
OWL Web Ontology Language

Contenido

- OWL2: Introducción
- Conceptos Básicos de OWL
- Sintaxis – DL, OWL, Manchester
- **Lenguaje OWL**
 - Clases
 - Propiedades: : Object Properties, Data Properties
 - Individuos

Bases OWL: Sintaxis

- Toda la información en lenguajes ontológicos se expresa mediante un **tripleta**.
- Un tripleta consta de un sujeto, un predicado y un objeto.

Ejemplos:

subject predicate object

Azuay tieneCapital Cuenca

Cuenca tieneAlcalde Pedro_Palacios

Pedro_Palacios fechaNacimiento 1976

- Otra palabra para un tripleta es una **declaración** o un **hecho**.

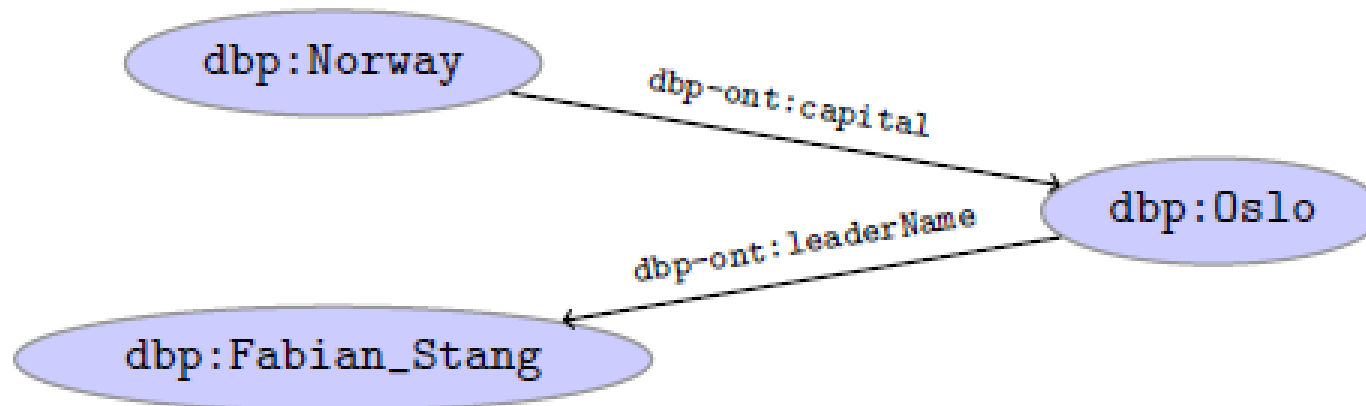
Bases OWL: Sintaxis

- Un grafo es un conjunto de tripletas. P.ej.,

dbp:Noruega dbp-ont:capital **dbp:Oslo** .

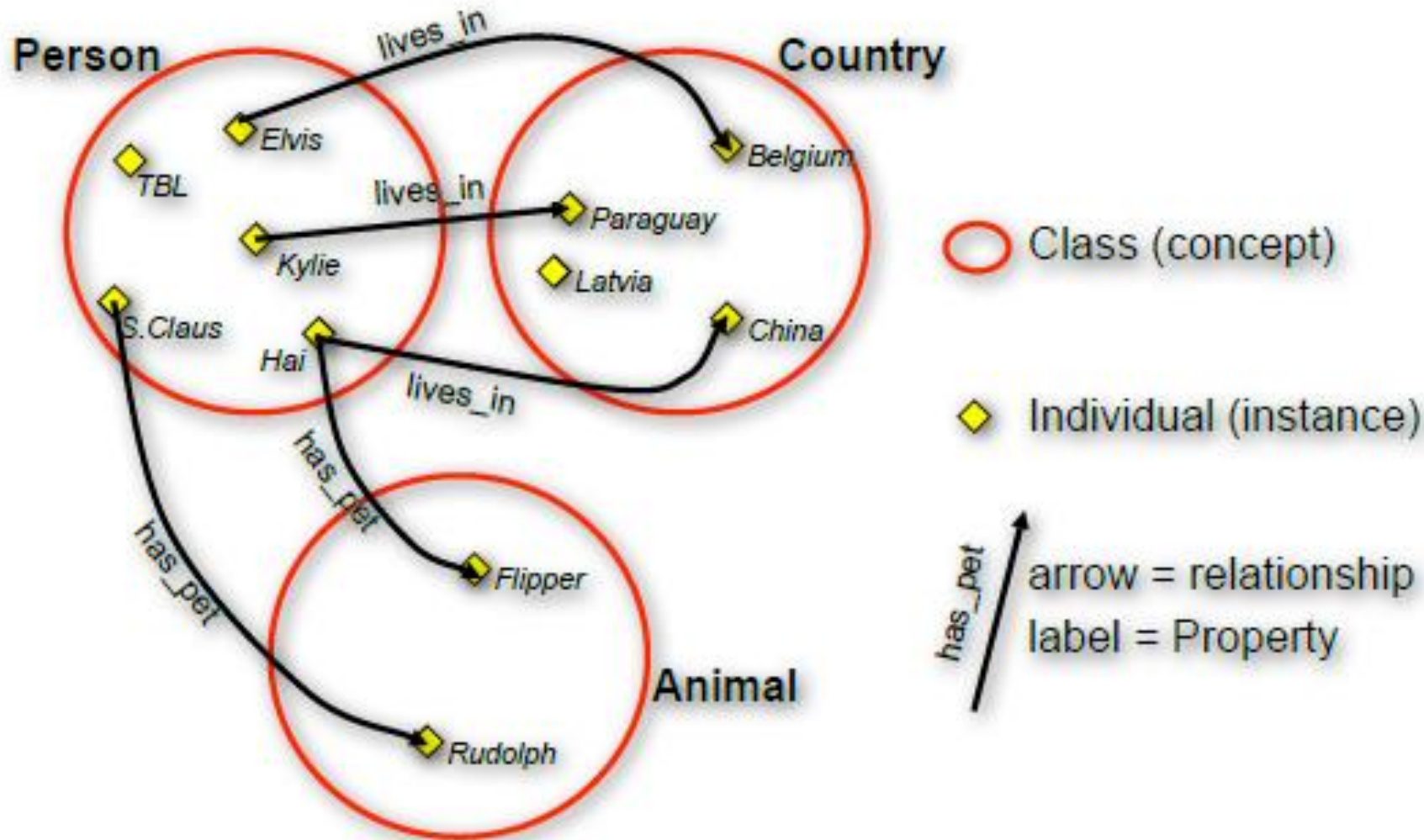
dbp:Oslo dbp-ont:leaderName **dbp:Fabian_Stang** .

- es un grafo que contiene dos tripletas.
- Los grafos a menudo se representan como un gráfico etiquetado dirigido:



Elementos Básicos de Ontologías OWL

Resumen de construcciones OWL

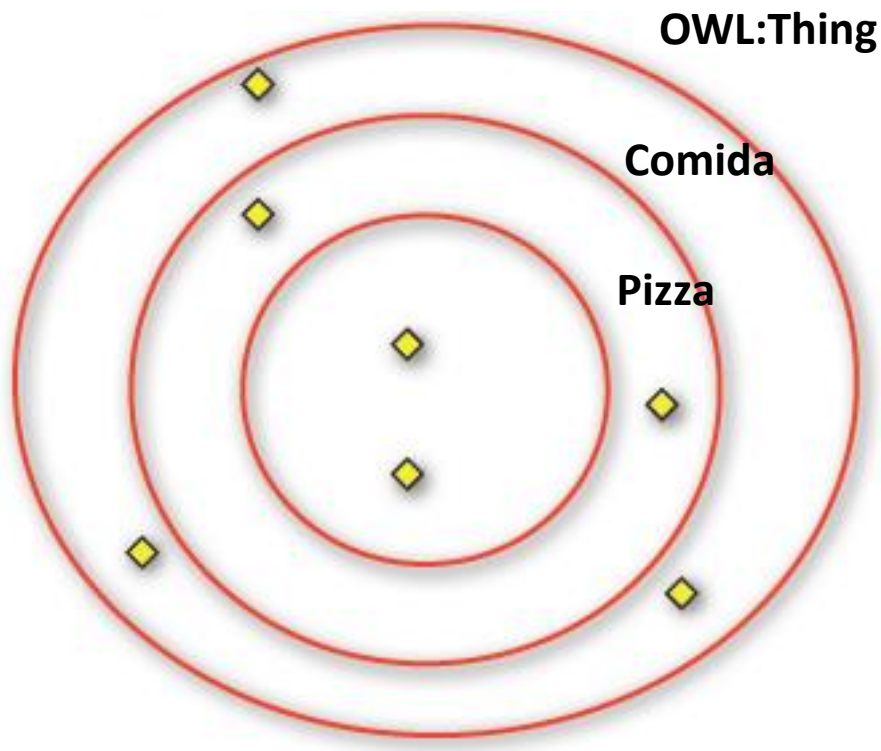


Clases: Declaración

- Hay dos clases predefinidas
 - *owl:Thing* (clase que contiene todos los clases)
 - *owl:Nothing* (clase vacía)
- Cada clase (*class*) en OWL es miembro de la clase *owl:Thing*
- Clases son definidas usando **owl:Class**
- Definición de una clase
:Persona a owl:Class .

Clases: Inclusión

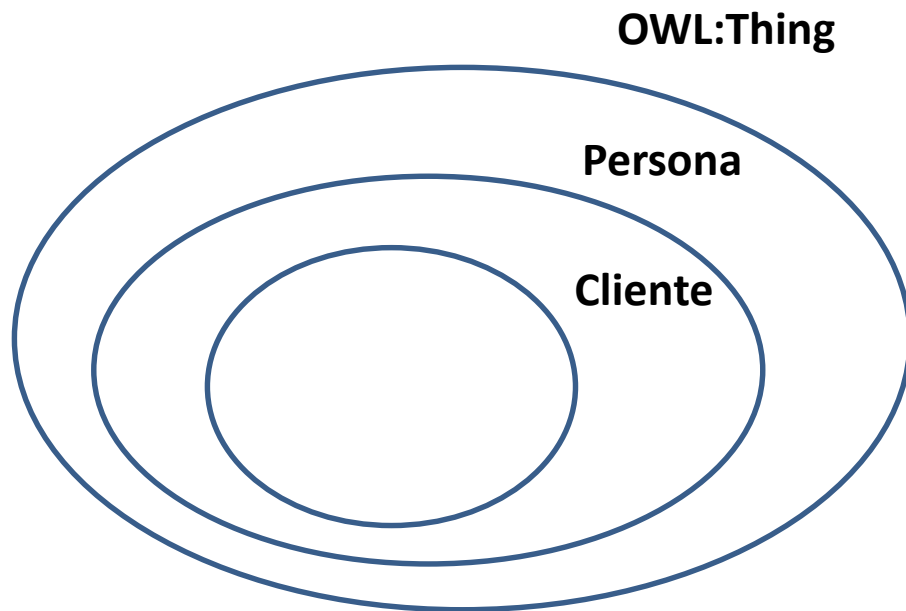
- Superclase/subclase son relaciones de tipo “es un(a)”
- **Todos** los miembros de una subclase son miembros de su superclase



- ▶ Comida subsume Pizza
- ▶ Comida es una superclase de pizza.
- ▶ Pizza es una subclase de Comida
- ▶ Todos los miembros de Pizza son también miembros de Comida
- ▶ Todo es miembro de OWL:Thing

Clases: Inclusión

- Superclase/subclase son relaciones de tipo “es un(a)”
- **Todos** los miembros de una subclase son miembros de su superclase

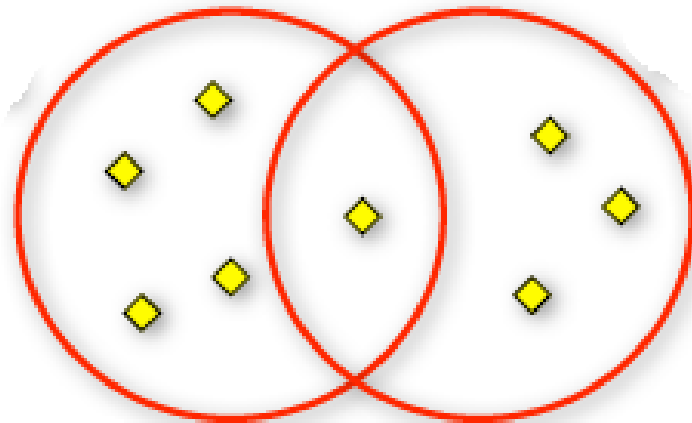


$\text{Cliente} \subseteq \text{Persona}$

$\text{Persona} \subseteq \text{OWL:Thing}$

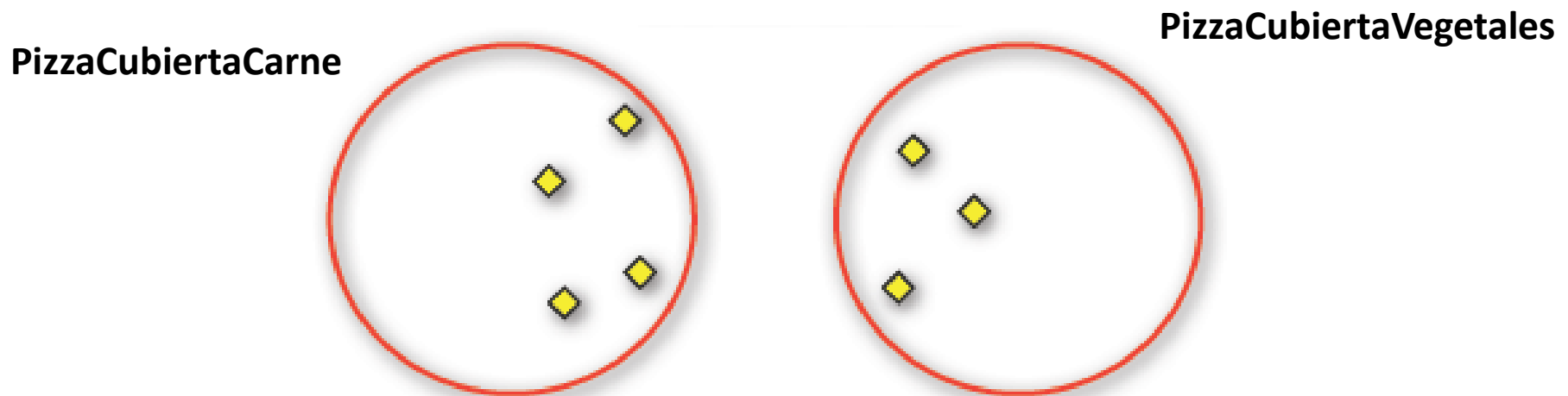
Clases: Disjuntas

- Las clases en OWL se asume que se "superponen"
 - Los individuos de una clase A también pueden ser individuos de la clase B
 - **No se puede suponer que un individuo no es miembro de una clase en particular simplemente porque no se ha declarado que sea miembro de esa clase.**



Clases: Disjuntas

- Indicar que dos clases son disjuntas significa que por ejemplo PizzaCubiertaCarne y PizzaCubiertaVegetales no pueden ser lo mismo al mismo tiempo.

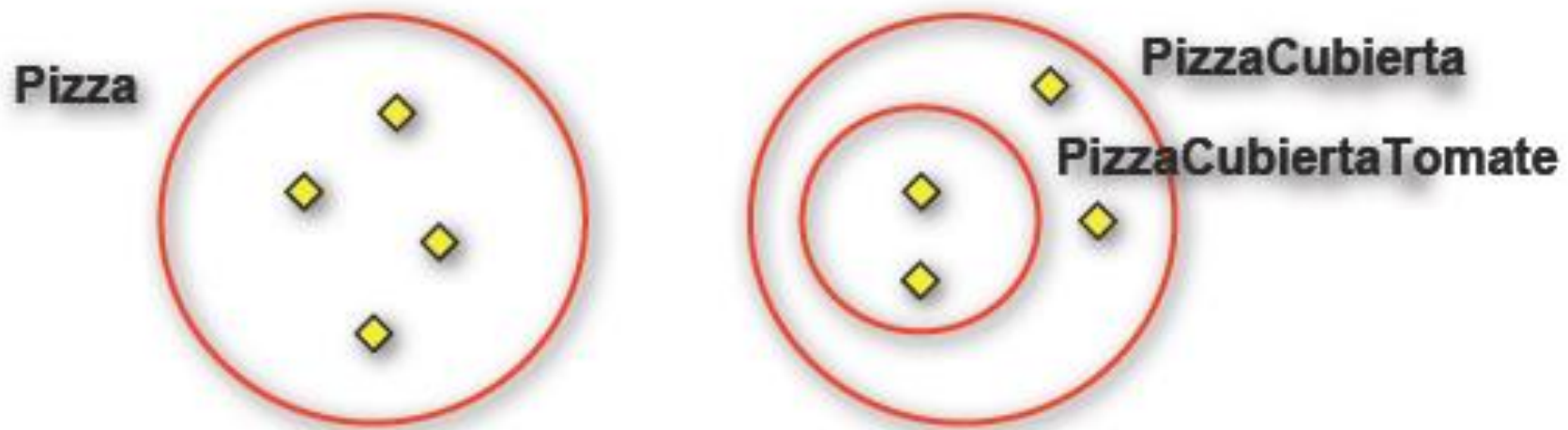


PizzaCubiertaCarne no puede nunca ser una subclase de **PizzaCubiertaVegetales** (y vice-versa)

Esto ayuda ha encontrar errores

Clases: Disjuntas

Las clases disjuntas se heredan de la jerarquía de inclusión.

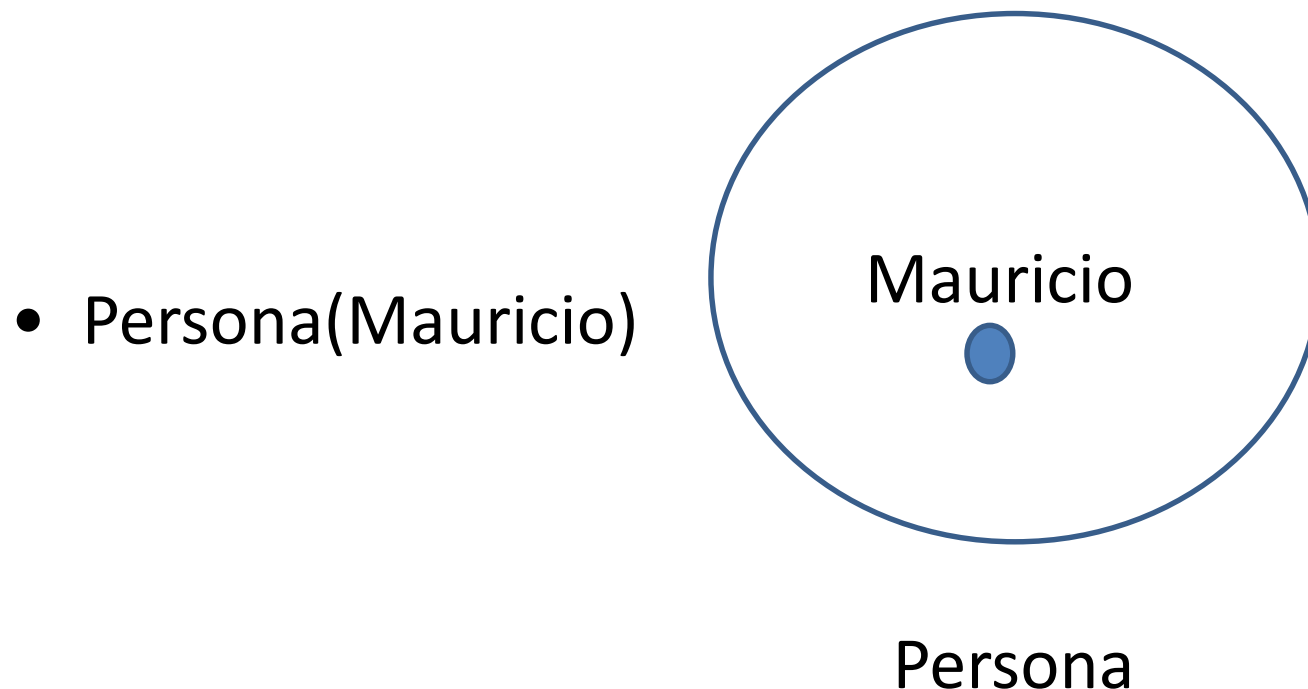


$$\text{Pizza} \sqcap \text{PizzaCubierta} \subseteq \perp$$

Algo que es un ***PizzaCubiertaTomate*** no puede ser una ***Pizza*** porque su superclase, ***PizzaCubierta***, es disjunto a ***Pizza***

Miembros de una Clase: Instancias

- **rdf:type**, que relaciona el recurso con su clase (el recurso se declara como una instancia de esa clase)
- ex:mauricio **rdf:type** ex:Persona .
- ex:mauricio **a** ex:Persona .



Inicio Tutorial 1 –OWL

Agregando Clases

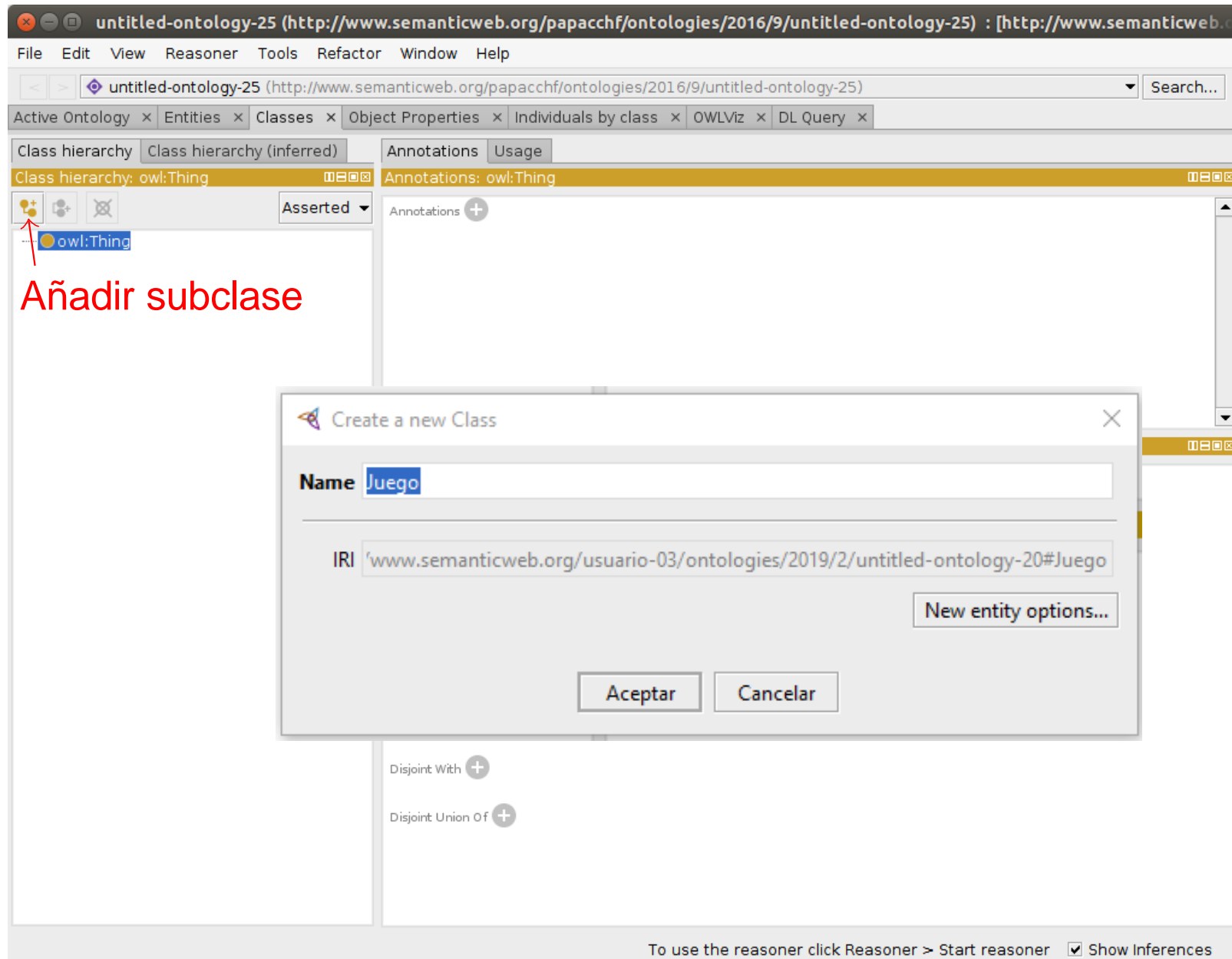
Ontología sobre Juegos de Video

Supongamos que queremos construir una ontología sobre videojuegos de la siguiente manera.

Clases	Clases	Propiedades	definiciones
<ul style="list-style-type: none">- Juego<ul style="list-style-type: none">- JuegoFamoso<ul style="list-style-type: none">- LoL- Ajedrez- Sudoku- Plataforma<ul style="list-style-type: none">- Windows- MacOSX- Linux	<ul style="list-style-type: none">- TipoJuego<ul style="list-style-type: none">- UnSoloJugador- MultiJugador- DeRoles- Elinea- DificultadJuego<ul style="list-style-type: none">- Difícil- Normal- Fácil	<ul style="list-style-type: none">tieneDificultadtienePlataformatieneTipo	<ul style="list-style-type: none">JuegoMultiPlataformaJuegoDifícilJuegoNormalJuegoFacilJuegoParaLinuxJuegoParaWindowsJuegoParaMacOSX...

Agregando clases

**Asegúrese de tener abierta la pestaña
"Classes" Window → Tabs → Classes**



Agregando clases

Asegúrese de tener abierta la pestaña "Classes"

Window → Tabs → Classes

The screenshot shows the Protégé ontology editor window titled 'untitled-ontology-20'. The 'Classes' tab is active. The 'Class hierarchy' panel on the left shows a tree structure with 'owl:Thing' as the root and 'Juego' as a child. The 'Annotations' panel on the right is empty. The 'Description' panel on the right shows a list of class axioms with plus signs next to them. Red arrows point to the 'Juego' class in the hierarchy with the labels 'Añadir Hermano' and 'Borrar Clase'.

Class hierarchy: Juego

owl:Thing

Juego

Annotations: Juego

Description: Juego

Equivalent To +

SubClass Of +

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

Disjoint Union Of +

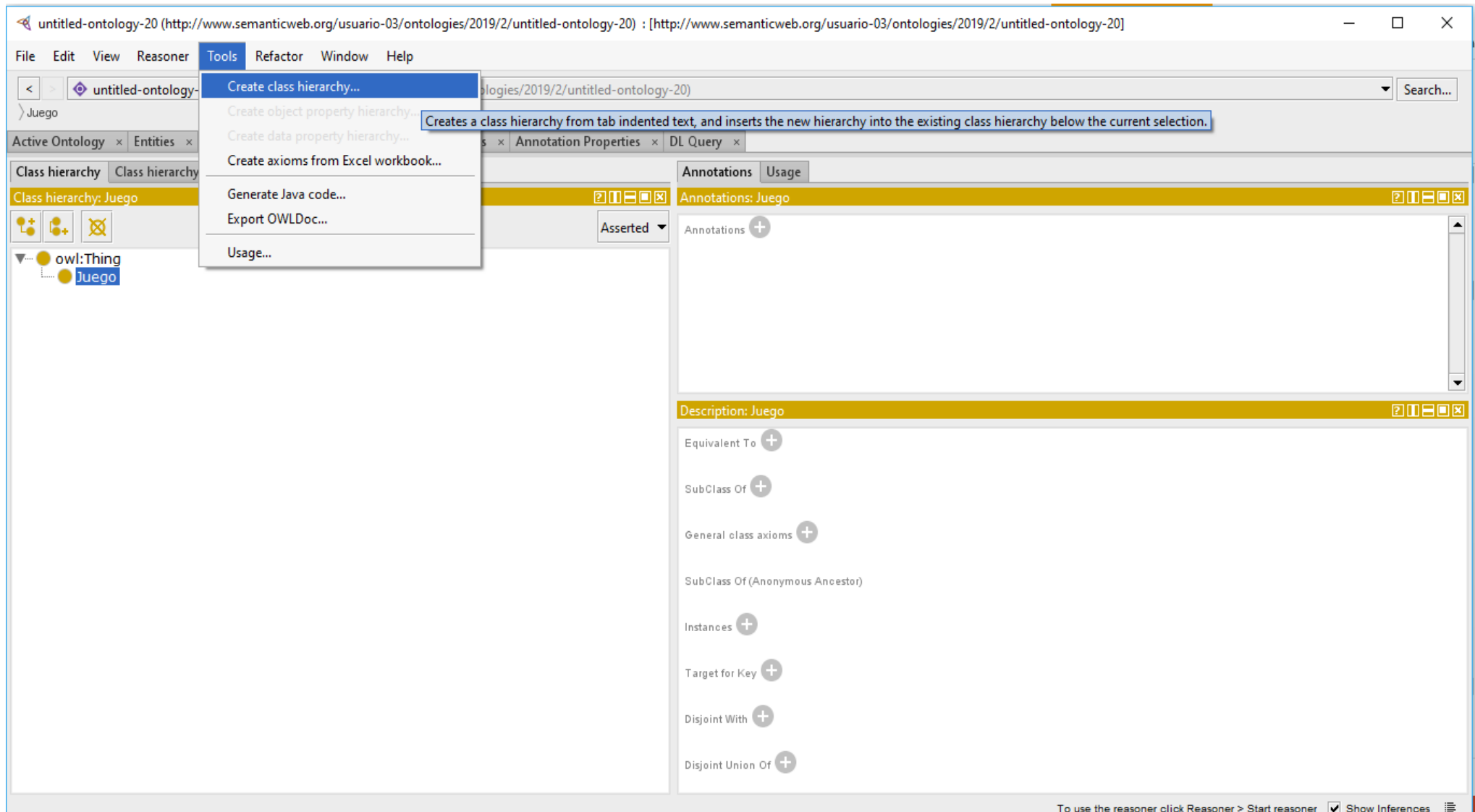
Añadir Hermano

Borrar Clase

Agregando Jerarquía de Clases

Permite acelerar el proceso de añadir clases.

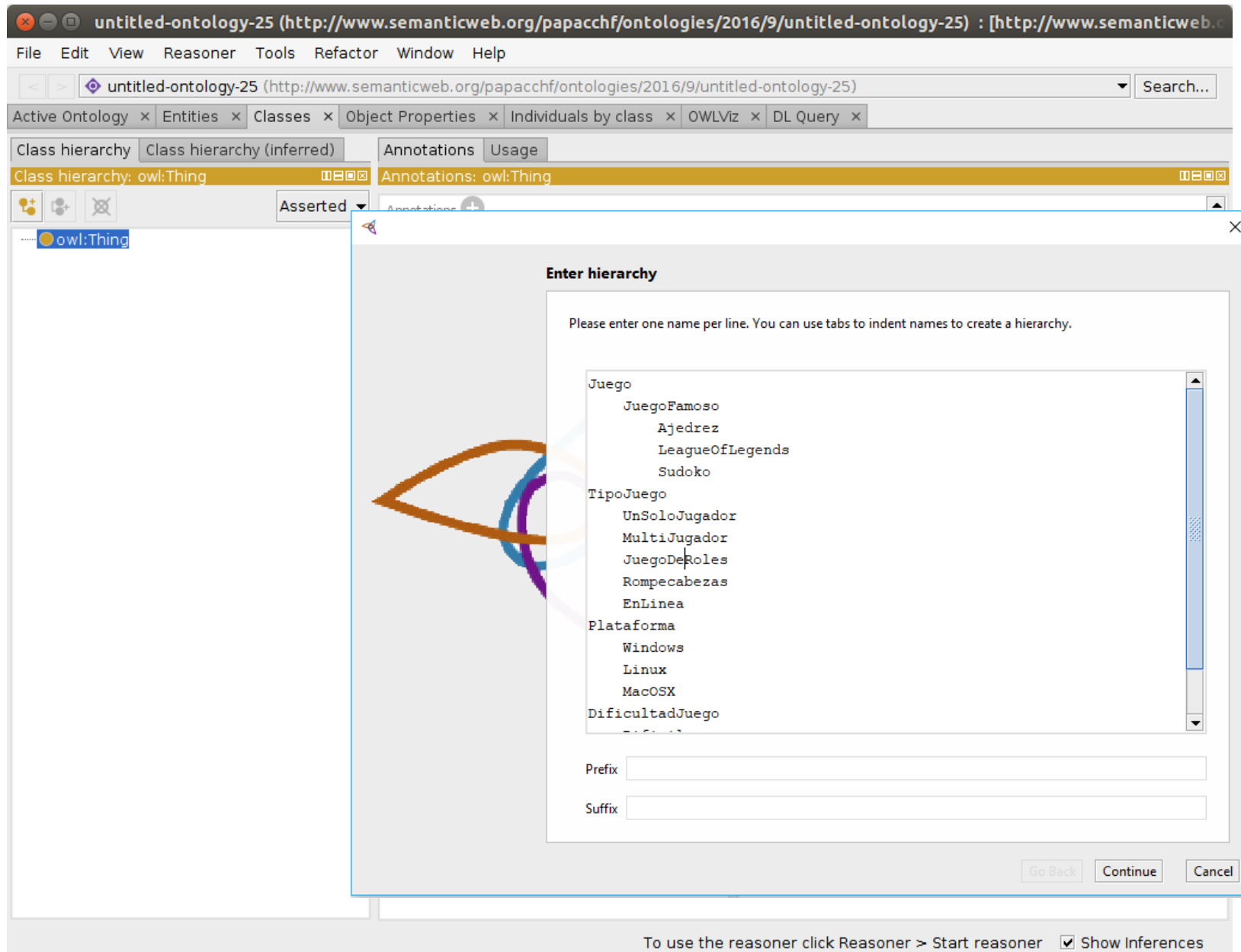
Tools → Create class hierarchy. . .



Agregando Jerarquía de Clases

Permite acelerar el proceso de añadir clases.

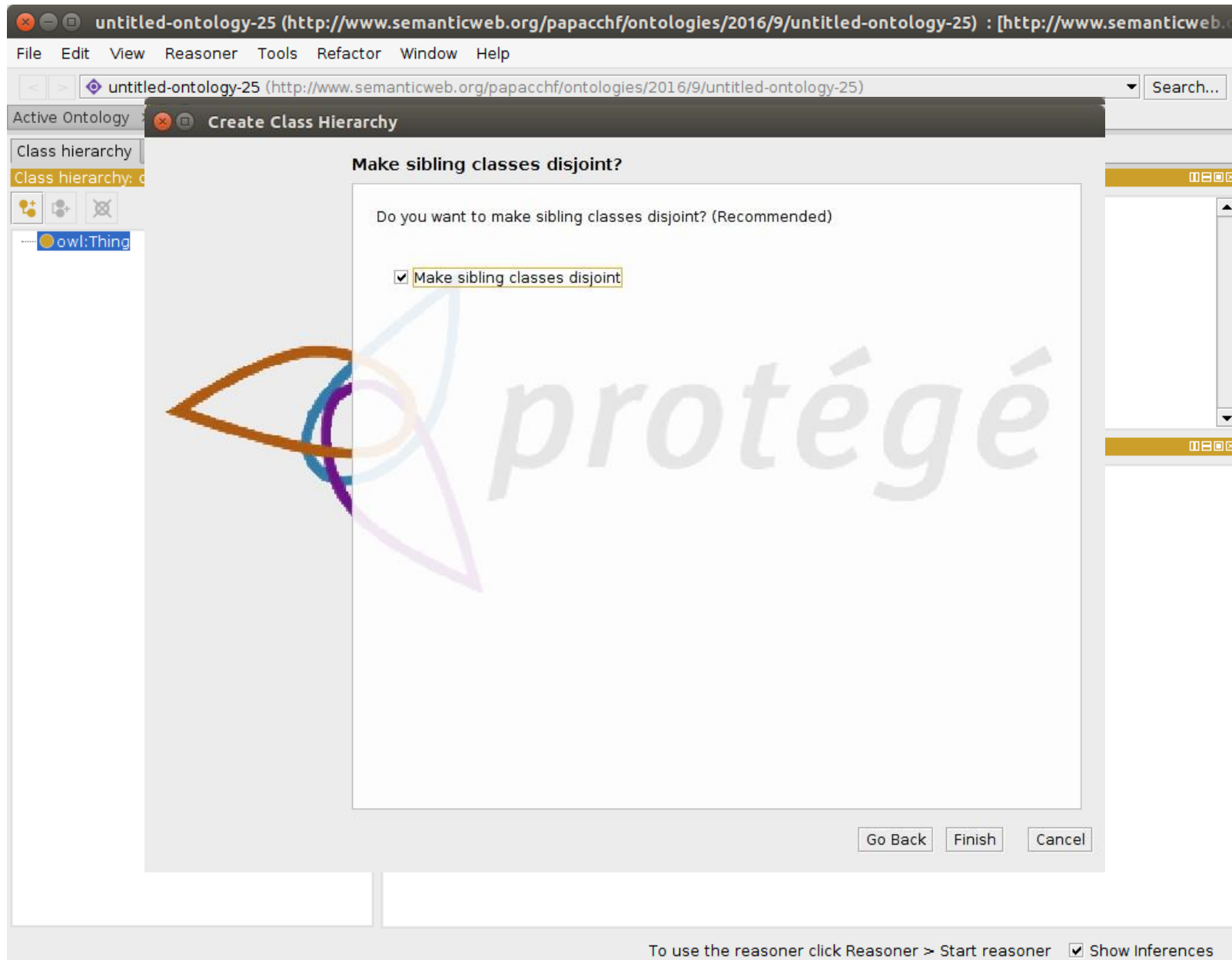
Tools → Create class hierarchy. . .



Agregando Jerarquía de Clases

Permite acelerar el proceso de añadir clases.

Tools → Create class hierarchy. . .



Agregando Jerarquía de Clases

Esto permite observar el uso de una clase particular.
Class hierarchy → Usage . . .

The screenshot shows a web application interface for managing an ontology. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. Below the menu is a search bar and a tabbed interface with the following tabs: Active Ontology, Entities, Classes, Object Properties, Data Properties, Annotation Properties, and DL Query. The 'Classes' tab is active, and the 'Class hierarchy' sub-tab is selected. The left pane displays the 'Class hierarchy: Juego' tree, showing a hierarchy starting from 'owl:Thing' and branching into 'DificultadJuego' (with sub-classes 'Difícil', 'Fácil', 'Normal'), 'Juego' (highlighted with a red arrow), 'JuegoFamoso' (with sub-classes 'Ajedrez', 'LeagueOfLegends', 'Sudoku'), 'Plataforma' (with sub-classes 'Linux', 'MacOSX', 'Windows'), and 'TipoJuego' (with sub-classes 'EnLinea', 'JuegoDeRoles', 'MultiJugador', 'Rompecabezas', 'UnSoloJugador'). The right pane shows the 'Usage: Juego' view, which is highlighted with a red box. It displays 'Found 12 uses of Juego' and lists several uses: 'DificultadJuego' (with 'DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego'), 'Juego' (with 'Class: Juego' and 'DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego'), 'JuegoFamoso' (with 'JuegoFamoso SubClassOf Juego'), and 'Plataforma'. Below the usage list is a 'Description: Juego' section with expandable sections for 'Equivalent To', 'SubClass Of', 'General class axioms', 'SubClass Of (Anonymous Ancestor)', 'Instances', and 'Target for Key'. A red arrow points to the 'Usage' tab in the top navigation bar. A red text overlay on the right side of the image reads: 'Axiomas que involucran la clase "Juego"'. At the bottom right, there is a footer with the text 'To use the reasoner click Reasoner > Start reasoner' and a checkbox for 'Show Inferences'.

Axiomas que involucran la clase "Juego"

Agregando Jerarquía de Clases

Es posible observar la descripción de una clase.

Class hierarchy → Usage . . .

The screenshot displays the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The address bar shows the ontology URL: <http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20>. The main workspace is divided into three panes:

- Left Pane (Class hierarchy):** Displays a tree view of the ontology classes. The 'Juego' class is highlighted with a red arrow. The hierarchy includes:
 - owl:Thing
 - DificultadJuego
 - Difícil
 - Facil
 - Normal
 - Juego (highlighted)
 - JuegoFamoso
 - Ajedrez
 - LeagueOfLegends
 - Sudoku
 - Plataforma
 - Linux
 - MacOSX
 - Windows
 - TipoJuego
 - EnLinea
 - JuegoDeRoles
 - MultiJugador
 - Rompecabezas
 - UnSoloJugador
- Middle Pane (Usage):** Shows the usage of the 'Juego' class. It lists 12 uses, including:
 - DificultadJuego (DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego)
 - Juego (Class: Juego, DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego)
 - JuegoFamoso (SubClassOf: Juego)
 - Plataforma
- Right Pane (Description: Juego):** Provides a detailed description of the 'Juego' class. It includes:
 - General class axioms
 - SubClass Of (Anonymous Ancestor)
 - Instances
 - Target for Key
 - Disjoint With: TipoJuego, Plataforma, DificultadJuego
 - Disjoint Union Of

Descripción de la clase "Juego"

Agregando Jerarquía de Clases

Es posible observar la descripción de una clase.

Class hierarchy → Usage . . .

The screenshot shows a Semantic Web editor interface with the following components:

- Menu Bar:** File, Edit, View, Reasoner, Tools, Refactor, Window, Help.
- Address Bar:** untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) : [http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20]
- Tab Bar:** Active Ontology, Entities, Classes, Object Properties, Data Properties, Annotation Properties, DL Query.
- Class hierarchy:** A tree view showing the hierarchy of classes. The 'Juego' class is highlighted with a red arrow. The hierarchy includes: owl:Thing, DificultadJuego (Facil, Normal, Dificil), Juego (JuegoFamoso, Ajedrez, LeagueOfLegends, Sudoku), Plataforma (Linux, MacOSX, Windows), and TipoJuego (EnLinea, JuegoDeRoles, MultiJugador, Rompecabezas, UnSoloJugador).
- Usage:** A list of 14 uses of the 'Juego' class, including 'DisjointClasses' and 'SubClassOf' relationships.
- Description: Juego:** A panel showing the description of the 'Juego' class. It includes a 'SubClass Of' section with 'owl:Thing' selected, and a 'General class axioms' section.

The text $Juego \subseteq T$ is written in red next to the 'SubClass Of' section.

Agregando Jerarquía de Clases

Es posible observar la descripción de una clase.

Class hierarchy → Usage . . .

The screenshot shows the Protégé ontology editor interface. The top menu bar includes File, Edit, View, Reasoner, Tools, Refactor, Window, and Help. The main window displays the 'untitled-ontology-20' ontology. The left pane shows the 'Class hierarchy' for 'Juego', with a red arrow pointing to the 'Juego' class. The right pane shows the 'Usage' of 'Juego', listing 14 uses. The bottom pane shows the 'Description' of 'Juego', including general class axioms, subClass Of (Anonymous Ancestor), instances, target for key, and disjoint with.

Class hierarchy: Juego

- owl:Thing
 - DificultadJuego
 - Difícil
 - Fácil
 - Normal
 - Juego
 - JuegoFamoso
 - Ajedrez
 - LeagueOfLegends
 - Sudoku
 - Plataforma
 - Linux
 - MacOSX
 - Windows
 - TipoJuego
 - EnLinea
 - JuegoDeRoles
 - MultiJugador
 - Rompecabezas
 - UnSoloJugador

Usage: Juego

Found 14 uses of Juego

- DificultadJuego
 - DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego
- Juego
 - Juego SubClassOf owl:Thing
 - DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego
 - Class: Juego
- JuegoFamoso
 - JuegoFamoso SubClassOf Juego

Description: Juego

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

- TipoJuego, Plataforma, DificultadJuego

Disjoint Union Of +

$Juego \sqcap DificultadJuego \sqsubseteq \perp$

$Juego \sqcap TipoJuego \sqsubseteq \perp$

$Juego \sqcap Plataforma \sqsubseteq \perp$

Agregando Jerarquía de Clases

Es posible observar la descripción de una clase.

Class hierarchy → Usage . . .

The screenshot shows a web-based 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 search bar. The main workspace is divided into several panes. On the left, the 'Class hierarchy' pane shows a tree structure starting from 'owl:Thing'. A red arrow points to the 'Juego' class, which has several subclasses: 'DificultadJuego' (with subclasses 'Difícil', 'Facil', 'Normal'), 'JuegoFamoso' (with subclasses 'Ajedrez', 'LeagueOfLegends', 'Sudoku'), 'Plataforma' (with subclasses 'Linux', 'MacOSX', 'Windows'), and 'TipoJuego' (with subclasses 'EnLinea', 'JuegoDeRoles', 'MultiJugador', 'Rompecabezas', 'UnSoloJugador'). The 'Usage' pane on the right shows 'Found 14 uses of Juego'. It lists 'DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego' and 'Juego SubClassOf owl:Thing'. Below this, the 'Description: Juego' pane is visible, showing various axiom types like 'Equivalent To', 'SubClass Of', 'General class axioms', 'SubClass Of (Anonymous Ancestor)', 'Instances', 'Target for Key', 'Disjoint With', and 'Disjoint Union Of'. A red box highlights the 'Description: Juego' pane. Inside this box, a red arrow points to the 'Borrar axioma' (Delete axiom) button, and another red arrow points to the 'Editar axioma' (Edit axiom) button. The bottom status bar indicates 'To use the reasoner click Reasoner > Start reasoner' and 'Show Inferences'.

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20) : [http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20]

File Edit View Reasoner Tools Refactor Window Help

untitled-ontology-20 (http://www.semanticweb.org/usuario-03/ontologies/2019/2/untitled-ontology-20)

Juego

Active Ontology x Entities x Classes x Object Properties x Data Properties x Annotation Properties x DL Query x

Class hierarchy Class hierarchy (inferred) Annotations Usage

Class hierarchy: Juego

Usage: Juego

Show: ☒ this ☒ disjoints ☒ named sub/superclasses

Found 14 uses of Juego

DisjointClasses: DificultadJuego, Juego, Plataforma, TipoJuego

Juego SubClassOf owl:Thing

Description: Juego

Equivalent To +

SubClass Of +

owl:Thing

General class axioms +

SubClass Of (Anonymous Ancestor)

Instances +

Target for Key +

Disjoint With +

TipoJuego, Plataforma, DificultadJuego

Disjoint Union Of +

Borrar axioma

Editar axioma

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

Fin Tutorial 1 –OWL Agregando Clases

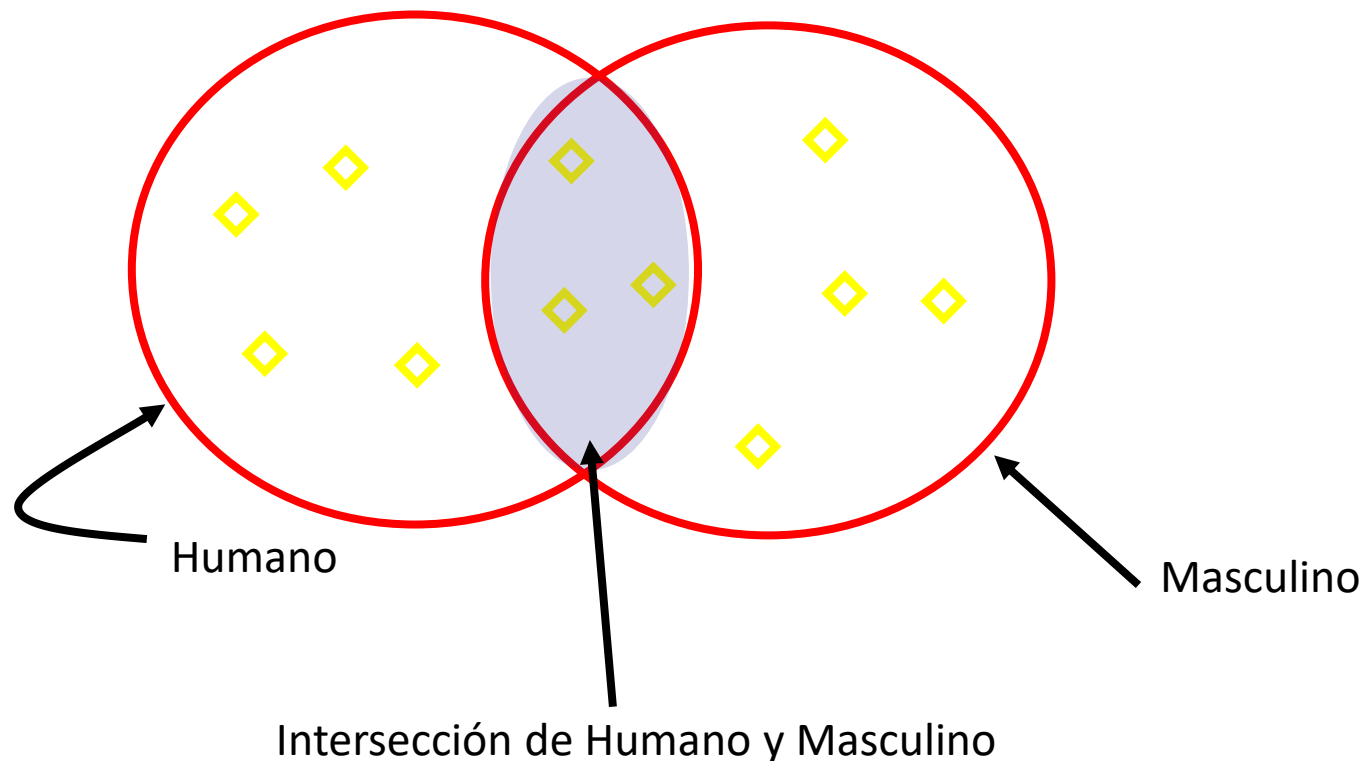
Clases Complejas en OWL

Clases Complejas

- (OWL DL) provee constructores con los cuales se pueden formar clases basadas en operaciones básicas de conjuntos:
 - Intersección
 - Union
 - Complemento
- Clases Disjuntas
-

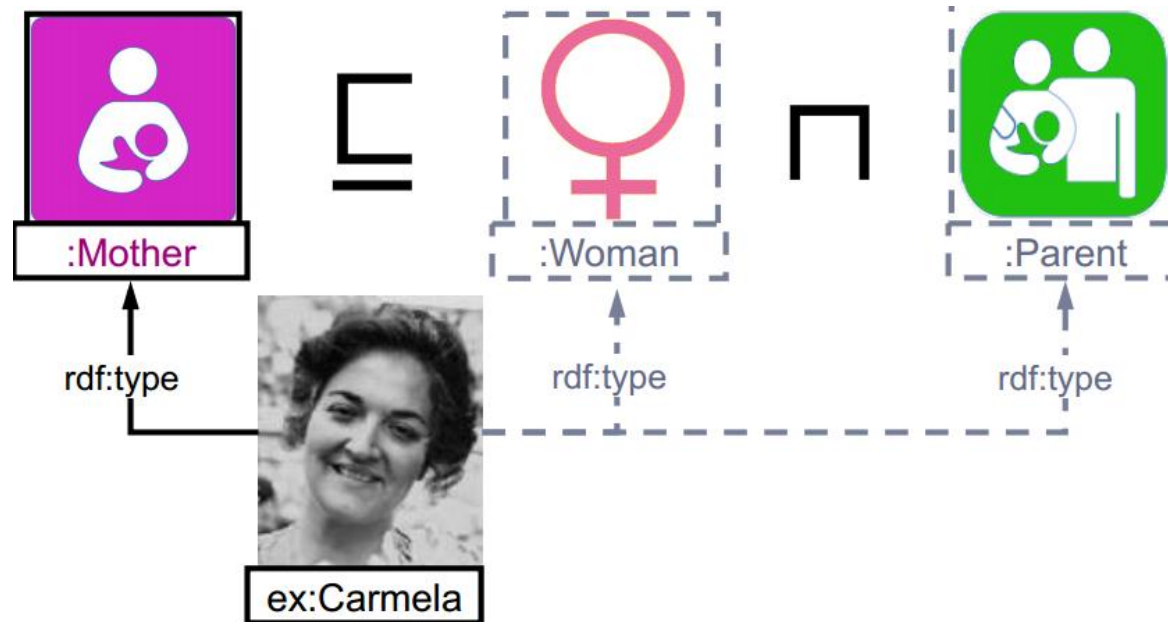
Clases Complejas: Intersección de Clases

- Las instancias/individuos de la intersección de dos clases son simultáneamente instancias de ambas clases
- owl:intersectionOf



Clases Complejas: Intersección de Clases

- Ejemplo: owl:intersectionOf



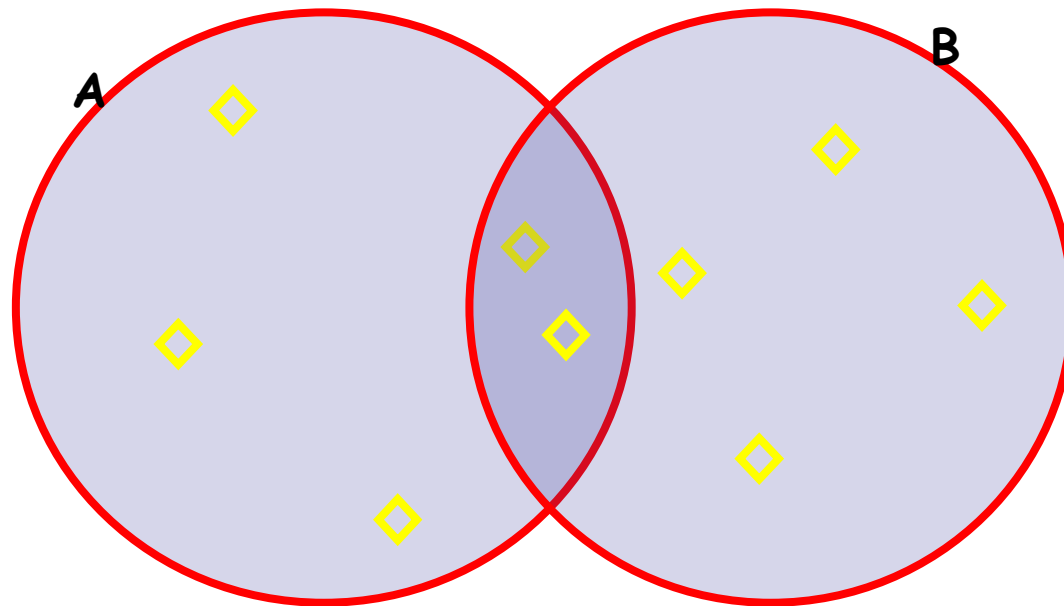
`ex:Carmela rdf:type :Mother .`

`:Mother rdfs:subClassOf [owl:intersectionOf (:Woman :Parent)]`

`ex:Carmela rdf:type :Woman , :Parent .`

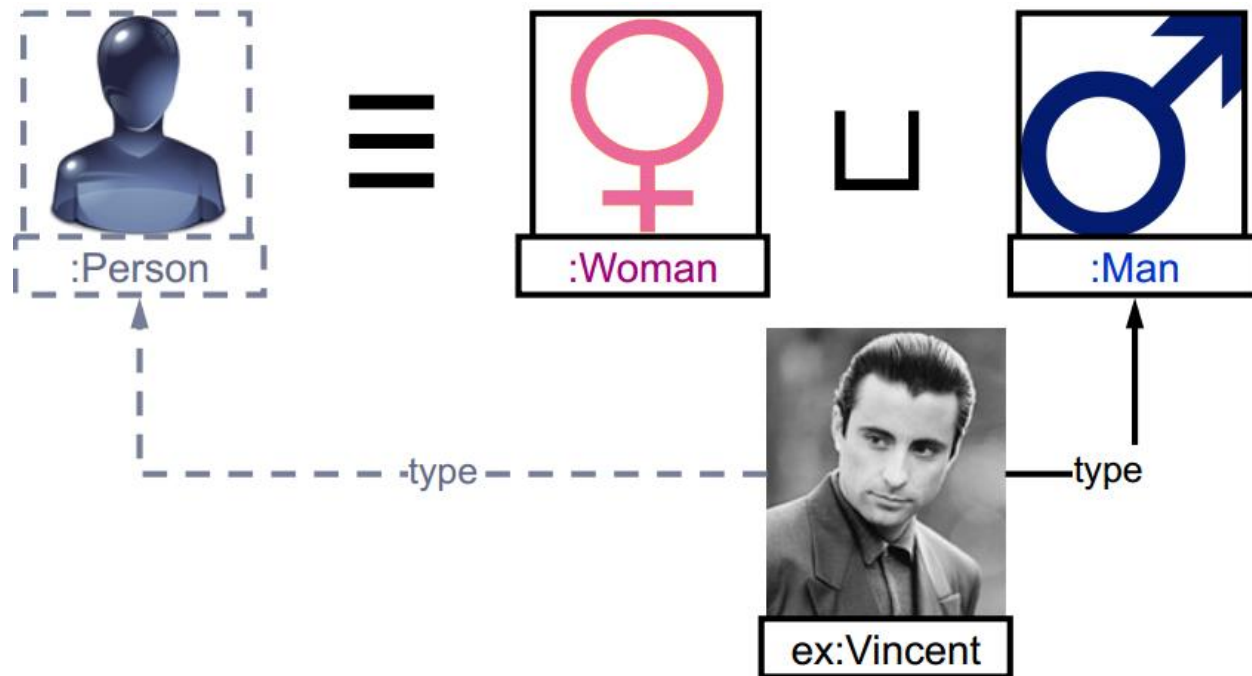
Clases Complejas: Unión de Clases

- Instancias/Individuos de la unión de dos clases son la instancia de una o ambas clases.
- owl:unionOf



Clases Complejas: Unión de Clases

- Ejemplo: owl:unionOf



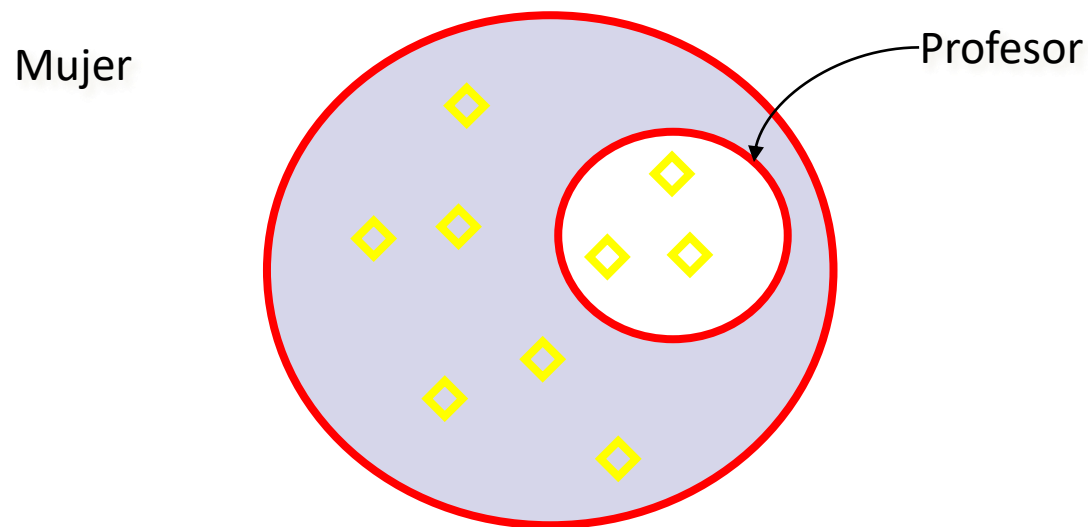
`ex:Vincent rdf:type :Man .`

`:Person owl:equivalentClass [owl:unionOf (:Woman :Man)]`

`ex:Vincent rdf:type :Person .`

Clases Complejas: Complemento

Una clase complemento se especifica negando otra clase. Contendrá los individuos que no están en la clase negada.



Mujer and (not Profesor)

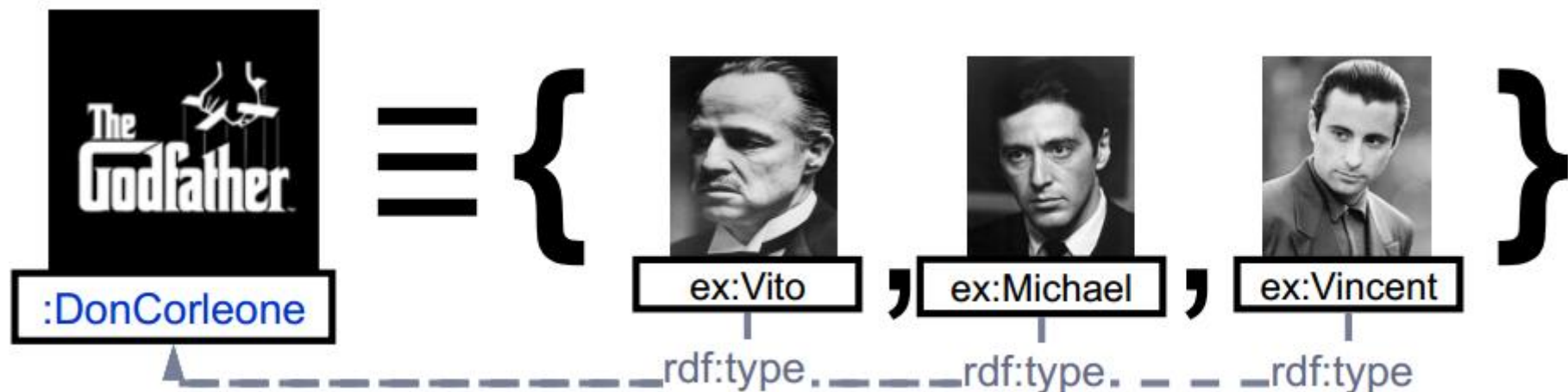
Clases Complejas: Clases Enumeradas

- OWL provee la forma de especificar una clase vía una enumeración directa de sus miembros
 - **owl:oneOf**.
- Ejemplo:

```
<owl:Class rdf:ID="ColorVino">  
  <rdfs:subClassOf rdf:resource="#DescripcionVino"/>  
  <owl:oneOf rdf:parseType="Collection">  
    <WineColor rdf:about="#Blanco" />  
    <WineColor rdf:about="#Rosa" />  
    <WineColor rdf:about="#Rojo" />  
  </owl:oneOf>  
</owl:Class>
```

Clases Complejas: Clases Enumeradas

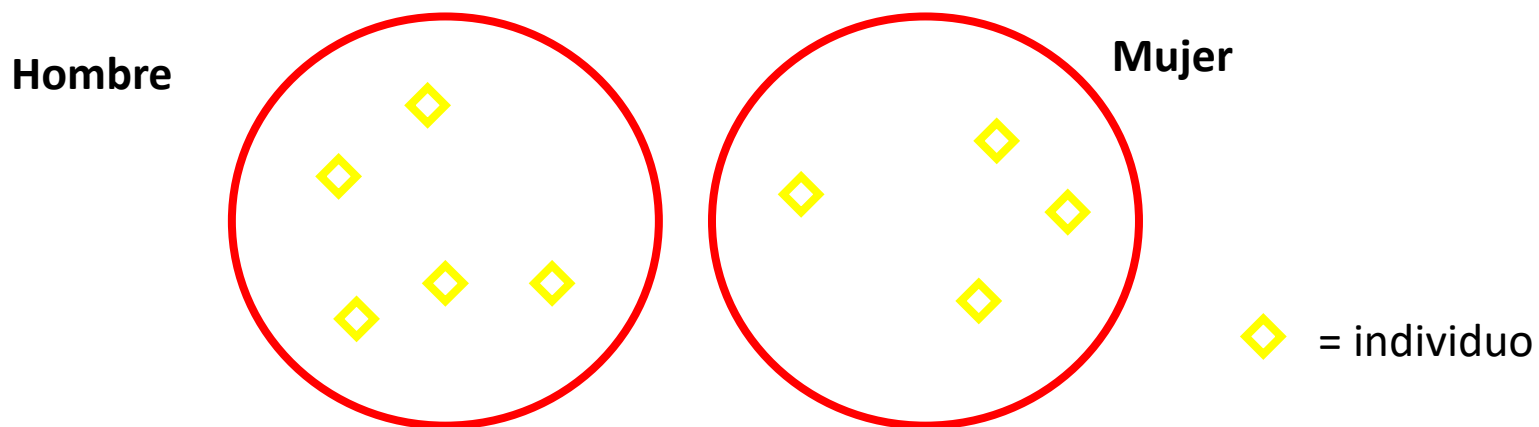
- Ejemplo: `owl:oneOf`



```
:DonCorleone owl:equivalentClass  
    [ owl:oneOf (ex:Vito ex:Michael ex:Vincent) ]  
ex:Vito rdf:type :DonCorleone .  
ex:Michael rdf:type :DonCorleone .  
ex:Vincent rdf:type :DonCorleone .
```

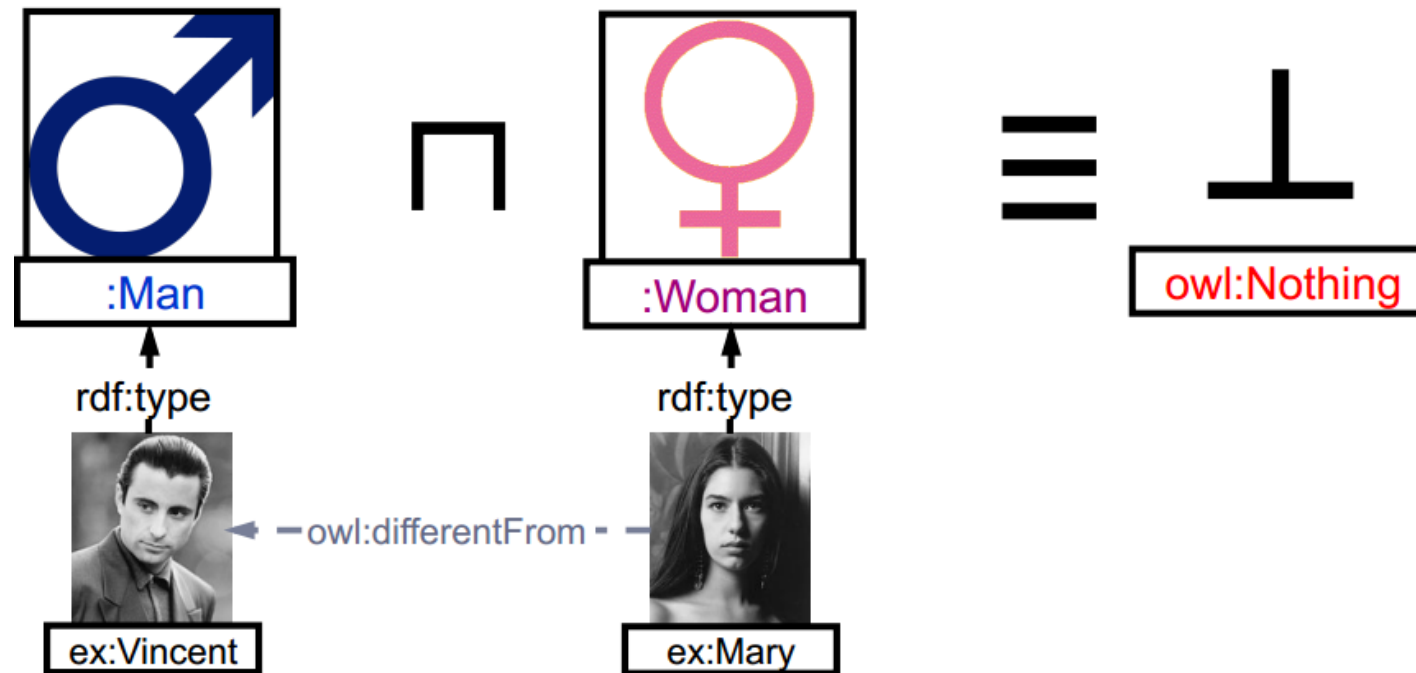
Clases Complejas: Clases Disjuntas

- OWL asume que las clases se superponen
- La desunión de un conjunto de clases puede ser expresado usando **owl:disjointWith**
 - Un individuo que es miembro de una clase no puede ser simultáneamente una instancia de otra.



Clases Complejas: Clases Disjuntas

- Ejemplo `owl:disjointWith`



```
ex:Vincent rdf:type :Man . ex:Mary rdf:type :Woman .  
:Man owl:disjointWith :Woman .  
ex:Mary owl:differentFrom ex:Vincent .
```