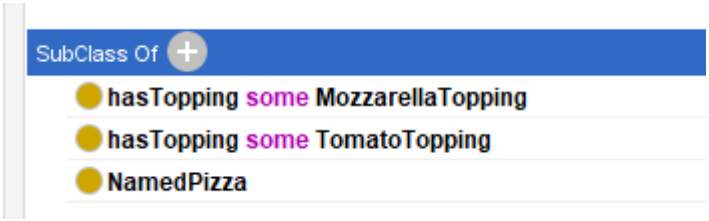
25. Каква вредност за „Inferred anonymous superclasses” има класата MargheritaPizza? Зошто?

26. На сличен начин како кај точка 22, дефинирајте ограничувања за MargheritaPizza дека својството hasTopping мора да има someValueFrom од MozzarellaTopping. Каков запис има во полето Superclasses сега за MargheritaPizza?

    Дефинирајте ограничување дека својството hasTopping мора да има и someValueFrom од TomatoTopping.

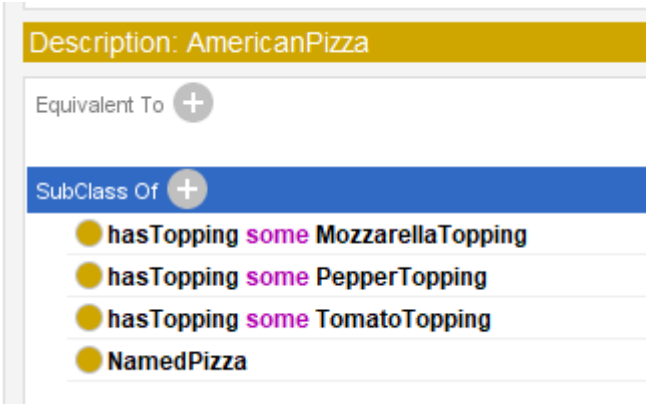
    SUPERCLASSES ⇔ SubClassOf – new versions of Protégé

    Полето SubClassOf за MargheritaPizza има записи:



27. Креирајте нова класа AmericanPizza, преку клонирање на класата MargheritaPizza. Тоа може да се направи со селектирање на MargheritaPizza и одбирање на опцијата "Edit – Duplicate selected class". Кај AmericanPizza дефинирајте уште едно дополнително ограничување: додадете PepperoniTopping.

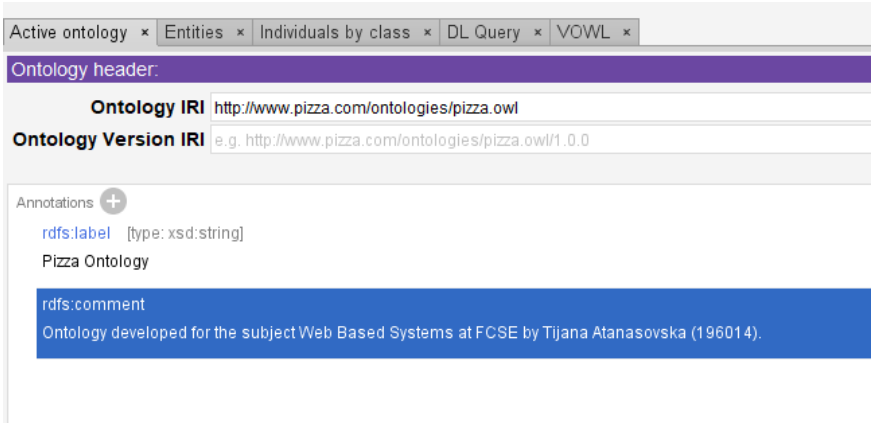
    Приказ на креираното:



28. Дефинирајте ги AmericanPizza и MargheritaPizza како дисјунктни класи.

IV. Дефинирање на детали за самата онтологија

29. Отворете го табот Active Ontology. Преку Annotations опцијата, додадете коментар (comment) со кој ќе ја опишете онтологијата и лабела (label) која ќе го означи името на самата онтологија (Pizza Ontology).



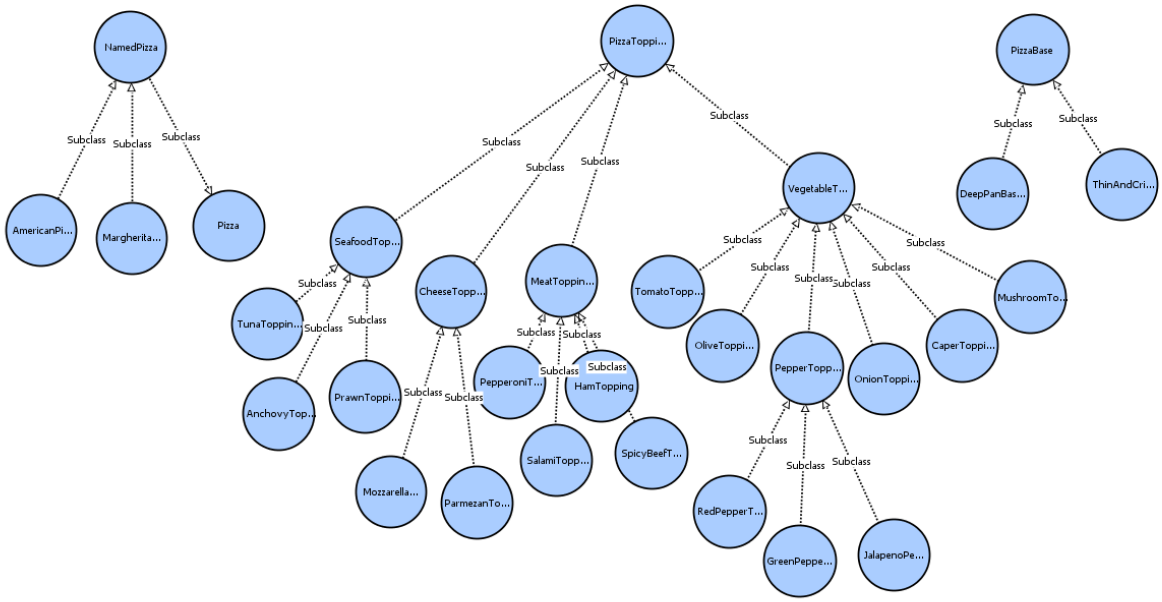
V. Визуелизација на онтологијата

ProtégéVOWL е додаток за Protégé кој овозможува визуелизација на онтологии, во форма на граф. За визуелизација, тој ја користи VOWL v2 визуелната нотацијата за исцртување OWL онтологии.

Преземете го ProtégéVOWL додатокот и инсталирајте го според упатството на неговата веб страна.

30. Отворете го табот VOWL. Пробајте да ги разместите класите од онтологијата така што нивната хиерархиска структура ќе биде јасно видлива.
31. Разгледајте ги класите и нивната поврзаност со релации. Дали структурата одговара на она што го очекувавте додека ја креиравте онтологијата?

    ДА!



VI. Онтологијата како Turtle датотека

32. По default, онтологијата е креирана со OWL/XML синтакса. Искористете ја опцијата за избор на формат на онтологијата при Save As... и снимете ја онтологијата и како Turtle датотека (Pizza.ttl).
33. Отворете ја Turtle датотеката во текстуален едитор и разгледајте ги креираните тројки со кои е дефинирана онтологијата од вежбата.

ПРиказ на дел од датотеката:

*[196014]D3_OWL - Notepad

File Edit Format View Help

```
@prefix : <http://www.pizza.com/ontologies/pizza.owl#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xml: <http://www.w3.org/XML/1998/namespace> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@base <http://www.pizza.com/ontologies/pizza.owl> .

<http://www.pizza.com/ontologies/pizza.owl> rdf:type owl:Ontology ;
                                            rdfs:comment "Ontology developed for the subject
                                               Web Based Systems at FCSE by Tijana Atanasovska (196014)." ;
                                            rdfs:label "Pizza Ontology"^^xsd:string .

#################################################################
#    Object Properties
#################################################################

###  http://www.pizza.com/ontologies/pizza.owl#hasBase
:hasBase rdf:type owl:ObjectProperty ;
         rdfs:subPropertyOf :hasIngredient ;
         owl:inverseOf :isBaseOf ;
         rdf:type owl:FunctionalProperty ;
         rdfs:domain [ rdf:type owl:Restriction ;
                       owl:onProperty :hasBase ;
                       owl:someValuesFrom :Pizza
                     ] ;
         rdfs:range [ rdf:type owl:Restriction ;
                      owl:onProperty :hasBase ;
                      owl:someValuesFrom :PizzaBase
                    ] .


###  http://www.pizza.com/ontologies/pizza.owl#hasIngredient
:hasIngredient rdf:type owl:ObjectProperty ;
               owl:inverseOf :isIngredientOf ;
               rdf:type owl:TransitiveProperty .


###  http://www.pizza.com/ontologies/pizza.owl#hasTopping
:hasTopping rdf:type owl:ObjectProperty ;
            rdfs:subPropertyOf :hasIngredient ;
            owl:inverseOf :isToppingOf ;
```