

# Construcción de una ontología OWL con Protégé 4.3

Flavio E. Spetale

[spetale@cifasis-conicet.gov.ar](mailto:spetale@cifasis-conicet.gov.ar)

Basado: T. Rodríguez y J. Aguilar

**A Practical Guide To Building OWL  
Ontologies Using Protégé 4**



# Construcción de ontología OWL

- ☛ Las ontologías son usadas para capturar el conocimiento sobre algún dominio de interés.
- ☛ Una ontología describe los conceptos dentro del dominio y la relación que tiene entre esos conceptos.
- ☛ Un lenguaje estándar para hacer ontologías es OWL desarrollado por W3C.
- ☛ OWL permite describir conceptos y además cuenta con un conjunto de operadores (intersección, unión, y negación).
- ☛ OWL esta basado en lógica descriptiva que permite el uso de un razonador.

# ¿Que son las lógicas descriptivas?

- ☛ Son lógicas completas con semántica formal:
  - Fragmentos decidibles de logica de primer orden
  - Estrechamente relacionado con la lógica proposicional/ lógica dinámica
- ☛ Propiedades computacionales bien definidas (peor caso alta complejidad).
- ☛ Amppliamente utilizado para el lenguaje de ontologías.



# Lógica descriptiva: Sintaxis

La sintaxis de un miembro de la familia de la lógica descriptiva se caracteriza por su definición recursiva, en la cual se establecen los constructores que se pueden usar para formar términos conceptuales.

Algunos constructores están relacionados con constructores lógicos en lógica de primer orden (FOL), como intersección o conjunción de conceptos, unión o disyunción de conceptos, negación o complemento de conceptos, restricción universal y restricción existencial.

Otros constructores no tienen una construcción correspondiente en FOL, incluidas las restricciones de funciones, por ejemplo, inversa, transitividad y funcionalidad.

# Componentes de una ontología OWL

**Conceptos** (clases): Animal, Doctor, Perro  
Equivalente a los predicados unarios en FOL

**Relaciones** (propiedades): Tiene.Padre, Ama, Ladra  
Equivalente a los predicados binarios en FOL

**Instancias** (individuos): Shadow, Flavio, Pilar  
Equivalente a constantes en FOL

Ontologías	OWL	PROTÉGÉ
Instancias	Individuos	Casos (instance)
Relaciones	Propiedades	Slots
Conceptos	Clases	Clases

# Individuos de una ontología OWL

Representan objetos del dominio de interés  
y son también conocidos como instancias.





# Propiedades de una ontología OWL

Son relaciones binarias sobre los individuos y pueden ser inversas, transitivas o simétricas.



# Clases de una ontología OWL

Se entienden como conjuntos que contienen individuos y pueden ser organizadas dentro de una jerarquía de clases y subclases conocida como taxonomía.





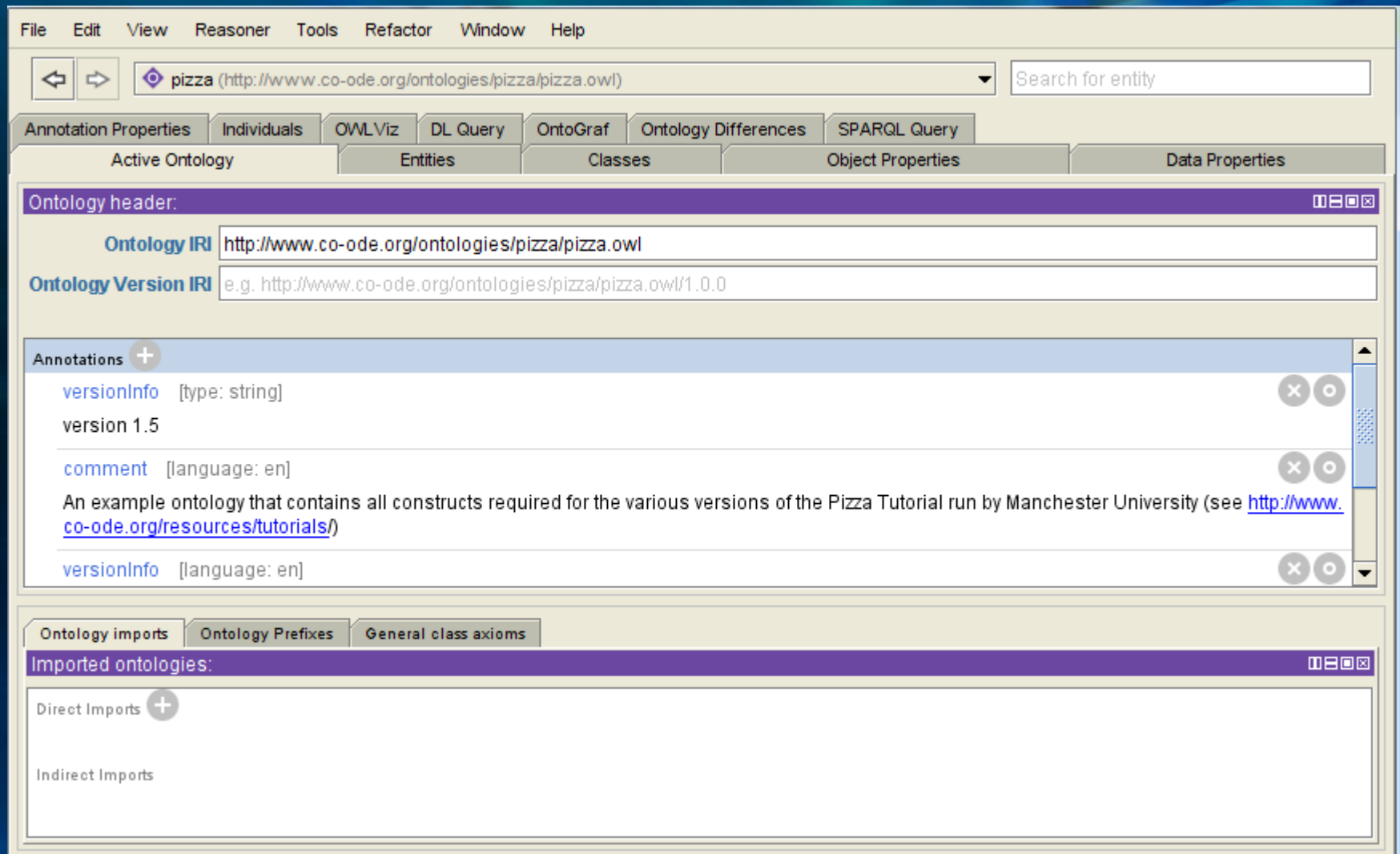
# Correspondencia entre OWL y DL

Constructor OWL	Representación DL	Ejemplo
owl:equivalentTo (C,D)	$C \equiv D (C \sqsubseteq D \text{ y } D \sqsubseteq C)$	<i>Persona</i> $\equiv$ <i>Humano</i>
rdfs:subClassOf (C,D)	$C \sqsubseteq D$	<i>Padres</i> $\sqsubseteq$ <i>Persona</i>
owl:complementOf (C,D)	$C \equiv \neg D (\text{negacion})$	<i>Varon</i> $\equiv \neg$ <i>Mujer</i>
owl:disjointWith (C,D)	$C \sqsubseteq \neg D$	<i>Padre</i> $\sqsubseteq \neg$ <i>Madre</i>
owl:intersectionOf (C,D)	$C \sqcap D (\text{conjuncion})$	<i>Padres</i> $\sqcap$ <i>Varon</i>
owl:unionOf (C,D)	$C \sqcup D (\text{disjuncion})$	<i>Padre</i> $\sqcup$ <i>Madre</i>
owl:oneOf (I1, I2)	$\{I_1\} \sqcup \{I_2\}$	$\{\text{Juan}\} \sqcup \{\text{Maria}\}$
owl:someValuesFrom(P,C)	$\exists P.C (\text{existencial})$	$\exists \text{tieneHijo.Hija}$
owl:allValuesFrom(P,C)	$\forall P.C (\text{universal})$	$\forall \text{tieneHijo.Hijo}$
owl:hasValue (P,I1)	$\exists P.\{I_1\}$	$\exists \text{tieneHijo.}\{\text{Juan}\}$
owl:cardinality(P,n)	$= n.P$	$= 2.\text{tienePadres}$
owl:minCardinality(P,n)	$\geq n.P$	$\geq 1.\text{tieneHija}$
owl:maxCardinality(P,n)	$\leq n.P$	$\leq 2.\text{tieneHijos}$

Un concepto en DL se refiere a una clase en OWL.

Un rol en DL es una propiedad en OWL.

# Interfaz de Protégé



# Comentarios en Protégé

The screenshot displays the Protégé ontology editor interface. The main window shows the 'pizza' ontology with its header information, including the IRI `http://www.co-ode.org/ontologies/pizza` and version `1.5`. The 'Annotations' tab is active, showing the 'comment' property with the value 'An example ontology that contains all constructs recognized by OWL 2.0. This ontology is part of the co-ode.org resources/tutorials/.' The 'comment' property is selected in the left-hand pane, and the 'Value' dialog box is open on the right. The dialog box shows the 'comment' property selected in the list of properties, and the value 'Ontología de pizza que describe varias pizzas según los ingredientes.' is entered in the 'Value' field. The 'Lang' dropdown is set to 'es'.

File Edit View Reasoner Tools Refactor Window

Annotation Properties Individuals OWLViz DL Query

Active Ontology Entities

Ontology header:

Ontology IRI `http://www.co-ode.org/ontologies/pizza`

Ontology Version IRI e.g. `http://www.co-ode.org/ontologies/pizza/1.5`

Annotations +

versionInfo [type: string]  
version 1.5

comment [language: en]  
An example ontology that contains all constructs recognized by OWL 2.0. This ontology is part of the [co-ode.org/resources/tutorials/](http://co-ode.org/resources/tutorials/).

versionInfo [language: en]

Ontology imports Ontology Prefixes General class axioms

Imported ontologies:

Direct Imports +

Indirect Imports

backwardCompatible  
comment  
deprecated  
incompatibleWith  
isDefinedBy  
label  
priorVersion  
seeAlso  
versionInfo

Constant Entity IRI IRI Editor Property values

Value

Ontología de pizza que describe varias pizzas según los ingredientes.

Type Lang es

Aceptar Cancelar



# Clases en Protégé

The screenshot displays the Protégé ontology editor interface. At the top, a menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. Below the menu, a toolbar shows navigation icons and a dropdown menu currently set to 'untitled-ontology-7' with the URL <http://www.semanticweb.org/flavio/ontologies/2014/4/untitled-ontology-7>. A search bar labeled 'Search for entity' is positioned to the right of the dropdown.

The main workspace is divided into several panes. On the left, the 'Class hierarchy' pane shows a tree structure with 'Thing' as the root class. The right side of the workspace is divided into two main sections: 'Annotations' and 'Usage'. The 'Annotations' section shows a list of annotations for the selected class, and the 'Usage' section shows a list of usage information. Below these, the 'Description: Thing' pane is visible, containing a list of properties such as 'SubClass Of', 'SubClass Of (Anonymous)', 'Members', 'Target for Key', 'Disjoint With', and 'Disjoint Union Of', each with a plus icon for expansion.

A modal dialog box titled 'Create a new OWLClass' is open in the foreground. It contains the following fields and controls:

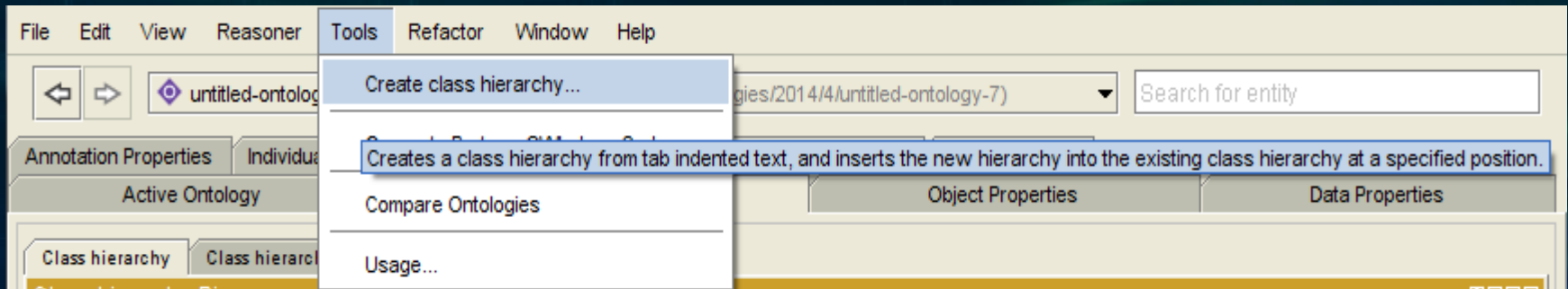
- Name:** A text field containing the word 'Pizza'.
- URI:** A text field containing the URI <http://www.semanticweb.org/flavio/ontologies/2014/4/untitled-ontology-7#Pizza>.
- New entity options...** A button located to the right of the URI field.
- Acceptar** and **Cancelar** buttons at the bottom of the dialog.

At the bottom of the Protégé window, a status bar contains the text 'To use the reasoner click Reasoner->Start reasoner' and a checked checkbox labeled 'Show Inferences'.

# Clases en Protégé

Para generar clases y subclases en un solo paso se utiliza la herramienta:

*Create class hierarchy*



# Clases en Protégé

**Create Class Hierarchy**

**Enter hierarchy**

Please enter the hierarchy that you want to create. You should use tabs to indent names!

Prefix

Suffix

Base

- DelgadaCrujiente
- Gruesa
- Ingredientes

Go Back Continue Cancel

Disjoint With +

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

Background ontology structure:

- Thing
  - Pizza



# Clases en Protégé

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. Below the menu is a toolbar with navigation icons and a dropdown menu showing the current ontology: 'untitled-ontology-7' with its URI. A search bar labeled 'Search for entity' is also present.

The main workspace is divided into several tabs: 'Annotation Properties', 'Individuals', 'OWL Viz', 'DL Query', 'OntoGraf', 'Ontology Differences', and 'SPARQL Query'. Below these are tabs for 'Active Ontology', 'Entities', 'Classes', 'Object Properties', and 'Data Properties'. The 'Classes' tab is currently selected.

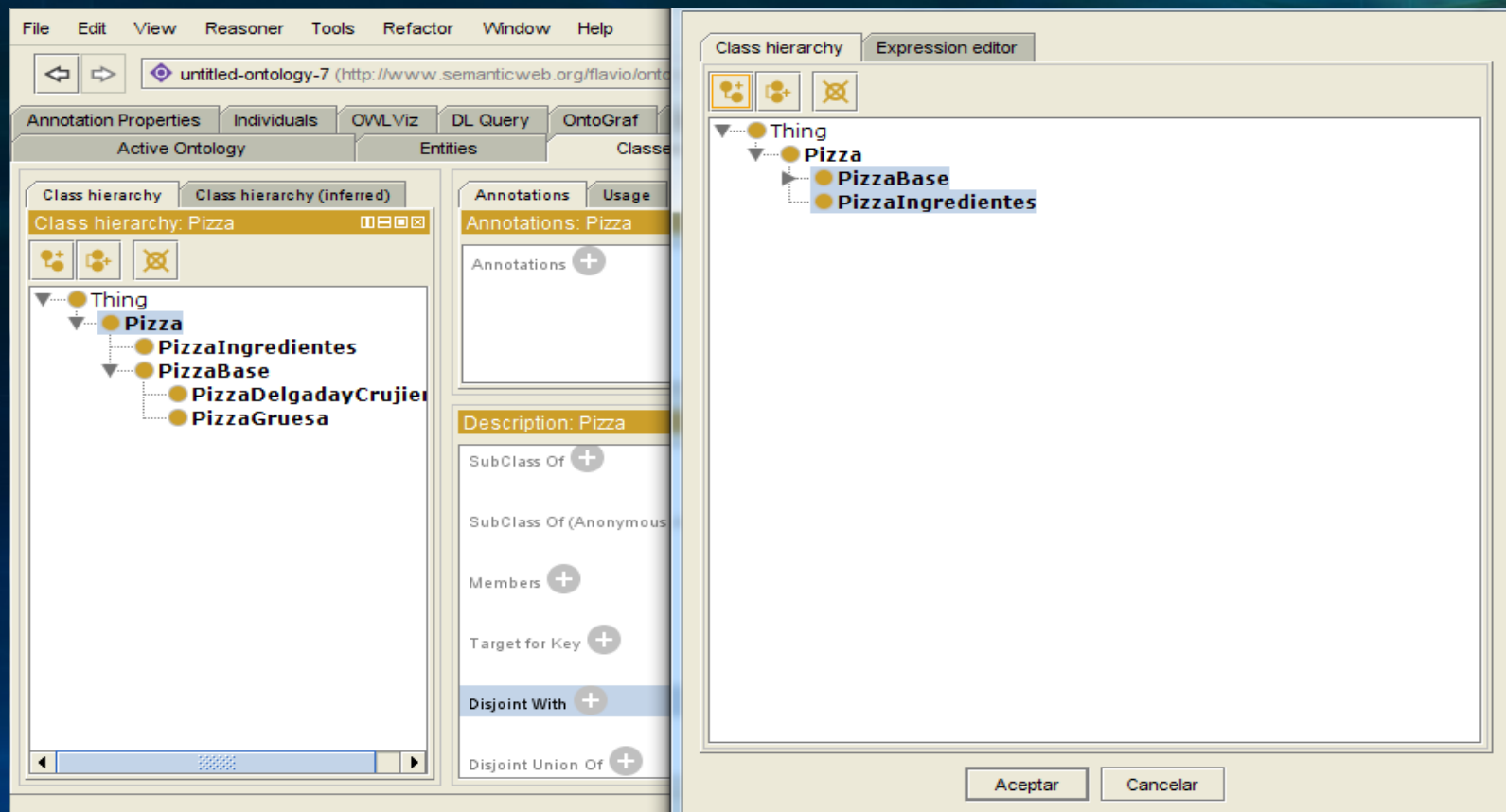
On the left side, the 'Class hierarchy' panel shows a tree structure. The 'Class hierarchy (inferred)' tab is active, displaying a hierarchy starting with 'Thing' as the root. Under 'Thing', there is a class 'Pizza'. Under 'Pizza', there are two subclasses: 'PizzaIngredientes' and 'PizzaBase'. Under 'PizzaBase', there are two subclasses: 'PizzaDelgadayCrujiente' and 'PizzaGruesa'.

On the right side, the 'Annotations' and 'Usage' tabs are visible. The 'Annotations' tab is active, showing a list of annotations for the selected class 'Pizza'. Below this, the 'Description: Pizza' panel is visible, showing a list of properties and their values for the class 'Pizza'. The properties listed are: 'Equivalent To', 'SubClass Of', 'SubClass Of (Anonymous Ancestor)', 'Members', 'Target for Key', and 'Disjoint With'. Each property has a plus sign icon next to it, indicating that it can be edited or expanded.

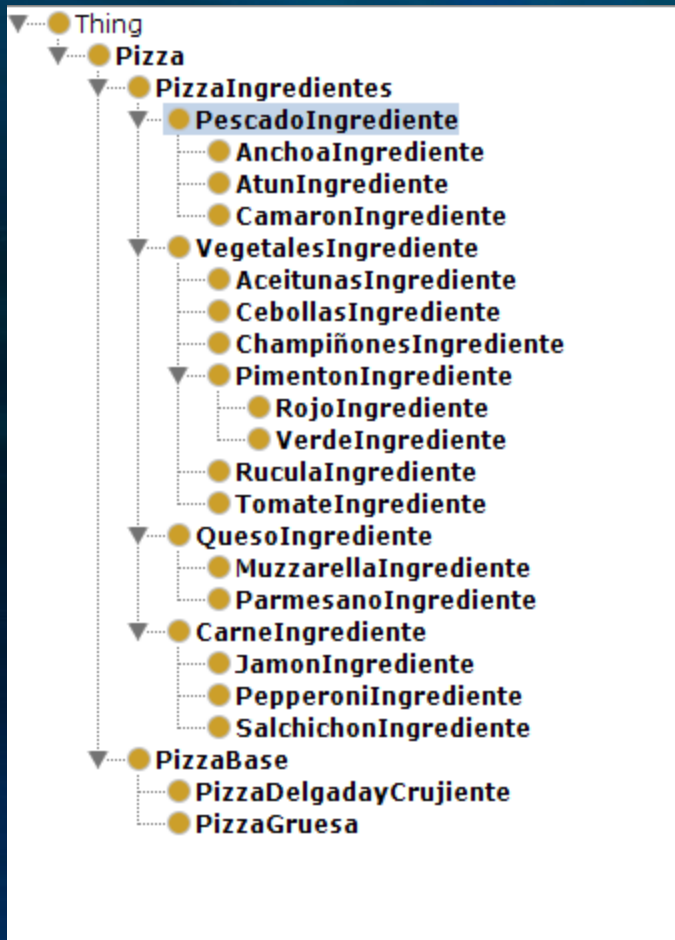
At the bottom of the interface, there is a status bar with the text: 'To use the reasoner click Reasoner->Start reasoner' and a checkbox labeled 'Show Inferences' which is currently checked.

# Clases Disjuntas en Protégé

Después de adicionar varias clases en la jerarquía de la ontología, se requiere establecer clases disjuntas, que indican que un objeto o individuo no puede ser instancia de más de una de estas clases que se establecieron disjuntas.



# Clases Disjuntas en Protégé



Description: PescadoIngrediente

Equivalent To +

SubClass Of +

- PizzaIngredientes**

SubClass Of (Anonymous Ancestor)

Members +

Target for Key +

Disjoint With +

- CarneIngrediente, QuesoIngrediente, VegetalesIngrediente**

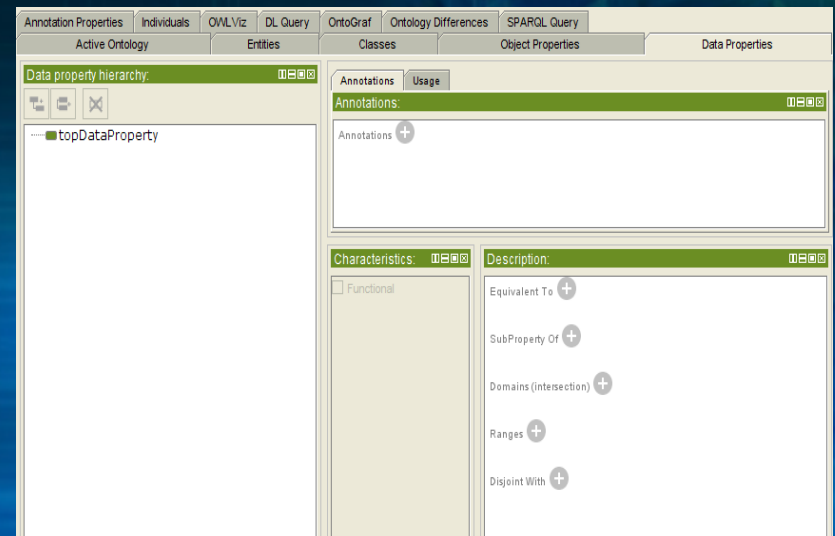
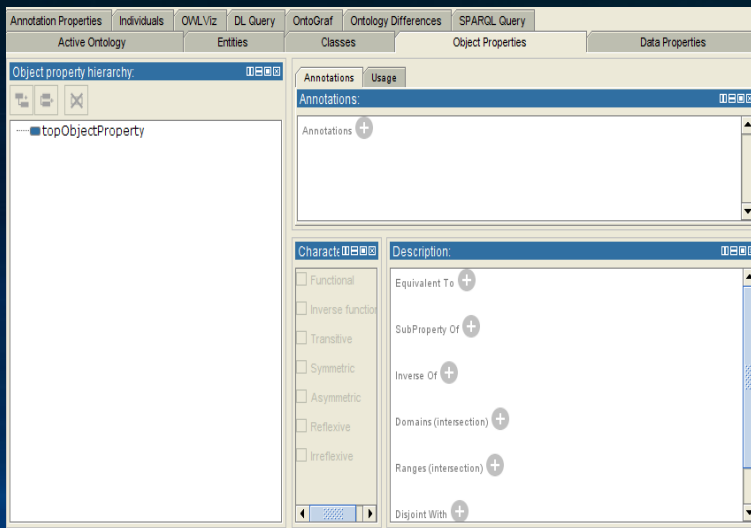
Disjoint Union Of +



# Propiedades OWL en Protégé

Existen dos tipos de propiedades en OWL:

- ✿ “ObjectProperties”, que permite relacionar un individuo con otro
- ✿ “DatatypeProperties”, que relaciona un individuo con un XML Schema Datatype value o un literal RDF



# Propiedades OWL en Protégé

Los elementos que debe tener un ObjectProperty son:

**Nombre**

**Dominio:** hace referencia a la clase o clases iniciales

**Rango:** hace referencia a la clase o clases finales.

Ejemplo la relación: es profesor

Nombre: es\_profesor

Dominio: Docente

Rango: Estudiante

# Propiedades OWL en Protégé

The screenshot displays the Protégé OWL editor interface with the following components:

- Top Navigation Bar:** Includes tabs for Annotation Properties, Individuals, OWLViz, DL Query, OntoGraf, Ontology Differences, and SPARQL Query.
- Left Panel (Object property hierarchy: TieneBase):** Shows a tree structure where 'TieneBase' is a subclass of 'TieneIngredientes', which is a subclass of 'Tiene\_Ingrediente'.
- Right Panel (Annotations: TieneBase):** Contains a section for adding annotations to the 'TieneBase' class.
- Bottom Left Panel (Characteristics):** A list of checkboxes for defining class characteristics: Functional, Inverse function, Transitive, Symmetric, Asymmetric, Reflexive, and Irreflexive.
- Bottom Right Panel (Description: TieneBase):** A section for defining the class description, including fields for 'Equivalent To', 'SubProperty Of' (currently set to 'Tiene\_Ingrediente'), 'Inverse Of', 'Domains (intersection)', and 'Ranges (intersection)'.



# Propiedades Inversas en Protégé

Cada **ObjectProperty** debe tener su correspondiente propiedad inversa.

Si una propiedad enlaza un objeto A con otro B, entonces la propiedad inversa enlaza el objeto B con el A.

The screenshot displays the Protégé ontology editor interface. The top navigation bar includes tabs for 'Annotation Properties', 'Individuals', 'OWL Viz', 'DL Query', 'OntoGraf', 'Ontology Differences', and 'SPARQL Query'. Below this, the 'Active Ontology' section shows 'untitled-ontology-7' with its URI. The main workspace is divided into several panes:

- Object property hierarchy: EsIngredientesDe:** A tree view showing the hierarchy of object properties. The 'EsIngredientesDe' property is highlighted, showing its subproperties: 'EsIngredientesDe', 'EsBaseDe', 'Tiene\_Ingrediente', 'TieneIngredientes', and 'TieneBase'.
- Annotations: EsIngredientesDe:** A pane for adding annotations to the selected property.
- Characteristics:** A list of checkboxes for property characteristics: Functional, Inverse function, Transitive, Symmetric, Asymmetric, Reflexive, and Irreflexive.
- Description: EsIngredientesDe:** A pane for defining the property's characteristics. It includes sections for 'Equivalent To', 'SubProperty Of' (with 'EsIngredienteDe' selected), 'Inverse Of', 'Domains (intersection)', and 'Ranges (intersection)'.

The right-hand pane shows the 'EsIngredientesDe' property's details, including its domain and range, and a list of its subproperties: 'EsIngredienteDe', 'EsBaseDe', 'Tiene\_Ingrediente', 'TieneBase', and 'TieneIngredientes'.

# Características de las propiedades en Protégé

OWL permite que el significado de las propiedades sea enriquecido con características de las propiedades

- ☛ Funcional
- ☛ Funcional Inversa
- ☛ Simétrica
- ☛ Transitiva

# Propiedad Funcional en Protégé

Define que a lo sumo un objeto puede estar relacionado con otro objeto.

Ejemplo, si se tienen tres objetos que son A, B y C y se tiene una propiedad funcional *tienePadre*, entonces se podrían asociar los objetos A y B por medio de la propiedad y daría como resultado A *tienePadre* B. Igualmente se podrían asociar los objetos A y C por medio de la propiedad y daría como resultado A *tienePadre* C. Como *tienePadre* es propiedad funcional, *se concluye que B y C son el mismo objeto*. En caso contrario estaríamos en una contradicción.



# Propiedad Funcional Inversa en Protégé

Indica que puede estar a lo sumo un objeto relacionado con otro.

Ejemplo, si se tienen tres objetos que son A, B y C y se tiene una propiedad funcional **esPadreDe**, entonces se podría asociar el objeto B y A por medio de la propiedad y daría como resultado B **esPadreDe** A. Igualmente se podrían asociar los objetos C y A por medio de la propiedad y daría como resultado C **esPadreDe** A. Como **esPadreDe** es propiedad funcional inversa, se concluye que B y C son el mismo objeto.



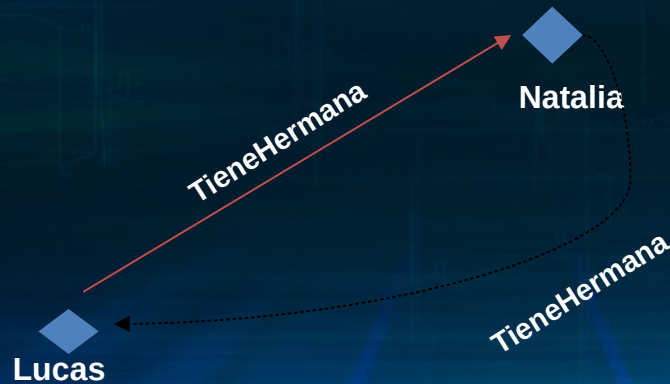
# Propiedad Transitiva en Protégé

Relaciona dos objetos A y B, y además hay una propiedad que relaciona al objeto B con otro C, entonces se puede inferir que el objeto A está relacionado con el objeto C mediante la propiedad transitiva.



# Propiedad Simétrica en Protégé

Relaciona a los objetos A y B, entonces el objeto B es relacionado por medio de la propiedad P con el objeto A.



# Propiedades en Protégé

This screenshot shows the Protégé interface with the 'Tiene\_Ingridiente' object property selected. The left pane displays the 'Object property hierarchy' for 'Tiene\_Ingridiente', showing a tree structure: topObjectProperty, EsIngridienteDe, Tiene\_Ingridiente, TieneBase, and TieneIngredientes. The right pane shows the 'Annotations' and 'Usage' tabs for 'Tiene\_Ingridiente'. The 'Character' tab is active, showing the 'Description: Tiene\_Ingridiente' and a list of properties: Functional, Inverse function, Transitive, Symmetric (checked), Asymmetric, Reflexive, and Irreflexive. The 'Domains (intersection)' and 'Ranges (intersection)' sections are also visible.

This screenshot shows the Protégé interface with the 'TieneBase' object property selected. The left pane displays the 'Object property hierarchy' for 'TieneBase', showing a tree structure: topObjectProperty, EsIngridienteDe, Tiene\_Ingridiente, TieneBase, and TieneIngredientes. The right pane shows the 'Annotations' and 'Usage' tabs for 'TieneBase'. The 'Character' tab is active, showing the 'Description: TieneBase' and a list of properties: Functional (checked), Inverse function, Transitive, Symmetric, Asymmetric, Reflexive, and Irreflexive. The 'Domains (intersection)' and 'Ranges (intersection)' sections are also visible. The 'Inverse Of' section shows a relationship with 'EsBaseDe'.

# Rango de una Propiedad en Protégé

The screenshot displays the Protégé ontology editor interface. The top navigation bar includes tabs for Annotation Properties, Individuals, OWL Viz, DL Query, OntoGraf, Ontology Differences, and SPARQL Query. Below this, the 'Active Ontology' section shows the 'TieneBase' ontology. The 'Object Properties' tab is selected, showing the 'Object property hierarchy: TieneBase'. The hierarchy includes 'topObjectProperty', 'EsIngredienteDe', 'Tiene\_Ingrediente', 'TieneBase', and 'TieneIngredientes'. The 'TieneBase' property is highlighted.

The 'TieneBase' dialog box is open, showing the 'Object restriction creator' and 'Data restriction creator' tabs. The 'Class expression editor' is active, displaying a hierarchy with 'Thing', 'Pizza', 'PizzaBase', and 'PizzaIngredientes'. The 'Aceptar' (Accept) and 'Cancelar' (Cancel) buttons are at the bottom.

The 'Character' panel on the right shows the following options:

- ☒ Functional
- ☐ Inverse function
- ☐ Transitive
- ☐ Symmetric
- ☐ Asymmetric
- ☐ Reflexive
- ☐ Irreflexive

The 'Description: TieneBase' panel shows the following configuration:

- SubProperty Of: **Tiene\_Ingrediente**
- Inverse Of: **EsBaseDe**
- Domains (intersection):
- Ranges (intersection):
- Disjoint With:



# Dominio de una Propiedad en Protégé

The screenshot displays the Protégé ontology editor interface. The top menu bar includes 'Annotation Properties', 'Individuals', 'OWL Viz', 'DL Query', 'OntoGraf', 'Ontology Differences', and 'SPARQL Query'. Below this, a secondary bar shows 'Active Ontology', 'Entities', 'Classes', 'Object Properties', and 'Data Properties'. The 'Object Properties' tab is active, showing the 'Object property hierarchy: TieneBase' on the left. This hierarchy includes 'topObjectProperty', 'EsIngredienteDe', 'EsBaseDe', 'EsIngredientesDe', 'Tiene\_Ingrediente', 'TieneBase', and 'TieneIngredientes'. The 'TieneBase' property is selected. The right pane shows the 'Annotations' and 'Usage' tabs for 'TieneBase'. The 'Annotations' tab is active, showing a list of annotations. The bottom-left pane shows the 'TieneBase' class expression editor, with the 'Class hierarchy' tab active, displaying a hierarchy starting from 'Thing' and including 'Pizza'. The bottom-right pane shows the 'Character' and 'Description' tabs for 'TieneBase'. The 'Character' tab is active, showing a list of character properties: 'Functional' (checked), 'Inverse function', 'Transitive', 'Symmetric', 'Asymmetric', 'Reflexive', and 'Irreflexive'. The 'Description' tab is also visible, showing the 'Subproperty of' section with 'Tiene\_Ingrediente' and the 'Inverse Of' section with 'EsBaseDe'. The 'Domains (intersection)' section is highlighted, showing 'PizzaBase'. The 'Ranges (intersection)' section is also visible, showing 'PizzaBase'. The bottom status bar indicates 'To use the reasoner click Reasoner->Start reasoner' and 'Show Inferences' (checked).

Object property hierarchy: TieneBase

- topObjectProperty
  - EsIngredienteDe
    - EsBaseDe
    - EsIngredientesDe
  - Tiene\_Ingrediente
    - TieneBase
    - TieneIngredientes

TieneBase

Class expression editor | Class hierarchy | Object restriction creator | Data restriction creator

- Thing
  - Pizza

Aceptar Cancelar

Character

- ☒ Functional
- ☐ Inverse function
- ☐ Transitive
- ☐ Symmetric
- ☐ Asymmetric
- ☐ Reflexive
- ☐ Irreflexive

Description: TieneBase

Subproperty of

- Tiene\_Ingrediente

Inverse Of

- EsBaseDe

Domains (intersection)

- PizzaBase

Ranges (intersection)

- PizzaBase

Disjoint With

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

# Restricciones de una Propiedad en Protégé

Las propiedades son utilizadas para crear restricciones en las clases en una ontología OWL.

Usualmente el nombre de la propiedad debería sugerir las restricciones impuestas a los objetos de la clase.

Las restricciones OWL se presentan en las siguientes tres categorías:

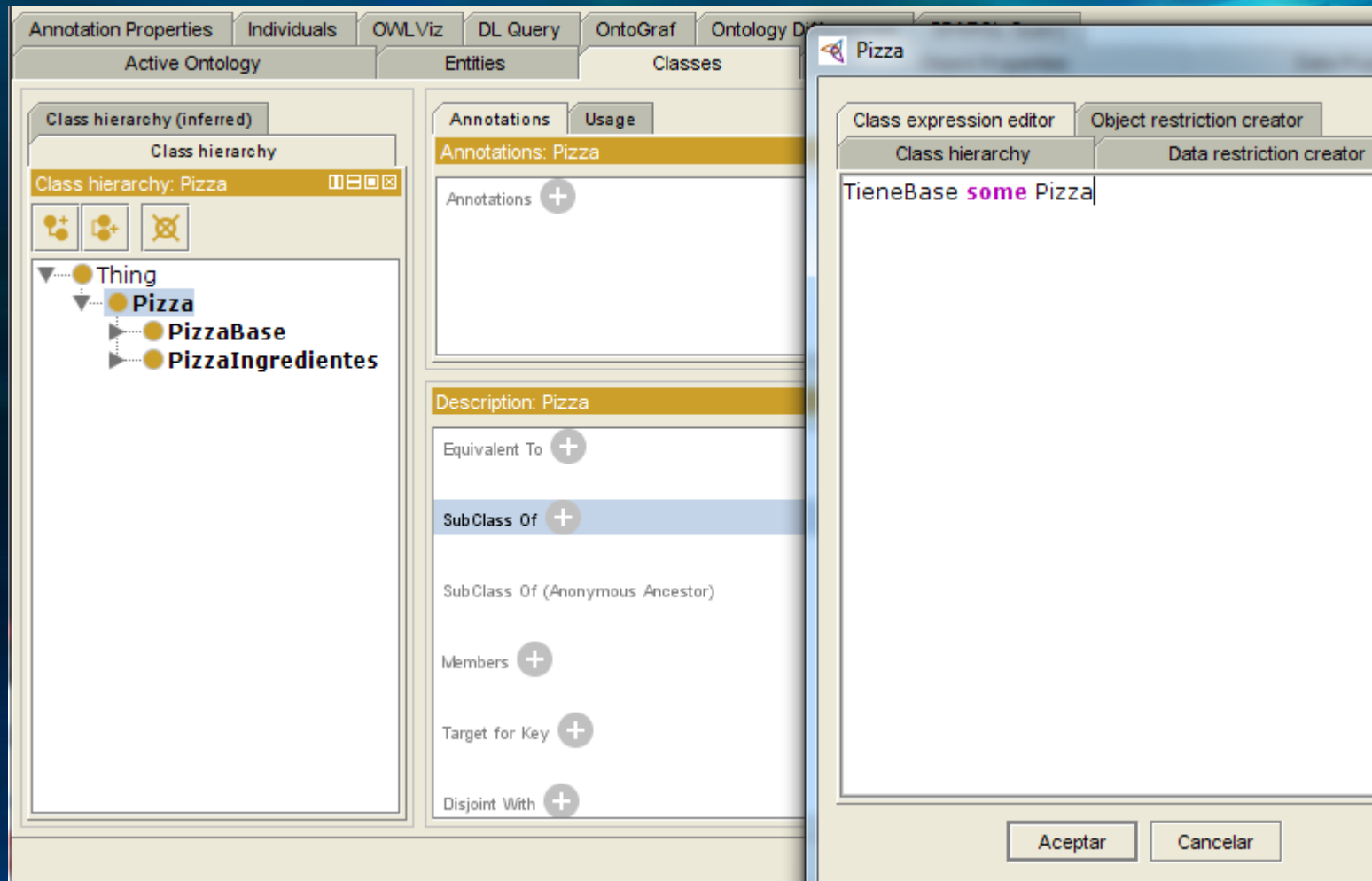
- ✿ Restricciones de cuantificación.
- ✿ Restricciones de cardinalidad.
- ✿ Restricciones de valor.

# Restricciones de una Propiedad en Protégé

Cuantificador existencial ( $\exists$ ), el cual permite indicar la existencia de al menos un objeto. En Protégé 4 la palabra clave **some** es usado para denotar  $\exists$ .

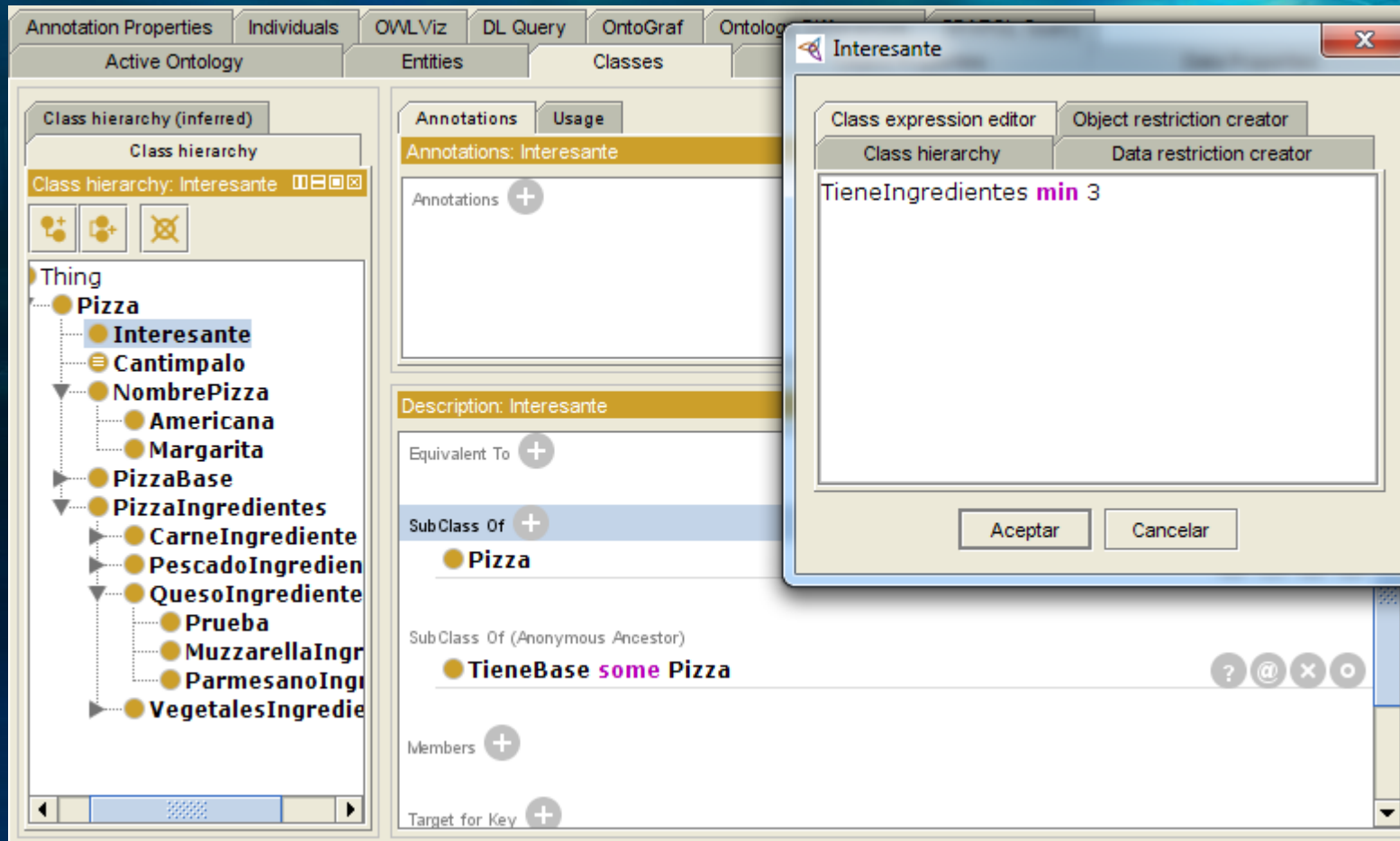
Cuantificador universal ( $\forall$ ), el cual permite indicar la existencia de todos los objetos. En Protégé 4. la palabra clave es **only** es usado para denotar  $\forall$ .

# Restricciones de una Propiedad en Protégé

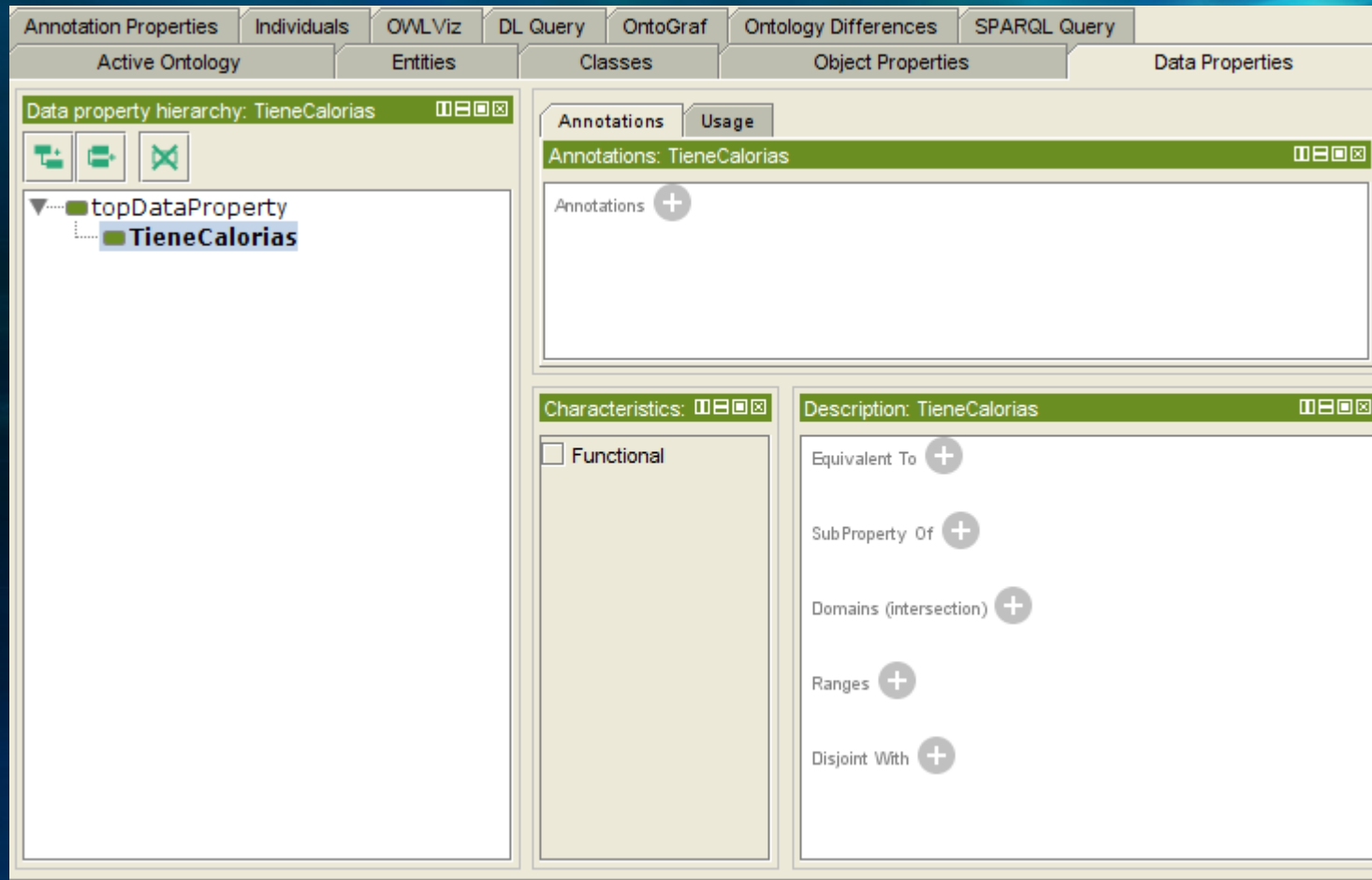




# Restricción de Cardinalidad en Protégé



# Propiedad Datatype en Protégé



# Propiedad Datatype en Protégé

The screenshot displays the Protégé ontology editor interface. The top menu bar includes options like Annotation Properties, Individuals, OWL Viz, DL Query, OntoGraf, Ontology Differences, and SPARQL Query. The main workspace is divided into several panes:

- Class hierarchy (inferred):** Shows a tree structure starting from 'Thing' and branching into 'Pizza', 'Interesante', 'Cantimpalo', 'NombrePizza', 'Americana', 'Margarita', 'PizzaBase', 'PizzaIngredientes', 'CarneIngredientes', 'PescadoIngredientes', 'QuesoIngredientes', 'Prueba', 'MuzzarellaIngredientes', 'ParmesanoIngredientes', and 'VegetalesIngredientes'.
- Annotations: Pizza:** A pane for adding annotations to the 'Pizza' class.
- Description: Pizza:** A pane for defining the class description, showing 'Equivalent To', 'SubClass Of' (with 'TieneBase' as a child), 'SubClass Of (Anonymous Ancestor)', 'Members', and 'Target for Key'.
- Pizza (Class expression editor):** A pane for editing the class expression, showing 'Restricted property' (with 'topDataProperty' and 'TieneCalorias' as children) and 'Restriction type' (set to 'Some (existential)').
- Data restriction creator:** A pane for creating data restrictions, showing 'Restriction filler' (with a list of datatypes including 'integer', 'language', 'Literal', 'long', 'Name', 'NCName', 'negativeInteger', and 'NMToken') and 'Cardinality' (set to 1).

The 'Pizza' class is currently selected, and the 'TieneCalorias' property is being configured as a data property with a 'Some (existential)' restriction and a cardinality of 1. The 'Aceptar' (Accept) and 'Cancelar' (Cancel) buttons are visible at the bottom right.

# Propiedad Datatype en Protégé

The screenshot displays the Protégé ontology editor interface. On the left, the 'Class hierarchy (inferred)' panel shows a tree structure starting with 'Thing', followed by 'Pizza', 'Interesante', 'Cantimpalo', 'NombrePizza', 'PizzaBase', 'PizzaIngrediente', and a list of ingredients: 'CarneIngrediente', 'PescadoIngrediente', 'QuesoIngrediente', 'Muzzarela', 'Parmesano', 'Prueba', and 'VegetalesIngrediente'. The 'Individuals: NewCantimpalo' panel is active, showing a single individual 'NewCantimpalo'. A dialog box titled 'NewCantimpalo' is open, showing the 'Data Property' tab. The 'topDataProperty' is set to 'TieneCalorias'. The 'Value' field contains '200' and the 'Type' is set to 'integer'. The 'Aceptar' button is highlighted. Below the dialog, the 'Description: NewCantimpalo' panel shows the type 'Cantimpalo' and the 'Property assertions: NewCantimpalo' panel shows the 'Data property assertions' section.

untitled-ontology-7 (http://www.semanticweb.org/flavio/ontology/)

File Edit View Reasoner Tools Refactor Window Help

Annotation Properties Individuals OWLViz DL Query Ontology

Active Ontology Entities Classes

Class hierarchy (inferred)

Class hierarchy

Class hierarchy: Interesting 11 11 11 11

Thing

Pizza

Interesante

Cantimpalo

NombrePizza

PizzaBase

PizzaIngrediente

CarneIngrediente

PescadoIngrediente

QuesoIngrediente

Muzzarela

Parmesano

Prueba

VegetalesIngrediente

Individuals: NewCantimpalo 11 11 11 11

NewCantimpalo

Data Property

topDataProperty

TieneCalorias

Value

200

Type integer

Aceptar Cancelar

Description: NewCantimpalo 11 11 11 11

Types +

Cantimpalo

Same Individual As +

Different Individuals +

Property assertions: NewCantimpalo 11 11 11 11

Object property assertions +

Data property assertions +

Negative object property assertions +

Negative data property assertions +



# Razonadores en Protégé

Una base de conocimiento (KB) se compone de un TBox y un ABox.

Un TBox describe conocimiento intencional en la forma de conceptos (clases) y definiciones de roles (propiedades).

Un ABox describe conocimiento por extensión y consiste de un conjunto finito de aserciones acerca de los individuos mientras utiliza los términos de la ontología. Conviene notar que un ABox representa un conocimiento incompleto acerca del mundo.

Las clases pueden ser organizadas en una jerarquía de superclases-subclases, conocido como *taxonomía*.

Estas relaciones pueden ser procesadas por un razonador en OWL-DL. Virtualmente toda consulta a una ontología OWL DL debe ser realizada utilizando un razonador que deduzca conocimiento implícito.

Los razonadores pueden ser agrupados en dos categorías: razonadores de lógica descriptiva y razonadores de programación lógica.

# Razonadores en Protégé

Reasoning Characteristics

	CB	CEL	FaCT++	HermiT	Pellet	RP	SR	TrOWL (REL)
Methodology	consequence-based	completion rules	tableau-based	hypertableau	tableau-based	tableau-based	completion rules	approximation (completion rules)
Soundness	+	+	+	+	+	+	+	+
Completeness	+	+	+	+	+	+	+	-
Expressivity	Horn <i>SHIF</i>	$\mathcal{EL}^+$	<i>SROIQ(D)</i>	<i>SROIQ(D)</i>	<i>SROIQ(D)</i>	<i>SHIQ(D-)</i>	$\mathcal{EL}^+$	third-party reasoner (approximating SROIQ; subset of $\mathcal{EL}^{++}$ )
Incremental Classification (addition/removal)	-/-	+/-	-/-	-/-	+/+	-/-	+/-	-/-
Rule Support	-	-	-	+(SWRL)	+(SWRL)	+(SWRL, nRQL)	-	-
Justifications	-	+	-	-	+	+	-	-
ABox Reasoning	-	+	+	+	+(SPARQL)	+(SPARQL, nRQL)	-	+(SPARQL)

# Razonadores l gica descriptiva

Los razonadores DL brindan los siguientes servicios de inferencia:

- ✕ Validaci n de la consistencia de una ontolog a: el razonador puede comprobar si una ontolog a no contiene hechos contradictorios
- ✕ Validaci n del cumplimiento de los conceptos de la ontolog a: el razonador determina si es posible que una clase tenga instancias. En el caso de que un concepto no sea satisfecho la ontolog a ser  inconsistente.
- ✕ Clasificaci n de la ontolog a: el razonador computa a partir de los axiomas declarados en el TBox, las relaciones de subclase entre todos los conceptos declarados expl citamente a fin de construir la jerarqu a de clases.



# Razonadores Lógica descriptiva

Los razonadores DL brindan los siguientes servicios de inferencia:

- × Posibilita la resolución de consultas durante la recuperación de información basada en ontologías: a partir de la jerarquía de clases se pueden formular consultas como conocer todas las subclases de un concepto, inferir nuevas subclases de un concepto, las superclases directas, etc.
- × Precisiones sobre los conceptos de la jerarquía: el razonador puede inferir cuáles son las clases a las que directamente pertenece y mediante la jerarquía inferida obtener todas las clases a las cuales indirectamente pertenece una clase o individuo dentro de la ontología.



# HermiT

HermiT es uno de los actuales proyectos de investigación del Laboratorio de Computación de la Universidad de Oxford.

Aunque puede ser empleado con cualquier ontología, los investigadores toman como punto de partida los requerimientos de ontologías médicas.

HermiT es un razonador para ontologías escritas empleando OWL y construido empleando cálculo hypertableau a fin de proveer razonamiento más eficiente.

Se anuncia como un razonador veloz y capaz de resolver ontologías complejas.

# Pellet

Pellet fue desarrollado por el laboratorio Mindswap de la Universidad de Maryland (USA).

Entre sus características se destacan que admite la expresividad completa de OWL DL, razonamiento acerca de nominales (clases enumeradas o definidas por extensión), absorción, ramificación semántica y fue extendido para soportar las nuevas características propuestas en OWL 1.1 y OWL 2.

Pellet trata de un razonador DL basado sobre algoritmos tableaux. El núcleo del razonador, es el algoritmo tableaux (desarrollado para lógicas descriptivas potentes) el cual verifica la consistencia de la KB, es decir el par ABox y Tbox.

# Pellet

Esencialmente un razonador tableaux posee sólo la funcionalidad de verificar la satisfactibilidad de un ABox con respecto a un TBox.

Este razonador implementa las mejores técnicas de optimización, lo que hace que su desempeño sea bueno, en especial cuando debe evaluar ontologías con mayor complejidad y expresividad; sin embargo no es tan eficiente como RacerPro o FACT++ en clasificaciones.



# FaCT++

FaCT++, desarrollado por la Universidad de Manchester bajo el proyecto europeo “WonderWeb”, es un razonador DL basado en el algoritmo tableaux para lógica descriptiva. Cubre los lenguajes de ontología OWL y OWL2.

FaCT++ es un buen razonador para la TBox de una ontología, sin embargo, carece de soporte para otros tipos de datos que no sean string o integer (como si ocurre con Pellet) y tampoco posee soporte para razonamiento con la ABox de una ontología.

Entre sus características se destaca:

- 1) trabaja de forma eficiente con TBox de ontologías de tamaño grande y mediano
- 2) es posible utilizar el lenguaje de consultas para RDF.



Mediante el algoritmo tableaux implementa un procedimiento de decisión para TBox y ABox.

También implementa nuevas características y optimizaciones, que permite personalizar para adicionar nuevas tácticas de razonamiento y la capacidad de razonar con lógicas descriptivas más potentes y cercanas a la expresividad de OWL-DL.

# Clase inconsistente en Protégé

Clasificando con el razonador Pellet, Fact++ o Hermit

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The Reasoner menu is open, showing options: Start reasoner (Ctrl-R), Synchronize reasoner, Stop reasoner, Explain inconsistent ontology, Configure..., a separator, FaCT++, Hermit 1.3.8, Pellet (selected), Pellet (Incremental), and None. The left pane shows the Class hierarchy (inf) with a tree structure: Thing (expanded) -> Pizza (expanded) -> PizzaAmericanaHot, PizzaAmericana, PizzaMargarita, PizzaTopping (expanded) -> CarneTopping, PescadoTopping, QuesoTopping (expanded) -> PruebaIncoTopping (highlighted in red), MozzarellaTopping, ParmesanoTopping, VegetablesTopping, PizzaBase (expanded) -> BaseDelgadayCrujiente, BaseGruesa. The right pane shows the Class Annotations and Class Usage tabs. The Annotations: Thing tab is active, displaying a list of annotations. The Description: Thing tab is also visible, showing sections for Equivalent classes, Superclasses, Inferred anonymous superclasses, Members, and Disjoint classes.

# Condición necesaria y suficiente en Protégé

The screenshot displays the Protégé ontology editor interface. The top navigation bar includes tabs for Annotation Properties, Individuals, OWL Viz, DL Query, OntoGraf, Ontology Differences, and SPARQL Query. Below this, a secondary bar shows Active Ontology, Entities, Classes, Object Properties, and Data Properties.

The left pane, titled 'Class hierarchy (inferred)', shows a tree structure of classes. The 'Cantimpalo' class is selected, and its hierarchy is visible, including 'Pizza', 'NombrePizza', 'Americana', 'Margarita', 'PizzaBase', 'PizzaIngredientes', 'CarneIngrediente', 'PescadoIngredien', 'QuesoIngrediente', 'Prueba', 'MuzzarellaIngr', 'ParmesanoIngr', and 'VegetalesIngredie'.

The right pane, titled 'Annotations' and 'Usage', shows the 'Annotations: Cantimpalo' section. Below this, the 'Description: Cantimpalo' section displays the following logical axioms:

- Equivalent To:  $\text{Pizza} \text{ and } \text{TieneIngredientes some QuesoIngrediente}$
- Sub Class Of:  $\text{TieneBase some Pizza}$

The interface also includes a 'Members' section and a 'Target for Key' section, both with a '+' icon to add new entries.