

TALLER No. 4A

OWL AVANZADO

Profesor: Jaime Alberto Guzmán Luna

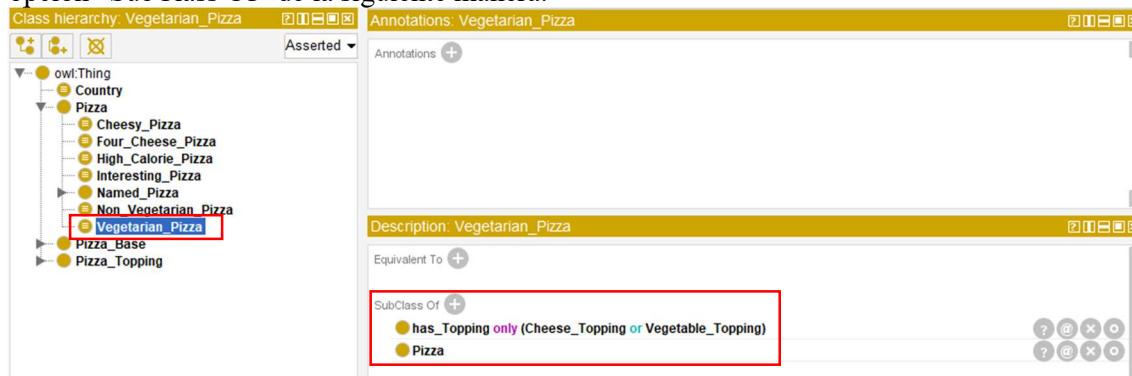
Contenido del taller:

1. OWL Avanzado

Retome el archivo de la ontología del taller 3...

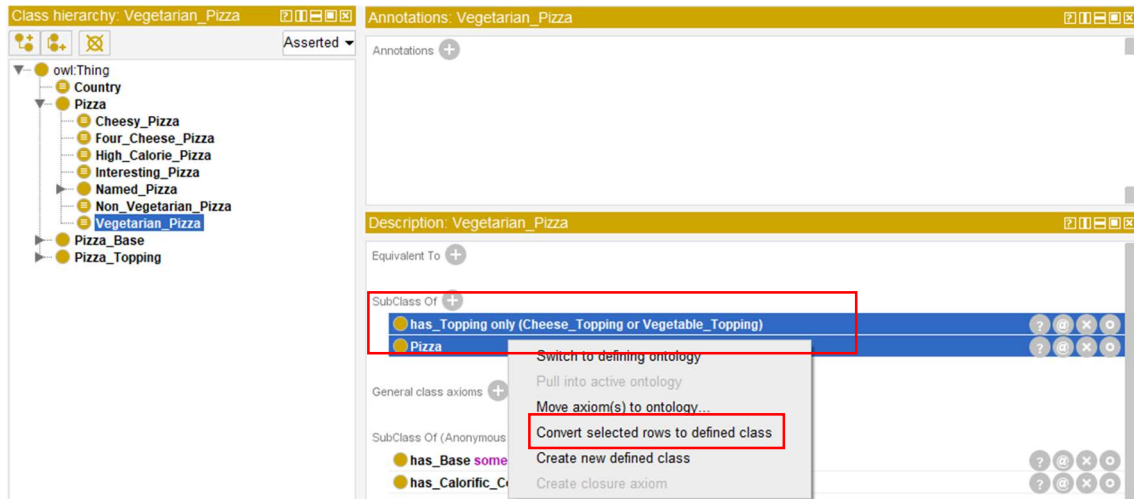
Restricciones universales:

Hasta el momento hemos utilizado el cuantificador existencial. Existe también el cuantificador universal (\forall) utilizado para restringir las relaciones para una propiedad dada a los individuos que son miembros de una clase específica. Por ejemplo, la restricción universal \forall has Topping Mozzarella Topping, describe los individuos que tienen todas las propiedades has Topping con miembros de la clase Mozzarella Topping. Supongamos que queremos crear la clase Vegetarian_Pizza. Los individuos que son miembros de la clase Vegetarian_Pizza deben tener únicamente Cheese_Topping o Vegetable_Topping. Entonces agregamos una restricción por la opción “SubClass Of” de la siguiente manera:

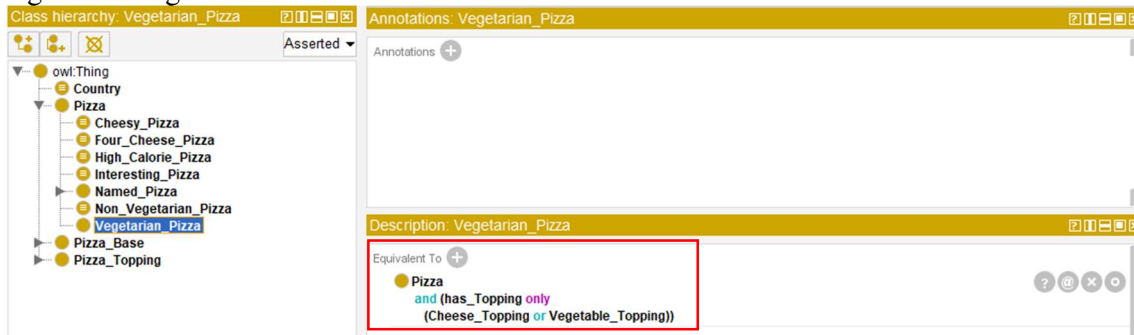


Si analizamos bien, encontramos que entre las pizzas nombradas hay algunas que cumplen con ser pizzas vegetarianas ya que tienen queso y vegetales, por ejemplo, la Soho_Pizza tiene Mozzarella_Topping, Olive_Topping, Parmesan_Topping y Tomato_Topping. También Margherita_Pizza que tiene Mozzarella_Topping y Tomato_Topping. Pero vemos que el razonador NO ha clasificado a Margherita_Pizza y Soho-Pizza como subclases de Vegetarian_Pizza. ¿Por qué? Esto se debe al razonamiento del mundo abierto. Así, las definiciones de Soho_Pizza y de Margherita_Pizza están incompletas, realmente no son definiciones sino descripciones, es decir solamente están definidas con condiciones necesarias. Ahora bien, podemos convertir la descripción de la clase Vegetarian_Pizza en una definición. Para ello seleccionamos las dos restricciones (usando la tecla Ctrl + Click) y ubicando el cursor sobre las restricciones marcadas, damos click derecho. Aparece una ventana y escogemos la opción “*Convert selected rows to defined class*”, como se aprecia en la siguiente imagen:

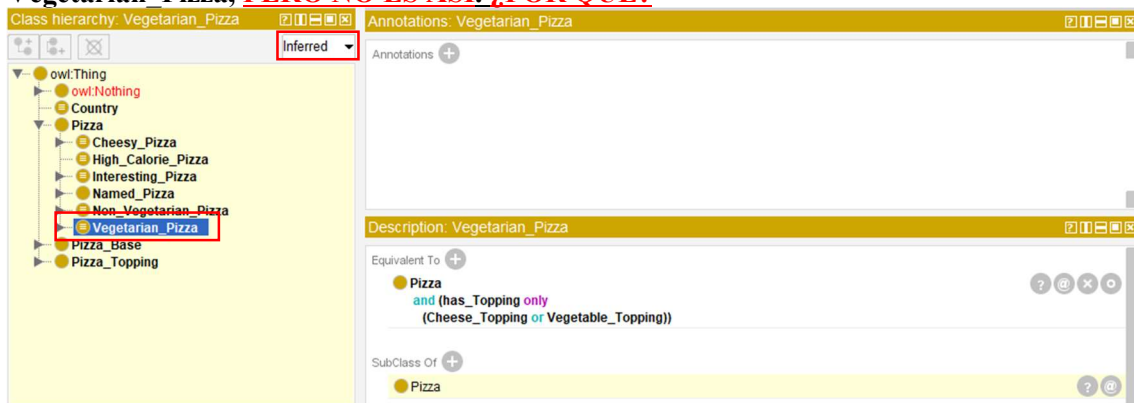
SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



Y entonces aparece la restricción en la sección “Equivalent To +”, como se aprecia en la siguiente imagen:



Esto significa que ahora la clase **Vegetarian_Pizza** es una clase Definida y por lo tanto el razonador podrá hacer clasificaciones automáticas. Si le decimos al razonador que se sincronice, entonces aparecerá en la jerarquía de clases, la clase **Vegetarian_Pizza** marcada como una clase definida (tres rayitas blancas dentro del círculo al lado de la clase) y en la jerarquía *Inferred* **DEBERIAMOS** ver las clases **Margherita_Pizza** y **Soho_Pizza** como subclases de la clase **Vegetarian_Pizza**, **PERO NO ES ASI. ¿POR QUÉ?**



Clasificación automática y el razonamiento del mundo abierto

En efecto, la lógica de descripción de OWL trabaja sobre la base del « *Open World Assumption* » (OWA). Eso significa que el razonador no puede decir que una cosa no existe mientras no se diga explícitamente que no existe. **Es decir que si alguna cosa no se establece como verdadera, entonces no se puede decir que es falsa.**

Entonces, en nuestro caso, nosotros dijimos que **Margherita_Pizza** tenía toppings

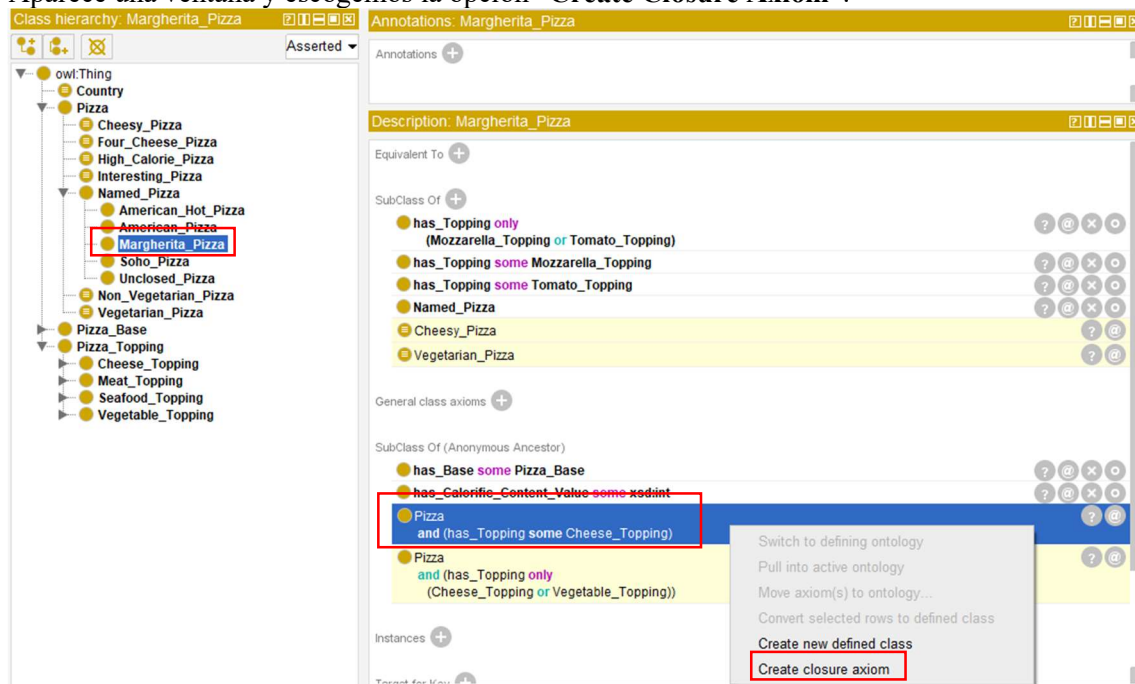
SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Mozzarella_Topping y **Tomato_Topping**, pero no dijimos explícitamente que **Margherita_Pizza** tenía únicamente esos toppings. Dada la «*Open World Assumption*» el razonador puede deducir que **Margherita_Pizza** podría tener, además, otros toppings diferentes a **Mozzarella_Topping** y **Tomato_Topping**.

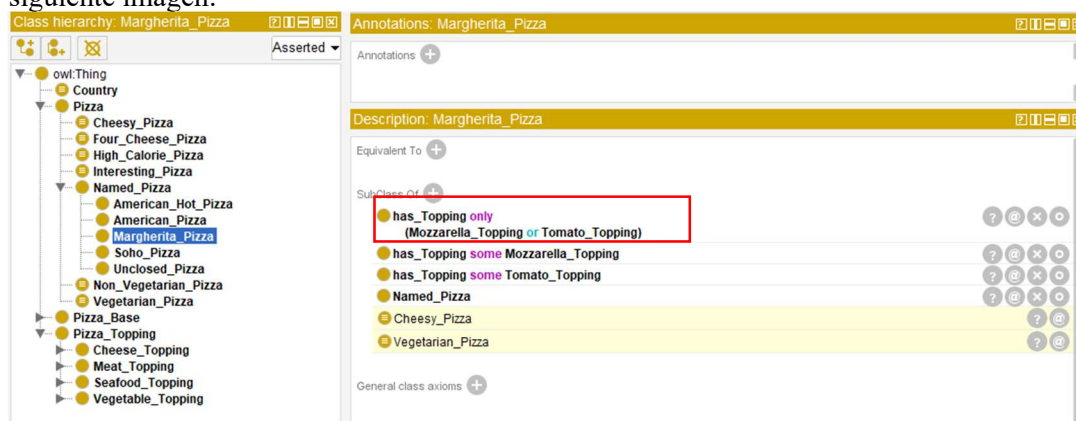
Entonces, es necesario agregar axiomas de cierre («*Closure Axioms*») a la propiedad **has_Topping** para decir que **Margherita_Pizza** tiene únicamente **Mozzarella_Topping** y **Tomato_Topping**.

Un axioma de cierre consiste en una restricción universal definida sobre una propiedad para decir que ella solo puede cumplir lo que se especificó en el *filler*. Esta restricción debe tener un filler que es la unión de los fillers de la restricción existencial.

Vamos a hacerlo para **Margherita_Pizza**. Nos ubicamos sobre esa clase y en la sección “**SubClass Of (Anonymous Ancestor)**” damos click derecho sobre la descripción de la clase. Aparece una ventana y escogemos la opción “**Create Closure Axiom**”.

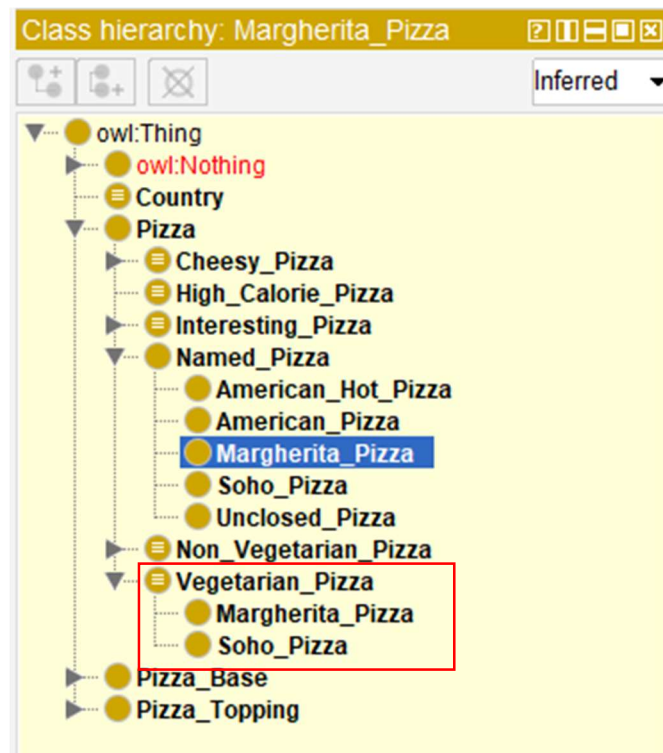


Entonces aparece el axioma de cierre en la sección “**SubClass Of +**”, como se aprecia en la siguiente imagen:

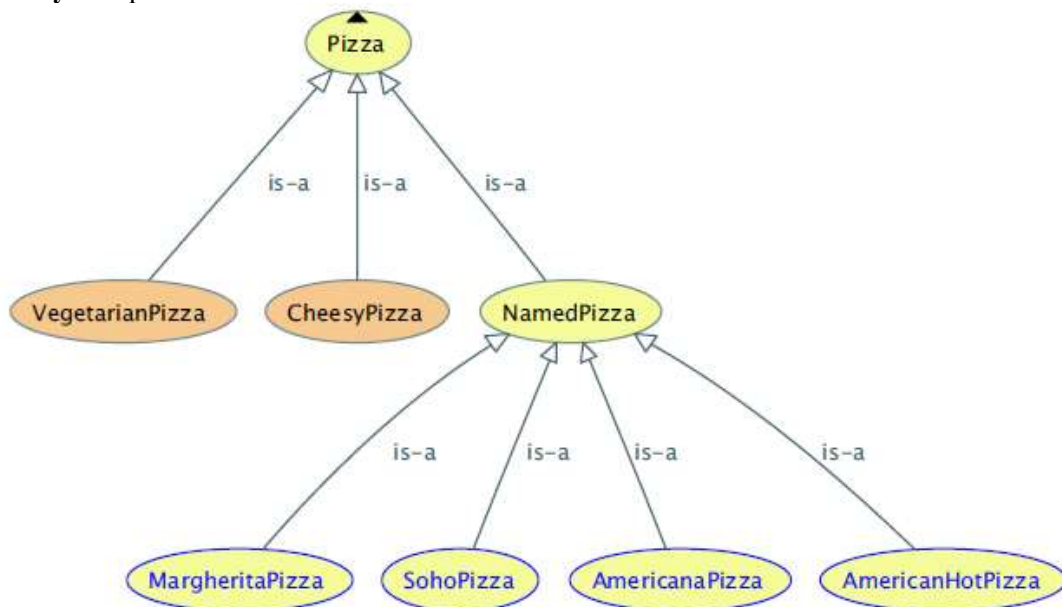


Hacemos lo mismo para **Soho_Pizza**, es decir creamos un axioma de cierre para **Soho_Pizza**. Luego, sincronizamos el razonador y podemos apreciar que ahora si se encuentran en la jerarquía inferida **Margherita_Pizza** y **Soho_Pizza** como subclases de **Vegetarian_Pizza**.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



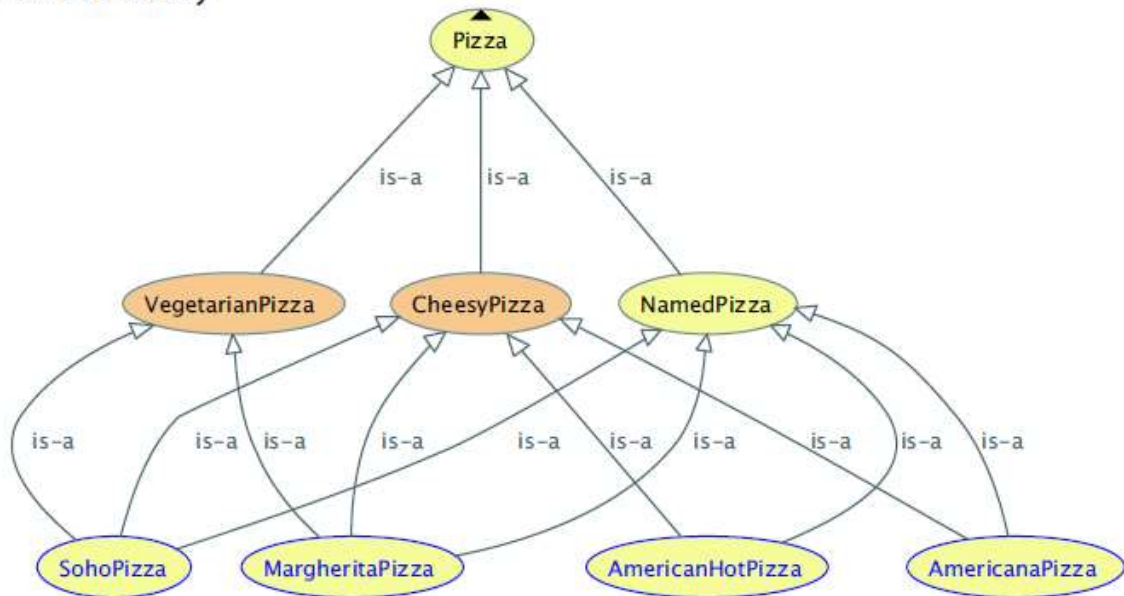
La « **Asserted Hierarchy** » es más simple que la « **Inferred hierarchy** ». La « **Asserted Hierarchy** » se parece a:



Una vez que los axiomas son introducidos, la « **Inferred Hierarchy** » es más rica que la « **Asserted hierarchy** ».

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Inferred Hierarchy



NOTA: La clasificación automática en la ontología solamente se hace para clases Definidas o Completas, es decir aquellas a las cuales se les definieron las condiciones necesarias y suficientes (en Protégé aparecen con tres rayitas blancas). Ver Cheesy Pizza y Vegetarian Pizza

Restricciones de cardinalidad

En OWL se pueden describir clases de individuos que tengan menos, más o exactamente un número de relaciones con otros individuos o valores *DataType*. Para una propiedad dada **P**, una restricción de cardinalidad mínima especifica el número mínimo de relaciones **P** en las cuales un individuo puede participar. Una restricción de cardinalidad máxima especifica el número máximo de relaciones **P** en las cuales un individuo puede participar. Una restricción de cardinalidad exacta especifica el número exacto de relaciones **P** en las cuales un individuo puede participar.

En nuestra ontología de Pizzas, podemos adicionar una restricción de cardinalidad para definir por ejemplo una **Interesting Pizza** como una **Pizza** que tiene al menos 3 toppings.

The screenshot shows the Protégé ontology editor. On the left, the 'Class hierarchy: Interesting_Pizza' is displayed, showing a tree structure where 'Interesting_Pizza' is a subclass of 'Pizza'. On the right, the 'Annotations: Interesting_Pizza' panel shows the 'Description: Interesting_Pizza' as 'Equivalent To' 'has_Topping min 3 Pizza_Topping'. The 'SubClass Of' panel shows 'Interesting_Pizza' as a subclass of 'Pizza'.

Ahora, si hacemos la reclasificación de la ontología, podemos ver en la jerarquía inferida que el razonador encontró que **American_Hot_Pizza**, **American_Pizza** y **Soho_Pizza** son pizzas interesantes (**Interesting_Pizza**) porque ellas tienen 3 o más toppings. **Margherita_Pizza** no aparece como subclase de la clase '**Interesting_Pizza**', porque ella tiene solamente 2 toppings.

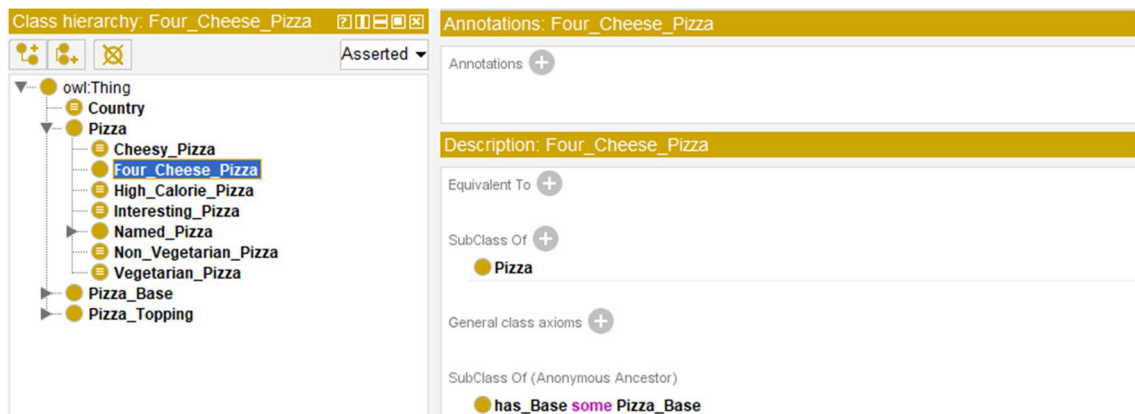
SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



Restricciones de cardinalidad calificada

En ejercicios pasados tratamos el tema de restricciones de cardinalidad, el cual especifica un número exacto de **P** relaciones que un individuo debe cumplir para suplir esa restricción y ser parte de esa clase. En contraste una restricción de cardinalidad calificada hace referencia a tener un numero específico de relaciones que deben cumplir, pero le adicionamos otra restricción a esa **P** relación, el tipo de dato al que “apunta” es de un tipo específico. Para nuestra ontología añadiremos una nueva subclase a **Pizza** llamada **Four_cheese_Pizza**, y definiremos que una **Four_cheese_Pizza** es un individuo que contiene exactamente 4 **toppings** de tipo **Cheese_Topping**.

Primero creamos la clase y agregamos una nueva restricción, con el botón (+) de Sub Class Of.



Luego agregamos la restricción, desde la pestaña de “**Object restriction creator**”, empezaremos definiendo que la pizza tendrá una restricción **has_Topping**, usremos el operador **Exactly** y una cardinalidad de 4, luego seleccionamos el tipo al que apuntara nuestra restricción, para este caso es **Cheese_Topping**.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Four_Cheese_Pizza

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

Restricted property

- owl:topObjectProperty
 - has_Country_Of_Origin
 - has_Ingredient
 - has_Base
 - has_Topping
 - is_Ingredient_Of

Restriction filler

- owl:Thing
 - Country
 - Pizza
 - Pizza_Base
 - Pizza_Topping
 - Cheese_Topping
 - Meat_Topping
 - Seafood_Topping
 - Vegetable_Topping

Restriction type

Exactly (exact cardinality) | Cardinality: 4

Some (existential)
Only (universal)
Min (min cardinality)
Exactly (exact cardinality)
Max (max cardinality)

Asserted

Use the reasoner click Reasoner > Start reasoner | Show Inference

Finalmente volvemos la clase definida junto a nuestra restricción de los toppings y nos quedaría de la siguiente manera:

Class hierarchy: Four_Cheese_Pizza

- owl:Thing
 - Country
 - Pizza
 - Cheesy_Pizza
 - Four_Cheese_Pizza
 - High_Calorie_Pizza
 - Interesting_Pizza
 - Named_Pizza
 - Non_Vegetarian_Pizza
 - Vegetarian_Pizza
 - Pizza_Base
 - Pizza_Topping

Annotations: Four_Cheese_Pizza

Annotations +

Description: Four_Cheese_Pizza

Equivalent To +

- has_Topping exactly 4 Cheese_Topping

SubClass Of +

- Pizza

General class axioms +

SubClass Of (Anonymous Ancestor)

- has_Base some Pizza_Base

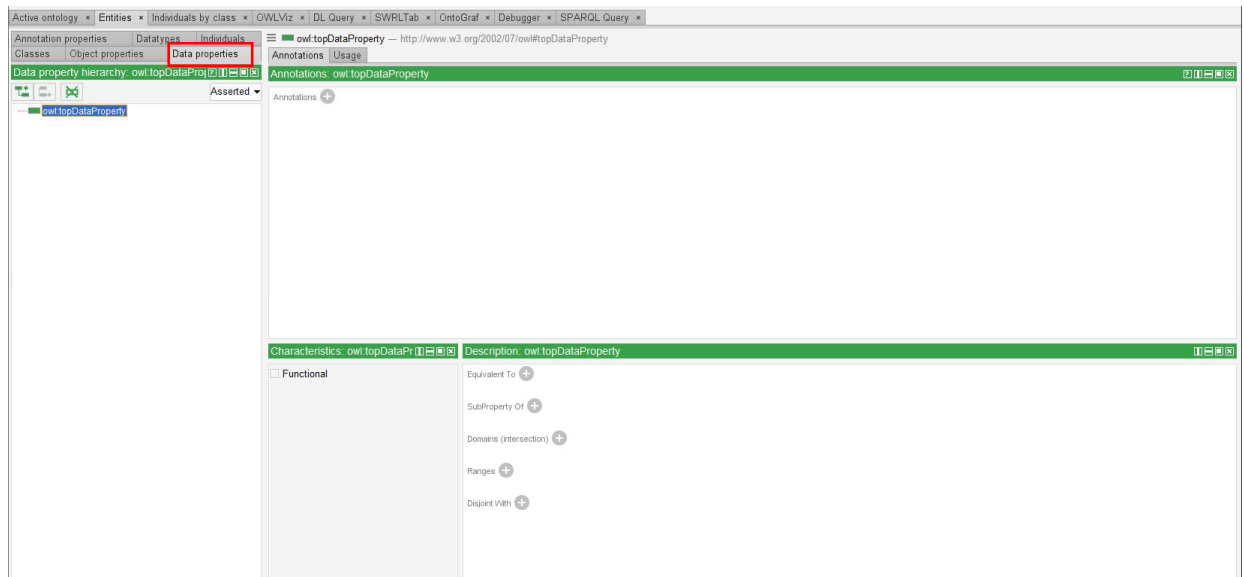
Con la definición anterior, creamos una clase en la cual hacen parte unos individuos que deben contener 4 **Cheese_Topping** para ser parte, pero esta restricción no nos limita a que estas pizzas que son creadas deben tener solo toppings **Cheese_Topping** para ser clasificadas en esta clase, en el caso de que queramos restringir que una **Four_Cheese_Pizza** es un individuo que contiene 4 **Cheese_Topping** y nada más debemos utilizar el cuantificador universal **Only**, esto en pocas palabras diría que el único **topping** que es permitido sería el de **Cheese_Topping**.

Datatype Properties

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

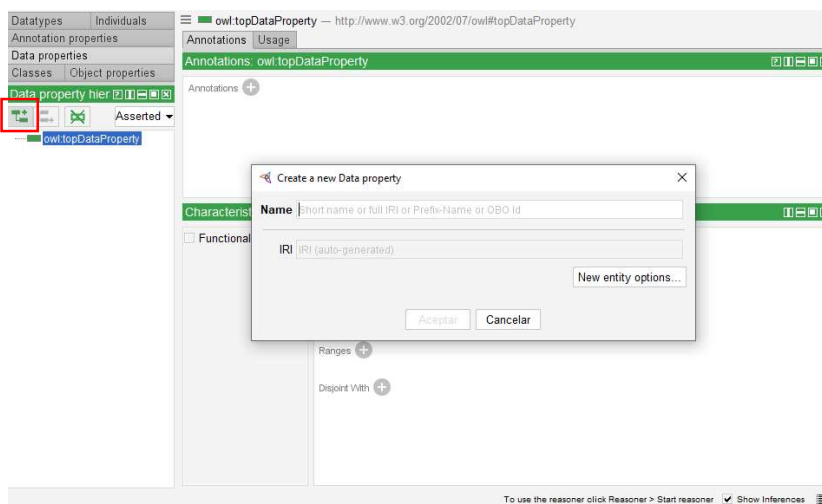
En ejercicios anteriores creamos propiedades de tipo **Object Properties**, aquí hablaremos de las **Datatype properties**, estas son las propiedades que unen un individual a un valor **XML Schema Datatype** o a un **rdf literal**.

Para crear una **Datatype Property** en Protégé lo hacemos desde la pestaña **Data properties...**



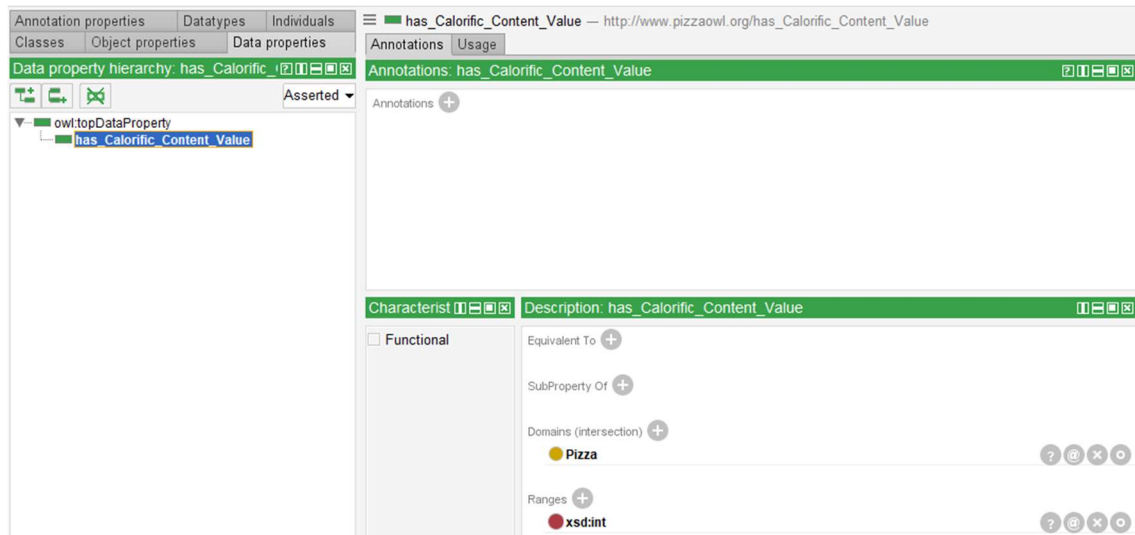
Continuaremos con nuestra ontología de las Pizzas y usando una **Data Properties**, añadiremos una propiedad llamada **has_Calorific_Content_Value**, que almacenara el valor calorífico de las Pizzas creadas.

Primero crearemos la propiedad **has_Calorific_Content_Value**, para esto le damos en el primero botón para crear una nueva **Data property**, y creamos nuestra propiedad:




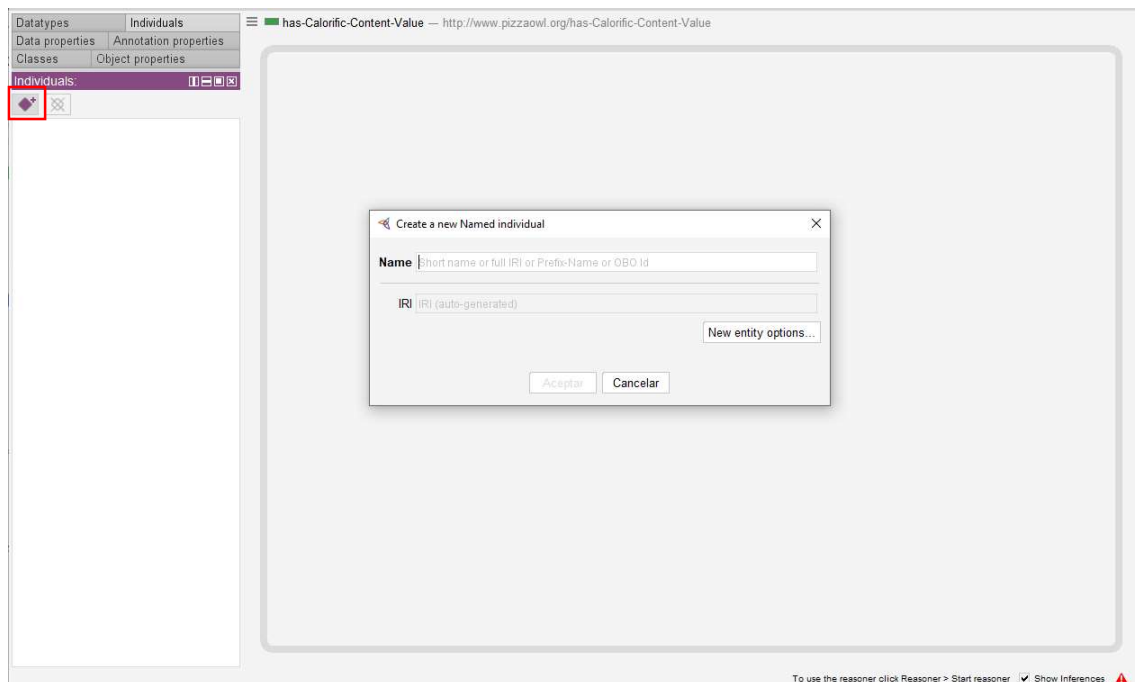
Añadimos nuestro dominio y rango, en el caso del dominio esta propiedad es asignada a la clase **Pizza** y como rango números enteros **xsd:int**.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



Individuals:

En protégé podemos añadir individuos (**Individuals**), objetos de clases de nuestra ontología, esto se hace desde la pestaña “**Individuals**” en protégé, para crear un nuevo Individuo seleccionamos el botón  y luego le damos un nombre a nuestro **Individual**, estos nombres también deben ser únicos al momento de nombrarlos, mas adelante ampliaremos el tema de los **Individuals**.



Continuando con nuestra ontología de la Pizza, agregaremos unos cuantos **Individuals** de tipo Pizza, y le daremos sus propiedades respectivamente. Cree los siguientes **Individuals** y asigne el valor respectivamente de la propiedad **has_Calorific_Content_Value** en cada individuo:

Pizza_Margherita, valor calorífico de 237

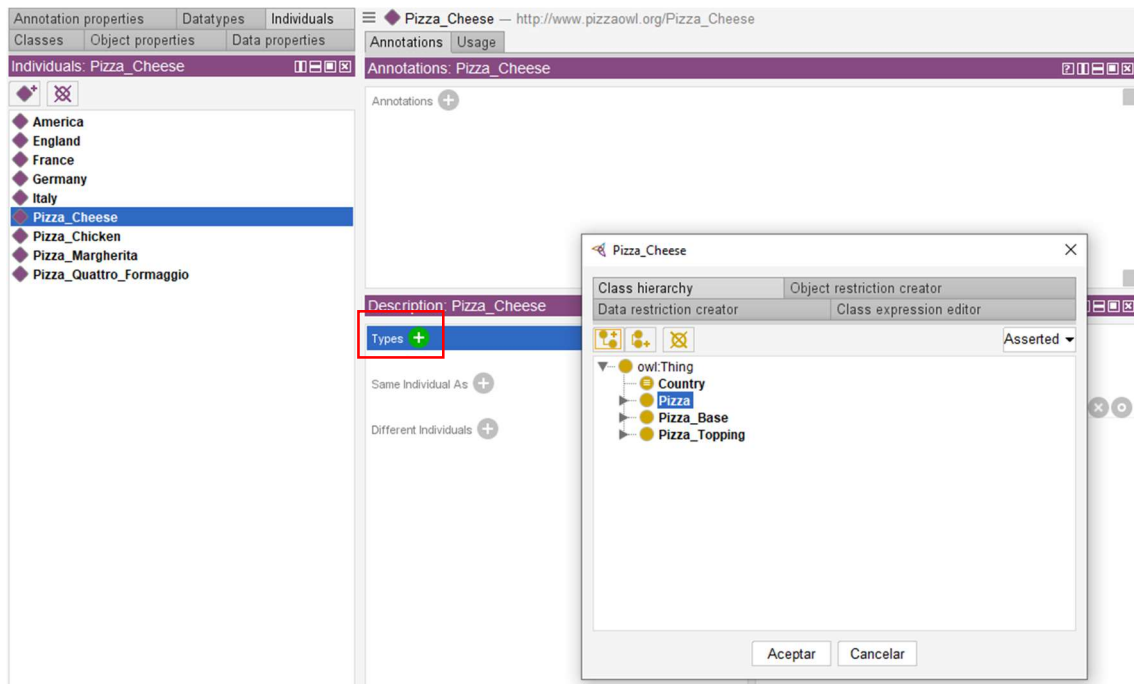
Pizza_Cheese, valor calorífico de 400

Pizza_Chicken, valor calorífico de 350

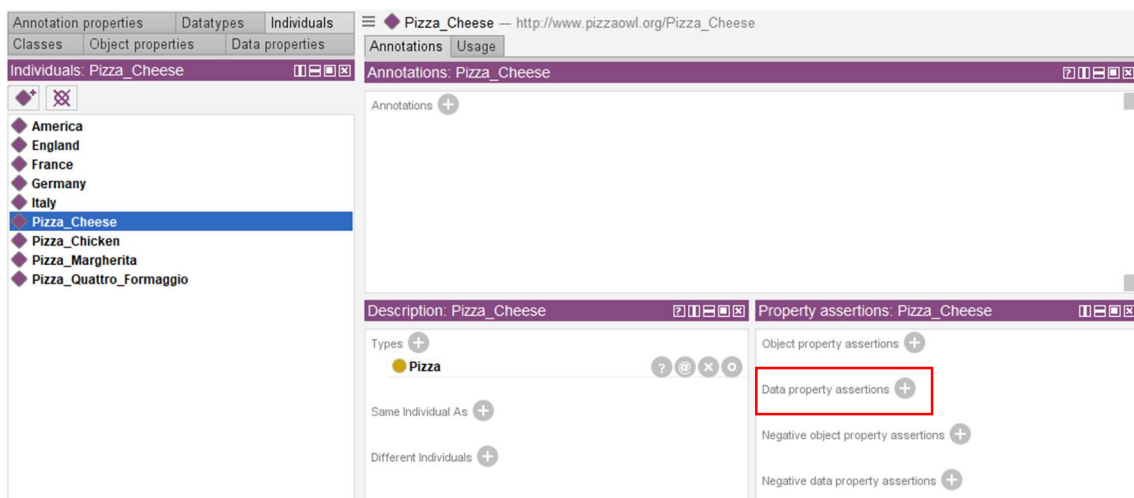
Pizza_Quattro_Formaggio, valor calorífico de 732

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Tenga en cuenta que luego de crear un **Individual** debe darle un tipo, para esto desde el apartado de “**Types**” en la descripción del **Individual** le daremos al botón (+), y seleccionamos el tipo de nuestro **Individual** en este caso es de tipo **Pizza**.

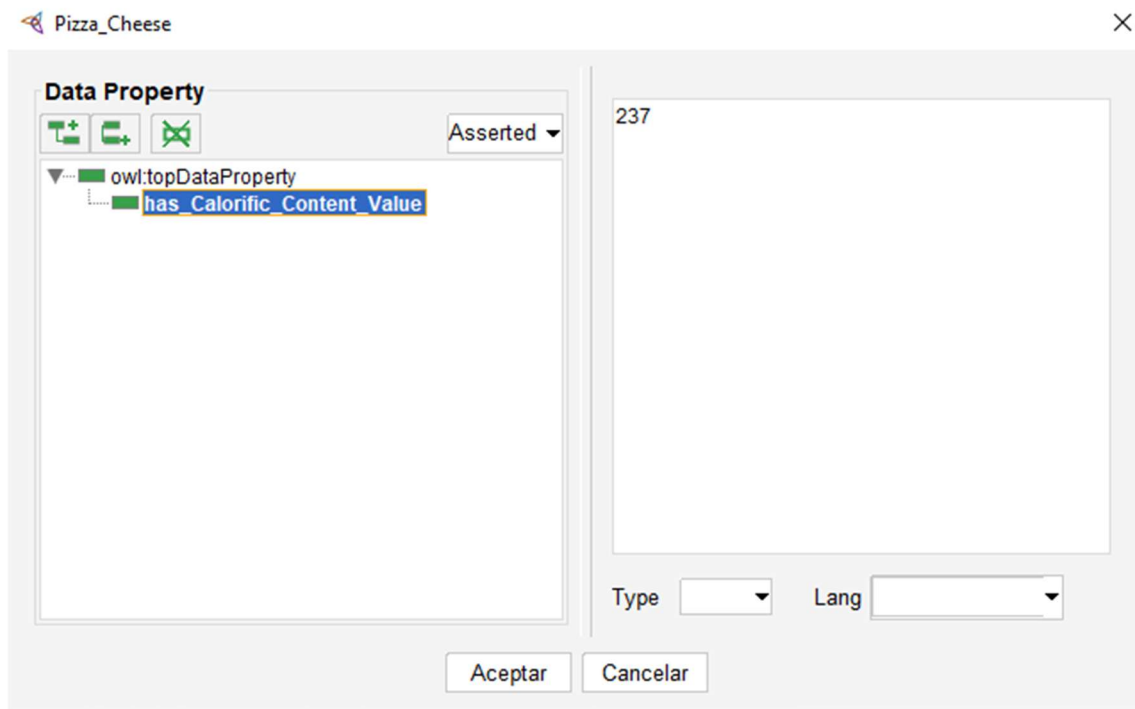


Para agregar el valor calorifico de nuestras pizzas, lo hacemos desde el apartado “**Data property assertions**”.



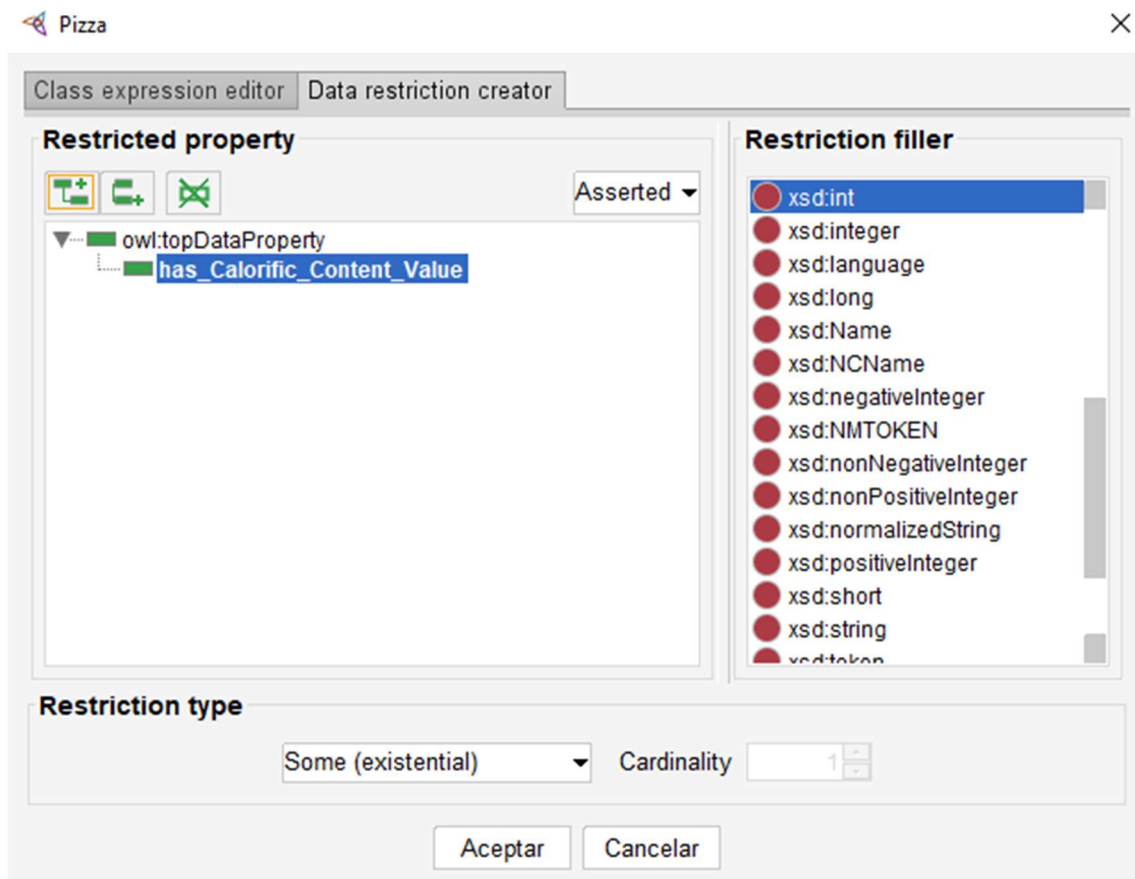
Se nos despliega el siguiente cuadro desde el cual seleccionamos el tipo de **Data Property** que le estamos asignando, en este caso **has_Calorific_Content_Value**, en el lado derecho ingresamos el valor y el tipo respectivamente.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



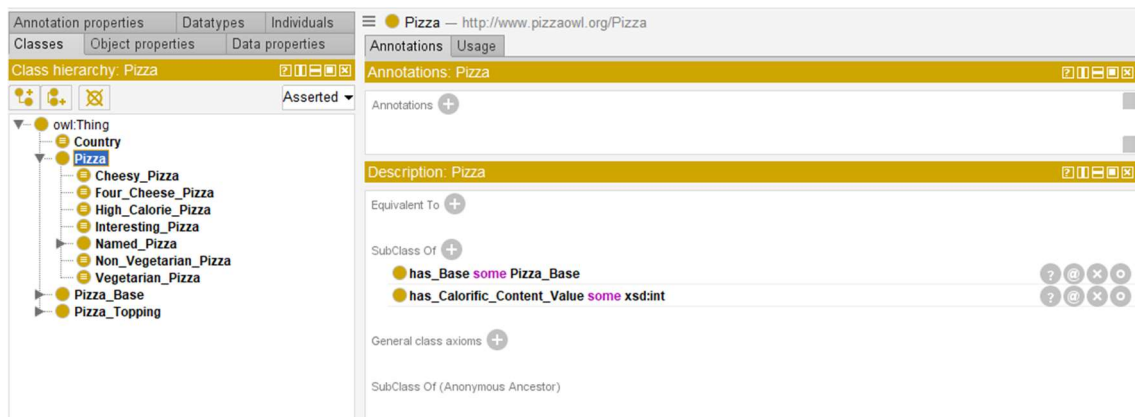
Datatype restrictions:

Así como se creaban restricciones con **object properties**, también podemos crear restricciones de valores en las Clases, primero crearemos una restricción que define que todas las pizzas deben tener una propiedad **has_Calorific_Content_Value** y un valor, para esto crearemos una restricción en la clase **Pizza**.

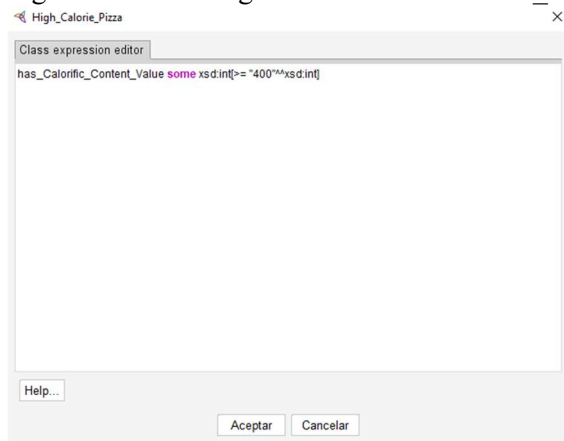


SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

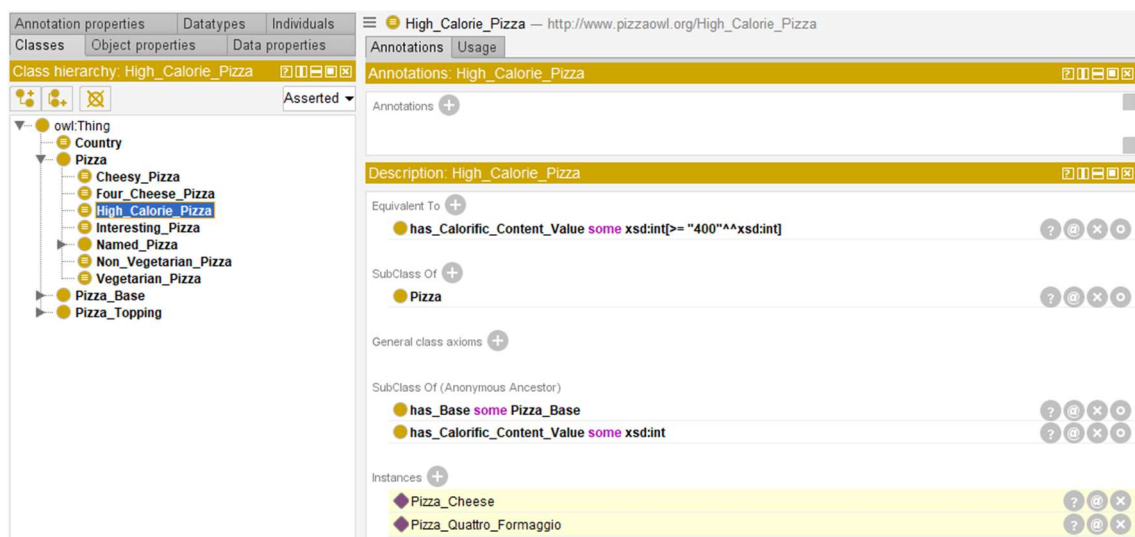
Quedaría de la siguiente manera:



Adicionalmente podemos crear restricciones de rangos de valores, crearemos una clase nueva para clasificar las Pizzas que tengan un valor calorífico mayor a 400, para esto crearemos una nueva subclase de **Pizza** llamada **High_Calorie_Pizza** y diremos que los pertenecientes a esta clase son aquellos con un valor calorífico mayor a 400, crearemos una restricción nueva y luego volveremos la clase definida. Desde la pestaña de restricciones “**Class expresión editor**” ingresaremos la siguiente restricción “has_Calorific_Content_Value some xsd:int[>= 400]”



Luego de activar el razonador podemos notar que dos de los individuos han sido clasificados en nuestra clase **High_Calorie_Pizza**.



SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

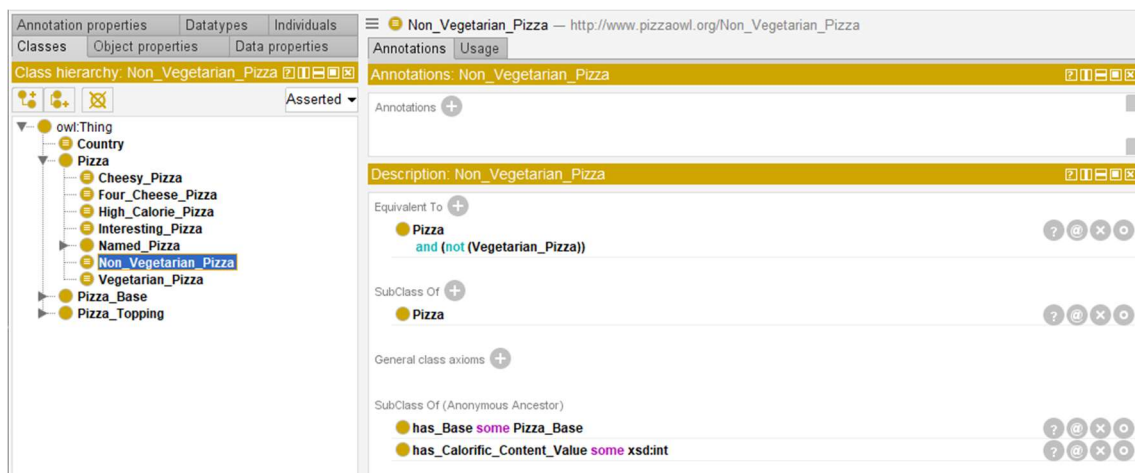
Mas del mundo razonamiento en el mundo abierto

Por medio de un ejemplo grafico veremos que si tenemos una cierta clase que no restringe sus valores puede no ser clasificada en dos grupos que son opuestos, uno que los contiene y el otro su complemento.

Crearemos una nueva subclase de **Pizza** llamada **Non_Vegetarian_Pizza**, y estará definida por todas las pizzas que no sean **Vegetarian_Pizza**, para esto añadiremos una restricción y volveremos definida la clase, de la siguiente forma:

- Diremos que la clase **Non_Vegetarian_Pizza** es disjunta de la clase **Vegetarian_Pizza**.
- Diremos que la restricción para esta clase será “**Pizza and not Vegetarian_Pizza**”
- Volveremos la clase definida

El resultado seria algo así:



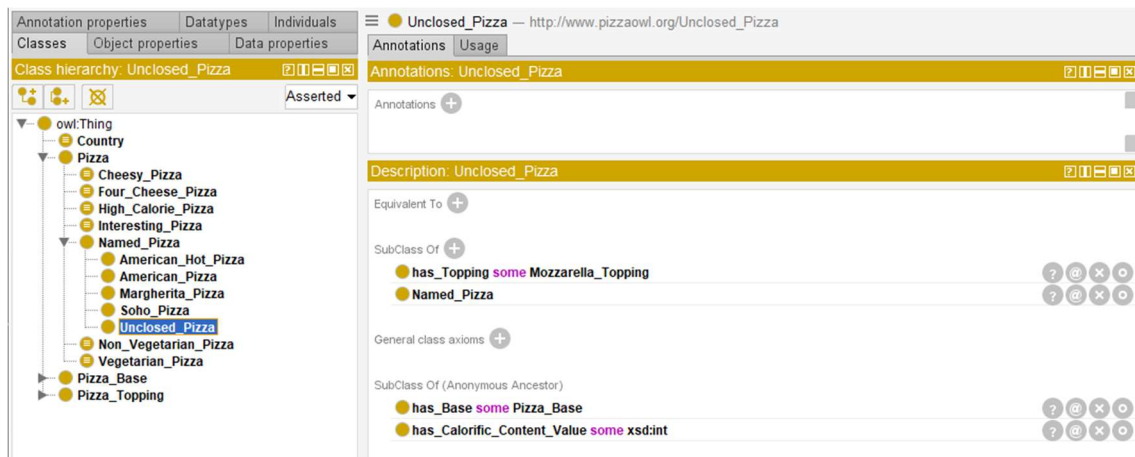
Al prender el razonador obtendremos:



Podemos observar que las Clases que no son vegetarianas son **American_Hot_Pizza** y **American_Pizza**, hasta el momento observamos que nuestra ontología hace lo que en cuestión nosotros asumiríamos como correcto, todas estas clases tienen un “**closure axiom**”, por lo que no quedan en la vaguedad del mundo abierto.

Ahora agreguemos una nueva subclase a la Clase **Named_Pizza** llamada **Unclosed_Pizza** y diremos que esta pizza tendrá una restricción de **has_Topping some Mozzarella_Topping**.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



Al encender el razonador notamos que esta no se clasifica como una **Vegetarian_Pizza** o una **Non_Vegetarian_Pizza**. ¿Por qué?

Esto se debe al mundo abierto, una **Unclosed_Pizza** es aquella que tenga una **topping** de **Mozzarella_Topping**, pero esto no restringe el hecho de que puede tener toppings de solo el tipo **Vegetable_Topping** o de otros tipos, al no poder determinar si es o no es una **Vegetarian_Pizza**, no podrá clasificarse tampoco como **Non_Vegetarian_Pizza**, ya que es el complemento de esta.

Creación de individuals:

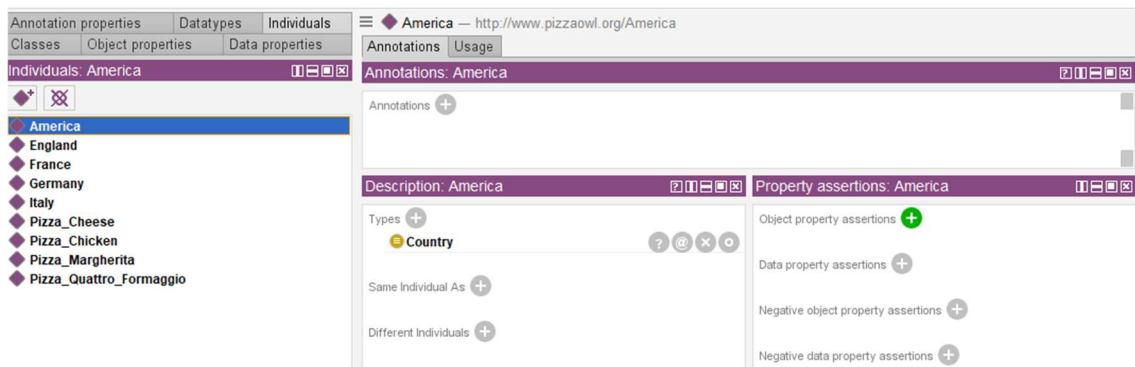
Como anteriormente hablábamos los **Individuals** son como objetos (Individuos) de nuestras clases, supongamos que queremos agregar un país de origen nuestras diferentes definiciones de la ontología como el país de origen de un **topping** o de una **Pizza**.

Esta es la pestaña de “**Individuals**”



Para el caso de nuestra ontología de la pizza añadiremos una nueva clase **Country**, y añadiremos unos cuantos individual de tipo country, que serán **Italy**, **America**, **England**, **France**, y **Germany**.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

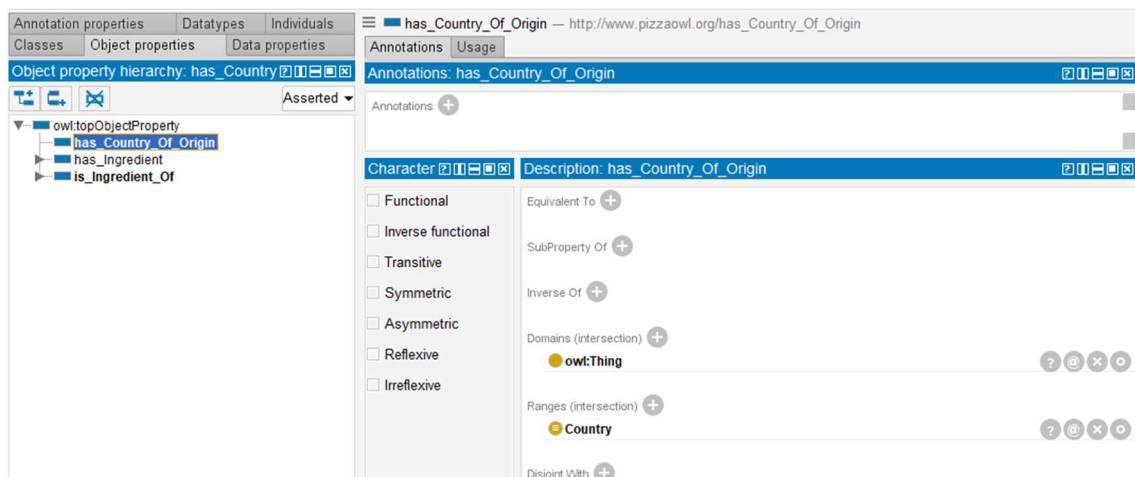


Cabe recalcar que **OWL** no utiliza la asunción de nombre únicos (Unique Name Assumption-UNA), de igual manera que las clases estos pueden reconocerse por otros nombres o diferenciarse entre sí, para esto existe en la descripción de un **Individual** las propiedades **Same Individual As** y **Different Individuals**.

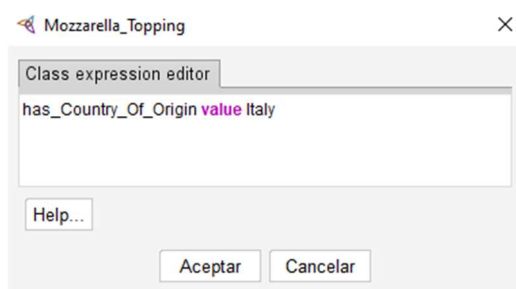
hasValue Restricciones:

Existen otras restricciones que podemos definir a una clase, los hasValue Restricciones, hacen referencia a que cierta clase tendrá una propiedad o relación de un cierto valor en específico, para entenderlo mejor usaremos el siguiente ejemplo:

Queremos decir que el país de origen del **Mozarella_Topping** es **Italy**, para esto debemos agregar una nueva propiedad llamada **has_Country_Of_Origin**.

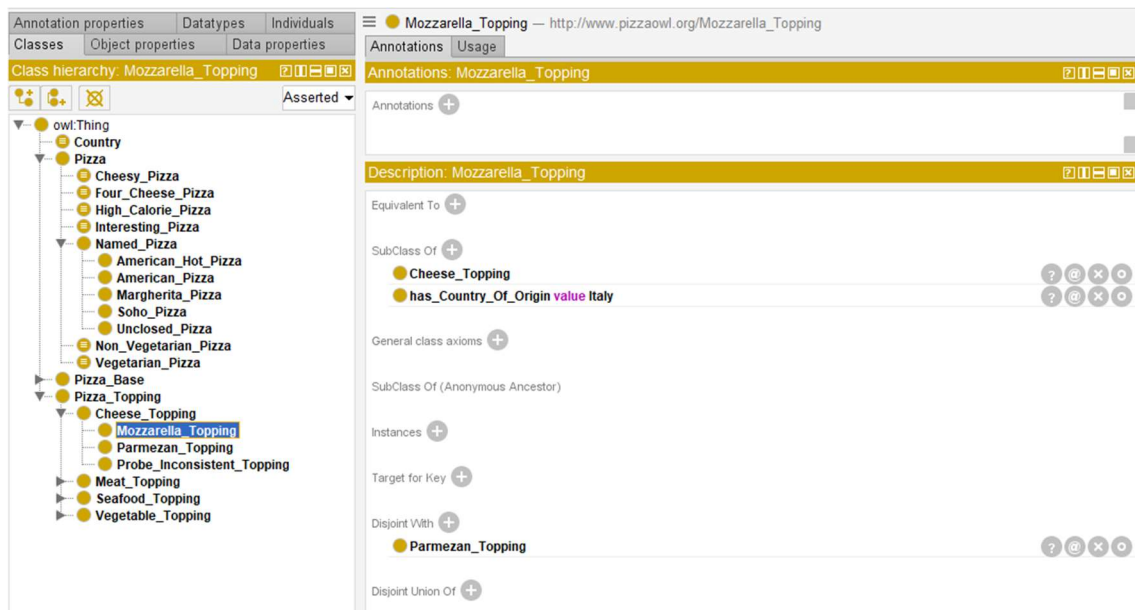


Luego seleccionamos la clase **Mozarella_Topping** y agregamos una restricción de la siguiente manera:



Y luego nos quedaría al final lo siguiente:

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



Con esta definición queremos decir que para que un Individuo al crearlo y clasificarlo correctamente sea de tipo **Mozzarella_Topping** debe tener como valor de la propiedad **has_Country_Of_Origin** el individuo **Italy**.

Enumerated Classes:

Las **Enumerated Class** hacen referencia a clases que tienen unos valores ya predefinidos e inamovibles, por ejemplo, si tenemos la clase **Días-De-La-Semana**, en ella queremos que los únicos valores que se observen sean Lunes, Martes, Miércoles, Jueves, Viernes, Sábado y Domingo, estas clases con valores inamovibles las llamamos **Enumerated Classes**.

Para dar continuidad con el ejercicio convertiremos a **Enumerated Class** la Clase **Country**, en protégé para definir que una clase es una **Enumerated Class** lo hacemos de la siguiente manera en la descripción de la clase **Country** añadiremos en el apartado de “**Equivalent to**” la siguiente definición {**America, England, France, Germany, Italy**}, note que encerramos los diferentes **Individuals** anteriormente creados entre corchetes ({}), de esta manera y separados por comas definimos los valores de una **Enumerated Class**. **Tenga en cuenta que los valores de una Enumerated Class deben ser creados con anterioridad y estos siempre serán Individuals.**

Propiedades de anotación:

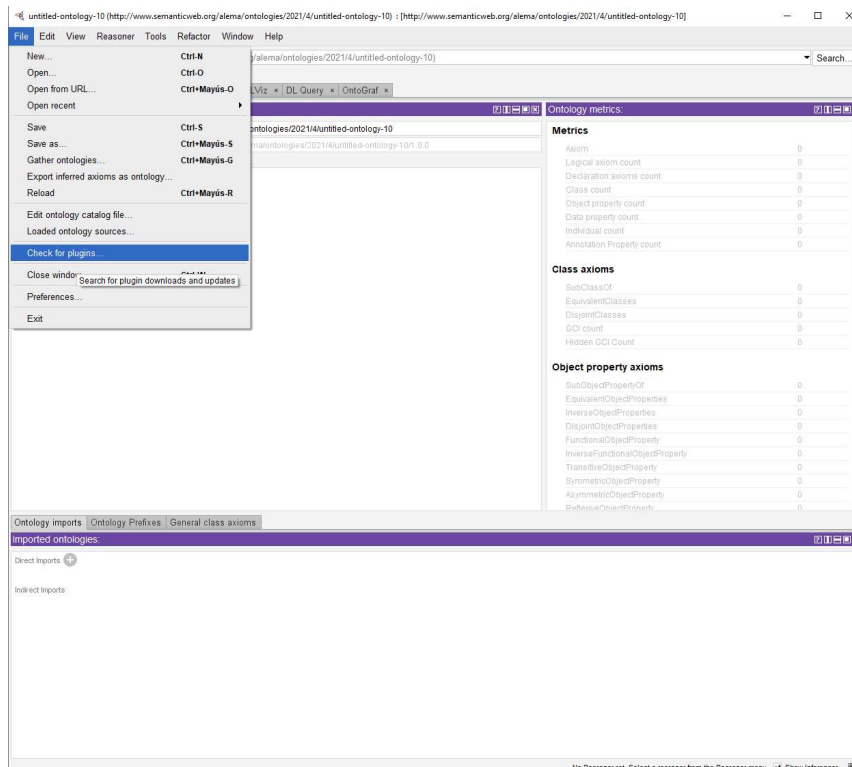
Leer de la página 94, 7.4 **Annotation Properties**

http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_3.pdf

Visualización gráfica:

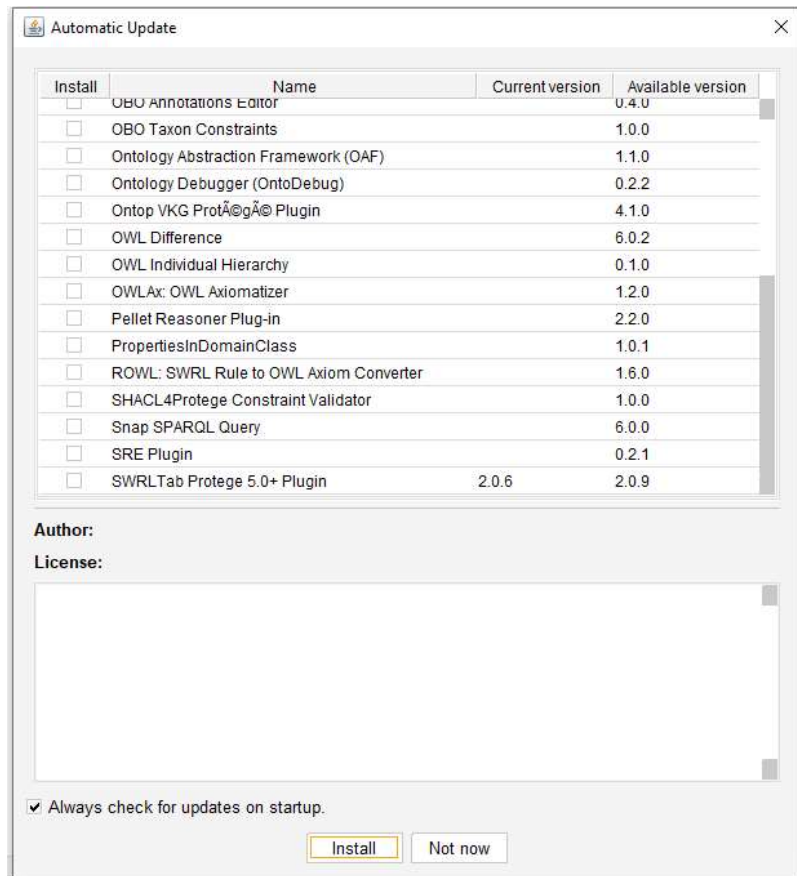
Si deseamos visualizar las relaciones entre las clases, propiedades, rangos y dominios, entre otras cosas, podemos usar una herramienta llamada OntoGraf, que nos ayuda a ver de manera grafica estas relaciones. Para esto debemos instalarla si aún no tenemos dicho plugin, esto lo hacemos, dando click en file > Check for plugins... y esperamos

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB



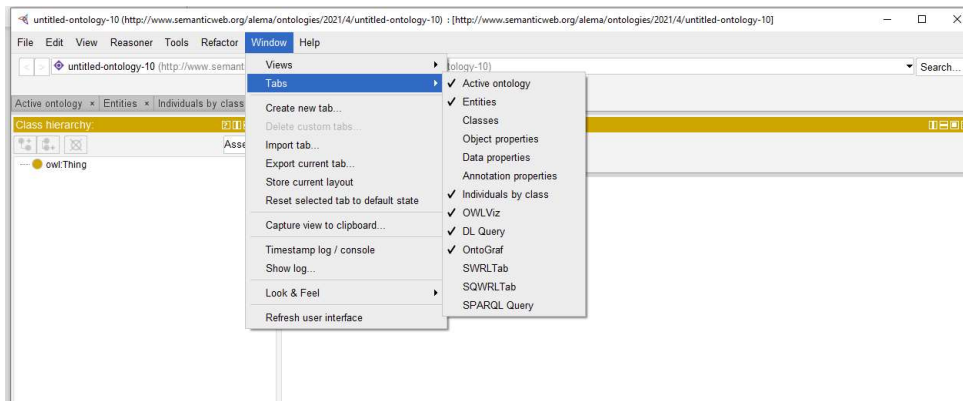
Luego se nos desplegara una ventana similar a la siguiente, alli buscaremos el plugin ontograf, y daremos click en install.

Nota: Si no aparece dicho plugin en el listado, lo más probable es que ya lo tenga instalado.

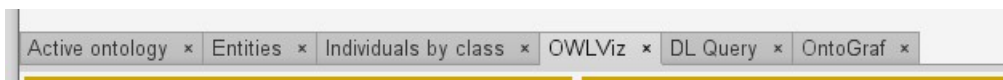


SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

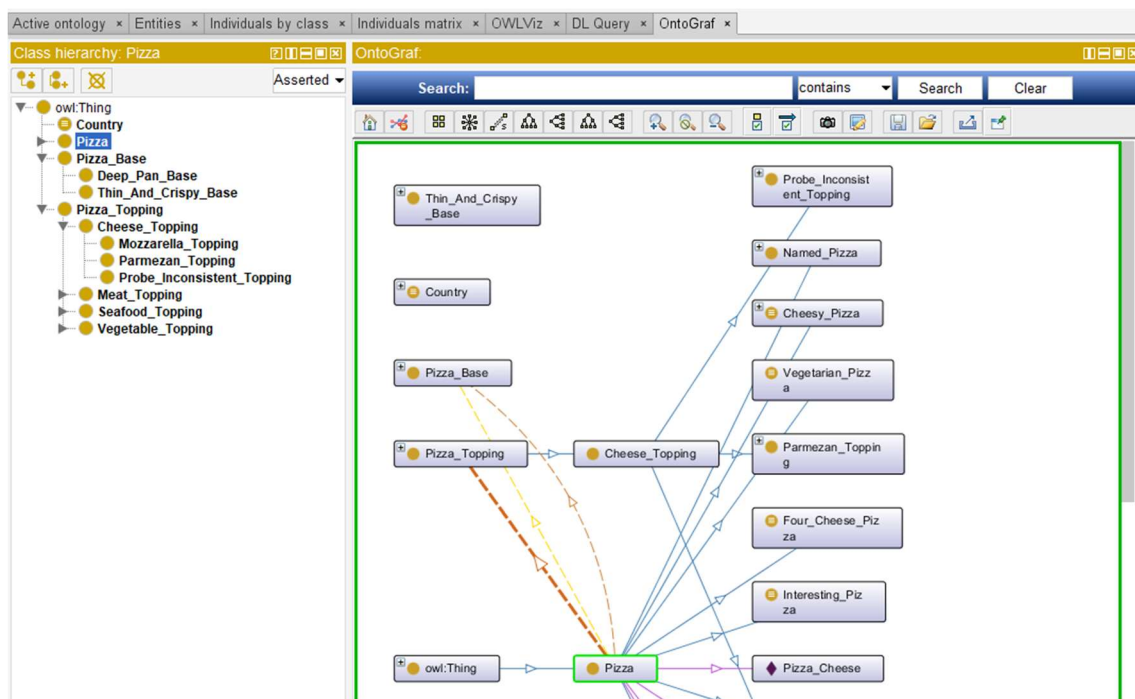
Para visualizar dicho plugin, daremos click en Window > Tabs> y buscamos el plugin en este caso OntoGraf



Ahora se visualizará en las Tabs de protegé



Ahora, al entrar en la Tab de OntoGraf, la manera de usar dicha herramienta es dando click en las clases en el lado izquierdo, y abriendo el grafo, con el botón +, en lado derecho, si nos paramos en las líneas entre clases o individuos, veremos el tipo de relación que tienen dichos recursos.

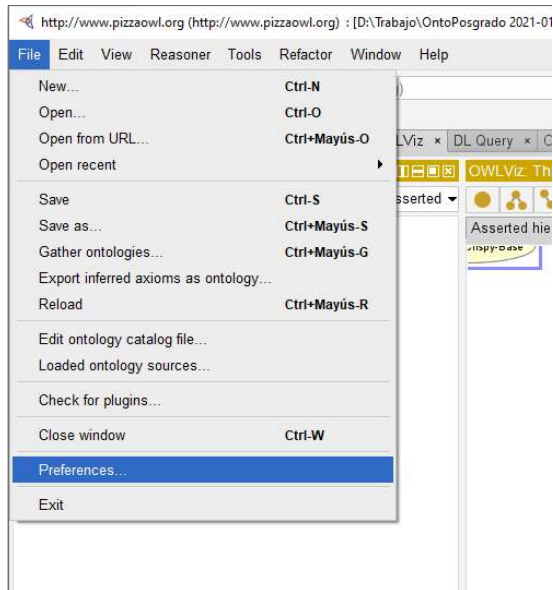


Ahora bien, si deseamos ver otra visualización útil de nuestra ontología, podemos hacer uso del plugin OWL Viz, se instala de manera similar a OntoGraf.

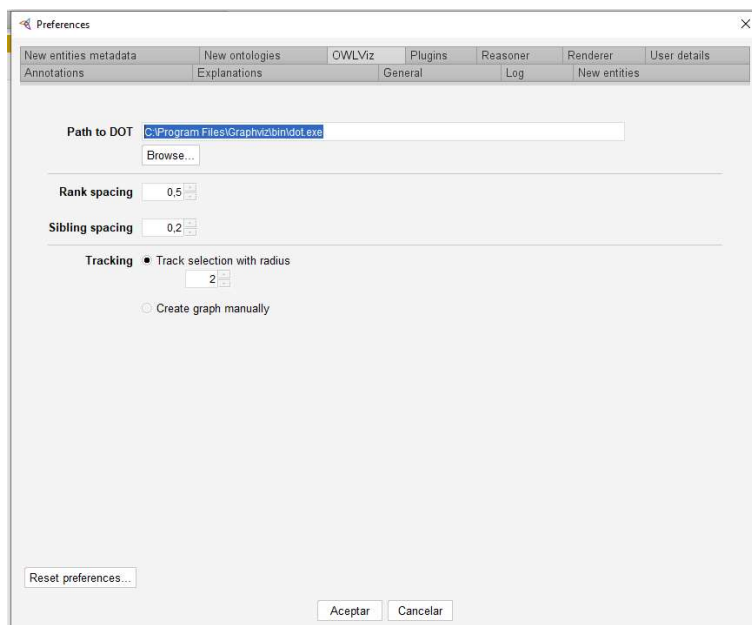
Antes de empezar a usar OWL Viz, debemos hacer una pequeña configuración previa. Debemos instala Graph Viz y especificar su ruta de instalación, para esto primero debemos instalar Graph

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Viz, en el siguiente enlace puede descargarlo e instálelo <https://graphviz.org/download/>, luego por defecto dicha librería se instalará en la dirección C:\Program Files\Graphviz , en caso de que no sea así, busque el directorio donde se instaló, ahora indicaremos en que lugar se encuentra instalado Graph Viz en protegé, para ello en protegé, de click en File > Preferences...



Allí seleccione OWLViz, y cambie la dirección “Path to DOT”, del lugar de instalación de GraphViz, dicha dirección de manera predeterminada será C:\Program Files\Graphviz\bin\dot.exe , en caso contrario especifique el lugar del dot.exe y luego de click en aceptar.



Ahora entrando en la Tab OWLViz, podrá visualizar ciertas relaciones, y también algunas diferencias al momento de encender el razonador.

SISTEMAS DE RECUPERACIÓN DE INFORMACIÓN EN LA WEB

Referencias

- Tutorial-Ontologia-OWL-Pizzas-Español [[Gloria Lucia Giraldo Gomez](#)]
- Protégé OWL Tutorial (Pizza Ontology)
[\[http://owl.cs.manchester.ac.uk/publications/talks-and-tutorials/protg-owl-tutorial/\]](http://owl.cs.manchester.ac.uk/publications/talks-and-tutorials/protg-owl-tutorial/)