

Въведение в онтологии и OWL



Онтологии
OWL
Типове
Конструкции
Примери

Използвани ресурси

- Илюстрации и коментари: Vagan Terziyan
- W3C документи
 - Guide: <http://www.w3.org/TR/owl-guide/>
 - Reference: <http://www.w3.org/TR/owl-ref/>
 - Semantics and Abstract Syntax:
<http://www.w3.org/TR/owl-semantics/>
- OWL справочници
 - Ian Horrocks, Sean Bechhofer:
<http://www.cs.man.ac.uk/~horrocks/Slides/Innsbruck-tutorial/>
 - Roger L. Costello, David B. Jacobs:
<http://www.xfront.com/owl/>
- Примерни онтологии:
<http://www.daml.org/ontologies/>

Семантичен Уеб

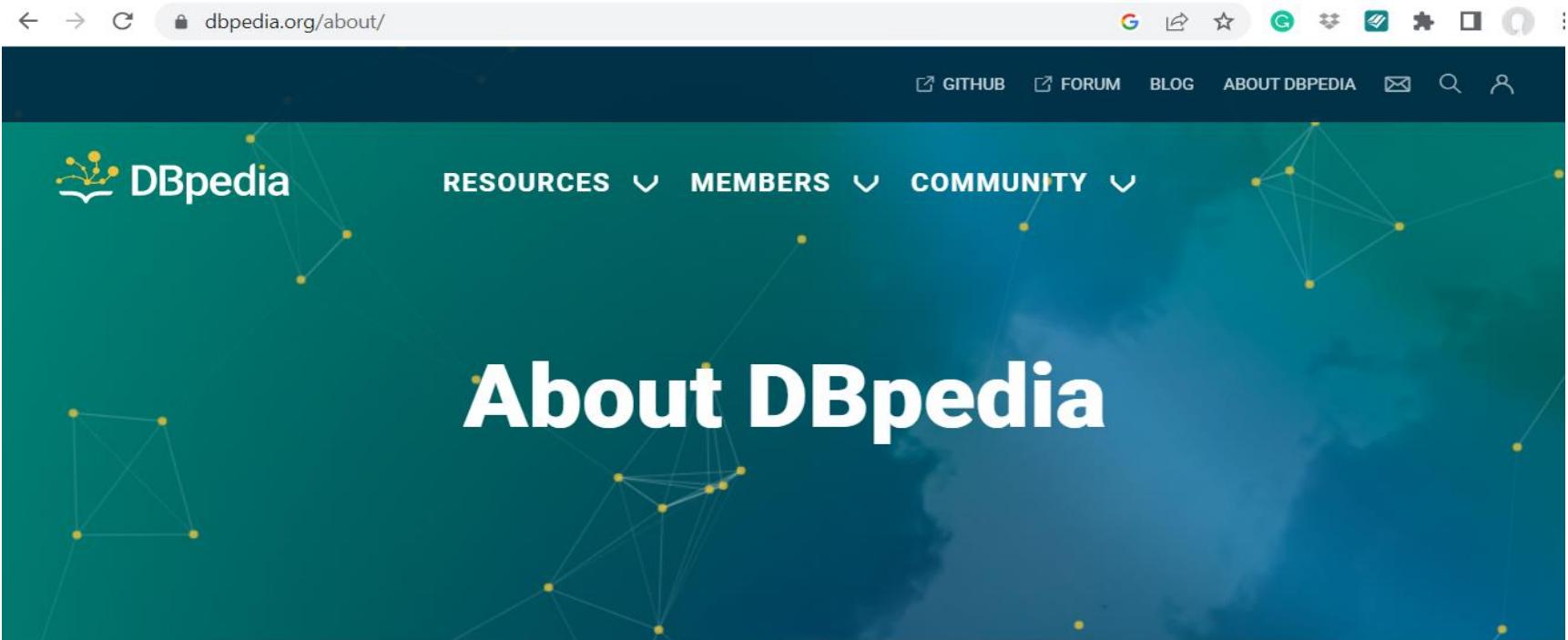


- През 1998 г. **Sir Tim Berners-Lee** предлага визията за семантичния уеб в статията “Semantic Web Road Map”
- <http://www.w3.org/DesignIssues/Semantic>
- Тази визия е доразвита в статия от 2001: Tim Berners-Lee et al., "The Semantic Web", Scientific American, May 2001

Дефиниция:

- "The Semantic Web is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation."
[Berners-Lee et al. 2001]

Linked Data

A screenshot of the DBpedia 'About' page. The page has a dark teal background featuring a network graph pattern with yellow nodes and white lines. At the top left is the DBpedia logo. At the top right are links to GitHub, Forum, Blog, and About DBpedia, along with standard browser icons. The main title 'About DBpedia' is centered in large, white, sans-serif font. Below the title is a detailed paragraph about DBpedia's mission and its role as a knowledge graph.

DBpedia is a crowd-sourced community effort to extract structured content from the information created in various Wikimedia projects. This structured information resembles an **open knowledge graph (OKG)** which is available for everyone on the Web. A knowledge graph is a special kind of database which stores knowledge in a machine-readable form and provides a means for information to be collected, organised, shared, searched and utilised. Google uses a similar approach to create those knowledge cards during search. We hope that this work will make it easier for the huge amount of information in Wikimedia projects to be used in some new interesting ways.

DBpedia data is served as Linked Data, which is revolutionizing the way applications interact with the Web. One can navigate this Web of facts with standard Web browsers, automated crawlers or pose complex queries with SQL-like query languages (e.g. SPARQL). Have you thought of asking the Web about all cities with low criminality, warm weather

Примерен ресурс:

<https://dbpedia.org/page/Plovdiv>

 DBpedia [Browse using](#) [Formats](#) [Faceted Browser](#) [Sparql Endpoint](#)

About: [Plovdiv](#)

An Entity of Type: [List of cities and towns in Bulgaria](#), from Named Graph: <http://dbpedia.org>, within Data Space: [dbpedia.org](#)

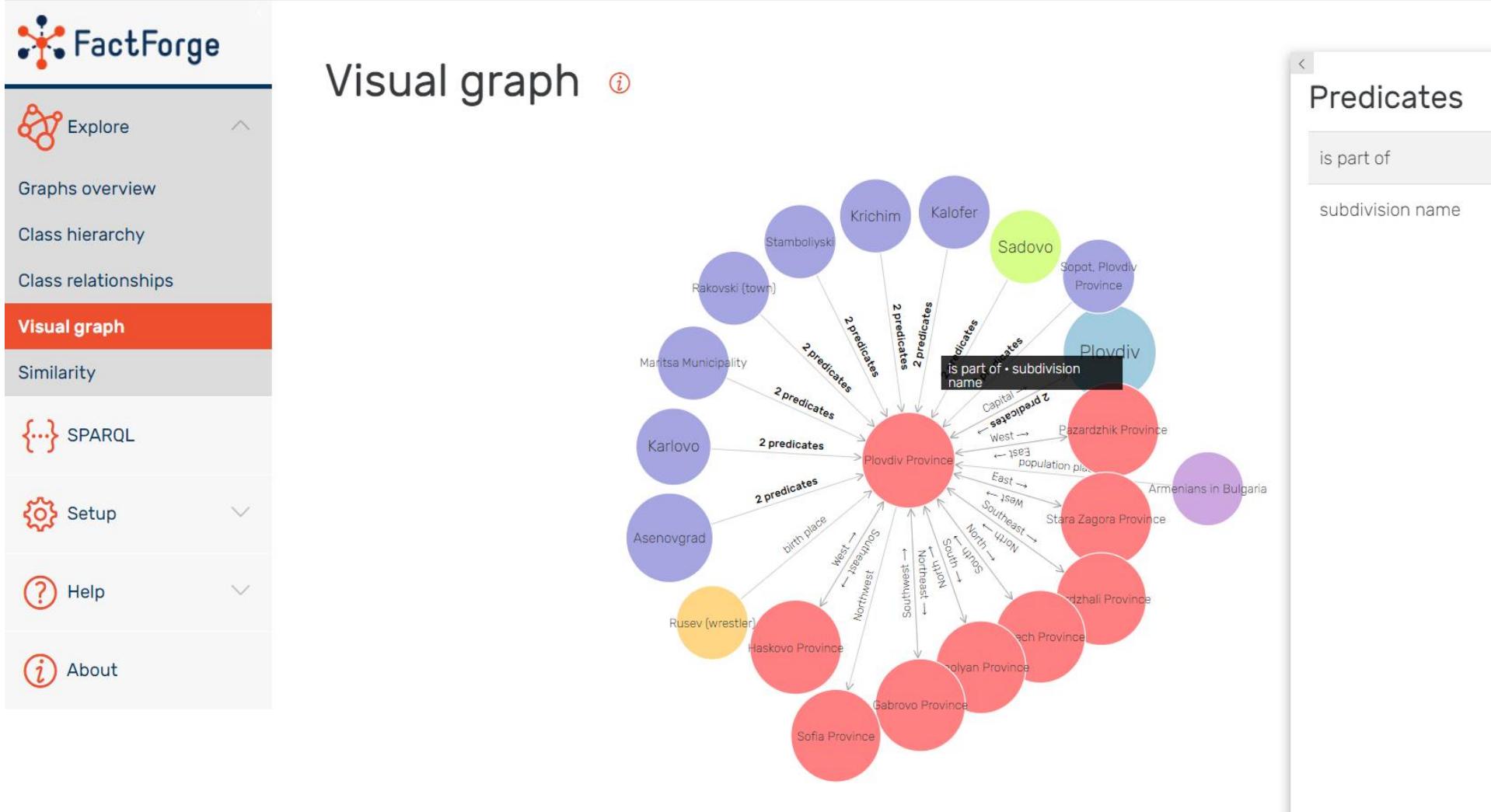
Plovdiv (Bulgarian: Пловдив, pronounced [ˈplovdif]), is the second-largest city in Bulgaria, standing on the banks of the Maritsa river in the historical region of Thrace. It has a population of 346,893 as of 2018 and 675,000 in the greater metropolitan area. Plovdiv is the cultural capital of Bulgaria and was the European Capital of Culture in 2019. It is an important economic, transport, cultural, and educational center. Plovdiv joined the UNESCO Global Network of Learning Cities in 2016.



Property	Value
dbo:PopulatedPlace/areaTotal	<ul style="list-style-type: none">• 101.98
dbo:abstract	<ul style="list-style-type: none">• Plovdiv (en búlgaro, Пловдив; en griego Philippopolis, "Φιλίππούπολις"; en turco, "Filibe") es una ciudad de Bulgaria y la capital de la provincia de Plovdiv. Con una población de 346 893 habitantes en 2018 y con un área metropolitana de 675 000 habitantes, es la segunda ciudad más poblada del país, después de la capital, Sofía. Está situada en las tierras bajas de Tracia, a la orilla del río Maritsa y de las siete colinas. Aunque existen restos arqueológicos pertenecientes al Neolítico, la ciudad fue fundada como Filípolis por Filipo II de Macedonia, padre de Alejandro Magno, en el 342 antes de Cristo. La ciudad cambió de manos entre el Reino de Macedonia y el Reino odrisio hasta la definitiva conquista por Filipo V de Macedonia. La República romana conquistó el reino macedonio en 183 a. C. y la convirtió en la capital de la provincia Tracia, siendo un gran nudo de comunicaciones gracias a la Vía Militaris y su acceso al río Danubio, el mar Negro y el mar Egeo. En el siglo III sufrió el saqueo de los godos de Cniva y en el siglo V la destrucción de los hunos de Atila. Fue anexionada al Primer imperio búlgaro en 863 durante el reinado el kan Boris I de Bulgaria. En 1219 se convirtió en la capital del Ducado de Filípolis, parte del Imperio latino de los cruzados, aunque el Segundo imperio búlgaro

Hubs for Open Data

◀ ▶ ⌂ ⚠ Not secure | factforge.net/graphs-visualizations?uri=http%3F%2Fdbpedia.org%2Fresource%2FPlovdiv_Province



Онтологична визия за Семантичния Уеб

Необходимост от онтологии

Онтологията е:

- документ, който официално и по стандартизиран начин определя юрархията на класовете в рамките на домейн, семантичните отношения между термовете (понятията) и правилата за извод

Използване на онтологии:

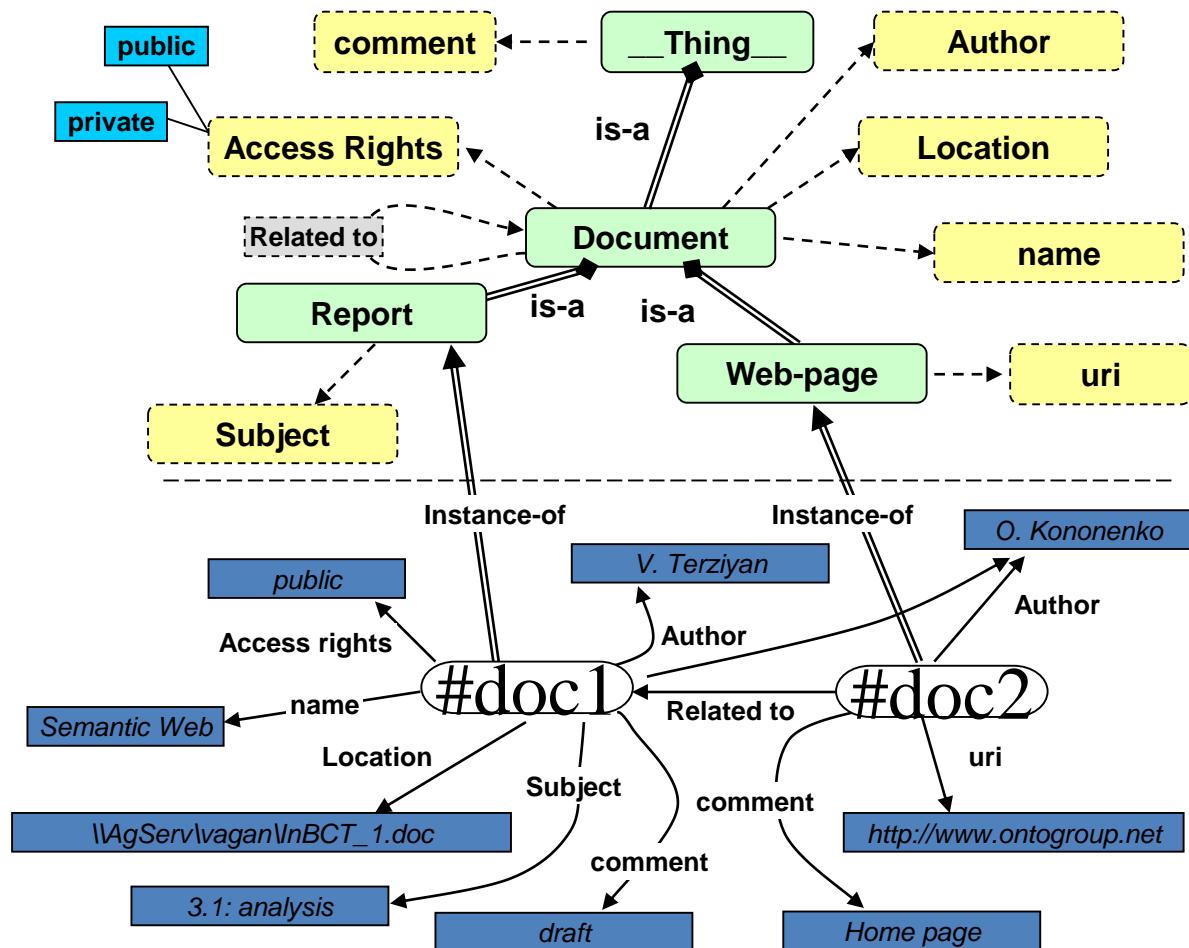
- споделянето на семантиката на данните между разпределени приложения

Онтологиите – основа на Семантичния уеб

Онтологиите като
ключова технология за
Семантичния уеб

**“... експлицитна
спецификация на
концептуализация ...”**

Онтология е формален и
богат начин за споделено
и общо предоставяне
разбиране на предметна
област, който може да се
използва от хора и
машини



Query 1: get all documents from location X, but not web-pages

Query 2: get documents related to Y, with more than one author, one of which is Terziyan

Query 3: are there web-pages of Z with “private” access related to documents with subject S?

Източник: [Vagan Terziyan](#)

Semantic Web: Interoperability



Придържането към една обща онтология е гаранция за съгласуваност и възможност за споделяне на данни и знания

Заявката днес

WWW Hotbot

What is Al Qaeda?

The screenshot shows a web browser window with the Lycos search engine interface. The search query "What is Al Qaeda?" has been entered into the address bar. The results page displays a list of 10 matches, each with a URL, a brief description, and a link to see results from the site only. The results include news articles about Osama bin Laden, the US government's efforts to protect embassies, and information about terrorist organizations like al-Qaida and the Taliban. A sidebar on the right contains advertisements for credit offers and a cartoon character.

Results for What is Al Qaeda?

People who did this search also searched for:

- Eilat
- Ashdod City In Israel
- Israel Military
- Israel Road Maps
- Reykjavik
- Keflavik
- Iceland Western European Country Military
- Iceland Western European Country Road Maps

WEB RESULTS Top 10 Matches [next >>](#)

- [1. Omaid Weekly -- Afghan Newspaper -- Latest News](http://www.omaid.com/english_section/latest_news.htm)
Afghan news updated daily from international news agencies, Afghan news sources and Omaid Weekly.
[See results from this site only.](http://www.omaid.com/english_section/latest_news.htm)
- [2. WHIPYU@Ossama](http://www.khri.win-uk.net/usa/ossama.htm)
U.S. charges bin Laden, aide with murder in Africa bombings State Department offers \$5 million reward The United States formally charged Saudi millionaire Osama bin Laden and a military aide with谋杀。
[See results from this site only.](http://www.khri.win-uk.net/usa/ossama.htm)
- [3. Series of Reports; USA Tries to Protect Embassies and Dismantle bin Laden's al-Qaida Terror Network - 17 Sept 98 to 24...](http://www.emergency.com/alqaida1.htm)
Excerpted from: ERRI DAILY INTELLIGENCE REPORT-ERRI Risk Assessment Services-Thursday, September 17, 1998-Vol. 4 - 260 ERRI MORNING NEWS
SUMMARY WASHINGTON (EmergencyNet News) - The Washington Post wa
[See results from this site only.](http://www.emergency.com/alqaida1.htm)
- [4. Ossama urged Muslims to kill Americans](http://terroristwatch.tripod.com/osamahist.html)
Terrorist Watch is an informational site dedicated to the acquisition and dissemination of factual information concerning the terrorist activities in the US by Ossama bin Laden, al Qa'ida, Iran, Iraq,
[See results from this site only.](http://terroristwatch.tripod.com/osamahist.html)
- [5. Update: Summary of Recent EmergencyNet News Reports on Osama Bin Laden and the Al-Qaeda Organization - 01 Jan 99 to 05...](http://www.emergency.com/1999/bnldn99a.htm)
Excerpted from: ERRI DAILY INTELLIGENCE REPORT-ERRI Risk Assessment Services-Tuesday, January 5, 1998-Vol. 5, No. 005 Hussein3.gif (12759 bytes) NEW YORK CITY
(EmergencyNet News) - Newsweek magazine
[See results from this site only.](http://www.emergency.com/1999/bnldn99a.htm)
- [6. al-Qa'ida \(The Base\) / Maktab al-Khidamat \(MAK - Services Office\) / International Islamic Front for Jihad Against the...](http://www.fas.org/irp/world/parastataladin.htm)
A profile of Terrorist Organizations and Other Para-States.
[See results from this site only.](http://www.fas.org/irp/world/parastataladin.htm)
- [7. TLAM](http://www.enrakidere.org.com/tarana.htm)
Analysis of Strike On the Morning of 20 August 1998, US Naval forces subordinate to Commander, US Fifth Fleet -steaming in the Arabian Gulf launched an attack on known terrorist training facilities (T
[See results from this site only.](http://www.enrakidere.org.com/tarana.htm)

The answer may be somewhere in this list of URLs

Семантична заявка

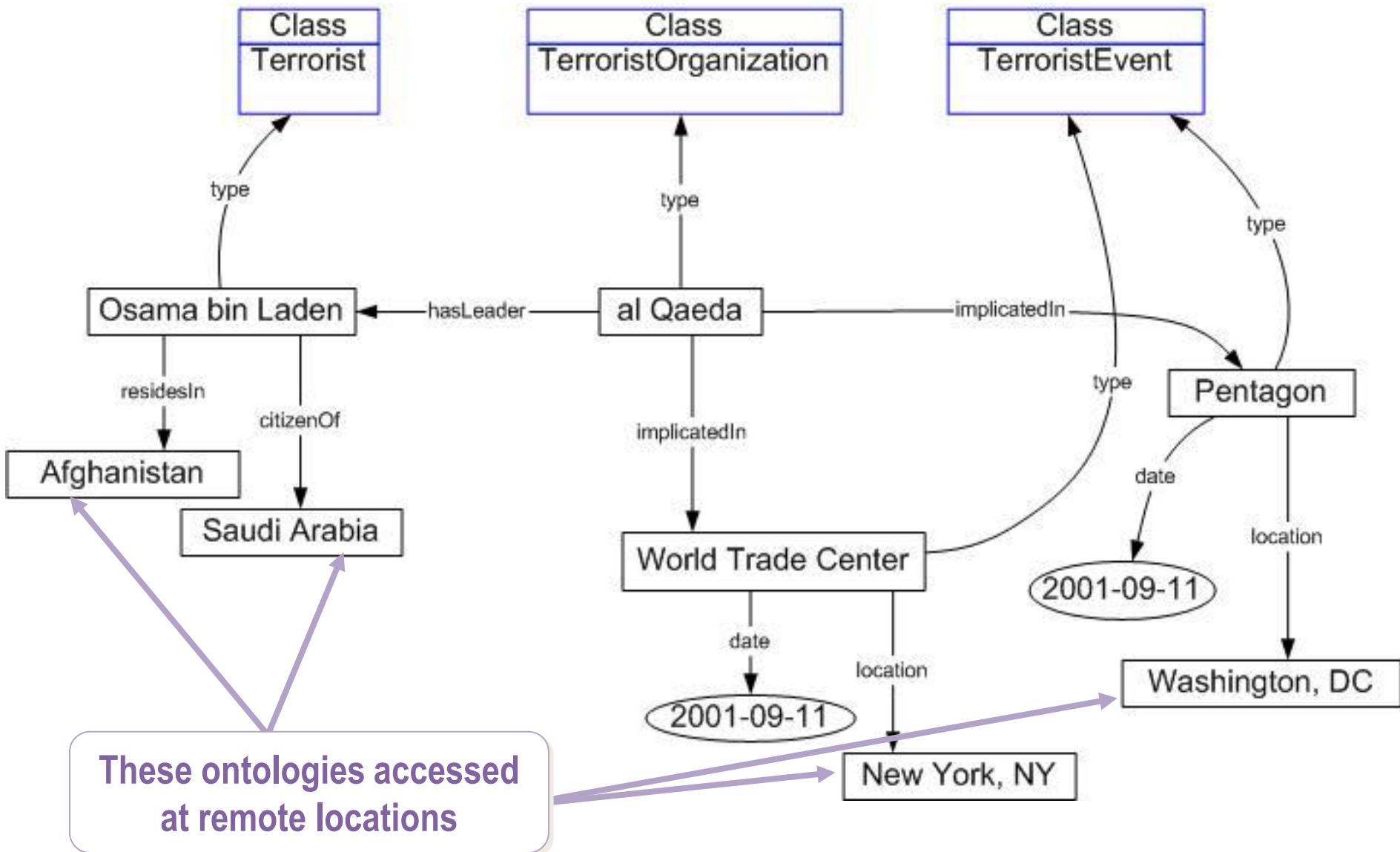
What is Al Qaeda?

A terrorist organization

Would you like additional information on?

- Membership**
- Locations**
- Structure**
- Finances**
- Tactics**
- Other terrorist organizations**

Примерна онтология

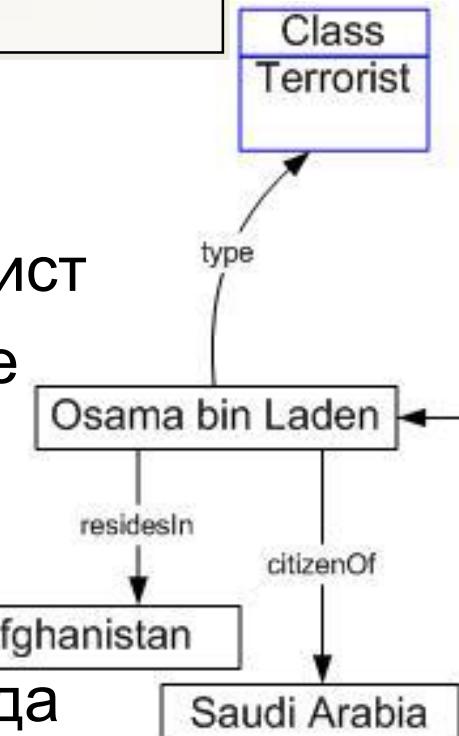


RDF-базирано заключение

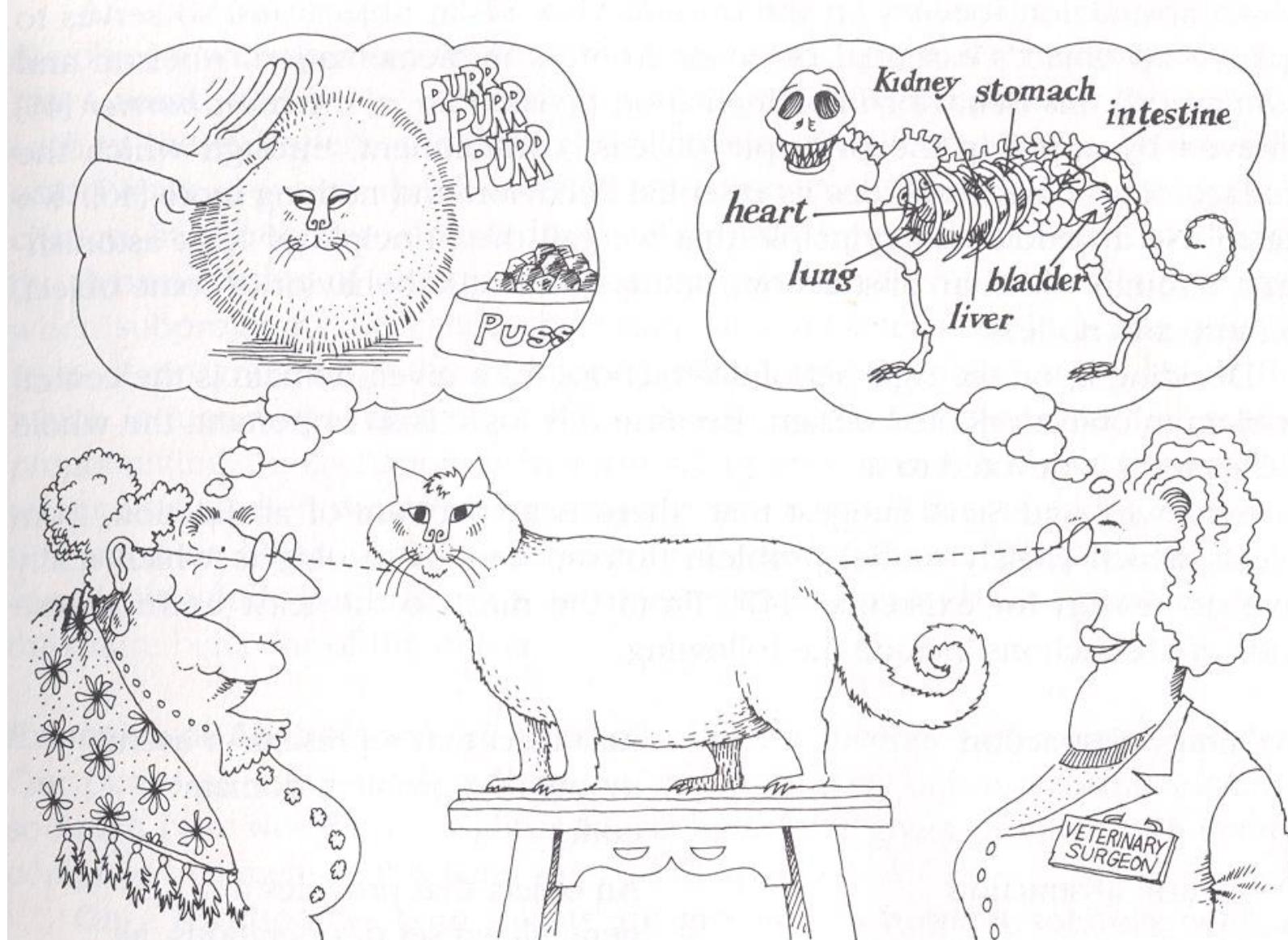
```
<daml:Class rdf:id="Bin Laden">  
  <rdfs:subClassOf rdf:resource="#terrorist"/>  
</daml:Class>
```

предполага

- Ако х е Бин Ладен, той трябва да е терорист
- Ако х е терорист, тогава той може да бъде или да не бъде Бин Ладен
- Ако х не е терорист, тогава той не е Бин Ладен
- Ако х не е Бин Ладен, той може да е или да не е терорист

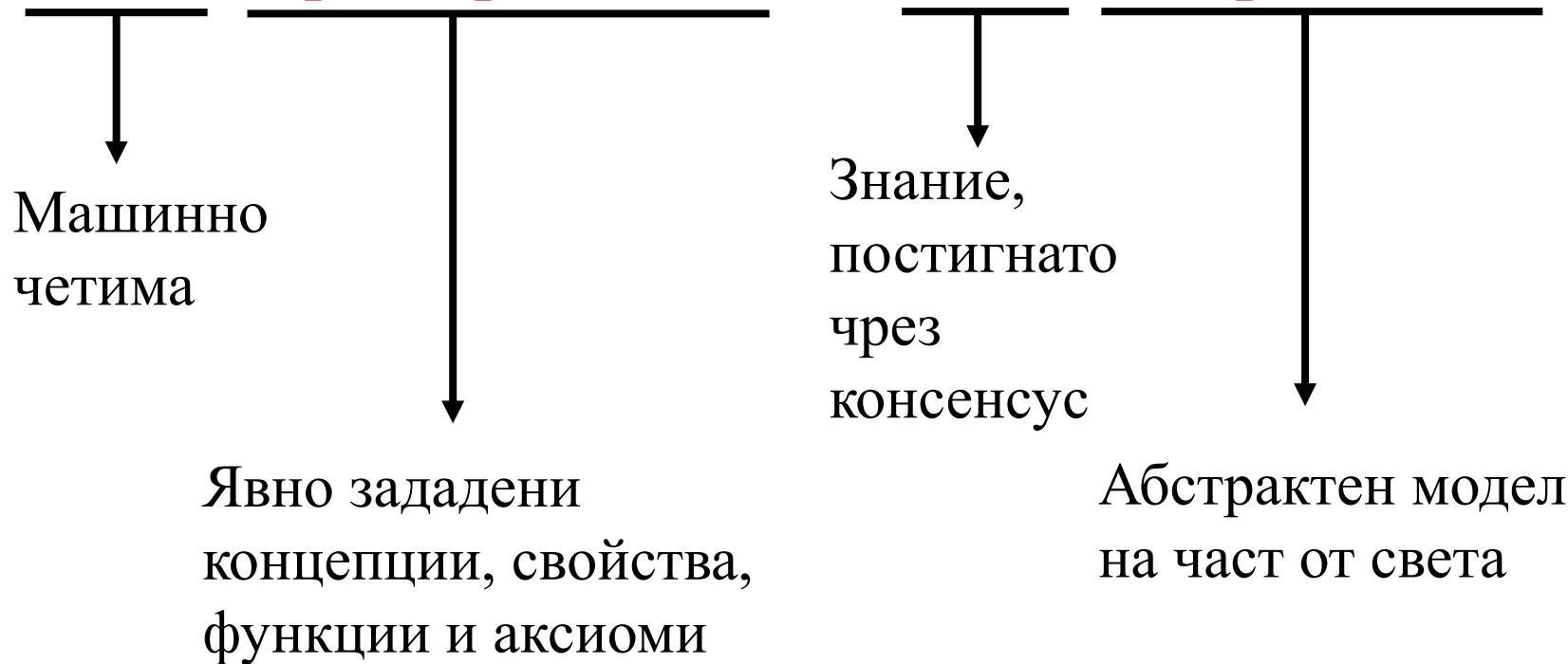


Комуникации между хора



Какво е онтология?

Formal, explicit specification of a shared conceptualization

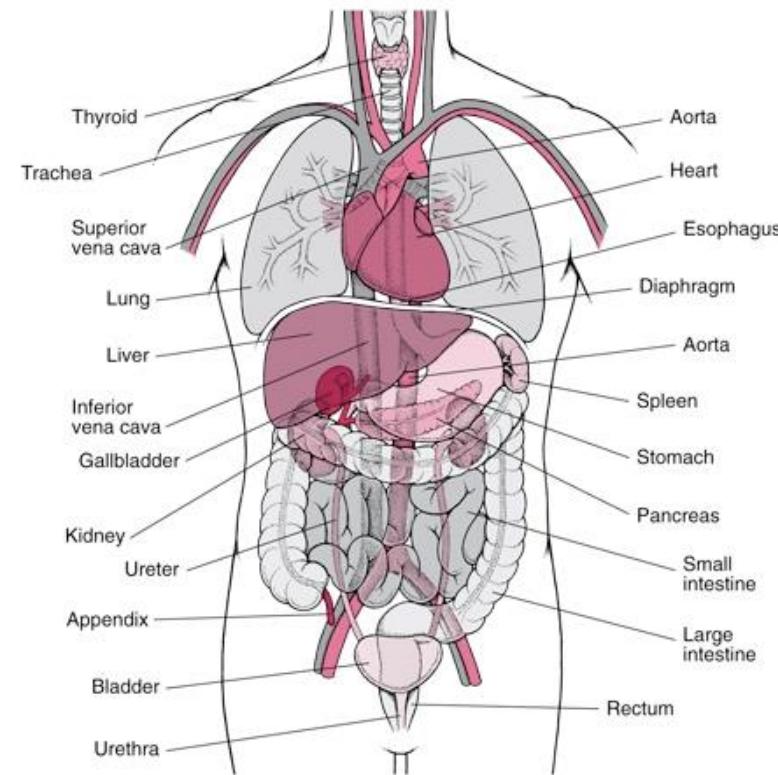


Tom Gruber, Knowledge Systems Laboratory at Stanford University

Какво е онтология?

Модел на (някои от) аспектите на част от света

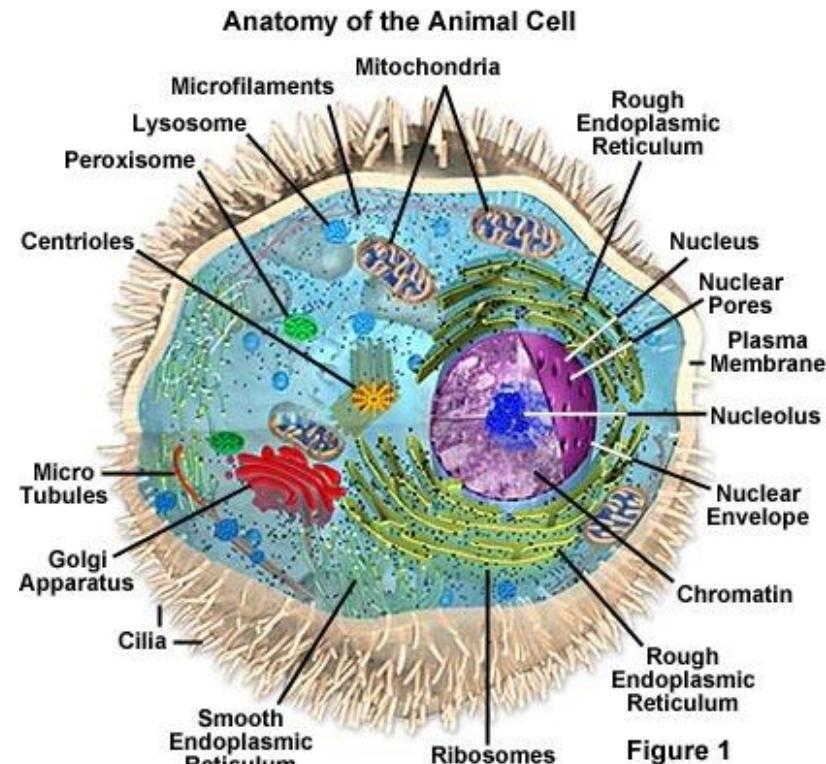
- Въвежда **речник** свързан с предметната област, напр.:
 - Анатомия



Какво е онтология?

Модел на (някои от) аспектите на част от света

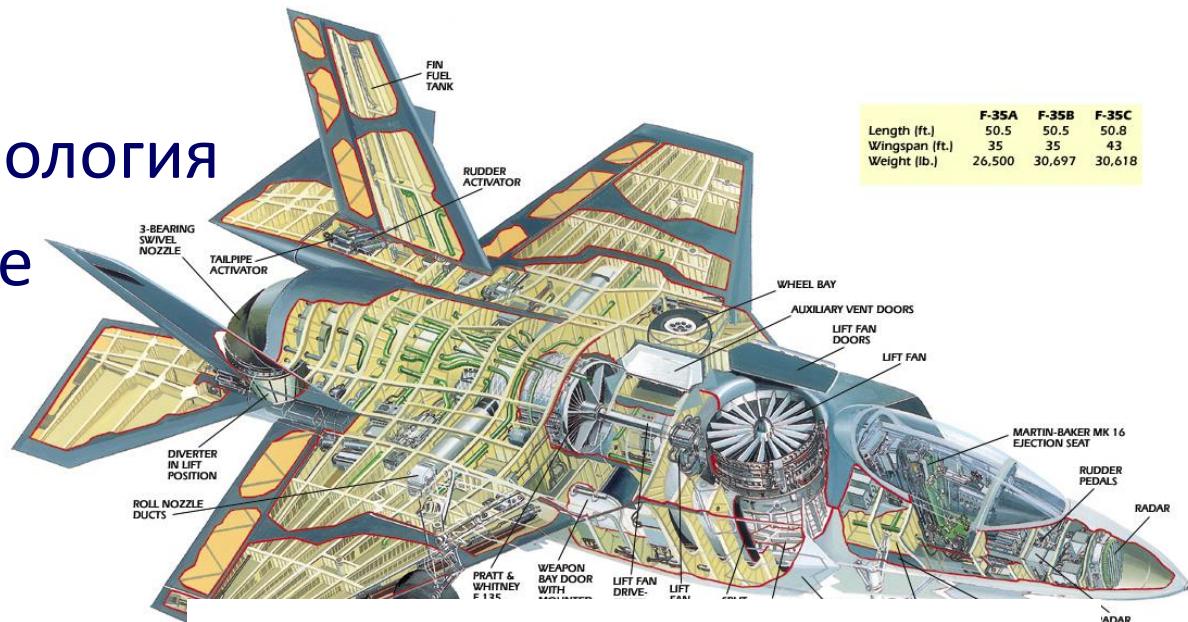
- Въвежда **речник** свързан с предметната област, напр.:
 - Анатомия
 - Молекулярна биология



Какво е онтология?

Модел на (някои от) аспектите на част от света

- Въвежда **речник** свързан с предметната област, напр.:
 - Анатомия
 - Молекулярна биология
 - Самолетостроене



Какво е онтология?

Модел на (някои от) аспектите на част от света

- Въвежда **речник** свързан с предметната област, напр.:
 - Анатомия
 - Молекулярна биология
 - Самолетостроене
 - Сандвичи

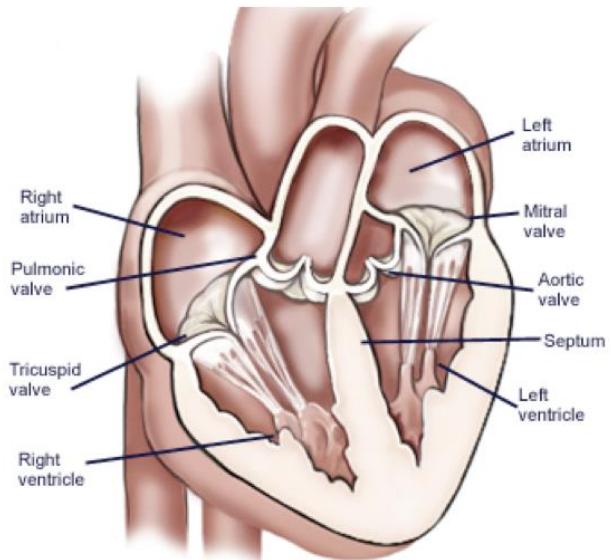


Какво е онтология?

Модел на (някои от) аспектите на част от света

- Въвежда **речник** свързан с предметната област
- Определя **значението** на термините:

**Heart is a muscular organ that
is part of the circulatory system**



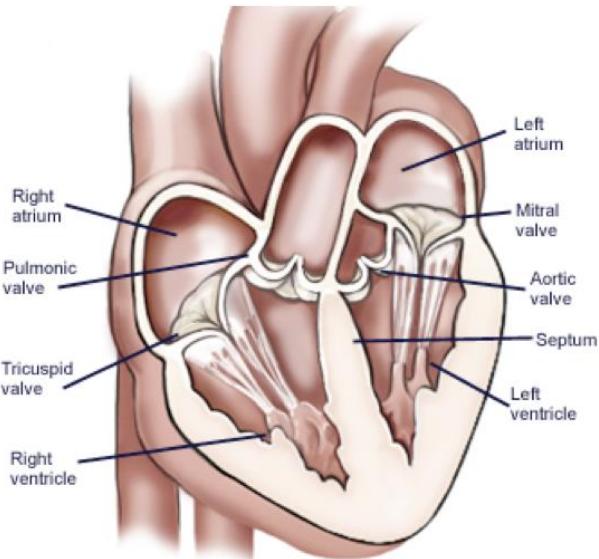
Какво е онтология?

Модел на (някои от) аспектите на част от света

- Въвежда **речник** свързан с предметната област
- Определя **значението** на термините:

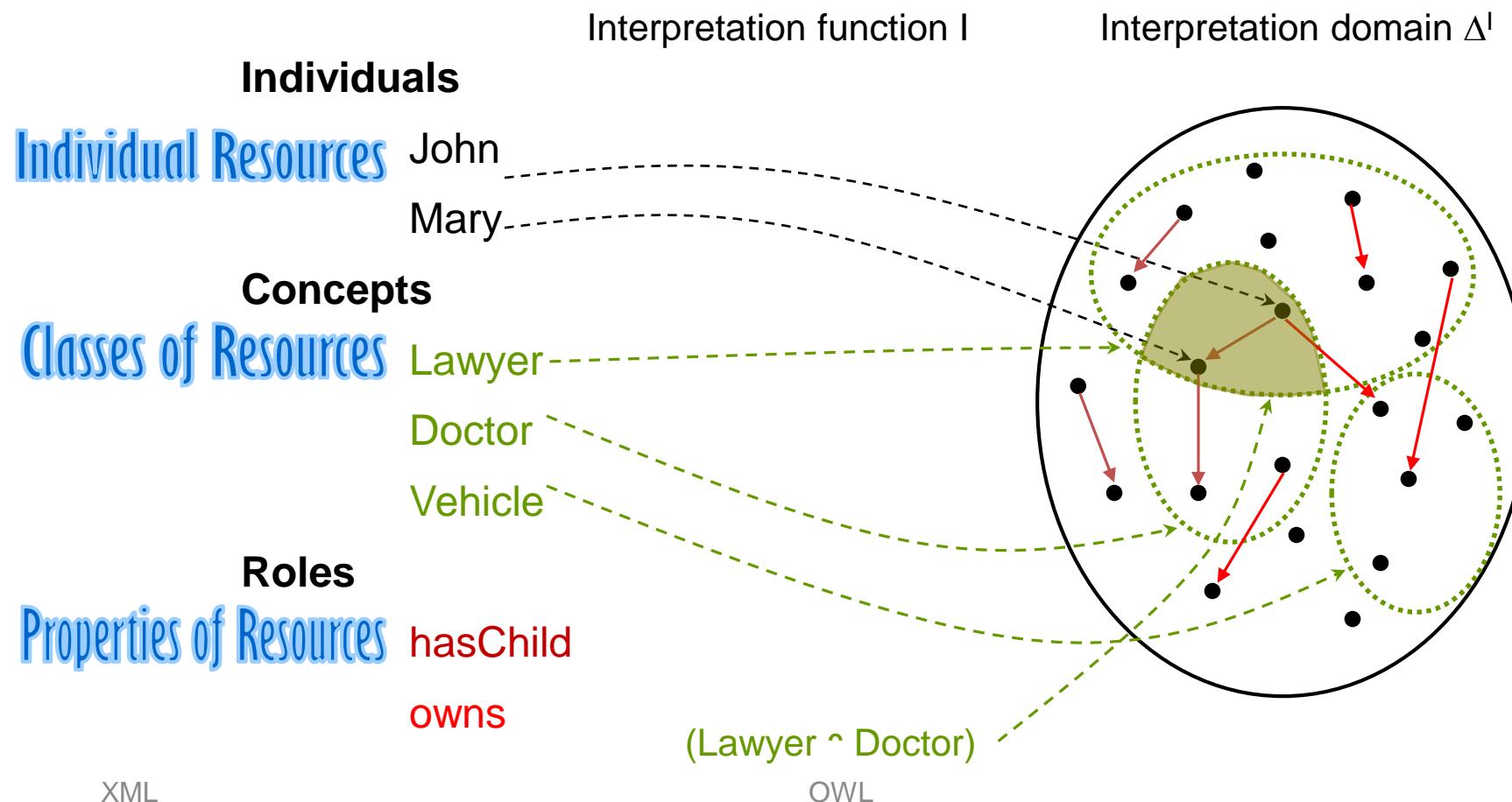
Heart **is a muscular organ that is part of the circulatory system**

- **Формализация** чрез логика

$$\begin{aligned} \forall x. [\text{Heart}(x) \rightarrow \text{MuscularOrgan}(x) \wedge \\ \exists y. [\text{isPartOf}(x, y) \wedge \\ \text{CirculatorySystem}(y)]] \end{aligned}$$


DL семантика

Задаване на семантика:



Предимства на използване на онтологии

- { Комуникация между хора
- { Оперативна съвместимост между софтуерни агенти

- { Повторна употреба на знания
- { Знанието за домейн става явно
- { Анализ на знанията за определен домейн

Building an ontology is not a goal in itself.

Knowledge sharing and reuse

Елементи на онтологиите

- Концепции (класове) + тяхната йерархия
- Свойства на класовете (слотове/атрибути)
- Ограничения над с-ва (type, cardinality, domain)
- Релации м/у концепции (disjoint, equality)
- Екземпляри (Instances)

Как изграждаме онтология?

Стъпки:

1. определяме домейн и обхват
2. изброяваме важните термини
3. дефинираме класове и клас-йерархии
4. определяме релации между класовете
5. дефинираме слотове (свойства)
6. дефинираме ограничения над с-ва
(кардиналност, тип на стойност, ...)
7. правила, аксиоми и събития

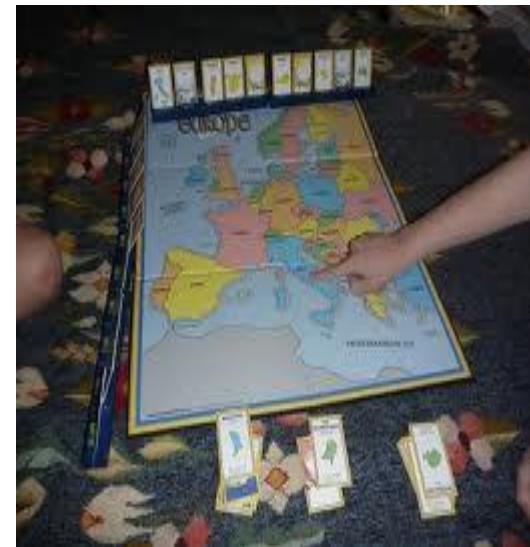
Стъпка 1: Определете областта и обхвата

Домейн: география

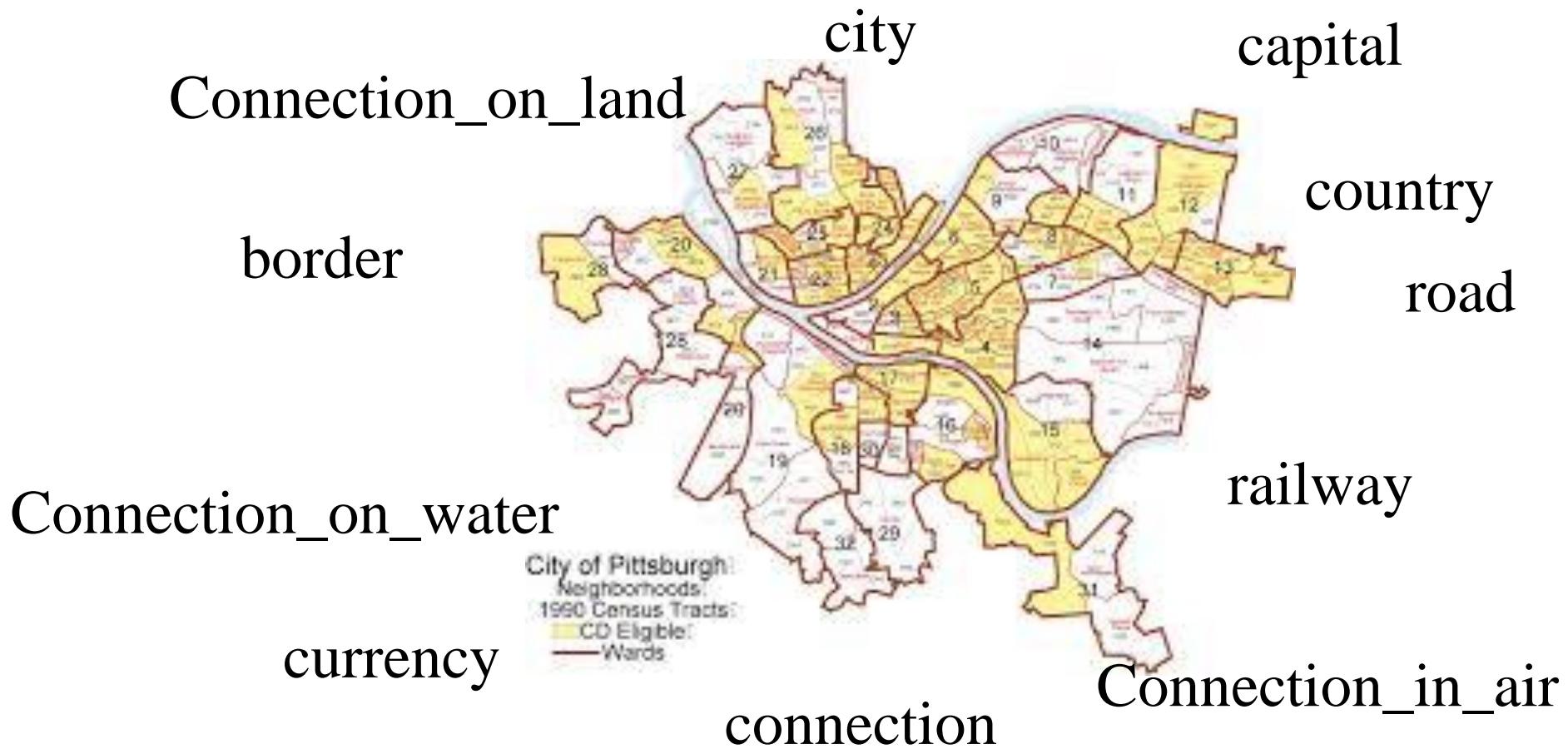
Приложение: агент-маршрутизатор

Възможни въпроси:

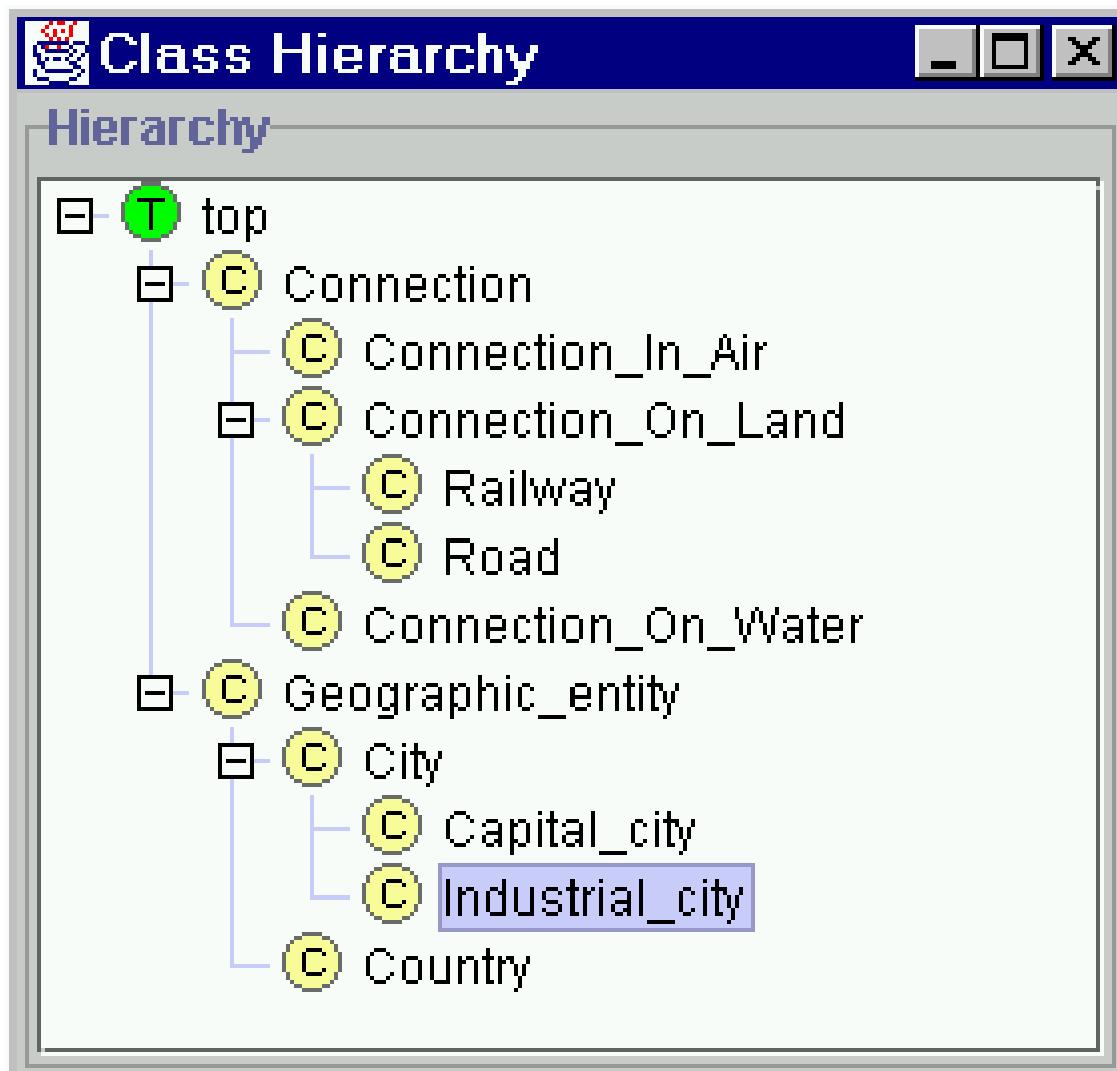
- Разстояние между два града?
- Какви са връзките между два града?
- В коя държава е даден град?
- Колко граници се пресичат?



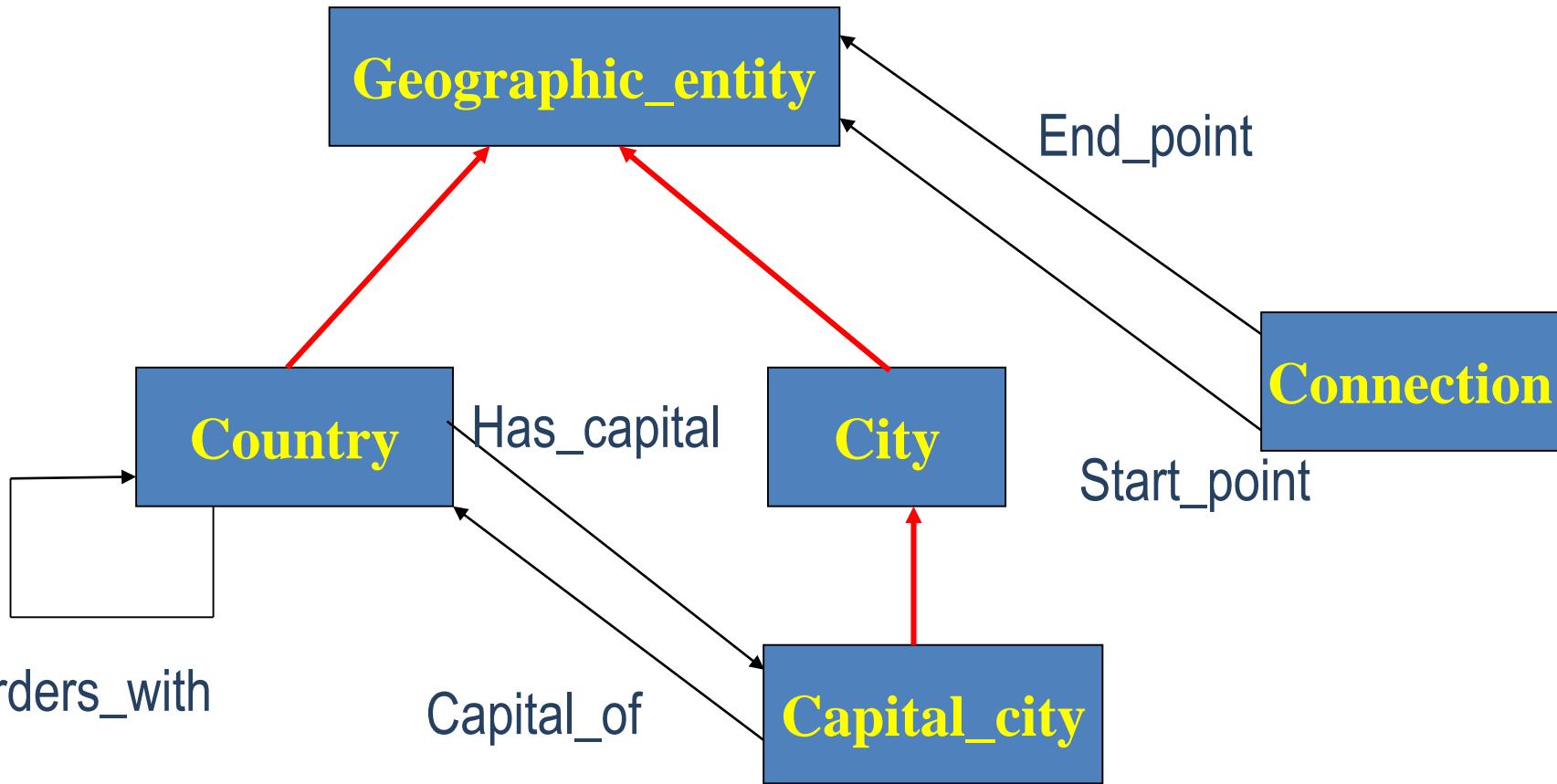
Стъпка 2: Избройте важните термини



Стъпка 3: Определяне на класове и йерархия



Стъпка 4: Определяне на релации между класовете



Стъпка 5/6: Слотове и ограничения

- Slot-cardinality

*Ex: Borders_with **multiple**, Start_point **single***

- Slot-value type

*Ex: Borders_with - **Country***

Стъпка 7: Правила, аксиоми и събития

Правила: съждения (statements) във формата на *if-then* (*antecedent-consequent*) изречения, които описват логическите изводи, които могат да се извлекат от едно твърдение в определена форма

Аксиоми: твърдения (включително правила) в логическа форма, които заедно съставляват цялостната теория, описвана от онтологията в своята област на приложение.

За разлика от генеративните граматики и формалната логика, където аксиомите включват само изявления, утвърдени като априорни знания, тук аксиомите също така включват различни деривации.

Събития: промяна на атрибутите или отношения

Web Ontology Language (OWL)

10 февруари 2004 г. World Wide Web Consortium съобщи окончателното одобрение на два ключови Semantic Web технологии, ревизираният Resource Description Framework (RDF) и **Web Ontology Language (OWL)**.

<http://www.w3.org/2004/01/sws-pressrelease.html.en>

Въведение в OWL

- **Какво е OWL?**

- OWL е език за дефиниране на уеб онтологии и свързаните с тях бази от знания (Knowledge Bases)
- OWL е преразглеждане на DAML+OIL (Web-based representation and inference layer for ontologies) езици за уеб онтологии, относно използването им в практиката



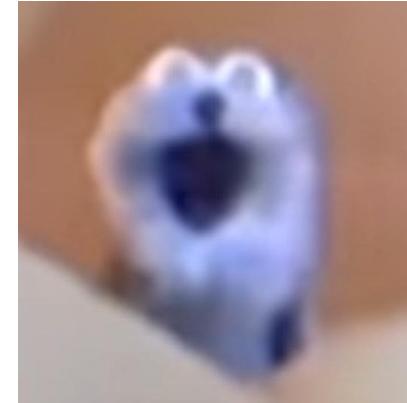
Инструмент за проектиране на онтологии – Protégé

- Protégé е редактор за онтологии и бази знания (<http://protege.stanford.edu>).
- Protégé е Java инструмент с отворен код, който предоставя разширяема архитектура за създаването на персонализирани приложения.
- Наличен е също така и онлайн на <http://webprotege.stanford.edu/>

OWL ни помага...

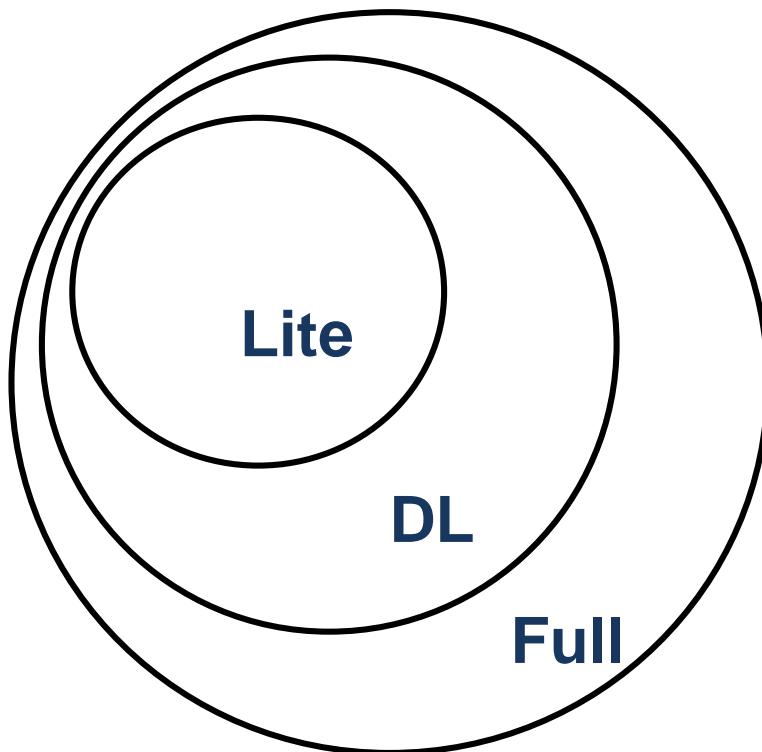
...да опишем нещо, вместо само да го именоваме.

Класът (BlueThing) няма смисъл сам по себе си



Но има смисъл класът (BlueThing complete
owl:Thing
restriction (hasColour someValuesFrom (Blue))
)

OWL се предлага в три варианта



Lite

частично ограничен, за да се подпомогне кривата на обучение

DL = Description Logic

Дескриптивната логика е вид First Order Logic (FOL) и служи за разсъждения

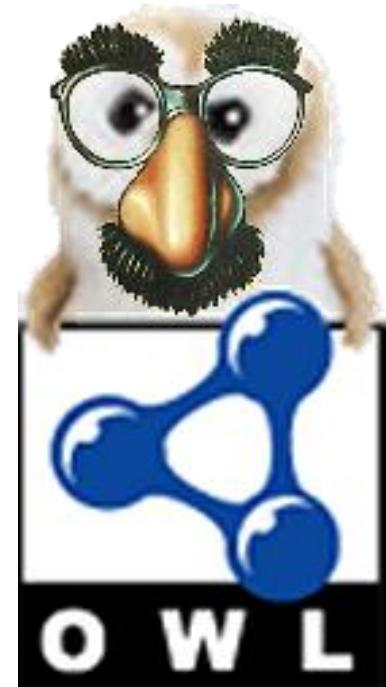
Full

неограничено ползване на OWL конструкции, но не предполага разсъждения



OWL представяния

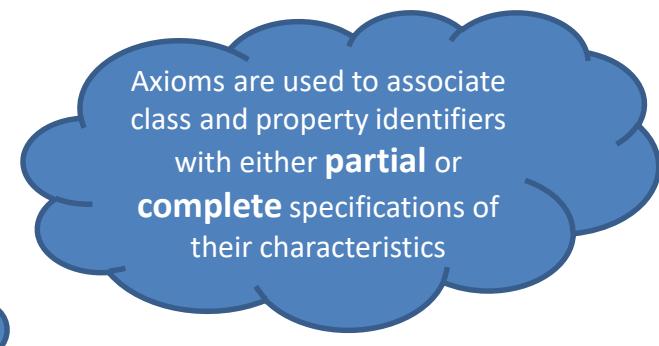
- OWL често се смята за разширение на RDF, което не е съвсем вярно
- OWL е синтактично независим език, който има няколко общи представяния
 - Abstract Syntax
 - N3
 - RDF / XML



OWL синтаксис: abstract syntax

- Най-ясен и четим за човека
синтаксис

```
Class(SpicyPizza complete
      annotation(rdfs:label "PizzaTemperada"@pt)
      annotation(rdfs:comment "Any pizza that
has a spicy topping is a SpicyPizza"@en)
      Pizza
      restriction(hasTopping
                  someValuesFrom(SpicyTopping)) )
```



OWL синтаксис: N3

- Препоръчва се за четими от човек фрагменти

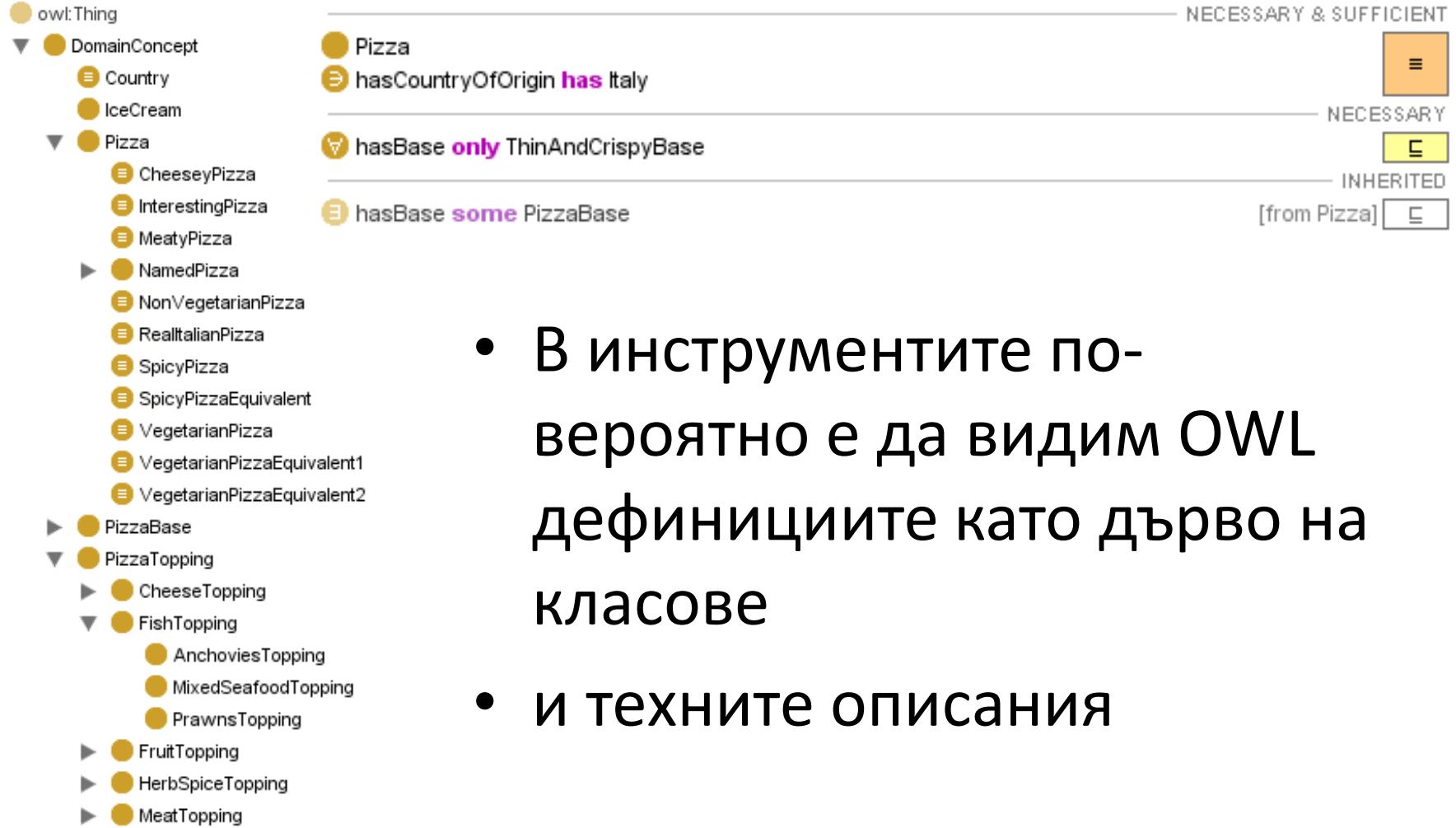
```
Default:SpicyPizza
  a owl:Class ;
  rdfs:comment "Any pizza that has a spicy topping is a
                 SpicyPizza"@en ;
  rdfs:label "PizzaTemperada"@pt ;
  owl:equivalentClass
    [ a owl:Class ;
      owl:intersectionOf (default:Pizza
        [ a owl:Restriction ;
          owl:onProperty default:hasTopping ;
          owl:someValuesFrom default:SpicyTopping
        ])
    ]
  .
```

OWL синтаксис : RDF/XML

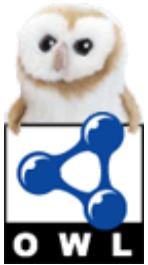
- Препоръчва се за сериализация

```
<owl:Class rdf:ID="SpicyPizza">
    <rdfs:label xml:lang="pt">PizzaTemperada</rdfs:label>
    <rdfs:comment xml:lang="en">Any pizza that has a spicy
topping is a SpicyPizza</rdfs:comment>
    <owl:equivalentClass>
        <owl:Class>
            <owl:intersectionOf rdf:parseType="Collection">
                <owl:Class rdf:about="#Pizza"/>
                <owl:Restriction>
                    <owl:onProperty>
                        <owl:ObjectProperty rdf:about="#hasTopping"/>
                    </owl:onProperty>
                    <owl:someValuesFrom rdf:resource="#SpicyTopping"/>
                </owl:Restriction>
            </owl:intersectionOf>
        </owl:Class>
    </owl:equivalentClass>
</owl:Class>
```

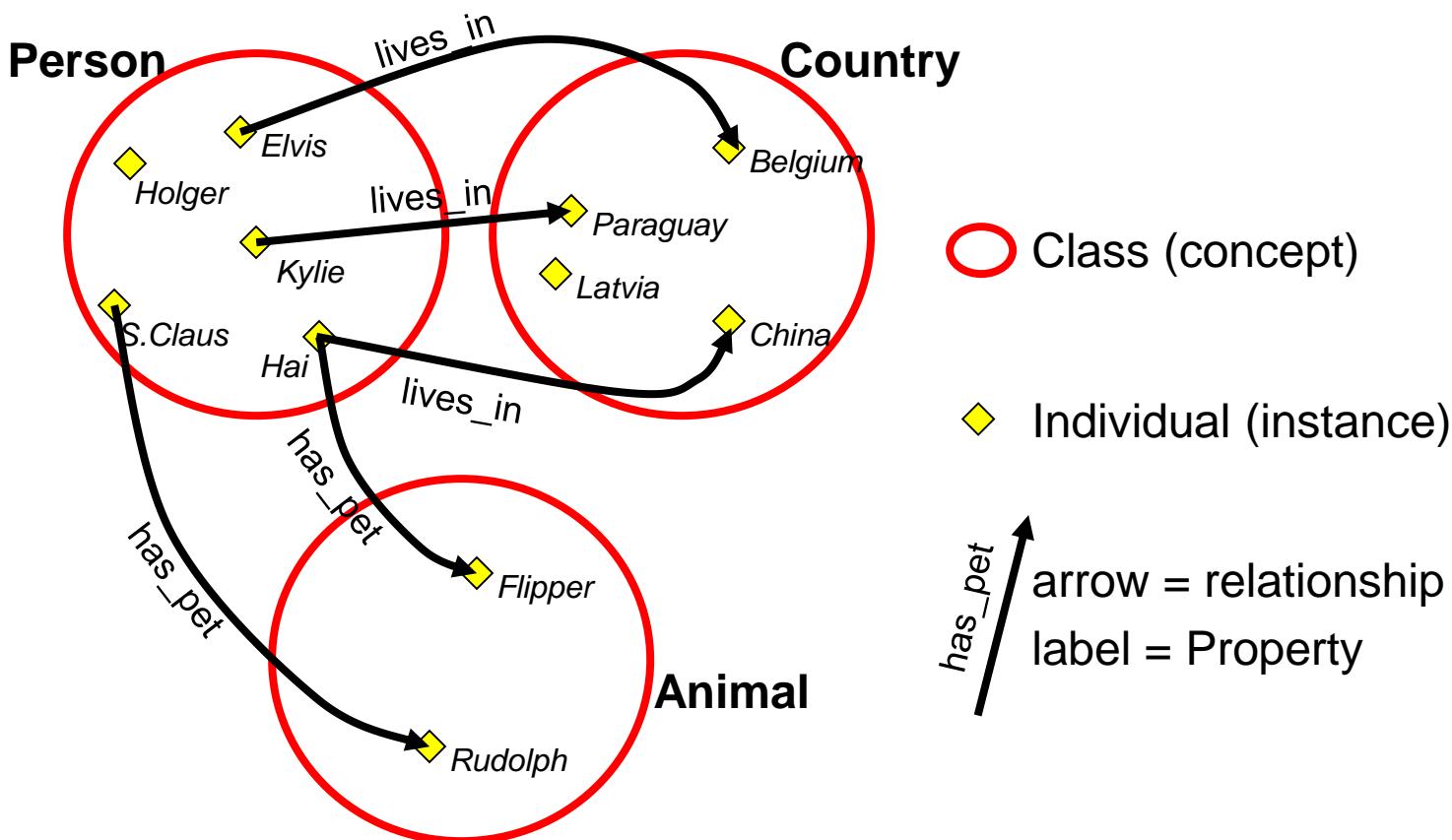
Програмните инструменти „скриват“ синтаксиса



- В инструментите по-вероятно е да видим OWL дефинициите като дърво на класове
- и техните описания



OWL конструкции



OWL конструкции: Classes

Примери: Mammal, Tree, Person, Building, Fluid, Company

- Класовете са множества от екземпляри
- като “Type”, “Concept”, “Category”, “Kind”
- Членството в класа зависи от логическото описание, а не от името
- Класът има описание, тип и релации
- Класове не трябва да бъде непременно назовани - те могат да бъдат и логически изрази – например **things that have colour Blue**

Описание на клас

Става чрез:

- идентификатор на клас (URI)
- изчерпателно изброяване (enumeration) на индивиди, които заедно формират екземплярите на клас
- ограничение на свойство (property restriction)
- пресичане (intersection) на две или повече описания на класове
- обединение (union) на две или повече описания на класа
- допълнение (complement) към описание на клас

Описание чрез изброяване

```
<owl:Class>
  <owl:oneOf rdf:parseType="Collection">
    <owl:Thing rdf:about="#Eurasia"/>
    <owl:Thing rdf:about="#Africa"/>
    <owl:Thing rdf:about="#NorthAmerica"/>
    <owl:Thing rdf:about="#SouthAmerica"/>
    <owl:Thing rdf:about="#Australia"/>
    <owl:Thing rdf:about="#Antarctica"/>
  </owl:oneOf>
</owl:Class>
```

OWL конструкции: Properties

Примери: **hasPart**, **isInhabitedBy**, **isNextTo**, **occursBefore**

- Свойствата (Properties) свързват екземплярите, тоест индивидите (Individuals)
- Индивидите са свързани чрез дадено свойство
- Релациите в OWL са **бинарни**:

Subject → predicate → Object

Individual a → hasProperty → Individual b

nick_drummond → givesTalk → owl_overview_talk_Dec_2005

OWL конструкции: Individuals

Примери: me, you, this talk, this room

- Екземплярите (Individuals) са **обектите** в предметната област
- като “Instance”, “Object”
- Един екземпляр може да бъде (често) член на множество класове
- *...every OWL class is associated with a set of individuals, called the **class extension**...*
[\(https://www.w3.org/TR/owl-ref/\)](https://www.w3.org/TR/owl-ref/)

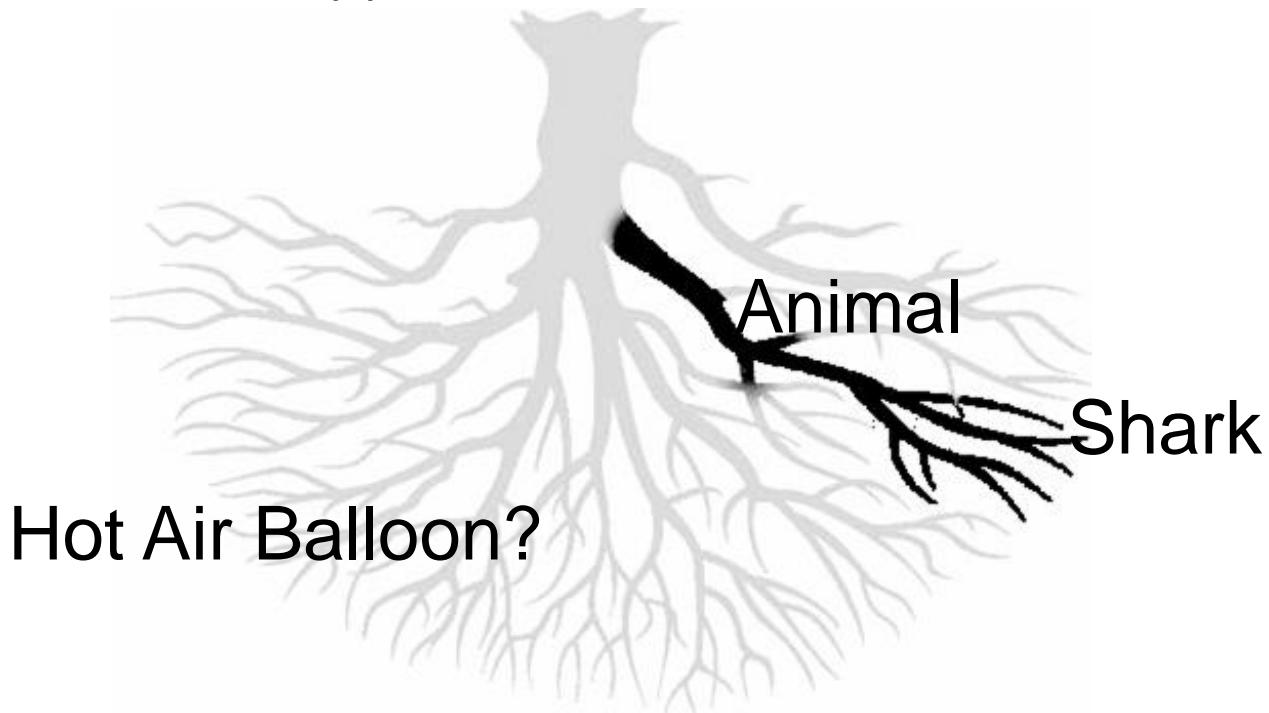
Използване на класове и екземпляри

- Повечето онтологии се състоят почти изцяло от класове
- Използваме индивиди (екземпляри), когато е необходимо да опишем даден клас
- Внимавайте при добавянето на индивиди към онтологията, тъй като те могат да ограничат многократната ѝ употреба
- Напр. не можем да създадем нов вид (клас) Program, ако Program е физическо лице

Описание на йерархия от класове

Две важни неща за класа:

- Къде можем да ги сложим?
- Къде не можем да ги сложим?



owl:Thing

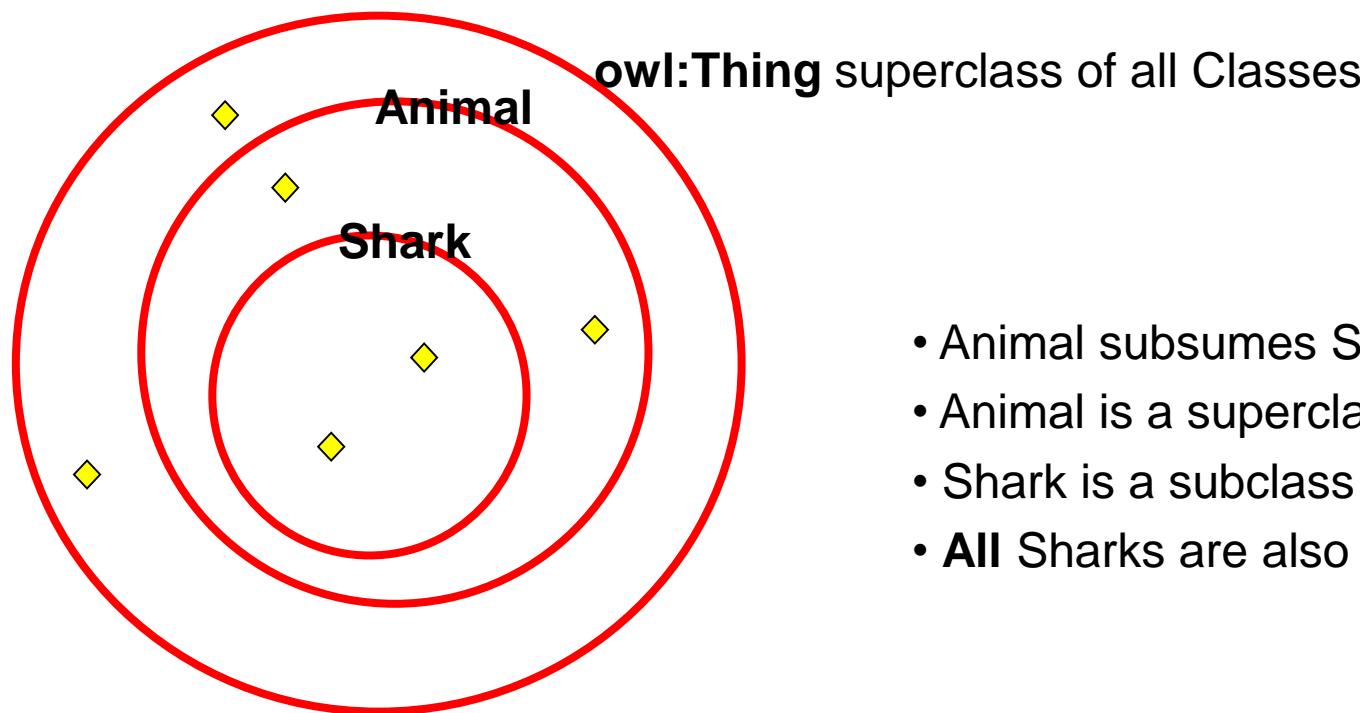
▼ Animal

Shark

Къде можем да сложим класа?

Включване (Subsumption) в OWL

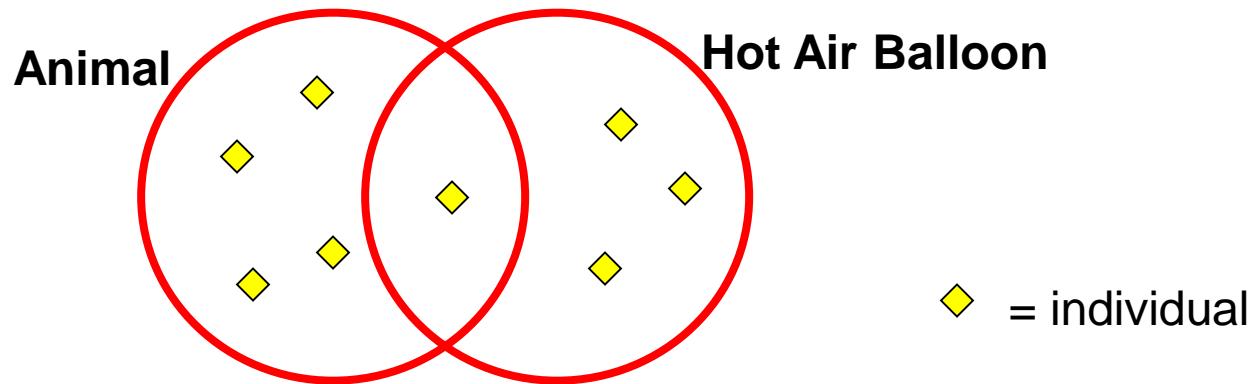
- Включването е основната ос (отношения) в OWL
- Superclass/subclass релация, от тип “is-a”
- **Всички** членове на подклас трябва да бъдат членове на неговите супер-класове



Къде не можем да сложим класа?

Disjointness in OWL

Независимо от това къде съществуват в юерархията, OWL предполага, че класовете могат да се препокриват

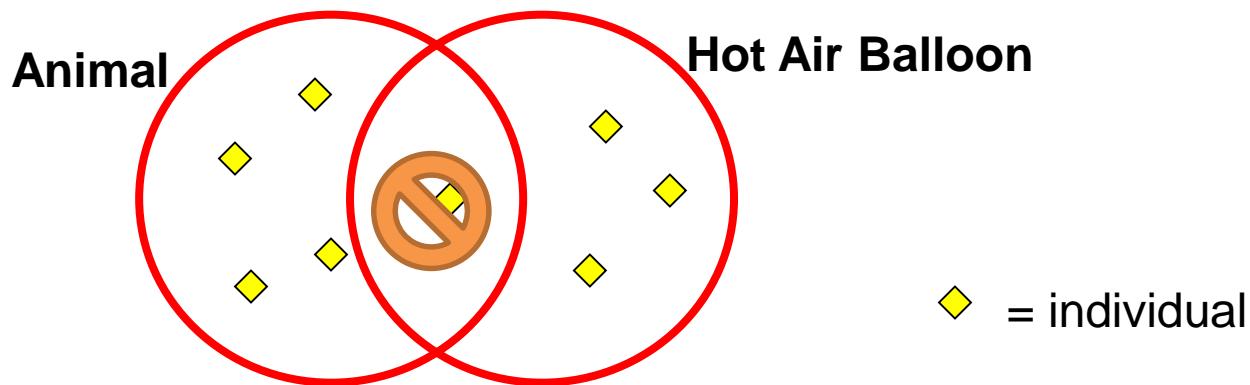


По подразбиране, един индивид може да е **Animal** и **Hot Air Balloon** в едно и също време

Къде не можем да сложим класа?

Disjointness in OWL

Ако два класа са disjoint – тогава...



...едно нещо не може да бъде **Animal** и
Hot Air Balloon едновременно

Пример

- Има два типа животни, Male и Female.

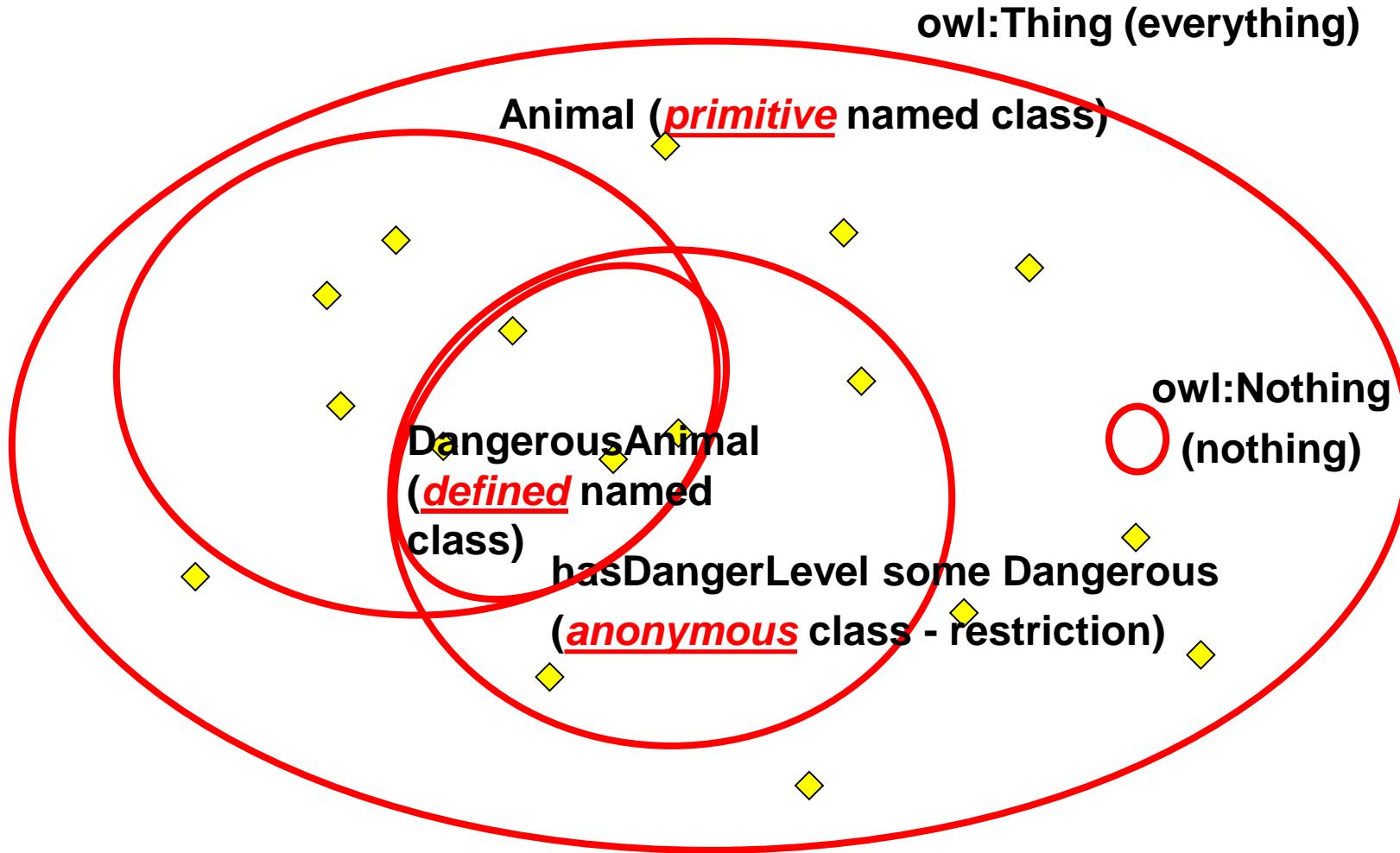
```
<rdfs:Class rdf:id="Male">
  <rdfs:subClassOf rdf:resource="#Animal"/>
</rdfs:Class>
```

- Елементът `subClassOf` element заявява, че субектът му - `Male` – е под клас на обекта му, т.е. на ресурса `#Animal`.

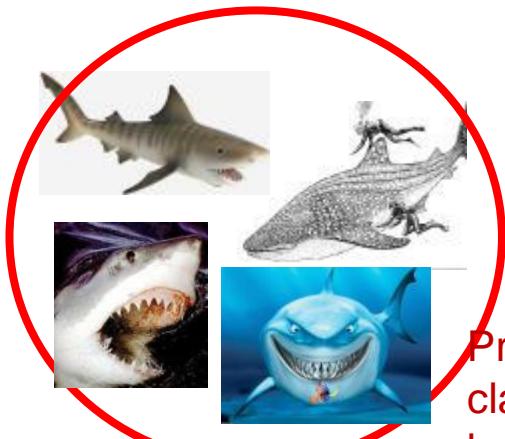
```
<rdfs:Class rdf:id="Female">
  <rdfs:subClassOf rdf:resource="#Animal"/>
  <owl:disjointWith rdf:resource="#Male"/>
</rdfs:Class>
```

- Някои животни са `Female`, но никое не е `Male` и `Female` (в тази онтология), понеже двета класа са `disjoint` (свойство, зададено чрез елемента `disjointWith`).

Типове класове



Примитивни спрямо дефинирани класове



Sharks

“Natural Kinds”

Описва задължителните свойства на членовете на класа, напр.

live underwater:

“All sharks live underwater, but not everything that lives underwater is a shark”
XML

Defined classes have necessary and sufficient conditions



Blue Things

“Smart Class” – като заявка

Както примитивните, но описват и достатъчните свойства за членство в класа, напр.

have colour Blue:

“**All** things that have colour blue are members of this class”

Анонимни класове

- Създадени от логически изрази
 - Unions and Intersections (Or, And)
 - Complements (Not)
 - Enumerations (specified membership)
 - Restrictions (related to Property use)
- Членовете на анонимен клас са множество от индивиди, които отговарят на неговата логическа дефиниция

Релации в OWL

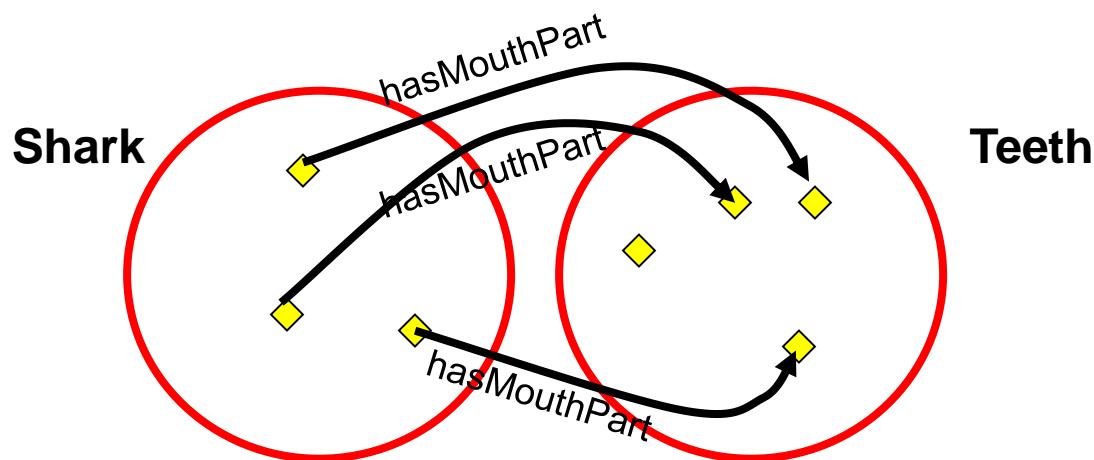
- В OWL-DL, взаимоотношения могат да се формират само между индивидите или между даден индивид и стойност на данни.
(в OWL-Full, класовете могат да имат релации помежду си, но с тях не може да се разсъждава)
- Релациите се задават чрез свойства (**Properties**)
- Можем да ограничим използването на Properties:
 - глобално – чрез указване на неща за самото Property
 - локално – чрез ограничаване на използването им в даден Class

Ограничения (Restrictions)

- Ограничения са тип анонимен клас
- Те описват отношенията, които трябва да притежават членовете на този клас

Пример

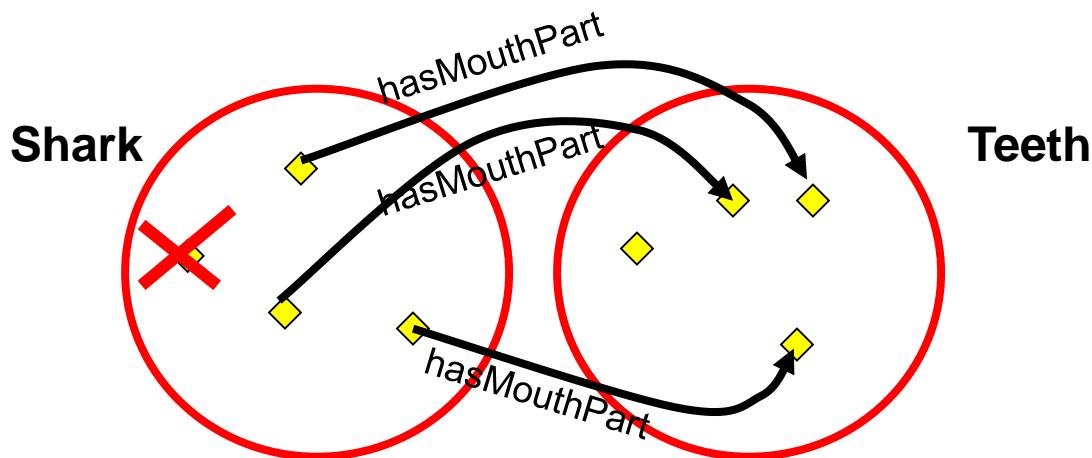
Ограничение за примитивния клас **Shark**:
задължително член на **Shark** *hasMouthPart* **Teeth**



“Every member of the **Shark** class must have *at least one* mouthpart from the class **Teeth**”

Пример

Екзистенциално ограничение за примитивния клас **Shark**:
задължително член на **Shark** *hasMouthPart* **Teeth**



“There can be no member of **Shark**, that does not have *at least one hasMouthPart relationship* with a member of class **Teeth**”

Ограничения на свойство (property restriction)

- ***Value constraints*** - built-in OWL property
 - *owl:allValuesFrom*
 - *owl:someValuesFrom*
 - *owl:hasValue*
- ***Cardinality constraints***
 - *owl:maxCardinality*
 - *owl:minCardinality*
 - *owl:Cardinality*

Типове ограничения

\exists	Existential, someValuesFrom	“Some”, “At least one”
\forall	Universal, allValuesFrom	“Only”
\exists	hasValue	“equals x”
$=$	Cardinality	“Exactly n”
\leq	Max Cardinality	“At most n”
\geq	Min Cardinality	“At least n”

“someValuesFrom” vs “allValuesFrom”

- В контекста на OWL (който като DL език) те отразяват ограниченията върху свойствата, особено за класа на стойностите на свойствата
- Напр. ако искате да заявите за автомобила **Car**, който има свойството **manufactured_by**, това свойство да е с обхват, ограничен до членове на производителя на класа
- **allValuesFrom** изисква всички стойности на свойството да са от този клас
- **someValuesFrom** изисква поне една стойност на свойството да е от този клас

:Person

```
a owl:Class ;  
rdfs:subClassOf  
[ a owl:Restriction ;  
  owl:onProperty :creatorOf ;  
  owl:allValuesFrom :Artefact  
].
```

all things created
by persons are
artefacts

artefacts are
things that are
created by at least
one person

:Artefact

```
a owl:Class ;  
owl:equivalentClass  
[ a owl:Restriction ;  
  owl:onProperty :createdBy ;  
  owl:someValuesFrom :Person  
].
```

```
<owl:Restriction>
```

```
  <owl:onProperty rdf:resource="#hasParent" />
```

```
  <owl:allValuesFrom rdf:resource="#Human" />
```

```
</owl:Restriction>
```

an anonymous OWL class of all individuals for which the hasParent property only has values of class Human

```
<owl:Restriction>
```

```
  <owl:onProperty rdf:resource="#hasParent" />
```

```
  <owl:someValuesFrom rdf:resource="#Physician" />
```

```
</owl:Restriction>
```

a class of individuals which have at least one parent who is a physician

*"Red wine is a subclass of all things
that **have** a red color."*

:RedWine

a owl:Class ;

rdfs:subClassOf

[a owl:Restriction ;

owl:onProperty :color ;

owl:hasValue

red^^<<http://www.w3.org/2001/XMLSchema#string>>

].



```
<owl:Class>
<owl:intersectionOf rdf:parseType="Collection">
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
<owl:Thing rdf:about="#Tosca" />
<owl:Thing rdf:about="#Salome" />
</owl:oneOf>
</owl:Class>
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
<owl:Thing rdf:about="#Turandot" />
<owl:Thing rdf:about="#Tosca" />
</owl:oneOf>
</owl:Class>
</owl:intersectionOf>
</owl:Class>
```

Result: a class with
one individual,
namely **Tosca**

```
<owl:Class>
<owl:unionOf rdf:parseType="Collection">
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
<owl:Thing rdf:about="#Tosca" />
<owl:Thing rdf:about="#Salome" />
</owl:oneOf>
</owl:Class>
<owl:Class>
<owl:oneOf rdf:parseType="Collection">
<owl:Thing rdf:about="#Turandot" />
<owl:Thing rdf:about="#Tosca" />
</owl:oneOf>
</owl:Class>
</owl:unionOf>
</owl:Class>
```

Result: a class with three individuals, namely Tosca, Salome, and Turandot

//An **owl:complementOf** statement describes a class for which the class extension contains exactly those individuals that do not belong to the class extension of the class description that is the object of the statement.

```
<owl:Class>  
  <owl:complementOf>  
    <owl:Class rdf:about="#Meat"/>  
  </owl:complementOf>  
</owl:Class>
```



this class description
contains all
individuals that do
not belong to the
class Meat



Типове свойства

- Different Types:
 - Object Property
 - свързва членове на класа (individuals) с членове
 - Datatype Property
 - свързва членове с данни (int, string, float etc)
 - Annotation Property
 - прикрепва метаданни към classes, individuals или properties

Характеристики на свойствата

- Домейн (Domain) и обхват (range)
- Йерархия от свойства
- Обратни свойства (Inverse properties)
- Свойствата могат да бъдат:
 - Transitive
 - Functional
 - Inverse Functional
 - Symmetric

OWL в един слайд

- **Symmetric:** if $P(x, y)$ then $P(y, x)$
- **Transitive:** if $P(x, y)$ and $P(y, z)$ then $P(x, z)$
- **Functional:** if $P(x, y)$ and $P(x, z)$ then $y=z$
- **InverseOf:** if $P_1(x, y)$ then $P_2(y, x)$
- **InverseFunctional:** if $P(y, x)$ and $P(z, x)$ then $y=z$
- **allValuesFrom:** $P(x, y)$ and $y=\text{allValuesFrom}(C)$
- **someValuesFrom:** $P(x, y)$ and $y=\text{someValuesFrom}(C)$
- **hasValue:** $P(x, y)$ and $y=\text{hasValue}(v)$
- **cardinality:** $\text{cardinality}(P) = N$
- **minCardinality:** $\text{minCardinality}(P) = N$
- **maxCardinality:** $\text{maxCardinality}(P) = N$
- **equivalentProperty:** $P_1 = P_2$
- **intersectionOf:** $C = \text{intersectionOf}(C_1, C_2, \dots)$
- **unionOf:** $C = \text{unionOf}(C_1, C_2, \dots)$
- **complementOf:** $C = \text{complementOf}(C_1)$
- **oneOf:** $C = \text{one of}(v_1, v_2, \dots)$
- **equivalentClass:** $C_1 = C_2$
- **disjointWith:** $C_1 \neq C_2$
- **sameIndividualAs:** $I_1 = I_2$
- **differentFrom:** $I_1 \neq I_2$
- **AllDifferent:** $I_1 \neq I_2, I_1 \neq I_3, I_2 \neq I_3, \dots$
- **Thing:** I_1, I_2, \dots

Properties limited to a single value

Legend:

Properties are indicated by: P, P_1, P_2 , etc.

Specific classes are indicated by: x, y, z

Generic classes are indicated by: $C, C_1, C_2\dots$

Values are indicated by: $v, v_1, v_2\dots$

Instance documents are indicated by: I_1, I_2, I_3 , etc.

A number is indicated by: N

$P(x, y)$ is read as: “property P relates x to y ”

Други примери

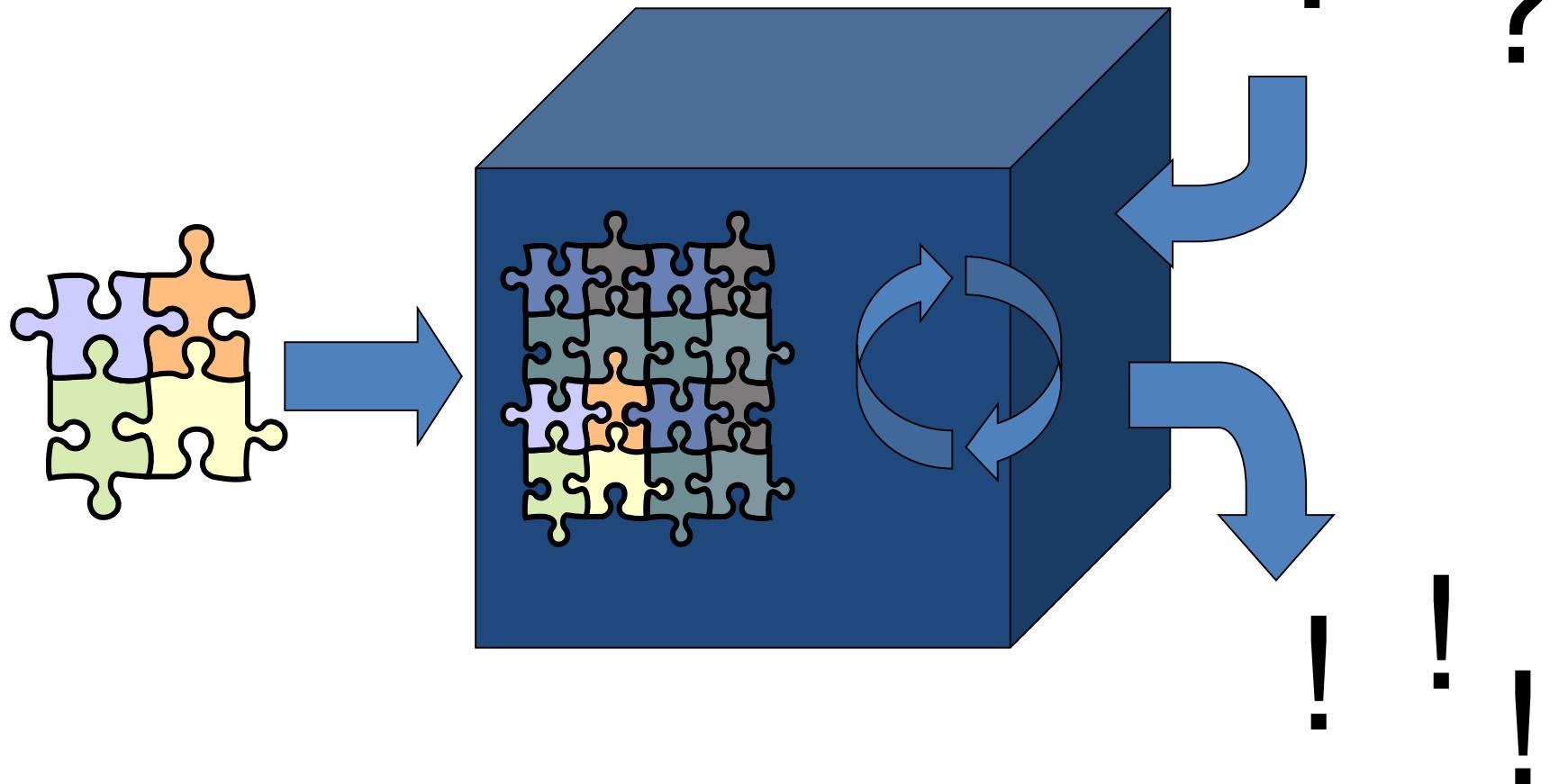
- $\text{Woman} \equiv \text{Person} \sqcap \text{Female}$
- $\text{Man} \equiv \text{Person} \sqcap \neg \text{Woman}$
- $\text{Mother} \equiv \text{Woman} \sqcap \exists \text{hasChild}.\text{Person}$
- $\text{Father} \equiv \text{Man} \sqcap \exists \text{hasChild}.\text{Person}$
- $\text{Parent} \equiv \text{Father} \sqcup \text{Mother}$
- $\text{Grandmother} \equiv \text{Mother} \sqcap \exists \text{hasChild}.\text{Parent}$

Можем да заключим (макар да не е изрично дефинирано), че:

- $\text{Grandmother} \sqsubseteq \text{Person}$
- $\text{Grandmother} \sqsubseteq \text{Woman}$



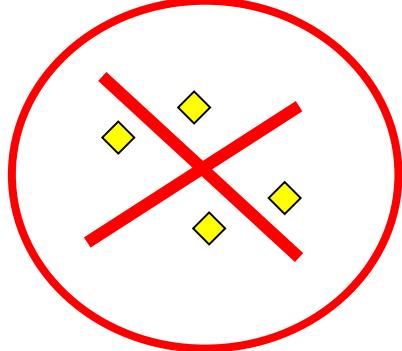
Разъждения (Reasoning) & изводи (Inference)



Reasoners: Inference

- Reasoners се използват за заключения относно информация, която не е изрично налична в онтологията
- Наричат се още Класификатори
- Стандартни услуги на reasoner са:
 - Проверка за консистентност - Consistency Checking
 - Проверка за включване - Subsumption Checking (Automatic Subsumption)
 - Проверка за еквивалентност - Equivalence Checking
 - Проверка за инстанциране - Instantiation Checking

Проверка за консистентност

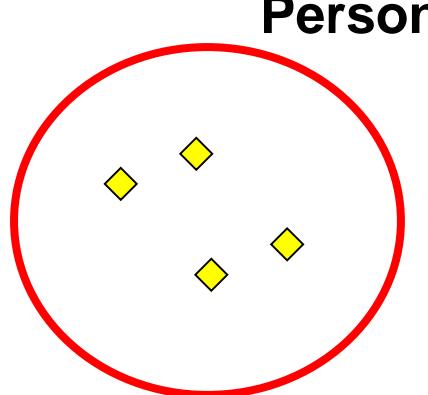


Shark (primitive class)

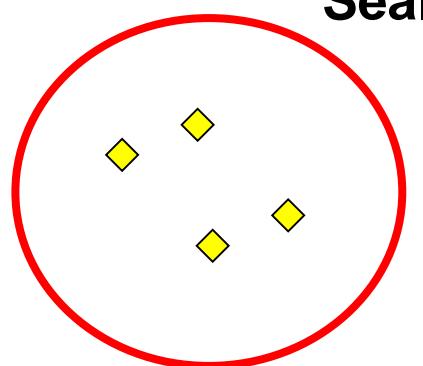
Animal and

eats some (Person and Seal)

Inconsistent = cannot contain any individuals



Person



Seal

XML

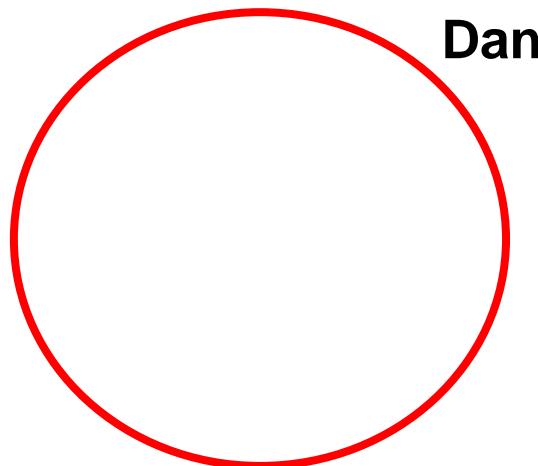
OWL

Disjoint (Person, Seal)
Person **and** Seal = empty
Cannot have some empty



Автоматична класификация

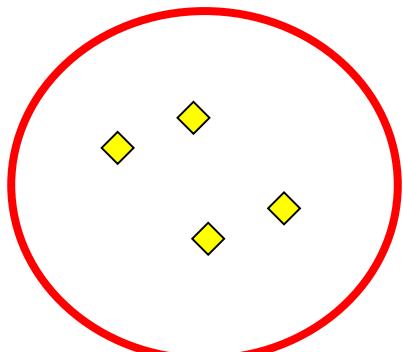
Тривиален пример



DangerousAnimal (defined class)

Animal and

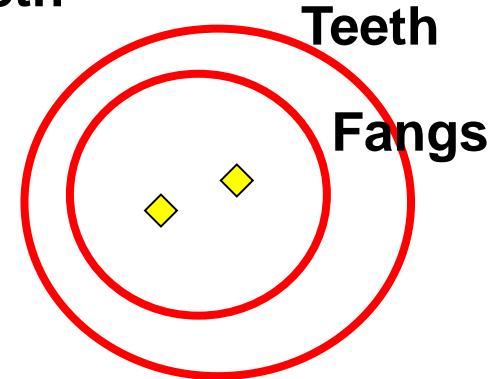
hasMouthPart some **Teeth**



Shark (primitive class)

Animal and

hasMouthPart some **Fangs**



Кога използваме Reasoner

- При разработка - използваме компилатор на онтологии за проверка на значението
- При публикуване - за изводи, направени за потребителски приложения
- По време на изпълнение на приложения - като механизъм за заявки (особено полезно за по-малките онтологии)
 - DL queries
 - SPARQL queries

SPARQL

- SPARQL е език за заявки за RDF графи.
Научните бази данни използват RDF/OWL и SPARQL като начин за описание, свързване и извличане на сложни знания.

- SPARQL заявка за списък с всички тройки:

SELECT * WHERE {

?sub ?pred ?obj .

}

LIMIT 100

OFFSET 0

DL queries

The screenshot shows the Protégé 5.5 interface for performing DL queries. On the left is the Class hierarchy browser, displaying classes like owl:Thing, Person, Customer, Employee, Pizza, PizzaBase, PizzaTopping, and Spiciness. The main area is titled "DL Query" and contains a "Query (class expression)" input field with the query: "Customer and purchasedPizza some (hasTopping some (hasSpiciness value Hot))". Below the query are "Execute" and "Add to ontology" buttons. To the right, the "Query results" section shows Superclasses (Customer, Person, owl:Thing), Subclasses (owl:Nothing), and Instances (Customer1, Customer4, Customer8, Customer9). A "Query for" sidebar lists options for Direct superclasses, Superclasses (checked), Equivalent classes, Direct subclasses, Subclasses (checked), and Instances (checked). A "Result filters" sidebar includes a "Name contains" search bar. At the bottom, there are buttons for "Reasoner active" and "Show Inferences".

Michael DeBellis (2021) *A Practical Guide to Building OWL Ontologies Using Protégé 5.5 and Plugins, Edition 3.2*
https://drive.google.com/file/d/1A3Y8T6nIfXQ_UQOpCAr_HFSCwpTqELeP/view

Let's find all the Pizzas purchased by a Customer.

SPARQL queries

SPARQL query:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
PREFIX pizza: <http://www.semanticweb.org/pizzatutorial/ontologies/2020/PizzaTutorial#>
SELECT *
WHERE { ?customer pizza:purchasedPizza ?pizza }
```

customer	pizza
Customer8	ChicagoAmericanaHotPizza1
Customer4	AmericanaHotPizza3
Customer4	HotVeggiePizza1
Customer5	MediumVeggiePizza1
Customer9	HotVeggiePizza3
Customer2	CustomPizza2
Customer8	CustomPizza1
Customer1	AmericanaHotPizza1
Customer5	AmericanaPizza1
Customer3	SohoPizza1
Customer1	AmericanaHotPizza2

Michael DeBellis (2021) A Practical Guide to Building OWL Ontologies Using Protégé 5.5 and Plugins, Edition 3.2
https://drive.google.com/file/d/1A3Y8T6nIfXQ_UQOpCAr_HFSCwpTqELeP/view

Semantic Web Rule Language (SWRL)

- SWRL е създаден за определени видове изводи, които не могат да бъдат направени чрез DL
- Както всички системи с правила, SWRL се състои от лява страна (наречена antecedent) и дясна страна (наречена consequent - следствие). Двете са разделени от \rightarrow . Всеки израз в SWRL правило е разделен със знак \wedge .
- Следствието от правилото се задейства, ако и само ако всеки израз в предходния е изпълнен.
- Всички параметри (променливи, които са заместващи символи и се обвързват динамично при задействане на правилото) се предхождат от ?.

SWRL изрази

- 1. За класове - това е името на клас, последвано от скоби с параметър вътре. Например `Customer(?c)` ще свърже `?c` към екземпляр на класа `Customer`.
- За свойства - това е името на свойство, последвано от скоби и два параметъра: първият за индивида, който се тества, а вторият за свързване със стойността на това свойство за този индивид. Например `purchasedPizza(?c, ?p)` ще свърже `?p` с всяка пица, закупена от всеки клиент `?c`.
- Вградени функции. SWRL има редица вградени функции за извършване на математически тестове, тестове за низове и т.н. Всички вградени SWRL са предшествани от префиксът `swrlb`. Например вграденият математически `swrlb:greaterThan(?np, 1)` е успешен, ако стойността на `?np` е по-голяма от 1.

Вградените SWRL са документирани тук:
<https://www.w3.org/Submission/SWRL/>

SWRL примери

- Customer(?c) ^ numberOfPizzasPurchased(?c, ?np)
- Customer(?c) ^ numberOfPizzasPurchased(?c, ?np) ^ hasSpicinessPreference(?c, ?spr) ^ isMilderThan(?spr, Hot) ^ swrlb:greaterThan(?np, 1) -> hasDiscount(?c, 0.1).

Michael DeBellis (2021) A Practical Guide to Building OWL Ontologies Using Protégé 5.5 and Plugins, Edition 3.2
https://drive.google.com/file/d/1A3Y8T6nIfXQ_UQOpCAr_HFSCwpTqELeP/view

Грешки при моделиране с онтологии

- Основни грешки:
 - Неправилно използване на домейн и обхват
 - Неразбиране на intersections и други конструкции
 - Неразбиране на Open World Assumption
 - Злоупотреба или липса на disjoints
- За допълнителен прочит:
 - OWL Pizzas: Common errors & common patterns
<http://www.co-ode.org/resources/papers/>

OWL редактори



<http://www.xml.com/pub/a/2004/07/14/onto.html>



- Представлява среда за моделиране на знания
- Безплатна, софтуер с отворен код
- Разработена от Станфорд (департамент по медицинска информатика)
- Разполага с голяма потребителска общност

<http://protege.stanford.edu>

OWL пример в Protégé (1)

- Клас (Class)
 - Person superclass
 - Man, Woman subclasses
- Свойства (Properties)
 - isWifeOf, isHusbandOf
- Х-ки на свойства, ограничения (restrictions)
 - inverseOf
 - domain
 - range
 - cardinality
- Изрази с класове
 - disjointWith

OWL Example in Protégé (2)

MyOntology Protégé 2.0 beta (file:/C:/ellisr/ontology/MyOntology.pprj, OWL files) [] [] []

Project Edit Window OWL Help

OWLClasses Properties Forms Individuals Ontology

Class Hierarchy V C X

- :THING
- Person
 - Woman
 - Man

Annotations V C + -

Property	Value

Properties at Class V C X + -

Name	Type	Cardinality	Other Facets
isWifeOf	Instance	single	classes={Man}

Restrictions V C \forall_c \exists_c \exists_c^* $=_c$ \geq_c \leq_c X

Property	Restriction	Filler

Definition V C + - X

Disjoint classes V C + E + - X

- Man

XML

OWL

89

OWL пример в Protégé (3)

MyOntology Protégé 2.0 beta (file:/C:/ellisr/ontology/MyOntology.pprj, OWL files)

Project Edit Window OWL Help

OWLClasses Properties Forms Individuals Ontology

Properties V C X

isHusbandOf (type=owl:ObjectProperty)

Name Labels SameAs DifferentFrom Annotations

isHusbandOf

Documentation

Cardinality HasValue V C + -

required at least 1

multiple at most 1

Equivalent V C + -

Range Some Values From

Instance

Classes V C + - X

Woman

Domain defined

Domain V C + -

Man

Inverse Property V C + -

isWifeOf

OWL Symmetric Transitive AnnotationProperty InverseFunctional

XML 90



SUBCLASS EXPLORER

For Project: pizza.owl

Asserted Hierarchy

- owl:Thing
- DomainConcept
 - Country
 - IceCream
 - Pizza
 - CheeseyPizza
 - InterestingPizza
 - MeatyPizza
 - NamedPizza
 - NonVegetarianPizza
 - RealItalianPizza
 - SpicyPizza
 - SpicyPizzaEquivalent
 - VegetarianPizza
 - VegetarianPizzaEquivalent1
 - VegetarianPizzaEquivalent2
 - PizzaBase
 - PizzaTopping
 - CheeseTopping
 - FishTopping
 - AnchoviesTopping
 - MixedSeafoodTopping
 - PrawnsTopping
 - FruitTopping
 - HerbSpiceTopping
 - MeatTopping

CLASS EDITOR

For Class: RealItalianPizza (instance of owl:Class) Inferred View

Annotations

Property	Value	Lang
rdfs:comment	This defined class has conditions that are part of the definition: ie any Pizza that has the country of origin, Italy is a RealItalianPizza. It also has conditions that merely describe the members - that all RealItalianPizzas must only have ThinAndCrispy bases.	en
rdfs:label	PizzaitalianaReal	pt

Asserted Conditions

- Pizza NECESSARY & SUFFICIENT
- hasCountryOfOrigin has Italy NECESSARY
- hasBase only ThinAndCrispyBase INHERITED [from Pizza]
- hasBase some PizzaBase

Disjoints

Logic View Properties View



Програмиране с OWL



- Protégé OWL API
 - https://protegewiki.stanford.edu/wiki/ProtegeOWL_API_Programmers_Guide
- Wonderweb OWL API
 - <http://wonderweb.mpi.ac.uk/owl/>
- Jena Ontology API
 - <https://jena.apache.org/documentation/ontology/>
- OWL API
 - <https://owlapi.sourceforge.net/>

Reasoners

- FaCT++
- Pellet
- RACER
-

ПРИМЕРНИ ОНТОЛОГИИ

- ОВО – Open BioMedical Ontologies
- The Gene Ontology
- Bio tutorial and Pizza tutorial examples on the CO-ODE site
- Libraries are commonly published on OWL editor websites
- Search using Google or Swoogle

Примерни приложения

- PizzaFinder (dummy query application)
- COHSE – dynamic hyperlinking using ontologies
- Protein Phosphatase Modelling – ask Robert Stevens
- OWL Validator
- GONG (Gene Ontology Next Generation)
- АКТ
<http://www.aktors.org/>
- The Semantic Web Challenge
<http://challenge.semanticweb.org/>



За самостоятелна работа

- Примери и наръчник



Pizza tutorial

<http://owl.cs.manchester.ac.uk/research/co-ode/>

Michael DeBellis (2021) A Practical Guide to Building OWL
Ontologies Using Protégé 5.5 and Plugins, Edition 3.2

https://drive.google.com/file/d/1A3Y8T6nIfXQ_UQOpCAr_HFSC_wpTqELeP/view



<mailto:bbontchev@fmi.uni-sofia.bg>

time to say
goodbye