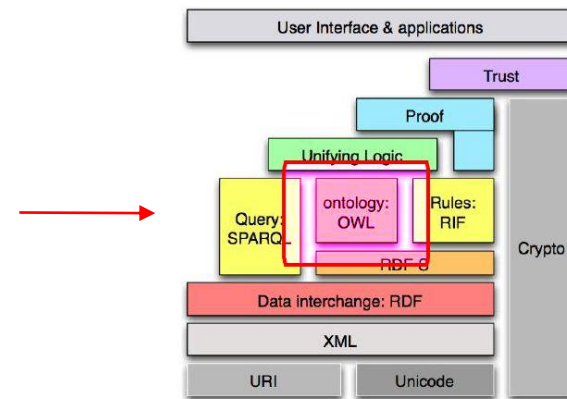


Tecnologías para la Web Semántica

Arquitectura IV
OWL

OWL



Web Ontology Language: Lenguaje para definir ontologías.

- ▶ Estandar web.
- ▶ Construido sobre RDF para procesar información en la web.
- ▶ Diseñado para ser interpretado por computadoras, no para ser leído por las personas.
- ▶ Utiliza sintaxis XML.
- ▶ OWL es similar a RDF pero:
 - Lenguaje más potente
 - Provee mayor interoperabilidad
 - Mayor vocabulario
 - Mejor sintaxis



Lenguajes de Representación de Ontologías

La arquitectura de Niveles de T. Berners Lee

- ▶ XML permite estructurar documentos según vocabularios definidos por el usuario
- ▶ RDF proporciona un modelo para describir aserciones sobre recursos Web.
- ▶ RDF Schema proporciona primitivas para organizar objetos en jerarquías (ontologías simples)
- ▶ OWL permite expresar relaciones más complejas entre objetos (ontologías complejas)

Lenguajes de Representación de Ontologías

Limitaciones de RDF(S)

- ▶ Expresar la **disyunción de clases**.
 - Ej. No podemos especificar que hombre y mujer son clases disjuntas
- ▶ Definir **clases como combinación de otras** (unión, intersección o complemento).
 - Ej. no podemos especificar que la clase persona es la unión de hombre y mujer
- ▶ Expresar **restricciones sobre la cardinalidad de propiedades**.
 - Ej no podemos especificar que una asignatura debe tener al menos un profesor
- ▶ Describir **propiedades específicas de las propiedades**.
 - Ej. No podemos expresar que una propiedad es una función (valor único), que es transitiva o que es la inversa de otra

Necesitamos un lenguaje ontológico mas rico que RDFS

Lenguajes de ontologías

Requerimientos para lenguajes de Representación de Ontologías (Extensión de RDF(S)):

- ▶ Una **sintaxis bien definida**: condición necesaria para información procesable por máquinas.
- ▶ Una **semántica formal**: prerequisite para soporte de razonamiento.
- ▶ **Soporte de razonamiento**: verificar la consistencia de la ontología.
- ▶ **Suficiente poder expresivo**.

OWL

- ▶ La información Web tiene un significado preciso.
- ▶ La información Web puede ser procesada por computadoras.
- ▶ Las computadoras pueden integrar la información de la web.
- ▶ OWL está diseñado para:
 - Proveer una forma común para procesar el contenido de la web en vez de mostrarlo.
 - Permitir la lectura por aplicaciones en vez de humanos.

OWL

- ▶ En una ontología OWL encontramos:
 - **Clases** + jerarquía de clases
 - **Propiedades** (Slots) / values
 - **Relaciones**. Relaciones entre clases. (herencia, disyunción, equivalencia)
 - **Restricciones**. Restricciones sobre las propiedades (tipo, cardinalidad)
 - Características de propiedades (transitividad, ...)
 - Anotaciones
 - Individuos
- ▶ Tareas de razonamiento: clasificación, chequeo de consistencia

OWL. Sintaxis

- ▶ Basada en XML
- ▶ Una ontología en owl comienza con la declaración del elemento raíz RDF
- ▶ Incluye los espacios de nombre para las ontologías RDF, RDFS y OWL

<rdf:RDF

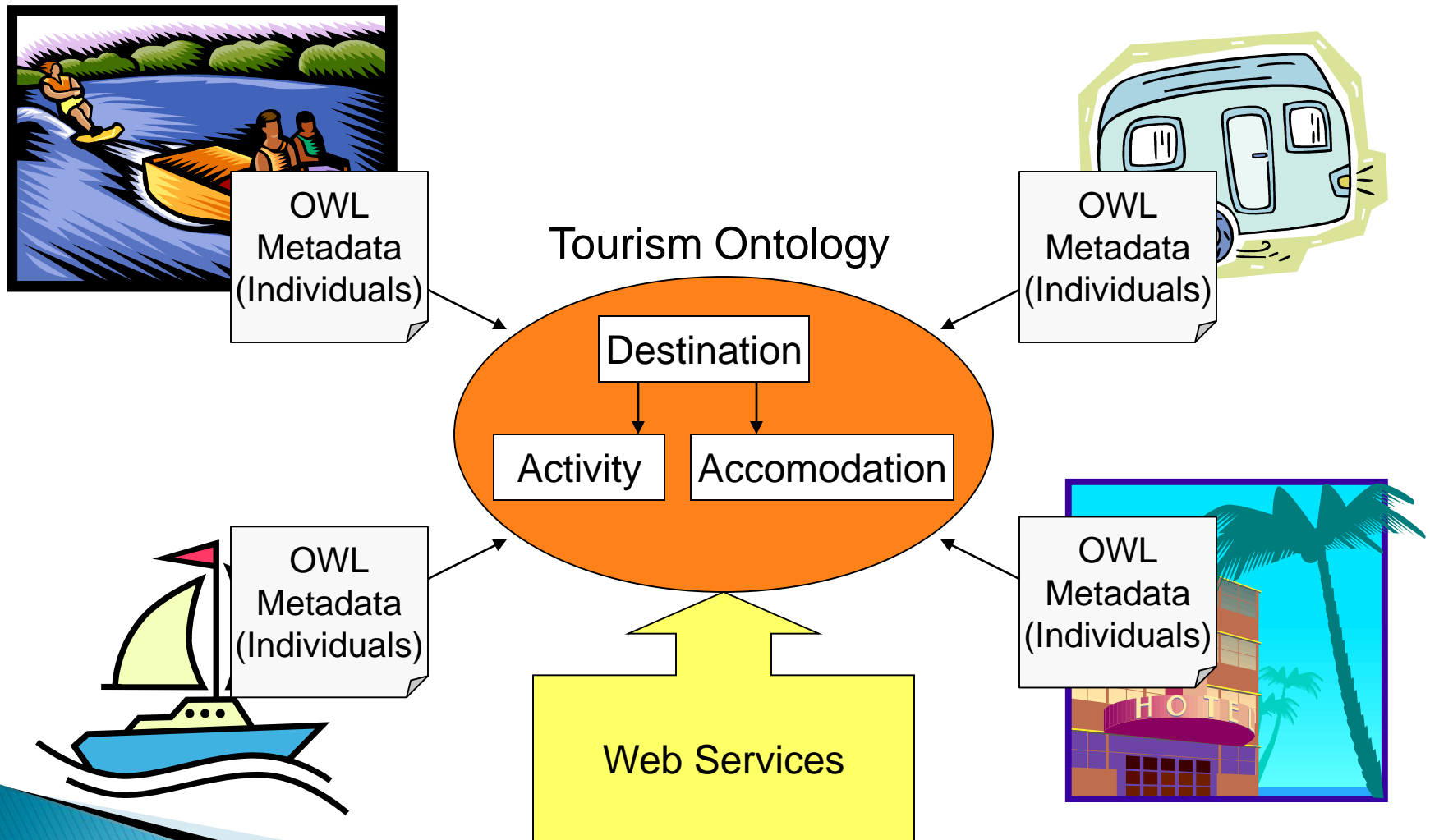
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"

xmlns:xsd="http://www.w3.org/2001/XMLSchema#"






xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"

xmlns:owl = "http://www.w3.org/2002/07/owl#"

Tourism Ontology



OWL. Sintaxis

-  Individuals (e.g., “FourSeasons”)
-  Properties
 -  ObjectProperties (references)
 -  DatatypeProperties (simple values)
-  Classes (e.g., “Hotel”)

OWL. Sintaxis

► Individuos

- Representan objetos en el dominio.
- Cosas específicas.

 Sydney

 BondiBeach

 SydneysOlympicBeach

```
<Region rdf:ID="CentralCoastRegion" />
```

equivalent to:

```
<owl:Thing rdf:ID="CentralCoastRegion" />  
<owl:Thing rdf:about="#CentralCoastRegion">  
  <rdf:type rdf:resource="#Region" />  
</owl:Thing>
```

OWL instancias

► Individuos

- Se definen utilizando solamente vocabulario RDF

```
<Person rdf:ID="Adam">  
  <rdfs:label>Adam</rdfs:label>  
  <rdfs:comment>Adam is a person.</rdfs:comment>  
  <age><xsd:integer rdf:value="13"/></age>  
  <shoesize><xsd:decimal rdf:value="9.5"/></shoesize>  
</Person>
```

OWL. Sintaxis

► Propiedades

En Owl hay dos clases de propiedades:

- **Propiedades de objetos:** define relaciones binarias entre objetos

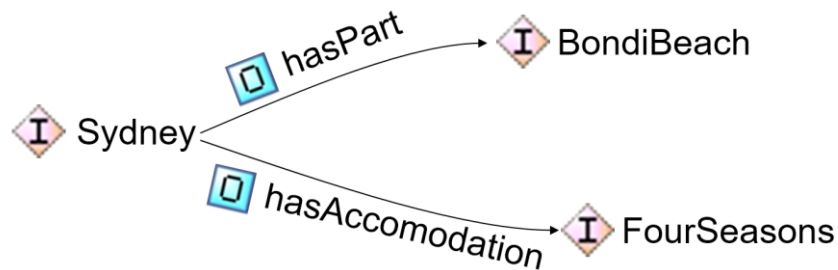
Ej. is-TaughtBy, supervises

- **Propiedades de tipos de dato:** relaciona objetos con valores de tipos de dato

Ej. phone, title, age, etc.

OWL. Sintaxis

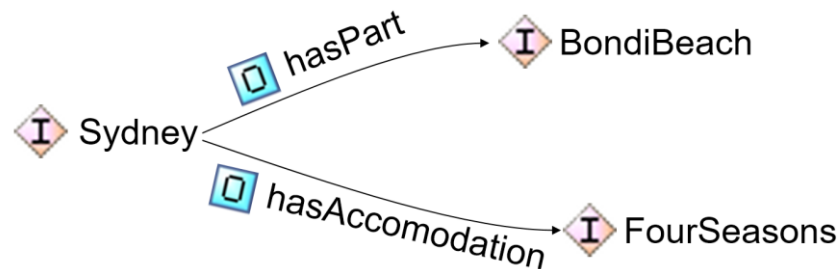
- ▶ Propiedades
 - Vincula dos individuos.
 - Establece relaciones (0..n, n..m)



OWL. Sintaxis

► Propiedades

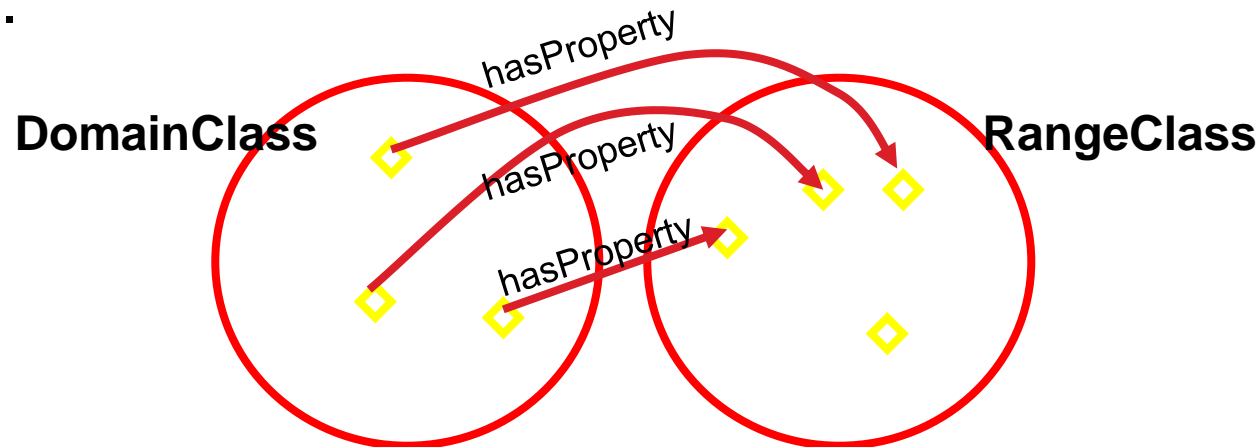
- Vincula dos individuos.
- Establece relaciones (0..n, n..m)



```
<owl:ObjectProperty rdf:ID="hasPart">  
  <rdfs:domain rdf:resource="#Destination" />  
  <rdfs:range rdf:resource="#Beaches" />  
</owl:ObjectProperty>
```

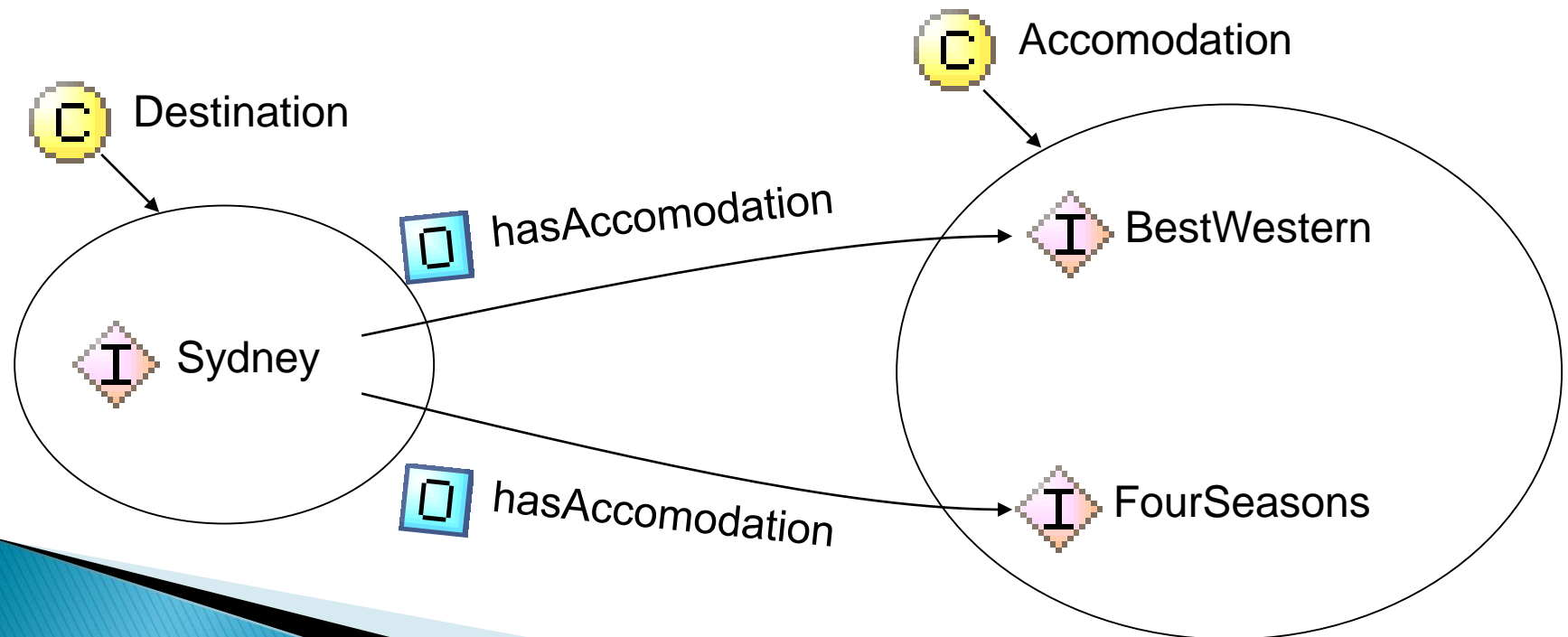
OWL. Sintaxis

- ▶ Si una relación es:
subject_individual → hasProperty → object_individual
- ▶ El **dominio** es la clase del **sujeto** individuo
- ▶ El **rango** es:
 - la clase del **objeto** individuo
 - un tipo de dato si hasProperty es una propiedad de tipo de dato.



OWL. Sintaxis

- ▶ Características de las propiedades
 - Dominio: “lado izquierdo de la relación” (Destination)
 - Range: “lado derecho” (Accommodation)



OWL. Sintaxis

► Propiedades. Dominio.

- Individuos pueden tomar valores solamente valores de las propiedades con el que se establezca un “matcheo” con el dominio.
 - “Only Destinations can have Accommodations”

OWL. Sintaxis

- ▶ Propiedades definen relaciones binarias entre objetos

```
<owl:ObjectProperty rdf:ID="estaCasadoCon">  
  <rdf:type rdf:resource="&owl;FunctionalProperty" />  
  <rdfs:domain rdf:resource="#Mujer" />  
  <rdfs:range rdf:resource="#Hombre" />  
</owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:ID="isTaughtBy">  
  <owl:domain rdf:resource="#course"/>  
  <owl:range rdf:resource="#academicStaffMember"/>  
  <rdfs:subPropertyOf rdf:resource="#involves"/>  
</owl:ObjectProperty>
```

OWL. Sintaxis

Propiedad, Dominio & Rango

```
<owl:ObjectProperty rdf:ID="madeFromGrape">  
  <rdfs:domain rdf:resource="#Wine"/>  
  <rdfs:range rdf:resource="#WineGrape"/>  
</owl:ObjectProperty>
```

```
<owl:Thing rdf:ID="LindemansBin65Chardonnay"> <madeFromGrape  
  rdf:resource="#ChardonnayGrape" />  
</owl:Thing>
```

=> LindemansBin65Chardonnay is a wine

OWL. Sintaxis

- ▶ Propiedades de tipos de dato:
 - Relaciona objetos con valores de tipos de dato.
 - OWL no tiene tipos predefinidos de datos
 - Permite utilizar tipos de dato de XML Schema
 - Atributos

```
<owl:DatatypeProperty rdf:ID="arrivalDate">  
    <rdfs:domain rdf:resource="#Travel"/>  
    <rdfs:range rdf:resource="&xsd:date"/>  
</owl:DatatypeProperty>
```

OWL. Sintaxis

► Restricción de propiedades: Ejemplo de Cardinalidad

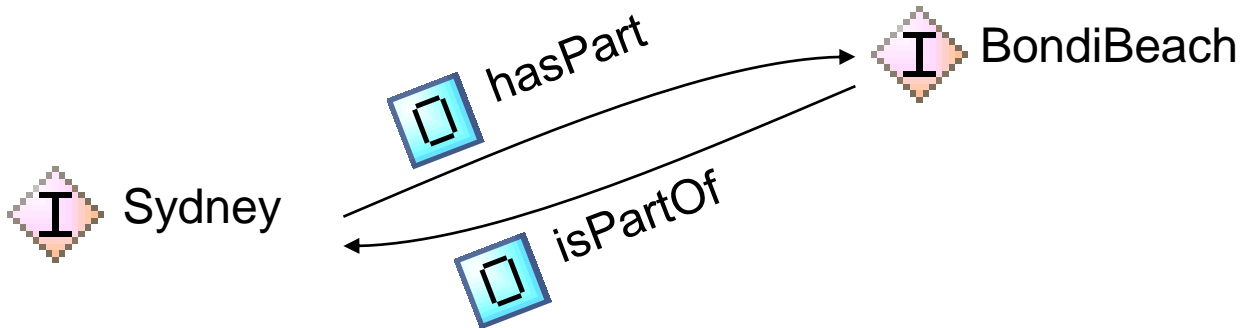
```
<owl:Class rdf:ID="Wine">
  <rdfs:subClassOf rdf:resource="&food;PotableLiquid"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#madeFromGrape"/>
      <owl:minCardinality
rdf:datatype="&xsd;nonNegativeInteger">1 </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf> ...
</owl:Class>
```

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#madeFromGrape"/>
  <owl:minCardinality rdf:datatype="&xsd;nonNegativeInteger">1 </owl:minCardinality>
</owl:Restriction>
```

OWL. Sintaxis

► Propiedades inversas:

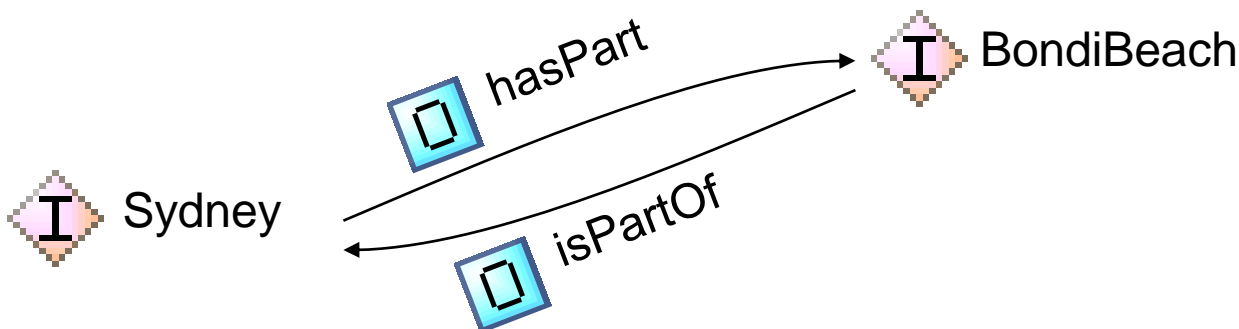
- Representan relaciones bidireccionales
- *Si se le agrega un valor a una de las propiedades también se le agrega a la otra.*



OWL. Sintaxis

► Propiedades inversas:

- Representan relaciones bidireccionales
- *Si se le agrega un valor a una de las propiedades también se le agrega a la otra.*



```
<owl:ObjectProperty rdf:ID="hasPart">  
  <rdf:type rdf:resource="<rdf:type rdf:resource=" />  
</owl:ObjectProperty="&owl;FunctionalProperty">  
<owl:ObjectProperty rdf:ID="isPartOf"> <owl:inverseOf rdf:resource="#hasPart" />  
</owl:ObjectProperty>
```

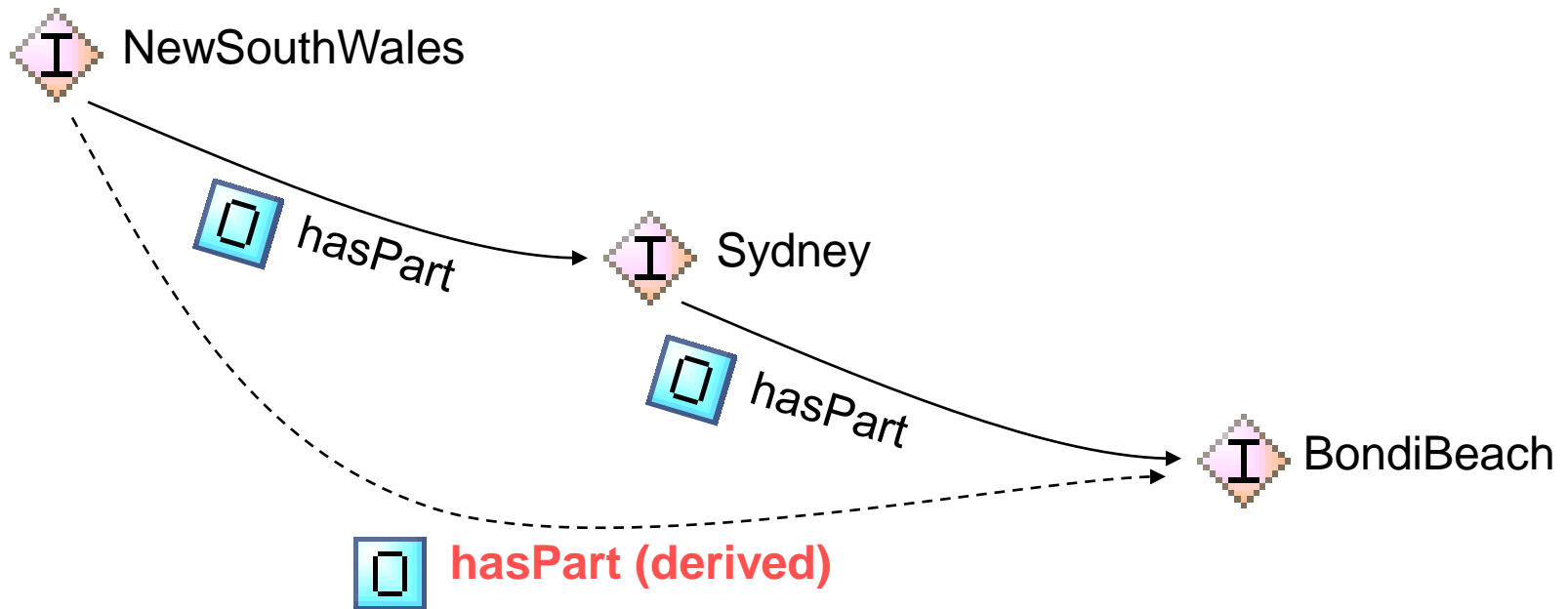

OWL propiedades inversas

```
<owl:ObjectProperty rdf:ID="teaches">  
  <rdfs:range rdf:resource="#course"/>  
  <rdfs:domain rdf:resource="#academicStaffMember"/>  
  <owl:inverseOf rdf:resource="#isTaughtBy"/>  
</owl:ObjectProperty>
```

OWL. Sintaxis

► Propiedades transitivas:

- Si A está relacionado con B y B está relacionado con C, entonces A también está relacionado con C
- Muy utilizado en relaciones del tipo parte-de



OWL. Sintaxis

► Propiedades transitivas:

```
<owl:ObjectProperty rdf:ID="locatedIn">
  <rdf:type rdf:resource="&owl;TransitiveProperty" />
  <rdfs:domain rdf:resource="&owl;Thing" />
  <rdfs:range rdf:resource="#Region" />
</owl:ObjectProperty>

<Region rdf:ID="SantaCruzMountainsRegion">
  <locatedIn rdf:resource="#CaliforniaRegion" />
</Region>

<Region rdf:ID="CaliforniaRegion">
  <locatedIn rdf:resource="#USRegion" />
</Region>
```

OWL. Sintaxis


- ▶ Ejemplo de subpropiedades:

```
<owl:ObjectProperty rdf:ID="hasWineDescriptor"> <rdfs:domain rdf:resource="#Wine" />  
    <rdfs:range rdf:resource="#WineDescriptor" />  
</owl:ObjectProperty>
```

```
owl:ObjectProperty rdf:ID="hasColor"> <rdfs:subPropertyOf rdf:resource="#hasWineDescriptor" />  
    <rdfs:range rdf:resource="#WineColor" /> ...  
</owl:ObjectProperty>
```

OWL. Sintaxis

- ▶ Propiedades de tipos de dato:
 - Vinculan individuos con valores primitivos (integers, floats, strings, Booleans, etc)
 - Anotaciones sin significado

 Sydney
hasSize = 4,500,000 isCapital = true rdfs:comment = “Don’t miss the opera house”

OWL. Sintaxis

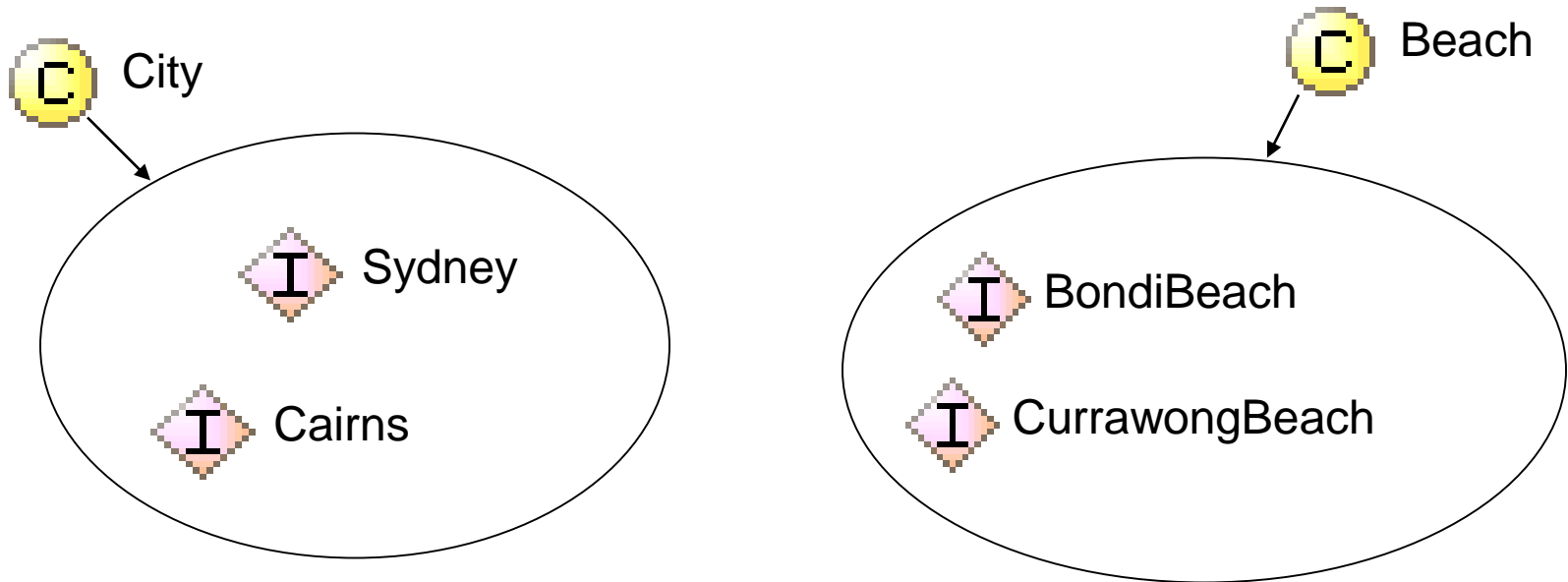
- ▶ Propiedades de tipos de dato:

```
<owl:DatatypeProperty rdf:ID="age">  
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema  
    #nonNegativeInteger"/>  
</owl:DatatypeProperty>
```

OWL. Sintaxis

► Clases

- Conjunto de individuos con características communes.
- Los individuos son instancias de al menos una clase



OWL. Sintaxis

► Clases

- Las clases se definen utilizando un elemento owl:class
- **owl:Class** es subclase de **rdfs:Class**

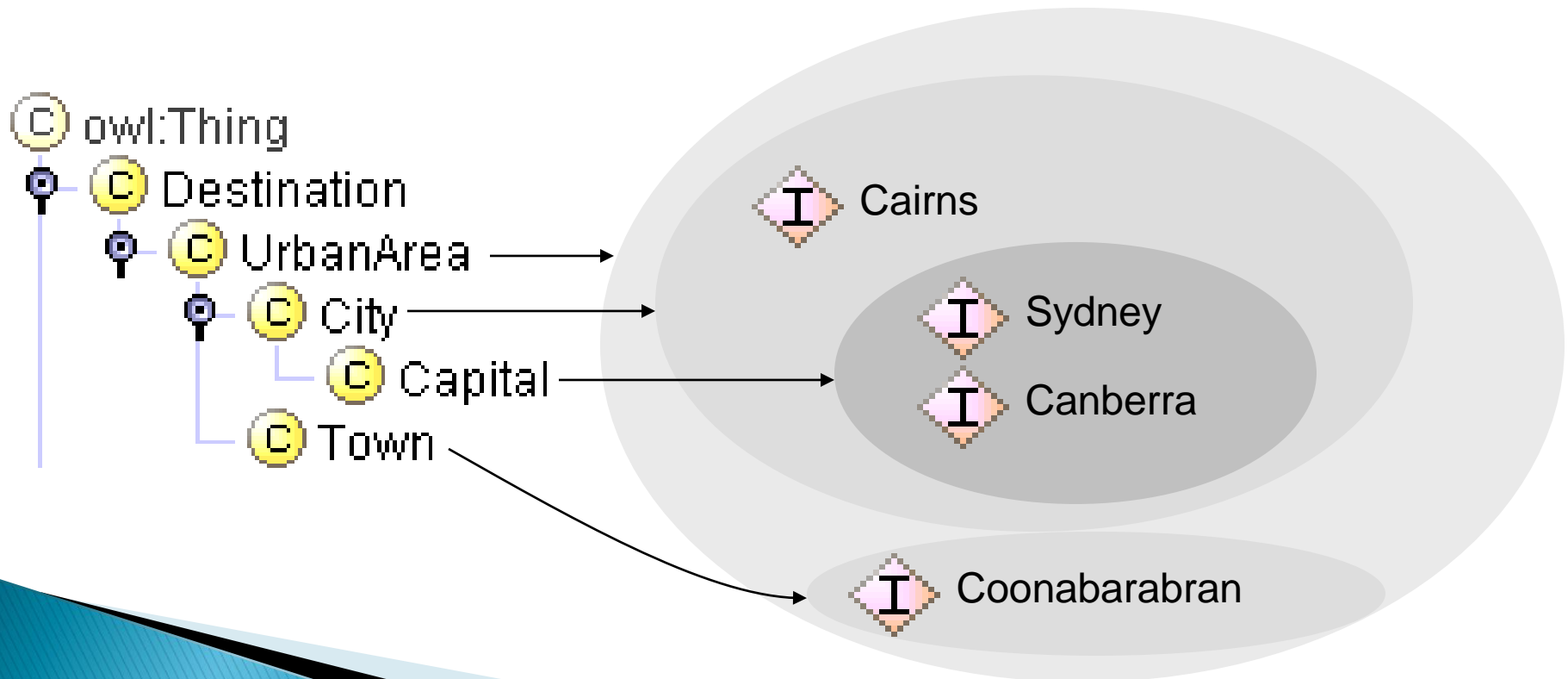
```
<owl:Class rdf:ID="profesorAsociado">  
  <rdfs:subClassOf rdf:resource="#miembroStaffAcademico"/>  
</owl:Class>
```

```
<owl:Class rdf:ID="Winery"/>  
<owl:Class rdf:ID="Region"/>  
<owl:Class rdf:ID="ConsumableThing"/>
```


OWL. Sintaxis

► Relación de superclase

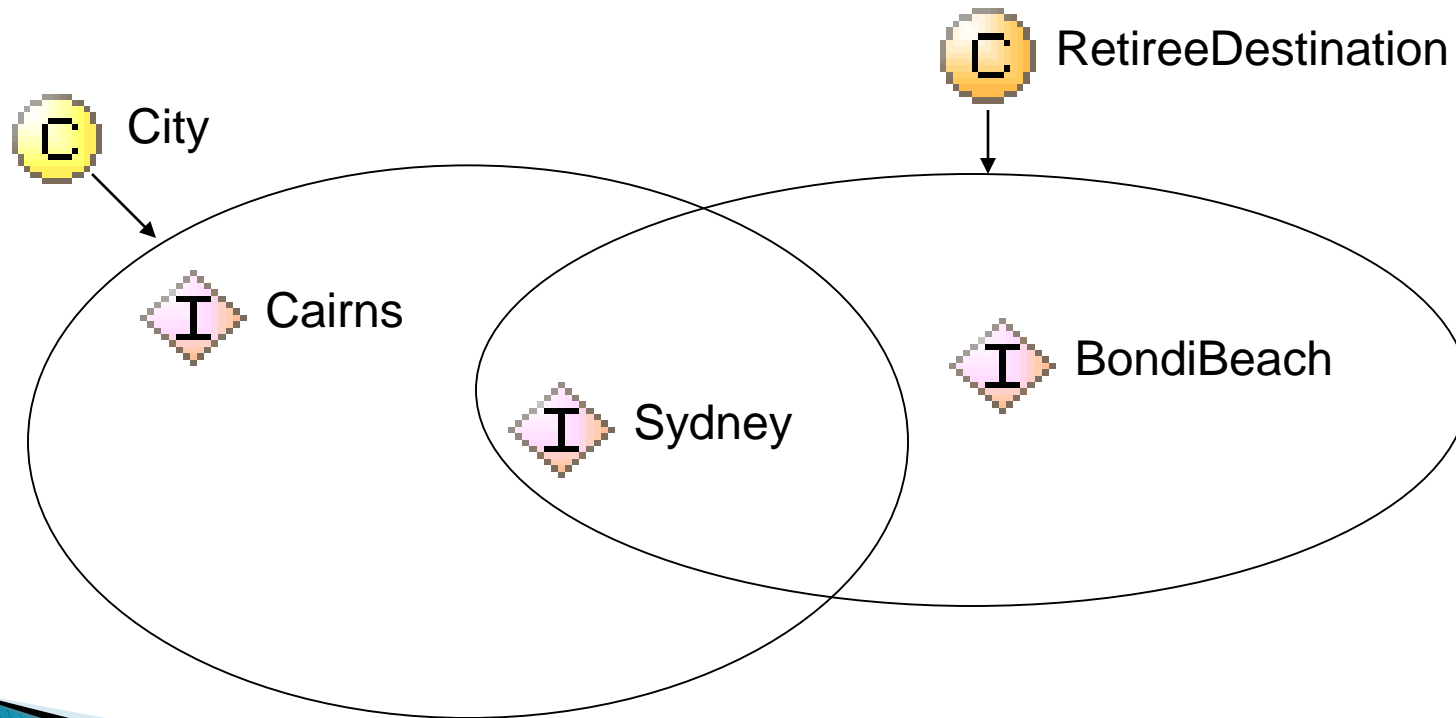
- Clases pueden organizarse en jerarquías
- Las instancias directas de una subclase son también instancias (indirectas) de la superclase



OWL. Sintaxis

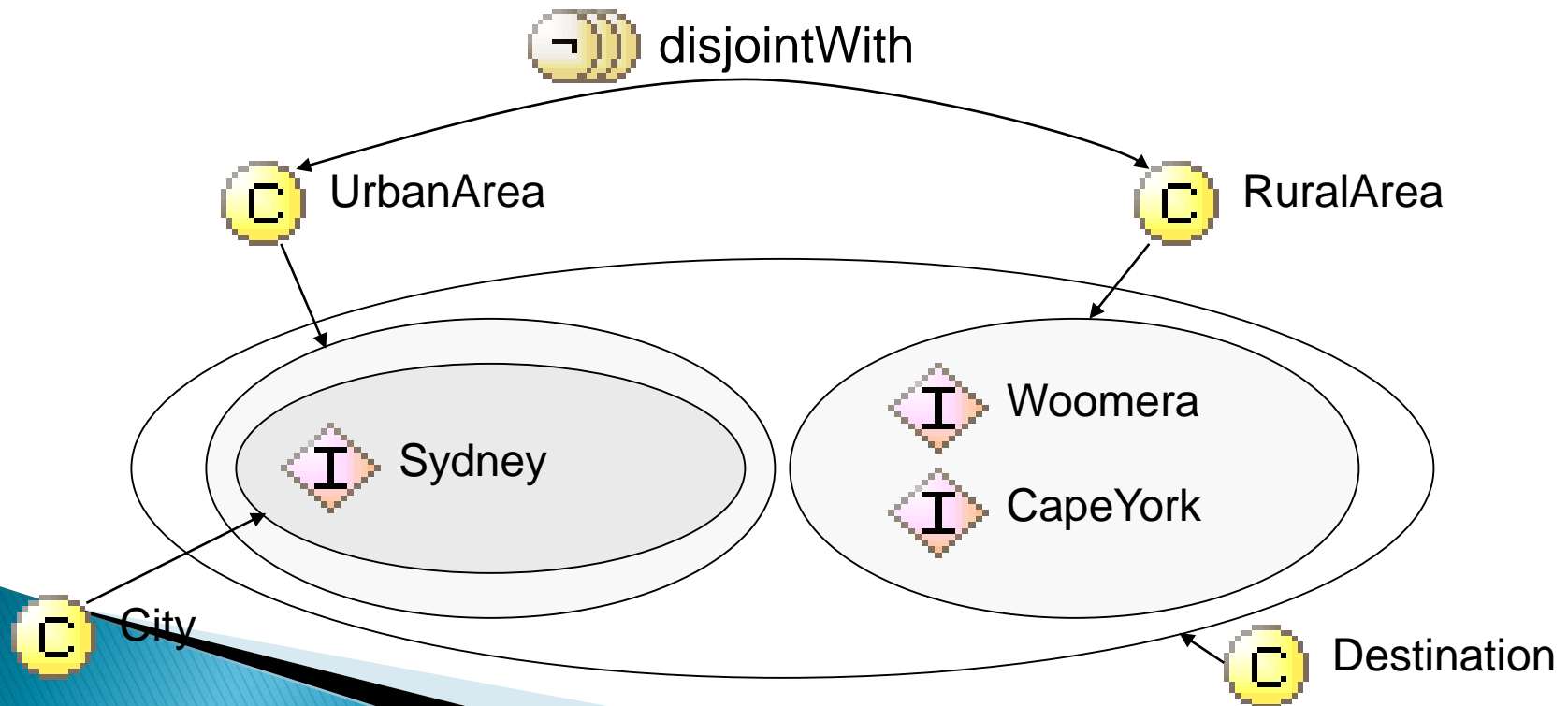
► Clase

- Las clases pueden superponerse arbitrariamente



OWL – Sintaxis

- ▶ Disyunción de clases.
 - Se define usando owl:disjointWith
 - Todas las clases podrían potencialmente solaparse
 - En muchos casos queremos estar seguros que no compartan instancias



OWL – Sintaxis

- ▶ Disyunción de clases

```
<owl:Class rdf:about="#Man"> <owl:disjointWith rdf:resource="#Woman"/>  
</owl:Class>
```

OWL – Sintaxis

► Clases equivalents

- owl:equivalentClass define equivalencia de clases

```
<owl:Class rdf:ID="faculty">  
    <owl:equivalentClass rdf:resource=    "#academicStaffMember"/>  
</owl:Class>
```

```
< owl:equivalentProperty  
    <owl:ObjectProperty rdf:ID="lecturesIn">  
        <owl:equivalentProperty    rdf:resource="#teaches"/>  
</owl:ObjectProperty>
```

OWL – Sintaxis

► Clases vs. Individuos

◦ Niveles de representación:







- En determinados contextos una clase puede ser considerada una instancia de otra cosa.
- Grape, conjunto de *grape varieties*. CabernetSauvignonGrape es una instancia de esta clase, pero podría ser considerada una clase, el conjunto de todas las actuales Cabernet Sauvignon grapes.

◦ Subclase vs. instancia: es fácil confundir la relación *instance-of* con la relación *subclass*!

- CabernetSauvignonGrape como instancia de Grape, o subclase de Grape.
- Pero: la clase Grape es el conjunto de todas *grape varieties*, cualquier subclase debería ser un subconjunto.
- CabernetSauvignonGrape es una instancia de Grape, no describe un subconjunto de Grape varieties, es un grape variety.




OWL. Sintaxis

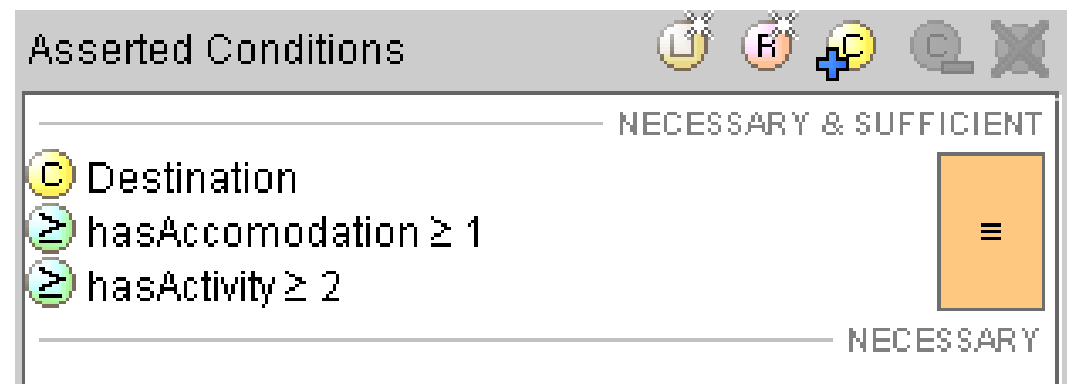
► Restricciones

- Definen condiciones para valores de propiedad
 -  allValuesFrom especifica cuantificación universal
 -  someValuesFrom especifica cuantificación existencial
 -  hasValue especifica un valor
 -  minCardinality
 -  maxCardinality
 -  cardinality
- Una clase anónima consta de todos los individuos que cumplen la condición

OWL. Sintaxis

► Restricciones de cardinalidad

- Significado: La propiedad debe tener como máximo/mínimo/exactamente X valores
-  es el atajo para  y 
- Ejemplo: Un FamilyDestination es un Destination que tiene como mínimo un Accomodation y como mínimo 2 Activities



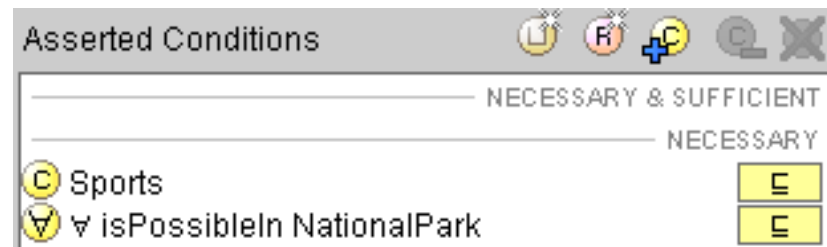
OWL. Sintaxis

► Propiedades

- Una (restricción) clase se adjunta a un elemento **owl:Restriction**
- Este elemento contiene un elemento **owl:onProperty** y una o más **declaraciones de restricción**
- Un tipo define **restricciones de cardinalidad**

OWL. Sintaxis

- ▶ Restricciones: allValuesFrom
 - Significado: Todos los valores de la propiedad deben ser de un cierto tipo.
 - Warning: También individuos sin valores completan esta condición (satisfacción trivial)
 - Ejemplo: Hiking is a Sport that only is possible in NationalParks



OWL. Sintaxis

► Propiedades AllValuesFrom

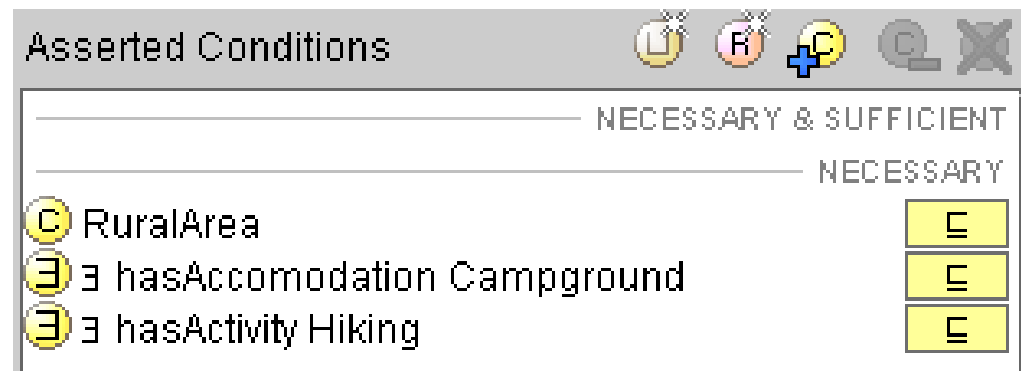
```
<owl:Class rdf:about="#firstYearCourse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#isTaughtBy"/>
      <owl:allValuesFrom
        rdf:resource="#Professor"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

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

OWL. Sintaxis

► Propiedades someValuesFrom

- Significado: Como mínimo un valor de la propiedad debe ser de un determinado tipo
- Otros pueden existir también.
- Ejemplo: A NationalPark is a RuralArea that has at least one Campground and offers at least one Hiking opportunity



OWL. Sintaxis

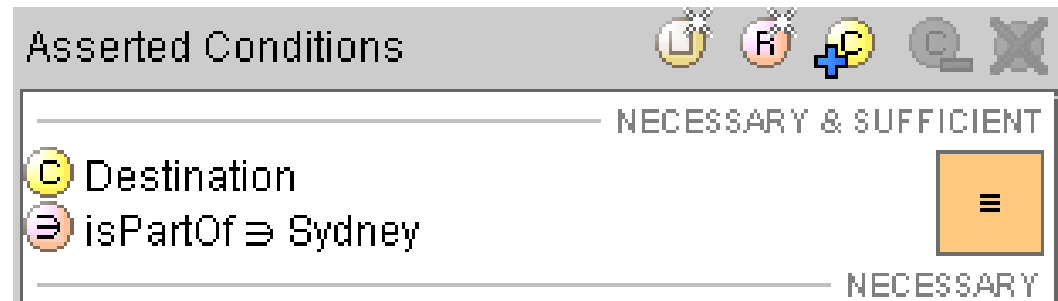
► Propiedades someValuesFrom

```
<owl:Class rdf:about="#academicStaffMember">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#teaches"/>
      <owl:someValuesFrom rdf:resource=
"#undergraduateCourse"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

OWL. Sintaxis

► Propiedades hasValue

- Significado: Como mínimo uno de los valores de la propiedad es un valor cierto
- Similar a someValuesFrom pero con individuos y valores primitivos
- Ejemplo: A PartOfSydney is a Destination where one of the values of the isPartOf property is Sydney



OWL. Sintaxis

► Propiedades hasValue

- Significado: Como mínimo uno de los valores de la propiedad es un valor cierto
- Similar a someValuesFrom pero con individuos y valores primitivos
- Ejemplo: A PartOfSydney is a Destination where one of the values of the isPartOf property is Sydney

```
<owl:Class rdf:about="#mathCourse">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource=
        "#isTaughtBy"/>
      <owl:hasValue rdf:resource=
        "#949352"/>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

OWL. Sintaxis

- ▶ Restricciones de cardinalidad
 - Podemos especificar número mínimo y máximo utilizando **owl:minCardinality** y **owl:maxCardinality**
 - Es posible especificar un número preciso usando el mismo número para mínimo y máximo
 - Por conveniencia, OWL ofrece también **owl:cardinality**

```
<owl:Class rdf:about="#course">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#isTaughtBy"/>
      <owl:minCardinality rdf:datatype=
"&xsd;nonNegativeInteger">
        1
      </owl:minCardinality>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```


OWL. Constructores

Conjunción

Constructor	DL Syntax	Example	FOL Syntax
intersectionOf	$C_1 \sqcap \dots \sqcap C_n$	Human \sqcap Male	$C_1(x) \wedge \dots \wedge C_n(x)$
unionOf	$C_1 \sqcup \dots \sqcup C_n$	Doctor \sqcup Lawyer	$C_1(x) \vee \dots \vee C_n(x)$
complementOf	$\neg C$	\neg Male	$\neg C(x)$
oneOf	$\{x_1\} \sqcup \dots \sqcup \{x_n\}$	{john} \sqcup {mary}	$x = x_1 \vee \dots \vee x = x_n$
allValuesFrom	$\forall P.C$	\forall hasChild.Doctor	$\forall y.P(x, y) \rightarrow C(y)$
someValuesFrom	$\exists P.C$	\exists hasChild.Lawyer	$\exists y.P(x, y) \wedge C(y)$
maxCardinality	$\leq nP$	≤ 1 hasChild	$\exists \leq n y.P(x, y)$
minCardinality	$\geq nP$	≥ 2 hasChild	$\exists \geq n y.P(x, y)$

```

<owl:Class>
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class rdf:about="#Human">
    <owl:Class rdf:about="#Male">
  </owl:intersectionOf>
</owl:Class>

```

C es un concepto (class); P es un rol (property); x_i es un individuo/nominal

OWL. Axiomas

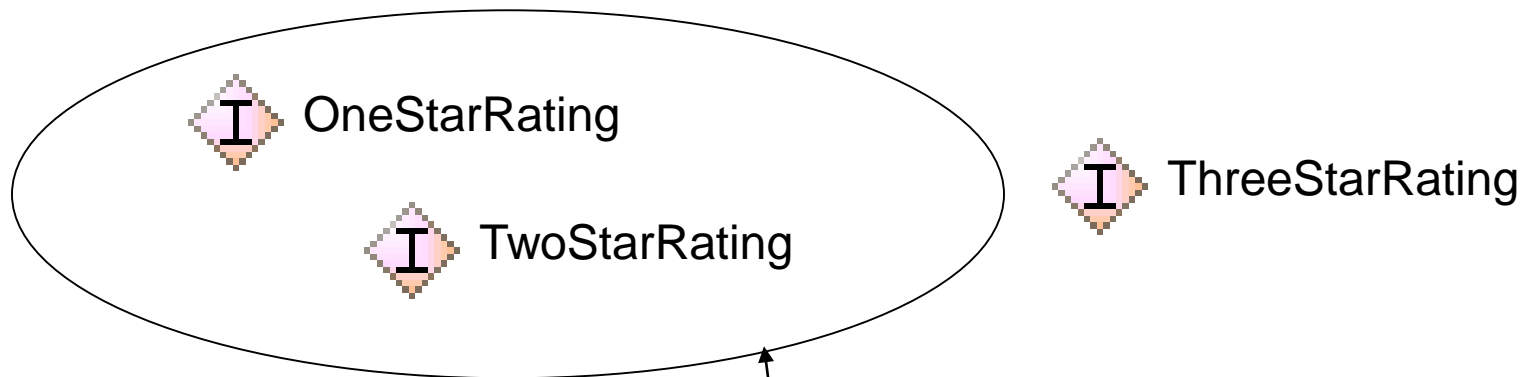
OWL Syntax	DL Syntax	Example
subClassOf	$C_1 \sqsubseteq C_2$	Human \sqsubseteq Animal \sqcap Biped
equivalentClass	$C_1 \equiv C_2$	Man \equiv Human \sqcap Male
subPropertyOf	$P_1 \sqsubseteq P_2$	hasDaughter \sqsubseteq hasChild
equivalentProperty	$P_1 \equiv P_2$	cost \equiv price
transitiveProperty	$P^+ \sqsubseteq P$	ancestor ⁺ \sqsubseteq ancestor

OWL Syntax	DL Syntax	Example
type	$a : C$	John : Happy-Father
property	$\langle a, b \rangle : R$	$\langle \text{John}, \text{Mary} \rangle : \text{has-child}$

```
<owl:Class rdf:ID="profesorAsociado">
  <owl:subClassOf rdf:resource="miembroStaffAcademico"/>
</owl:Class>
```

OWL. Sintaxis

- ▶ Clases enumeradas
 - Consiste exactamente en los individuos listados



Asserted Conditions

NECESSARY & SUFFICIENT

Accommodation

\exists hasRating {OneStarRating TwoStarRating}




NECESSARY

 BudgetAccommodation

Enumeración

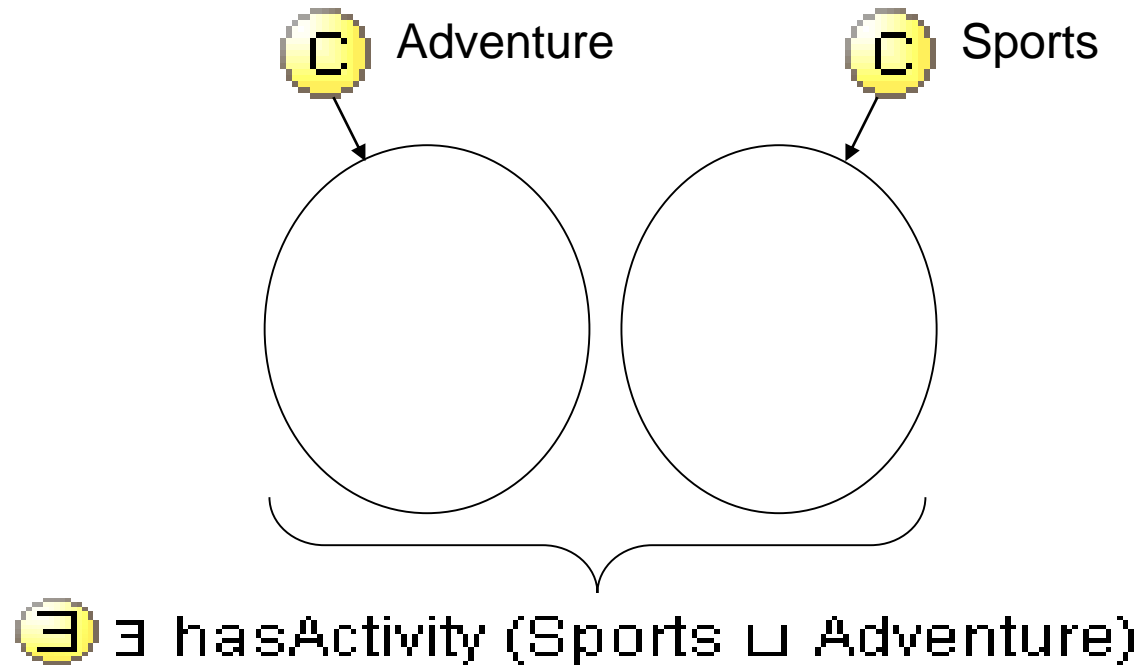
```
<owl:Class>  
  <owl:oneOf rdf:parseType="Collection">  
    <owl:Thing rdf:about="#Eurasia"/>  
    <owl:Thing rdf:about="#Africa"/>  
    <owl:Thing rdf:about="#NorthAmerica"/>  
    <owl:Thing rdf:about="#SouthAmerica"/>  
    <owl:Thing rdf:about="#Australia"/>  
    <owl:Thing rdf:about="#Antarctica"/>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

OWL. Sintaxis

- ▶ Definiciones lógicas de clases
- ▶ Define clases fuera de otras clases
 -  unionOf (or)
 -  intersectionOf (and)
 -  complementOf (not)
- ▶ Permite anidamientos arbitrarios de descripciones de clases
A and (B or C) and not D)

OWL. Sintaxis

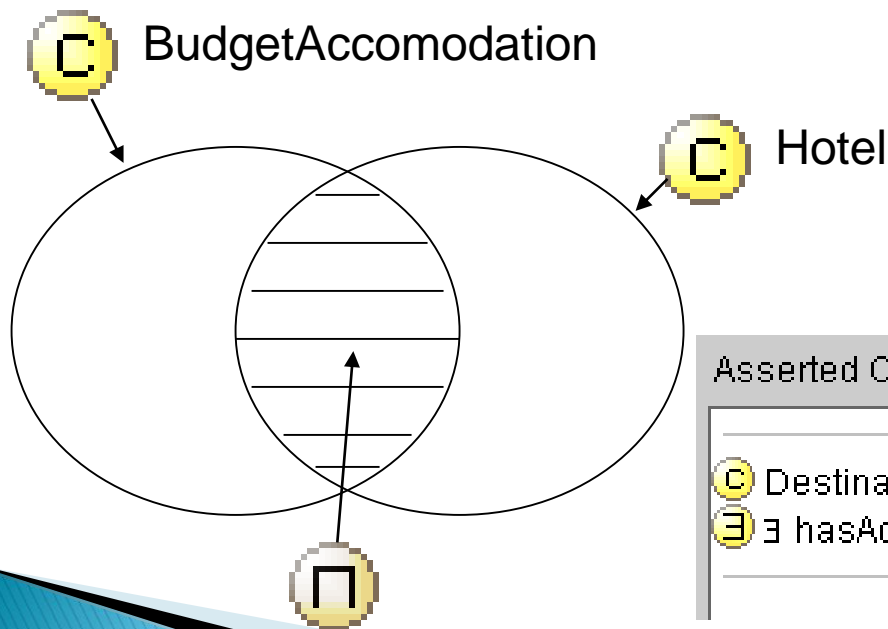
- ▶ **unionOf**
 - La clase de individuos que pertenecen a una clase A o clase B (o ambas)
 - Example: Adventure or Sports activities



OWL. Sintaxis

► intersectionOf

- La clase de individuos que pertenecen a ambas clases A y B
- Ejemplo: A BudgetHotelDestination is a destination with accomodation that is a budget accomodation **and** a hotel



Asserted Conditions

Destination

\exists hasAccommodation (BudgetAccommodation \sqcap Hotel)

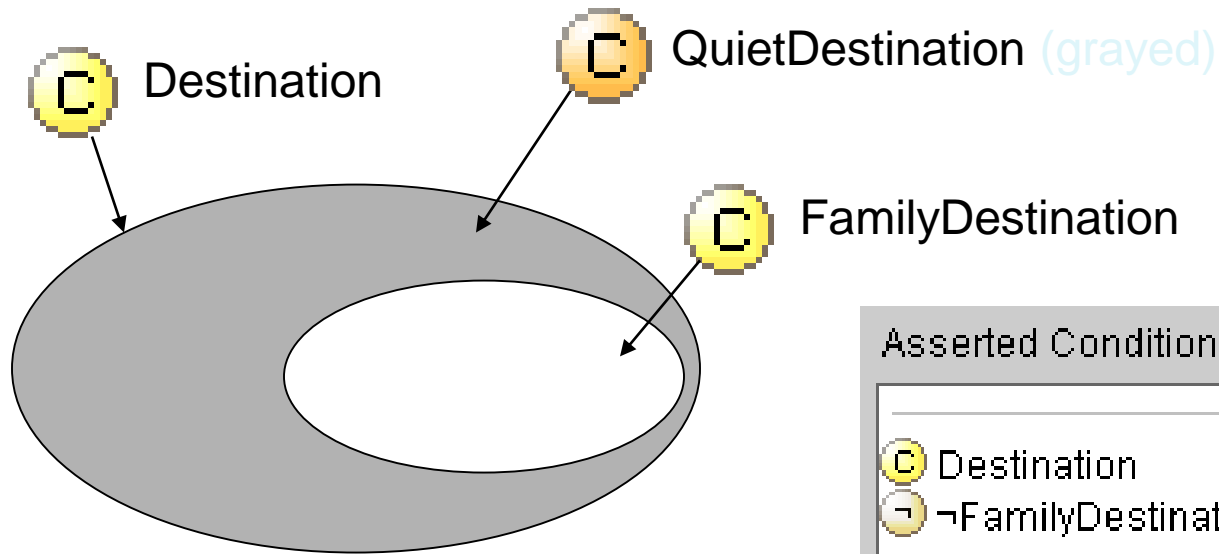
NECESSARY & SUFFICIENT

NECESSARY

OWL. Sintaxis

► complementOf

- La clase de todos los individuos que no pertenecen a una cierta clase
- Ejemplo: A quiet destination is a destination that is **not** a family destination



Asserted Conditions

NECESSARY & SUFFICIENT


Destination


\neg FamilyDestination

NECESSARY

OWL. Sintaxis

- ▶ **Condiciones de clase**

 **Condiciones necesarias:**
(Clases parciales/primitivas)
“Si sabemos que algo es una X, entonces debe cumplir las condiciones....”

 ▶ **Condiciones Necesarias y suficientes:**
(Clases completas/definidas)
“Si algo complete las condiciones....., entonces es una X.”

OWL. Sintaxis




 NationalPark



Asserted Conditions

NECESSARY & SUFFICIENT

NECESSARY

-  RuralArea
-  \exists hasAccommodation Campground
-  \exists hasActivity Hiking

(not everything that fulfills these conditions is a NationalPark)



 QuietDestination



Asserted Conditions

NECESSARY & SUFFICIENT

NECESSARY

-  Destination
-  \neg FamilyDestination

(everything that fulfills these conditions is a QuietDestination)




Owl. Sintaxis

NationalPark

Asserted Conditions

NECESSARY & SUFFICIENT

NECESSARY

-  RuralArea
-  \exists hasAccommodation Campground
-  \exists hasActivity Hiking

- ▶ A RuralArea is a Destination
- ▶ A Campground is BudgetAccommodation
- ▶ Hiking is a Sport
- ▶ Therefore:
Every NationalPark is a Backpackers-Destination

BackpackersDestination

Asserted Conditions

NECESSARY & SUFFICIENT

NECESSARY

-  Destination
-  \exists hasAccommodation BudgetAccommodation
-  \exists hasActivity (Sports \sqcup Adventure)

