UNIVERSITAT OBERTA DE CATALUNYA

Enginyeria Tècnica en Informàtica de Sistemes

XML i Web Semàntica

Alumne/a: Felix Abarca Corrales

Dirigit per: Antoni Pérez Navarro

CURS 2004-05 Setembre

Resum

La Web Semàntica constitueix una nova visió de la web actual. Es pretén aconseguir una web molt més estructurada, on els continguts tinguin una semàntica explícita i estiguin lògicament relacionats entre ells. Les màquines prendran, aleshores, el rol que fins ara desenvolupen els humans en l'explotació de les dades de la web. El desenvolupament i l'expansió d'aquesta xarxa de coneixement estructurada, permetrà l'aparició d'un nou tipus de programari intel·ligent que podrà inferir coneixements gràcies a les relacions semàntiques indicades en els documents. En aquest treball es realitza una descripció dels llenguatges que constitueixen la base de la construcció de la Web Semàntica: l'XML, l'RDF i l'OWL. El seu rol i el seu grau d'importància dins del desenvolupament de la Web Semàntica s'analitza al llarg de la present memòria. El concepte d'ontologia com a mitjà de representació estructurada de la informació també es discuteix en el treball. En concret, es realitza un estudi de la base de dades lèxica *WordNet*. Finalment, es presenta el disseny i la implementació d'una ontologia per representar les relacions lèxiques dels mots del català. A partir d'aquesta ontologia es crea una petita base de dades basada en la temàtica dels animals de companyia. Aquest cas pràctic permet extreure conclusions sobre els avantatges d'introduir metadades en els documents electrònics, i sobre les facilitats que ofereixen les aplicacions actuals pel desenvolupament d'aquest tipus de documents.

Índex de Continguts

Índex de Figures

Índex de Taules

Capítol 1 - Introducció

En aquest capítol, el lector trobarà conceptes generals, com ara la justificació del treball (apartat 1.1), els seus objectius generals (apartat 1.2) i el conjunt de productes obtinguts com a resultat de l'estudi (apartat 1.5). També es presenten de manera somera les tasques i les fites identificades en la fase preliminar de planificació del projecte (apartat 1.4). Finalment, es proporciona una visió global de l'estructura d'aquesta memòria (apartat 1.6).

1.1 Justificació del TFC

La proliferació de la Web comporta la necessitat de tractar dades estructurades. **XML** (de l'anglès, *eXtensible Markup Language*) és el llenguatge que, avui en dia, es fa servir per intercanviar informació entre diferents sistemes d'informació atès la flexibilitat i independència de les fonts de dades i de formats fixes que proporciona.

Actualment es pretén que els ordinadors puguin entendre la semàntica de la informació que hi ha a la web i sigui possible integrar diverses fonts d'informació malgrat la procedència i formats diferents de cadascuna d'elles: s'està buscant una Web Semàntica.

Aquesta nova àrea de recerca està clarament lligada amb el món de les bases de dades, les quals hauran d'oferir, i de fet en molts casos ja ofereixen, un nou ventall de possibilitats per poder tractar, emmagatzemar, i gestionar aquest nou tipus d'informació estructurada, és a dir, classificada segons uns criteris semàntics i interrelacionada amb conceptes afins, alhora que permetin la comunicació entre sistemes de manera transparent a l'usuari.

1.2 Objectius del TFC

L'objectiu fonamental d'aquest treball és la introducció al concepte de la Web Semàntica, les ontologies i el coneixement de les eines disponibles actualment que permeten el seu disseny i construcció.

De manera més detallada, els objectius del TFC són els següents:

- Conèixer els conceptes bàsics relacionats amb la Web Semàntica i les ontologies,
- Conèixer els llenguatges disponibles actualment per a la representació de dades estructurades, les seves relacions i la seva distribució,
- Analitzar el programari actual relacionat amb el disseny i la construcció de la Web Semàntica,
- Realitzar un cas pràctic per tal de consolidar els conceptes teòrics.

1.3 Enfocament

Partint dels objectius definits a l'apartat anterior, el treball es pot dividir en dues parts ben diferenciades:

1. La primera part proporciona una introducció a les tècniques i tecnologies que proporcionen la base teòrica per a la construcció d'ontologies i la Web Semàntica. Aquesta part la formen els capítols 2, 3, 4, 5 i 6.

2. La segona part té un enfocament més pràctic i en ella es descriu l'ús d'un programari determinat, el *Protégé*, i es desenvolupa un cas simple, però il·lustratiu, de les tecnologies presentades a la primera part del treball. Aquesta part comprèn el capítol 7.

1.4 Planificació del projecte

En aquest apartat es presentaran de forma resumida les tasques planificades, així com les fites establertes. Cal tenir present que la planificació del present projecte ha estat objecte d'un informe dedicat [1]. En aquest informe, es descriuen amb detall les diferents tasques i fites identificades per a la consecució dels objectius presentats a l'apartat 1.2.

La llista de tasques identificades i la seva durada estimada és la següent:
- Estudi dels llenguatges XML i XML-Schema (2 setmanes)
- Estudi dels llenguatges RDF i RDF-Schema (3 setmanes)
- Estudi del llenguatge OWL (4 setmanes)
- Eina de desenvolupament *Protégé* (1 setmana)
- Disseny d'una web semàntica (3 setmanes)
- Lliurament PAC 2 (1 setmana)
- Lliurament PAC 3 (1 setmana)
- Lliurament Final (2 setmanes)

Quant a les fites acordades i el seu contingut, es poden resumir en la taula següent:

Fita	Contingut	Data
Lliurament preliminar de la PAC 2	Esborrany dels capítols 1, 2 i 3 de la memòria	11 d'abril
Lliurament final de la PAC 2	Capítols 1, 2 i 3 de la memòria	18 d'abril
Lliurament preliminar de la PAC 3	Esborrany dels capítols 4 i 5 de la memòria	16 de maig
Lliurament final de la PAC 3	Capítols 4 i 5 de la memòria	23 de maig
Lliurament preliminar de la memòria i la presentació	Esborrany de la memòria completa del treball i presentació	6 de juny
Lliurament final de la memòria, presentació i treball pràctic	Memòria completa del treball, presentació i material del cas pràctic	13 de juny

Taula 1 – Fites del projecte

1.5 Productes obtinguts

A la fi del present projecte els productes obtinguts són, bàsicament, els esmentats com a contingut de la darrera fita de la Taula 1, és a dir:

- **Memòria del projecte.** El present document. Donades les característiques pròpies del funcionament de la UOC, es lliurarà en format electrònic *PDF*.

- **Presentació del projecte.** Es tracta d'una presentació del contingut del treball en format de diapositives per a la seva exposició. De la mateixa manera que en el cas anterior, també es lliurarà en format electrònic *PowerPoint*.

- **Productes del cas pràctic.** En aquest cas, es tracta dels fitxers produïts amb el programari utilitzat per dissenyar i implementar una Web Semàntica.

1.6 Estructura de la memòria

La memòria està estructurada amb l'objectiu de proporcionar una aproximació gradual al tema central del TFC: la construcció, l'aprenentatge i la comprensió d'una Web Semàntica.

Així doncs, els primers capítols donen una introducció als diferents llenguatges que, d'una manera més o menys directa, poden contribuir a la construcció de la Web Semàntica. És el cas del capítol 2, on es proporciona una descripció d'XML i d'XML Schema. En la mateixa línia, el capítol 3 descriu els llenguatges RDF i RDF Schema.

En el capítol 4 es presenta el concepte d'ontologies aplicades al camp de la informàtica. En aquest mateix capítol s'ofereix una visió general de l'ontologia *WordNet.* Per completar els coneixements sobre ontologies, el capítol 5 proporciona una descripció força detallada del llenguatge OWL i una introducció al programari *Protégé*.

En el capítol 6 es presenta la visió de la Web Semàntica tal i com ha estat concebuda pel W3C.

En el capítol 7 es descriu la realització del cas pràctic (la creació d'una petita Web Semàntica), tant des del punt de vista del seu disseny com de la seva implementació.

Finalment, el capítol 8 presenta les conclusions i possibles línies futures d'investigació.

Capítol 2 - XML i XML-Schema

Aquest capítol proporciona una introducció als llenguatges XML i XML Schema. En primer lloc es parlarà de l'XML (apartat 2.1) i dels espais de noms (apartat 2.1.2). Després s'introduiran les DTD (apartat 2.2.1) i la seva evolució en l'XML Schema (apartat 2.2.2). A continuació es citaran algunes aplicacions basades en aquest llenguatge (apartat 2.3) i s'exposaran alguns dels seus avantatges i inconvenients (apartat 2.4). Per acabar el capítol, es presentaran les conclusions (apartat 2.5).

2.1 El llenguatge XML

El llenguatge extensible de marques **XML** (*eXtensible Markup Language*) és un metallenguatge, és a dir, un llenguatge que permet descriure altres llenguatges. El que realment vol dir això és que XML és més aviat un format estàndard per a l'estructuració de dades que no pas un llenguatge de programació.

La definició actual d'XML [2] és una recomanació del W3C (*World Wide Web Consortium*) [3] i està basat en l'estàndard SGML (*Standard Generalized Markup Language*) [4]. De fet, XML va ser creat com una especialització de l'SGML per a aplicacions web.

XML ofereix la possibilitat de definir etiquetes (en anglès *tags*) i les relacions estructurals entre elles. Com a resultat, els dissenyadors poden crear etiquetes específiques segons les seves necessitats, per tal de definir, intercanviar i validar informació entre múltiples sistemes i aplicacions.

2.1.1 Sintaxi XML: documents ben formats

Els documents XML es componen de dos elements bàsics: els **elements** i les **etiquetes**. Els elements són les entitats, és a dir, el contingut del document, mentre que les etiquetes es fan servir per descriure i estructurar els elements. A simple vista, un document XML és una estructura jeràrquica d'etiquetes que organitza els elements.

Tot document XML comença amb una declaració que l'identifica com a document de tipus XML. Tot i que no és obligatòria, es considera com una bona pràctica.

L'aspecte d'aquesta declaració és: `<?xml version="1.0"?>`. Aquesta declaració pot incloure altres informacions, com ara el joc de caràcters que es fa servir en el document: `<?xml version="1.0" encoding="UTF-16"?>`.

Les etiquetes en un fitxer XML segueixen el format següent:

```
<nom_etiqueta> …… </nom_etiqueta>
```

És a dir, sempre hi ha una etiqueta que marca l'inici de l'element, i una altra que marca l'acabament. Les etiquetes poden estar estructurades de forma jeràrquica, és a dir, de manera que unes etiquetes continguin les altres, tal com es mostra a l'Exemple 1.

```
<?xml version="1.0"?>
<etiqueta_1>
      <etiqueta_1_1>
            <etiqueta_1_1_1>Descripcio de l'element</etiqueta_1_1_1>
            <etiqueta_1_1_2>Descripcio de l'element</etiqueta_1_1_2>
      </etiqueta_1_1>
      <etiqueta_1_2>Descripcio de l'element</etiqueta_1_2>
</etiqueta_1>
```

Exemple 1 – Estructura jeràrquica d'etiquetes XML

Per tal que l'estructura jeràrquica sigui consistent, és indispensable que la definició de les etiquetes sigui coherent, és a dir, no és correcte que l'etiqueta d'acabament d'un element s'especifiqui abans que l'etiqueta d'acabament de tots els seus subelements. L'Exemple 1 és una mostra de jerarquia correcta, mentre que a l'Exemple 2 la jerarquia és incorrecta, ja que l'etiqueta de tancament `</etiqueta_1_1>` apareix abans que l'etiqueta `</etiqueta_1_1_2>`.

```
<?xml version="1.0"?>
<etiqueta_1>
      <etiqueta_1_1>
            <etiqueta_1_1_1>
            </etiqueta_1_1_1>
            <etiqueta_1_1_2>
            </etiqueta_1_1>
            </etiqueta_1_1_2>
</etiqueta_1>
```

Exemple 2 – Jerarquia incorrecta d'etiquetes

En tot document, sempre existeix un element **arrel** a partir del qual s'estructuren tots els altres. En el cas de l'Exemple 1, l'element arrel està identificat per l'etiqueta `etiqueta_1`. Una particularitat de les etiquetes és que poden contenir atributs que fan referència a l'element delimitat per les etiquetes. En aquest cas, la sintaxi de l'etiqueta és:

```
<nom_etiqueta nom_atribut_1="valor_atribut_1" … nom_atribut_n="valor_atribut_n">
```

De manera general, per tal que un document XML sigui considerat sintàcticament correcte, ha de respectar una sèrie de regles:

1. Tota etiqueta no buida ha de tenir una etiqueta de tancament associada
2. L'organització jeràrquica d'etiquetes ha de ser perfecta
3. Un document només pot tenir un element arrel
4. Existeix una convenció per anomenar les etiquetes:

- El nom ha de començar per una lletra o pel caràcter _. Les xifres i signes de puntuació només es poden fer servir a partir del segon caràcter
- El nom no ha de tenir espais en blanc
- El nom no ha de tenir el caràcter ":"
- El nom no pot començar per la cadena `xml`
- El nom s'ha d'especificar tot just després del caràcter "<" que marca l'inici de l'etiqueta

5. Quan una etiqueta conté atributs, el valor de cadascun d'ells ha d'estar delimitat pel caràcter ".

Quan un document XML respecta totes aquestes regles es diu que el document està **ben format**, és a dir, que és sintàcticament correcte. Per acabar aquest apartat, es presenta l'Exemple 3 de document XML ben format:

```
<?xml version="1.0"?>
<Empresa nom="GTD" data_fundacio="1991-10-12">
	<Delegacio nom="Central BCN" empleats="100">Text associat</Delegacio>
	<Delegacio nom="Guayana" empleats="25">Text associat</Delegacio>
	<Delegacio nom="Madrid" empleats="1">
		<Empleat>
			<Nom>Juan Zapata</Nom>
			<Carrec>Director delegacio</Carrec>
		</Empleat>
	</Delegacio>
</Empresa>
```

Exemple 3 – Exemple de document XML ben format

2.1.2 Espais de noms

Una de les funcionalitats més importants d'XML és el fet de poder aprofitar la definició d'elements i atributs que s'ha fet en un fitxer, en la construcció de nous fitxers. És a dir, en la creació d'un fitxer XML es pot reutilitzar el vocabulari definit en d'altres fitxers XML.

Ara bé, es pot donar el cas que dos vocabularis continguin elements amb el mateix nom, però amb un significat diferent. Si es volen utilitzar ambdós vocabularis en la creació d'un fitxer XML, es necessita un mecanisme que permeti referir-se de manera inequívoca a un element d'un vocabulari precís. Aquest mecanisme s'anomena espai de noms (en anglès, *namespaces*). El consorci W3C va definir una especificació per a aquest mecanisme que es pot consultar a [5].

Els espais de noms que es facin servir en un fitxer XML s'han d'especificar dins l'etiqueta arrel de la forma següent:

```
<nom_etiqueta xmlns:prefix="id_vocabulari">
```

on `prefix` és una cadena de caràcters que es farà servir quan es vulguin crear elements o atributs que són definits en l'espai de noms que es troba en `id_vocabulari`.

Quan es vol crear dins un fitxer XML un element que segueixi l'estructura definida en l'espai de noms `xy`, s'ha d'indicar de la següent manera:

```
<xy:nom_element> ... ... </xy:nom_element>
```

Existeixen diversos projectes a la web coordinats per comunitats d'usuaris que tenen per objectiu la creació d'espais de noms detallats que puguin ser utilitzats en els documents XML d'altres usuaris. Un dels espais de noms més important és el

Dublin Core [6]. En termes generals, aquest espai de noms defineix una jerarquia de termes per descriure el propòsit, el context i l'origen d'un document. Per exemple, aquest vocabulari dóna una definició estàndard per al concepte de *creador* d'un document. Qualsevol document XML pot fer servir el vocabulari definit pel Dublin Core i, fins i tot, el pot enriquir per tal de satisfer les seves necessitats particulars.

2.2 DTD i XML Schema

A més de ben formats, els documents XML poden ser també **vàlids**. Per això, és necessari que segueixin una estructura determinada per una definició de tipus de document **DTD** (*Document Type Declarations*), o bé, per un **XML Schema**. En els següents subapartats es dóna una descripció d'ambdós llenguatges.

2.2.1 Les DTD

Les DTD estableixen quins elements es poden incloure dins d'un document XML, les relacions jeràrquiques entre ells i els atributs que es poden definir. Per incloure una DTD des d'un document XML, s'ha d'especificar una línia al principi del document que, en el cas més general, tindrà el format:

```
<!DOCTYPE nom_DTD SYSTEM "cami_DTD">
```

on *cami_DTD* indica una adreça URL o un directori on es troba la definició de la DTD. Els processadors de documents XML (en anglès *parsers*) utilitzen aquesta informació per comprovar si el document segueix la gramàtica definida en la DTD. En cas afirmatiu, es tractarà d'un document XML **vàlid**.

La definició de la sintaxi d'un document DTD es considera fora de l'abast del present treball. El lector interessat pot consultar [2]. Val a dir, però, que la seva sintaxi es força diferent d'XML i que té importants limitacions, per exemple, que no és possible definir tipus de dades o que la definició d'elements amb contingut mixt és poc flexible.

2.2.2 XML Schema

La proposta del W3C com a successor de les DTDs és el llenguatge XML Schema [7] i [8]. Aquest llenguatge té molts avantatges respecte les DTDs. D'entrada, proporciona una gramàtica més rica i variada per definir l'estructura dels elements. En XML Schema és possible, per exemple, especificar el nombre d'ocurrències exactes d'un determinat element, es poden definir valors per defecte i també es pot escollir un sol element d'un conjunt.

En primer lloc, una diferència important respecte les DTD és que XML Schema utilitza el llenguatge XML per a codificar els esquemes. Això simplifica el desenvolupament d'eines automàtiques pel processament i la validació dels esquemes, ja que es poden fer servir els mateixos *parsers* que per a XML.

L'especificació de l'esquema s'emmagatzema en un fitxer separat del document XML que el fa servir. Aquests fitxers tenen extensió *xsd*. La primera etiqueta especificada en un esquema és:

```
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

Un document XML que vulgui referir-se a un esquema determinat ha d'incloure una clàusula dins l'etiqueta arrel que indiqui com trobar el fitxer *xsd* associat a l'esquema. Això es fa de la mateixa manera que s'ha descrit a l'apartat 2.1.2 pel cas dels espais de noms, és a dir:

```
<etiqueta_arrel xmlns:prefix_esquema="cami_fitxer_xsd">
```

L'XML Schema proporciona la possibilitat de definir tipus pels elements. Així, per l'Exemple 3, es podria crear un esquema que establís que l'atribut `empleats` de l'element `Delegacio` ha de ser un enter, o bé, que l'atribut `data_fundacio` de l'element `Empresa` ha de ser de tipus `data`. En l'especificació del llenguatge [8] s'estableixen una sèrie de tipus bàsics, com ara, els *strings* ("Hola món"), els *booleans* (true, false), els *decimals* (7.08), *float* (1234E5), *time* (12:34:51.123)...

Però també s'ofereixen mecanismes per permetre als usuaris la possibilitat de definir nous tipus derivats dels predefinits. Per això s'utilitza l'element `<xsd:simpleType>` per definir el nou tipus simple, i l'element `<xsd:restriction>` per indicar quin és el tipus base i quines restriccions se li apliquen i que caracteritzaran el nou tipus. L'Exemple 4 mostra la definició d'un tipus simple basat en la definició d'un enter amb valors compresos entre 1000 i 9999 (ambdós inclosos):

```
<xsd:simpleType name="nouEnter">
      <xsd:restriction base="xsd:integer">
            <xsd:minInclusive value="1000"/>
            <xsd:maxInclusive value="9999"/>
      </xsd:restriction>
</xsd:simpleType>
```

Exemple 4 – Definició en XML-Schema d'un nou tipus simple

També és possible definir tipus complexos que es diferencien dels simples pel fet que poden tenir atributs i/o subelements. Per definir aquests tipus es fa servir l'element `<xsd:complexType>`. L'Exemple 5 demostra com construir un nou tipus anomenat `preuInternacional` que conté un atribut per indicar el tipus de moneda que rep el nom de `moneda` i és de tipus `string` (cadena de caràcters).

```
<xsd:element name="preuInternacional">
      <xsd:complexType>
            <xsd:simpleContent>
                  <xsd:extension base="xsd:decimal">
                        <xsd:attribute name="moneda" type="xsd:string"/>
                  </xsd:extension>
            </xsd:simpleContent>
      </xsd:complexType>
</xsd:element>
```

Exemple 5 – Definició en XML-Schema d'un nou tipus complex

L'etiqueta `<xsd:simpleContent>` indica que l'element `preuInternacional` només contindrà dades, però no subelements. L'etiqueta `<xsd:extension>` és similar a l'etiqueta de l'Exemple 4, en el sentit que permet derivar un tipus predefinit. En aquest cas, es deriva el tipus `xsd:decimal` afegint-hi un nou atribut mitjançant l'element `<xsd:attribute>`.

Aquests mecanismes d'inclusió i derivació que s'acaben d'introduir en l'Exemple 4 i l'Exemple 5, constitueixen un altre avantatge respecte les DTD, ja que permeten la reutilització de definicions d'elements o l'adaptació d'elements existents a les necessitats particulars dels usuaris.

2.3 Aplicacions basades en XML i XML Schema

XML i XML Schema es poden utilitzar en qualsevol aplicació que pretengui emmagatzemar, recuperar o tractar informació estructurada i validar la seva estructura i contingut. Segons això, pràcticament tota aplicació informàtica els pot fer servir.

Algunes de les aplicacions d'XML serien:

- Publicació: XML com a substitut d'HTML, XML com a format de documents científics.
- Intercanvi d'informació entre aplicacions
- Persistència: configuració d'aplicacions, serialització d'objectes, bases de dades XML.

Existeixen multitud d'eines que permeten treballar amb XML. A Internet es poden trobar nombroses pàgines Web dedicades a recopilar tota mena d'utilitats per treballar amb aquests llenguatges [9], [10] i [11]. En molts casos, es tracta d'eines gratuïtes.

2.4 Avantatges i inconvenients d'XML

Entre els avantatges del llenguatge XML es poden destacar els següents:

- És compatible amb tots els protocols de transport de dades: SMTP, HTTP, FTP, etc, ja que va ser dissenyat per ser utilitzat en aplicacions web
- La tecnologia per processar els documents XML és molt barata o fins i tot gratuïta
- XML és un llenguatge molt flexible pel fet que les etiquetes són definides pels dissenyadors
- L'estàndard XML és obert i en la seva definició han participat nombroses institucions
- XML permet l'intercanvi consistent d'informació independentment de les diferències de plataforma o de les aplicacions que exploten els documents

Com a inconvenients del llenguatge es poden citar:

- Els documents XML no estan optimitzats en termes d'ocupació d'espai en disc, temps de processament del seu contingut o temps de transferència entre diferents màquines
- XML és ideal per informació de tipus text, però no serveix per contenir dades binàries
- Els DTDs resulten insuficients per determinades aplicacions on es requereix un control més exhaustiu sobre el tipus d'informació del document XML, i els XML Schema poden resultar massa complexos

Un informe detallat sobre els avantatges i inconvenients en l'ús d'XML es pot trobar a [12].

2.5 Conclusions

En termes de la Web Semàntica (veure 6.1), XML no proporciona cap interpretació de les dades que codifica. Per als humans, l'estructura jeràrquica d'un document XML ja permet obtenir certa informació sobre el significat de la informació que conté. Però per a una màquina no hi ha cap especificació formal ni de la semàntica ni de l'ús que es fa de la informació.

Per assolir aquests nous objectius, que són claus per a la construcció de la Web Semàntica, es necessita desenvolupar altres llenguatges que permetin especificar de forma inequívoca la semàntica dels elements i les seves possibles relacions. L'RDF i l'RDF Schema són dos llenguatges que permeten assolir, en part, aquests objectius.

Capítol 3 - RDF i RDF-Schema

Aquest capítol proporciona una introducció als llenguatges RDF i RDF Schema. En primer lloc es tractarà l'RDF (apartat 3.1). Tot seguit, es comentaran els trets principals de l'RDF Schema (apartat 3.2). A continuació es citaran certes aplicacions basades en aquests dos llenguatges (apartat 3.3). Finalment s'exposaran les conclusions les capítol (apartat 3.4).

3.1 El llenguatge RDF

Actualment, la major part dels continguts disponibles a Internet estan pensats per ser explotats pels humans. D'aquesta manera, la informació està més aviat poc estructurada, ja que està pensada per a ser visualitzada, i això implica, entre altres coses, que les cerques de material siguin poc acurades. Aquest model esdevé cada cop menys i menys pràctic ja que el volum d'informació no para de créixer. Seria necessari un model que permetés un accés uniforme i racional a la informació, així com aplicacions intel·ligents capaces de processar grans volums de dades.

El llenguatge **RDF** (*Resource Definition Framework*) és una proposta per a aquest nou model d'estructuració de la informació. L'RDF defineix un model de dades que suporta la fàcil i ràpida integració de diferents fonts d'informació utilitzant conceptes semàntics. L'especificació del model de dades i de la sintaxi de l'RDF són recomanacions del consorci W3C publicades en els documents següents:

- *RDF Primer [13],*
- *Resource Description Framework (RDF): Concepts and Abstract Syntax [14],*
- *RDF Semantics [15],*
- *RDF Test Cases [16].*

3.1.1 Les URIs

Una **URI** (de l'anglès *Universal Resource Identifier*) és un mecanisme simple i extensible per identificar un recurs [17]. L'anàlisi de cadascun dels mots que formen l'acrònim URI, permet caracteritzar i definir completament què és una URI:

- *Uniform.* La uniformitat proporciona diversos avantatges. Permet la introducció de nous tipus d'identificadors de recursos sense que això interfereixi en la forma en la qual s'utilitzen els actuals. Permet que els identificadors es puguin fer servir en múltiples contexts.
- *Resource.* Un recurs és tot allò que pugui ser identificat mitjançant una URI. Com a exemples típics es poden citar els documents electrònics, les imatges, una font d'informació (*un diari*), un servei (*una passarel·la HTTP-SMS*) o una col·lecció d'altres recursos. És important destacar que un recurs no ha de ser necessàriament accessible a través d'Internet. De la mateixa manera, els conceptes abstractes, els tipus de relacions (*amistat*) o, fins i tot, els valors numèrics (*zero, infinit*) també poden ser recursos.
- *Identifier.* El propòsit d'un identificador és distingir un recurs de la resta de recursos, independentment de com s'aconsegueixi aquesta distinció (per nom, adreça o context).

En definitiva, una URI és una cadena de caràcters que permet identificar un recurs independentment del context en què es trobi i de si és accessible o no a partir d'Internet.

Cal esmentar que les URLs (de l'anlès *Uniform Resource Locator*) són un subconjunt de les URIs que, a més d'identificar un recurs, proporcionen els mitjans per localitzar-lo descrivint el seu mecanisme d'accés principal, és a dir, la seva localització a la web. Això vol dir que les URLs només permeten identificar recursos que són accessibles a través d'Internet.

3.1.2 El model de dades

El model de dades de l'RDF utilitza les **URIs** (*Universal Resource Identifier*) per identificar de manera inequívoca els **objectes**. Les relacions entre els objectes s'anomenen **propietats**. Tot i que aquest mecanisme pugui semblar simple, és prou efectiu per representar i integrar la informació, ja que proporciona els elements mínims comuns a tots els models d'informació.

El model de dades de l'RDF recorda vagament al paradigma de l'orientació a objecte. Està format per **entitats**, representades per un identificador únic URI, i per relacions binàries, anomenades **sentències**, que relacionen entitats entre sí. D'una manera gràfica, en una sentència, l'origen de la relació és el **subjecte**, el final és **l'objecte** i l'arc que els uneix és la **propietat** (o també el predicat).

El model de dades diferencia els **recursos**, que són objectes que tenen associat una URI, dels **literals**, que són simples cadenes de caràcters. El subjecte i la propietat d'una sentència són sempre recursos, mentre que l'objecte pot ser també un recurs, o bé, un literal. Gràficament parlant, els recursos es representen mitjançant ovals i els literals mitjançant capses. A la Figura 1 i la Figura 2 es mostren exemples d'aquesta representació gràfica.

Figura 1 - Exemple simple de sentència RDF

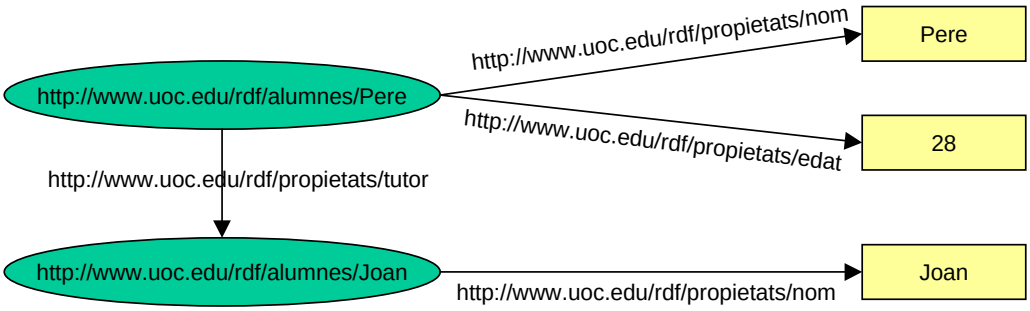

Figura 2 - Exemple més elaborat de sentència RDF

La Figura 1 mostra que l'entitat identificada per l'URI `http://www.uoc.es/rdf/alumnes/Pere` està relacionada amb el literal (el valor constant) `Pere` mitjançant una propietat identificada per l'URI `http://www.uoc.es/rdf/propietats/nom`. En un llenguatge més natural, es diria que el recurs `Pere` té com a `nom Pere`.

A la Figura 2 es mostra com l'entitat `Pere` està relacionada amb els literals `Pere` i `28` a través de les propietats identificades per les URIs `http://www.uoc.es/rdf/propietats/nom` i `http://www.uoc.es/rdf/propietats/edat`, respectivament. Al mateix temps, també es relaciona amb una altra entitat (`http://www.uoc.es/rdf/alumnes/Joan`) mitjançant la propietat `http://www.uoc.es/rdf/propietats/tutor`. Finalment, l'entitat `Joan` té com a `nom Joan`.

Tot i que pugui semblar estrany que les propietats siguin recursos amb una URI associada, això fa que s'eliminin les ambigüitats pròpies del llenguatge natural. Per exemple, el concepte de *tutor* no és el mateix en el context de la UOC (tal i com es fa servir en la Figura 2) que en un context legal. Les URIs permeten evitar les confusions, tal com s'ha discutit abans.

La sintaxi dels espais de noms del llenguatge XML (veure 2.1.2), també pot ser aplicada en RDF per abreujar les URIs en les sentències. Per exemple, en la Figura 1 anterior es pot substituir l'URI `hhttp://www.uoc.edu/rdf/propietats/` per un prefix d'espai de noms `prop`. Això dóna com a resultat sentències més simples (ex. `prop:nom`). Normalment, s'acostuma a fer servir el prefix `rdf` per referir-se al vocabulari definit en l'especificació de la sintaxi i el model del llenguatge RDF. Aquest prefix es correspon a l'espai de noms `http://www.w3.org/1999/02/22-rdf-syntax-ns#`.

En RDF és possible assignar cada recurs a un tipus concret. Per exemple, en els exemples anteriors, podria ser útil indicar que el recurs *Pere* és de tipus *Alumne*. De la mateixa manera, el tipus *Alumne* també podria ser un tipus de *Persona*, per exemple.

L'especificació d'RDF inclou alguns tipus predefinits i propietats per representar grups de recursos. Aquests grups reben el nom de **contenidors**, i els seus components s'anomenen **membres**. L'RDF defineix tres tipus de contenidor:

- *Seqüència.* Representa un conjunt de recursos o literals, on l'ordenació d'aquests membres és important. Per exemple, una *seqüència* es pot fer servir per emmagatzemar un grup de recursos en ordre alfabètic.
- *Sac.* Representa un conjunt de recursos o literals, on l'ordenació d'aquests membres no és important. Per exemple, un *sac* es pot fer servir per representar un conjunt de peces on l'ordre d'introducció o processament no és determinant.

- *Alternativa.* Representa un grup de recursos o literals que són alternatius, és a dir, només un d'ells és vàlid en un moment donat. Per exemple, es pot utilitzar aquest tipus de grup per representar una llista de pàgines d'Internet des d'on descarregar un mateix programari.

3.1.3 La sintaxi RDF: RDF/XML

Un cop definit el model de dades RDF, sorgeix la necessitat d'emmagatzemar les sentències i transferir-les en algun format determinat. Existeixen diferents sintaxis que permeten realitzar aquesta tasca. La sintaxi més coneguda i més utilitzada és l'RDF/XML [18], ja que permet aprofitar les nombroses eines existents pel processament de documents XML. Aquesta sintaxi no s'ha de confondre amb l'RDF, que és el model de dades descrit a l'apartat anterior.

L'estructura bàsica d'un fitxer RDF/XML es mostra a l'Exemple 6:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:alumnes="http://www.uoc.edu/rdf/alumnes/">
</rdf:RDF>
```

Exemple 6 – Fitxer bàsic RDF/XML

L'etiqueta `<rdf:RDF>` és l'arrel del fitxer i, com a tal, encapsula la resta d'informació. Dins d'aquesta etiqueta es declaren els espais de noms que es faran servir. Per defecte, es declara el vocabulari definit en l'especificació d'RDF amb el prefix `rdf`, i, en aquest exemple, també s'ha afegit un altre espai de noms, anomenat `alumnes`, que conté un vocabulari propi.

Les sentències RDF es representen mitjançant les etiquetes `<rdf:Description>`. Els diferents components d'una sentència RDF es representen de la manera següent:

- *Subjecte.* El subjecte de la sentència es representa sempre amb l'atribut `rdf:about` de l'etiqueta `<rdf:Description>`. L'Exemple 7, mostra una sentència RDF on el subjecte està identificat per l'URI `http://www.uoc.edu/rdf/alumnes/Pere`.

```
    <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere">
       ...
    </rdf:Description>
```

Exemple 7 – Representació del subjecte d'una sentència RDF

- *Propietat.* La propietat d'una sentència es pot representar mitjançant un atribut de l'etiqueta `<rdf:Description>` (veure Exemple 8), o bé, com una nova etiqueta que sigui filla de l'anterior (veure Exemple 9). En tots dos exemples, es representa la mateixa propietat: `http://www.uoc.edu/rdf/propietats/nom`.

- *Objecte*. La representació de l'objecte de la sentència depèn, en primer lloc, de com s'hagi representat la propietat: com a atribut (veure Exemple 8), o com a etiqueta dedicada (veure Exemple 9). En tots dos casos, s'indica que la propietat `propietats:nom` té el valor `Pere`.

```
    <?xml version="1.0"?>
    <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
             xmlns:propietats="http://www.uoc.edu/rdf/propietats/">

       <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere"
                        propietats:nom="Pere"/>

    </rdf:RDF>
```

Exemple 8 – Representació d'una propietat com a atribut i d'un objecte de tipus literal

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:propietats="http://www.uoc.edu/rdf/propietats/">

   <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere">
      <propietats:nom>Pere</propietats:nom>
   </rdf:Description>

</rdf:RDF>
```

Exemple 9 – Representació d'una propietat com a etiqueta i d'un objecte de tipus literal

Tant en l'Exemple 8 com en l'Exemple 9, l'objecte és un literal. Cal esmentar, a més, que en tots dos exemples es representa la sentència RDF que es mostra a la Figura 1. Quan l'objecte és un recurs en lloc de ser un literal, s'ha d'utilitzar una nova clàusula d'RDF anomenada `rdf:resource`.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:propietats="http://www.uoc.edu/rdf/propietats/">

   <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere"
                    propietats:nom="Pere"
                    propietats:edat="28">
      <propietats:tutor rdf:resource="http://www.uoc.edu/rdf/tutors/Joan">
   </rdf:Description>

   <rdf:Description rdf:about="http://www.uoc.edu/rdf/tutors/Joan"
                    propietats:nom="Joan"/>

</rdf:RDF>
```

Exemple 10 - Representació d'un objecte de tipus recurs

L'Exemple 10 mostra la representació de la sentència RDF de la Figura 2. La propietat `propietats:tutor` té com a valor un altre recurs identificat per la URI `http://www.uoc.edu/rdf/tutors/Joan`. La segona etiqueta `rdf:Description` representa la sentència corresponent al subjecte `Joan`, i estableix que la propietat `propietats:nom` té com a valor `Joan`.

Cal destacar que en RDF/XML no importa l'ordre de definició de les sentències en el fitxer, així que la segona etiqueta `rdf:Description` de l'Exemple 10 podria estar abans que la primera i el fitxer seguiria sent igualment vàlid. Una forma més compacta del mateix fitxer es mostra a l'Exemple 11.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:propietats="http://www.uoc.edu/rdf/propietats/">

   <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere"
                    propietats:nom="Pere"
                    propietats:edat="28">
      <propietats:tutor>
         <rdf:Description rdf:about="http://www.uoc.edu/rdf/tutors/Joan"
                          propietats:nom="Joan"/>
      </propietats:tutor>
   </rdf:Description>

</rdf:RDF>
```

Exemple 11 - Representació compacta de la sentència de la Figura 2

Per representar l'assignació de tipus en RDF/XML existeix una etiqueta específica `<rdf:type>`. L'Exemple 12 mostra com expressar que el recurs `Pere` és de tipus `Alumne`:

```
    <rdf:Description rdf:about="http://www.uoc.edu/rdf/alumnes/Pere">
       <rdf:type resource="http://www.uoc.edu/rdf/propietats/Alumne"/>
    </rdf:Description>
```

Exemple 12 – Assignació de tipus en RDF/XML

Una altra forma equivalent, però més compacta, es presenta a l'Exemple 13:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:propietats="http://www.uoc.edu/rdf/propietats/">

   <propietats:Alumne rdf:about=" http://www.uoc.edu/rdf/alumnes/Pere"/>

</rdf:RDF>
```

Exemple 13 – Assignació de tipus compacta

Com en l'Exemple 12, es fa servir també l'atribut `rdf:about`, però, en aquest cas, en lloc de fer servir l'etiqueta `rdf:Description`, es fa servir el tipus mateix, és a dir, `propietats:Alumne`.

3.2 RDF Schema

RDF defineix un model de dades simple per descriure propietats sobre els recursos i les relacions entre ells. No obstant, aquest model de dades no ofereix facilitats per definir tipus de dades específics ni classes de recursos. Tampoc no és possible definir quines propietats apliquen a una determinada classe de recursos. Per a determinades comunitats d'usuaris aquestes característiques són molt importants per tal de mantenir la coherència en les seves definicions de recursos. Per fer front a aquestes limitacions, el W3C ha definit l'**RDF Schema** [19].

L'RDF-Schema no defineix un vocabulari específic per a descriure tot tipus de recursos. El seu objectiu és especificar els mecanismes necessaris per definir vocabularis de recursos, classes i restriccions en la utilització de classes i relacions entre recursos.

El sistema de tipus d'RDF-Schema és similar al sistema de tipus dels llenguatges orientats a objectes. Per exemple, permet que els recursos es puguin definir com a instàncies d'una o més classes. A més, permet que aquestes classes es puguin organitzar de forma jeràrquica (veure Figura 3). La principal diferència respecte els llenguatges orientats a objectes és que en lloc de definir les classes en termes de les propietats que les seves instàncies tindran, un esquema RDF defineix les propietats en termes de les classes de recursos a les quals s'aplica.

Les primitives bàsiques per descriure esquemes en RDF-Schema són:

- definicions de **classe** (`rdfs:Class`) i **subclasse** (`rdfs:subClassOf`), la combinació d'ambdues permet la definició de jerarquia de classes.

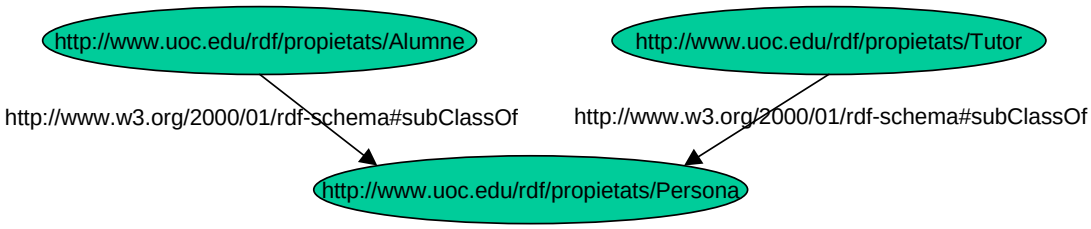

Figura 3 - Exemple bàsic de jerarquia de classes de recursos

La Figura 3 mostra una petita jerarquia de classes on la classe principal està identificada per l'URI `http:www.uoc.edu/rdf/propietats/Persona` i representa el concepte de *persona*. Els conceptes de *tutor* i d'*alumne* es poden considerar especialitzacions o subclasses de *persona*. Això queda indicat a la Figura 3 mitjançant la primitiva `http://www.w3.org/2000/01/rdf-schema#subClassOf`. La representació en RDF/XML d'aquesta figura es mostra a l'Exemple 14.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">

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

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

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

</rdf:RDF>
```

Exemple 14 – Codificació RDF/XML de la Figura 3

A l'Exemple 14, cadascuna de les classes es representa amb la clàusula `<rdf:Description rdf:ID="xxx">`, on `xxx` representa el nom de la classe (`Persona`, `Alumne` o `Tutor`). Tot seguit s'especifica el tipus del recurs mitjançant l'etiqueta `rdf:type`. En aquest cas, es tracta sempre de classes i, per tant, corresponen al tipus `http://www.w3.org/2000/01/rdf-schema#Class`. Finalment s'indica de quina classe superior hereten mitjançant l'ús de l'etiqueta `rdfs:subClassOf`. En el cas de `Tutor` i `Alumne`, són subclasses de `Persona` que, al mateix temps, és subclasse de la classe genèrica `http://www.w3.org/2000/01/rdf-schema#Resource`.

- definició de **propietats** (`rdf:Property`) i **sub-propietats** (`rdfs:subPropertyOf`) , la combinació d'ambdues permet la definició de jerarquia de propietats de forma similar a com s'ha vist per a les classes.
- clàusules de **domini** (`rdfs:domain`) i **rang** (`rdfs:range`), que estableixen les combinacions possibles entre propietats i classes.
- declaracions de **tipus** (`rdf:Type`), que permet declarar un recurs com a instància d'una determinada classe (veure Exemple 14).

Amb aquestes primitives es poden escriure esquemes per a un domini específic.

3.3 Aplicacions d'RDF i RDF-Schema

Existeixen múltiples aplicacions i projectes que fan servir els llenguatges RDF i RDF-Schema. A [20] i [21] es poden trobar llistes molt completes de projectes, d'on es poden destacar els següents:

- **Progos** [22], que proporciona un contenidor en Java per a qualsevol tipus de dades amb facilitats per manipular un espai estructurat semànticament.

- **Sesame RDF/RDFS** [23], una aplicació Java per a RDF i RDF-Schema que proporciona mecanismes de consulta a dades o a esquemes via diferents llenguatges de consulta. També inclou un navegador web amb capacitat per accedir a fitxers RDF/XML.

- **RDF Gateway** [24], una plataforma per a web semàntica per a *Windows* que permet tractar la informació mitjançant el llenguatge RDF. Incorpora un servidor d'aplicacions, un servidor web i una base de dades (amb un llenguatge de consulta basat en RDF).

Existeixen també múltiples editors per crear i examinar fitxers RDF/XML, així com llibreries per a diferents llenguatges de programació que permeten processar les informacions d'aquests fitxers. Una bona mostra d'aquestes utilitats es poden consultar a [21].

3.4 Conclusions

XML i RDF són diferents formalismes amb les seves pròpies funcionalitats i, per tant, cadascun d'ells tindrà un rol específic en la creació de la Web Semàntica (veure 6.1). XML va ser dissenyat per proporcionar una sintaxi fàcil per a processar i tractar les dades de la web. Amb XML es pot codificar qualsevol tipus d'informació que es vulgui intercanviar entre diferents màquines. A més, l'ús de l'XML Schema permet garantir la preservació de l'estructura de les dades. Això fa que XML sigui un llenguatge fonamental per a la construcció de la Web Semàntica, en el sentit que moltes tècniques faran servir XML com a sintaxi.

RDF proporciona un model estàndard per descriure fets sobre els recursos de la web i, d'aquesta manera, proporciona una certa interpretació de la informació. RDF Schema significa un pas més endavant en les capacitats d'interpretació de les dades.

Tanmateix, per a la realització de la Web Semàntica són necessàries altres tecnologies que permetin afegir més significat semàntic a les dades. En el proper capítol s'introduirà el concepte d'ontologia aplicat al camp de la informàtica. Mitjançant les ontologies es podrà estructurar la informació segons criteris semàntics i serà possible determinar de forma explícita les relacions entre els diferents conceptes d'un domini.

Capítol 4 - Ontologies

En aquest capítol es descriurà què és una ontologia des del punt de vista de la informàtica lingüística. Es donarà una visió dels components fonamentals d'una ontologia i de les seves aplicacions potencials en el camp de la informàtica (apartat 4.1). Tot seguit es descriurà una de les ontologies més conegudes en àmbits lingüístics: *WordNet* (apartat 4.2). Finalment, es presentaran algunes conclusions sobre els conceptes explicats al llarg del capítol (apartat 4.3).

4.1 Ontologia

Tot i que el concepte d'ontologia és força emprat des de fa molt de temps en filosofia, en els darrers anys s'ha introduït en el camp de la informàtica com a vocabulari comprensible per a les màquines. Aquest vocabulari s'especifica amb un nivell de precisió tal que permet que termes diferents estiguin relacionats de forma precisa i inequívoca. La definició operacional d'ontologia, segons [25] i [26], és la següent: "Una ontologia defineix els termes que s'utilitzen per descriure i representar un àrea de coneixement".

Les ontologies són utilitzades per la gent, les bases de dades i les aplicacions que necessiten compartir informació sobre un domini determinat. En aquest context, el terme de domini fa referència a un àrea específica de coneixement, com ara la medicina, gestió de finances, etc.

Les ontologies incorporen definicions de conceptes bàsics del domini al qual s'apliquen i les relacions entre aquests termes. Aquesta informació pot ser entesa pels ordinadors. Parlant en termes específics, una ontologia està formada pels següents elements principals:

- una descripció explícita i formal de **conceptes** (també anomenats **classes**) del domini,
- les **propietats** de cadascun d'aquests conceptes que permeten descriure les seves característiques i atributs,
- les **restriccions** de les propietats.

Les classes constitueixen la part principal de les ontologies, ja que descriuen els conceptes del domini. Una classe pot tenir subclasses que representin conceptes més específics que el representat per mitjà de la superclasse. D'aquesta manera,

s'estableix una jerarquia de classes basada en l'especialització de la definició de conceptes. A aquesta jerarquia també se l'anomena **taxonomia**.

Les ontologies tenen un enorme potencial pel desenvolupament d'aplicacions *software* molt més eficients, adaptatives i intel·ligents que les actuals, ja que permeten:

- Compartir el coneixement de l'estructura de la informació entre les persones i les màquines. Com a escenari pràctic es pot considerar l'existència de diferents webs amb dades mèdiques. Si totes aquestes webs utilitzen la mateixa ontologia per classificar i estructurar la informació, les aplicacions automàtiques, és a dir, els agents, podrien realitzar, per exemple, cerques complexes utilitzant les dades de totes aquestes webs, relacionant la informació de les diferents fonts.

- Separació entre el coneixement del domini i el coneixement operacional. Es podria, per exemple, descriure una tasca per configurar un producte a partir dels seus components. D'altra banda, es podria implementar un programa que realitzés aquesta configuració independentment del producte final i dels components. D'aquesta manera es podria crear una ontologia dels components d'un ordinador (coneixement del domini), i després aplicar l'algoritme (coneixement operacional) per a configurar ordinadors.

- Fer específiques les assumpcions del domini, és a dir, especificar amb precisió les relacions que els humans consideren obvies.

- Reutilització del coneixement del domini. Per exemple, hi ha models de diferents dominis que necessiten representar el concepte del temps. Aquesta representació inclou les nocions d'intervals de temps, dates concretes, mesures relatives, etc. Si una comunitat desenvolupa una ontologia que reculli totes aquestes nocions, altres grups poden utilitzar-la pels seus propòsits.

- Anàlisi del coneixement del domini. Fins que no s'ha creat una especificació precisa de tots els termes d'un domini i les seves relacions, és a dir, una ontologia, no es pot valorar quin és el grau de coneixement que es té del domini concret.

A la web existeixen nombroses ontologies que poden ser reutilitzades pels usuaris. En el subapartat 4.2 es descriurà una ontologia particularment interessant pel propòsit del present treball: *WordNet*. El cas pràctic que es desenvoluparà al capítol 7 segueix la mateixa estructura que aquesta ontologia.

4.2 WordNet

WordNet [27] és una base de dades lèxico-conceptual de l'idioma anglès que està estructurada com una xarxa semàntica, és a dir, formada per unitats lèxiques i relacions entre elles. El desenvolupament de *WordNet* va començar a la dècada dels 80 sota la direcció del psicolingüista George Miller a la Universitat de Princeton. La darrera versió és *WordNet* 2.1, publicada el març del 2005 [28].

A *WordNet* s'hi poden trobar noms, adjectius, verbs i adverbis. És el que s'anomenen categories obertes. Les categories tancades, com ara les preposicions o les conjuncions, no es representen en aquesta base de dades, ja que els autors consideren que formen part del coneixement sintàctic dels parlants, i no del coneixement lèxico-semàntic que és el que es vol representar.

La unitat bàsica a partir de la qual s'estructura *WordNet* és el conjunt de sinònims (en anglès, *synset*), que es considera representatiu d'un concepte lèxic. En *WordNet* les relacions s'estableixen entre conceptes i no pas entre paraules. La sinonímia i la taxonímia són les relacions fonamentals en *WordNet*. La sinonímia defineix els conceptes mentre que la taxonímia els ordena de forma jeràrquica. Les relacions implementades a *WordNet* es recullen a la Taula 2:

Relació	Categories aplicables	Descripció
Sinonímia	Totes	Dit del mot o l'expressió que té el mateix significat que un altre, encara que en pugui diferir en altres valors funcionals dins un enunciat concret.
Antonímia	Totes	Dit del mot de sentit directament oposat a un altre.
Hiperonímia	Noms	Dit del mot amb un camp significatiu que n'inclou un altre de menor extensió: *animal* respecte a *ocell*, *recipient* respecte a *pot*.
Hiponímia	Noms	Dit del mot amb un camp significatiu inclòs en un altre de major extensió: *rossinyol* respecte a *ocell*, *cadira* respecte a *moble*.
Meronímia	Noms	Relació semàntica entre una unitat lèxica que descriu una *part*, i la que descriu el *tot* corresponent a aquesta *part*: *braç* respecte a *cos humà*.
Implicació	Verbs	Relació formal consistent en el fet que una idea o proposició n'implica una altra.
Similitud	Adjectius	Qualitat de similar, semblança.
Atribut/Valor	Noms - adjectius	

Taula 2 – Relacions implementades a *WordNet*

4.2.1 Els noms

Els nodes superiors de l'estructura taxonòmica nominal constitueixen un conjunt de conceptes amb els quals qualsevol entitat del model de coneixement lèxic està relacionada [29]: *{acte, acció, activitat}*, *{animal, fauna}*, *{atribut, propietat}*, *{cos}*, *{coneixement}*, *{event, succés}*, *{sentiment, emoció}*, *{menjar}*, *{grup, col·lecció}*, *{lloc}*, *{motiu}*, *{artefacte}*, *{comunicació}*, *{objecte natural}*, *{fenomen natural}*, *{persona, ésser humà}*, *{planta, flora}*, *{possessió}*, *{procés}*, *{quantitat}*, *{relació}*, *{forma}*, *{estat, condició}* i *{substància}*.

A la Figura 4 es representa un fragment del lexicó relatiu als vehicles de motor.

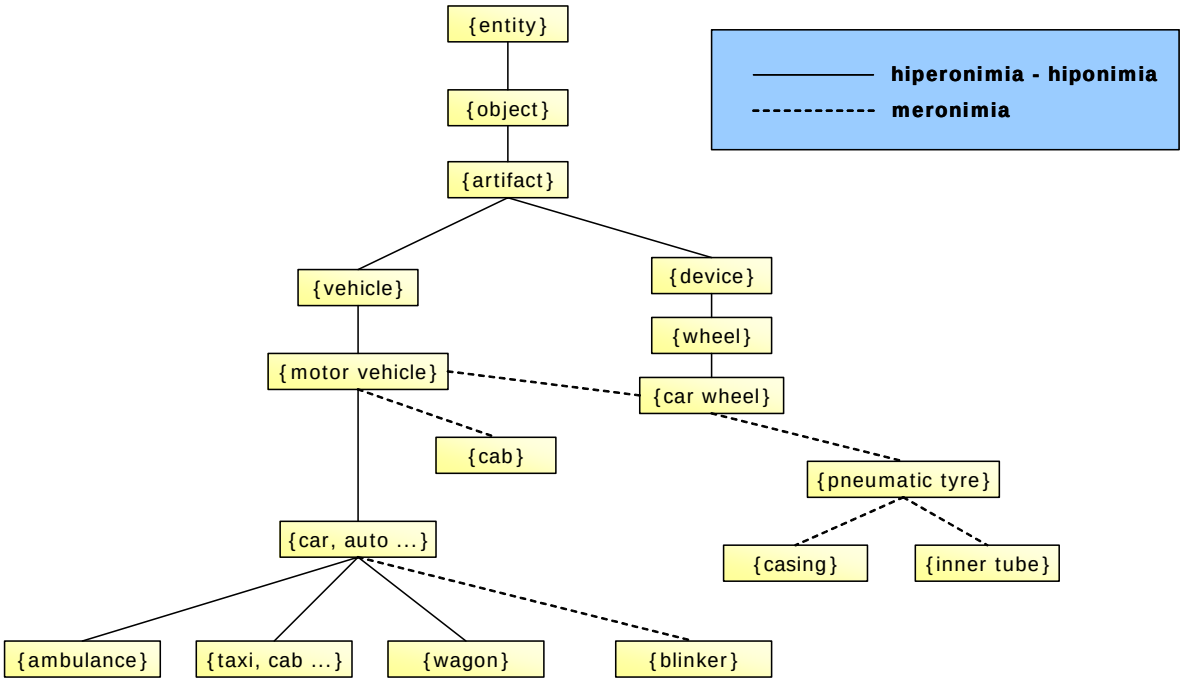

Figura 4 – Fragment de lexicó corresponent als vehicles de motor

S'observa com *car* (cotxe) està relacionat per sinonímia amb *auto* (automòbil), i taxonòmicament per hiponímia successiva amb *motor vehicle* (vehicle a motor), *vehicle* (vehicle), *artifact* (artefacte), *object* (objecte) i amb el concepte d'ordre màxim superior *entity* (entitat). Al mateix temps, *car* és hiperònim d'*ambulance* (ambulancia), *taxi* (taxi) i *wagon* (furgo). També es pot veure que *car* té la part pròpia *blinker* (intermitent).

En vista de la Figura 4, el lector es pot preguntar quan s'acaba la descomposició d'un concepte. En teoria, seria possible arribar fins al nivell de l'àtom, tot i que, des d'un punt de vista del coneixement lèxic, això no serviria de gaire. Com a norma general, la descomposició d'un objecte s'acaba quan les parts no serveixen per distingir l'objecte compost.

4.2.2 Els verbs

Els verbs són probablement la categoria sintàctica més important d'un llenguatge, ja que relacionen la resta dels elements de la frase. En *WordNet*, els verbs s'organitzen en 15 categories segons un criteri semàntic: *fisiologia, canvi, coneixement, comunicació, competició, consum, contacte, creació, emoció, moviment, percepció, possessió, relacions socials, meteorologia* i *estats*.

Les relacions que estructuren la xarxa verbal de *WordNet* són tres [30]:

- troponímia (o hiponímia verbal),
- implicació,
- causació.

Les tres són, de fet, subtipus d'implicació. La Figura 5 representa de forma esquemàtica aquestes relacions.

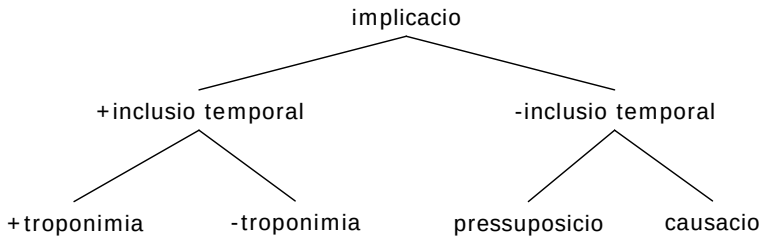

Figura 5 – Relacions verbals en *WordNet*

Es poden distingir dos tipus d'implicacions:

- Quan es pressuposa inclusió temporal entre events (en l'esquema: **+inclusió temporal**). Es divideix en:
  - **Troponímica**. Un event *x* serà tropònim de *y* quan *y* sigui un event més general que *x*, de forma que *x* s'entén com una manera particular de realitzar *y*. Per exemple, *donar* és una manera de *transferir*, essent l'event *transferir* més general que *donar*.
  - **No troponímica**. S'estableix entre dues accions o processos quan, com a mínim, durant el temps en què un d'ells s'executa, l'altre també ho fa, però no hi ha una relació de forma entre ells. Per exemple, si es produeix l'acció de *roncar*, necessàriament té lloc durant el temps en què es produeix l'acció de *dormir*.

- Quan no es pressuposa inclusió temporal entre events (en l'esquema: **-inclusió temporal**). Es divideix en:
  - **Pressuposició**. L'existència d'un event implica necessàriament l'existència anterior d'un event de tipus diferent. Per exemple, si algú es divorcia és perquè prèviament s'havia casat.
  - **Causació**. La realització d'una acció o procés implica necessàriament l'existència d'un altre event. Per exemple, si algú dóna alguna cosa, el resultat és necessàriament que algú té alguna cosa per donar.

4.2.3 Estructura de *WordNet*

El desenvolupament de *WordNet* es divideix en dues tasques principals [31]:

- Escriptura dels fitxers que contenen tota la informació lèxica
- Creació de programari capaç d'interpretar aquests fitxers i oferir serveis als usuaris

Els fitxers lexicogràfics constitueixen la base de *WordNet*. Estan escrits per lexicògrafs i recullen tota la informació lèxica. Contenen dos elements bàsics: els significats (en anglès, *synset*), i les paraules (en anglès, *word forms*). Els adverbis es troben en un únic fitxer. Els verbs i els noms estan organitzats en diferents fitxers segons criteris semàntics. Finalment, els adjectius es troben en dos fitxers: descriptius i relacionals.

Per compilar tots els fitxers lexicogràfics i generar la base de dades lèxica, es fa servir l'aplicació *grind*. Aquest eina permet, a més, identificar els errors sintàctics i estructurals i calcula l'índex polisèmic de cada paraula. Aquesta aplicació no és utilitzada per la gran majoria d'usuaris que només fan servir el *WordNet* per fer cerques. Tan sols la fan servir aquells usuaris que modifiquen els fitxers de base que componen el *WordNet* per crear una nova base de dades.

La distribució estàndard de *WordNet* inclou tres tipus d'interfícies per interactuar amb la base de dades:

- A partir de la línia de comandes mitjançant la instrucció *wn*
- A partir de programari creat per l'usuari utilitzant la llibreria *wn.h*
- A partir d'una interfície gràfica (veure Figura 6)

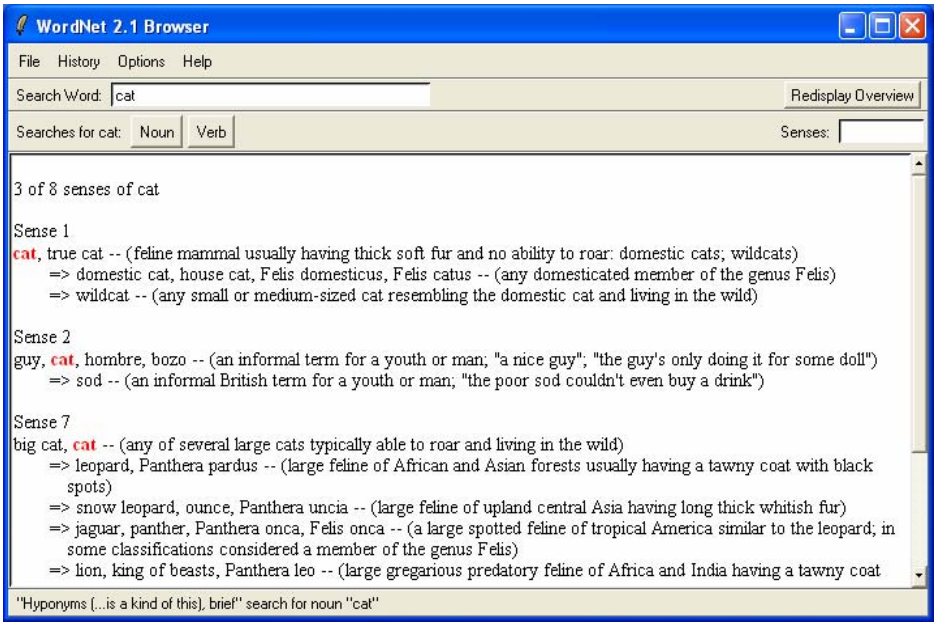

Figura 6 – Interfície gràfica per defecte del *WordNet*

Apart d'aquestes interfícies, a la web es poden trobar moltes altres. També és possible obtenir interfícies per a diferents llenguatges de programació. A [32] es recullen multitud d'enllaços que permeten descarregar extensions i interfícies per a *WordNet*.

4.2.4 Projectes relacionats

WordNet ha inspirat la creació d'altres bases de dades lèxiques en altres idiomes. Un dels projectes més importants és *EuroWordNet* [33], que segueix la mateixa estructura que *WordNet* però inclou diversos idiomes europeus com l'holandès, l'italià, el castellà, l'alemany, el francès, el txec i l'estonià. En la mateixa línia, també es troba el projecte *Global WordNet Association* [34] que pretén la definició de bases de dades lèxiques per a tots els idiomes del món.

4.3 Conclusions

L'explotació de les possibilitats que ofereixen les ontologies en el camp de la informàtica contribuirà, sens dubte, a un nou salt en el desenvolupament tecnològic. Malgrat això, fins no fa gaire temps, no ha estat gaire popular pel fet de no disposar d'estàndards de definició d'ontologies ni prou eines per facilitar el seu desenvolupament.

El panorama, però, ha canviat força en aquests últims anys gràcies a la iniciativa de la Web Semàntica (veure 6.1) liderada pel consorci W3C que ha establert uns estàndards i ha creat una sèrie de llenguatges que han estat acceptats per la major part de les comunitats d'usuaris i per les grans empreses de *software.*

En els capítols 2 i 3 s'han presentat alguns d'aquests llenguatges (XML, XML Schema, RDF i RDF Schema). En el proper capítol es descriurà el llenguatge OWL que constitueix, fins ara, l'estàndard mes complet per a la definició d'ontologies.

Capítol 5 - OWL

En aquest capítol es descriurà en detall el llenguatge OWL. Es tracta d'una nova recomanació del W3C que té com a objectiu el desenvolupament i l'expansió de la Web Semàntica. L'OWL constitueix un llenguatge estàndard per a la creació d'ontologies. Això ha de permetre la creació d'eines de desenvolupament comunes i ha de garantir l'operativitat entre diferents ontologies dins la web. Primer de tot, es presentaran les raons que van motivar la creació de l'OWL (apartat 5.1). Després es descriurà la sintaxi del llenguatge (apartat 5.2). A continuació, es donarà una breu descripció de l'editor gratuït d'OWL *Protégé* (apartat 5.3). Finalment, a mode de conclusió, es farà una recapitulació de tots els llenguatges que s'han introduït en el treball (apartat 5.4).

5.1 Justificació

Les ontologies tenen un rol molt important en la construcció de la Web Semàntica (veure 6.1) ja que, gràcies a elles, es podrà representar la semàntica dels documents i permetre que aplicacions web o agents especialitzats utilitzin aquest coneixement per donar serveis molt més efectius i satisfactoris que actualment. Mitjançant l'ús generalitzat de les ontologies, les aplicacions futures podran esdevenir "intel·ligents", en el sentit que treballaran d'una manera semblant al nivell de conceptualització dels humans.

Les ontologies juguen un paper fonamental en aquelles aplicacions que realitzen cerques creuades i obtenen informació de diferents fonts. Tot i que els DTDs (veure 2.2.1) i el llenguatge XML Schema (veure 2.2.2) són suficients per garantir un intercanvi consistent de dades entre grups d'usuaris que han acordat prèviament el format de la informació, el fet que cap d'aquests dos llenguatges proporcioni informació semàntica, impedeix que l'intercanvi d'informació sigui fiable quan s'afegeix un nou vocabulari XML. Amb l'ús d'RDF (veure 3.1) i RDF Schema (veure 3.2), es comença a introduir un contingut semàntic, encara que molt simple, en els documents. En el cas particular del llenguatge RDF Schema es podria considerar com un llenguatge ontològic simple. Ara bé, per aconseguir l'operativitat entre nombrosos esquemes, cadascun d'ells gestionat i creat de forma independent per una comunitat d'usuaris particular, es necessiten regles semàntiques més detallades que les proporcionades per l'RDF Schema.

Per respondre a aquesta nova necessitat, el W3C ha creat el llenguatge OWL (*Ontology Web Language*) [35]. L'OWL ha estat dissenyat per ésser utilitzat per aplicacions que necessiten processar el contingut de la informació en lloc de limitar-se a presentar-la als humans. L'OWL proporciona a les aplicacions un major grau d'interpretació del contingut dels documents que altres llenguatges com XML, RDF o RDF Schema. Això s'aconsegueix oferint un vocabulari més ampli i unes regles de semàntica formals.

En particular, els objectius del disseny de l'OWL foren:

1. *Utilització d'ontologies existents*. Les ontologies han de poder fer-se públiques per tal que qualsevol comunitat d'usuaris pugui fer-les servir per a crear la seva pròpia base de dades de coneixement. A més, ha de ser possible estendre una ontologia existent per afegir-hi noves definicions.

2. *Evolució d'ontologies existents.* Les ontologies poden evolucionar al llarg del seu cicle de vida. Els usuaris han de ser capaços d'indicar quina versió d'una determinada ontologia fan servir. En conseqüència, el llenguatge OWL ha de permetre que l'autor d'una ontologia pugui identificar els canvis que introdueix respecte a versions anteriors de forma explícita. Cal notar la diferència respecte el punt anterior: no és el mateix fer evolucionar una ontologia que estendre-la, ja que, en aquest últim cas, l'ontologia original no canvia.

3. *Integració d'ontologies existents.* Diferents ontologies poden representar el mateix concepte de diferents maneres. Per garantir l'operativitat entre elles, el llenguatge ha de proporcionar mecanismes que permetin relacionar les diferents representacions.

4. *Detecció d'inconsistències.* El llenguatge ha de proporcionar un mecanisme que permeti detectar les inconsistències en les ontologies o en les bases de coneixement.

5. *Proporcionar un compromís entre expressivitat i escalabilitat.* El llenguatge ha d'incorporar un bon ventall de construccions que permetin expressar la semàntica del document, però al mateix temps, ha de proporcionar els mecanismes que permetin explotar les ontologies de forma eficient.

6. *Simplicitat d'ús*. La sintaxi no ha de suposar una barrera per a l'ús del llenguatge.

7. *Compatibilitat amb d'altres estàndards.* El llenguatge ha de ser compatible amb d'altres que s'utilitzen normalment a la Web. En particular, amb XML, XML Schema i RDF.

8. *Internacionalització.* El llenguatge ha de permetre el desenvolupament d'ontologies multilingües.

L'OWL fou presentat com a recomanació del W3C el 10 de febrer del 2004. La seva especificació completa es recull en els següents documents:

- *OWL Overview* [36], proporciona una introducció simple a l'OWL
- *OWL Guide* [37], demostra l'ús del llenguatge amb un exemple complet
- *OWL Reference* [38], proporciona una descripció sistemàtica i compacta de totes les primitives de l'OWL
- *OWL Semantics and Abstract Syntax* [39], constitueix la definició formal i final del llenguatge

- *OWL Web Ontology Language Test Cases* [40], conté una sèrie de casos de test del llenguatge
- *OWL Use Cases and Requirements* [41], compila tots els requeriments de disseny de l'OWL i descriu alguns casos d'ús il·lustratius

5.2 Sintaxi del llenguatge

5.2.1 Subllenguatges

S'han definit tres subllenguatges de l'OWL. Cadascun d'ells proporciona un major grau de complexitat i d'expressivitat. Aquests subllenguatges són, en ordre de menor a major complexitat:

- *OWL Lite*. Està indicat pels usuaris que necessitin establir classificacions jeràrquiques i restriccions simples. Essent la versió més simple de l'OWL, disposa d'un major nombre d'eines compatibles.

- *OWL DL*. Pensat pels usuaris que desitgen la màxima expressivitat quant a qüestions semàntiques i, al mateix temps, volen conservar la completesa computacional (totes les conclusions seran computables) i la decidibilitat (les operacions computacionals acabaran en un temps finit).

- *OWL Full*. Pensat pels usuaris que volen la màxima expressivitat del llenguatge però sense garanties computacionals. És l'única versió que permet estendre el vocabulari predefinit del llenguatge. El seu principal inconvenient és que, avui dia, no existeix cap programari que sigui completament compatible amb les funcionalitats de l'*OWL Full*.

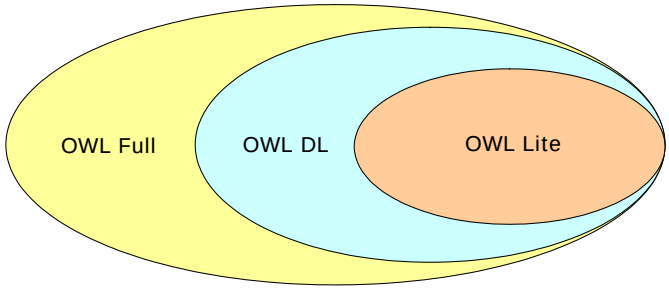

Figura 7 – Els tres subllenguatges de l'OWL

Com es veu a la Figura 7, cadascun d'aquests subllenguatges es una extensió del seu predecessor, tant en termes d'allò que es pot expressar com en termes d'allò que es pot concloure.

5.2.2 Estructura bàsica d'un document OWL

Tot document OWL, sigui del subllenguatge que sigui, és un document RDF, i tot document RDF és un document OWL Full.

El primer que es troba dins d'un document OWL és una sèrie de declaracions d'espais de noms XML (veure 2.1.2) dins d'una etiqueta `rdf:RDF`. El seu propòsit és identificar de forma inequívoca els identificadors que s'utilitzaran a la resta del document. L'Exemple 15 conté una capçalera genèrica.

```
<rdf:RDF
      xmlns ="ontology_URI"
      xmlns:ontology_prefix="ontology_URI"
      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#">
```

Exemple 15 – Capçalera genèrica d'un document OWL

Les dues primeres declaracions identifiquen l'espai de noms associat a l'ontologia. La primera d'elles declara que l'URI `ontology_URI` és l'espai de noms per defecte del document, ja que no li assigna cap prefix. La segona declaració identifica l'espai de noms de l'ontologia amb el prefix `ontology_prefix`.

La tercera declaració és la declaració convencional del vocabulari OWL. Tots els elements del document que tinguin com a prefix `owl:` pertanyen al llenguatge estàndard OWL. De manera similar, les tres declaracions restants declaren els elements dels llenguatges RDF (prefix `rdf:`), RDF Schema (prefix `rdfs:`) i XML Schema (prefix `xsd:`).

Després d'aquesta capçalera, s'acostuma a incloure una altra que conté diferents propietats de l'ontologia que es descriurà en el document. Es fa servir l'etiqueta `owl:Ontology` i les propietats que s'indiquen són comentaris, número de versió o la inclusió d'altres ontologies. L'Exemple 16 conté un petit exemple d'aquest tipus de capçalera.

```
<owl:Ontology rdf:about="">
      <owl:versionInfo>v 2.03 13/04/2004 fac </owl:versionInfo>
      <rdfs:comment>Exemple de capcelera</rdfs:comment>
      <owl:priorVersion rdf:resource=""/>
      <owl:imports rdf:resource=""/>
      <rdfs:label>Nom de l'ontologia</rdfs:label>
</owl:Ontology>
```

Exemple 16 – Exemple de capçalera d'un document OWL

A l'Exemple 16 es presenten els següents elements:

- L'atribut `rdf:about` de l'etiqueta `owl:Ontology` proporciona el nom o la referència de l'ontologia. En la majoria de casos no s'especifica cap valor. Això significa que el nom de l'ontologia és, de fet, l'URI del document que la conté.

- L'etiqueta `owl:versionInfo` és una cadena de caràcters que identifica la versió de l'ontologia. L'estàndard no imposa un format determinat per a aquesta informació. Depèn del programari que s'utilitzi per a la gestió de versions.

- L'etiqueta `rdfs:comment` serveix per incloure comentaris dins del document.

- L'etiqueta `owl:priorVersion` és un mecanisme que permet a aplicacions externes el control de versions de l'ontologia. L'atribut `rdf:resource` identifica la versió anterior de l'ontologia.

- L'etiqueta `owl:imports` proporciona el mecanisme per incloure altres ontologies. L'ontologia que s'ha d'importar s'indica mitjançant l'atribut `rdf:resource`.

- Finalment, l'etiqueta `rdfs:label` permet donar un nom a l'ontologia del document.

Després d'aquesta capçalera es troben totes les declaracions que formen l'ontologia. Per tancar el document de forma correcta, es fa servir l'etiqueta `</rdf:RDF>`.

5.2.3 Elements bàsics

❖ Classes i individus

L'avantatge més important en l'ús d'ontologies és la capacitat de raonar i extreure conclusions sobre els seus *individus*, és a dir, sobre la informació estructurada segons les regles marcades per l'ontologia. L'OWL proporciona el mecanisme de la definició de *classes* per tal d'acomplir aquesta tasca.

Els conceptes més bàsics d'un determinat domini s'han de considerar com a classes de l'ontologia. A partir d'aquestes classes primàries és possible crear arbres taxonòmics. Existeix una superclasse anomenada `owl:Thing`. Totes les classes definides per l'usuari seran sempre subclasses d'ella. També existeix una classe que representa el conjunt buit anomenada `owl:Nothing`. Aquesta classe no té mai cap instància i és sempre subclasse de totes les altres.

Per definir una classe en un document OWl es fa servir l'etiqueta `owl:Class`.

```
<owl:Class rdf:ID="NomClasse"/>
```

Exemple 17 – Definició de classe en OWL

En l'Exemple 17 es mostra la definició de la classe anomenada `NomClasse`. Es fa servir un atribut propi de l'RDF, `rdf:ID`, per identificar la nova classe. Aquesta etiqueta només permet declarar l'existència de la classe, però podria passar que no hi hagués cap membre que hi pertanyés. Per fer referència a aquesta classe a qualsevol altra part del document es fa servir l'identificador `#NomClasse`. També seria possible fer-ne referència des d'un altre document. Per fer això, caldria declarar l'espai de noms del document on s'ha definit la classe dins la capçalera del document OWL que vol fer-la servir (veure apartat 5.2.2). Dins del document, l'identificador `prefix:#NomClasse` s'utilitzaria per referir-se a la classe, on `prefix` seria el prefix que s'ha indicat a la declaració de l'espai de noms.

A partir d'una classe es poden derivar d'altres per tal de crear una taxonomia. Això és possible mitjançant l'ús de l'etiqueta `rdfs:subClassOf`. Cal notar les següents propietats:

- Si `X` és una subclasse d'`Y`, aleshores tota instància d'`X` ho és també d'`Y`
- Si `X` és una subclasse d'`Y`, i `Y` és una subclasse de `Z`, aleshores `X` és una subclasse de `Z`. És a dir, es tracta d'una relació transitiva

```
<owl:Class rdf:ID="ClasseFilla">
      <rdfs:subClassOf rdf:resource="#ClasseMare" />
</owl:Class>
```

Exemple 18 – Definició de subclasse

En l'Exemple 18 es mostra la definició de la classe `ClasseFilla` com a subclasse de `ClasseMare`. En certa manera, la relació de subclasse es pot entendre com una restricció en la definició de la nova classe.

Els individus són els membres de les classes. Per tal de declarar els individus en el document OWL es fa servir una etiqueta amb el següent format:

```
<NomClasse rdf:ID="NomIndividu" />
```

Figura 8 – Format OWL per a la definició dels individus

En la Figura 8 es declara l'individu de nom `NomIndividu` com a membre de la classe `NomClasse`.

La distinció entre classes i individus ha de quedar clara. Una classe és simplement una col·lecció de propietats que descriu un conjunt d'individus. Els individus són els membres d'aquest conjunt. Per tant, les classes representen fets generals del domini d'aplicació de l'ontologia, mentre que els individus són les entitats reals que es poden agrupar dins les classes definides.

❖ Propietats

Les classes només permeten crear classificacions jeràrquiques. Les *propietats*, però, permeten definir fets generals aplicables a tots els membres d'una classe, o bé, característiques específiques de determinats individus.

Una propietat és una relació binària. Es poden distingir dos tipus de propietats:

1. Propietats de tipus (en anglès, *datatype properties*). Són les relacions entre les instàncies de les classes i tipus de dades. Per definir-les en un document OWL es fa servir l'etiqueta `owl:DatatypeProperty`.
2. Propietats d'objecte (en anglès, *object properties*). Són les relacions entre instàncies de dues classes. Per definir-les en un document OWL es fa servir l'etiqueta `owl:ObjectProperty`.

En ambdós casos, els axiomes de propietats defineixen les característiques de les propietats.

```
<owl:ObjectProperty rdf:ID="colorCotxe"/>
```

Exemple 19 – Declaració d'existència d'una propietat

En l'Exemple 19 es representa el cas més simple d'axioma de propietat, ja que només es declara l'existència de la propietat `colorCotxe`.

Sovint, però, els axiomes de propietats són més complexos i defineixen característiques addicionals de les propietats. En concret, és possible definir les següents característiques:

- `rdfs:subPropertyOf`. Permet crear taxonomies a nivell de propietats de la mateixa forma en què s'ha discutit anteriorment per a les classes. Aquest axioma és aplicable tant per a les propietats de tipus com per a les propietats d'objecte.
- `rdfs:domain`. Aquest axioma especifica que el subjecte de la propietat a la que aplica ha de pertànyer a la classe indicada per l'axioma. Es pot donar el cas que s'especifiquin múltiples dominis. En aquest cas, el domini de la propietat resulta de la intersecció de les classes especificades com a dominis.
- `rdfs:range`. Aquest axioma especifica que el valor de la propietat a què aplica ha de pertànyer a la classe indicada per l'axioma (en el cas de les propietats d'objecte), o bé, a un determinat tipus de dades (en el cas de les propietats de tipus). Es pot donar el cas que s'especifiquin múltiples rangs. Aleshores, el rang resultant serà la intersecció de tots els rangs especificats.

```
<owl:ObjectProperty rdf:ID="descripcioCotxe">
      <rdfs:domain rdf:resource="#Cotxe"/>
      <rdfs:range rdf:resource="#Descripcio">
</owl:ObjectProperty>
```

```
<owl:ObjectProperty rdf:ID="colorCotxe">
      <rdfs:subPropertyOf rdf:resource="#descripcioCotxe"/>
      <rdsf:range   rdf:resource="#Color"/>
</owl:ObjectProperty>
```

Exemple 20 – Exemple de taxonomia simple amb dominis i rangs

En l'Exemple 20 es declara la propietat descripcioCotxe que relaciona les instàncies de les classes Cotxe (el domini) i Descripcio (el rang). A continuació, es declara la propietat colorCotxe com a subpropietat de descripcioCotxe. Com la seva definició no s'especifica un domini, hereta el domini de la propietat pare, és a dir, les instàncies de Cotxe. El rang és Color perquè així està especificat.

- owl:equivalentProperty. S'utilitza per indicar que dues propietats tenen la mateixa extensió, en termes de domini i rang.
- owl:inverseOf. Les propietats tenen una direcció concreta: del domini al rang. A vegades, però, pot ser útil definir la relació en ambdues direccions. En el cas general, l'axioma propietat1 owl:inverseOf propietat2, indica que per a tot parell (x, y) que pertanyi a l'extensió de la propietat1, existeix un parell (y, x) que pertany a l'extensió de la propietat2.
- owl:FunctionalProperty. Una propietat funcional és aquella que només pot tenir un únic valor y per a cada instància de x. Tant les propietats d'objecte com les de tipus poden ser declarades com a funcionals.

```
<owl:ObjectProperty rdf:ID="marit">
      <rdfs:domain rdf:resource="#Dona"/>
      <rdfs:range rdf:resource="#Home"/>
</owl:ObjectProperty>

<owl:FunctionalProperty rdf:about="#marit"/>
```

Exemple 21 – Exemple de propietat funcional

En l'Exemple 21 es declara la propietat funcional marit: tota instància de la classe Dona té com a marit una sola instància de la classe Home, com a màxim.

- owl:InverseFunctionalProperty. És la propietat inversa a l'anterior. És a dir, si es declara la propietat1 com a owl:InverseFuncionalProperty, no poden existir dues instàncies diferents x1 i x2 tals que (x1, y) i (x2, y) pertanyin a la propietat1.
- owl:TransitiveProperty. Quan es declara una propietat P com a transitiva, significa que si el parell (x, y) és una instància de P i el parell (y, z) també ho és, aleshores el parell (x, z) és també una instància de P.
- owl:SymmetricProperty. Una propietat simètrica és aquella que compleix que si el parell (x, y) és una instància de P, aleshores el parell (y, x) també ho és.

```
<owl:SymmetricProperty rdf:ID="amicDe">
      <rdfs:domain rdf:resource="#Persona"/>
      <rdfs:domain rdf:resource="#Persona"/>
</owl:SymmetryProperty>
```

Exemple 22 – Exemple de propietat simètrica

L'Exemple 22 mostra un exemple típic de propietat simètrica: l'amistat. Si x és amic de y, aleshores, y és amic de x (o almenys, teòricament).

En el cas particular de les propietats de tipus (les que relacionen instàncies de classes amb tipus de dades), es poden fer servir els literals de l'RDF, o bé, els tipus de dades simples definits en l'XML Schema. Els tipus recomanats en la definició de l'OWL estan especificats a [37].

```
<owl:DatatypeProperty rdf:ID="anyFabricacioCotxe">
      <rdfs:domain rdf:resource="#anyFabricacio"/>
      <rdfs:range rdf:resource="&xsd;positiveInteger"/>
</owl:DatatypeProperty>
```

Exemple 23 – Exemple de propietat de tipus

En l'Exemple 23 es declara la propietat de tipus `anyFabricacioCotxe`, que relaciona les instàncies de la classe `anyFabricacio` amb el tipus de dades `positiveInteger` de l'XML Schema.

❖ Restriccions de les propietats

Per a les propietats definides anteriorment, es poden aplicar restriccions per adaptar-les millor a les necessitats del domini d'estudi. Dins del document OWL, les restriccions s'han de declarar dins el context de l'etiqueta `owl:Restriction`. A més, l'etiqueta `owl:onProperty` serveix per identificar la propietat a la qual s'aplica la restricció.

Les restriccions definides pel llenguatge són:

- `owl:allValuesFrom`. Indica que tota instància de la propietat tingui com a valor un membre de la classe especificada per la clàusula `owl:allValuesFrom`, únicament.

```
<owl:Class rdf:ID="Cotxe">
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#fabricant"/>
         <owl:allValuesFrom rdf:resource="#FabricantCotxes"/>
      </owl:Restriction>
   </rdfs:subClassOf)
</owl:Class>
```

Exemple 24 – Ús de la restricció `owl:allValuesFrom`

En l'Exemple 24 s'especifica que quan la propietat `fabricant` s'aplica als membres de la classe `Cotxe`, el valor de la relació ha de ser necessàriament una instància de la classe `FabricantCotxes`.

- `owl:someValuesFrom`. És molt similar a l'anterior, però amb la diferència que, en aquest cas, ha d'existir com a mínim una instància de la propietat que tingui com a valor un membre de la classe indicada per la clàusula `owl:someValuesFrom`.
- `owl:cardinality`. Permet especificar el nombre exacte d'elements de la relació.

```
<owl:Class rdf:ID="Cotxe">
   <rdfs:subClassOf>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#fabricant"/>
         <owl:cardinality rdf:datatype="&xsd;nonNegativeInteger">
            1
         </owl:cardinality>
```

```
            </owl:Restriction>
      </rdfs:subClassOf)
   </owl:Class>
```

Exemple 25 - Ús de la restricció `owl:cardinality`

En l'Exemple 25 s'especifica que un `Cotxe` només pot tenir un únic `fabricant`.

- `owl:maxCardinality`. Indica el nombre màxim d'instàncies per a la propietat.
- `owl:minCardinality`. Indica el nombre mínim d'instàncies per a la propietat.
- `owl:hasValue`. Permet especificar classes basades en l'existència d'un valor de propietat concret. Per tant, un individu serà membre de la classe si almenys una de les seves propietats és igual a la indicada a la clàusula `owl:hasValue`.

```
<owl:Class rdf:ID="Cotxe">
   <rdfs:subClassOf>
         <owl:Restriction>
               <owl:onProperty rdf:resource="#fabricant"/>
               <owl:hasValue rdf:resource="#FabricantCotxes"/>
         </owl:Restriction>
   </rdfs:subClassOf)
</owl:Class>
```

Exemple 26 – Ús de la restricció `owl:hasValue`

En l'Exemple 26 s'especifica que tota instància de la classe `Cotxe` ha de tenir una propietat `fabricant` que tingui com a valor un membre de la classe `FabricantCotxes`.

5.2.4 Elements complexos

❖ Classes conjunt

L'OWL inclou algunes construccions que permet crear noves classes utilitzant operacions de conjunts que actuen de manera anàloga que en termes matemàtics.

- `owl:intersectionOf`. Els membres de la nova classe seran aquells que pertanyin a la intersecció entre les classes especificades, és a dir, els que pertanyen a totes les classes.

```
         <owl:Class rdf:about="#PeçaMotorGenerica">
               <owl:intersectionOf rdf:parseType="Collection">
                     <owl:Class rdf:about="#PeçaMotorCotxe"/>
                     <owl:Class rdf:about="#PeçaMotorMoto"/>
                     <owl:Class rdf:about="#PeçaMotorAvio"/>
               </ owl:intersectionOf>
         </owl:Class>
```

Exemple 27 – Ús de la construcció `owl:intersectionOf`

En l'Exemple 27, els membres de la classe `PeçaMotorGenerica` seran aquells que siguin membres de `PeçaMotorCotxe` i de `PeçaMotorMoto` i de `PeçaMotorAvio`.

- `owl:unionOf`. Els membres de la nova classe seran tots aquells que pertanyin a les classes especificades. S'utilitza de la mateixa forma que en el cas anterior.

- `owl:complementOf`. Els membres de la nova classe són tots aquells que no pertanyen a la classe especificada, però que sí pertanyen al domini d'aquesta.

❖ Classes enumerades

L'OWL permet la declaració d'una classe a partir de l'enumeració dels seus membres mitjançant l'ús de la construcció `owl:oneOf`. D'aquesta manera, s'especifica completament l'extensió de la classe de forma que no es poden declarar més individus que pertanyin a la classe.

```
<owl:Class rdf:ID="ColorCotxe">
      <owl:oneOf rdf:parseType="Collection">
            <owl:Thing rdf:about="#Blanc"/>
            <owl:Thing rdf:about="#Vermell"/>
            <owl:Thing rdf:about="#Blau"/>
      </owl:oneOf>
</owl:Class>
```

Exemple 28 – Ús de la construcció `owl:oneOf`

En l'Exemple 28 es defineix la classe `ColorCotxe` que tindrà com a membres els individus de les classes `Blanc`, `Vermell` i `Blau`.

❖ Classes disjuntes

La disjunció en un conjunt de classes s'expressa en l'OWL amb la construcció `owl:disjointWith`. Amb això es garanteix que un individu que és membre d'una classe no pot ser instància simultàniament de cap de les altres classes especificades.

```
<owl:Class rdf:ID="Pasta">
      <owl:disjointWith rdf:resource="#Carn"/>
      <owl:disjointWith rdf:resource="#Marisc"/>
      <owl:disjointWith rdf:resource="#Postres"/>
      <owl:disjointWith rdf:resource="#Fruita"/>
</owl:Class>
```

Exemple 29 – Ús de la construcció `owl:disjointWith`

En l'Exemple 29 es declara que la classe `Pasta` és disjunta de totes les altres. Però no es declara que, per exemple, les classes `Marisc` i `Fruita` siguin disjuntes. Per fer això, caldria una nova declaració que fes explícita aquesta disjunció.

5.2.5 Diferències entre els subllenguatges

La varietat de construccions sintàctiques que es poden fer servir en cadascun dels subllenguatges de l'OWL depèn directament del seu grau de complexitat. La Taula 3 recull les diferències entre els tres subllenguatges.

Construcció	OWL Lite	OWL DL	OWL Full	Construcció	OWL Lite	OWL DL	OWL Full
Class	✓	✓	✓	*intersectionOf*	✓	✓	✓
subClassOf	✓	✓	✓	*xsd datatypes*	✓	✓	✓
Property	✓	✓	✓	*versionInfo*	✓	✓	✓
subPropertyOf	✓	✓	✓	*priorVersion*	✓	✓	✓
domain	✓	✓	✓	*backwardCompatibleWith*	✓	✓	✓
range	✓	✓	✓	*incompatibleWith*	✓	✓	✓
Individual	✓	✓	✓	*DeprecatedClass*	✓	✓	✓
equivalentClass	✓	✓	✓	*DeprecatedProperty*	✓	✓	✓
equivalentProperty	✓	✓	✓	*label*	✓	✓	✓
sameAs	✓	✓	✓	*comment*	✓	✓	✓
differentFrom	✓	✓	✓	*seeAlso*	✓	✓	✓
AllDifferent	✓	✓	✓	*isDefinedBy*	✓	✓	✓
distinctMembers	✓	✓	✓	*AnnotationProperty*	✓	✓	✓
ObjectProperty	✓	✓	✓	*OntologyProperty*	✓	✓	✓
DatatypeProperty	✓	✓	✓	*oneOf*	×	✓	✓
inverseOf	✓	✓	✓	*dataRange*	×	✓	✓
TransitiveProperty	✓	✓	✓	*disjointWith*	×	✓	✓
SymmetricProperty	✓	✓	✓	*equivalentClass*	×[1]	✓	✓
FunctionalProperty	✓	✓	✓	*subClassOf*	×[1]	✓	✓
InverseFunctionalProperty	✓	✓	✓	*unionOf*	×	✓	✓
Restriction	✓	✓	✓	*complementOf*	×	✓	✓
onProperty	✓	✓	✓	*intersectionOf*	×	✓	✓
allValuesFrom	✓	✓	✓	*hasValue*	×	✓	✓
someValuesFrom	✓	✓	✓	*minCardinality*	✓[2]	✓	✓
Ontology	✓	✓	✓	*maxCardinality*	✓[2]	✓	✓
imports	✓	✓	✓	*cardinality*	✓[2]	✓	✓

Taula 3 – Diferències sintàctiques entre els subllenguatges de l'OWL

A la Taula 3, la columna *Construcció* conté les diferents clàusules del llenguatge OWL. Per a cadascuna d'elles, les columnes *OWL Lite, OWL DL* i *OWL Full* indiquen si és possible utilitzar aquesta instrucció en el subllenguatge corresponent (amb el símbol ✓) o no (amb el símbol ×).

A [38] es descriuen en detall totes les limitacions de cadascun dels subllenguatges.

5.3 *Protégé*: un editor OWL

Protégé [42] és un entorn gràfic gratuït per a la creació i edició d'ontologies i bases de coneixement. Existeix un *plug-in* gratuït [43] que permet crear i editar ontologies en llenguatge OWL. És extensible i independent de plataforma ja que està escrit en Java. L'únic requisit de maquinari és que ha de suportar la versió 1.3 o superior de JDK.

Les principals característiques d'aquest programari que el diferencien d'altres són:

[1] La limitació sobre OWL Lite és aplicable quan la construcció es fa servir en expressions de classe. En l'exemple següent es presenta un individu de la classe `NomClasse` que és alhora una subclasse de `ClasseMare`.

```
<NomClasse rdf:ID="NomIndividu">
        <rdfs:subClassOf rdf:resource="#ClasseMare"/>
</NomClasse>
```

[2] En OWL Lite només s'accepten els valors 0 o 1

- Interfície gràfica intuïtiva i fàcil d'utilitzar
- Arquitectura extensible a base de *plug-ins*

Els projectes creats mitjançant *Protégé* es poden enregistrar en forma de base de dades. El programari és compatible amb qualsevol base de dades que tingui un controlador JDBC 1.0. A la pràctica, això significa la major part de les bases de dades relacionals. A més, els projectes de *Protégé* es poden exportar en fitxers en format RDF Schema o en OWL (prèvia instal·lació del *plug-in* corresponent).

Protégé permet la definició de classes i subclasses. També permet la creació de propietats (anomenades *slots* en la terminologia del programari) i subpropietats, que es poden associar fàcilment a classes prèviament definides. Finalment, també és possible aplicar certes restriccions sobre les propietats. Aquests conceptes de classes, propietats i restriccions fan referència als elements bàsics de les ontologies (veure 4.1)

Altres funcionalitats implementades en el programari són:

- Possibilitat de crear instàncies de les classes definides per així formar una base de dades que respecti l'estructura de l'ontologia
- Creació de formularis per facilitar la interacció dels usuaris amb la base de dades
- Creació de consultes automatitzades de la base de dades

A la pàgina web de *Protégé* [42] es poden descarregar tant el programari, com diversos manuals d'usuari i exemples que ajuden a entendre el funcionament de l'aplicació.

5.4 Conclusions

L'OWL ha estat dissenyat per afavorir el desenvolupament i l'expansió de la Web Semàntica. Aquest llenguatge forma part d'un conjunt de recomanacions del W3C relacionades amb la nova visió de la web.

- **XML** proporciona una sintaxi per a documents estructurats, però no imposa regles semàntiques relatives al significat d'aquests documents.
- **XML Schema** és un llenguatge que imposa regles en l'estructura dels documents XML i hi afegeix tipus de dades.
- **RDF** és un model de dades per a objectes (o recursos) i les relacions entre ells, proporciona una semàntica simple per a aquest model de dades que es poden representar en llenguatge XML.
- **RDF Schema** és un vocabulari per descriure les propietats i les classes del recursos RDF amb un cert grau addicional de semàntica.
- **OWL** afegeix més vocabulari per descriure propietats i classes, com ara, les relacions entre classes, la cardinalitat, característiques de les propietats, etc.

En el proper capítol es proporcionarà una visió en detall de la Web Semàntica i com totes les tecnologies que s'han vist fins ara en aquest treball hi contribueixen.

Capítol 6 - La Web Semàntica

En aquest capítol es tractarà en profunditat el projecte de la Web Semàntica. Primer de tot s'oferirà una visió d'aquest ambiciós projecte que introdueix grans avantatges respecte a la web actual (apartat 6.1). Després es descriuran els principis bàsics de disseny (apartat 6.2) i l'estructura (apartat 6.3) que permetrà implantar aquests principis. També es citaran alguns projectes en desenvolupament (apartat 6.4) que permeten demostrar les principals virtuts de la Web Semàntica. Per acabar el capítol, es presentaran unes conclusions sobre aquesta nova iniciativa d'estructuració de la web actual (apartat 6.5).

6.1 Introducció

La web conté una enorme quantitat d'informació creada per multitud d'organitzacions, comunitats i individus diferents, amb una gran varietat de propòsits. Els usuaris de la web poden accedir a aquesta informació fàcilment especificant adreces URL en un navegador, o bé realitzant cerques mitjançant cercadors especialitzats, o bé seguint els enllaços que són presents en les pàgines que visiten. Aquesta simplicitat en l'ús de la web és un dels aspectes fonamentals de l'èxit d'Internet.

Ara bé, aquesta simplicitat té un cost: és molt fàcil perdre's entre tanta informació o accedir a recursos que no tenen res a veure amb allò que inicialment es busca. En l'última dècada ha existit una preocupació creixent per que les pàgines web fossin comprensibles pels usuaris a través del seu disseny. Per contra, pocs esforços s'han destinat per fer que tota aquesta informació fos també comprensible per a les màquines. El resultat és la web actual: una enorme recopilació de dades desordenades. Avui dia, els usuaris són els responsables de navegar i interpretar el contingut, mentre que els ordinadors només intervenen en les tasques de representació visual. Això fa que, sovint, la tasca de cercar informació útil dins d'aquest desordre sigui difícil i costosa.

Els documents actuals escrits en HTML (en anglès *HyperText Markup Language*) tenen una certa quantitat de metadades que permeten la seva indexació en cercadors. Degut al gran creixement de la web, aquestes metadades són

insuficients, ja que fan referència a la totalitat del document i no a cadascun dels seus components.

Un dels pilars bàsics de la Web Semàntica és aconseguir que els documents continguin una major quantitat d'informació sobre sí mateixos en un llenguatge que sigui comprensible per a les màquines que els emmagatzemen, i també per a aquelles que fan servir aquests documents. La Web Semàntica modificaria la forma de presentar la informació a la web per facilitar el seu processament per part de les màquines. En certa forma, es parla d'una ordenació del caos.

La Web Semàntica es basaria en l'estandardització de totes les seves dades, és a dir, tot allò que formés part de la web (pàgines, serveis, etc) hauria de presentar-se en el mateix format. D'aquesta manera, una nova forma de programari, els famosos agents intel·ligents, podrien explotar aquesta informació de manera molt més eficient, ja que serien capaços d'entendre-la. Això faria, per exemple, que les cerques d'informació fossin molt més precises. Segons [44], "la informació ha de ser compilada de forma que un cercador pugui comprendre-la, en lloc de mostrar-la simplement dins una llista".

A la web actual es produeix un fenomen de creixement caòtic de recursos i això fa que quedi fora de l'abast d'una persona. La semàntica és implícita. Hi ha falta d'ordre i d'organització. A la Web Semàntica la informació és processable pels programes. Hi ha una classificació, una estructura i una anotació. La semàntica és explícita mitjançant l'ús de metadades i existeixen vocabularis comuns (les ontologies).

Aquest nou enfocament proposa l'enriquiment de la web a través de l'estructuració de la informació, afegint-hi components semàntics que puguin ser processats de forma automàtica. La nova generació de llenguatges ve encapçalada per l´XML, l'RDF i l'OWL, que permetran la inclusió d'ontologies que determinaran les relacions entre els conceptes i especificaran les regles lògiques per tal que els agents les reconeguin i classifiquin.

Un dels objectius bàsics de la comunitat *W3C Semantic Web* [45] és la creació dels estàndards que permetin el desenvolupament no centralitzat de la Web Semàntica, de manera que tots els seus components siguin compatibles. En l'apartat 6.2 es detallaran els principis bàsics de la Web Semàntica.

6.2 Principis de la Web Semàntica

Els principis bàsics de la Web Semàntica definits pel W3C [46] són els següents:

1. Qualsevol cosa pot ser identificada mitjançant una URI. Les persones, els llocs i les coses del món físic poden ser identificades de forma inequívoca dins la Web Semàntica per mitjà d'una URI. Qualsevol persona que tingui control sobre un espai de noms a la web podrà crear les URIs. A més, els vocabularis que ja existeixin i que es vulguin fer servir en els documents, com ara el de Dublin [6], s'identifiquen també mitjançant una URI.

2. Els recursos i els enllaços poden tenir un tipus concret. La web actual està formada per recursos i enllaços. Els recursos són, normalment, documents HTML dissenyats per ser consumits per persones (veure Figura 9). És per això que gairebé mai hi trobem metadades en aquests documents que expliquin, per exemple, el seu propòsit o la seva relació amb d'altres. Per a una persona és fàcil deduir quin és el significat dels enllaços d'un recurs envers d'altres pel context en el qual es troba l'enllaç. Per a una màquina, això no és gens evident.

La Web Semàntica també està formada per recursos i enllaços, però la diferència amb la web actual és que són d'un tipus determinat (veure Figura 9). Això fa que les màquines puguin entendre la semàntica d'un document i les seves relacions amb d'altres.

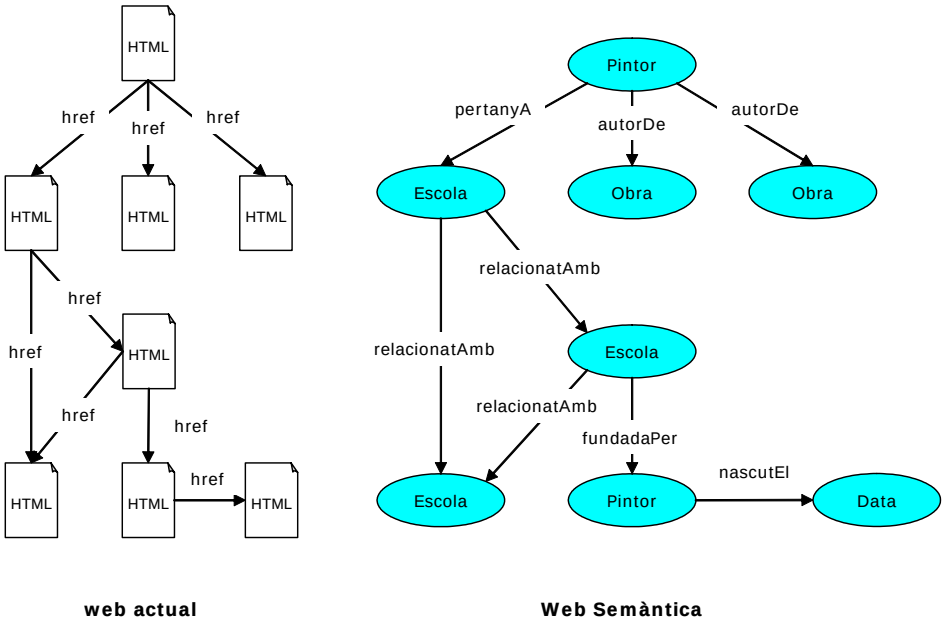

Figura 9 – Comparativa entre la web actual i la Web Semàntica a nivell de recursos i enllaços

3. La informació parcial és permesa. La Web Semàntica, de la mateixa forma que la web actual, no posa límits a la informació que pugui contenir. Qualsevol pot dir allò que vulgui sobre qualsevol tema, i crear enllaços de qualsevol tipus cap a d'altres recursos. En qualsevol moment es podrà afegir informació nova, modificar l'existent o, fins i tot, fer-la desaparèixer. Tot programari que interactui amb la Web Semàntica ha de ser dissenyat amb aquestes premisses.

4. No hi ha necessitat d'una veritat absoluta. Igual que en la web actual, no tot el que es publiqui a la Web Semàntica serà una veritat absoluta. És per això que es preveuran mecanismes que permetin a les aplicacions de verificar l'autenticitat de les dades en base a determinats criteris, com ara, qui és l'autor de la informació.

5. Suport per a l'evolució. La Web Semàntica proporcionarà mecanismes que permetin combinar i fer compatible el treball de comunitats d'usuaris independents. D'aquesta manera, els vocabularis creats per diferents grups podran ser reutilitzats per crear vocabularis més grans i complets de forma transparent.

6. Disseny simplista. La Web Semàntica fa que les coses simples siguin fàcils d'implementar i, alhora, permet realitzar tasques més complexes. L'objectiu del W3C [3] és crear els estàndard d'allò que sigui estrictament necessari. Aquesta filosofia permet la implementació d'aplicacions simples basades en tecnologies que ja existeixen.

6.3 Estructura de la Web Semàntica

Els principis de la Web Semàntica (veure apartat 6.2) s'implementen en forma de capes de tecnologies web i estàndards. La Figura 10 presenta aquest estructura en capes.

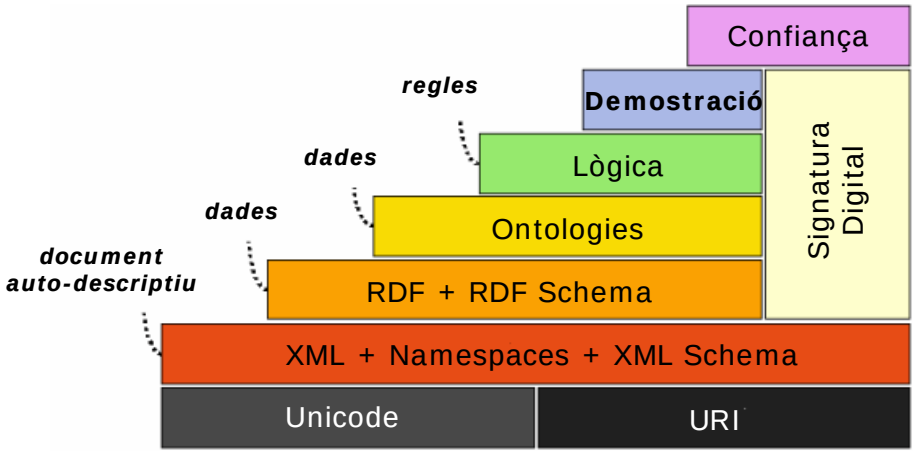

Figura 10 – Estructura en capes de la Web Semàntica

Les capes *Unicode* i *URI* asseguren que es faran servir conjunts de caràcters internacionals i proporciona els mitjans per identificar els objectes dins la Web Semàntica.

La capa *XML*, que inclou els espais de noms i les definicions d'esquemes, assegura la integració de les definicions que es facin a la Web Semàntica, amb d'altres estàndards basats en el llenguatge XML.

La capa *RDF* i *RDF Schema* permet la creació d'axiomes sobre els objectes i la definició de vocabularis que es podran identificar mitjançant una URI. En aquesta capa es poden assignar tipus als recursos i als enllaços.

La capa d'*Ontologies* suporta l'evolució dels vocabularis definits a la capa inferior, ja que permet la definició de relacions entre diferents conceptes.

La capa de *Signatura Digital* permet la detecció de modificacions en els documents.

Les capes superiors (*Lògica, Demostració* i *Confiança*) estan actualment en vies d'estudi. Avui dia només existeixen aplicacions simples que serveixen com a demostradores d'aquestes tecnologies. L'objectiu d'aquestes capes és el següent: la capa *Lògica* permet la definició de regles, mentre que la capa *Demostració* executa aquestes regles i avalua, juntament amb la capa *Confiança*, si confiar en la demostració realitzada o no.

6.4 Exemples

En aquest apartat es presenten alguns dels projectes que estan en plena fase de desenvolupament i que serveixen per demostrar la integració de les diverses tecnologies que formaran la Web Semàntica. A més, també permeten mostrar alguns dels avantatges de disposar d'una web estructurada semànticament.

- ***On-To-Knowledge*** [47]. L'objectiu d'aquest projecte és desenvolupar els mètodes i les eines necessàries per tal d'aprofitar al màxim les possibilitats que

ofereix l'estructuració de la informació entorn a les ontologies. Aquest projecte es centra, sobretot, en aplicacions que permetin accedir a grans bases de dades de coneixement distribuïdes, de manera natural pels usuaris.

- ***Ontoweb*** [48]. És una xarxa temàtica creada per la comissió europea. El seu objectiu és el de centralitzar les activitats dedicades a la creació de mètodes i eines relacionades amb les ontologies.

- ***Riboweb*** [49]. És un sistema accessible des d'Internet dedicat als ribosomes, un mecanisme cel·lular que permet crear les proteïnes a qualsevol organisme. El sistema utilitza quatre ontologies principals per organitzar tota la informació. La base de dades de coneixement conté dades que han estat prèviament contrastades i publicades, i mòduls computacionals que permeten processar aquestes dades per obtenir hipòtesis sobre l'estructura dels ribosomes.

A [50] es pot consultar una llista força més completa d'altres projectes relacionats amb el desenvolupament de la Web Semàntica.

6.5 Conclusions

Avui dia totes les activitats i projectes relacionats amb la Web Semàntica es troben en plena efervescència. Es tracta d'un concepte nou, una nova visió que, d'altra banda, és plenament lògica donats els problemes de manca d'estructuració de la web actual. Les diferències respecte els serveis actuals seran múltiples i, entre d'altres, es poden citar les següents:

Característica	Web actual	Web Semàntica
Llenguatge principal	HTML	XML
Forma i estructura	Documents no estructurats	Documents estructurats
Semàntica	Implícita en els documents	Explícita mitjançant l'ús de metadades
Relació entre contingut i forma	L'HTML fusiona la forma i el contingut	Forma i contingut són separats
Audiència	Humans	Humans i màquines

Taula 4 – Algunes diferències entre la web actual i la Web Semàntica

Un aspecte que no es recull en la Taula 4 i que també és força important, és que la Web Semàntica funciona pels principis de la comprensió parcial i la inferència de la informació, és a dir, es poden deduir nous coneixements a partir d'altres que s'han entès.

Tot i que la totalitat de les capes de la Web Semàntica no estan actualment definides (veure apartat 6.3), ja es poden implementar aplicacions que permetin demostrar el potencial d'aquesta nova visió de la web (veure apartat 6.4). El treball dels principals grups d'investigació [45] es centra en l'estandardització de les capes superiors del model presentat a la Figura 10. Això permetrà obtenir la definició global de la Web Semàntica.

Un altre punt on també es dediquen molts dels esforços és en desenvolupar el programari que permeti als usuaris entendre de manera fàcil com introduir les metadades en els seu documents. L'ideal seria que això es pogués realitzar de forma gairebé transparent. La proliferació de documents estructurats a la web permetrà l'aparició de noves aplicacions que facin canviar el concepte i la forma d'ús de la web actual.

Capítol 7 - Cas Pràctic

En aquest capítol es descriu el desenvolupament del cas pràctic. L'objectiu és dissenyar una ontologia que permeti la classificació de les paraules de la llengua catalana (apartat 7.1) de manera similar a l'estructuració que es fa servir en *WordNet*. Per implementar aquesta ontologia es farà servir el llenguatge OWL. Per facilitar el desenvolupament, s'utilitzarà el programari *Protégé* i el *plug-in* corresponent que permet treballar amb l'OWL. Un cop definida l'ontologia (apartats 7.2 i 7.3) es crearà una petita base de dades lèxica que permeti verificar si l'estructura ontològica és correcta (apartat 7.4). Finalment, es presentaran algunes conclusions derivades de la realització del cas pràctic (apartat 7.5).

7.1 Requeriments

Els requeriments del treball són els següents:

Req. 1 – *L'ontologia ha de ser implementada en llenguatge OWL.*

Req. 2 - *La base de dades lèxica ha de contenir noms, verbs, adjectius i adverbis.*

Req. 3 – *L'ontologia ha de representar les relacions taxonòmiques que existeixen entre els noms.*

Req. 4 – *L'ontologia ha de representar les relacions de sinonímia i antonímia entre mots.*

Req. 5 – *Les relacions d'implicació entre verbs han d'aparèixer a la base de dades.*

Req. 6 – *Les relacions del tipus "part-tot" han d'estar representades a l'ontologia.*

7.2 Disseny

7.2.1 Classes

7.2.1.1 Paraula

Primer de tot, cal identificar els conceptes principals del domini per tal de dissenyar les classes de l'ontologia. En aquest cas, el domini està format per totes les paraules de la llengua catalana. Així doncs, el primer concepte a representar serà el

de **Paraula**. És a dir, *Paraula* serà la primera de les classes. Segons el requeriment Req. 2, la base de dades lèxica ha de contenir noms, verbs, adjectius i adverbis. Tots aquests elements gramaticals són també conceptes del domini i, per tant, també hauran d'estar representats en forma de classe a l'ontologia. Com són tipus diferents de paraules, cadascun d'ells es representarà com a subclasse de *Paraula*.

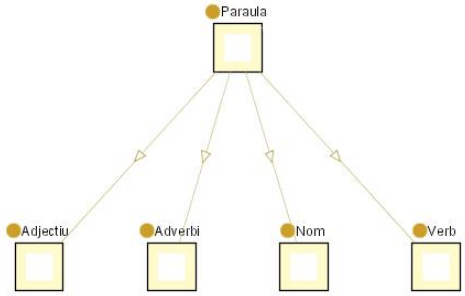

Figura 11 – Jerarquia de la classe *Paraula*

La Figura 11 mostra la classe *Paraula* i les seves quatre subclasses o especialitzacions: *Nom, Verb, Adjectiu* i *Adverbi.* Amb aquesta representació de classes es considera satisfet el requeriment Req. 2. Cal afegir que les subclasses s'han considerat disjuntes entre sí. Tot i que hi ha paraules que poden ser de diferents tipus tot i conservar la mateixa grafia, a l'hora de crear la base de dades lèxica i classificar de forma adequada cada paraula, s'han de considerar com a paraules diferents. Per exemple, la paraula *verd* actua com a nom en la frase "*el verd és el meu color preferit*", i com a adjectiu en "*el cotxe verd*". Ara bé, la taxonomia de *verd* com a nom no serà la mateixa que la de *verd* com a adjectiu. És per això que és important fer la disjunció entre subclasses.

7.2.1.2 Synset

En base a l'estructura de *WordNet*, un altre concepte molt important és el de conjunt de sinònims o *synset*, que conté totes les paraules amb un mateix significat, és a dir, paraules que són sinònimes entre elles. En conseqüència, una altra classe dins l'ontologia serà **Synset**. De la mateixa manera que per a *Paraula*, es poden distingir quatre tipus o especialitzacions de *Synset*: **SynsetNom**, **SynsetVerb**, **SynsetAdjectiu** i **SynsetAdverbi**. Com en el cas de *Paraula*, aquestes subclasses també son disjuntes, ja que un *synset* determinat no pot pertànyer al mateix temps a diferents tipus.

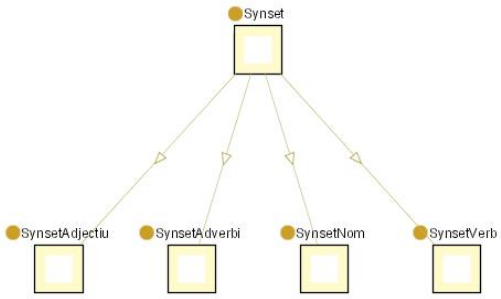

Figura 12 - Jerarquia de la classe *Synset*

La Figura 12 mostra la classe *Synset* i les seves quatre subclasses. Aquesta representació de classes permet satisfer parcialment el requeriment Req. 4, ja que els individus que pertanyin a una determinada subclasse, seran sinònims entre sí.

Un altre detall que cal mencionar és que tant la classe *Paraula* com *Synset* són subclasses d'`owl:Thing` i disjuntes entre sí. A partir d'aquesta classificació

jeràrquica de classes es poden començar a definir les propietats que relacionen els conceptes del domini.

7.2.2 Propietats

7.2.2.1 Sinonímia

La propietat més òbvia dins l'ontologia és la que relaciona els conceptes de *Synset* i *Paraula*. És clar que cadascun dels *synsets* està format per una o més paraules. Així doncs, la primera propietat identificada és **teParaula**.

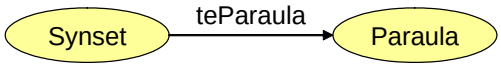

Figura 13 – Propietat *teParaula*

La Figura 13 mostra la relació *teParaula*. Es tracta d'una propietat d'objecte (veure apartat 5.2.3), on el domini el formen els individus de la classe *Synset*, i el rang està format per les instàncies de la classe *Paraula*.

7.2.2.2 Descripció

A *WordNet*, cada conjunt de sinònims té una definició associada que inclou, normalment, un exemple que il·lustra la semàntica del conjunt de paraules. Aquesta relació es pot representar en l'ontologia mitjançant una propietat de tipus (veure apartat 5.2.3) anomenada **glossari**, que té com a domini els individus de la classe *Synset* i com a rang el tipus de dades `xsd:string`. La Figura 14 representa gràficament aquesta relació.

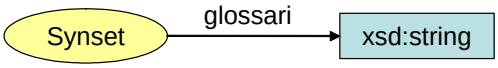

Figura 14 – Propietat *glossari*

7.2.2.3 Hiponímia i hiperonímia

Tal com s'ha discutit a l'apartat 4.2, la taxonomia del llenguatge, és a dir, les relacions hiponímia i d'hiperonímia, només s'estableixen per als noms i per als verbs. Aquestes relacions, a més, són aplicables als *synsets* i no a cadascuna de les paraules individualment. En conseqüència, es poden definir dues noves propietats: **esHiponimDe** i **esHiperonimDe**. Com es veu a la Figura 15, en ambdós casos, tant el domini com el rang el formen les instàncies de la classe *SynsetNom*. A més, es tracta de propietats inverses i transitives. És a dir, si X és hipònim de Y i Y és hipònim de Z, aleshores, X és hipònim de Z. El mateix és aplicable a la propietat *esHiperonimDe*. Amb aquestes dues propietats es satisfà el requeriment Req. 3.

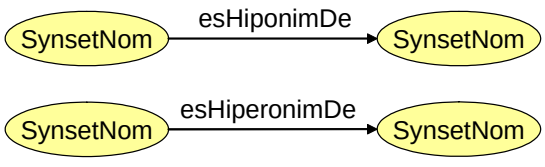

Figura 15 – Propietats *esHiponimDe* i *esHiperonimDe*

7.2.2.4 Meronímia i holonímia

Una particularitat referida exclusivament als noms és l'existència de les relacions de meronímia i holonímia. Dins l'ontologia es representen en forma de propietats d'objecte anomenades **esMeronimDe** i **esHolonimDe**. En ambdós casos, tant el

domini com el rang són els individus de la classe *SynsetNom* (veure Figura 16). Cal esmentar que aquestes propietats són inverses entre sí. Tal com s'estableix a [29], ni la meronímia ni l'holonímia es poden considerar propietats transitives. Un exemple per mostrar això seria el següent: si un *pany* és un merònim de *porta* i *porta* és un merònim de *casa*, no és molt natural afirmar que *la casa té un pany*, tot i que sigui correcte. Per aquest motiu, no es consideren relacions transitives.

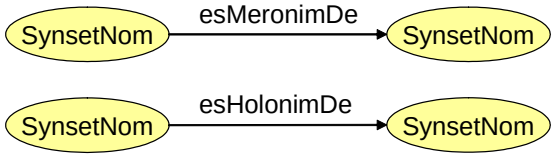

Figura 16 – Propietats *esMeronimDe* i *esHolonimDe*

A *WordNet* es representen 3 tipus de meronímia [29], encara que n'existeixen d'altres. Aquests tipus s'han implementat també en l'ontologia com:

- La relació "*part de*" o "*component de*", mitjançant les propietats d'objecte **esPartMeronimDe** i **esPartHolonimDe** que són inverses entre sí.
- La relació "*substància de*", mitjançant les propietats d'objecte **esSubstanciaMeronimDe** i **esSubstanciaHolonimDe** que són inverses entre sí. Aquesta relació fa referència als materials o ingredients a partir dels quals es construeix o es fa alguna cosa.
- La relació "*membre de*", mitjançant les propietats d'objecte **esMembreMeronimDe** i **esMembreHolonimDe** que són inverses entre sí. Fa referència a allò que forma part d'una comunitat, d'un grup o d'una classe.

El fet que aquestes relacions siguin especialitzacions dels conceptes de meronímia i holonímia permet crear la jerarquia de propietats que es mostra a la Figura 17. Això fa que totes elles heretin el domini i el rang de la seva propietat pare corresponent. Finalment, cal esmentar que amb aquest conjunt de propietats es satisfà el requeriment Req. 6.

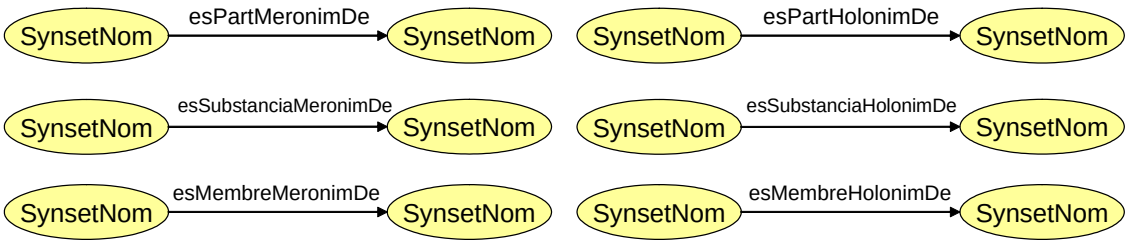

Figura 17 – Subpropietats de *esMeronimDe* i *esHolonimDe*

7.2.2.5 Atribut

Una altra propietat que s'ha inclòs dins l'ontologia és la que representa la relació entre els noms i els adjectius. Aquesta propietat rep el nom d'**atribut** (veure Figura 18). En aquest punt cal fer una precisió, ja que un nom pot tenir infinitat d'adjectius, així que només s'utilitzarà aquesta propietat per relacionar noms i adjectius que estiguin íntimament lligats, com ara, *passivitat* i *passiu*. La relació s'estableix entre els *synsets* de noms (el domini) i els *synsets* d'adjectius (el rang).

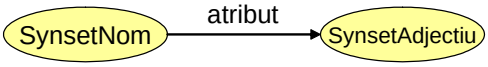

Figura 18 – Propietat *atribut*

7.2.2.6 Implicació verbal

Pel que fa als verbs, a l'apartat 4.2 s'ha indicat que existeixen tres relacions possibles a *WordNet*: troponímia, causació i implicació. Totes elles han estat implementades a l'ontologia mitjançant les propietats **esTroponimDe**, **causa** i **implica**, respectivament. En els tres casos, el domini i el rang de la relació són les instàncies de la classe *SynsetVerb* (veure Figura 19).

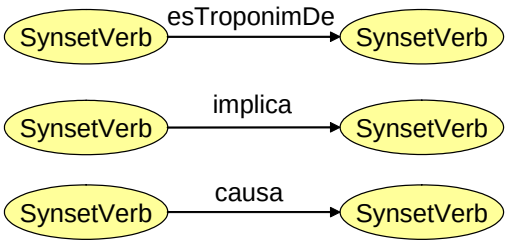

Figura 19 – Propietats *esTroponimDe*, *implica* i *causa*

Cal destacar que les propietats *esTroponimDe* i *implica* s'han definit com transitives. Per exemple, si X implica Y i Y implica Z, aleshores X implica Z. El cas de *esTroponimDe* és el mateix que s'ha explicat abans per les propietats *esHiponimDe* i *esHiperonimDe*. És important esmentar que la propietat *implica* no és simètrica ja que, per exemple, si aprovar implica haver estudiat, estudiar no implica necessàriament que s'aprovi. Amb les relacions presentades a la Figura 19 es satisfà el requeriment Req. 5.

7.2.2.7 Antonímia

Per acabar, s'ha definit la propietat **antònim** per representar la relació d'antonímia. Aquesta propietat és aplicable a les paraules i no als conjunts de sinònims, ja que es tracta d'una relació lèxica entre mots i no d'una relació semàntica. Per exemple, els *synsets* {pujar, ascendir} i {baixar, descendir} són conceptes oposats, però no antònims en el sentit estricte de la paraula. Els antònims són pujar/baixar i ascendir/descendir. És una propietat simètrica, ja que les afirmacions *A és antònim de B* i *B és antònim d'A* són equivalents.

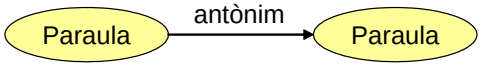

Figura 20 – Propietat *antònim*

La Figura 20 mostra la propietat *antònim*. Tant el domini com el rang correspon als individus de la classe *Paraula*. Amb aquesta propietat s'acaba de satisfer completament el requeriment Req. 4.

7.2.3 Restriccions

A les propietats definides en l'apartat 7.2.2 s'han aplicat les restriccions descrites a la Taula 5:

Propietat	Tipus restricció	Descripció
glossari	*cardinalitat = 1*	Tot *synset* ha de tenir una descripció associada.
teParaula	*cardinalitat ≥ 1*	Tot *synset* ha de contenir almenys una paraula.

Taula 5 – Restriccions sobre l'ontologia

7.3 Implementació

7.3.1 Programari

Per implementar l'ontologia s'han fet servir les eines següents:

- ***Protégé* versió 3.1 beta [42].** Veure apartat 5.3 per obtenir una descripció més completa d'aquest editor.
- ***Plug-in* OWL per *Protégé* versió 2.1 [43].** Es tracta d'un afegit a l'editor *Protégé* que permet crear i editar ontologies en el llenguatge OWL.
- ***Plug-in* Jambalaya per *Protégé* [51].** És un altre afegit per a *Protégé* que permet obtenir representacions gràfiques de la major part dels elements de l'ontologia.
- ***Plug-in* OWLViz per *Protégé* [52].** Un altre afegit gràfic per a l'editor.
- ***RACER* versió 1.7.24 [53].** És una aplicació que permet analitzar l'ontologia i inferir nous coneixements a partir d'ella, com ara, relacions implícites entre classes.

Cal destacar que totes aquestes aplicacions són programari de tipus gratuït i la seva descàrrega està disponible des d'Internet.

7.3.2 Elecció del subllenguatge OWL

El subllenguatge d'OWL escollit per implementar l'ontologia és l'**OWL DL**. El motiu d'aquesta elecció és el fet de representar com a disjuntes les subclasses de *Paraula*, així com les de *Synset*. Aquesta és l'única característica de l'ontologia que fa que no es pugui implementar en *OWL Lite* (veure Taula 3).

Fent servir l'opció de l'editor *Protégé* per determinar el subllenguatge emprat, s'obté el resultat que es mostra a la Figura 21.

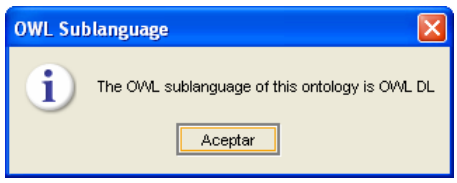

Figura 21 – Confirmació del subllenguatge OWL escollit

Cal esmentar que les relacions d'hiponímia i d'hiperonímia s'haguessin pogut representar mitjançant clàusules de tipus `rdfs:subClassOf` a l'hora de definir els individus. Per exemple, en l'Exemple 30 es declara un individu de la classe *SynsetNom* anomenat `rossinyol` que té una relació d'hiponímia amb un altre individu anomenat `ocell`. Ara bé, el fet de considerar els individus com a classes fa que el subllenguatge a utilitzar sigui l'*OWL Full* (veure Taula 3), i això representa molts inconvenients a l'hora de fer servir eines de verificació o de consulta de l'ontologia. Per això, finalment, s'ha decidit implementar aquestes relacions com a propietats d'objecte.

```
<SynsetNom rdf:ID="rossinyol">
    <rdfs:subClassOf rdf:resource="#ocell">
</SynsetNom>
```

Exemple 30 – Representació de l'hiponímia amb `rdfs:subClassOf`

7.4 Tests

7.4.1 Sintaxi i coherència

El primer test d'aquest tipus s'ha realitzat amb l'opció inclosa dins el *Protégé* que permet verificar l'integritat de l'ontologia fent diverses comprovacions respecte l'estàndard del subllenguatge escollit. Aquesta opció es troba en el menú `OWL` → `Run Ontology Tests`. Els resultats són correctes i no es detecta cap inconsistència.

Els tests de coherència s'han realitzat amb l'ajut de l'eina *RACER*. Aquesta aplicació permet analitzar l'ontologia creada i definir la jerarquia de classes i propietats inferides. L'anàlisi de les jerarquies inferides determina la coherència del model. En el cas de l'ontologia creada en aquest treball, el model inferit pel *RACER* és exactament igual al model definit. Així doncs, el resultat del test també és correcte.

7.4.2 Introducció d'individus

Un cop definida l'ontologia, s'han introduït diversos individus relacionats entre sí per comprovar si totes les relacions són consistents. El tema escollit ha estat els animals de companyia, a partir dels quals s'ha anat creant tota la taxonomia d'individus. La base d'aquesta jerarquia la formen els noms *gos*, *gat* i *tortuga*. Les relacions que es creen a partir d'aquestes paraules demostren gran part de les relacions definides a l'ontologia. A mode d'exemple, la Figura 22 mostra la taxonomia creada a partir de la paraula *gat*.

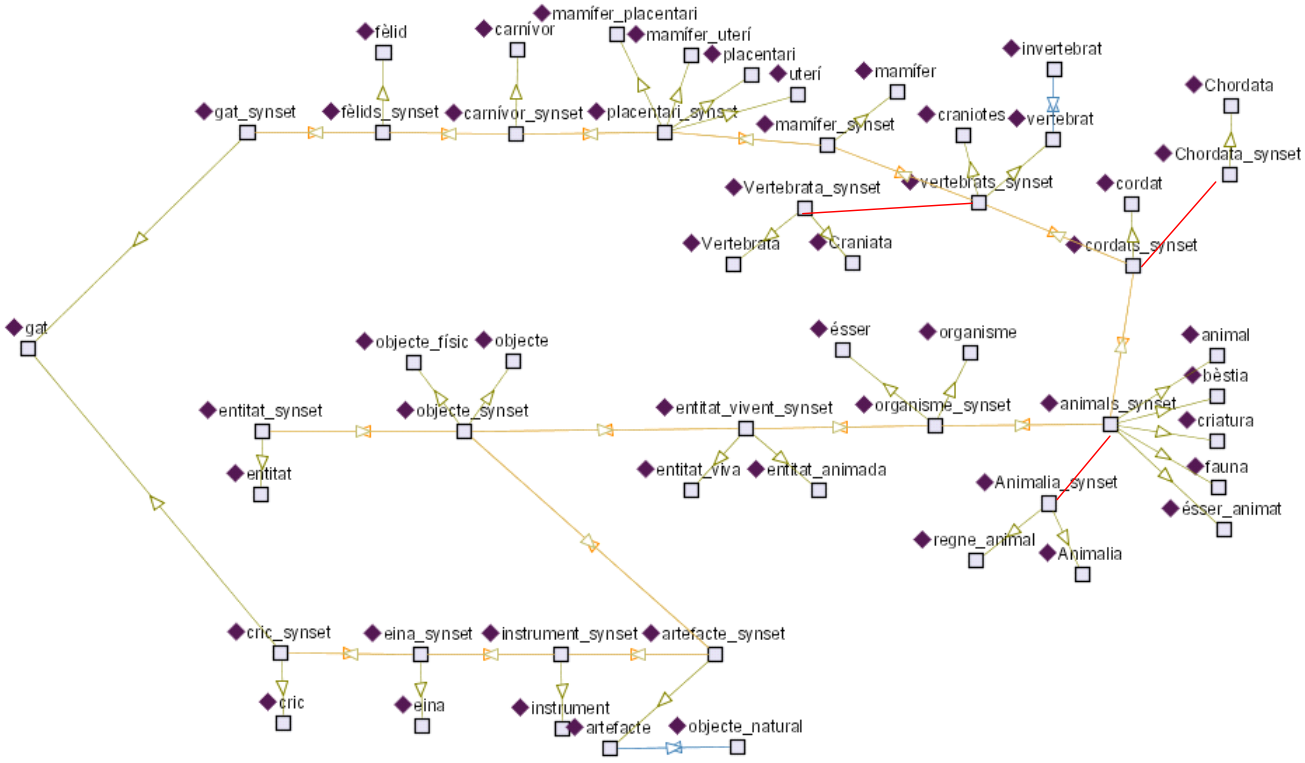

Figura 22 - Taxonomia implementada a partir del nom *gat*

El mot gat pertany a dos grups de sinònims: *gat_synset*, en referència a l'animal domèstic; *cric_synset*, en referència a l'eina mecànica. A partir de cadascun d'ells

s'ha creat una jerarquia de *synsets* fins a arribar a un node superior de l'estructura: *entitat_synset*. Entremig es troben individus com ara *animals_synset* que conté les paraules *animal*, *bèstia*, *criatura*, *fauna* i *ésser_animat*. Aquest individu també té una relació de tipus *esMembreMeronimDe* amb *Animalia_synset*, que representa el grup dels animals.

Per exemplificar les propietats verbals implementades a l'ontologia, la Figura 23 mostra les relacions que es creen a partir del verb **planar**. Aquest verb forma part del conjunt de sinònims *planar_synset*. Aquest conjunt implica l'acció descrita per l'individu *volar_synset*, és a dir, planar implica volar. Aquest nou *synset* conté la paraula *volar*, i és tropònim del conjunt *traslladar_synset*, format per les paraules *anar, traslladar* i *mudar.* Finalment, l'acció representada per *traslladar_synset* causa l'acció representada per *moure_synset*, que conté la paraula *moure*.

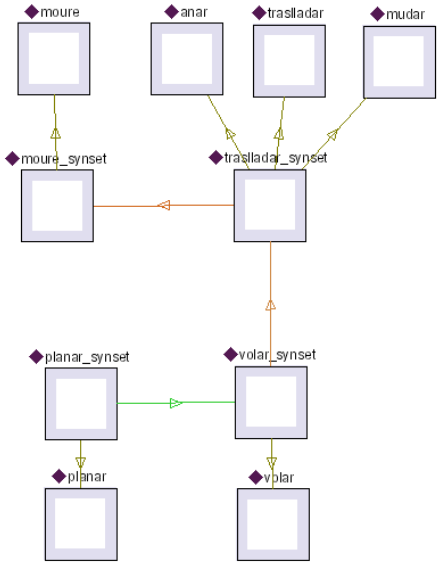

Figura 23 – Relacions verbals creades a partir de *planar*

7.5 Conclusions

La primera conclusió que es pot extraure després de la realització del cas pràctic, és que construir una ontologia no és una tasca fàcil ni intuïtiva. És necessari un coneixement profund del domini que es vol representar, ja que l'utilitat de l'ontologia dependrà en gran mesura de l'elecció de classes que s'hagi fet.

Les relacions entre els diferents conceptes tampoc acostuma a ser evident. A més, cal pensar en les característiques addicionals que puguin tenir aquestes propietats, com ara, el fet que siguin transitives o inverses.

El fet que el cas pràctic que s'ha implementat en aquest treball s'hagi basat en l'estructura de *WordNet* ha facilitat enormement el disseny de l'ontologia. Conceptes com ara els *synsets* o les diferents relacions de meronímia no són evidents si no es tenen grans coneixements lexicogràfics.

L'esforç més gran en crear una ontologia consisteix en reflexionar sobre el seu disseny: classes, propietats i restriccions. Un cop tots aquests conceptes es troben en un estat de maduresa avançat, ja es pot començar la fase de creació d'individus. Si hom es precipita i comença a introduir individus abans que l'estructura de base sigui prou madura, el risc que aquesta tasca sigui finalment inútil és força elevat.

Capítol 8 - Conclusions i Línies Futures

En aquest treball s'ha fet una anàlisi de les diverses tecnologies que representen la base per a la construcció de la Web Semàntica. Al llarg de la memòria, aquestes tecnologies s'han presentat partint de les que tenen un major grau de maduresa i acceptació en el mercat, és el cas de l'XML, fins a aquelles que es basen en estàndards que fa poc que s'han aprovat, és el cas de l'OWL.

La descripció dels llenguatges XML, RDF i OWL, juntament amb la presentació del concepte d'ontologies aplicades al camp de la informàtica, ha permès al lector adquirir els conceptes bàsics necessaris per introduir la visió de la Web Semàntica i, alhora, entendre els seus principals avantatges respecte a la web actual.

En aquest capítol es presenten les conclusions derivades de l'estudi realitzat. A més, es proposen línies futures d'investigació relaciones amb aquest camp emergent de la informàtica.

8.1 Conclusions

L'èxit del llenguatge XML és indiscutible. La indústria l'ha anat adoptant al llarg dels darrers anys i, avui dia, és àmpliament utilitzat. La llibertat que proporciona a l'hora de crear etiquetes pròpies, juntament amb l'existència de nombroses aplicacions que permeten verificar l'estructura dels documents, són molt apreciades i valorades. Si hi afegim els avantatges de l'XML Schema que permeten especificar els tipus dels elements o les estructures jeràrquiques, el suport massiu que ha tingut per part de la indústria informàtica queda justificat.

El fet que a la web sigui possible trobar multitud d'eines per editar, validar, modificar i visualitzar els documents escrits en XML, és una demostració més de l'èxit assolit per aquest llenguatge. Un altre aspecte que ha afavorit l'amplia acceptació de l'XML és el fet que es pugui fer servir com a metallenguatge, és a dir, per crear altres llenguatges basats en ell. Així, han aparegut altres estàndards derivats de l'XML, com ara l'XSL (de l'anglès *eXtensible Stylesheet Language*) o l'XMI (de l'anglès *XML Metadata Interchange*), que també disposen d'un nombre d'usuaris considerable.

L'RDF i l'RDF Schema constitueixen un primer esforç per a la introducció de contingut semàntic en els documents. Tot i que l'RDF no es pugui considerar un llenguatge pròpiament dit sinó un model de dades, la seva representació en XML fa possible introduir aquest model en format electrònic. Un cop més, l'XML demostra la seva versatilitat.

L'RDF permet crear ontologies simples però efectives. Aquest llenguatge, a més, promou l'inici del desenvolupament d'altres programaris que tenen una gran importància en el conjunt de la Web Semàntica: els agents intel·ligents. Es tracta de programes que són capaços d'interpretar les dades contingudes en un document (gràcies a les metadades introduïdes per l'autor) i permeten inferir nous coneixements a partir d'aquells que ja han après. Tot això és possible gràcies a la semàntica definida per l'RDF.

L'OWL constitueix finalment el grau d'expressió màxim que es pot assolir, fins ara, en la semàntica dels documents. És l'estàndard del W3C per representar les ontologies a la web. Partint de la base de l'RDF, l'OWL aporta moltes millores i més expressivitat. D'aquesta manera, els autors dels documents tenen més possibilitats a l'hora de definir relacions entre els diferents conceptes. Això facilita, sens dubte, les capacitats d'inferència automàtica de coneixement dels agents.

Les ontologies que es poden crear mitjançant l'ús de l'OWL són molt més complexes i completes que les creades amb l'RDF, en el sentit que els conceptes que s'hi representen estan millor relacionats i formen bases de dades de coneixement molt més robustes i coherents.

La sintaxi de l'OWL es basa de nou en l'XML. Això permet aprofitar totes les eines disponibles per l'XML, especialment, els processadors de contingut.

Tots aquests llenguatges constitueixen la base de l'estructura de la Web Semàntica (veure Figura 10). Les capes superiors es troben en fase d'anàlisi i disseny, de manera que, encara falten uns anys per a la publicació dels estàndards corresponents.

La visió de la Web Semàntica és, sens dubte, molt ambiciosa. A primera vista, es podria dir que gairebé utòpica. No serà fàcil començar a produir documents amb quantitats ingents de metadades. En principi, això només serà possible per a aquells usuaris que coneguin molt bé les possibilitats dels llenguatges actuals, ja que els editors actualment disponibles no donen gaires facilitats per a la introducció simple de les metadades.

És necessari un gran esforç pel desenvolupament d'eines informàtiques que facin el treball més simple als usuaris. Aquesta simplificació en la redacció dels documents OWL esdevindrà la clau per afavorir l'expansió de la Web Semàntica. Cal tenir present que un dels factors més importants que ha afavorit la ràpida expansió de la web actual i el seu ritme de creixement frenètic, ha estat, precisament, la simplicitat a l'hora de crear continguts per a la web. Actualment, qualsevol persona és capaç de crear la seva pròpia pàgina web sense gaire esforç ja que existeixen infinitat d'editors que fan gran part de la feina. Amb una bona dosi de paciència i inspiració artística, es poden crear veritables obres d'art electròniques sense saber ni tan sols què és l'HTML.

La gran tasca, però, és crear les aplicacions que permetin la migració de tota la informació actual en un format estructurat que sigui compatible amb els estàndards de la Web Semàntica. Aquest és el factor que pot fer evaporar tots els esforços i les il·lusions dipositades en aquest gran projecte. No serà gens fàcil desenvolupar

aquestes aplicacions. A priori, la tasca sembla titànica i, fins i tot, irrealitzable. Fins ara, continua essent una de les grans incògnites que envolten el projecte de la Web Semàntica.

Segurament, la Web Semàntica es començarà a implantar en comunitats específiques dedicades a la medicina, comunicacions, informàtica, etc. Seran com una mena d'illes de coneixement estructurat dins del gran oceà caòtic de la web. Durant un període de temps llarg, aquestes dues formes de veure la web hauran de conviure i, segurament, es crearan aplicacions o estàndards que permetin la connexió entre elles.

Quan s'assoleixi un grau de maduresa alt en aquests àmbits, arribarà el torn de fer el salt cap a d'altres camps amb més usuaris potencials. Seria el cas, per exemple, d'aplicacions d'*e-commerce* (activitats comercials per Internet). Si no s'arriba mai a crear el programari que permeti migrar la informació de la web actual a l'estructura de la Web Semàntica, s'hauran de crear estàndards que permetin l'intercanvi de dades entre ambdues.

En definitiva, la Web Semàntica és un projecte amb grans expectatives que pretén canviar de forma radical la manera amb què fem servir la web. En aquests moments és un edifici a mig construir, però que ja permet imaginar multitud d'aplicacions i oportunitats de negoci. I aquests poden ser els factors que finalment l'ajudin en el seu desenvolupament i expansió. Tot i que no serà fàcil.

8.2 Línies futures

Les línies futures de continuació d'aquest treball podrien ser múltiples:

- Fer un estudi sobre l'estat de desenvolupament de les capes superiors del model de la Web Semàntica: Lògica, Demostració i Confiança.

- Dissenyar una web semàntica més ambiciosa que la presentada en aquest treball, que permeti, per exemple, la importació d'ontologies ja existents o l'interacció amb altres ontologies.

- Fer un estudi dels llenguatges que existeixen en l'actualitat per fer consultes sobre els documents escrits en RDF i OWL.

- Estudiar la lògica de les aplicacions que permeten inferir nous coneixements a partir de les metadades especificades en documents RDF i OWL.

- Enriquir l'ontologia del cas pràctic afegint-hi noves relacions i més instàncies.

- Desenvolupar una interfície per a l'ontologia del cas pràctic que permeti fer consultes sobre la base de dades i mostri els resultats els resultats de manera anàloga a com ho fa *WordNet*.

Glossari

Agent. En termes informàtics, un agent és un programari autònom o semi-autònom proactiu i reactiu. Per ser considerat com un agent, un programari ha de ser un programa independent, capaç de prendre decisions i realitzar accions que satisfacin els objectius pels quals ha estat desenvolupat, basant-se en l'anàlisi del seu entorn.

Diagrama de Gantt. És un tipus de diagrama de barres molt popular desenvolupat per Henry L. Gantt el 1910. Permet mostrar la planificació de les tasques d'un projecte en una escala temporal.

DTD (*Document Type Definition*). Un DTD és una declaració en un document SGML o HTML que especifica restriccions en la seva estructura. El DTD pot ser inclòs dins l'arxiu del document, però normalment s'emmagatzema en un fitxer de text separat. El DTD descriu típicament tots els elements admissibles dins del document, els atributs possibles i, opcionalment, els valors permesos per a aquests atributs. També és possible indicar-hi l'estructura jeràrquica dels elements i la seva cardinalitat.

Espai de noms XML. És un estàndard del W3C que proporciona elements i atributs identificats de manera unívoca dins d'un document XML. Un document XML pot utilitzar elements o atributs que pertanyin a més d'un vocabulari. Si s'assigna un espai de noms a cadascun d'aquests vocabularis, s'elimina qualsevol possible ambigüitat entre elements o atributs amb el mateix nom que pertanyin a vocabularis diferents.

HTML (*Hyper Text Markup Language*). L'HTML és un llenguatge de marques dissenyat per estructurar documents i presentar-los en forma d'hipertext, que és el format estàndard de les pàgines web. Aquest llenguatge deriva de l'SGML.

Ontologia. Típicament, una ontologia és una estructura jeràrquica de dades que conté totes les entitats rellevants d'un domini i les relacions possibles entre elles. Existeixen diversos llenguatges per representar les ontologies de forma electrònica, com ara l'OWL o el DAML+OIL. Les ontologies constitueixen un dels pilars bàsics de la Web Semàntica.

OWL (*Ontology Web Language*). És un llenguatge de marques dissenyat per a la publicació i la compartició de dades per Internet mitjançant ontologies. Constitueix una extensió de l'RDF i es deriva del DAML+OIL. Es divideix en tres subllenguatges, cadascun d'ells amb un nivell de complexitat creixent: OWL Lite, OWL DL i OWL Full.

Protégé. És un entorn gràfic gratuït per a la creació i edició d'ontologies i bases de coneixement. Disposa d'un *plug-in* que el fa compatible amb el llenguatge OWL.

RDF (*Resource Description Framework*). L'RDF és l'especificació d'un model de metadades desenvolupat pel W3C. Aquest model es basa en la idea de convertir les declaracions dels recursos en expressions del tipus subjecte-predicat-objecte. El subjecte és el recurs, és a dir, allò que es descriu. El predicat és la relació o

propietat que es desitja establir per al subjecte. L'objecte és el valor de la propietat. Aquest model s'implementa normalment en el llenguatge XML.

RDF Schema. És una extensió de l'RDF que descriu com definir vocabularis RDF utilitzant el mateix llenguatge RDF.

SGML (*Standard Generalized Markup Language*). Aquest llenguatge serveix per especificar les regles d'etiquetat dels documents, tot i que no imposa cap conjunt d'etiquetes en particular. L'Organització Internacional d'Estàndards (la ISO) va normalitzar l'SGML el 1986. Llenguatges com l'XML o l'HTML són versions més simples de l'SGML.

URI (*Uniform Resource Identifier*). Es tracta d'una cadena de caràcters que serveixen per identificar unívocament qualsevol recurs (servei, lloc, persona, document…)

URL (*Uniform Resource Locator*). És una cadena de caràcters que identifica de manera unívoca a cadascun dels recursos d'informació disponibles a Internet.

W3C (World Wide Web Consortium). És l'organització que produeix els estàndards per a la web. Està dirigida per Tim Berners-Lee, creador de les URLs, el protocol HTTP i el llenguatge HTML. La majoria de les recomanacions creades pel W3C són recolzades pels principals fabricants de programari. El rol d'aquesta organització és vital, ja que amb la creació d'estàndards oberts garanteix que cap fabricant assoleixi el monopoli d'explotació de la web.

Web Semàntica. La Web Semàntica té com a objectiu la creació d'un medi universal per a l'intercanvi d'informació basat en representacions del significat dels recursos de la web, d'una manera intel·ligible per a les màquines. Amb això es pretén ampliar l'operativitat entre els sistemes informàtics i reduir la mediació d'operadors humans en els processos intel·ligents de flux d'informació. Aquesta iniciativa per a la construcció d'una nova web està encapçalada pel W3C.

WordNet. WordNet és una base de dades lèxica per l'idioma anglès. Agrupa les paraules en conjunts de sinònims anomenats *synsets* i enregistra les diverses relacions semàntiques entre aquests conjunts. El seu propòsit és permetre l'anàlisi automàtic de text i el desenvolupament d'aplicacions d'intel·ligència artificial.

XMI (*XML Metadata Interchange*). Es tracta d'un estàndard per l'intercanvi de metadades en format XML. L'ús més comú de l'XMI és com a format d'intercanvi per a models UML, tot i que es pot fer servir per a la serialització dels models de qualsevol altre llenguatge.

XML (*eXtensible Markup Language*). Nascut com a successor de l'HTML, aquest llenguatge permet separar l'estructura del document del seu contingut. Es basa en fitxers de text plans on utilitza etiquetes per separar els elements del document. A diferencia de l'HTML, l'XML defineix aquestes etiquetes en funció del tipus de dades que vol descriure i no de l'aparença final que tindran les dades en pantalla.

XML Schema. És una recomanació del W3C des del maig del 2001. Aquest llenguatge és una evolució dels DTDs. Permet descriure de manera formal el model de dades d'un document XML en termes de noms d'elements i d'atributs, relacions entre elements, estructura jeràrquica i tipus de dades.

XSL (*eXtensible Stylesheet Language*). És una família de llenguatges que permet descriure la manera de formatar o transformar els documents XML. La seva especificació forma part de les recomanacions del W3C.

Bibliografia

[1] Abarca F., *TFC: XML i Web Semàntica - Pla de Treball*, Març 2005

[2] W3C Recommendation, *Extensible Markup Language (XML) 1.0 (Third Edition)*, http://www.w3.org/TR/REC-xml/, 8 Març 2005

[3] W3C Consortium, *World Wide Web Consortium Homepage*, http://www.w3.org/, 7 Març 2005

[4] International Organization for Standarization, *Information Processing -- Text and Office Systems -- Standard Generalized Markup Language (SGML), ISO 8879:1986*, http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=16387, 8 Març 2005

[5] W3C Report, *Namespaces in XML*, http://www.w3.org/TR/REC-xml-names/, 8 Març 2005

[6] Dublin Core Metadata Initiative, http://dublincore.org/, 14 Abril 2005

[7] W3C Recommendation, *XML Schema Part 1: Structures Second Edition*, http://www.w3.org/TR/2004/REC-xmlschema-1-20041028/structures.html, 14 Març 2005

[8] W3C Recommendation, *XML Schema Part 2: Datatypes Second Edition*, http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/datatypes.html, 14 Març 2005

[9] XML.org, http://www.xml.org/, 8 Març 2005

[10] XML.com, http://www.xml.com/, 8 Març 2005

[11] W3C Architecture Domain, *XML Schema*, http://www.w3.org/XML/Schema#dev, 10 Març 2005

[12] ZapThink Foundation Report, The *Pros & Cons of XML*, http://www.zapthink.com/, 9 Març 2005

[13] Frank Manola, Eric Miller, eds, *RDF Primer*, http://www.w3.org/TR/rdf-primer/, W3C Recommendation 10 February 2004

[14] Graham Klyne, Jeremy Carroll, eds., *Resource Description Framework (RDF): Concepts and Abstract Syntax*, http://www.w3.org/TR/rdf-concepts/, W3C Recommendation 10 February 2004

[15] Patrick Hayes, ed., *RDF Semantics*, http://www.w3.org/TR/rdf-mt/, W3C Recommendation 10 February 2004

[16] Jan Grant, Dave Beckett, eds., *RDF Test Cases*, http://www.w3.org/TR/rdf-testcases/, W3C Recommendation 10 February 2004

[17] Web Naming and Addressing Overview (URIs, URLs, ...), http://www.w3.org/Addressing/, 22 Març 2005

[18] W3C Recommendation, *RDF/XML Syntax Specification*, http://www.w3.org/TR/rdf-syntax-grammar/, 21 Març 2005

[19] W3C Recommendation, *Resource Description Framework (RDF) Schema Specification 1.0*, http://www.w3.org/TR/2000/CR-rdf-schema-20000327/, 28 Març 2005

[20] Resource Description Framework (RDF) / W3C Semantic Web Activity, http://www.w3.org/RDF/, 29 Març 2005

[21] Dave Beckett´s Resource Description Framework (RDF) Resource Guide, http://www.ilrt.bristol.ac.uk/discovery/rdf/resources/, 30 Març 2005

[22] Thingy – Progos, http://progos.hu/thingy/, 30 Març 2005

[23] OpenRDF.org, http://www.openrdf.org/, 30 Març 2005

[24] IntelliDimension, http://www.intellidimension.com/default.rsp?topic=/pages/site/products/rdfgateway.rsp, 30 Març 2005

[25] Gruber, T.R., *A Traslation Approach to Portable Ontology Specifications,* 1993

[26] Gruber, T.R., *Toward principles for the design of ontologies used for knowledge sharing,* 1992

[27] George A. Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross and Katherine Miller, *Introduction to WordNet: An On-Line Lexical Database*, August 1993

[28] WordNet Homepage, http://wordnet.princeton.edu/, 25 Abril 2005

[29] George A. Miller, *Nouns in WordNet: A Lexical Inheritance System*, August 1993

[30] Christiane Fellbaum, *English Verbs as a Semantic Net*

[31] Richard Beckwith, George A. Miller and Randee Tengi, *Design and Implementation of the WordNet Lexical Database and Searching Software*

[32] WordNet Related Projects, http://wordnet.princeton.edu/links, 25 Abril 2005

[33] EuroWordNet, http://www.illc.uva.nl/EuroWordNet/, 29 Abril 2005

[34] The Global WordNet Association, http://www.globalwordnet.org/, 29 Abril 2005

[35] Web Ontology Language (OWL) / W3C Semantic Web Activity, http://www.w3.org/2004/OWL/, 18 Abril 2005

[36] Deborah L. McGuinness and Frank van Harmelen eds., *OWL Web Ontology Language Overview*, http://www.w3.org/TR/owl-features/, W3C Recommendation 10 Febrer 2004

[37] Smith, Welty, McGuinness, eds., *OWL Web Ontology Language Guide*, http://www.w3.org/TR/owl-guide/, W3C Recommendation 10 Febrer 2004

[38] Dean, Schreiber, eds., *OWL Web Ontology Language Reference*, http://www.w3.org/TR/owl-ref/, W3C Recommendation 10 Febrer 2004

[39] Patel-Schneider, Hayes, Horrocks, eds., *OWL Web Ontology Language Semantics and Abstract Syntax*, http://www.w3.org/TR/owl-semantics/, W3C Recommendation 10 Febrer 2004

[40] Jeremy J. Carroll, Jos De Roo, eds., *OWL Web Ontology Language Test Cases*, http://www.w3.org/TR/owl-test/, W3C Recommendation 10 Febrer 2004

[41] Jeff Heflin, ed., *OWL Web Ontology Language Use Cases and Requirements*, http://www.w3.org/TR/webont-req/, W3C Recommendation 10 Febrer 2004

[42] The Protégé Ontology Editor and Knowledge Acquisition System, http://protege.stanford.edu/, 2 Maig 2005

[43] Protégé OWL Plug-in, http://protege.stanford.edu/plugins/owl/, 8 Maig 2005

[44] T. Berners-Lee, J. Hendler, O. Lassila, *The Semantic Web*, Scientific American, Maig 2001

[45] W3C Semantic Web, http://www.w3.org/2001/sw/, 10 Maig 2005

[46] Marja-Riitta Koivunen and Eric Miller, *W3C Semantic Web Activity*, Proceedings of the Semantic Web Kick-off Seminar in Finland, http://www.w3.org/2001/12/semweb-fin/w3csw, Novembre 2001

[47] On-To-Knowledge Homepage, http://www.ontoknowledge.org, 16 Maig 2005

[48] OntoWeb Homepage, http://www.ontoweb.org, 16 Maig 2005

[49] RiboWeb Project, http://smi-web.stanford.edu/projects/helix/riboweb.html, 16 Maig 2005

[50] Resources fort the Semantic Web, http://www.semanticweb.org/resources.html, 16 Maig 2005

[51] Jambalaya | the CHISEL group, http://www.thechiselgroup.org/jambalaya, 24 Maig 2005

[52] CO-ODE / HyOntUse, http://www.co-ode.org/downloads/owlviz/co-ode-index.php, 24 Maig 2005

[53] RACER System Description, http://www.sts.tu-harburg.de/~r.f.moeller/racer/, 25 Maig 2005