



Diseño y Construcción de Ontologías para la Web Semántica

Ontology Learning



Miguel Angel Niño Zambrano

Msc. en Informática

PhD en Ingeniería Telemática

Contenido



LENGUAJES DE LA WEB SEMANTICA

ONTOLOGIAS PARA LA WEB SEMANTICA

METODOLOGIA DE CREACIÓN DE ONTOLOGIAS

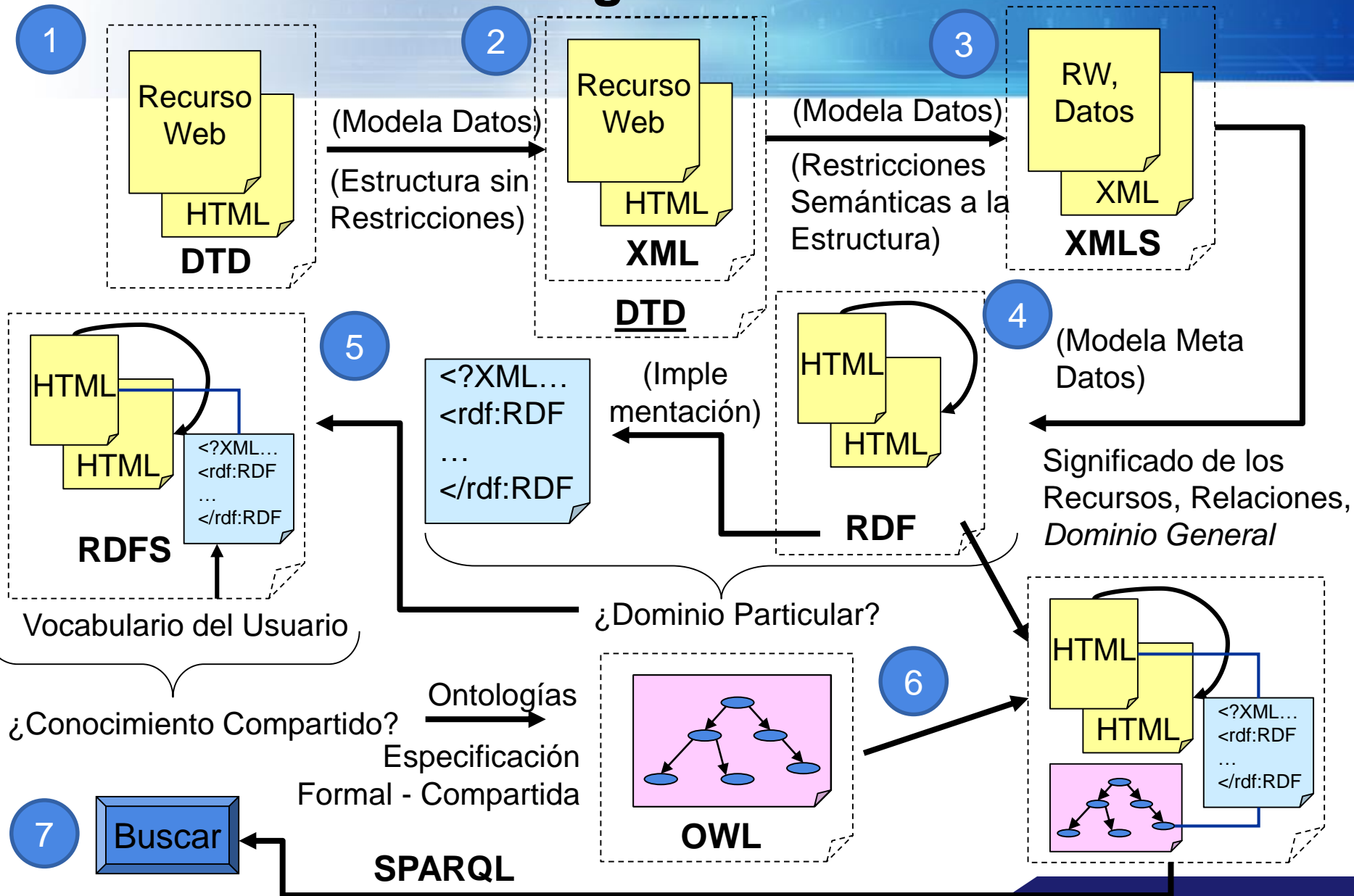
TALLER DE CONSTRUCCION DE ONTOLOGIAS

*Los procesos de estandarización desarrollados por la W3C ha permitido evolucionar la Web al punto que ya es posible tener **comportamientos semánticos**, como buscadores semánticos, agentes que consultan diferentes servicios web, interacción dinámica con los usuarios, entre otras funcionalidades que no se hubiesen podido desarrollar sin la interoperabilidad que generan los lenguajes de la Web semántica.*

Miguel Angel Niño

Lenguajes de la Web Semántica

Relación de Tecnologías en la Web Semántica

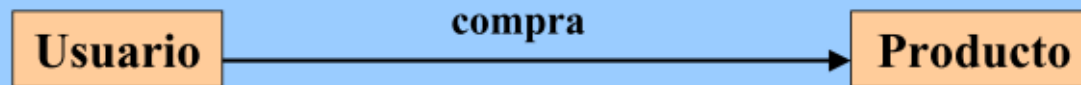


Lenguajes Web Esquema - “Schema”

- Los lenguajes web existentes extendieron para facilitar la descripción del contenido
 - XML → XML Schema (XMLS)
 - RDF → RDF Schema (RDFS)
- XMLS *no es un lenguaje ontológico*
 - Cambia el formato de los DTDs (esquemas de documento) en XML
 - Añade una **jerarquía de tipos extensible**.
 - Integers, Strings, etc.
 - Puede definir sub-types, e.j., positive integers.
- RDFS *se reconoce como un lenguaje ontológico*
 - Clases y propiedades
 - Sub/super-clases (y propiedades)
 - Rango y dominio (de propiedades)

¿Porque RDF y no XML para representar el conocimiento?

Ejemplo de la ambigüedad semántica de XML



Rubén compra la silla ref. 120 en la compra con código 112

```
<compra codigo="112">
  <usuario>Rubén </usuario>
  <producto>silla ref. 120</producto>
</compra>
```

```
<producto nombre="silla ref. 120">
  <comprador>Rubén
    <compra codigo="112" />
  </comprador>
</producto>
```

```
<usuario nombre="Rubén">
  <compra codigo="112">
    <producto>silla ref. 120</producto>
  </compra>
</usuario>
```

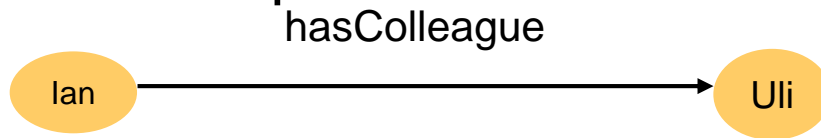
Miguel Ángel Abián, agosto de 2005

RDF

- (Resource Description Framework), proporciona información descriptiva sobre los recursos que se encuentran en la Web. es una base para *procesar metadatos*; proporciona *interoperabilidad* entre aplicaciones que intercambian información legible por máquinas en la Web. RDF se destaca por la facilidad para habilitar el *procesamiento automatizado* de los recursos Web.
- Existen otros lenguajes, aparte del XML, en los que se puede usar RDF, algunos son: N3, RxR, Turtle, N-Triplex, Trix.

Modelo de Datos de RDF

- Las sentencias son tripletas: <subject, predicate, object>:
 <Ian, hasColleague, Uli>
- Puede ser representado como una gráfica:



**Las sentencias describen propiedades de los Recursos.
Un recurso es cualquier objeto que puede ser apuntado
mediante una URI:**

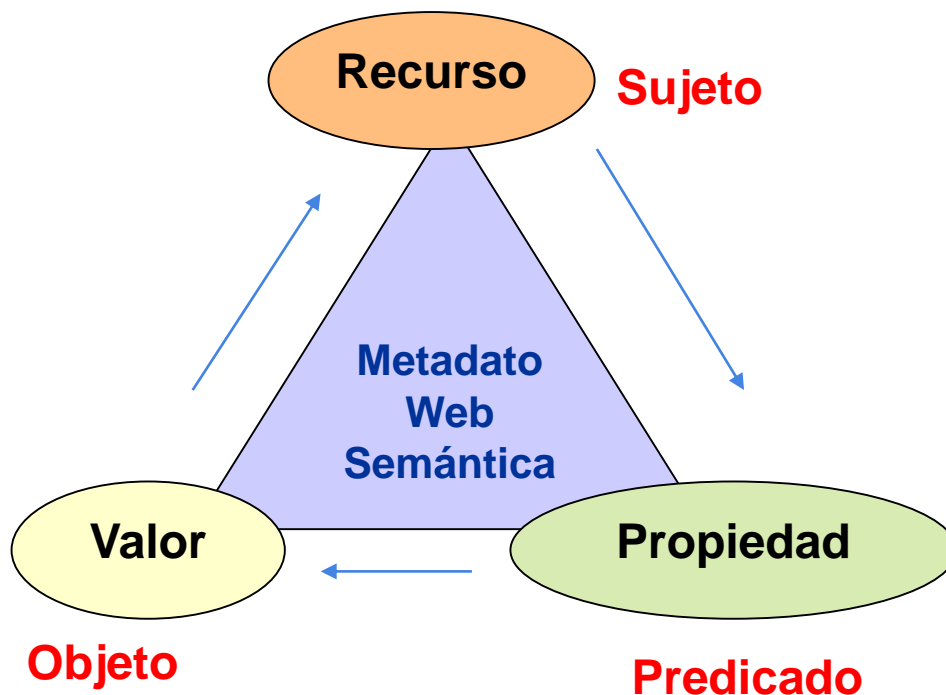
- a document, a picture, a paragraph on the Web;
- <http://www.cs.man.ac.uk/index.html>
- a book in the library, a real person (?)
- [isbn://5031-4444-3333](#)
- ...

Las propiedades en si mismas también son (URIs)

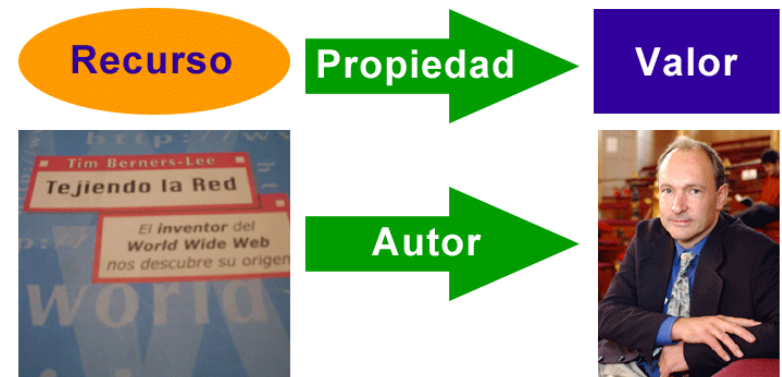
URIs

- URI = Uniform Resource Identifier
- “El conjunto genérico de todos los nombres / direcciones son cadenas cortas que hacen referencia a los recursos”
- URLs (Uniform Resource Locators) es un tipo particular de URI, usado para los recursos que pueden ser accedidos en la WWW (e.g., web pages)
- En RDF, URIs normalmente se ven como “normal” URLs, a menudo con identificadores de fragmentos para apuntar a partes específicas de un documento :
 - <http://www.somedomain.com/some/path/to/file#fragmentID>

Metadatos de la Web Semántica



Ejemplo



Ejemplo de Metadatos en RDF



TOMADO DE [4]

Ejemplo de archivo RDF - XML

- RDF tiene una sintaxis XML que tiene un significado específico:
 - Cada elemento “*Description*” describe un recurso.
 - Cada atributo o elemento anidados dentro de “*Description*” es una propiedad “*property*” de ese recurso.
 - Nosotros nos podemos referir a los Recursos mediante URIs
- Ejemplo:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<rdf:RDF xmlns:dc="http://purl.org/dc/elements/1.1/"
```

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

```
<rdf:Description rdf:about="http://www.example.org/">
```

```
<dc:title>Mi vida como escritora</dc:title>
```

```
</rdf:Description>
```

```
</rdf:RDF>
```

Sujeto

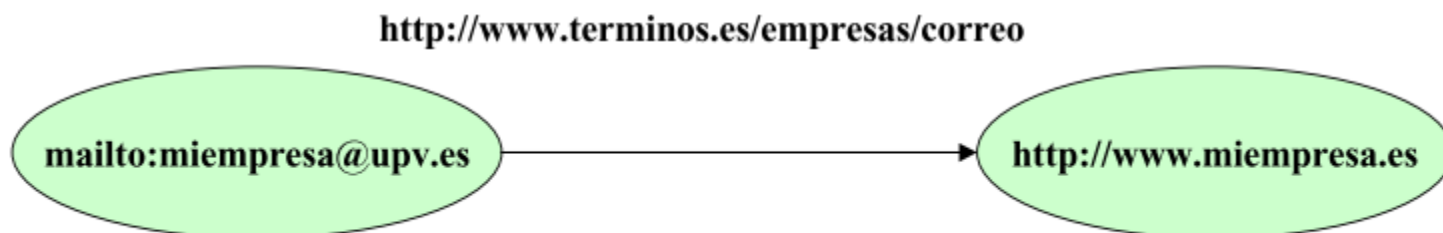
Predicado

Valor

Ejemplo Triplete RDF solo con URIs

El correo miempresa@upv.es pertenece a la empresa identificada por
<http://www.miempresa.es>

Sujeto Propiedad Objeto



Miguel Ángel Abián, julio de 2005

Tomado de: M. Á. Abián, "El futuro de la Web XML, RDF/RDFS, ontologías y la Web semántica," p. 103, 2005.

Ejemplo RDF en XML para representar relaciones de recursos

- Representar en RDF la siguiente sentencia: “*La página web <http://www.uv.es/documentacion/java> ha sido creada por Luisa Rodenas*”. En la forma de (sujeto, propiedad, objeto) quedaría así:

```
<?xml version="1.0", encoding="UTF-8"?>
```

```
<rdf:RDF
```

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

Encabezado

```
<rdf:Description rdf:about="http://www.uv.es/documentacion/java">
```

```
  <definiciones:creador>
```

```
    Luisa Rodenas
```

```
  </definiciones:creador>
```

```
</rdf:Description>
```

Cuerpo

```
</rdf:RDF>
```

Pie

```
<definiciones:creador rdf:resource="mailto:lrodenas@uv.es" />
```

¿Qué se puede hacer con RDF?

- **Recuperación de recursos** para proporcionar mejores prestaciones a los motores de búsqueda.
- **Catalogación** para describir el contenido y relaciones de contenido disponibles en un sitio Web, página o biblioteca digital.
- Se puede utilizar por los **agentes de software inteligentes** para facilitar el intercambio y compartir conocimiento.
- **Calificación** de contenido
- **Descripción de colecciones** de páginas que representan un “documento” lógico.
- Describir los derechos de **propiedad intelectual** de las páginas web.
- Expresar **preferencias de privacidad** de un usuario.
- Expresar **políticas de privacidad** de un sitio Web.

RDFS como modelado de datos para la Web

Recordemos que RDF es un lenguaje para representar información en la Web.

RDFS (RDF Schema), del W3C, es un **lenguaje para describir vocabularios en RDF**.

Mediante RDFS podemos **declarar propiedades, y establecer relaciones** entre propiedades y otros recursos. Por lo tanto lo hace el mejor candidato para representar el conocimiento.

¿Por qué se debe utilizar un RDFS?

- RDF no se asocia a ningún dominio en particular.
- Cada organización puede crear su **propia terminología y vocabulario** con RDFS (RDF Schema, esquema RDF). Además, RDFS permite especificar las entidades a las que pueden aplicarse los atributos del vocabulario.
- Es necesario definir un RDFS porque permite comprobar si un **conjunto de tripletas RDF** (un conjunto de metadatos, en definitiva) resulta **válido** o no para ese esquema. Al disponer de un esquema RDF, se puede comprobar si las **propiedades** aplicadas a los recursos son **correctas** (un reloj no puede tener una propiedad que sea NumeroSeguridadSocial, p. ej.) y si los valores vinculados a las propiedades tienen sentido (p. ej., carece de sentido que una propiedad vendeAccionDeBolsa tenga como valor a un simpático marsupial). (Abian)
- Son diferentes las funciones de XMLS vs. RDFS

Vocabulario RDF

VOCABULARIO DE RDFS (RDF SCHEMA)

Los usuarios de RDF pueden definir sus propias terminologías mediante el lenguaje de esquemas RDFS. Con RDFS se puede definir un vocabulario especializado, especificar las propiedades aplicables a las clases de objetos y describir las relaciones entre clases. El vocabulario de RDFS se define en <http://www.w3.org/2000/01/rdf-schema#>

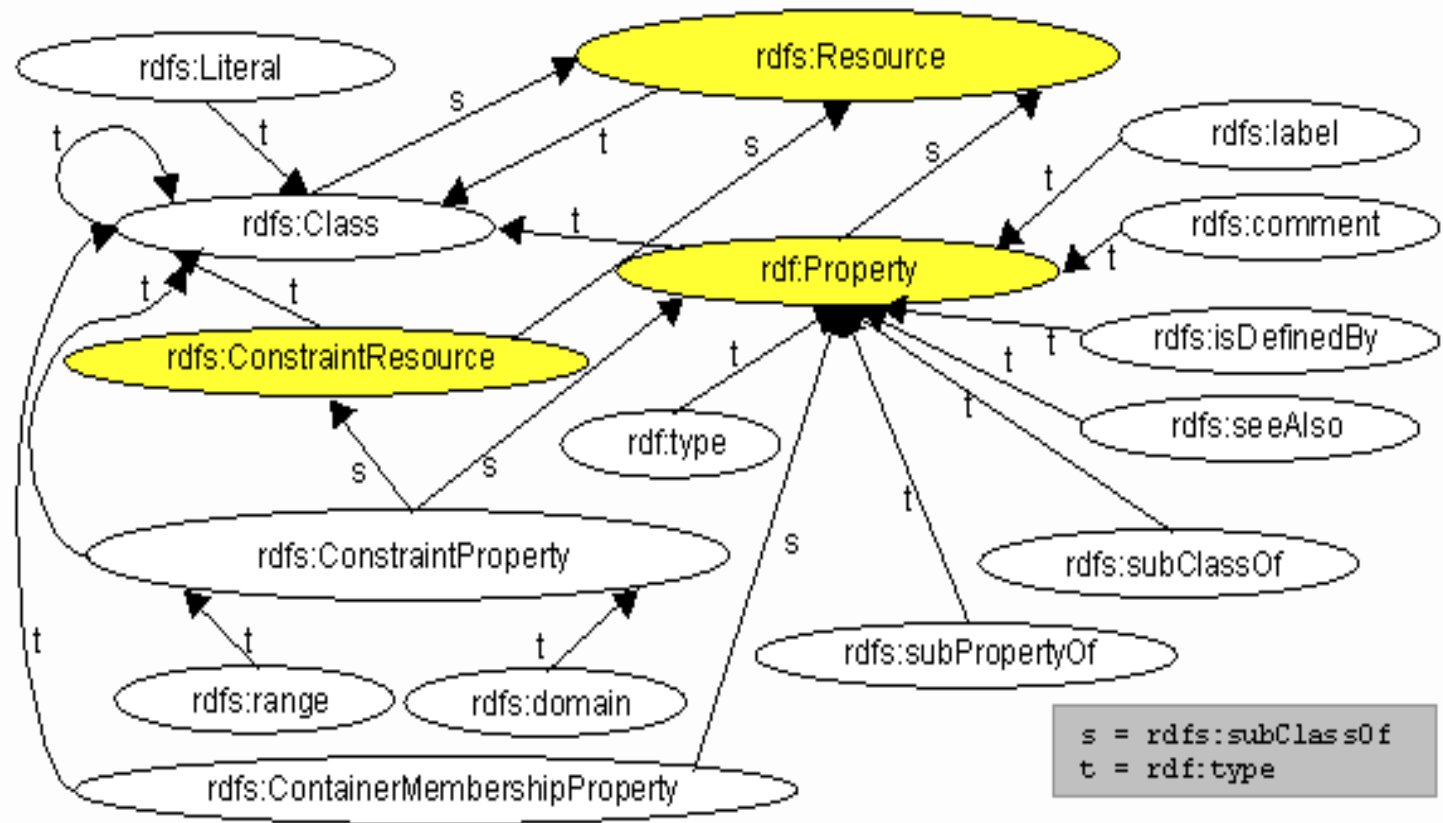
Clases de RDFS:

- a) `rdfs:Resource`
- b) `rdfs:Class`
- c) `rdfs:Literal`
- d) `rdfs:Datatype`
- e) `rdfs:Container`
- f) `rdfs:ContainerMembershipProperty`

Propiedades de RDFS:

- a) `rdfs:domain`
- b) `rdfs:range`
- c) `rdfs:subPropertyOf`
- d) `rdfs:subClassOf`
- e) `rdfs:member`
- f) `rdfs:seeAlso`
- g) `rdfs:isDefinedBy`
- h) `rdfs:comment`
- i) `rdfs:label`

Clases y Propiedades RDFS



Fuente: W3C

Lógica de primer orden de RDF/RDFS

- Por eficacia computacional, RDF y RDFS tienen también sus semánticas expresadas mediante tripletas (sujeto, propiedad, objeto). En las semánticas se incluye un sistema de inferencia robusto y completo. Las reglas del sistema de inferencia son de la forma:

C = Conjunto no vacío de tripletas

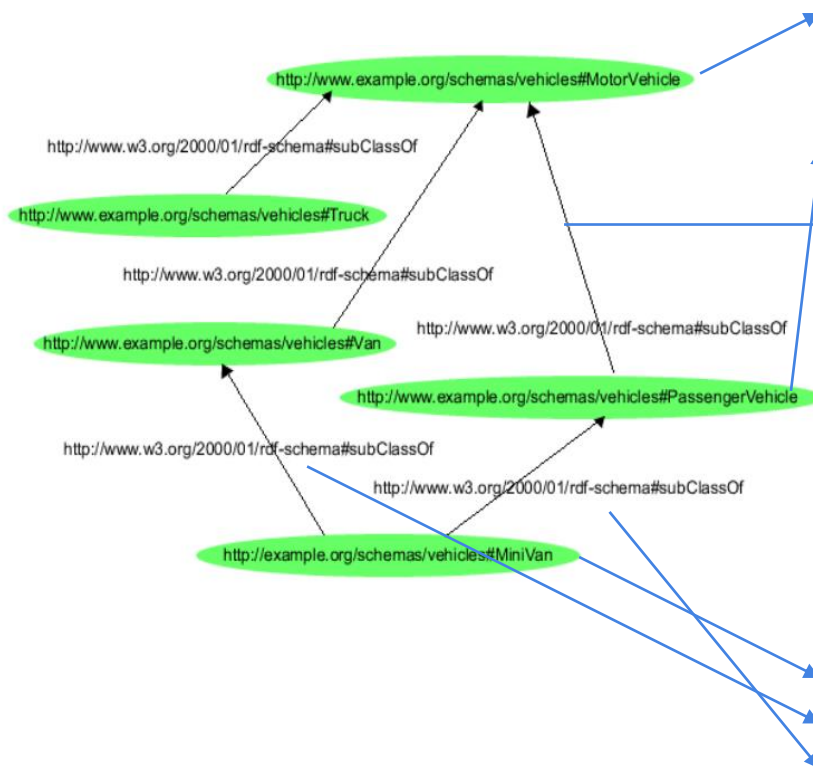
SI	C contiene determinadas tripletas
Entonces	se añaden a C ciertas tripletas

SI	C contiene la tripleta (?x, rdfs:subclassOf , ?y)
	y (?y, rdfs:subclassOf , ?z)
Entonces	C también contiene la tripleta (?x, rdfs:subClassOf , ?z)

Lógica de primer orden de RDF/RDFS

- Dadas las siguientes tripletas RDFS:
 - (InspectorSanitario, **rdf:type**, **rdfs:Class**)
 - (InspectorSanitario, **rdfs:SubclassOf**, Funcionario)
 - (Restaurante, **rdf:type**, **rdfs:Class**)
 - (inspecciona, **rdf:type**, **rdfs:property**)
 - (inspecciona, **rdf:domain**, InspectorSanitario)
 - (inspecciona, **rdf:range**, Restaurante)
- Aceptando lo anterior, se sigue que:
(JuanRodrigo, inspecciona, elBuenTenedor) → (JuanRodrigo, **rdf:type**, Funcionario)
- Es decir: si Juan Rodrigo inspecciona el restaurante “El buen tenedor”, entonces Juan Rodrigo es funcionario. Un agente inteligente podría usar esta deducción para buscar los datos sobre Juan Rodrigo en las bases de datos de empleados públicos o para pedir a la Administración información sobre él.

Ejemplo de Jerarquía de clases en RDFS



```

<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#"
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xml:base="http://example.org/schemas/vehicles">

  <rdf:Description rdf:ID="MotorVehicle">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
  </rdf:Description>

  <rdf:Description rdf:ID="PassengerVehicle">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

  <rdf:Description rdf:ID="Truck">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

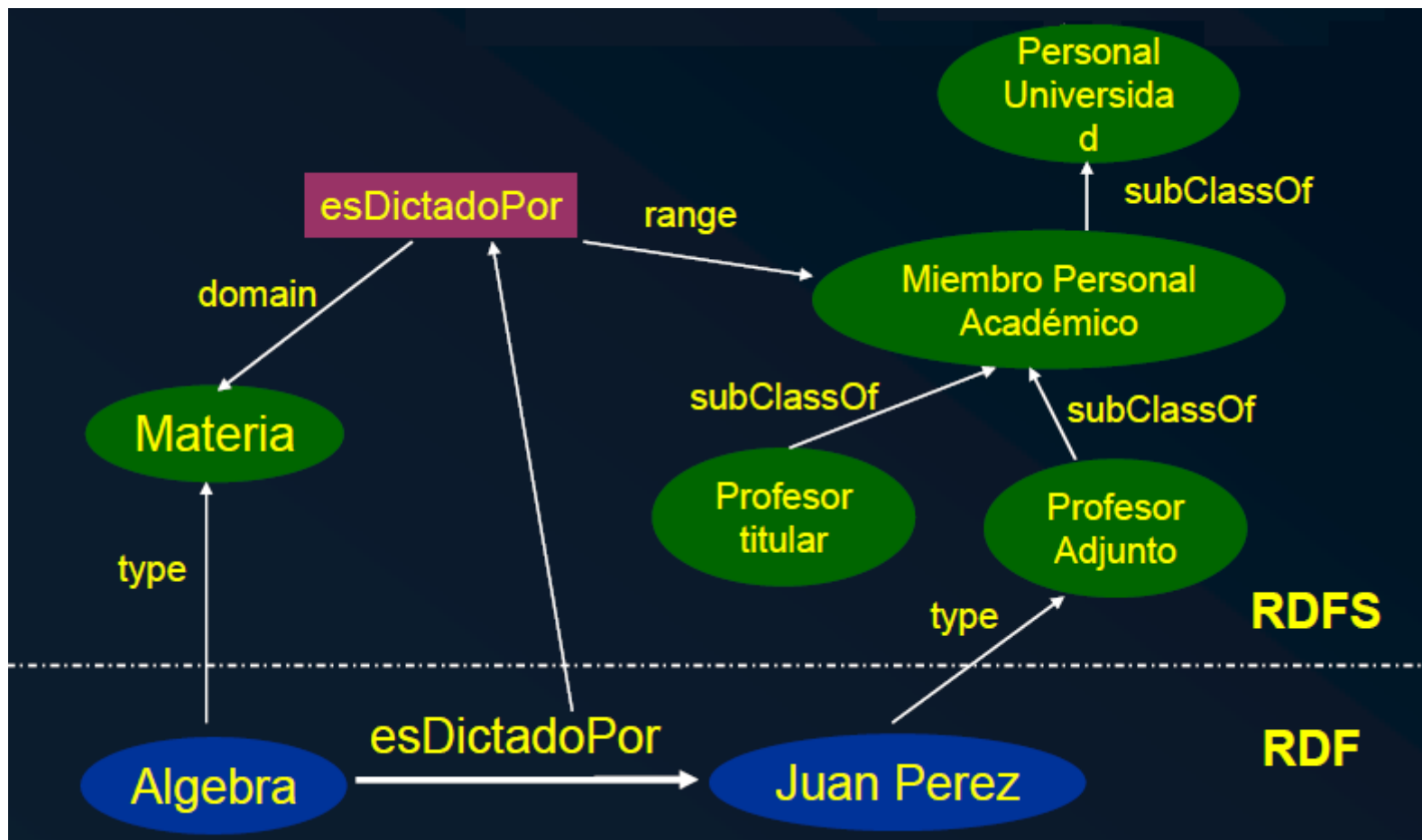
  <rdf:Description rdf:ID="Van">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#MotorVehicle"/>
  </rdf:Description>

  <rdf:Description rdf:ID="MiniVan">
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Class"/>
    <rdfs:subClassOf rdf:resource="#Van"/>
    <rdfs:subClassOf rdf:resource="#PassengerVehicle"/>
  </rdf:Description>

</rdf:RDF>

```

Ejemplo de interacción: RDFS - RDF



Problemas de RDFS

RDFS es muy potente, pero tiene algunas **carencias**:

- no permite **restricciones** de rango para sólo algunas clases
- no permite representar algunas características de propiedades (**transitiva, simétrica, inversa ó única**)
- No permite reflejar clases **disjuntas** (hombre - mujer).
- No permite expresar restricciones de **cardinalidad**.
- No permite algunas expresiones cuya semántica no se pueda expresar mediante la lógica de primer orden (la consecuencia es que al final no se puede afirmar o negar nada).

Para solucionar estas falencias se implemento el OWL (Ontology Web Lenguaje).

La Ontología es la **primera ciencia**. La Ontología implica **descubrir categorías e incluir objetos en ellas de maneras que tengan sentido**. Cuando Aristóteles miró alrededor del Mundo Antiguo, vio que era valioso empezar a categorizar sus partes. Dividió el mundo según sus elementos y procesos constituyentes, de los primeros intentos de clasificación de animales y plantas a la Física y la Metafísica. Desde Aristóteles, esta tareas se ha dividido entre varias ciencias discretas. **La mayoría de los científicos no piensan en lo que hacen como una ontología**; pero, en muchas maneras, es exactamente lo que comenzó Aristóteles hace dos mil años. Este trabajo continúa no sólo en las ciencias “duras”, sino también en las ciencias sociales. Es algo que hacemos todos nosotros, cada día. Cuando hacemos una **lista de cosas que hacer**, o de los discos y libros que más queremos comprar, o de vídeos que intentamos alquilar, estamos categorizando: estamos dedicándonos a una **rudimentaria ontología**. Concediendo prioridad a los elementos de una lista, asignamos relaciones entre varias cosas. La ontología puede ser relativamente simple o puede ser bastante compleja.

[Center for Commercial Ontolog. Prospectus, <http://www.acsu.buffalo.edu/~koepsell/center.htm>]

Ontologías para la Web Semántica

ONTOLOGIAS

- ¿Qué es una ontología?
 - “Una ontología es una **especificación formal** de una **conceptualización compartida**” (Borst (1997))
- ¿Dominio de Información?
 - un área de temática específica o un área de conocimiento, como la medicina, fabricación de herramientas, bienes inmuebles, reparación automovilística, gestión financiera,...



Son **vocabularios controlados** que las máquinas pueden entender y que son especificados con la suficiente precisión como para permitir diferenciar términos y **referenciarlos de manera precisa**

Información sobre Ontologías

- Una ontología define **términos** a utilizar para describir y representar un área de conocimiento.
- Las ontologías son **utilizadas por** las personas, las bases de datos, y las aplicaciones que necesitan compartir un dominio de información.
- Las ontologías incluyen **definiciones** de conceptos básicos del dominio, y las **relaciones** entre ellos, que son útiles para las máquinas.
- Codifican el **conocimiento de un dominio** y también el **conocimiento que extiende los dominios**. En este sentido, hacen que el conocimiento sea **reutilizable**.

¿Qué elementos se deben tener presentes de las ontologías?

- Las ontologías no son “sólo” contendores de información y relaciones.
- También debemos **confiar** en la en su información.
- Es necesario establecer criterios de **seguridad**, que garanticen en todo momento aquella información, que por su naturaleza, requiera **privacidad**.
- Cuatro aspectos importantes de la seguridad:
 - Autenticidad
 - Confidencialidad
 - Integridad de mensajes
 - Honorabilidad

Conceptos clave de las ontologías (1/6)

- **Clase (Concepto):** Descripción formal de una **entidad del universo o dominio** que se quiere representar. Constituye la pieza básica de estructuración del conocimiento. La decisión sobre qué considerar una clase del dominio no es fácil. Tal decisión debe ser tomada de acuerdo a los objetivos de la ontología (extraídos de las sesiones con los expertos, preguntas relevantes y de estándares existentes en el dominio). Una **clase puede tener subclases** que representan conceptos que son más específicos que dicha clase.
- **Clase abstracta:** Clase que **no permite** que existan **instancias** de ella. Se usa para agrupar conceptos, introducir cierto orden en la jerarquía, pero suelen ser demasiado generales para admitir instancias.
- **Instancia:** Representan **objetos concretos del dominio**, pertenecientes a una clase. La colección de instancias constituye la base de hechos (también denominada base de datos o base de conocimiento) del modelo.

Conceptos clave de las ontologías (2/6)

- **Instancia indirecta:** Cuando una clase es instancia indirecta de otra, quiere decir que es **instancia de alguna de sus clases derivadas**. En contraposición a instancia directa, donde no existen clases intermedias.
- **Propiedad (Atributo, Slot):** Característica que permite **describir más detalladamente la clase y sus instancias**. Establece que la clase o concepto posee una propiedad que se concretará mediante un valor. Los **valores** de las **propiedades** o atributos pueden ser **tipos básicos** como cadenas de caracteres o números, pero también pueden ser otras **clases** o **instancias** (vid. Relación).
- **Faceta (Restricción sobre las propiedades):** Es alguna **propiedad de la propiedad**. Por ejemplo, la cardinalidad, si la propiedad es obligatoria o no, etc.

Conceptos clave de las ontologías (3/6)

- **Relación:** Interacción o **enlace entre los conceptos** o clases del dominio que se modeliza. Algunas relaciones semánticas básicas son: **subclase de, parte de, parte exhaustiva de, conectado a, es un**, etc. Suelen configurar la taxonomía del dominio. Las relaciones más simples se modelizan mediante una propiedad de una clase cuyo valor es una instancia de otro concepto. Por ejemplo, dadas las clases AUTOR y OBRA, podemos definir la relación CREACIÓN como una propiedad (de cardinalidad múltiple) de la clase AUTOR cuyos valores sean instancias de la clase OBRA. Si las relaciones son más complejas (deben tener a su vez propiedades o deben organizarse en jerarquías, por ejemplo) se suelen definir mediante clases (conceptos) que las representen.

Conceptos clave de las ontologías (4/6)

- **Axioma:** Regla que se añade a la ontología y que permite describir el comportamiento de los conceptos o clases. Se establecen a partir de valores específicos de las propiedades. Por ejemplo: “Para todo A que cumpla la condición C, entonces A es B”. Permiten dejar constancia de que ciertos valores de propiedades introducidos son coherentes con las restricciones de la ontología, o bien inferir posteriormente valores de atributos que no se han introducido explícitamente. De esta forma, **a través de los axiomas es posible inferir conocimiento no codificado explícitamente en la ontología.**

Conceptos clave de las ontologías (5/6)

- **Herencia múltiple:** Se da cuando una clase dada hereda o cuenta con las propiedades de dos clases padre con las que establece dos relaciones del tipo 'es_un'.
- **Derivación:** Organización de las clases de la ontología en un árbol de jerarquía mediante sucesivas relaciones 'es_un' (también llamadas kind_of, is_a, o herencias) con la propiedad de herencia. Esta organización permite el encadenamiento sucesivo de herencias desde las clases de nivel superior a las clases situadas en niveles inferiores, llamadas clases derivadas.

Conceptos clave de las ontologías (6/6)

- **Anotación:** Es el proceso de relleno de instancias a partir de texto libre. Existen dos maneras de anotar texto: la más usual es la **inclusión de etiquetas semánticas** dentro del texto que se está procesando. Esto implica que el formato del texto debe ser editable y procesable. En el caso de no disponer de este formato, la segunda manera consiste en **rellenar las instancias directamente en el modelo**, dejando el texto original sin modificar.
- **Herencia:** Propiedad de la **relación 'es_un'** que permite que las clases relacionadas (heredadas) cuenten con los atributos de la clase con la cual se relacionan (clase padre).

Clasificación

Guarino (1997) clasifica las ontologías de acuerdo con su dependencia y relación con una tarea específica desde un punto de vista:

- Ontologías de Alto Nivel o Genéricas
- Ontologías de Dominio
- Ontologías de Tareas o de Técnicas básicas
- Ontologías de Aplicación

Tipos de ontologías

Según el ámbito del conocimiento al que se apliquen:

- **Ontologías generales**
- **Ontologías de dominio**
- **Ontologías específicas**

Según el tipo de agente al que vayan destinadas:

- **Ontologías lingüísticas**
- **Ontologías no lingüísticas**
- **Ontologías mixtas**

Según el grado o nivel de abstracción y razonamiento lógico que permitan:

- **Ontologías descriptivas**
- **Ontologías lógicas**

Beneficios de las ontologías

- Proporcionan una forma de representar y compartir el conocimiento utilizando un vocabulario común
- Permiten usar un formato de intercambio de conocimiento
- Proporcionan un protocolo específico de comunicación
- Permiten una reutilización del conocimiento

Limitaciones

- El modelo de datos toma un solo punto de vista del mundo. Describe los objetos o instancias de interés, pero bajo una sola posible interpretación.
- La reutilización de conocimiento complejo es imposible sin tomar en cuenta los diferentes puntos de vista.
- Algo de esto existe en bases de datos pero toda la información tiene que estar presente: los puntos de vista no añaden información por lo que la visión global es limitada.
- Por otro lado, existen desarrollos en modelo de datos orientados a objetos. Sin embargo, la representación de relaciones entre objetos sigue siendo pobre

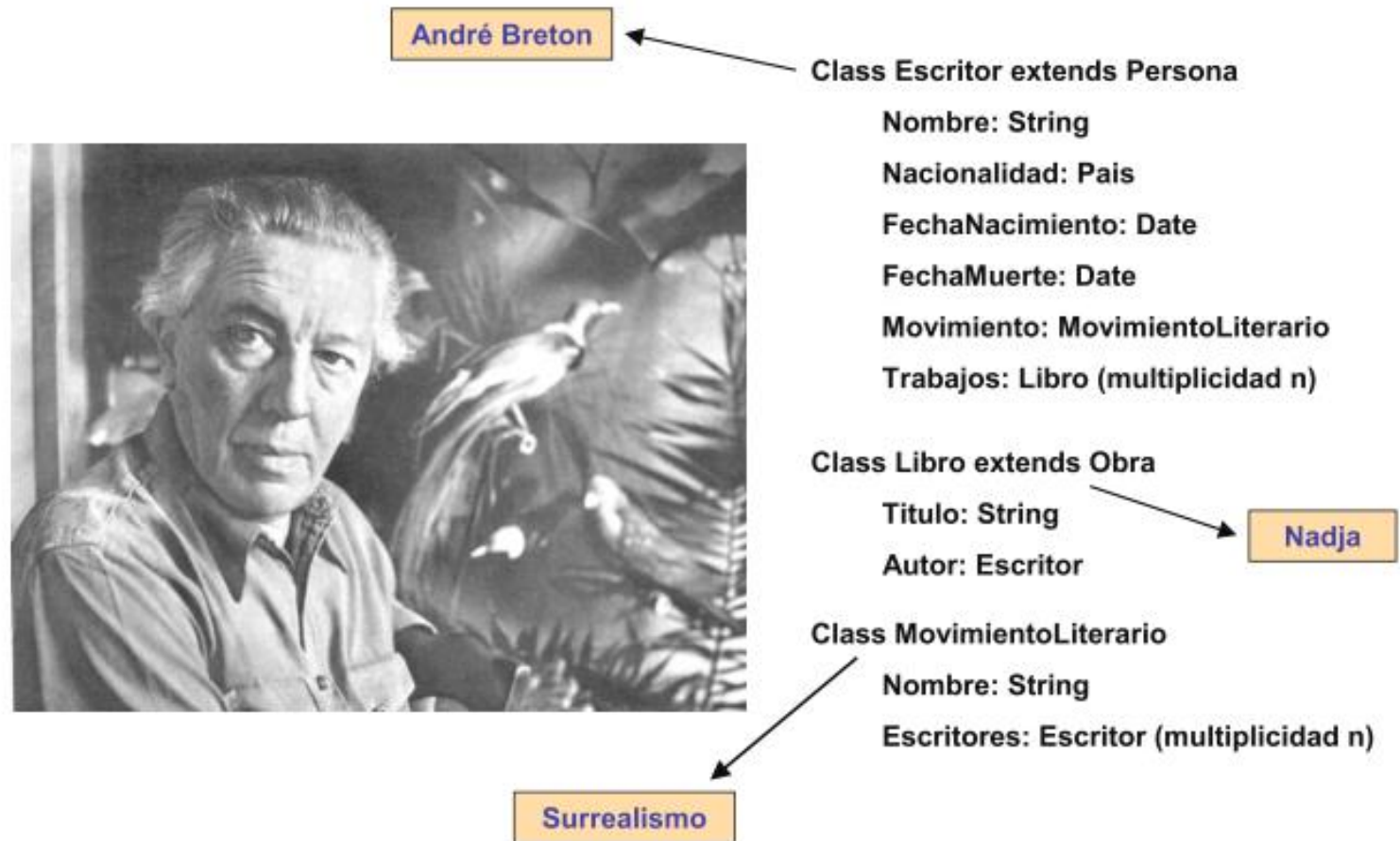
Usos de las Ontologías

- Repositorios para la organización del conocimiento
- Herramienta para la adquisición de información
- Herramientas de referencia en la construcción de sistemas de [bases de conocimiento](#) que aporten consistencia, fiabilidad y falta de ambigüedad a la hora de recuperar información
- Normalizar los atributos de los [metadatos](#) aplicables a los documentos
- Crear una [red](#) de relaciones que aporte especificación y fiabilidad
- Permitir compartir [conocimiento](#)
- Posibilitar el trabajo cooperativo al funcionar como soporte común de conocimiento entre organizaciones, comunidades científicas, etc.
- Permitir la integración de diferentes perspectivas de [usuarios](#)

Usos de las Ontologías

- Permitir el tratamiento ponderado del conocimiento para recuperar información de forma automatizada.
- Permitir la construcción automatizada de mapas conceptuales y mapas temáticos.
- Permitir la reutilización del conocimiento existente en nuevos sistemas.
- Permitir la interoperabilidad entre sistemas distintos.
- Establecer modelos normativos.
- Servir de base para la construcción de lenguajes de representación del conocimiento.

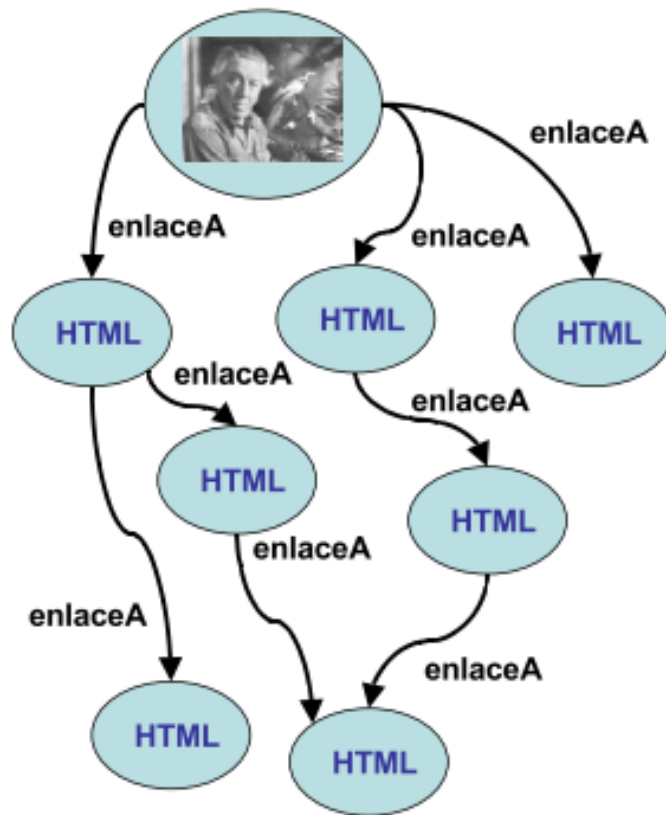
Ejemplo 1: Ontología de Escritores



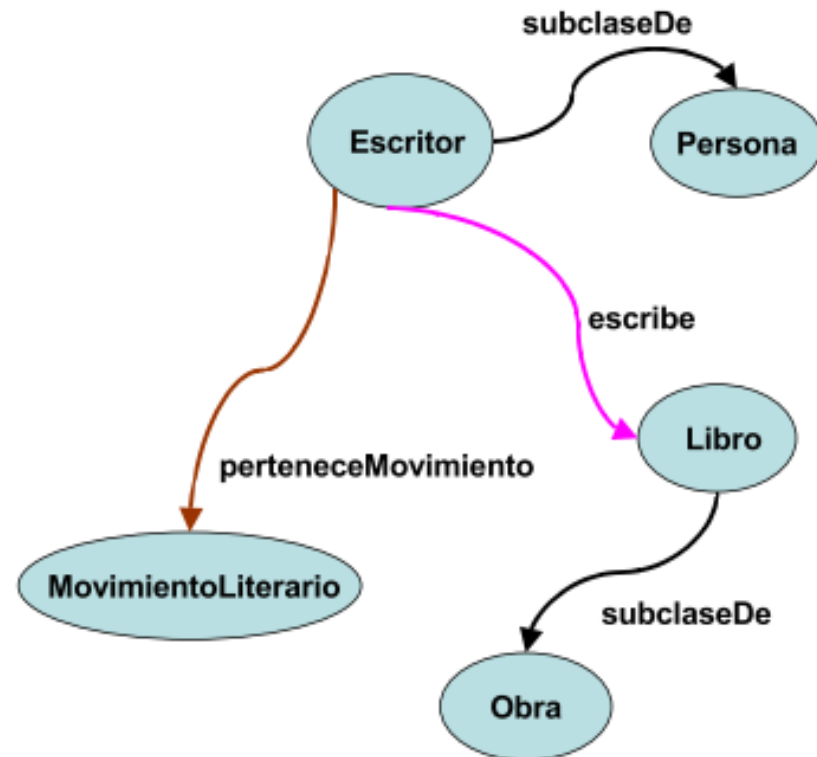
Miguel Ángel Abián, julio de 2005

Ejemplo 2: Ontología de redes de recursos

Red actual



Red semántica



Lenguajes de Representación de Ontologías

Hay muchos lenguajes que se usan para representar las ontologías, éstos son los más conocidos:

- **RDFS**
- **OWL**
- **DAML+OIL** (similar a OWL y superado por OWL)

OWL

- **OWL** (acrónimo de **Web Ontology Language**), es un lenguaje desarrollado por el **W3C**. Es una “**versión mejorada**” de **RDFS**.
- Entre las propiedades que incluye para restringir las instancias de una clase, se destacan **disjointWith** (expresa la exclusividad mutua de las clases), **unionOf** (expresa uniones de clases) y **oneOf** (enumera todas las instancias de una clase). Para restringir los valores de las propiedades, se puede usar **rdfs:domain** y **rdfs:range** (proviene de RDFS), **TransitiveProperty** (indica que una propiedad es transitiva), **inverseOf** (indica que una propiedad es inversa de otra), **minCardinality** (especifica el número mínimo de elementos que participan en una relación), **maxCardinality** (especifica el número máximo de elementos que participan en una relación).

Características del Lenguaje OWL

- Existen tres tipos de especificaciones OWL:
 - **OWL full** es la unión de la sintaxis OWL y RDF
 - **OWL DL** restringida a un fragmento (similar a DAML+OIL)
 - **OWL Lite** es “fácil de implementar” subconjunto de OWL DL
- Basado en lógica de primer orden o *Description Logic* (DL)
- OWL DL se beneficia de varios años en la investigación de descripciones lógicas
 - **Semánticas** bien definidas
 - **Propiedades formales** bien entendidas (complejidad, decidibilidad)
 - **Algoritmos de razonamiento** conocidos.
 - **Sistemas implementados** (altamente optimizados)

Constructores de Clase OWL

Constructor	DL Syntax	Example	Modal Syntax
intersectionOf	$C_1 \sqcap \dots \sqcap C_n$	Human \sqcap Male	$C_1 \wedge \dots \wedge C_n$
unionOf	$C_1 \sqcup \dots \sqcup C_n$	Doctor \sqcup Lawyer	$C_1 \vee \dots \vee C_n$
complementOf	$\neg C$	\neg Male	$\neg C$
oneOf	$\{x_1\} \sqcup \dots \sqcup \{x_n\}$	{john} \sqcup {mary}	$x_1 \vee \dots \vee x_n$
allValuesFrom	$\forall P.C$	\forall hasChild.Doctor	$[P]C$
someValuesFrom	$\exists P.C$	\exists hasChild.Lawyer	$\langle P \rangle C$
maxCardinality	$\leq_n P$	≤ 1 hasChild	$[P]_{n+1}$
minCardinality	$\geq_n P$	≥ 2 hasChild	$\langle P \rangle_n$

- XMLS [datatypes](#) as well as classes in 8P.C and 9P.C
 - E.g., 9hasAge.nonNegativeInteger
- Arbitrarily complex [nesting](#) of constructors
 - E.g., Person \sqcup 8hasChild.Doctor \sqcap 9hasChild.Doctor
- Para mayor entendimiento del DL [ver DL Wikipedia](#)

OWL: Ejemplo sintaxis abstracta

Ontology(
 Class(pp:planta partial)
 Class(pp:hierba partial pp:planta)
 Class(pp:arbol partial pp:planta)

 Class(pp:macho partial)
 Class(pp:hembra partial)
 Class(pp:joven partial)
 Class(pp:adulto partial)
 Class(pp:anciano partial pp:adulto)

 Class(pp:mascota complete restriction(pp:es_mascota_de someValuesFrom(owl:Thing)))
)

Definición OWL de la clase *planta*.

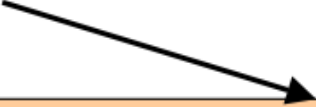
Definición OWL de que *hierba* es una subclase de *planta*. OWL permite expresar lo mismo con *SubClassOf*.

Definición OWL de que *anciano* es una subclase de *adulto*. OWL permite expresar lo mismo con *SubClassOf*.

La clase *mascota* sólo puede tomar valores de la clase OWL *Thing* cuando participa en la relación *es_mascota_de*.


OWL: Ejemplo sintaxis abstracta

```
Class(pp:animal partial restriction(pp:come someValuesFrom(owl:Thing)))  
Class(pp:gato partial pp:animal)  
Class(pp:perro partial pp:animal)  
Class(pp:oveja partial pp:animal restriction(pp:come allValuesFrom pp:hierba))  
Class(pp:perro partial pp:animal)
```




Expresión OWL de que una *oveja* sólo *come* (relación) *hierba*.

```
Class(pp:persona partial pp:animal)  
Class(pp:hombre complete intersectionOf(pp:persona pp:macho pp:adulto))
```



La clase *hombre* es la intersección de las clases *persona*, *macho* y *adulto*.

```
Class(pp:mujer complete intersectionOf(pp:hembra pp:persona pp:adulto))  
Class(pp:senyora+anciana complete intersectionOf(pp:anciano pp:hembra pp:persona))  
Class(pp:senyora+anciana partial  
  intersectionOf(restriction(pp:tiene_mascota allValuesFrom(pp:gato))  
    restriction(pp:tiene_mascota someValuesFrom(pp:animal))))
```



La clase *senyoraanciana* es la intersección de las clases *persona*, *hembra* y *anciano*. Además, para la clase *senyoraanciana*, la relación *tiene_mascota* sólo puede tomar valores de la clase *gato*. En una palabra: una señora anciana sólo puede tener gatos como mascotas.

OWL: Ejemplo sintaxis abstracta

Un amante de los animales tiene como mínimo tres mascotas.

```
Class(pp:amante+animales complete
  intersectionOf(pp:persona restriction(pp:tiene_mascota minCardinality(3))))
Class(pp:propietario+mascota complete
  intersectionOf(restriction(pp:tiene_mascota someValuesFrom(pp:animal)) pp:persona))
Class(pp:propietario+gato complete
  intersectionOf(pp:persona restriction(pp:tiene_mascota someValuesFrom(pp:gato))))
```

```
DisjointClasses(pp:perro pp:gato)
DisjointClasses(pp:joven pp:adulto)
```

La propiedad *come* es la inversa de *comido_por*.

```
ObjectProperty(pp:comido_por)
ObjectProperty(pp:come inverseOf(pp:comido_por) domain(pp:animal))
ObjectProperty(pp:tiene_mascota domain(pp:persona) range(pp:animal))
ObjectProperty(pp:es_mascota_de inverseOf(pp:tiene_mascota))
```

Una instancia no puede pertenecer a la clase *gato* y a la clase *perro*.

```
SubPropertyOf(pp:tiene_mascota pp:gusta_a)
```

Las personas tienen animales como mascotas.

```
Individual(pp:Miguel type(owl:persona))
Individual(pp:Dolly type(pp:oveja))
Individual(pp:Fido type(pp:dog) value(pp:es_mascota_de pp:Luis))
```

Sentencias sobre individuos: Miguel es una persona, Dolly es una oveja, Fido es un perro y es mascota de Luis.

```
Individual(pp:Consolacion type(pp:anciano) type(pp:hembra)
  value(pp:tiene_mascota pp:Sonrisas))
```

La señora Consolación es una anciana que tiene una mascota llamada Sonrisas.

La ontología anterior, pese a su simplicidad, ya permite realizar algunos razonamientos automáticos. De la última sentencia, p. ej., un programa deduciría que **Sonrisas es un gato**, pues Consolación es una persona (los propietarios de mascotas son personas) y, por tanto, es una señora anciana (pues es persona, mujer y anciana). Como todas las mascotas de las señoras ancianas son gatos, Sonrisas debe ser un minino.

OWL: Ejemplo sintaxis RDF-XML (I)

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<rdf:RDF xmlns:ns0="http://cohse.semanticweb.org/ontologies/people#"
  xmlns:owl="http://www.w3.org/2002/07/owl#"
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
  xml:base="http://cohse.semanticweb.org/ontologies/people"
  xmlns="http://cohse.semanticweb.org/ontologies/people#">
  <owl:Ontology rdf:about=""/>
  <owl:Class rdf:about="#white+van+man">
    <rdfs:label>white van man</rdfs:label>
    <rdfs:comment><![CDATA[A white van man is a man who drives a white
van.]]></rdfs:comment>
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
```

[...]

OWL: ejemplo sintáxis RDF-XML (II)

[...]

```
<owl:Class rdf:about="#man"/>
```

```
<owl:Restriction>
```

```
  <owl:onProperty rdf:resource="#drives"/>
```

```
  <owl:someValuesFrom>
```

```
    <owl:Class>
```

```
      <owl:intersectionOf rdf:parseType="Collection">
```

```
        <owl:Class rdf:about="#white+thing"/>
```

```
        <owl:Class rdf:about="#van"/>
```

```
      </owl:intersectionOf>
```

```
    </owl:Class>
```

```
  </owl:someValuesFrom>
```

```
</owl:Restriction>
```

```
</owl:intersectionOf>
```

```
</owl:Class>
```

```
</owl:equivalentClass>
```

```
</owl:Class>
```

[...]

En el novel campo de la Ingeniería del Conocimiento diversos grupos de investigación buscan un método de desarrollo de ontologías adecuado pero, las variables son tantas y tan diversas que es prácticamente imposible obtener un sólo método adecuado para todos los casos.

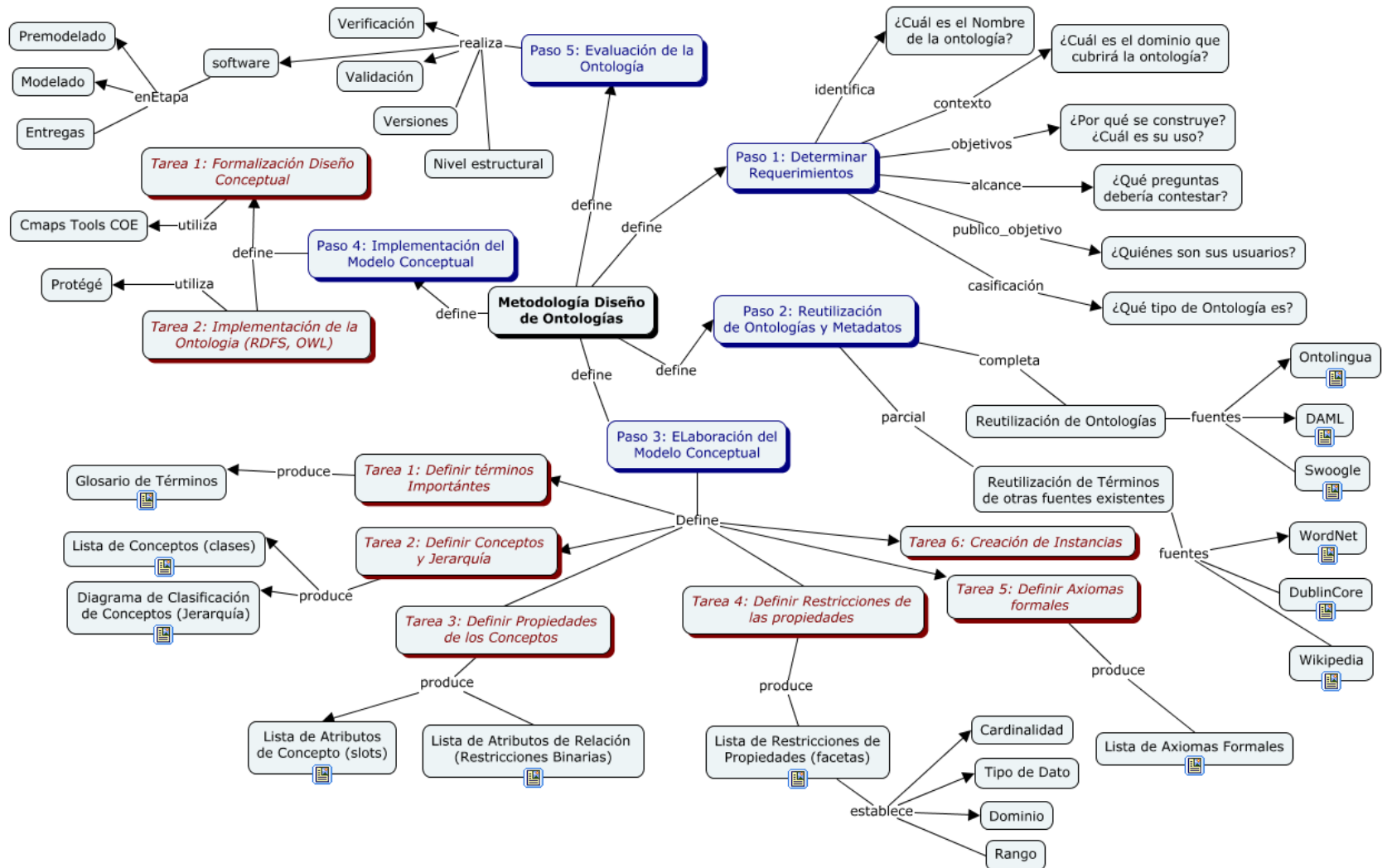
Ruben Dario Alvarado, 2010

Metodología de Creación de Ontologías

Metodología propuesta

- Existen varias metodologías para el desarrollo de ontologías. Para este taller se hace una adaptación resumida de las propuestas de:
 - Noy and D. McGuinness, *Ontology development 101: A guide to creating your first ontology*. 2001
 - O. Corcho, M. Fernández-López, A. Gómez-Pérez, and A. López-Cima, "Building legal ontologies with METHONTOLOGY and WebODE", 2005
 - "Metodología para el Desarrollo de Ontologías" de Rubén Dario Alvarado, 2010
 - "Tutorial de Ontologías", Jesus Contreras, Juan Antonio Martínez.

Pasos del Método



Reglas Fundamentales a tener en cuenta

1. No existe una única **forma correcta de modelar un dominio**; al contrario, siempre hay alternativas posibles. La mejor solución casi siempre depende del **propósito** u objetivo final de la ontología o de sus aplicaciones.
2. El desarrollo de una ontología es necesariamente un **proceso iterativo**.
3. Los **conceptos** o clases de una ontología deberían ser muy **cercanos a objetos** o entes (tanto físicos como lógicos) y a las relaciones existentes en nuestro dominio de interés. Estos conceptos o clases serán muy probablemente **nombres (objetos)** o **verbos (relaciones)** que encontraremos en las frases que describen nuestro dominio.

Paso 1: Determinar Requerimientos

Especificación	Descripción
¿Cuál es su Nombre? (Identificación)	Es bueno buscar un nombre adecuado que englobe la temática principal de la ontología. Es conveniente asignar un acrónimo el cual será utilizado como nombre del archivo de la ontología.
¿Qué dominio cubrirá la ontología? (Contexto)	Establece el contexto y temática principal que modela ontología. En lo posible establecer predicados que permitan determinar que objetos físicos o abstractos hacen parte de la temática.
¿Por qué se construye? ¿Cuál es su uso? (objetivos)	Se debe pensar en la funcionalidad de la aplicación final, la cual está alineada a los objetivos . Entre mejor detallado se identifique su uso mejor será la identificación de los conceptos a modelar.
¿Qué preguntas debería contestar? (Alcance)	Las preguntas establecidas deben poder responderse recorriendo los conceptos de la ontología. Las preguntas deben referirse a conceptos específicos del contexto modelado y su correcta identificación permitirá definir los límites del dominio modelado y el punto de vista desde el cual se modela.
¿Quiénes son sus usuarios? (Público Objetivo)	Se establecen los usuarios con respecto a las funciones que tendrán al interactuar con la ontología. Ej. si el usuario puede editar y cambiar clases y relaciones debe tener conocimiento del tema. Tener en cuenta la jerga de los usuarios con respecto a la formal.
¿Qué tipo de Ontología es? (Clasificación)	Conviene clasificarla en: Ontología de Nivel Superior, General, de Dominio, tarea o aplicación.

Paso 2: Reutilización de Ontologías

- En este punto se debe identificar si se utilizarán ontologías existentes que pueden ser generales o del mismo dominio, pero que complementan un aspecto que se está modelando. Una forma es alinear ontologías para reutilizar conceptos y jerarquías de interés. Un buen referente es el trabajo de:
 - M. Kavouras, *A unified ontological framework for semantic integration*, 2005.
 - M. Kavouras and M. Kokla, *Ontology-based fusion of geographic databases*: ntua.gr, 2000.
- Repositorios de Ontologías:
 - Ontolingua (<http://www.ksl.stanford.edu/software/ontolingua/>).
 - DAML ontology library (<http://www.daml.org/ontologies/>).

Paso 3: Elaboración del Modelo Conceptual

3.1 Definir los términos importantes de la ontología

- Realizar una tabulación de los conceptos, propiedades, relaciones, instancias y axiomas con el fin de identificar claramente los elementos que deben ser incluidos en la ontología.
- Se puede partir del cuerpo teórico construido, realizando una categorización sintáctica, en la cual los sustantivos son posibles clases o conceptos, los adjetivos posibles atributos o propiedades y los verbos posibles relaciones entre conceptos.
- En la tabla de términos presentada el tipo puede ser: conceptos, instancias, atributos, relaciones entre conceptos.

Nombre	Sinónimos	Acrónimos	Descripción	Tipo

Propuesta por: O. Corcho, M. Fernández-López, A. Gómez-Pérez, and A. López-Cima, Methontology

Paso 3: Elaboración del Modelo Conceptual

3.2 Definir las clases y su jerarquía

- Se seleccionan del glosario de términos los conceptos. METHONTOLOGY propone utilizar las cuatro relaciones taxonómicas definidas en la Frame Ontology:
 - **Subclase-de:** Un concepto C1 es Subclase-de otro concepto C2 si y sólo si todas las instancias de C1 son también instancias de C2. Por ejemplo: persona física es subclase de persona, dado que todas las personas físicas son personas. Un concepto puede ser subclase de más de un concepto en la taxonomía. Por ejemplo: el concepto compañía de control público y privado es subclase de los conceptos compañía privada y compañía pública.
 - **Descomposición-Disjunta:** Una Descomposición-Disjunta de un concepto C es un conjunto de subconceptos de C que no tienen instancias comunes y que no cubren C, es decir, puede haber instancias del concepto C que no son instancias de ninguno de los conceptos que forman la descomposición. Por ejemplo: los conceptos ministerio y juzgado forman una descomposición disjunta del concepto organización porque una organización no puede ser simultáneamente un ministerio y un juzgado. Además, pueden existir instancias o subclases del concepto organización que no son instancias o subclases de ninguno de esos dos conceptos.

Paso 3: Elaboración del Modelo Conceptual

3.2 Definir las clases y su jerarquía

- **Descomposición-Exhaustiva:** Una Descomposición-Exhaustiva de un concepto C es un conjunto de subconceptos de C que lo cubren, es decir, tal que no existe ninguna instancia de C que no sea instancia de al menos uno de los conceptos de la descomposición. Los conceptos que pertenecen a este conjunto pueden tener instancias y subconceptos comunes. Por ejemplo: los conceptos compañía privada y compañía pública forman una descomposición exhaustiva del concepto compañía porque no hay compañías que no sean instancia de al menos uno de esos conceptos, y pueden existir instancias y subclases comunes.
- **Partición:** Una Partición de un concepto C es un conjunto de subconceptos de C que no tienen instancias ni subconceptos comunes y que cubren C. Por ejemplo: los conceptos menor de edad y mayor de edad forman una partición del concepto persona física, dado que una persona física debe ser menor o mayor de edad, y nunca ambos.

Paso 3: Elaboración del Modelo Conceptual

- Algunas reglas generales que nos ayudan a decidir cuándo introducir una clase nueva, según Noy & McGuinness (1995), se detallan a continuación:
 - Una nueva subclase de una clase generalmente:
 1. Debería tener nuevas propiedades que no posee la clase
 2. Debería tener diferentes valores para las propiedades que los de la clase
 3. Debería participar en diferentes relaciones que la clase.
 - En jerarquías terminológicas, las nuevas clases no tienen por qué introducir nuevas propiedades.
 - Si un factor es importante en el dominio y pensamos en los objetos con diferentes valores para ese factor como diferentes clases de objetos, entonces deberíamos crear una nueva clase o clases considerando dicho factor.
 - Las instancias son los conceptos más específicos representados en una ontología.
 - Si los conceptos de un dominio forman una jerarquía natural, debemos representarlos como clases, aunque sean clases abstractas.

Paso 3: Elaboración del Modelo Conceptual

- Cuando desarrollamos la taxonomía de clases se deben considerar ciertas reglas generales (Noy & McGuinness, 1995). Así:
 - Si una clase A es una superclase de la clase B, entonces toda instancia de B es también una instancia de A.
 - Una subclase de una clase representa un concepto que es un tipo especial o una subespecie dentro del concepto representado por la clase.
 - Si B es una subclase de A y C es una subclase de B, entonces C es una subclase de A.
 - No debemos emplear sinónimos de un mismo concepto para representar clases diferentes, sino que los sinónimos deben considerarse denominaciones diferentes para un mismo y único concepto.
 - No deben aparecer ciclos o bucles en la jerarquía de clases.
 - Los conceptos de un mismo nivel de la jerarquía o clases hermanas (excepto los que derivan directamente de la raíz) deben presentar el mismo nivel de generalidad.
 - Cada clase debería tener entre 2 y 12 subclases directas.

Paso 3: Elaboración del Modelo Conceptual

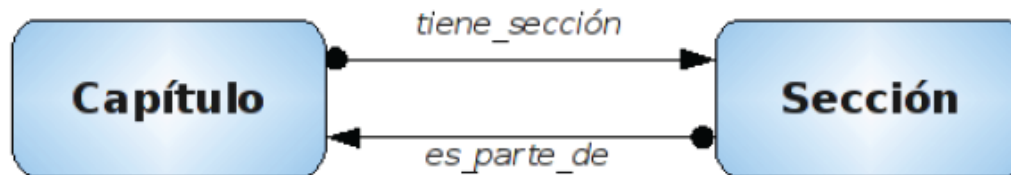
3.3 Definir las propiedades de las clases

1. **Definición de atributos de clase:** En general, las características de los objetos que pueden convertirse en propiedades o slots en una ontología son:
 - **Características intrínsecas**, como el sabor de un vino.
 - **Características extrínsecas**, como el nombre de un vino.
 - **Partes del objeto**, si el concepto o la clase está estructurado, tanto físicamente como de manera abstracta o convencional.

Clase	Instancia	Atributos de Clase	Atributos de Instancia	Relaciones
Asignatura	Matemáticas	Universidad	Unicauca	Tiene guía didáctica
		Área	Profesional / Técnica	
Guía Didáctica	Guía didáctica de matemáticas	Periodo	Abril - agosto, octubre - febrero	alcanza objetivo; tiene unidad de información; tiene índice;

Paso 3: Elaboración del Modelo Conceptual

1. **Relaciones con otros conceptos o clases:** Para esto se definen relaciones binarias entre conceptos. El objetivo de estos diagramas es establecer las relaciones ad hoc existentes entre conceptos de la misma o de distintas taxonomías de conceptos.



- Se construye la tabla de restricciones binarias:

Nombre de la Relación	Concepto origen	Cardinalidad Máxima	Concepto Destino	Relación Inversa
tiene_sección	Capítulo	N	Sección	es_parte_de
alcanza	Guía didáctica	N	Objetivo	es_meta_de

Paso 3: Elaboración del Modelo Conceptual

3.3 Definir las restricciones de las propiedades (Atributos de Instancia)

- En una ontología, las **propiedades** pueden tener diferentes **facetas**, éstas describen o caracterizan el tipo de valor que posee una propiedad.
- Las **restricciones** más comunes aplicadas a las propiedades son: los **valores permitidos y el número de valores posibles** (cardinalidad). A continuación una breve descripción de cada una de ellas:
 - **Cardinalidad:** Establece cuántos valores puede tener una propiedad o slot. Algunos sistemas distinguen únicamente entre cardinalidad simple (como máximo un valor) y cardinalidad múltiple (se permiten cualquier número de valores).
 - **Tipo de valor:** Describe qué tipo de valores puede poseer una propiedad. Los más frecuentes son: string (cadena), number, boolean, symbol (enumerado), e instance.
 - **Dominio y rango de una propiedad o slot:** Se suele denominar rango de una propiedad a las clases permitidas para una propiedad de tipo instancia. El dominio de una propiedad es el conjunto de clases que describe o caracteriza dicha propiedad.

Paso 3: Elaboración del Modelo Conceptual

- Se crea la tabla de restricciones de propiedades (restricciones de atributos de instancia)

Nombre del concepto	Nombre Atributo de Instancia	Tipo de Valor	Cardinalidad	Otras restricciones
Asignatura	Nombre	string	Simple o (1;1)	
	Área	string	Múltiple o (1:N)	
Objetivo	Descripción	string	Simple	
	Tipo	Symbol	Simple	Valores permitidos (General. Específico)
Guía Didáctica	Ciclo	Numérico	Múltiple o (2:10)	Valores permitidos (1..9)

- El **dominio** y **rango** de propiedades de relación (propiedades tipo Instancia) entre conceptos ya se ha definido en la tabla de restricciones binarias

Paso 3: Elaboración del Modelo Conceptual

3.4 Definición de los axiomas formales

- La tabla de axiomas lógicos define las expresiones lógicas que, en la ontología, son siempre verdaderas. La definición de cada axioma incluye el nombre, la descripción de la regla en lenguaje natural, el concepto al que se refiere el axioma, la expresión lógica que describe formalmente el axioma utilizando FOPC (cálculo de predicados de primer orden) y la relación

Axioma 1	
Nombre	Guía didáctica de una asignatura.
Descripción	Algunas asignaturas tienen una guía didáctica.
Conceptos	Asignatura, guía didáctica.
Expresión Matemática	$\exists (y) (Asignatura(x) \wedge Guía\ didáctica(y) \rightarrow Tiene(x, y))$
Relaciones	tieneGuiaDidactica

Axioma 2	
Nombre	Objetivos de cada asignatura.
Descripción	Toda asignatura tiene objetivos.
Conceptos	Asignatura, objetivo.
Expresión Matemática	$\forall (x, y) (Asignatura(x) \wedge Objetivo(y) \rightarrow Tiene(x, y))$
Relaciones	tieneObjetivo

Nombre del axioma	Descripción	Expresión	Conceptos	Relaciones	Variables
incompatibilidad acusado-demandante	una misma persona no puede ser el acusado y el demandante en el mismo juicio	no (existe(?X,?Y) (persona(?X) y juicio(?Y) y demandado(?Y,?X) y demandante(?Y,?X)))	persona juicio	demandado demandante	?X ?Y

Paso 4: Creación de Instancias

- El último paso consiste en crear las instancias individuales de las clases en la jerarquía. La definición de una instancia individual de una clase exige:
 1. Elegir una clase
 2. Crear una instancia individual para esa clase.
 3. Rellenar los valores de las propiedades. Por ejemplo, una instancia de vino de Rioja (Vino-Vino Tinto- Rioja) podría ser Marqués de Riscal.
- Esta instancia podría tener definidos los siguientes valores:
 - cuerpo: Poco
 - color: Rojo
 - sabor: Suave
 - nivel de tanino: Bajo
 - uva: Tempranillo
 - fabricante: Marqués de Riscal (instancia de la clase Bodega)
 - localización: Rioja Alta (instancia de la clase Región)
 - nivel de azúcar: Seco

Paso 5: Construcción de la Ontología

Se propone realizar el diseño del modelo conceptual utilizando la herramienta CMAP TOOLS COE y Protégé así:

1. Crear un modelo formal de los elementos identificados en los pasos anteriores. En este paso se utiliza la herramienta **CMAP TOOLS bajo la filosofía COE** (Concept-map Ontology Environment) . En este paso se definen las clases, la jerarquía de clases y demás relaciones siguiendo los patrones COE en lo posible.
2. El modelo formalizado se convierte en un modelo computable, es decir en un formato entendible por el computador para su procesamiento. En este punto se utiliza la herramienta **Protégé** para cargar un archivo RDF o OWL el cual es exportado desde la herramienta CMAP TOOLS con la utilidad de exportar. También se deben cargar las ontologías a ser alineadas y realizar las conexiones del caso.
3. Realizar ajustes de manera iterativa hasta obtener la ontología deseada, comprobándola con los objetivos del primer paso.

Paso 5: Construcción de la Ontología

*Existen varios **editores de ontologías gratuitos**. Entre ellos, destacan:*

- *Protégé [<http://protege.stanford.edu/>]*
- *Kaon [<http://kaon.semanticweb.org/>]*
- *OILed [<http://oiled.man.ac.uk>]*
- *ORIENT*
[<http://www.alphaworks.ibm.com/tech/semanticskt>].

Se escoge Protégé por se el más usado, documentado, fácil de usar y soporta diferentes lenguajes de ontologías y razonadores, además de ser extensible.

Paso 6: Evaluación de la Ontología

Evaluación utilizando la Herramienta Protégé

- **Validación de la Ontología**
 - Mediante el uso del razonador Pellet, Protégé permite validar algunos aspectos de la ontología, entre ellos: chequear la consistencia de la ontología, obtener automáticamente la clasificación taxonómica y computar los tipos inferidos.
- **Comprobación de la consistencia**
 - Permite constatar que no existen Contradicciones en la ontología. La semántica de OWL define una especificación formal para la definición de la consistencia en una ontología empleando Pellet
- **Verificación de inferencia de clases**
 - Encuentra las clases más específicas a las que pertenece una instancia; en otras palabras, determina la clase a la que pertenece cada uno de los individuos

Paso 6: Evaluación de la Ontología

Evaluación utilizando otros mecanismos

■ Evaluación de **Versiones**

- Ya que la creación es un proceso iterativo, cada vez que se obtenga una versión para evaluación preliminar se relacionan: #clases, #atributos, #relaciones por versión y luego se pueden comparar como variaron.

■ Evaluación a **Nivel Estructural**

- Para este tipo de evaluación se consideran los criterios provistos en **Gangemi et al.** Ya que permiten evaluar la ontología en función de su estructura y funcionalidad. En cuanto a la escala de evaluación, se puede tomar en consideración la proporcionada por **Yao, Orme y Etzkorn (2005)** así:

Cuadro 15: Escala de valores para la evaluación

Criterio	Valor
Bajo	0.00
Moderado	0.25
Promedio	0.50
Alto	0.75
Excelente	1.00

Cuadro 16: Métricas de Cohesión de OntoWikiUTPL

Criterio	Valor
Número de clases raíces (NCR)	14
Número de clases hoja (NCH)	17
Promedio de profundidad de herencia del árbol de nodos hoja (PPH-ANH)	0.86

Paso 6: Evaluación de la Ontología

- Evaluación como Software
 - **Emisión de un juicio** técnico del contenido con respecto a un marco de referencia. Establecer si satisface los criterios de diseño establecidos.
 - **Verificación:** se chequea la construcción correcta, es decir, que las definiciones implementen los requerimientos y den respuestas a las preguntas de competencia preestablecidas.
 - **Validación:** se refiere a que las definiciones de la ontología modelen lo más exactamente posible el dominio para el cual fueron creadas.





Paso 6: Evaluación de la Ontología

■ Evaluación como Software

- Considerando tres posibles estados de las ontologías se puede hacer evaluaciones:
 - **Pre-modelado:** Revisión y evaluación de los materiales disponibles para la construcción de la ontología
 - **Modelado:** Se comprueba la calidad de los significados y la consistencia y redundancia de los conceptos, utilizando otras ontologías disponibles y las preguntas de competencia, además se evalúan los posibles errores sintácticos cometidos durante la codificación.
 - **Entregadas:** consiste en determinar su calidad, comparando contra otras ontologías diferentes pero equivalentes. Para ello se establece un conjunto de criterios cualitativos y cuantitativos, que se miden mediante un conjunto de métricas preestablecidas.

Diferentes Perspectivas de Evaluación

Criterios de evaluación de ontologías

Criterio Autor	 Taxonomía	 Lenguaje	 Aplicación	 Vocabulario	Arquitectura Requerimientos	Aceptación Social	Razonamiento Automático	Software
(1)	√	-	√	√	-	-	-	-
(2)	-	√	√	√	-	-	√	-
(3)	√	-	√	√	-	-	-	-
(4)	-	√	√	√	-	√	-	-
(5)	√	√	√	√	√	-	-	-
(6)	√	√	√	√	√	-	-	√

(1) Brewster y cols, 2004

(2) Obrst y cols, 2007

(3) Porzel y Malaka, 2004

(4) Burton-Jones y cols, 2005

(5) Brank y cols, 2005

(6) Lozano-Tello, 2002

√ Considera - No considera

La mayoría se evalúa sobre la comparación con ontologías existentes

[1] E. Ramos, H. Núñez, and R. Casañas, "Esquema para evaluar ontologías únicas para un dominio de conocimiento," *Revista Venezolana de Información, Tecnología y Conocimiento*, 2009.

Cuando las computadoras tengan la habilidad de razonar, nos van a cuestionar, porque aún no tienen la habilidad de la creatividad, la cual no es posible formalizarla, dado su carácter etéreo y místico, lo cual lleva a introducir errores o acciones que no van con la lógica.

Miguel Angel Niño, 2015

Taller de Construcción de Ontologías

Modelo COE, ¿Qué es? ¿Qué se puede hacer?

- COE es una herramienta para la visualización, composición y edición integrada de ontologías RDF / OWL, integrado a la suite de software de mapas conceptuales IHMC CmapTools. Con COE se puede:
 - **Importar** OWL/RDF a partir de archivos RDF/XML, sobre la web a través un URI y presentarla en el formato CMAP.
 - **Analizar** la ontología viendo sus triples en varios formatos, que enumera los conceptos que utiliza, y la búsqueda de relaciones conceptuales.
 - **Editar** la ontología usando CmapTools, la creación de nuevos nodos y arcos de operaciones de manera visual y con la posibilidad de usar y crear plantillas de patrones.
 - **Navegar** rápidamente a través de una ontología, encontrando rápidamente los casos concepto, todos los triples que contienen un concepto dado, etc.
 - **Guardar o publicar** su Cmap como una imagen o un archivo XML.
 - **Colaborar** con otros usuarios COE remota en tiempo real.
 - **Grabar y reproducir** sus sesiones COE para la formación o con fines estadísticos
 - **Exportar** su Cmap a OWL / RDF en varios formatos, incluyendo RDF / XML, Turtle y N3.

Breve Capacitación en COE

- Interrelaciones entre individuos y Clases.
 - Maria es la madre de Ana.
 - Maria cumple años el 08/10/1950.
 - María es una Madre (is a).
- Interrelaciones entre Clases
 - Una madre es una persona. (are)
 - Una madre es igual a un padre femenino. (same as)
 - Una madre no puede ser un padre. (different from)
 - El padre de Ana, Juan esta muerto y la madre de Ana esta viva.
Vivo y Muerto son opuestos (exact opposite of)
- Describiendo Clases
 - Definir una clase, listando sus miembros.
 - **Categorización de Clases:** Ej. Definir los países de norte América . (one of).

Breve Capacitación en COE

■ Describiendo Clases

- Una clase se puede definir también como la combinación de condiciones de otras clases.
 - **Satisface Todas las condiciones (intercepción de clases) (all of) (AND):**
Ej. Un Hombre lo podemos definir como una persona de género masculino
 - **Satisface algunas condiciones (unión de clases) (any of) (OR):** Ej. El género Femenino es una mujer, chica o señora.
 - Por medio de una restricción de una propiedad. Hay varios clases de restricciones así:
 - **Clases que deben tener al menos una instancia con otra (can be):**
Ej. Madres de hijos son madres que al menos pueden tener un varón.
 - **Clases de exigen que todos los valores relacionados están en la otra clase (must be) y se puede combinar con (at least, at most, exactly):** Ej. Padres de hijos con al menos dos hijos.
 - **Clases con restricciones sobre individuos (must be exactly).** Ej: la clase madre de Mary es madre sólo de Mary.
 - **Las restricciones son siempre tratadas en COE como condiciones necesarias** en una clase, más que definiciones exactas. COE las coloca en rojo. Ej. Madre Americana de hijo.

Breve Capacitación en COE

- Relaciones entre Propiedades
 - Una *propiedad de un objeto* puede tener asociado un **Dominio** y un **Rango**, los cuales son clases.
 - Cualquier cosa en la que se aplica una propiedad, debe estar en el dominio y cualquier valor que tenga debe estar en el rango.
 - Ej. Definir dominio y rango de Una madre es madre de una persona.
- Las propiedades pueden ser declaradas para satisfacer algunas condiciones matemáticas útiles.
 - **Simétrica** (\Leftrightarrow): aplica en ambos sentidos, la dirección del enlace no es importante. Ej. Propiedad esHermanoDe
 - **Transitiva** ($\Rightarrow \Rightarrow$): se hereda por las cadenas de relaciones. Ej. serUnAntepasado.
 - **Funcional** ($\Rightarrow =$): es cuyo valor es siempre único. Ej. tieneMadre
 - **Inversa funcional** ($= \Leftarrow$): si no hay dos o más cosas que tienen el mismo valor de esa propiedad. Ej. tieneIdentificación

Breve Capacitación en COE

- Distinguir entre los tipos básicos de propiedades. Sólo se debe exigir esto en el modelo cuando se crean Ontologías OWL-DL
 - Existen dos tipos básicos de propiedades:
 - **Propiedades de Objeto** (**ObjectProperty**): relacionan objetos con un enlace.
 - **Propiedades de Datos** (**DataProperty**): relacionan un objeto con un tipo de datos.
 - Dos **propiedades** pueden ser **inversas** una a otra. Ej. esMadreDe y tieneMadre.
 - Una **propiedad** puede ser **subpropiedad** de otra. Ej. esMadreDe puede ser subpropiedad de esParienteDe

Pasos para crear una ontología en CMAPS COE

1. Seleccionar un Estado del Arte o fuente de conocimiento confiable.
2. Crear un Clase y verificar
 1. Propiedades de datos (atributos)
 2. Individuos (si son básicos)
3. Crear relaciones entre clases (**ObjectProperties**)
 1. Verificar Equivalencias – Diferencias
same as, same class as, different from
 2. Verificar Características especiales de relaciones
cannot be (disjuntas), exact opposite of (exclusión máxima extremos)
 3. Describir las clases como condiciones de otras
one of, all of, any of, can be, must be, must be + [at least, at most, exactly]
4. Definir las Restricciones de las Propiedades
 1. Crear Dominios y Rangos con plantilla (**Domain and Range**), quedan líneas punteadas.
 2. Definir las condiciones matemáticas si las hay
simetric (<=>), transitive (=>=>), funtional (>>=), inverse functional (= <<=)
 3. Asegurarse que tipo de propiedades son, si son inversas o subpropiedades
ObjectProoerty / datatypePorperty, inverse, subProperty

Pasos de uso CMAPS COE

5. Preguntas para decidir sobre relaciones de Clase

- ¿B es subclase de A?: si (B **are** A), sino (B Objectproperty A)
- ¿B puede relacionarse con A? si (A **can be** B), sino (A objectproperty B)
- ¿B debe relacionarse con A?: si (A **must be** B), sino (A objectproperty B)

Cuantificadores

- **Existencial: can be (some)**: clases de individuos que participan en al menos una relación en una propiedad específica para miembros de una clase determinada
- **Universal: must be (only)**: clases de individuos que para una determinada propiedad sólo tienen relaciones con los miembros de una clase particular.

Ejemplo: Ontología de Vehículos de Transporte

Marco de Referencia

- Se desea crear una aplicación que permita consultar información sobre los diferentes vehículos de transporte, de la siguiente manera:
 1. Identificar conceptos del dominio, examinando los términos de la consulta del usuario.
 2. Ampliar la información de conceptos, buscando términos más especializados o más generales.
 3. Responder preguntas tales como:
 - ¿Qué caracteriza un vehículo?
 - ¿Qué tipos de vehículos podemos tener?
 - ¿Cuál es el ambiente de trabajo de los vehículos?
 - ¿Cuánto cuesta un pasaje de bus?

Ejemplo: Marco de referencia del Dominio

- Descripción del Dominio

Un vehículo es un medio de transporte y también puede ser un medio de carga. Los medios de transporte se pueden clasificar como aéreos (aviones, avionetas), marítimos (lancha, barco) y terrestre (auto, camioneta, bus). Cuando se utilizan los vehículos como medio de carga podemos diferenciar que transportan dos tipos (pesada y liviana). El medio de transporte utilizado depende del ambiente de trabajo que usa, así si el ambiente es vía terrestre se puede viajar por ferrovía o calles, si es vía marítima se puede viajar por río, mar y océano y finalmente si es por vía aérea se puede viajar en el espacio aéreo. Sin embargo el medio de transporte tiene un alcance que depende de la distancia que alcanza y se mide en kilómetros enteros. El alcance puede clasificarse como local, nacional o internacional y si es nacional estamos hablando que es interprovincial o intercantonal. Un vehículo es usado por un usuario y a su vez podemos decir que un usuario usa un vehículo, este usuario puede ser un pasajero o un conductor, si es un pasajero entonces tiene un pasaje con un valor, pero si es un conductor tiene una licencia identificada alfanuméricamente.

Ejemplo: Marco de referencia del Dominio

- Descripción del Dominio (continuación)

Es necesario conocer los nombres de los usuarios. Un vehículo puede ser descrito mediante el año de fabricación, la capacidad y el tipo de vehículo que puede tomar valores (personal, público). Los vehículos utilizan un mecanismo de desplazamiento como alas, hélices y llantas, que permiten moverlos, así un medio de desplazamiento mueve un vehículo y a su vez un vehículo es movido por un medio de desplazamiento. Finalmente se sabe que los vehículos deben usar un combustible que puede ser gasolina, diesel o especial. Es importante saber cual es el precio del combustible y el octanaje.

- Desarrollar la ontología utilizando CMAP Tools COE y Protégé.



Trabajo en
Taller

Obteniendo Información: SPARQL

- (**S**imple **P**rotocol and **R**DF **Q**uery **L**anguage), es un lenguaje de consulta sobre RDF, que permite hacer búsquedas sobre los recursos de la Web Semántica utilizando distintas fuentes de datos.
- El objetivo de la Web Semántica es que las máquinas comprendan la información de la Web. Se utiliza RDF con la sintaxis de XML para expresar la información (metadatos).
- Con esta información las máquinas pueden comprender la información y utilizar la lógica y otras reglas para establecer relaciones y realizar deducciones, gracias a SPARQL.
- Para esto utiliza razonadores como: Pellet, Fact++ y Hermit

Estructura de la Consulta

SELECT ?x ?y ... ?n

WHERE { ?x predicado objeto .
 sujeto ?y objeto .

 sujeto predicado ?n .
 }

Ejemplo



```
SELECT ?materia
WHERE
{
  ?materia esDictadoPor JuanPerez .
}
```

Consultas a la Ontología

- **Consulta General**

```
select * from  
<file:/C:/Users/Brayan/Documents/clases/sistemas%20del%20Conocimiento/owl/Vehiculo%20V5.owl> where  
{?s ?p ?o}
```

- **Cuánto cuesta un boleto en bus**

```
select * from  
<file:/C:/Users/Brayan/Documents/clases/sistemas%20del%20Conocimiento/owl/Vehiculo%20V5.owl> where  
{<http://www.co-ode.org/ontologies/ont.owl#Pasajero1>  
<http://localhost/default#tienePasaje> ?o}
```

Consulta a la Ontología

■ Usuarios existentes

PREFIX table: <http://localhost/default#tieneNombre>

PREFIX ruta:

<file:/C:/Users/Brayan/Documents/clases/sistemas%20del%20Conocimiento/owl/Vehiculo%20V5.owl>

SELECT ?Modelo ?Descripcion

FROM ruta:

WHERE { ?Modelo table: ?Descripcion FILTER
regex(?Descripcion, "Pablo", "i")}

Ejemplos API VSNET

- Instalar del administrador de paquetes Nuget el IKVM.
- Descargar Jena de:
<http://psg.mtu.edu/pub/apache/jena/binaries/apache-jena-2.13.0.zip>
- Descargar OWLAPI
- Demostración de uso de APIS.

Muchas Gracias

Contacto:

Miguel Ángel Niño Zambrano

Programa de Doctorado en Ingeniería Telemática de la Universidad del Cauca

Afiliación: Profesor Titular Universidad del Cauca, Facultad de Ingeniería Electrónica y Telecomunicaciones – Grupo de Investigación en Ingeniería Telemática - GIT y Grupo de Investigación en Tecnologías de la Información –GTI

Correo: manzamb@unicauca.edu.co, manzamb@hotmail.com

¿Preguntas?

