

Pizza Ontology

*“a review of core concepts
for building a pizza
ontology”*

presented by:

Atif Khan
infoTrellis
<http://www.infotrellis.com/>

presentation material based on:

Horridge, Matthew.

*"A Practical Guide To Building OWL Ontologies Using Protégé 4
and CO-ODE Tools – Edition 1.3". The University of Manchester (2011).*

Outline

- Disclaimer
 - I am **not** an ontology engineer
- Goal
 - duration ~ 30 mins
 - review some basic ontology components
 - concepts, object properties, data properties, individuals
 - classification
 - introduce Protégé ontology editor
 - share my experience building the Pizza Ontology

Core Terminology

- Ontology

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

R. Studer (1998)

- Components

- *concepts* define aggregation of things
- *individuals* are instances of concepts
- *properties* link concepts/individuals

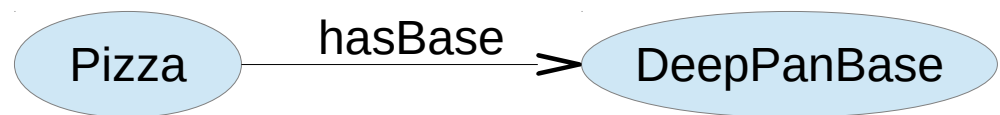
Core Terminology

- Triples
 - a ***representation*** of ontological components
 - using the following notation

Subject verb Object

- example: “a pizza has a deep pan base”

Pizza hasBase DeepPanBase



Why Use Ontologies

- Precision of:
 - representation/expression
 - information sharing
 - knowledge inference



“Now! *That* should clear up a few things around here!”

<http://photos1.blogger.com/blogger2/1715/1669/1600/larson-oct-1987.gif>

Creating a Pizza Ontology



Protégé

Version 4.2.0 (Build 284)



OWL

Web Ontology Language

<http://www.w3.org/TR/owl-features/>

Define Core Concepts

- Identify core concepts
 - *Pizza*
 - *Pizza Base*
 - *Pizza Toppings*

untitled-ontology-4 (http://www.semanticweb.org/atif/ontologies/2013/0/untitled-ontology-4) : [http://www.semanticweb.org/atif/ontolo

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-4 Search for entity

Property matrix Individuals Individuals matrix OWLViz DL Query OntoGraf SPARQL Query Ontology Differ example1.owl

Active Ontology Entities Classes Object Properties Data Properties Class matrix Annotation Properties

Class hierarchy (inferred)

Class hierarchy

Class hierarchy: PizzaBase

- Thing
 - PizzaBase
 - PizzaTopping
 - Pizza

Annotations Usage

Annotations: PizzaBase

Annotations +

Description: PizzaBase

Equivalent To +

SubClass Of +

SubClass Of (Anonymous Ancestor)

Members +

Target for Key +

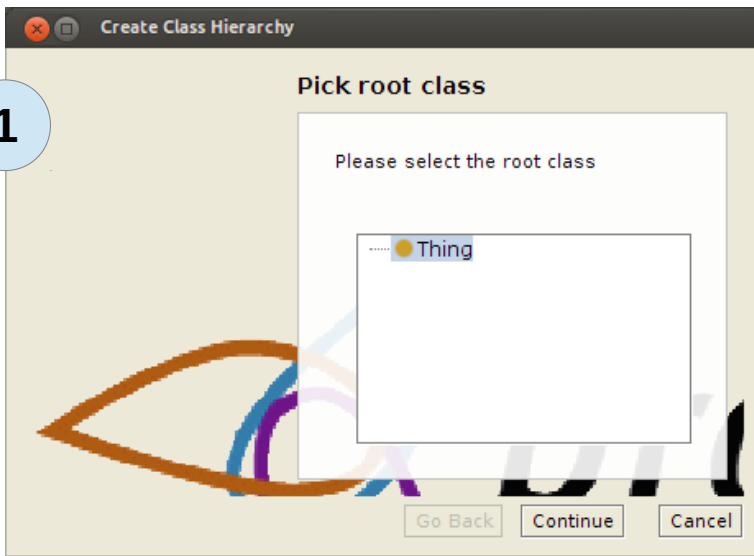
Disjoint With +

No Reasoner set. Select a reasoner from the Reasoner menu Show Inferences

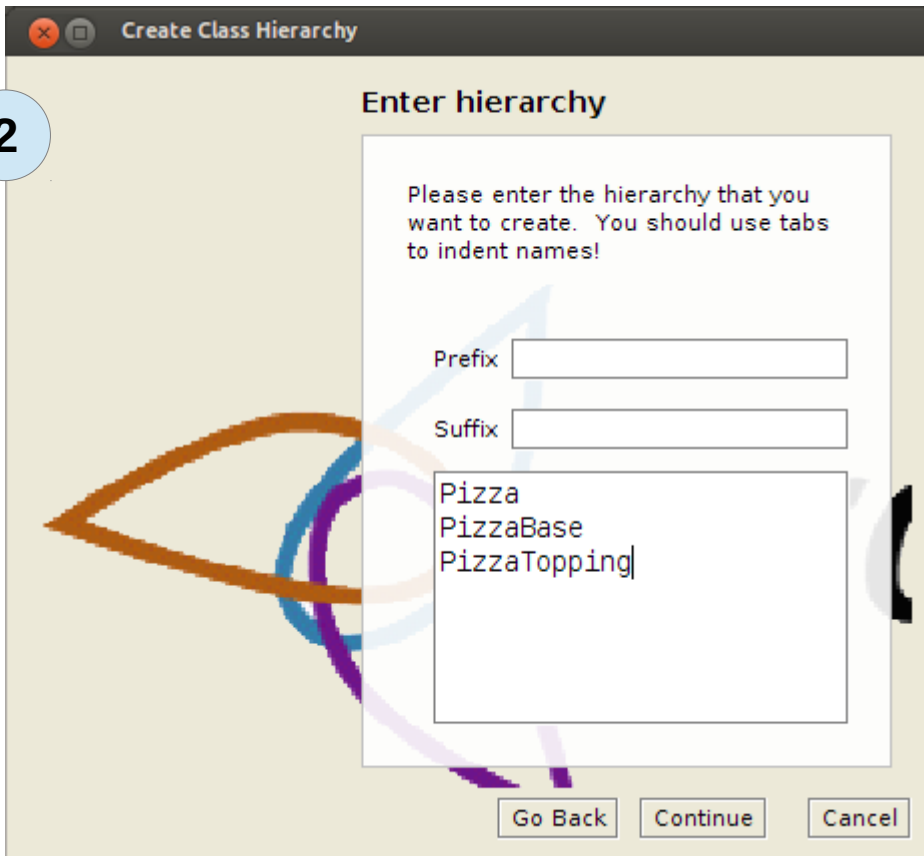
Define Core Concepts

- Unique name assumption
 - need to explicitly define **sameness** & **uniqueness** using
 - Equivalent to
 - Disjoint with

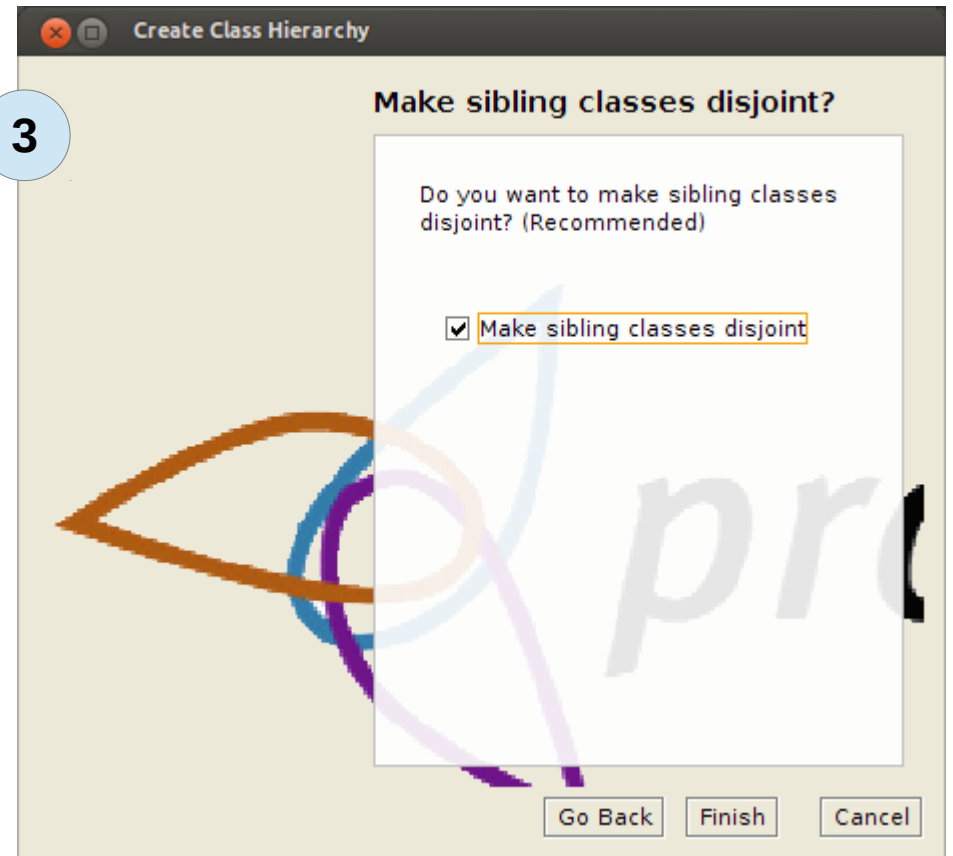
1



2



3



The screenshot shows the Protege ontology editor interface. At the top, the browser address bar displays 'untitled-ontology-4 (http://www.semanticweb.org/atif/ontologies/2013/0/untitled-ontology-4)'. The menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. Below the menu is a search bar for entities. The main workspace is divided into several panes. On the left, the 'Class hierarchy (inferred)' pane shows a tree structure: Thing (parent), PizzaBase (child), PizzaTopping (child), and Pizza (child). A blue arrow labeled 'Concept Hierarchy' points to this tree. The right side of the workspace is split into two panes. The top pane, 'Annotations: Pizza', is currently empty. The bottom pane, 'Description: Pizza', shows the class description for 'Pizza'. It lists 'SubClass Of' as 'Thing' and 'Disjoint With' as 'PizzaTopping, PizzaBase'. A blue arrow labeled 'Disjoint Concepts' points to the 'Disjoint With' section. At the bottom of the interface, a status bar indicates 'No Reasoner set. Select a reasoner from the Reasoner menu' and a checked checkbox for 'Show Inferences'.

Define Properties

- Link concepts using properties
 - a pizza has a deep pan base (*hasBase*)
 - a pizza has a mozzarella cheese topping (*hasCheeseTopping*)
 - a pizza has a tomato and cheese topping (*hasTomatoTopping*) and (*hasCheeseTopping*)
- Property Hierarchy
 - hasBase ► hasIngredient ◀ hasTopping

Define Inverse Properties

- Inverse property
 - each object property may have a corresponding inverse property
 - “a pizza has a deep pan base”
 \equiv a deep pan is a base of a pizza
- (isBaseOf)* is inverse of *(hasBase)*
(hasBase) is inverse of *(isBaseOf)*

The screenshot shows a web-based ontology editor interface. At the top, the browser address bar displays 'untitled-ontology-4 (http://www.semanticweb.org/atif/ontologies/2013/0/untitled-ontology-4)'. The main menu includes 'File', 'Edit', 'View', 'Reasoner', 'Tools', 'Refactor', 'Window', and 'Help'. Below the menu is a search bar for entities. The interface is divided into several panes:

- Object property hierarchy: isToppingOf**: A tree view on the left showing the hierarchy of properties. It includes 'topObjectProperty', 'isIngredientOf', 'isToppingOf', 'isBaseOf', 'hasIngredient', 'hasBase', and 'hasTopping'. A blue oval highlights this area with the text "is_a" hierarchy.
- Annotations: isToppingOf**: A pane showing annotations for the selected property.
- Characteristics**: A list of checkboxes for property characteristics: Functional, Inverse functional, Transitive, Symmetric, Asymmetric, Reflexive, and Irreflexive.
- Description: isToppingOf**: A pane showing the description of the property. It includes sections for 'Equivalent To', 'SubProperty Of', 'Inverse Of', 'Domains (intersection)', and 'Ranges (intersection)'. The 'Inverse Of' section is highlighted with a blue oval and labeled 'inverse property', showing that 'isToppingOf' is the inverse of 'hasTopping'.

At the bottom of the interface, a status bar indicates 'No Reasoner set. Select a reasoner from the Reasoner menu' and a checked checkbox for 'Show Inferences'.

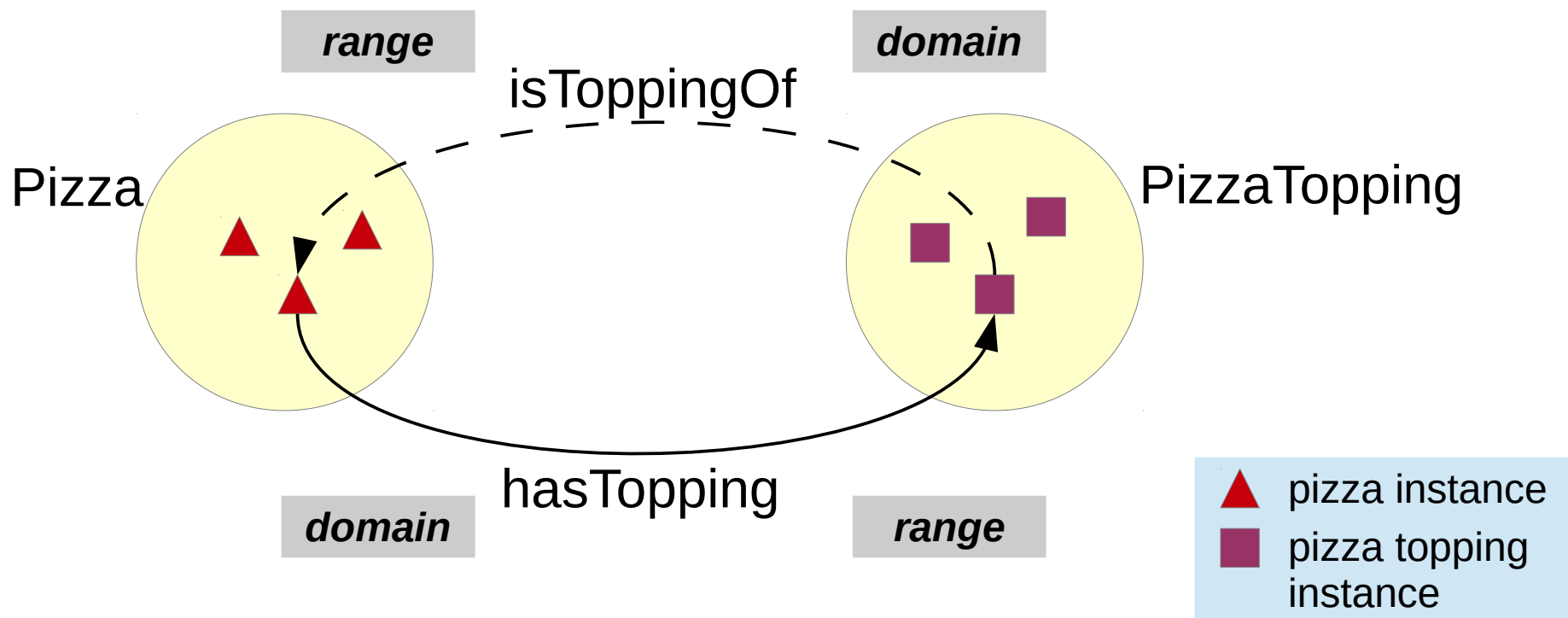
Characteristics of Properties

- OWL *primitives* to enrich relationship definitions (see §4.6)
 - functional & inverse functional
 - transitive
 - symmetric & anti-symmetric
 - reflexive & irreflexive

Property Domains & Ranges

- Definition

- properties link individuals from the domain to individuals from the range



untitled-ontology-4 (http://www.semanticweb.org/atif/ontologies/2013/0/untitled-ontology-4) : [//home/atif/Desktop/PizzaOntology/presentation/examples/example3.owl]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-4 (http://www.semanticweb.org/atif/ontologies/2013/0/untitled-ontology-4) Search for entity

Individuals Property matrix Individuals matrix OWLviz DL Query OntoGraf Ontology Differences SPARQL Query

Active Ontology Entities Classes Object Properties Data Properties Annotation Properties Class matrix

Object property hierarchy: hasTopping

- topObjectProperty
 - isIngredientOf
 - isToppingOf
 - isBaseOf
 - hasIngredient
 - hasBase
 - hasTopping

Annotations Usage

Annotations: hasTopping

Annotations +

example3.owl

Characteristic

- Functional
- Inverse functional
- Transitive
- Symmetric
- Asymmetric
- Reflexive
- Irreflexive

Description: hasTopping

Subproperty of

- hasIngredient

Inverse Of +

- isToppingOf

Domains (intersection) +

- Pizza

Range

Ranges (intersection) +

- PizzaTopping

Disjoint With +

- hasBase

No Reasoner set. Select a reasoner from the Reasoner menu Show Inferences

Property Restrictions

- Restriction \approx Anonymous Class
 - a restriction is a class definition that groups individuals together based on one or more object properties
- Example
 - class of individuals that have at least one “*hasTopping*” relationship to individuals that are members of MozzarellaTopping

Existential Restrictions

- Intention
 - describe “some values from” restrictions
- Example
 - a pizza must have a pizza base

Existential Restrictions

The screenshot shows a software window titled "Pizza" with a dark header bar. In the top right corner of the window, there is a red rectangular label with the text "example3.owl". The main area of the window contains four tabbed panels: "Class expression editor" (selected), "Object restriction creator", "Data restriction creator", and "Class hierarchy". The "Class expression editor" tab is active, and its text area contains the OWL class expression "hasBase some PizzaBase", where "some" is highlighted in pink. At the bottom of the window, there are two buttons: "OK" and "Cancel".

Existential Restrictions

- Implication of “*hasBase some PizzaBase*”
 - if something is a Pizza then it is **necessary** for it to have a kind of PizzaBase

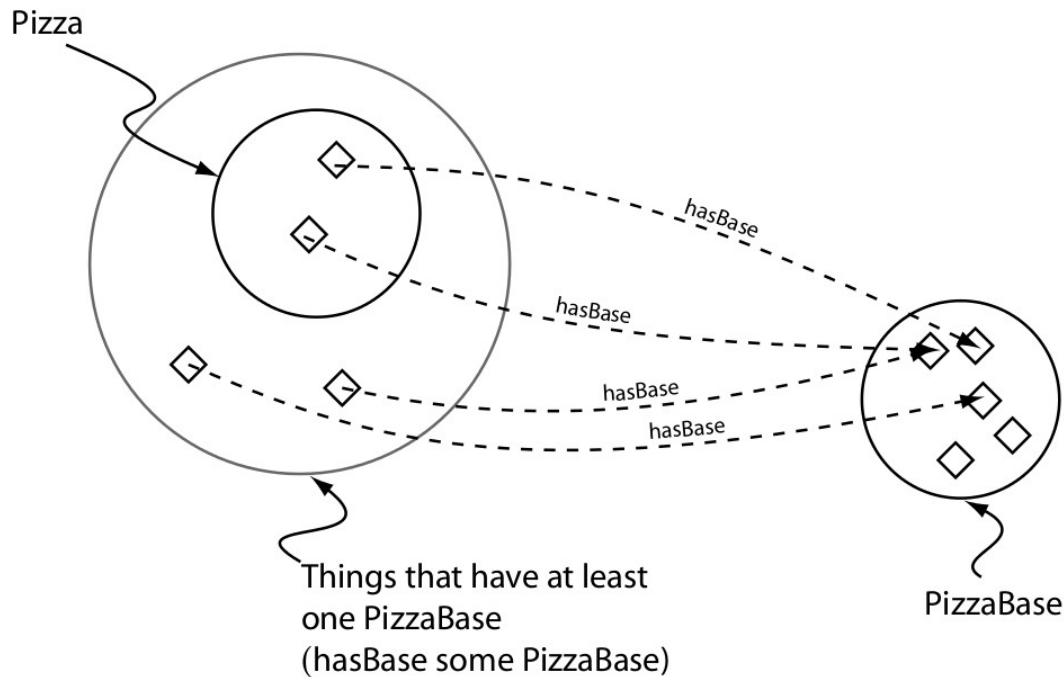


Figure 4.34: A Schematic Description of a Pizza — In order for something to be a Pizza it is necessary for it to have a (at least one) PizzaBase — A Pizza is a subclass of the things that have at least one PizzaBase

(M . Horridge, 2011)

Using the Reasoner (Classifier)

- Using a reasoner we can
 - determine class inconsistencies
 - e.g. inconsistent pizza
 - discovering implicit information
 - using necessary and sufficient conditions
 - e.g. cheesy pizza

Inconsistent Pizza Topping

The screenshot displays the Protégé interface for the ontology `example4.owl`. The left pane shows the class hierarchy, where `ProbeInconsistentTopping` is highlighted. The right pane shows the description of `ProbeInconsistentTopping`, which is a subclass of both `CheeseTopping` and `VegetableTopping`. A comment annotation states: "This class should be inconsistent when the ontology is classified".

Class hierarchy: `ProbeInconsistentTopping`

- Thing
 - Pizza
 - PizzaBase
 - PizzaTopping
 - CheeseTopping
 - MozzarellaTopping
 - ParmezanTopping
 - ProbeInconsistentTopping**
 - MeatTopping
 - SeafoodTopping
 - VegetableTopping
 - CaperTopping
 - MushroomTopping
 - OliveTopping
 - OnionTopping
 - PepperTopping
 - ProbeInconsistentTopping**
 - TomatoTopping

Annotations: `ProbeInconsistentTopping`

Annotations +

comment

This class should be inconsistent when the ontology is classified

Description: `ProbeInconsistentTopping`

Equivalent To +

SubClass Of +

- CheeseTopping
- VegetableTopping

SubClass Of (Anonymous Ancestor)

Members +

Target for Key +

Disjoint With +

Disjoint Union Of +

To use the reasoner click Reasoner->Start reasoner Show Inferences

Inconsistent Pizza Topping

The screenshot shows a web ontology editor interface with several tabs: Active Ontology, Entities, Classes, Object Properties, Data Properties, and Class matrix. The 'Classes' tab is active, showing a class hierarchy on the left and a description panel on the right. A red box in the top right corner contains the text 'example4.owl'.

Class hierarchy (inferred):

- Thing
 - Nothing
 - ProbeInconsistentTopping**
 - Pizza
 - PizzaBase
 - PizzaTopping
 - ValuePartition

Annotations: Nothing

Annotations +

Description: Nothing

Equivalent To +

- ProbeInconsistentTopping**

SubClass Of +

SubClass Of (Anonymous Ancestor)

- hasSpiciness **some** Medium
- hasSpiciness **some** Hot
- hasSpiciness **some** Mild
- hasTopping **some** ParmezanTopping

Reasoner active Show Inferences

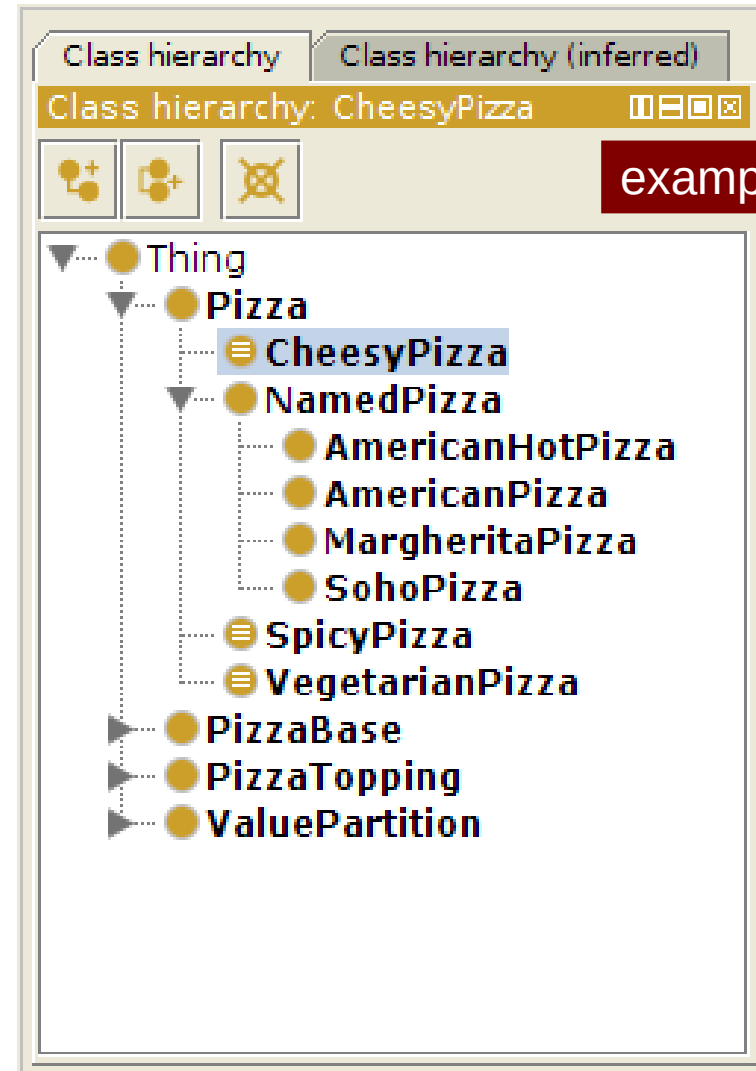
Using the Reasoner (Classifier)

- Using a reasoner we can
 - determine class inconsistencies
 - e.g. inconsistent pizza
 - discovering implicit information
 - using necessary and sufficient conditions
 - e.g. cheesy pizza

Cheesy Pizza

Explicit & Implicit definitions

- NamedPizza and its sub-classes are explicitly defined
- Discover sub-classes of CheesyPizza



Cheesy Pizza

example4.owl

The screenshot displays a web ontology editor interface. The top navigation bar includes tabs for 'Active Ontology', 'Entities', 'Classes', 'Object Properties', 'Data Properties', 'Class matrix', and 'Annotation Properties'. The 'Classes' tab is active, showing a 'Class hierarchy (inferred): Cheesy' tree on the left. The tree shows a hierarchy starting with 'Thing', followed by 'Nothing', 'Pizza', and 'CheesyPizza'. Under 'CheesyPizza', there are subclasses: 'AmericanHotPizza', 'AmericanPizza', 'MargheritaPizza', and 'SohoPizza'. Below these are 'NamedPizza' (with subclasses 'AmericanHotPizza', 'AmericanPizza', 'MargheritaPizza', 'SohoPizza'), 'SpicyPizza' (with subclass 'AmericanHotPizza'), and 'VegetarianPizza' (with subclass 'MargheritaPizza'). Other classes include 'PizzaBase', 'PizzaTopping', and 'ValuePartition'. A red dashed box highlights the 'CheesyPizza' node and its immediate subclasses.

The right pane shows the 'Description: CheesyPizza' section. It contains the following description:

- Equivalent To: **Pizza and (hasTopping some CheeseTopping)**
- SubClass Of: **Pizza**
- SubClass Of (Anonymous Ancestor): **hasBase some PizzaBase**

At the bottom of the editor, the status bar indicates 'Reasoner active' and 'Show Inferences' is checked.

Universal Restrictions

- Intention
 - describe “*all and only values from*” restrictions
- Example
 - a “*vegetarian pizza*”
can only have
cheese or vegetable toppings

Universal Restrictions

example4.owl

The screenshot shows a web ontology editor interface with several tabs: Active Ontology, Entities, Classes, Object Properties, Data Properties, Annotation Properties, and Class matrix. The 'Classes' tab is active, displaying a class hierarchy on the left and a detailed description for 'VegetarianPizza' on the right.

Class hierarchy (inferred):

- Class hierarchy
- Class hierarchy: VegetarianPizz

Class hierarchy:

- Thing
 - Pizza
 - CheesyPizza
 - NamedPizza
 - SpicyPizza
 - VegetarianPizza**
 - PizzaBase
 - PizzaTopping
 - ValuePartition

No Reasoner set. Select a reasoner from the Reasoner menu. Show Inferences

Universal Restrictions

- Run the reasoner
 - expected behaviour:
 - Soho pizza and Margherita pizza should be classified as vegetarian pizzas
 - actual behaviour
 - reasoner does not find any vegetarian pizza subclasses

Open World Assumption

- OWA – What it means:
 - missing information is **not** confirmation of negation
 - in other words:
 - SohoPizza and MargheritaPizza toppings must be explicitly limited to their toppings

SohoPizza:

hasTopping **only** (
MozzarellaTopping
or TomatoTopping
or OliveTopping
or ParmezanTopping)

MargheritaPizza:

hasTopping **only** (
MozzarellaTopping
or TomatoTopping)

Universal Restrictions

example5.owl

The screenshot displays a software interface with two main panels. The left panel, titled 'Class hierarchy (inferred)', shows a tree structure of classes. The right panel, titled 'Annotations: MargheritaPizza', shows the annotations for the selected class.

Class hierarchy (inferred):

- Thing
 - Nothing
 - Pizza
 - CheesyPizza
 - NamedPizza
 - SpicyPizza
 - VegetarianPizza
 - MargheritaPizza**
 - SohoPizza
 - PizzaBase
 - PizzaTopping
 - ValuePartition

Annotations: MargheritaPizza

Annotations +

`comment` [type: string] @ X O

A pizza that only has Mozzarella and Tomato toppings

Description: MargheritaPizza

Equivalent To +

SubClass Of +

- hasTopping only (MozzarellaTopping or TomatoTopping)** ? @ X O
- hasTopping some MozzarellaTopping ? @ X O
- hasTopping some TomatoTopping ? @ X O
- NamedPizza ? @ X O
- CheesyPizza ? @
- VegetarianPizza ? @

SubClass Of (Anonymous Ancestor)

Reasoner active Show Inferences

Working with Protégé

- Protégé is simply an ontology IDE
 - editing
 - visualization
 - validation
- not required but extremely useful for
 - managing large ontologies
 - discovering existing ontologies