Estudi de com OWL amplia la funcionalitat d'RDF i aquest, a la seva vegada, d'XML

Autor: **Ildefonso Molinero Gomez**
ETIG

Consultor: **Antoni Pérez Navarro**

18/06/2004

1. Resum

Estudi de com OWL amplia la funcionalitat d'RDF i aquest, a la seva vegada, d'XML

Internet és el mitjà d'accés a la informació més important que hi ha avui dia. Està a l'abast de gairebé tothom (principalment als països desenvolupats) i està en constant creixement i evolució. És precisament aquest creixement un dels principals factors que fa que hi hagi la necessitat d'una evolució per tal d'adaptar-se a les noves expectatives i necessitats. Aquesta evolució passa per fer d'Internet una eina més útil i fàcil d'usar per les persones.

Degut a les característiques de la web actual, la cerca d'informació, que és la principal utilitat d'Internet, és cada vegada més difícil i sobre tot és cada vegada més difícil trobar allò que realment es vol trobar. Qualsevol usuari assidu d'Internet s'ha pogut veure alguna vegada frustrat en voler trobar alguna informació concreta amb qualsevol dels motors de cerca actuals que hi ha a l'abast a Internet els quals molt sovint no li ofereixen els resultats previstos.

Aquest document explica com és aquesta web actual i perquè no en treu millor profit de la ingent quantitat d'informació que té. Aquesta web es anomenada dins del document com **web actual** o **sintàctica** que està basada en el llenguatge HTML. Es podria dir que és una *web caracteritzada per tenir una quantitat immensa d'informació desorganitzada i desclassificada difícil d'assimilar per les persones*.

També s'explica en detall quina és l'alternativa que la comunitat informàtica proposa per tal de superar aquests inconvenients; aquesta alternativa és un concepte anomenat **web semàntica** basat en altres conceptes com ara l'**ontologia** i les **metadades** i llenguatges com l'**XML**, **RDF** i **OWL**, aquests dos últims en constant evolució i que són estàndards (a l'igual que XML) recentment aprovats pel consorci W3C (www.w3.org) el qual és el referent dins de la comunitat informàtica mundial en quant a l'ús de tecnologies per la web. *La web semàntica és una extensió de la web actual però on la informació té un significat ben definit que millora el treball cooperatiu entre persones i computadors*.

Per tal d'explicar la web semàntica es fa un repàs dels conceptes que el componen:

- El llenguatge **XML** per descriure l'estructura de les dades contingudes a les pàgines web.
- El llenguatge **RDF** per a definir les dades d'una manera estructurada i definir la semàntica.
- L'**ontologia** per a descriure els conceptes del domini.
- La **capa lògica** per a relacionar i processar la informació.
- **OWL** com a llenguatge per definir ontologies.

Eric Miller, líder de l'activitat de *web semàntica* del W3C va explicar, "*En realitat la web semàntica és més una evolució que una revolució web. Es construeix a través de canvis incrementals, duent als documents i dades ja disponibles en la web descripcions llegibles per màquines. XML, RDF i OWL permeten a la web ser una infrastructura global per a la compartició de dades i documents, el que també fa més fàcil i fiable la recerca i reutilització d'informació.*"

Aquest document s'ha fet com a treball final de carrera d'Enginyeria Informàtica de Gestió de la Universitat Oberta de Catalunya i té com a propòsit el ser útil com a guia introductòria dels conceptes de la web semàntica que de bon segur donaran molt que parlar en un futur proper. Tots els conceptes explicats al document mostren exemples d'ús per tal de facilitar la comprensió.

Índex

2. Introducció

El principal objectiu d'aquest document és explicar la **web semàntica** i els elements que la componen, com són **XML**, **RDF** i **OWL**. En primer lloc s'explicarà el concepte d'XML (*eXtensible Markup Language*), també les seves característiques, àrea d'aplicació, avantatges respecte **HTML**, mancances i també s'explicarà la seva estructura i es veuran exemples que poden ajudar a entendre'l millor.

Es veurà també una ampliació de l'XML com ara l'**XML Schema** i es mostraran diferents maneres com els documents XML poden ser visualitzats.

El següent punt important que es tractarà és **RDF** (*Resource Description Framework*). Abans d'entrar en detall però, es repassaran altres conceptes necessaris per entendre RDF, **metadades** i **ontologies**. Serà en aquest punt quan s'entrarà en detall a veure els conceptes de **web actual o sintàctica** i **web semàntica**.

Es veuran els inconvenients que presenta la web actual i que s'està fent per tal d'evolucionar cap a la web semàntica.

Una vegada clars aquests conceptes es tractarà l'RDF en més profunditat, tot veient els conceptes fonamentals, les característiques principals, el model de dades, la sintaxi bàsica, etc. I també es veuran exemples per reforçar l'entesa. Al final d'aquest punt es farà una reflexió sobre les aportacions d'RDF i els inconvenients que presenta.

Finalment es veurà **OWL** (Ontologic Web Language) el qual es podria dir que és una extensió d'RDF i es veuran els seus tres subllenguatges: OWL Full, OWL DL i OWL Lite.

3. XML

En aquest apartat es veurà el *eXtensible Markup Language* (**XML**) i les tecnologies i conceptes que hi estan relacionats. Abans d'entrar al detall del que és XML, s'explicaran una sèrie de conceptes que ajudaran a donar la base per entendre-ho millor.

En primer lloc es farà una **introducció** (veure apartat 3.1) per continuar amb una visió de l'**origen, motivació i objectius de l'XML** (apartat 3.2) que ajudarà a endinsar-se en detalls un a mica més tècnics a **característiques principals** (apartat 3.3) on s'explicarà tot un seguit de conceptes bàsics per tal d'entendre l'XML.

A continuació es veurà l'**estructura d'un document XML** (apartat 3.4) on s'explicarà que hi ha dos metallenguatges (llenguatges que serveixen per a definir altres llenguatges) amb els que definir els llenguatges que es poden obtenir a partir d'XML, aquests són **DTD** (apartat 3.4.1) i **XML Schema** (apartat 3.4.2).

Seguidament es farà un repàs de les diferents maneres com es poden **presentar els documents XML** (apartat 3.5).

3.1. Introducció

XML és un estàndard per a l'intercanvi de dades a la *web* i per a l'intercanvi d'informació estructurada entre diferents plataformes. És un metallenguatge que permet definir llenguatges de marcat adequat a usos determinats.

Els elements que el composen poden donar informació sobre el que contenen, no necessàriament sobre la seva estructura física o presentació. **XML descriu el contingut del que etiqueta**.

Està estructurat seguint unes regles, la qual cosa permet, per exemple, realitzar motors de cerca més eficients que permeten un accés més ràpid a les dades. També l'intercanvi d'informació i cooperació entre les empreses, simplificant el comerç electrònic, l'intercanvi de dades, etc.

El principal avantatge d'XML és que s'ha convertit en un estàndard que la majoria dels fabricants de programari han acceptat i ara per ara serveix per a la majoria de les plataformes actuals, sistemes operatius, navegadors, llenguatges de programació, bases de dades, etc.

Abans de l'aparició d'XML, si una empresa volia intercanviar dades entre departaments que utilitzaven diferents plataformes, diferents sistemes operatius o simplement diferents programes, havia de comprar o fer algun programari *middelware* que fes la conversió (que fes de pont entre els dos sistemes 'diferents' per tal que s'entenguessin), amb el cost en temps i diners que això suposa.

L'XML sembla que ha estat abastament acceptat per la majoria de fabricants de programari com un format d'intercanvi de dades, la qual cosa ha fet que sigui un llenguatge que la majoria del programari actual pot entendre.

Posant un símil, és com l'ús de l'anglès als negocis. Aquest s'ha establert com un estàndard per compartir informació. Hi ha molts llenguatges arreu del món, i cap no és millor que un altre, però només un s'ha establert com a comú per a tots els negocis. Una cosa semblant passa amb l'XML, hi ha molts llenguatges que serveixen per emmagatzemar, gestionar i intercanviar dades, però sembla que XML es consolida com un estàndard per a fer-ho.

XML però, no ho fa tot i en cap cas no es pot veure com un substitut dels SGBD (Sistemes de Bases de Dades Relacionals) actuals[1], els quals incorporen seguretat en l'accés a les dades i potents sistemes de control i gestió de les dades que emmagatzemen, com ara: el control d'usuaris, els *stored procedures*, *triggers*, sistemes de *logs*, sistemes integrats de còpies de seguretat i d'importació/exportació de dades, optimització de la velocitat d'accés a les dades mitjançant índexs, capacitat de crear *constraints*, etc.

Per contra, el fet de no tenir tots aquests sistemes de control i gestió de dades fa que XML sigui molt lleuger, tan lleuger que de fet no és més que codi ASCII estructurat i amb unes regles.

3.2. Origen, motivació i objectius d'XML

XML neix el Febrer de 1998 amb la seva versió 1.0 com un subconjunt d'**SGML**[2]. Neix quan **HTML**[3] ja portava temps funcionant a la *web* amb èxit, però amb poca flexibilitat.

L'HTML és un llenguatge de marcat (llenguatge basat en etiquetes) que serveix per codificar documents i distribuir-los per la ***World Wide Web***. És un llenguatge que és interpretat pels navegadors[4], els quals mostren el seu contingut.

Una de les mancances de l'HTML és que la informació que codifica fa referència només a la seva presentació, no al seu contingut; és a dir, un document HTML, que és interpretat seqüencialment de dalt a baix, només va indicant a cada línia de codi, què es visualitzarà i com (color, estil, mida, etc.).

[1] XML no pretén ser un substitut de les bases de dades, però si pot ser un mitjà d'intercanvi de dades entre diferents bases de dades.
[2] *Standard Generalizaed Markup Language*. Metallenguatge per definir llenguatges de marcat, predecesor d'XML.
[3] *HyperText Markup Language*. Llenguatge de marcat, estandarditzat pel W3C, utilitzat per la creació de documents que es volen publicar a la Web.
[4] Programari utilitzat per accedir i navegar per Internet.

Vegem-ho amb un exemple. Imaginem una *web* de venda de llibres. El potencial d'aquesta *web* és el seu catàleg de llibres, el qual es vol mostrar a través d'Internet per tal que els seus clients puguin triar el producte i fer la seva compra.

El programari que permetrà a l'empresa mostrar els seus productes a la *web* és el conegut HTML, es a dir, a través de pàgines HTML, es mostraran els productes (llibres) de l'empresa propietària. Ara bé, si aquesta empresa vol flexibilitat pel que fa a la manera de presentar els productes, la renovació dels mateixos, mostrar l'existència d'*stocks*, etc., no tindrà més remei que modificar totes les pàgines HTML per tal de reflectir aquests canvis.

Aquest fet mostra que treballar només amb HTML pot resultar poc pràctic i molt costós. Això passa perquè HTML tracta la informació (els llibres) de la mateixa manera que tracta la seva presentació, no en fa distinció, el llibre no és més que una altra línia de codi HTML.

Per contrarestar aquestes mancances existeixen les aplicacions **CGI**, que són programes compilats que s'executen als servidors d'aplicacions[5] *web* i que tenen accés a les bases de dades (justament on hi hauria el catàleg de llibres). Aquestes aplicacions generen codi HTML dinàmicament[6] per tal que el pugui consultar el client final des del navegador del seu ordinador.

Amb l'ús d'aquestes aplicacions de servidor citades anteriorment, es pot veure com una de les mancances principals de l'HTML, el tractament de les dades, queda solucionada, ja que aquests programes tenen accés a les bases de dades.

Però ara el problema és un altre, resulta que el complex món d'Internet intercomunica ordinadors de tot el món sense tenir en compte quin sistema operatiu s'està utilitzant. Aquest fet fa veure que per la banda dels servidors d'aplicacions *web* hi ha diferents plataformes, les més importants i predominants són Unix i Microsoft Windows i per la part dels clients hi ha una varietat molt més gran encara, sent però també predominants els mateixos que abans.

Però no només els sistemes operatius són diferents a la part dels servidors d'aplicacions, també ho són les bases de dades que aquests utilitzen, Microsoft SQL Server, Oracle, IBM DB2, MySQL, Informix**,** etc. I a la part del client la cosa no és millor, ja que els clients poden utilitzar diferents navegadors per accedir a Internet, com per exemple, Microsoft Explorer, Netscape, Mozilla, etc.

Aleshores va sorgir una proposta d'estàndard multiplataforma anomenat XML, que amb el temps ha estat àmpliament acceptat i ara per ara la majoria de sistemes operatius, navegadors i bases de dades el saben interpretar.

Una de les claus d'aquesta acceptació potser ha estat la seva senzillesa: No és més que codi ASCII que es pot editar amb qualsevol editor de textos. Evidentment també ha jugat un paper molt important el fet d'estar avalat pel consorci **W3C**[7], que s'ha convertit en un referents d'estàndards per a Internet.

XML es pot utilitzar com a format d'intercanvi de dades entre diferents plataformes, bases de dades, etc., o també es pot utilitzar per ser visualitzat directament a Internet.

3.3. Característiques principals

En aquest punt es veuran en detall les parts que componen un document XML, tot explicant-les. Es veuran també les regles que aquests han de complir per tal de ser **vàlids** i **ben formats** i també com aquests documents són interpretats i qui els interpreta. També es podrà comprovar com amb **DTD** i **XML-Schemes** es poden definir estructures pels documents XML.

[5] Els servidors d'aplicacions són els servidors on hi ha instal·lades les aplicacions informàtiques que consulten els clients des dels ordinadors personals majoritariament.
[6] També existeixen les versions d'aplicacions de servidor amb llenguatge interpretat com ASP, PHP, etc.
[7] Word Wide Web Consortium. www.w3.org.

3.3.1. Conceptes bàsics

Els documents XML són fitxers de text que contenen un conjunt d'elements organitzats en forma jeràrquica: existeix un **element** que és l'**arrel** i que pot tenir elements que al seu torn poden tenir altres elements.

Els elements XML han de tenir un **nom** i poden tenir **atributs** i **contingut**. Per a inserir un element dins del document s'utilitzen **etiquetes** o **tags** : una etiqueta es forma tancant el nom de l'element entre els caràcters < i >**,** per a indicar el final d'un element s'utilitza el caràcter **/.**

```
<ciutat/>                                  element XML sense contingut ( buit )

<ciutat>Lisboa</ciutat>                    element XML amb contingut

<ciutat codi="101">Lisboa</ciutat>         element XML amb contingut i un atribut

<ciutats>                                  element XML que conté altres elements
   <ciutat codi="101">Lisboa</ciutat>
   <ciutat codi="102">Paris</ciutat>
   <ciutat codi="103">Roma</ciutat>
</ciutats>
```

Exemple 1.

Al llarg del document es podran veure molts exemples de codi XML, en els quals es podrà apreciar que els noms dels elements són de tota mena, aquests noms són només un exemple qualsevol, ja que els noms dels **elements** i dels **atributs** a dins dels documents XML els defineix l'usuari.

3.3.2. Documents XML ben formats i vàlids

Els documents ben formats (*well-formed*) són els que segueixen les regles de l'XML al peu de la lletra, és a dir, la sintaxis del document ha de seguir l'estàndard del consorci W3C i no han de tenir necessàriament un DTD (veure apartat 3.4.1) o un XML-Schema associat (veure apartat 3.4.2).

En un DTD o un XML-Schema definim com serà el document; és a dir, definim els elements i atributs i com s'estructuren i relacionen. Això vol dir que, si en l'elaboració dels nostres documents XML no n'utilitzem cap, el *parser* (veure apartat 3.3.5) no pot proporcionar informació sobre la validesa del document; és a dir, no ens pot indicar que els elements i atributs que utilitzem són els correctes i que es troben en l'ordre adient, sinó que, simplement indicarà si el document està ben format o no; és a dir, si respecta les regles sintàctiques del llenguatge XML.

Un document XML està **ben format** si:

- Compleix la regla anomenada "document" (veure apartat 3.3.2.1).
- Respecta totes les restriccions de bona formació donades a l'especificació.
- Cadascuna de les entitats analitzades que es referència directa o indirectament en el document està ben formada.

Un document serà vàlid si està ben format i segueix la definició d'un DTD o d'un XML-Schema.

3.3.2.1 La regla "document"

Complir la regla "document" significa:

1. Que el document XML conté un o més elements.
2. L'element arrel no apareix en el contingut de cap altre element.
3. Si l'etiqueta de començament d'un element està en el contingut d'algun altre element, l'etiqueta de fi està en el contingut del mateix element.

El següent exemple ***no*** *és un document XML ben format* ja que no conté cap element, per la qual cosa, no compleix la regla número 1.

```
El meu primer document XML
```

Exemple 2.

El següent exemple ***sí*** *que és un document XML ben format*, per tenir almenys l'element "p".

```
<p>El meu primer document XML</p>
```

Exemple 3.

L'exemple següent ***no*** *és un document XML ben format* per incomplir la regla número 2, segons la qual només pot existir un únic element arrel.

```
<p>El meu primer document XML</p>
<p>El meu primer document XML</p>
```

Exemple 4.

L'exemple següent però, ***sí*** *que és un document XML ben format* en convertir-se l'element "document" en l'element arrel, ser únic i no formar part del contingut de cap altre element.

```
<document>
  <p>El meu primer document XML</p>
  <p>El meu primer document XML</p>
</document>
```

Exemple 5.

En canvi, el següent exemple ***no*** *és un document XML ben format* per incomplir la regla 3, ja que l'etiqueta d'inici de l'element "ressaltar" està dins del contingut de l'element "p", però la seva etiqueta final està fora.

```
<document>
  <p>El meu primer <ressaltar>document XML</p></ressaltar>
  <p>El meu primer document XML</p>
</document>
```

Exemple 6.

A continuació podem veure l'exemple anterior corregit:

```
<document>
  <p>El meu primer <ressaltar>document XML</ressaltar></p>
  <p>El meu primer document XML</p>
</document>
```

Exemple 7.

3.3.3. Sintaxi correcta i restriccions de bona formació

A l'hora de crear documents XML s'ha de conèixer la sintaxi de com s'escriuen els **elements**, **atributs** i **entitats**, i si s'incompleix, el *parser* donarà un error de mala formació.

A continuació és pot veure un exemple amb 4 errors:

```
<?xml version="1.0"?>
<document>
  <p>El meu primer<ressaltar importancia=1>document XML</ressaltar></p>
  <p>Comença amb la etiqueta <document&gt;</p>
  <p>A continuació posem un element sense contingut</p>
  <imatge fitxer="imatge.gif">
</document>
```

Exemple 8.

1. El valor de l'atribut "importancia", no està entre cometes. En XML és obligatori utilitzar-les.

   La manera correcta seria:

   ```
   ...<ressaltar importancia="1">document XML</ressaltar>...
   ```

   Exemple 9.

2. L'etiqueta final, de l'element "p" està malament tancada. En comptes del caràcter "]" , hauria de tenir el símbol major que ">".

   ```
   <p>El meu primer <ressaltar importancia=1>documento XML</ressaltar></p>
   ```

   Exemple 10.

3. S'està utilitzant el símbol menor que "<" sense que formi part de la definició d'una etiqueta. En ser un caràcter reservat, s'hauria d'escriure com a l'entitat predefinida <.

   ```
   <p>Comença amb la etiqueta &lt;document&gt;</p>
   ```

   Exemple 11.

4. L'element buit "imatge" està escrit de forma incorrecta. En ser un element sense contingut hauria d'estar escrit amb una etiqueta d'element buit:

```
<imatge fitxer="imatge.gif"/>
```

Exemple 12.

o també de la següent manera:

```
<imatge fitxer="imatge.gif"></imatge>
```

Exemple 13.

A continuació es pot veure l'exemple anterior escrit de manera correcta:

```
<?xml version="1.0"?>
<document>
  <p>El meu primer <ressaltar importancia="1">document XML</ressaltar></p>
  <p>Comença amb la etiqueta &lt;document&gt;</p>
  <p>A continuació posem un element sense contingut</p>
  <imatge fitxer="imatge.gif"/>
</document>
```

Exemple 14.

3.3.4. Altres regles

- L'XML és sensible a la utilització de majúscules i minúscules.

Al següent exemple els elements "p" i "P" són diferents.

```
<p>El meu primer document XML</p>
<P>El meu primer document XML</P>
```

Exemple 15.

- El nom de la etiqueta d'inici i final ha de ser el mateix.

El següent exemple és incorrecte ja que en fer diferència entre majúscules i minúscules, el *parser* no entén ambdues etiquetes com del mateix element..

```
<p>El meu primer document XML</P>
```

Exemple 16.

- Cap nom d'atribut pot aparèixer més d'una vegada a la mateixa etiqueta d'inici o d'element buit.

El següent exemple és incorrecte perquè l'atribut *importancia* apareix 2 vegades dins la mateixa etiqueta:

```
<ressaltar importancia="1" importancia="2">
  document XML
</ressaltar>...
```

Exemple 17.

3.3.5. Parsers XML

Per a llegir, actualitzar, crear i manipular un document XML, es necessita un *parser* d'XML. Hi ha *parsers* per a tots els llenguatges i plataformes, Microsoft Windows, el sistema operatiu més conegut, proporciona un *parser* d'XML amb el navegador *Internet Explorer*.

El *parser* d'XML és l'eina principal de qualsevol aplicació XML. A través d'ell es pot comprovar si els documents estan **ben formats** i són **vàlids**. Microsoft ofereix el seu *parser* a la seva llibreria MSXML3.DLL que instal·la *Microsoft Explorer*. Aquesta biblioteca té el *parser* XML, exposa el **DOM**[8] per a la manipulació del document XML i té el motor d'**XSL** (veure apartat 3.5.4).

Actualment hi ha molts *parsers* i per a tots els llenguatges i plataformes: Java, C, Phyton, Visual Basic, Perl, Tcl, Delphi, etc., tot i que els escrits en **Java** són majoria. L'XML contribueix amb dades independents de la plataforma (documents i dades portables).

3.3.6. XML i els navegadors

XML és un llenguatge capaç de descriure el contingut de la pàgina, no el seu aspecte. I com un navegador pot mostrar alguna cosa de la qual no sap el seu aspecte?.

La idea és associar al document XML un full d'estil. Es pot fer amb **CSS** (veure apartat 3.5.3) o bé amb una solució més completa: **XSL** (veure apartat 3.5.4) que permet afegir lògica de processament al full d'estil, en forma d'una mena de llenguatge de programació.

D'aquesta manera, depenent del full d'estil associat al document o de la lògica emprada, es podrà visualitzar en qualsevol plataforma: PalmPC, PC, microones, nevera, navegador textual, Microsoft Internet Explorer, Netscape, etc., i amb l'aspecte (colors, fonts, etc.) que es vulgui.

3.3.7. Parts d'un document XML

Els documents XML han de començar amb una línia de declaració XML, després un **pròleg XML** on s'especifiquen alguns aspectes sobre el contingut i l'estructura del document com el **tipus de document** o **DOCTYPE**, a continuació del pròleg es troben els **elements XML** començant per l'element **arrel** que conté la declaració de l'**espai de noms** o ***namespace*** (***xmlns***); veure apartat 3.4.3.

```
                                                        Pròleg XML
<?xml version="1.0" encoding="iso-8859-1"?>               Declaració XML

<!DOCTYPE html                                               Tipus de document
 system "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">     ( DTD )

                                                          Elements
 <html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es">   Element arrel
   <head>
     <title>Títol del document</title>
   </head>
   <body>
     <p>Un paràgraf</p>
   </body>
 </html>
```

Exemple 18.

[8] Document Object Model és el model d'objectes que dóna accés als programadors a l'interior del document per a poder-lo manipul·lar.

A continuació s'expliquen les parts principals que componen un document XML.

3.3.7.1 Pròleg

Els documents XML comencen amb un pròleg, en el que bàsicament es defineix una declaració XML.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

Exemple 19.

- **Version**: Indica la versió d'XML del document.
- **Encoding**: Indica quina codificació s'utilitzarà per interpretar els caràcters del document XML. Per exemple, si una persona que vol escriure text en català no indica la codificació adient, el parser d'XML donarà error en veure dins del document la lletra **ç**, ja que no la reconeix com part del seu llenguatge.

A la declaració anterior es pot veure informació sobre la versió d'XML utilitzat, en el cas de l'exemple la 1.0. Actualment s'està treballant a la versió 1.1.

També es pot veure informació sobre el tipus de codificació de caràcters utilitzat. Nosaltres utilitzarem *ISO-8859-1* que correspon a la codificació llatina.

3.3.7.2 Elements

Tot document XML es composa d'un o més **elements.** Cada element està delimitats per **etiquetes** de començament i etiquetes de fi en el cas que tinguin contingut:

```
<p>El meu primer <ressaltar importancia="1">document XML</ressaltar></p>
```

Exemple 20.

i per una etiqueta d'**element buit** en el cas de ser elements sense contingut:

```
<imatge fitxer="imatge.gif"/>
```

Exemple 21.

Cada element pot tenir dades de caràcter, altres elements, totes dues coses a l'hora o pot ser que estiguin buits.

3.3.7.3 Atributs

Els elements XML poden tenir **atributs** (propietats) que ofereixen informació sobre l'element. A continuació es pot veure un exemple d'elements amb atributs.

```
<p>El meu primer <ressaltar importancia="1">document XML</ressaltar></p>
 .....
<imatge fitxer="imatge.gif"/>
```

Exemple 22.

A l'exemple 22 podem veure com:

- L'element, "ressaltar" va associat amb l'atribut "importancia", que indicarà el grau de rellevància del seu contingut[9].
- L'element "imatge" va associat amb l'atribut "fitxer", on indicarem l'arxiu que conté la imatge.

Els valors dels atributs han d'estar delimitats amb cometes dobles (") o cometa simple ('). Quan s'utilitza un per a delimitar el valor de l'atribut, l'altre es pot utilitzar a dins.

```
<noticia titol="Jesulín se separa" autor='Pepe "Porras"'>
....
</noticia>
```

Exemple 23.

A vegades, un element amb contingut pot modelar-se com un element buit amb atributs.

A continuació es podran veure tres maneres diferents de codificar la mateixa informació.

- En el primer cas utilitzem 3 **etiquetes** diferents: *menjar*, *nom* i *tipus* i 2 **continguts**: *llenties* i *llegums*.
- En el segon cas utilitzem 1 sola **etiqueta**: *menjar*, 1 **atribut**: *tipus* i un **contingut**: *llenties*.
- En el tercer cas utilitzem 1 sola **etiqueta** : *menjar* i dos **atributs**: *tipus* i *nom*.

```
<menjar><nom>llenties</nom><tipus>llegums</tipus></menjar>
....
<menjar tipus="llegums">llenties</menjar>
....
<menjar tipus="llegums" nom="llenties"/>
....
```

Exemple 24.

3.3.7.4 Comentaris

Poden aparèixer en qualsevol punt del document, fins i tot fora de la resta de les marques, és a dir, fora de les declaracions d'etiquetes o d'altres comentaris. Comencen amb "<!--" i acaben amb "-->" . La cadena "--" no pot aparèixer dins d'un comentari. Els comentaris no són interpretats pels *parsers* (veure apartat 3.3.5), però es poden utilitzar per afegir text lliure per tal de fer anotacions o explicacions de qualsevol mena. A continuació es pot veure un exemple de comentari.

[9] Aquests noms són només un exemple, ja que els noms dels elements i dels atributs a dins dels documents XML els defineix l'usuari.

```
<!-- això és un comentari -->
```

Exemple 25.

3.3.7.5 Entitats predefinides

En XML 1.0 es defineixen cinc entitats per a representar caràcters especials i que no s'interpretin com a marques pel *parser* XML. Així podem utilitzar per exemple, el caràcter "<" sense que s'interpreti com començament d'una etiqueta XML.

Entitat	Caràcter
&amp;	&
&lt;	<
&gt;	>
&apos;	'
&quot;	"

```
<exemple>
  &lt;HTML&gt;
    &lt;HEAD&gt;
      &lt;TITLE&gt;Rock &amp; Roll&lt;/TITLE&gt;
    &lt;/HEAD&gt;
    ...
</exemple>
```

Exemple 26.

3.3.7.6 Seccions CDATA

Les seccions CDATA també permeten especificar dades, utilitzant qualsevol caràcter, especial o no, sense que s'interpreti com una marca XML. La raó és que així es pot llegir més fàcilment el document XML sense haver de desxifrar els codis de les entitats anteriorment esmentades. Les seccions CDATA comencen per la cadena "<![CDATA" i acaben amb la cadena "]]>" i només aquesta darrera es reconeix com a marca.

```
<exemple>
   <![CDATA[
     <HTML>
       <HEAD>
        <TITLE>Rock & Roll</TITLE>
       </HEAD>
...
   ]]>
</exemple>
```

Exemple 27.

3.3.7.7 Instruccions de processament

Les instruccions de processament van entre **<?** i **?>** i permeten als documents XML tenir instruccions per les aplicacions.

```
<?xml version="1.0"?>
<?xml-stylesheet href="enquesta.xsl" type="text/xsl"?>
....
```

Exemple 28.

La instrucció de processament anterior indica que el document XML utilitzarà la plantilla XSL[10] *enquesta.xsl* per ser visualitzat.

3.4. Estructura d'un document XML

Un document XML pot contenir molts tipus d'informació. Es a dir, poden haver molts llenguatges escrits en XML per qualsevol col·lectiu de usuaris. Per exemple,

- Si l'utilitza el *col·lectiu de metges* es podria crear un llenguatge en XML específic per emmagatzemar diagnòstics dels pacients. Aquest llenguatge es podria dir *PacientsML*.
- Si els *distribuïdors de pel·lícules* utilitzen XML podran crear els seus propis llenguatges per guardar la informació de les pel·lícules. Aquest llenguatge es podria dir *Pel·liculesML*.
- Si s'utilitza per escriure *aplicacions per a mòbils* es pot utilitzar un llenguatge per aplicacions inalámbriques (*Wireless*), que es digui *WML*.

Com es pot veure, es poden crear infinits llenguatges a partir de l'XML. Per a especificar cadascun dels usos d'XML, o el que es el mateix, per a especificar cadascun dels subllenguatges que es poden crear a partir d'XML, s'utilitzen uns llenguatges propis.

Hi ha dos **metallenguatges** (llenguatges que serveixen per a definir altres llenguatges) amb els que definir els llenguatges que es poden obtenir a partir d'XML, són **DTD** i **XML Schema**.

Aquests metallenguatges es defineixen especificant quines etiquetes es poden trobar o han de trobar-se als documents XML, en quin ordre, dins de quines altres, a més d'especificar els atributs que poden tenir o han de tenir cadascuna de les etiquetes.

Per exemple, si dos departaments d'una empresa s'han d'intercanviar informació en format XML, poden definir un DTD o un XML-Schema de manera que tots els documents XML que utilitzen per a intercanviar dades segueixin les regles indicades en aquests documents (nom dels elements, tipus, etc.). Així XML està més estructurat i es pot utilitzar com a base de dades.

Un detall important d'assenyalar a l'hora de parlar dels **DTD** o **XML Schema** és que aquests llenguatges també **permeten comprovar la integritat de les dades** en qualsevol moment.

Una gran part de les línies de codi que escriu un programador estan orientades a comprovar la integritat de les dades, és a dir, comprovar si on es suposa que ha d'haver un número efectivament hi ha un número, si el número és enter o qualsevol altre comprovació. Aquests metallenguatges d'XML serveixen per comprovar que les dades que un document XML conté són vàlides, comprovant si el que hi ha a l'XML concorda amb el que hauria d'haver.

[10] XLS és veurà a l'apartat 3.5.4

3.4.1. DTD

Document Type Definition defineix com és l'estructura dels blocs d'un document XML, és a dir, els elements que formaran el document i com aquests estan relacionats. Un DTD conté una llista dels elements vàlids i especifica el seu tipus.

DTD té una sintaxi especial, diferent a l'XML. DTD es pot incloure en el document XML o fer una referència des d'ell mateix a un fitxer extern.

3.4.1.1 DTD en el document XML

Es pot incloure la DTD en el propi document XML utilitzant la definició:

```
<!DOCTYPE root-element [element-declarations]>
```

Exemple 29.

Per exemple:

```
<?xml version="1.0"?>
  <!DOCTYPE missatge [
    <!ELEMENT missatge (origen,desti,text)>
    <!ELEMENT origen (#CDATA)>
    <!ELEMENT desti  (#CDATA)>
    <!ELEMENT text   (#CDATA)>
  ]>
  <missatge>
    <origen>El poble</origen>
    <desti>Governants</desti>
    <text>S'ha de destinar més diners al tercer mon</text>
  </missatge>
```

Exemple 30.

- **!DOCTYPE missatge** Indica que aquest document és del tipus "missatge".
- **!ELEMENT missatge** Indica que els nodes <missatge> poden tenir tres elements: origen, desti i text.
- **!ELEMENT origen** (així com desti i text) indica que són elements del tipus "#CDATA".

3.4.1.2 Referència a una DTD externa

La referència es fa a la definició del tipus de document:

```
<!DOCTYPE root-element SYSTEM "filename">
```

Exemple 31.

L'exemple anterior constaria de dos fitxers, el document XML i el DTD.

El contingut de *missatges.xml* serà:

```
<?xml version="1.0"?>
  <!DOCTYPE missatge SYSTEM "missatge.dtd">
  <missatge>
    <origen>El poble</origen>
    <desti>Governants</desti>
    <text>S'ha de destinar més diners al tercer mon</text>
  </missatge>
```

Exemple 32.

El contingut de *missatge.dtd* serà:

```
  <!ELEMENT missatge (origen,desti,text)>
  <!ELEMENT origen  (#CDATA)>
  <!ELEMENT desti   (#CDATA)>
  <!ELEMENT text    (#CDATA)>
```

Exemple 33.

3.4.2. XML-Schema

Atès que el DTD (veure apartat 3.4.1) té una sintaxi molt especial, es va intentar trobar una manera d'escriure en XML la definició d'un altre llenguatge XML. Aleshores es va definir el llenguatge XML-Schema. **XML-Schema a l'igual que DTD descriu l'estructura d'un document XML**.

Originalment XML-Schema va ser una proposta de Microsoft, però al Maig del 2001 va ser una recomanació oficial de W3C. Actualment els DTD estan sent substituïts pels XML-Schema.

3.4.2.1 Característiques

A continuació s'enumeren algunes característiques dels XML-Schema:

- Defineixen els elements que poden aparèixer a un document.
- Defineixen els atributs que poden aparèixer a un document.
- Defineixen quins elements són elements fills.
- Defineixen l'ordre dels elements fills.
- Defineixen el nombre d'elements fills.
- Defineixen si un element és buit o put incloure text.
- Defineixen el tipus de dades per elements i atributs.

Amb XML no s'estableix cap regla sobre el contingut dels documents però amb XML-Schema es posen unes restriccions, establint unes regles i seguint unes pautes sobre el contingut dels documents XML. **Amb XML-Schema es dóna un pas que acosta una mica més al que seria la lògica que utilitzen les bases de dades**.

XML-Schema, en utilitzar sintaxi XML, ja que té l'estructura d'un document XML, té com a avantatges que:

- No s'ha d'aprendre cap altre llenguatge, només XML.
- Es poden emprar editors XML per editar els XML-Schema.
- Es poden emprar *parsers* XML per analitzar els XML-Schema.
- Es pot manipular l'XML-Schema amb el DOM d'XML.
- Els XML-Schema es poden transformar amb XSLT.

3.4.2.2 Aportacions d'XML-Schema a XML

Ja s'ha comentat que XML s'havia establert com un estàndard multiplataforma que es podia utilitzar si es volia guanyar en compatibilitat, ara bé, per tal de poder posar un ordre en la comunicació entre un emissor i un receptor que utilitzen XML s'han d'establir unes regles que hauran de complir tant l'emissor com el receptor. Aquestes regles són les que s'estableixen a l'XML-Schema.

A continuació es poden veure uns exemples en els quals es pot comparar un document XML i el seu equivalent XML-Schema.

El document XML és el següent:

```
<?xml version="1.0"?>
<nota>
<receptor>Antoni</receptor>
<emissor>Jose</emissor>
<tema>Noticia</tema>
<missatge>El Barça a guanyat</missatge>
</nota>
```

Exemple 34.

Al document XML-Schema corresponent a l'XML anterior es defineixen els elements:

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.w3schools.com"
xmlns="http://www.w3schools.com"
elementFormDefault="qualified">
<xs:element name="nota">
  <xs:complexType>
    <xs:sequence>
    <xs:element name="emissor"  type="xs:string"/>
    <xs:element name="receptor" type="xs:string"/>
    <xs:element name="tema"     type="xs:string"/>
    <xs:element name="missatge" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Exemple 35.

L'element **nota** és de *tipus complex* perquè conté altres elements. Els altres elements (**emissor**, **receptor**, **tema**, **missatge**) són de *tipus simple* perquè no contenen altres elements. Els elements simples anteriors s'han definit de tipus cadena (*string*).

3.4.3. XML namespaces

XML deixa oberta l'opció de creació d'etiquetes pròpies per part de l'usuari, això vol dir que si tothom es dediqués a anar definint etiquetes, un document acabaria sent un caos de diferents etiquetes procedents de diferents llocs i el que és pitjor, d'etiquetes amb el mateix nom que, en realitat, signifiquen coses diferents.

El concepte d'espais de noms (*namespaces*) permet particionar el conjunt de tots els noms possibles, de forma que es pugui definir a quina zona d'aquell espai correspon una etiqueta. D'aquesta forma, **etiquetes amb el mateix nom, però definides per dos autors diferents, poden diferenciar-se en l'espai de noms**.

L'espai de noms no és essencial en tots els documents, però resulta útil quan s'usen etiquetes que venen de diferents procedències (per exemple, etiquetes noves dins d'un document XML), o etiquetes que es volen processar de forma diferent. L'espai de noms d'una etiqueta s'indica amb <prefix**:**namespace:etiqueta>. Per exemple, s'usen espais de noms en el document següent:

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<!--  Descripció dels elements d'una casa somiada -->

<mc:micasa xmlns:mc="http://www.miweb.org/micasa">
  <mc:habitacio id="menjador">
    <mc:moble>aparador</mc:moble>
    <mc:moble>cadira "d'època"</mc:moble>
  </mc:habitacio>
</mc:micasa>
```

Exemple 36.

En aquest document s'usa la primera línia per declarar el prefix de l'espai de noms mitjançant l'atribut **xmlns** (*XML namespace*). En aquest cas, s'ha triat el prefix *mc*. Al seu torn, l'espai de noms ha de tenir assignat un **URI** (*Universal Resource Identification*). L'única cosa que es requereix d'aquest URI és que sigui únic en el document. El que sigui un URI significa, entre d'altres coses, que si un va a aquesta adreça no té perquè haver-hi res. Es tracta simplement d'assignar un identificador únic.

3.5. Presentació de documents XML

A continuació es mostraran diferents possibilitats que hi ha per a visualitzar el contingut dels documents XML. En alguns casos es pot afegir una línia dins del document XML fent referència a un document extern el qual està codificat per fer la presentació de les dades del document, ja s'ha comentat però al llarg d'aquest document que no és una pràctica recomanable alterar el document XML amb referències a la seva presentació.

Una altra possibilitat, la més recomanable, és tenir plantilles **XSL** (veure apartat 3.5.4), que **són fitxers codificats per mostrar els documents XML**. El cas ideal és que dins del fitxer XSL hi hagi una referència al fitxer XML que es vol mostrar.

3.5.1. En aplicacions *Desktop*

Els documents XML es poden visualitzar des de qualsevol aplicació *desktop* (aplicació programada amb els llenguatges de programació comuns:: Vbasic, C, Foxpro, Java, etc.) com es poden visualitzar qualsevol altre tipus de dades que provenen de bases de dades, fitxers de text, etc. L'únic que s'ha de fer és accedir a les dades (a través del *parser* d'XML) i a continuació mostrar-les com es faria en el cas que no fossin XML.

Per accedir al *parser* des d'una aplicació *desktop* es fa una referència dins del codi de la aplicació al model d'objectes d'aquest *parser* i des d'aquest moment ja es té accés al *Document Object Model* d'XML el qual té un model d'objectes que permet treballar amb el document XML.

3.5.2. En el Navegador

Per a descriure com s'han de presentar els documents XML en el navegador es pot optar per dues solucions: les mateixes descripcions CSS que s'utilitzen amb HTML i/o les descripcions que es basen en XSL (eXtensible Stylesheet Language, llenguatge de full d'estil extensible).

3.5.3. CSS

L'ús dels fitxers d'estils ***Cascading Style Sheets* està mes estès per al seu ús en documents HTML** però també es poden usar per veure documents XML, tot i que aquesta opció no és molt usual.

Els CSS són com una descripció del format en el qual es desitja que apareguin les entitats definides en un document. Utilitzar CSS amb XML és similar, amb l'excepció que les etiquetes són diferents, a les d'HTML. Un codi com el següent:

```
AUTOR {display:block; font-family:Arial; font-size:small; width:30em}
PREU {display:block; padding:1.2em; font-size:x-small}
TITOL {font-size:x-large; text-align:center; color:#888833}
```

Exemple 37.

seria perfectament vàlid per a que les dades de les etiquetes <autor>, <preu> i <titol> es presentessin segons la seva descripció.

CSS és eficaç per descriure formats i presentacions, però no serveix per decidir quines dades han de veure's per pantalla i quines no.

Per utilitzar CSS amb els documents XML s'ha d'incloure una referència al fitxer d'estils CSS a la capçalera del fitxer. Aquesta referència anirà dins d'una instrucció de processament (veure apartat 3.3.7.7) on s'especificarà que el tipus de document CSS al qual es fa referència és del tipus *text/css*.

```
<?xml-stylesheet type="text/css" href="empleats.css"?>
```

Exemple 38.

La línia de codi anterior indica que, per poder visualitzar el fitxer XML amb estils CSS, s'ha hagut de modificar el fitxer XML (s'ha modificat perquè s'ha inserit aquesta línia que fa referència a aquest fitxer) i això va totalment en contra de la filosofia de l'XML, on les dades i la manera en com aquestes es visualitzen han de ser independents. Dit d'una altre manera, la situació ideal és què la manera com s'ha de visualitzar un fitxer XML ha de venir donada per referències externes al fitxer XML, no per l'alteració del propi document.

A continuació es mostra un exemple de fitxer XML en el qual s'utilitza un fitxer d'estils CSS per visualitzar-lo.

```
<?xml-stylesheet type="text/css" href="empleats.css"?>
<empleats>
  <empleat>
    <nom>Juan Lopez Garcia</nom>
    <carrec>administratiu</carrec>
    <departament>Administracio</departament>
  </empleat>
  ....
</empleats>
```

Exemple 39.

El contingut del fitxer d'estils *empleats.css* és el següent:

```
empleats {display:block;visibility:visible;}
empleat {display:list-item;list-style-type:disc;}
nom {display:block;font-size:12pt;}
carrec {display:block;font-size:12pt;}
departament {display:block;font-size:12pt;}
```

Exemple 40.

En obrir el document XML amb el navegador veurem el seu contingut:

```
Juan Lopez Garcia
administratiu
Administracio
```

Exemple 41.

3.5.4. XSL

La manera natural de mostrar el contingut dels documents XML des del navegador és amb fitxers d'estils XSL. L'**eXtensible StyleSheets Language és un llenguatge que permet definir una presentació o format per un document XML**. Un mateix document XML pot tenir més d'un full d'estil XSL que el mostra en diferents formats (HTML, PDF, RTF, VRML, PostScript, só, etc.).

Bàsicament, XSL és un llenguatge que defineix una transformació entre un document XML d'entrada, i un altre document XML de sortida (a la figura 1 venen indicats com a XML 1 i XML 2 respectivament). *XSL és a l'XML el que CSS és als documents HTML*.

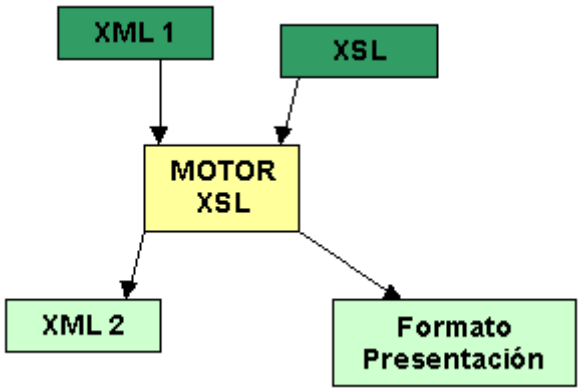

Figura 1.

XSL és un llenguatge XML amb unes etiquetes especifiques destinades a la presentació de les dades contingudes dins dels fitxers XML. Quan el sistema operatiu executa un fitxer amb extensió *xsl* invoca el *parser* de l'XSL i aquest ja sap com interpretar cadascuna de les etiquetes que trobarà dins del document XSL.

Quan s'obre un fitxer XML des d'un navegador, es pot veure l'estructura del fitxer XML (etiquetes o *tags*) i les dades (els valors dels *tags*) i es poden expandir i contraure els nodes que el componen. Aquesta manera de visualitzar el fitxer s'aconsegueix gràcies a l'ús d'una plantilla XSL que aplica per defecte el navegador per veure els documents XML. Aquesta plantilla és molt útil perquè a més permet conèixer si el document està **ben format** (*well-formed*), veure apartat 3.3.2.

Com s'ha mencionat anteriorment, en obrir un fitxer XML des del navegador, aquest mostra la seva estructura (*tags* i dades), però si a la capçalera del fitxer especifiquem que és de tipus *text/css* el navegador aplica una plantilla diferent i mostra només les dades.

```
<?xml-stylesheet type="text/css"?>
```

Exemple 42.

En el cas de l'exemple anterior, el navegador mostraria:

```
Juan Lopez Garcia administratiu Administracio
```

Exemple 43.

Es poden definir documents XSL propis per tal de poder mostrar les dades del fitxer XML en el navegador de la manera que es cregui convenient, i es poden tenir diferents fitxers XSL per a un mateix fitxer XML, de manera que es pot visualitzar de diferents maneres.

Seguint el mateix mètode que a l'anterior exemple, es pot fer referència al document XSL des de la capçalera del document XML (s'ha de recordar que aquesta manera no és la més correcta).

En aquest cas s'especifica que el tipus de document és *text/xsl*:

```
<?xml-stylesheet type="text/xsl" href="empleats.xsl"?>
```

Exemple 44.

El document XML és el mateix que en el cas anterior, però amb el tipus i la referència de la capçalera canviats:

```
<?xml-stylesheet type="text/xsl" href="empleats.xsl"?>
<empleats>
  <empleat>
    <nom>Juan Lopez Garcia</nom>
    <carrec>administratiu</carrec>
    <departament>Administracio</departament>
  </empleat>
  ....
</empleats>
```

Exemple 45.

El fitxer d'estils *empleats.xsl* és el següent:

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
  <xsl:for-each select="empleats/empleat">
    <xsl:value-of select="nom"/>
  </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Exemple 46.

En aquest cas en obrir el document XML amb el navegador es podrà veure:

```
Juan Lopez Garcia
```

Exemple 47.

ja que a la següent clàusula s'ha especificat només el tag *nom*.

```
  <xsl:value-of select="nom"/>
```

Exemple 48.

En el cas que es volguessin veure tots els elements:

```
Juan Lopez Garcia administratiu Administracio
```

Exemple 49.

La clàusula anterior hauria de ser:

```
  <xsl:value-of select="."/>
```

Exemple 50.

XSL no només permet especificar com es volen presentar les dades d'un document XML, sinó que també serveix per filtrar les dades d'acord amb certes condicions. S'assembla una mica a un llenguatge de programació. Possibilita l'execució de bucles, sentències del tipus IF...THEN, seleccions per comparació, operacions lògiques, ordenacions de dades, utilització de plantilles i altres qüestions similars.

Per fer-se una petita idea de com és un codi XSL, a continuació es mostra un senzill exemple que permetria veure tots els continguts de les etiquetes <titol>, <autor> i <preu> d'un document XML, mitjançant un bucle sense condicions:

```
<xsl:template match="/">
<HTML>
 <BODY>
   <xsl:for-each select="/LLIBRES/LLIBRE">
     Titol:
     <xsl:value-of select="TITOL"/><BR/>
     Autor:
     <xsl:value-of select="AUTOR"/><BR/>
     Preu:
     <xsl:value-of select="PREU"/><BR/> euros
   </xsl:for-each>
 </BODY>
</HTML>
</xsl:template>
```

Exemple 51.

4. RDF

En aquest apartat es veurà el *Resource Description Framework* (**RDF**) i les tecnologies i conceptes que hi estan relacionats. Abans d'entrar al detall de què és RDF, s'explicaran una sèrie de conceptes que ajudaran a donar la base per entendre´l millor.

En primer lloc s'explicarà el concepte de **metadades**, que bàsicament són dades que contenen altres dades (veure apartat 4.1). Després es comentaran els **models de metadades d'XML i RDF** (veure 4.2), per després passar a parlar del concepte d'**ontologia** (apartat 4.3), on és veurà entre d'altres coses que el principal propòsit d'una **ontologia** és habilitar la comunicació entre sistemes de computadors de manera independent de la tecnologia, arquitectura i domini d'aplicació.

Es farà una ràpida introducció sobre la **representació del coneixement** (apartat 4.4) tot veient que les metadades i les ontologies en formen part i es veurà quins **llenguatges de representació** existeixen (apartat 4.4.1).

A continuació es farà una visió de les **webs actuals** (apartat 4.5) i de les **webs semàntiques** (apartat 4.6), aquestes últimes basades en RDF. En el cas de la **web actual**, les dades estan preparades per ser utilitzades per les persones, no estan optimitzades per ser utilitzades pels ordinadors. La idea de la **web semàntica** és que les dades de la web puguin ser utilitzades i compreses pels ordinadors (informació en comptes de dades) sense la supervisió humana, dotant així a la web de molta més potència.

Una vegada introduïts els conceptes bàsics s'explicarà què és **RDF** (apartat 4.7) amb més detall així com el seu **model de dades** (apartat 4.7.4) i la seva **sintaxi bàsica** (apartat 4.7.5). Aquests són els aspectes més tècnics.

RDF/XML (apartat 4.8) és el següent punt que es tractarà. És una sintaxi basada en XML per representar RDF.

El següent punt serà **RDF Schema** (apartat 4.8). Es veurà com RDF Schema complementa RDF. Els RDF Schema poden especificar restriccions que s'han de seguir pels models de dades.

Una vegada introduït el concepte d'RDF, es veurà la relació que hi ha entre un concepte nou: metadades **Dublin Core** (apartat 7) i els dos grans temes tractats en aquest document fins aquest punt: **XML** i **RDF** (apartat 4.10). Es veurà com les especificacions RDF proporcionen una infrastructura potent per l'intercanvi de coneixement a la web i utilitzen XML com a sintaxi d'intercanvi.

Com a conclusió de l'estudi, es veuran els avantatges i problemes que representa utilitzar RDF (apartats 4.11.1 i 4.11.2).

4.1. Metadades

Les metadades s'han utilitzat des de fa molt de temps a l'àrea de ciències de la informació on es mantenen índexs per a la catalogació dels llibres. Entre els esquemes clàssics es troba l'ús de fitxes manuals que indiquen el títol, autor, país, ubicació física, etc.

La definició simple de metadades és "dades que contenen altres dades". A la web les dades són els documents, objectes multimedia i els enllaços que la componen. Aquestes dades s'anomenen recursos i les metadades són afirmacions sobre els recursos. A índexs com *Yahoo* s'apliquen esquemes de catalogació per mantenir metadades de recursos.

El problema amb aquests índexs és que per poder arribar a tots els llocs web utilitzen descripcions mínimes i només a nivell de document (títol, *URI*, descripció), a més emmagatzemen les metadades de

forma centralitzada i agents automàtics no poden afegir nous recursos a l'índex sinó que han de ser afegits per persones que cataloguen els documents.

Proveir la infrastructura necessària per què cada lloc web tingui les seves pròpies metadades permet mantenir la informació de catalogació i descripció de manera descentralitzada. La solució recomanada per W3C[11] és utilitzar **RDF** (veure apartat 4.7).

4.2. Models de metadades XML i RDF

Per tal de mantenir metadades de forma distribuïda i interoperables a la web s'ha proposat **RDF** el qual té un millor model que **XML** per manegar dades distribuïdes. XML és un llenguatge per documents semiestructurats, el seu model és en arbre i l'ordre dels elements és important; en canvi RDF és un llenguatge per **metadades**, el model és de un graf etiquetat i dirigit i l'ordre no és rellevant.

XML, en incorporar etiquetes que tracten sobre el significat de la informació, aporta semàntica als documents. Per a un domini en particular, XML és una alternativa com a llenguatge de marcat semàntic, utilitzant **XML Schemes** per a definir vocabularis, o bé una combinació d'XML i RDF. Per permetre interoperabilitat entre aplicacions és necessari suportar una diversitat de dominis. Això significa que s'ha de permetre mantenir diferents vocabularis i les relacions a nivell lògic que existeixen entre ells.

Per descriure amb RDF un document a la web existeix entre d'altres el vocabulari **Dublin Core** (veure apartat 7) el qual constitueix una base pel marcat a la web. Però amb això no hi ha prou per aplicacions en dominis específics, per la qual cosa és necessari crear nous vocabularis. Aquests vocabularis s'anomenen **ontologies**.

4.3. Ontologia

És una descripció formal i explícita dels conceptes dins d'un domini de coneixement determinat, com per exemple els continguts d'una pàgina web i les seves relacions, de manera que pugui ser interpretat pels computadors i permeti realitzar cerques a través de la semàntica en comptes de la sintaxi.

El principal propòsit d'una ontologia és habilitar la comunicació entre sistemes de computadors de manera independent de la tecnologia, arquitectura i domini d'aplicació.

Ontologia és un terme utilitzat per descriure i representar un àrea de coneixement. Les ontologies són utilitzades per persones, bases de dades i aplicacions que necessiten compartir informació especifica sobre un determinat assumpte (o domini), com la medicina, la fabricació d'eines, immobiliàries, reparació d'automòbils, etc.

Les ontologies inclouen definicions utilitzables per les màquines sobre conceptes bàsics del domini i de les relacions existents entre els mateixos. Codifiquen coneixement en un domini i també coneixement que s'expandeix a través de varis dominis. D'aquesta manera fan que el coneixement sigui reutilitzable[12].

El terme ontologia ha estat utilitzat durant molts anys al camp de la intel·ligència artificial i la comunitat de la representació del coneixement, però ara comença a formar part de la terminologia estàndard d'una amplia comunitat incloent el model d'objectes i XML.

L'ontologia té cabuda i fa aportacions a moltes àrees de l'enginyeria de sistemes de la informació. Per exemple, en disseny de bases de dades, en sistemes d'objectes, en sistemes basats en el coneixement i dins de moltes altres àrees com el *datawarehousing*, gestió del coneixement, integració empresarial, etc.

[11] Word Wide Web Consortium. www.w3.org.
[12] Informació provinent d'una nota de premsa: http://www.w3c-es.org/Prensa/2004/nota040210.html

Les ontologies inclouen:

- Definicions de conceptes bàsics del domini.
- Relacions entre ells i entre els elements d'altres dominis.
- Propietats que poden tenir aquests conceptes.

Hi ha diferents ontologies com ara **Dublin Core** (veure apartat 7) o **vCard**[13]. Dublin Core és la més coneguda i és utilitzada per a la catalogació de documents. Les ontologies però també es poden crear independentment.

Normalment les ontologies tenen dos components:

- Noms de conceptes.
  - *Elefant*: És un concepte els membres del qual són animals.
  - *Herbívor*: És un concepte els membres del qual són exactament aquells animals que només mengen plantes o parts de plantes.
  - *Elefant adult*: És un concepte els membres del qual són elefants que tenen una edat superior a 20 anys.
- Coneixement de base o restriccions.
  - *Els* elefants adults pesen més de 2000 Kg.
  - *Tots* els elefants són elefants d'Àfrica o Índia.
  - *Cap* individu és carnívor i herbívor.

4.3.1. Avantatges d'utilitzar ontologies

Les ontologies proporcionen una forma de representar i compartir el coneixement utilitzant un vocabulari comú. Si dues webs que tracten el mateix tema utilitzen la mateixa ontologia, tindran els mateixos termes per descriure el seu contingut. Això té com a avantatge que es podrà extreure informació dels dos llocs i complementar-la per a respondre consultes dels usuaris o aplicacions.

Es pot utilitzar una ontologia com a base de coneixement, o sigui, com una gran base de dades interpretable i utilitzable per les aplicacions.

Permeten analitzar i reutilitzar el coneixement. Una ontologia pot ser reutilitzada per altres llocs que la necessitin en comptes de crear-la de nou; o també pot ser utilitzada conjuntament amb altres per crear una ontologia més gran que englobi unes quantes.

Existeixen múltiples llenguatges per a la representació d'ontologies (OIL, DAML, DAML+OIL, SHOE, TopicMaps), encara que el llenguatge que sembla que agafa més força és **OWL** (veure apartat 8).

De moment les ontologies són una bona solució per donar una semàntica a les dades, encara que aquesta solució està molt limitada a dominis molt concrets, ja que resulta molt difícil fer una ontologia general per a molts dominis alhora. Un altre problema és que depenen del punt de vista del dissenyador i per tant són molt subjectives.

[13] Per més informació: http://www.imc.org

4.4. Representació del coneixement

Les **metadades** i les **ontologies** formen part del camp de la representació del coneixement. Aquest és un camp molt prometedor però que encara no ha pogut desplegar-se àmpliament.

Per a representar el coneixement contingut a les bases de dades es necessita:

- La definició de semàntiques (ontologies).
- Un conjunt de regles lògiques.
- Motors d'inferència.

4.4.1. Llenguatges de representació

Per a descriure la **semàntica** es requereix d'un llenguatge apropiat anomenat llenguatge de representació. Aquests llenguatges tenen la tendència a estar representats en **XML**.

A continuació es fa menció del més importants:

- OML (*Ontology Markup Language*).
- XOL (*Ontology eXchange Language*).
- SHOE (*Simple HTML Ontology Extensions*) és una extensió d'HTML.
- RDF (*Resource Description Framework*) i RDFS (*Resource Description Framework Schemas*) impulsats pel W3C consortium.
- Topics Maps (Mapes temàtics) estàndard ISO.

En el cas d'aquest document el llenguatge de representació emprat és **RDF** i **RDFS**.

4.5. *La web actual (web sintàctica)*

Originalment la web es va desenvolupar com un producte per a ser entès per les persones i no per les màquines. Però Internet ha tingut un èxit i un creixement tan gran que ha superat qualsevol de les millors expectatives. Conseqüentment, la situació actual és que hi ha una immensa quantitat d'informació desorganitzada i desclasificada que no pot ser assimilada per les persones.

És necessari que els ordinadors l'entenguin i la presentin tal com la necessitem, per això cal documentar les dades amb noves dades que les descriguin i els donin significat interpretable per a les màquines i les persones.

A continuació es mostren algunes de les característiques de la web actual:

- És el magatzem d'informació més gran recopilat per persones humanes.

  - Enorme biblioteca amb documents (anomenats pàgines web) connectats entre ells mitjançant enllaços.
  - Grans quantitats d'informació sobre qualsevol assumpte.
  - Accés quasi instantani des de qualsevol lloc amb connexió a Internet.
  - Sistema no centralitzat. Qualsevol persona pot afegir més informació.

- Té dificultats per:
  - Localitzar informació.

- Cercadors basats en:
  - Paraules clau (sense informació del context).
  - Catàlegs (problema d'actualització).
  - Automatitzar tasques.

- Dispositius computacionals immersos en els nostres entorns habituals.
  - Col·leccions multimedia (vídeos, imatges, sons).

- Té un esquema de noms:
  - Identitat única per als documents.

- Grups de persones actualitzen/comparteixen informació.

Tasques difícils a la web sintàctica:

- Cerques complexes.

- Localitzar informació en magatzems de dades.
  - Cerca de viatges.
  - Comparar preus de productes.

- Trobar i utilitzar "serveis web".

- Delegar tasques complexes a agents de la web
  - Organitzar un viatge a algun lloc amb platja no massa car en el qual parlin anglès.
  - Cercar i comparar noticies que parlin de les darreres eleccions.

Conclusió: Els ordinadors s'encarreguen de la tasca fàcil que és la presentació visual i les persones fan les tasques difícils: Navegar i interpretar el contingut.

4.6. La web semàntica

La web va ser dissenyada com un espai d'informació, amb l'objectiu no només de ser útil per les comunicacions entre persones sinó també per permetre a les màquines participar i ajudar. Un dels grans obstacles que hi ha hagut però, ha estat el fet que gran part de la informació de la web està dissenyada per

ser consumida per les persones, i tot i que aquesta informació pugui venir d'una base de dades ben definida, aquesta estructura no és evident per a un motor que rastreja la web cercant informació. **La web semàntica, desenvolupa llenguatges per a la informació ja existent de manera que les màquines la puguin processar**.

La web semàntica és una extensió de la web actual on la informació té un significat ben definit, la qual cosa fa que el treball cooperatiu entre persones i computadors estigui millorat. Proveeix un marc comú que permet compartir dades i reutilitzar-les a través d'aplicacions, empreses, etc. És un esforç col·laboratiu portat per **W3C** i amb la participació d'un gran nombre d'investigadors i empreses. Està basat en **RDF** que integra una varietat d'aplicacions que utilitzen sintaxi **XML**.

La idea de la web semàntica és que les dades puguin ser utilitzades i compreses pels ordinadors sense la supervisió humana, de manera que la web pugui ser dissenyada per a tractar la informació de les pàgines de forma semiautomàtica. Convertir la informació en coneixement seguint un esquema comú i consensuat sobre els dominis.

La web semàntica pretén desenvolupar llenguatges que facilitin la inclusió en la web de contingut llegible per les màquines.

Amb les característiques citades es millorarà la cerca d'informació i es potencia el desenvolupament de tot tipus d'aplicacions que requereixen compartir, reutilitzar, etc., qualsevol tipus de coneixement a través de la xarxa Internet.

Una web semàntica està composta de les següents parts:

- El llenguatge XML per descriure l'estructura de les dades contingudes a les pàgines web.
- El llenguatge RDF per a definir les dades d'una manera estructurada i definir la semàntica.
- L'ontologia per a descriure els conceptes del domini.
- La capa lògica per a relacionar i processar la informació.

Algunes de les aplicacions de la web semàntica:

- Sistemes de cerca basats en el context.
  Actualment els sistemes es basen en catàlegs o paraules clau.
- Agents d'Internet.
  Sistemes automàtics d'obtenció i gestió d'informació (reserves de viatges, comparació de preus, participació en subhastes, etc.).
- Sistemes d'informació emergents.
  Sistema de cerca autònoma que informa quan troba alguna cosa interessant.

Noves característiques de la web:

– *No centralitzada***:** Problemes per garantir integritat de la informació.
– *Informació Dinàmica***:** pot canviar la informació i fins i tot el coneixement sobre aquesta informació.
– *Molta informació***:** El sistema no pot pretendre acaparar tota la informació.

4.6.1. Conceptes i estructura fonamental

La web semàntica es basa en dos punts fonamentals:

- La descripció del significat.
- La manipulació automàtica d'aquestes descripcions.

La descripció del significat s'articula amb:

- Semàntica.
- Metadades.
- Ontologies.

La manipulació s'efectua mitjançant:

- Lògica.
- Motors d'inferència.

4.7. RDF

RDF són les sigles de *Resource Description Framework*. Sorgeix a l'agost de 1997 sota els auspicis del *World Wide Web Corsortium* (**W3C**) amb la finalitat de crear un format que permetés establir compatibilitat entre els diversos sistemes de **metadades**[14].

Està recolzat per protagonistes molt influents a l'escena industrial, tal com els creadors de navegadors[15] (Netscape i Microsoft) i motors de cerca.

Es nodreix dels treballs de diferents col·lectius com altres iniciatives del W3C (**PICS**[16] pel control de continguts o **P3P**[17] destinat a salvaguardar la privacitat a la web) i també dels treballs de la comunitat bibliotecària al voltant del ***Dublin Core*** (veure apartat 7) que és un dels models de metainformació que primer va adoptar la sintaxi de l'RDF.

4.7.1. Introducció

És sabut que Internet és una font gairebé inesgotable d'informació de tot tipus disposada a ser consumida per tothom que hi tingui accés. Per accedir a aquesta informació n'hi ha prou amb escriure al navegador l'adreça Internat del lloc web[18] desitjat.

Cada vegada més es pot trobar juntament amb la publicitat d'un determinat producte la seva adreça Internet, això fa que si algú està interessat en tenir més informació del producte, només ha de visitar aquest lloc web escrivint al navegador aquesta adreça.

Per exemple: Mirant l'aparador d'una agencia immobiliària es pot veure a qualsevol foto o full l'adreça Internet de l'agència. Això permet connectar-se des d'un computador, per exemple des de casa, a

[14] Dades que contenen altres dades.
[15] Programari utilitzat per accedir i navegar per Internet.
[16] Platform for Internet Content Selection. http://www.w3.org/PICS/.
[17] Platform for Privacy Preferences Project. http://www.w3.org/P3P/.
[18] Espai dins d'Internet adreçable mitjançant una adreça Internet (URL) escrita dins del navegador.

l'agencia per tal de veure al llarg de les 24 hores del dia i en el moment que sigui més convenient, tota la varietat que ofereix l'empresa; per exemple: www.sasi.es.

Una altra opció molt utilitzada quan no es té l'adreça Internet és escriure-la al navegador de manera intuïtiva de la següent manera: *www.nom_comercial.com* o *www.nom_comercial.es*, etc., on *nom_comercial* és el nom de l'empresa o producte del qual interessa accedir a la seva informació i que molt sovint està registrat a Internet amb el seu nom comercial. Per exemple, si es vol informació sobre un cotxe de la marca Renault, intuïtivament es pot escriure www.renault.es. Aquesta practica molt sovint dóna un resultat satisfactori ja que s'accedeix allà on realment es volia accedir.

Però quan no se sap exactament l'adreça Internet on hi ha la informació desitjada, o es vol consultar la mateixa informació de diferents fonts per tal de comparar, o no se sap exactament què es vol, s'utilitzen els coneguts motors de cerca, que són programes als quals s'accedeix amb una adreça Internet coneguda prèviament (www.google.com, www.yahoo.com, etc.) i a través dels quals l'usuari escriu la paraula o paraules que vol utilitzar com a clau per tal de fer la cerca de la informació desitjada i com a resultat rep una sèrie d'enllaços a pàgines on hi ha referències a les paraules utilitzades anteriorment, de manera que li pot ajudar a trobar el que vol.

Però aquest resultat no sempre és satisfactori i molt sovint és inexacte, inesperat o està fora de context, ja que pot mostrar pàgines que no tenen relació amb la idea que es tenia en el moment de fer la cerca, o pot mostrar una gran quantitat d'enllaços que s'han de filtrar per tal de poder trobar informació que pugui ser útil.

Això normalment és una feina tediosa (p.e. fent una cerca al motor de cerca *google* de les paraules *viaje* i *caribe* es troben 126.000 links) i molt sovint la navegació a través dels links trobats acaba per derivar en una navegació per llocs que ofereixen informació que no té res a veure amb la desitjada originalment. Per exemple, un dels links trobats en fer la cerca anterior fa referència a la 'Visa America latina y el *Caribe*' que ofereix 'seguro de accidentes de *viaje*', la qual cosa segurament no té res a veure amb la informació que volia la persona que feia la cerca.

Tots aquests problemes són deguts al sistema utilitzat en la creació de pàgines web dins del marc de les **webs actuals** (veure apartat 4.5), on la informació es presenta sense cap estructura semàntica i, per tant, l'única manera de cercar-la és comparant patrons de paraules. En el cas que es feia menció anteriorment, el motor de cerca *google* s'ha limitat a fer una cerca de les paraules *caribe* i *viaje* arreu d'Internet, però si es fa la cerca de les paraules *caribean* i *travel* es troben 36.300 més i així es podria continuar amb totes les llengües. Amb aquest exemple es pot veure que aquestes cerques no estan optimitzades.

La **web semàntica** (veure apartat 4.6) tracta d'optimitzar aquestes cerques i que aquestes puguin ser més útils i selectives. Hi ha vàries iniciatives que passen per transformar les webs tradicionals o actuals a webs semàntiques. Aquest pas es realitza creant una **ontologia** (veure apartat 4.3) de classes sobre el tema que s'està treballant mitjançant un **llenguatge semàntic** com ara **RDF**.

4.7.2. Conceptes bàsics

RDF és un llenguatge per representar informació sobre recursos a la *World Wide Web*. Especialment es va crear amb la intenció de representar metadades sobre recursos web, com un títol, autor, data de modificació d'una pàgina web, *copyright* i informació de drets sobre un document web o el període de disponibilitat d'un recurs compartit. No obstant, com a generalització del concepte de "recurs web", RDF també pot ser usat per representar informació sobre coses que poden ser identificades a la web. Per exemple, informació sobre productes disponibles des de llocs de compres on-line (p.e. informació sobre especificacions, preus, disponibilitat, etc.), o la descripció de la informació de les preferències d'enviament d'un usuari web.

RDF està fet per situacions on la informació necessita ser processada per aplicacions, en comptes de ser només mostrada a les persones. Proveeix un marc comú per expressar aquesta informació, així aquesta pot ser intercanviada entre aplicacions sense pèrdua de significat. L'habilitat de poder intercanviar informació entre diferents aplicacions significa que la informació pot estar disponible per altres aplicacions diferents a aquelles per a les quals es va crear originalment.

A més, RDF està basat en la idea d'identificar coses usant identificadors web o *URIs*[19], i descriure recursos en termes de simples propietats i els seus valors. Això permet a RDF representar declaracions simples sobre recursos com a un gràfic de **nodes** i **arcs** que representen els **recursos**, i les seves **propietats** i **valors**.

A continuació es mostra a la Figura 1 un exemple per ajudar a entendre aquests termes. Es farà una representació en forma de graf RDF de la declaració: *"Hi ha una persona identificada per http://www.w3.org/People/EM/contact#me, el nom del qual és "Eric Miller", el seu email és em@w3.org i el seu distintiu és Dr"*.

Figura 2.

La Figura 2 il·lustra que RDF utilitza **URIs** per identificar:

- Persones, p.e., "Eric Miller", identificat per http://www.w3.org/People/EM/contact#me
- Tipus de coses, p.e., "Person", identificat per http://www.w3.org/2000/10/swap/pim/contact#Person
- Propietats d'aquestes coses, p.e., bústia de correu electrònic, identificat per http://www.w3.org/2000/10/swap/pim/contact#mailbox
- Valors d'aquestes propietats, p.e. mailto:em@w3.org és el valor de la propietat mailbox (RDF també literals com "Eric Miller", i valors d'altres tipus com enters i dates, com a valors de les propietats).

Aquí s'ha vist una petita introducció als grafs RDF, però més endavant és veuran amb més detall (veure apartat 4.7.4).

[19] URI: *Uniform Resource Identifier*. És el conjunt genèric de tots els noms i adreces que fan referència a un recurs.

4.7.3. Característiques principals

Degut al gran volum d'informació que la web conté no és possible gestionar-la manualment i els processos d'automatització són molt difícils d'implementar. La solució que es proposa és l'ús de **metadades** per descriure les dades contingudes a la web.

RDF és una base per processar metadades, proveint interoperabilitat entre aplicacions que intercanvien informació que pot ser interpretada pels computadors a la web, proporcionant un bon mecanisme d'intercanvi d'informació a través de la web.

Destaca per la facilitat que té per habilitar el processament automatitzat dels recursos web. Pot utilitzar-se en diferents àrees d'aplicació; per exemple: en millorar prestacions als motors de cerca, en catalogació per descriure el contingut i les relacions de contingut disponibles en un lloc web, per expressar les preferències de privacitat d'un usuari, així com les polítiques de privacitat d'un lloc web.

RDF juntament amb les signatures digitals serà la clau per a construir la 'web de confiança' per al comerç electrònic, la cooperació i altres aplicacions.

Un dels objectius d'RDF és fer possible especificar la semàntica per les bases de dades en XML de forma normalitzada i interoperable[20]. RDF i XML són complementaris. És molt important també entendre que aquesta sintaxi XML és només una sintaxi possible per RDF i que poden sorgir formes alternatives per a representar el mateix model de dades RDF.

Per facilitar la definició de metadades, RDF conta amb una col·lecció de categories (produïda normalment per un propòsit o domini específic) denominat *schema* (veure apartat 4.8). Les categories s'organitzen en una jerarquia i proporcionen extensibilitat a través d'un refinament de subcategories.

RDF està influït de varies fonts diferents. Les principals influences provenen de la pròpia *Comunitat de Normalització de la Web* en forma de metadades HTML[21] i PICS, la *comunitat bibliotecària*, la *comunitat dels documents estructurats* en forma de SGML[22] i sobre tot XML.

Veiem a continuació diferents descripcions que defineixen el model RDF:

- Sistema que permet la interoperabilitat entre aplicacions mitjançant l'intercanvi d'informació llegible per computador a través de la web (Brickley, Dan y Guha, R. V., 2000).
- Mecanisme que facilita la automatització de processos susceptibles de ser fets amb recursos web (Lassila, Ora, 1998).
- Infrastructura que permet la codificació, intercanvi i reutilització de metadades estructurats (Miller, Eric, 1998).
- És una forma d'expressar relacions entre objectes (Hjelm, Johan, 2001).

RDF a més posseeix un vocabulari extensible, basat en URIs.

Es poden destacar tres aspectes de la semàntica funcional del format RDF: un **model de dades**, una **sintaxi** i un **esquema (schema)**.

[20] RDF utilitza XML com a sintaxi d'intercanvi.
[21] *HyperText Markup Language*. Llenguatge de marcat, estandarditzat pel W3C, utilitzat per la creació de documents que es volen publicar a la Web.
[22] *Standard Generalizaed Markup Language*. Metallenguatge per definir llenguatges de marcat, predecesor d'XML.

4.7.4. Model de dades

El fonament o base d'RDF és un model per representar propietats designades i valors de propietats.

Un objecte d'informació o **recurs**, també conegut per subjecte, es descriu a través d'un conjunt de propietats denominades **descripció RDF** (<rdf:description>). L'essència d'RDF és doncs, un model formal per la representació de les **propietats** i els **valors** d'aquestes propietats, veure Figura 3.

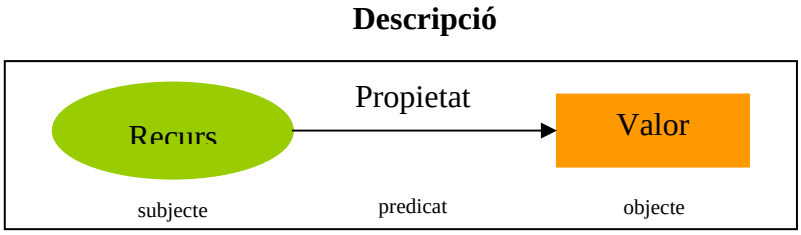

Figura 3.

Les **propietats** d'RDF es poden entendre com atributs dels **recursos** i en aquest sentit corresponen als parells tradicionals d'atribut-valor. A més, aquestes propietats també representen les relacions entre els diferents recursos d'informació, de tal forma que aquest model pot semblar un esquema entitat-relació de les bases de dades relacionals.

El model de dades d'RDF és una forma de sintaxi neutral per representar expressions RDF. La representació del model de dades s'utilitza per avaluar l'equivalència en significat. Dues expressions RDF són equivalents només si les seves representacions del model de dades són les mateixes.

El model de dades bàsic consisteix en tres tipus d'objectes:

- **Recursos**

Qualsevol cosa que pugui ser adreçada mitjançant una URI és un recurs o subjecte. Totes les coses descrites per expressions RDF es denominen **recursos**. Un recurs pot ser una pàgina web completa; com el document HTML "http://www.w3.org/Overview.html" per exemple; pot ser una part d'una pàgina web; p.e. un element HTML o XML específic dins del document font. També pot ser una col·lecció completa de pàgines; p.e. un lloc web complet o un objecte que no sigui directament accessible via web, p.e. un llibre imprès.

Els recursos es designen sempre per URIs. Qualsevol cosa pot tenir un URI; la extensibilitat d'URIs permet la introducció d'identificadors per a qualsevol entitat imaginable.

- **Propietats**

Una *propietat* és un aspecte específic, característica, atribut, o relació utilitzat per descriure un recurs. Cada propietat té un significat específic, defineix els seus valors permesos, els tipus de recursos que poden descriure, i les seves relacions amb altres propietats. Una propietat té associada una URI i un significat concret i pot relacionar-se amb altres propietats.

- **Descripcions ({subjecte} té {predicat} {objecte})**

També conegudes com sentencies RDF (*RDF statement*). Són el conjunt d'un recurs específic juntament amb una propietat denominada, més el valor d'aquesta propietat per aquest recurs. Aquestes tres parts individuals d'una sentencia es denominen, respectivament, **subjecte**, **predicat** i **objecte**. L'objecte d'una sentència (és a dir, el valor de la propietat) pot ser un altre recurs o pot ser un literal; és a dir, un recurs (especificat per un URI) o una cadena simple de caràcters o altres tipus de dades primitives.

Tot seguit es mostra un exemple amb un grup de sentències/descripcions representats pels seu corresponents grups de nodes i arcs. Les descripcions que es representaran seran les següents sentències:

http://www.example.org/index.html té una **data-creació** i el seu valor és **16 Agost 1999**
http://www.example.org/index.html té un **llenguatge** i el seu valor és **Català**

Exemple 52.

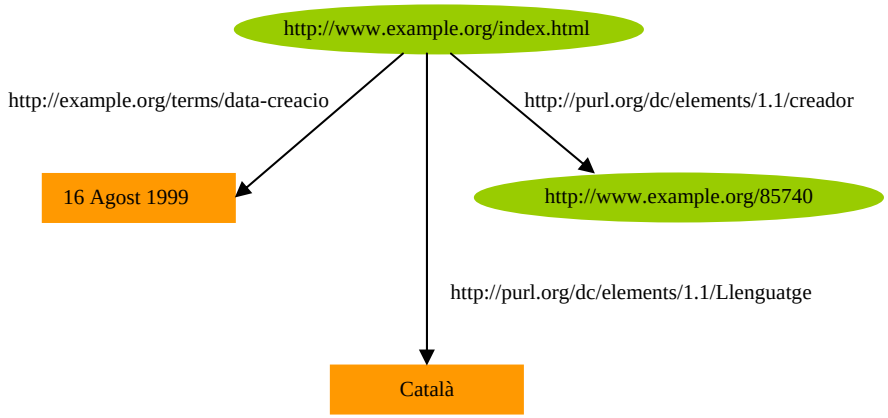

Figura 4.

El graf mostrat a la Figura 4 il·lustra que en les descripcions RDF els objectes poden ser URIrefs o valors constants, anomenats literals, representats per cadenes de caràcters, per tal de poder representar cert tipus de valors de propietats. Els literals no poden ser usats com a subjectes o predicats en sentències RDF, només com a objectes.

Als grafs RDF els nodes que són URIrefs es mostren com a el·lipses, mentre que els nodes que són literals es mostren com a quadrats.

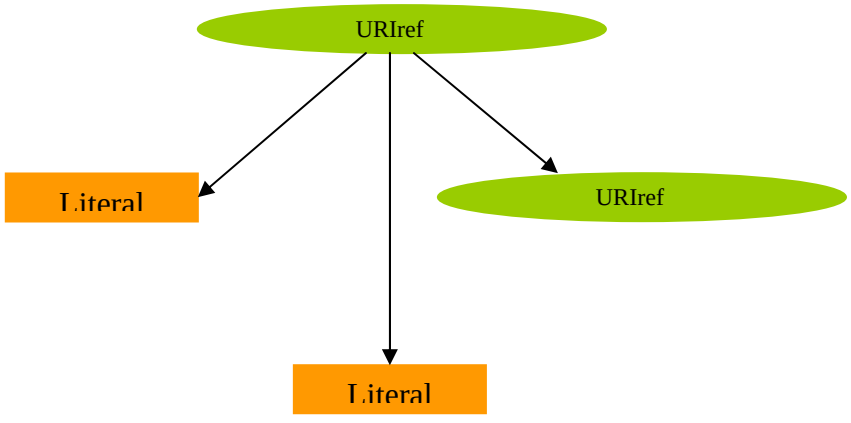

Figura 5.

Hi ha una alternativa als grafs per escriure les sentències/descripcions, aquesta alternativa s'anomena **triples**. En la notació triples, cada sentència/descripció del graf és escrit com una simple tripleta de subjecte, predicat i objecte, en aquest ordre.

4.7.5. Sintaxi bàsica

El model de dades RDF proporciona un marc abstracte i conceptual per definir i utilitzar metadades. Necessita també una sintaxi concreta per crear i intercanviar metadades, per exemple XML. RDF necessita també els *namespaces* (veure apartat 9) per associar amb precisió cada propietat amb l'esquema que defineix la propietat.

La sintaxi bàsica és la d'XML1.0. A més es poden distingir dos tipus de construccions sintàctiques per codificar RDF: la serializada que expressa d'una forma molt regular totes les capacitats d'un model de dades RDF; i la sintaxi abreujada que inclou construccions addicionals.

4.8. RDF/XML

RDF/XML és una sintaxi basada en XML per representar RDF. L'Exemple mostrat a continuació és una petita part d'RDF en RDF/XML corresponent al graf de la Figura 2:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">

  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>

</rdf:RDF>
```

Exemple 53: RDF/XML descripció d'Eric Miller.

Aquest RDF/XML conté URIs, i també propietats com **mailbox** i **fullName** i els seus respectius valors em@w3.org, i Eric Miller. També es pot veure com s'utilitza l'**RDF** com a mecanisme per expressar el model i sintaxi de metadades RDF (**<xmlns:rdf**).

A l'igual que HTML, aquest RDF/XML és processable per computadors i en usar URIs, pot unir peces d'informació a través de la web. No obstant, a diferència de l'*hypertext* convencional, els URIs d'RDF poden fer referència a qualsevol cosa identificable, incloent coses que no són directament recuperables a la web (com per exemple la persona Eric Miller). El resultat és que a més de descriure aquestes coses com a pàgines web, RDF pot també descriure altres coses com cotxes, negocis, gent, nous events, etc.

4.9. RDF Schema

El **model de dades** i la **sintaxi** anteriorment mencionats no faciliten els mecanismes per definir les propietats ni les relacions entre els predicats i altres recursos o subjectes; per això s'ha definit també una especificació per definir els esquemes.

Un *RDF Schema* és un conjunt d'informacions relatives a les classes de recursos que serveix per explicitar les relacions jeràrquiques que estableixen entre ells, o bé per matisar el caràcter obligatori o opcional de les propietats i altres restriccions com el número d'ocurrències, etc. És una extensió d'RDF que proporciona primitives addicionals i enriqueix el model bàsic proporcionant un vocabulari de conceptes i relacions d'herència. Va ser creat per a ser aplicable a tota la variada gamma de recursos de la web.

RDF Schema proporciona informació sobre la interpretació d'una sentencia en un model de dades RDF. Mentre una DTD o un XML Schema poden utilitzar-se per validar la sintaxi d'una expressió XML, amb un esquema sintàctic no hi ha prou per als objectius d'RDF. Els RDF Schema també poden especificar restriccions que s'han de seguir per aquests models de dades.

Molt sovint es presenta el problema de la diversitat d'interessos de les diferents comunitats d'usuaris d'informació web, que es veu reflectida en una disparitat terminológica. Aquesta diversitat es soluciona a RDF utilitzant l'element *namespace* (***xmlns***, veure apartat 9) d'XML que permet expressar un espai o esquema inequívoc en consignar el recurs que defineix la semàntica corresponent al principi d'un registre de metadades.

Per exemple, si l'esquema pel registre de metadades és el Dublin Core[23], s'expressaria:

```
<xmlns:rdf="http://www.w3.org/TR/REC-rdf-syntax">
<xmlns:rdfs="http://www.w3.org/TR/PR-rdf-schema">
<xmlns:DC="http://purl.org/DC">
```

Exemple 54.

A l'Exemple anterior, els noms dels elements i atributs utilitzats al registre RDF es qualificaran respectivament segons:

- L'***RDF*** com a mecanisme per expressar el model i sintaxi de metadades RDF (**<xmlns:rdf**).
- La proposta de recomanació pel ***RDF-Schema*** (**<xmlns:rdfs**).
- El model del ***Dublin Core***, com a vocabulari de designació de tipus d'atributs definit per la comunitat DC (**<xmlns:DC**).

De tal forma que aquestes declaracions del *namespace*, amb els seus respectius URIs (URL), defineixen els esquemes corresponents.

A continuació es mostrarà un exemple que pot ajudar a apropar-se al model de dades, sintaxi i RDF-Schema explicats fins ara. En aquest exemple, *Persona* és una classe amb la seva corresponent descripció llegible per l'home de la "classe de persones". *Animal* és una classe presumptament definida en un altre esquema. Totes les persones són animals, per això es declara que *Persona* es subclasse *d'Animal*. Una persona pot tenir una propietat edat. El valor d'edat és un nombre enter. Una persona pot també tenir una propietat *dni* (Document Nacional d'Identitat). El valor de *dni* és un número enter. L'estat civil d'una persona és un d'aquests: {Solter, Casat, Divorciat, Vidu}. Això s'aconsegueix a través de l'ús de la restricció *rdfs***:***range*: es defineix tant una propietat *estatCivil* com una classe *EstatCivil*. Aleshores s'utilitza *rdfs***:***range* per a establir que la propietat *estatCivil* només 'dona sentit' quan té un valor que és un objecte específic *[instance]* de la classe *EstatCivil*. Així l'esquema defineix un nombre d'objectes específics d'aquesta classe. Si els recursos declarats per a ser del tipus *EstatCivil* en un altre gràfic es confien, és una decisió de nivell d'aplicació.

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

<rdfs:Class rdf:ID="Persona">
  <rdfs:comment>Tipus de persones.</rdfs:comment>
  <rdfs:subClassOf
rdf:resource="http://www.w3.org/2000/03/example/classes#Animal"/>
</rdfs:Class>
```

[23] Els metadades Dublin Core es van dissenyar per facilitar la recuperació de recursos electrónics de forma similar a una fitxa de catáleg a les biblioteques.

```
<rdf:Property ID="estatCivil">
  <rdfs:range rdf:resource="#EstatCivil"/>
  <rdfs:domain rdf:resource="#Persona"/>
</rdf:Property>

<rdf:Property ID="dni">
  <rdfs:comment>Document Nacional d'Identitat</rdfs:comment>
  <rdfs:range
rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
  <rdfs:domain rdf:resource="#Persona"/>
</rdf:Property>

<rdf:Property ID="edat">
  <rdfs:range
rdf:resource="http://www.w3.org/2000/03/example/classes#Integer"/>
  <rdfs:domain rdf:resource="#Persona"/>
</rdf:Property>

<rdfs:Class rdf:ID="EstatCivil"/>

<EstatCivil rdf:ID="Casat"/>
<EstatCivil rdf:ID="Separat"/>
<EstatCivil rdf:ID="Solter"/>
<EstatCivil rdf:ID="Vidu"/>

</rdf:RDF>
```

Exemple 55.

En general, RDF Schema:

- Permet definir no només les propietats d'un recurs (ex. títol, autor, matèria, mida, color, etc.) sinó també definir els tipus de recursos que es descriuran (llibres, pàgines web, persones, empreses, etc.).

- Proporciona informació sobre la interpretació d'una sentència donada en un model de dades RDF (semàntica).

- Permet definir l'herència de classes. Això permet la extensibilitat en quant a elaboració de nous esquemes.

- Permet definir les relacions entre recursos i propietats, la qual cosa ajuda a millorar els processos de cerca.

- Especifica restriccions que han de seguir-se per aquests models de dades.
  El nucli del vocabulari de l'esquema es defineix en un *namespace*.

4.9.1. Classes, propietats i restriccions

- **Classes**

Els següents recursos són les classes principals (*core classes*) que es defineixen com part del vocabulari de l'RDF Schema. Cada model RDF que es traça sobre el *namespace* de l'RDF Schema implícitament els inclou.

rdfs:Resource: Totes les coses que es descriguin per expressions RDF es diuen recursos (*resources*) i es consideren com instàncies (*instances*) de la classe *rdfs:Resource*.

```
  <rdfs:range rdf:resource="#EstatCivil"/>
```

Exemple 56.

rdf:Property: Representa el subconjunt de recursos RDF que són propietats.

```
<rdf:Property ID="estatCivil">
```

Exemple 57.

rdfs:Class: Aquest es correspon amb el concepte genèric d'un tipus (*Type*) o categoria (*Category*). Quan un esquema defineix una nova classe, el recurs que representa la classe ha de tenir una propietat *rdf:type* i el valor del seu recurs és *rdfs:Class*. Les classes RDF poden definir-se per representar qualsevol cosa, com pàgines web, persones, tipus de documents, bases de dades o conceptes abstractes.

```
<rdfs:Class rdf:ID="EstatCivil"/>
```

Exemple 58.

- **Propietats**

Cada model RDF que utilitza la mecànica de l'esquema també inclou (implícitament) les propietats principals següents. Són objectes específics d'una categoria (*instances*) de la classe *rdf:Property* i proporcionen un mecanisme per expressar les relacions entre les classes i els seus objectes específics de categoria (*instances*) o superclasses.

rdf:type: Aquesta propietat indica que un recurs és membre d'una classe, de manera que té totes les característiques que es preveuen d'un membre d'aquesta classe. Quan un recurs té una propietat *rdf:type* i el seu valor és alguna classe específica, es diu que el recurs és un objecte específic de la categoria (*instance*) de la classe especificada.

rdfs:subClassOf: Aquesta propietat especifica una relació subconjunt/superconjunt entre classes. La propietat *rdfs:subClassOf* és transitiva. Si la classe A és una subclasse d'un altre classe B més amplia, i B és una subclasse de C, aleshores A és també implícitament una subclasse de C. Només els objectes específics d'una categoria (*instances*) de *rdfs:Class* poden tenir la propietat *rdfs:subClassOf* i el valor de la propietat és sempre de *rdf:type rdfs:Class*. Una classe pot ser subclasse de més d'una classe.

```
  <rdfs:subClassOf
rdf:resource="http://www.w3.org/2000/03/example/classes#Animal"/>
```

Exemple 59.

rdfs:subPropertyOf: És un objecte específic d'una categoria (*instance*) de *rdf:Property* que s'utilitza per especificar que una propietat és una especialització d'una altre. Una propietat pot ser una especialització de zero, una o més propietats.

- **Restriccions**

Amb les restriccions es pot, per exemple, descriure les limitacions dels tipus de valors que són vàlids per una propietat, o les classes per les que té sentit assignar aquestes propietats. Per exemple, a un esquema RDF es pot establir que una propietat **author** s'utilitzi per indicar els recursos que són membres de la classe **Person**.

rdfs:ConstraintResource: El propòsit d'aquesta classe és proporcionar un mecanisme que permeti als processadors RDF avaluar la seva habilitat per usar la informació restringida associada amb un model RDF.

rdfs:ConstraintProperty: Aquest recurs defineix una subclasse d'*rdf:Property*, on tots els objectes específics de la categoria (*instances*) són propietats utilitzades per especificar restriccions. Aquesta classe és una subclasse de *rdfs:ConstraintResource* i correspon al subconjunt d'aquella classe que representa propietats.

rdfs:range: El valor d'una propietat *range* sempre és una classe. Les restriccions de rang només s'apliquen a les propietats. Una propietat pot tenir com a màxim una propietat de rang. És possible no tenir rang, en aquest cas la classe del valor de la propietat no és obligatori.

```
<rdf:Property ID="estatCivil">
  <rdfs:range rdf:resource="#EstatCivil"/>
  <rdfs:domain rdf:resource="#Persona"/>
</rdf:Property>
```

Exemple 60.

rdfs:domain: Un objecte específic de la categoria de *ConstraintProperty* que s'utilitzi per indicar a la classe(s) que els seus membres poden usar una propietat. Una propietat pot tenir, zero, un o més d'una classe com a domini.

```
<rdf:Property ID="estatCivil">
  <rdfs:range rdf:resource="#EstatCivil"/>
  <rdfs:domain rdf:resource="#Persona"/>
</rdf:Property>
```

Exemple 61.

4.10. Relació entre les metadades Dublin Core, RDF i XML

XML proporciona un conjunt de regles per a la creació de vocabularis que doten d'estructura a les dades i documents de la web. XML dona regles clares per la sintaxi; els *XML Schemes* serveixen com un mètode de composició de vocabularis web. XML és una sintaxi base per documents potents i flexibles, però no imposa restriccions semàntiques al significat d'aquests documents.

RDF és un estàndard per realitzar descripcions senzilles. XML és a la sintaxi, el que RDF a la semàntica - un conjunt clar de regles per proporcionar informació descriptiva senzilla. **RDF Schema** aleshores proveeix una manera de combinar aquestes descripcions en un vocabulari únic.

Les especificacions RDF proporcionen una infrastructura potent per a l'intercanvi de coneixement a la web, utilitzant XML com a sintaxi d'intercanvi.

El Conjunt d'elements **Dublin Core** (veure apartat 7) i l'RDF són dues especificacions diferents. Cap de les dues necessita a l'altre, però la seva co-evolució forma un complement natural dins de la més gran arquitectura de metadades de la web.

Ambdues comunitats, la del Dublin Core i la d'RDF tenen membres en comú i han evolucionat juntes, de forma paral·lela. La comunitat Dublin Core va facilitar molts dels requisits bàsics que s'utilitzen en el disseny d'RDF. Al seu torn, el desenvolupament d'RDF proporciona a la comunitat del Dublin Core un model de dades subjacent molt més formal que ha ajudat a determinar les *best practices* i solucions universals per molts problemes detallats que es varen trobar durant el procés de desenvolupament.

A continuació hi ha un exemple de com es pot usar un vocabulari Dublin Core per definir semàntica addicional sobre recursos descrits dins d'un fragment RDF:

```
<?xml version="1.0" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/">

  <rdf:Description about="http://purl.org/DC/documents/notes-cox-816.htm">
    <dc:title>Recording qualified Dublin Core metadata in HTML</dc:title>
    <dc:description> We describe a notation for recording
      qualified Dublin Core metadata in HTML meta elements. The syntax
      includes recommended usage of the standard HTML syntax to record
      the different classes of qualification needed to represent the
      model.
    </dc:description>
    <dc:date>1999-08-18</dc:date>
    <dc:format>text/html</dc:format>
    <dc:language>en</dc:language>
    <dc:publisher>Dublin Core Metadata Initiative</dc:publisher>
  </rdf:Description>
</rdf:RDF>
```

Exemple 62.

A l'exemple de la Figura 5 es poden veure els següents detalls:

1. El fitxer és un fitxer XML versió 1.0.

2. S'utilitzen els espais de noms (*namespaces*) d'RDF i de Dublin Core:
   RDF namespace: `xmlns:rdf=http://www.w3.org/1999/02/22-rdf-syntax-ns#`
   Dublin Core namespace: `xmlns:dc=http://purl.org/dc/elements/1.1/`

3. Es fa ús només d'un element de l'espai de noms RDF, l'element **Description** (<rdf:Description).

4. Els elements de l'espai de noms Dublin Core (<dc:) que apareixen són tots part dels elements estàndard de Dublin Core[24].

4.11. Conclusions

Després d'aquesta petita presentació d'RDF mostrant-lo com a model semàntic per la representació del contingut web, es pot deixar entreveure les possibilitats que porta el seu model i sintaxi, així com la flexibilitat del seu esquema.

Tot i que ja és una recomanació del W3C, RDF sembla més aviat un estàndard futurible per a les biblioteques digitals del futur.

Es pot dir que RDF és una mena de contenidor de la resta de formats. Es podria dir que l'ús coherent de metadades i l'aplicació de esquemes de metadades preservarà la interoperabilitat semàntica del coneixement electrònic distribuït per la xarxa, i conseqüentment permetrà desenvolupar aplicacions web avançades per la recuperació d'informació o veritables biblioteques virtuals/digitals.

A continuació es mostrarà a mode de resum un llistat amb avantatges i problemes que presenta RDF.

[24] Es poden veure tots els elements del Dublin Core a:
http://es.dublincore.org/documents/dces/index.shtml

4.11.1. Aportacions que ofereix RDF

- Declara meta-informació de forma global (URI).
- Informació manipulable per una màquina.
- Incorporació gradual a Internet.
  - Basat en XML.
- Model simple (graf dirigit acíclic).
- RDF Schema permet definir restriccions.
  - Validació de descripcions.
- Fàcil extensibilitat.
  - Mecanismes d'herència (*subClassOf*, *subProperty*, etc.).
  - Creixement descentralitzat i distribuït.
- Sistema de concretització (*reification)*.
  - Permet establir enunciats sobre enunciats.
- Nombroses aplicacions existents.
  - Annotea, Mozillation, Open Directory Project, etc.

4.11.2. Problemes que presenta RDF

- No ofereix mecanisme d'inferència.
- Expressivitat limitada.
  - Informació negativa, classes disjuntes, etc.
- Ontologies compartides.
  - Unió de descripcions pot produir incoherències i contradiccions.
  - Detecció de inconsistències.
- Evolució d'ontologies.
  - Errors en definicions originals, nous models,...
  - Compatibilitat de descripcions?.
- Interoperabilitat d'ontologies.
  - Relació entre diferents representacions.
- Facilitat d'ús: sintaxi XML?.
- Que fer amb la informació ja existent?.
- Alguns sistemes proposats.
  - SHOE, DAML+OIL, MetaLog.

5. OWL

Els motius que han originat la creació d'aquest llenguatge venen a ser els mateixos que varen motivar l'existència d'RDF. A continuació es citen els elements relacionats i que cal conèixer per entendre l'OWL a mínim la *introducció a XML* (apartat 3.1), l'*origen, motivació i objectius d'XML* (apartat 3.2), *metadades* (apartat 4.1), *model de metadades XML i RDF* (apartat 4.2), *ontologia* (apartat 4.3), *la web actual* (apartat 4.5), *la web semàntica* (apartat 4.6) i *RDF introducció* (apartat 4.7.1).

En aquest apartat es veurà el *Ontologic Web Language* (**OWL**). En primer lloc es farà una petita **introducció** (apartat 5.1) on ja es podrà veure que **OWL** ve a ser el mateix que **RDF** però ampliat. A continuació s'entrarà més en detall sobre aquest llenguatge (apartat 4.7) i es veurà la relació existent entre els tres llenguatges principals tractats en aquest document, **XML**, **RDF** i **OWL** (apartat 5.3).

El següent punt serà l'explicació dels tres subllenguatges de l'OWL (apartat 5.5), que són **OWL Lite** (apartat 5.5.1), **OWL DL** (apartat 5.5.2) i **OWL Full** (apartat 5.5.3). A la descripció d'aquests llenguatges es veuran els elements que els componen tot veient exemples que poden ajudar a entendre'ls.

5.1. Introducció

OWL és un llenguatge creat per proveir una manera comú de processar la informació a la web. Va ser dissenyat per ser interpretat per les aplicacions informàtiques en comptes de pels humans. Intentant simplificar es podria dir que **OWL** és el mateix que **RDF** (veure apartat 4.7) però amb un llenguatge més complert per interoperar amb els computadors.

5.2. OWL

OWL són les sigles de *Ontology Web Language*. El 10 de Febrer de 2004 el Consorci World Wide Web va anunciar la aprovació final de dos tecnologies claus per la Web Semàntica, la Infrastructura per la Descripció de Recursos (RDF) revisada i el Llenguatge d'Ontologies Web (OWL). RDF i OWL són estàndards de la Web Semàntica que proporcionen una infrastructura per a la integració empresarial i la compartició i reutilització de dades a la Web. Aquests formats estàndard per compartir dades amplien les fronteres d'aplicacions, empreses i comunitats - tots aquests tipus diferents d'"usuari" poden compartir la mateixa informació, fins i tot si no comparteixen el mateix programari.

"*RDF i OWL posen les bases per a aplicacions de Web Semàntica*," va dir Tim Berners-Lee, director del W3C i inventor de la *World Wide Web*. "*La seva aprovació com a recomanacions del W3C arriba en un moment en el qual sorgeixen nous productes en àrees tan diverses com la integració empresarial i el suport a les decisions mèdiques. No és diferent dels primers dies de la Web, quan una vegada que la gent veia com funcionava, entenia el seu potencial. Estem entrant en aquesta fase ara, en la qual la gent pot veure els inicis de la Web Semàntica funcionant*".

Una Recomanació del Consorci W3C s'entén per la indústria i per la comunitat web en general com un estàndard web. Cada recomanació és una especificació estable desenvolupada per un Grup de Treball del W3C i revisada pels Membres del W3C. Les Recomanacions promocionen la interoperabilitat de les tecnologies de la Web mitjançant la comunicació expressa del consens de la indústria construït pel grup de Treball.

El programari amb capacitats de Web Semàntica que utilitza RDF i OWL inclou:

- Aplicacions de creació de continguts: els autors poden connectar metadades (assumpte, creador, localització, idioma, drets d'autoria, o qualsevol altre terme) amb documents, fent que els nous documents enriquits puguin ser buscats.

- Eines per a l'administració de llocs web: els llocs Web grans poden administrar-se de forma dinàmica d'acord a categories de contingut, personalitzades per als administradors del lloc.

- Programari que treu profit d'ambdós, RDF i OWL: les Organitzacions poden integrar aplicacions empresarials, publicació i subscripcions utilitzant models flexibles.

- Reutilització de dades entre aplicacions: els formats RDF i OWL són estàndard, no propietaris, que permeten reutilitzar dades procedents de diverses fonts.

5.3. Interrelacions entre XML, RDF i OWL

S'ha escrit molt sobre la *web semàntica*, com si fos una tecnologia que anés a reemplaçar a la web d'avui. "*En realitat*," va explicar Eric Miller, líder de l'activitat de *web semàntica* del W3C, "*és més una evolució que una revolució web*". La web semàntica es construeix a través de canvis incrementals, duent als documents i dades ja disponibles en la web descripcions llegibles per màquines. XML, RDF i OWL permeten a la web ser una infrastructura global per a la compartició de dades i documents, el que també fa més fàcil i fiable la recerca i reutilització d'informació."

L'activitat de *web semàntica* del W3C es recolza en treballs d'altres activitats del W3C, com XML. Es concentra a desenvolupar tecnologies estàndard, damunt d'XML, que permetin el creixement de la web semàntica.

5.3.1. XML Proporciona Regles i Sintaxis per Documents Estructurats

En la seva base, XML proporciona un conjunt de regles per a la creació de vocabularis que dotin d'estructura a les dades i documents de la web. XML dóna regles clares per a la sintaxi; els *XML Schema* serveixen llavors com un mètode de composició de vocabularis web. XML és una sintaxi base per a documents potents i flexibles, però no imposa restriccions semàntiques al significat d'aquests documents.

5.3.2. RDF Ofereix una Infrastructura de Dades per a la Web

Com ja s'ha dit anteriorment, RDF és un estàndard per a realitzar descripcions senzilles. XML és a la sintaxi, el que RDF a la semàntica, un conjunt clar de regles per a proporcionar informació descriptiva senzilla. L'*RDF Schema* llavors proporciona una manera de combinar aquestes descripcions en un vocabulari únic. RDF s'integra en una varietat d'aplicacions incloent:

- Catàlegs de biblioteca.

- Directoris mundials.

- Sindicació i agregació de noticies, programari i contingut.

- Col·leccions personals de música, fotos i esdeveniments.

En aquests casos, cadascun utilitza XML com sintaxi d'intercanvi. Les especificacions RDF proporcionen una infrastructura potent per a l'intercanvi de coneixement en la Web.

"*RDF és part de la base d'un avanç significatiu en el potencial de la Web. Finalment, es podran veure aplicacions i usuaris combinar la informació representada en RDF de múltiples fonts en la web, en maneres que fins ara han estat inconcebibles*," explica Brian McBride, President del Grup de Treball RDF Essencial, "El Grup de Treball RDF essencial ha convertit les especificacions RDF en una base pràctica i matemàticament precisa sobre la qual es pot construir OWL i la resta de la web semàntica.".

5.3.3. OWL ofereix ontologies que funcionen a la Web

El següent que es necessita és una manera per la qual desenvolupar vocabularis per a un assumpte - o un domini - específic. Aquest és el paper d'una ontologia (veure apartat 4.3). Una ontologia defineix els termes utilitzats per a descriure i representar un àrea de coneixement. Les Ontologies són utilitzades per persones, bases de dades i aplicacions que necessiten compartir informació específica sobre un determinat assumpte (o domini) - com la medicina, fabricació d'eines, immobiliàries, reparació d'automòbils, administració financera, etc. Les ontologies inclouen definicions utilitzables per màquines, sobre conceptes bàsics del domini i de les relacions existents entre els mateixos. Codifiquen coneixement en un domini i també coneixement que s'expandeix a través de diversos dominis. D'aquesta manera, fan que aquest coneixement sigui reutilitzable.

OWL proporciona un llenguatge per a la definició d'ontologies estructurades, basades en la web, que ofereix una integració i interoperabilitat de dades més rica entre comunitats descriptives. Els llenguatges anteriors es van utilitzar per a desenvolupar eines i ontologies per a comunitats d'usuaris específiques (particularment en les ciències i en aplicacions de comerç electrònic de companyies específiques), però no van ser definits per a ser compatibles amb l'arquitectura de la *World Wide Web* en general, i de la web semàntica en particular.

OWL utilitza URIs per a fixar noms i la infrastructura per a descripcions en la Web proporcionada per RDF per a agregar les següents capacitats a les ontologies:

- Habilitat de ser distribuïda por molts sistemes.
- Escalabilitat a les necessitats de la web.
- Compatibilitat amb estàndards web per la accessibilitat i la internacionalització.
- Obertura i extensibilitat.

OWL es construeix sobre RDF i RDF Schema i afegeix més vocabulari per a la descripció de classes i propietats: entre unes altres, relacions entre classes (p.e. inconnexió), cardinalitat (p.e."exactament u"), igualtat, major riquesa de tipus en les propietats, característiques de propietats (p.e. simetria), i classes enumerades.

"OWL suposa un gran pas endavant en la representació i organització de coneixement en la *World Wide Web*. Es posiciona com guanyador entre les necessitats de la indústria d'un llenguatge que dirigeixi els seus casos d'ús web actuals, i les restriccions de desenvolupar un llenguatge d'ontologies que serveixi d'engranatge a principis científics establerts i experiència d'investigació." van explicar Jim Hendler i Guus Schreiber, co-presidents del grup de treball d'ontologia web. "Més de cinquanta participants en el grup de treball han dissenyat amb èxit un llenguatge que contempla ambdós conjunts de preocupacions i que és protegit de la mateixa forma per acadèmics i practicants."

5.4. OWL com a extensió d'RDF

OWL com a extensió d'RDF presenta algunes característique que s'enumeren a continuació.

- Definició de classes mitjançant restriccions sobre propietats.
- Definició de classes mitjançant operacions booleanes sobre altres classes.
- Relacions entre classes, p.e. inclusió, disjunció, equivalència.
- Propietats de les relacions, p.e. simètrica, transitiva .
- Cardinalitat.
- Igualtat.
- Classes enumerades.

5.5. Subllenguatges d'OWL

OWL prové de tres subllenguatges. Cadascun d'aquests subllenguatges és una extensió del seu predecessor:

- OWL Lite: És un *subset* d'OWL DL que és fàcil d'implementar.

  OWL Lite dóna suport principalment a aquells usuaris que necessiten una jerarquia de classificació i restriccions simples. Per exemple, mentre suporta restriccions de cardinalitat només permet als valors de cardinalitat de 0 o 1. Deuria ser més simple de proporcionar el suport d'instrument a l'OWL Lite que els seus parents més expressius, i l'OWL Lite proporciona un camí de migració ràpid per a tesauri i altres taxonomies. L'OWL Lite també té una complexitat formal inferior que l'OWL DL.

- OWL DL (OWL *Description Logics*): Restringit al fragment FOL (aprox. DAML + OIL).

  OWL DL està fet per a aquells usuaris que volen la màxima expressivitat. OWL DL inclou tots els constructors del llenguatge OWL, però només poden ser utilitzats sota unes determinades restriccions (per exemple, mentre una classe ha de ser subclasse de múltiples classes no pot ser instanciada per altra classe). OWL DL deu el seu nom per la seva correspondència amb *description logics*, un camp de la recerca que ha estudiat la lògica que forma la fundació formal d'OWL.

- OWL Full: Unió de la sintaxi OWL i RDF.

  Està pensat per usuaris que volen màxima expressivitat i la llibertat sintàctica que ofereix RDF sense garanties computacionals. Permet a una ontologia augmentar el significat d'un vocabulari predefinit (RDF o OWL).

5.5.1. OWL Lite

Aquest subllenguatge usa només una part de les característiques del llenguatge OWL i té més limitacions en l'ús del llenguatge OWL que OWL DL i OWL Full.

OWL Lite té com a principals característiques:

- Jerarquia de classificació.
- Restriccions de cardinalitat, només poden ser 0 o 1.
- Facilita la compatibilitat amb altres models / paradigmes.
- Facilita el desenvolupament d'eines d'autor.

A continuació es mostra una sinopsi dels elements d'aquest llenguatge.

RDF Schema Features	**(In)Equality**	**Property Characteristics**
Class (Thing, Nothing) *rdfs:subClassOf* *rdf:Property* *rdfs:subPropertyOf* *rdfs:domain* *rdfs:range* *Individual*	*equivalentClass* *equivalentProperty* *sameAs* *differentFrom* *AllDifferent* *distinctMembers*	*ObjectProperty* *DatatypeProperty* *inverseOf* *TransitiveProperty* *SymmetricProperty* *FunctionalProperty* *InverseFunctionalProperty*
Property Restrictions	**Restricted Cardinality**	**Header Information**
Restriction *onProperty* *allValuesFrom* *someValuesFrom*	*minCardinality* (només 0 o 1) *maxCardinality* (només 0 o 1) *cardinality* (només 0 o 1)	*Ontology* *imports*
Class Intersection	**Versioning**	**Annotation Properties**
intersectionOf	*versionInfo* *priorVersion* *backwardCompatibleWith* *incompatibleWith* *DeprecatedClass* *DeprecatedProperty*	*rdfs:label* *rdfs:comment* *rdfs:seeAlso* *rdfs:isDefinedBy* *AnnotationProperty* *OntologyProperty*
Datatypes		
xsd datatypes		

5.5.1.1 OWL Lite RDF Schema

A continuació es descriuen les característiques d'OWL Lite relacionades amb RDF Schema.

- ***Class***: Defineix un grup d'elements individuals que estan junts perquè comparteixen algunes propietats.

  Per exemple, *Joan* i *Maria* són ambdós membres de la classe *Persona*. Les classes es poden organitzar en una jerarquia especialitzada usant *subClassOf*.

```
<owl:Class rdf:ID="Joan"/>
<owl:Class rdf:ID="Maria"/>
```

  Exemple 63.

  rdfs:subClassOf: Es poden crear jerarquies de classes fent una o més sentències on una classe és subclasse d'un altra classe.

  Per exemple, la classe *Persona* es pot assignar com a subclasse de la classe *Mare*.

```
<owl:Class rdf:ID="Persona">
  <rdfs:subClassOf rdf:resource="#Mare" />
</owl:Class>
```

  Exemple 64.

- ***rdf:Property***: Les propietats poden ser usades per declarar relacions entre individus o d'individus a valors de dades. Exemples de propietats són *teFill*, *teParents*, *teGerma* i *teEdat*. Els tres primers poden ser usats per relacionar una instància d'una classe *Persona* a un altre instància de la classe *Persona* (esdeveniment de *ObjectProperty*), i l'últim (*teEdat*) pot ser usat relacionar una instància de la classe *Persona* amb una instància del tipus de dades *Integer* (esdeveniment de *DatatypeProperty*). Tant *owl:ObjectProperty* com *owl:DatatypeProperty* són les subclasses de la classe RDF *rdf:Property*.

- ***rdfs:subPropertyOf***: Les jerarquies *Property* poden ser creades fent una o múltiples declaracions on una propietat és una subpropietat d'una o més propietats.

  Per exemple, *teGerma* pot ser declarat per a ser una subpropietat de *teParents*. D'això es pot deduir que si un individu és relacionat amb l'altre per la propietat *teGerma*, llavors també està relacionat amb un altre per la propietat *teParents*.

```
<owl:ObjectProperty rdf:ID="teGerma">
  <rdfs:subPropertyOf rdf:resource="#teParents" />
</owl:ObjectProperty>
```

  Exemple 65.

- ***rdfs:domain***: Un domini d'una propietat limita els individus als quals la propietat pot ser aplicada. Si una propietat relaciona a un individu amb altre individu i la propietat té una classe com un dels seus dominis, llavors l'individu deu pertànyer a la classe.

Per exemple, la propietat *teFill* pot ser declarada per a tenir el domini de *mamífer*. D'això es pot deduir que si *Eduard teFill Anna*, llavors *Eduard* ha de ser un *mamífer*.

```
<owl:ObjectProperty rdf:ID="teFill">
  <rdfs:domain rdf:resource="#mamifer"/>
</owl:ObjectProperty>
```

Exemple 66.

- ***rdfs:range***: La rang d'una propietat limita als individus que la propietat pot tenir com el seu valor. Si una propietat relaciona dos individus i la propietat té una classe amb el seu rang, llavors altre individu deu pertànyer a la classe rang.

  Per exemple, la propietat *teFill* pot ser declarada per a tenir el rang de *mamífer*. D'això es pot deduir que si *Marta* està relacionada amb *Miquel* per la propietat *teFill*, (p.e., *Miquel* és el fill de *Marta*), llavors *Miquel* és un *mamífer*.

```
<owl:ObjectProperty rdf:ID="teFill">
  <rdfs:range rdf:resource="#mamifer"/>
</owl:ObjectProperty>
```

Exemple 67.

- ***Individual*** : Els individus són instàncies de classes i les propietats poden ser usades per relacionar individus. Per exemple, un individu que es digui *Miquel* pot ser descrit com una instància de la classe *Persona* i la propietat *hasEmployer* pot ser usada relacionar a l'individu *Miquel* amb l'individu *StanfordUniversity*.

5.5.1.2 OWL Lite Igualtat i desigualtat

Les següents característiques d'OWL Lite estan relacionades amb la igualtat o la desigualtat

- ***equivalentClass*** : Dues classes poden ser declarades per a ser equivalents. Les classes equivalents tenen les mateixes instàncies. La igualtat pot ser usada per crear classes sinònimes.

  Per exemple, *Cotxe* pot ser declarat per a ser *equivalentClass* a *Automòbil*. D'això es pot deduir que qualsevol individu que és una instància de *Cotxe* també és una instància *d'Automòbil* i viceversa.

```
<owl:Class rdf:ID="Cotxe">
  <owl:equivalentClass rdf:resource="&Automobil;Cotxe"/>
</owl:Class>
```

Exemple 68.

- ***equivalentProperty***: Dues propietats poden ser declarades per a ser equivalents. Propietats equivalents relacionen a un individu amb el mateix joc d'altres individus. La igualtat pot ser usada per crear propietats sinònimes.

  Per exemple, l'individu *Pep* pot ser declarat per a ser el mateix individu que *Josep*.

```
<Persona rdf:ID="Pep">
  <owl:equivalentProperty rdf:resource="#Josep" />
</Persona>
```

Exemple 69.

- ***sameAs***: Dos individus poden ser declarats per a ser el mateix. Aquests constructors poden ser usats per crear un nombre de noms diferents que es refereixen al mateix individu.

  Per exemple, l'individu *Pep* pot ser declarat per a ser el mateix individu que *Josep*.

```
<Persona rdf:ID="Pep">
  <owl:sameAs rdf:resource="#Josep" />
</Persona>
```

Exemple 70.

- ***differentFrom***: Un individu pot ser declarat per a ser diferent d'altres individus. Per exemple, l'individu *Carles* pot ser declarat per a ser diferent dels individus *Rosa* i *Josep*. Així, si els individus *Carles* i *Rosa* són ambdós valors d'una propietat que és declarada per a ser funcional (així la propietat té com a molt un valor), llavors hi ha una contradicció.

  Explícitament la declaració que els individus són diferents pot ser important quan s'usen llenguatges com l'OWL (i RDF) que no assumeixen que els individus tenen un i només un nom.

  Per exemple, sense la informació addicional no es pot deduir que *Carles* i *Rosa* es refereixen a individus diferents.

```
<Persona rdf:ID="Carles" />

<Persona rdf:ID="Rosa">
  <owl:differentFrom rdf:resource="#Carles"/>
</Persona>

<Persona rdf:ID="Josep">
  <owl:differentFrom rdf:resource="#Carles"/>
</Persona>
```

Exemple 71.

- ***AllDifferent***: Un nombre d'individus poden ser declarats per a ser mútuament diferents en una declaració *AllDifferent*. Per exemple, *Carles*, *Rosa*, i Josep podrien ser declarats per a ser mútuament diferents usant el constructor *AllDifferent*. A diferència de la declaració *differentFrom*, això també faria complir que *Josep* i *Rosa* són diferents (no exactament que *Carles* és diferent de *Rosa* i *Carles* és diferent de *Josep*).

  El constructor *AllDifferent* és particularment útil quan hi ha jocs d'objectes diferents i quan s'està interessat en forçar noms únics dintre d'aquells jocs d'objectes. Això és usat en conjunció amb *distinctMembers* per declarar que tots els membres d'una llista són diferents.

```
<owl:AllDifferent>
  <owl:distinctMembers rdf:parseType="Collection">
    <per:Persona rdf:about="#Carles" />
    <per:Persona rdf:about="#Rosa" />
    <per:Persona rdf:about="#Josep" />
```

```
  </owl:distinctMembers>
</owl:AllDifferent>
```

Exemple 72.

5.5.1.3 OWL Lite Property Characteristics

Hi ha identificadors especials en OWL Lite que són usats per a proporcionar informació sobre propietats i els seus valors.

- ***inverseOf***: Una propietat pot ser declarada per a ser l'invers d'un altra propietat. Si la propietat P1 és declarada per a ser l'invers de la propietat P2, llavors si *X* és relacionat amb *Y* per la propietat P2, llavors *Y* és relacionat amb *X* per la propietat P1. Per exemple, si *esFill* és l'invers de *esPare* i Rosa *esPare Marta*, llavors es pot deduir que *Marta esFill Rosa*.

```
<owl:ObjectProperty rdf:ID="Rosa">
  <owl:inverseOf rdf:resource="#Marta" />
</owl:ObjectProperty>
```

Exemple 73.

- ***TransitiveProperty***: Les propietats poden ser declarades per a ser transitives. Si una propietat és transitiva, llavors si el parell (x, y) és una instància de la propietat transitiva P i el parell (y, z) és una instància de P, llavors el parell (x, z) és també una instància de P.

  Per exemple, si l'antecessor (*ancestor*) està declarat per a ser transitiu, i si *Sara* és un antecessor de *Marta* (p.e., (*Sara*, *Marta*) és una instància de la propietat *antecessor*) i *Marta* és un antecessor de *Rosa* (p.e., (*Marta*, *Rosa*) és una instància de la propietat *antecessor*), llavors es pot deduir que *Sara* és un antecessor de *Rosa* (p.e., (*Sara*, *Rosa*) és una instància de la propietat *antecessor*).

  A l'exemple següent la propietat *antecessor* és transitiva.

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

<Persona rdf:ID="Marta">
  <antecessor rdf:resource="#Sara" />
</Persona>

<Persona rdf:ID="Rosa">
  <antecessor rdf:resource="#Marta" />
</Persona>
```

Exemple 74.

- ***SymmetricProperty***: Les propietats poden ser declarades per a ser simètriques. Si una propietat és simètrica, llavors si el parell (x, y) és una instància de la propietat simètrica P, llavors el parell (y, x) és també una instància de P.

Per exemple, *amic* pot ser declarat per a ser una propietat simètrica. Llavors veient que *Josep* és *amic* de *Rosa* es pot deduir que *Rosa* és *amic* de *Josep*.

```
<owl:ObjectProperty rdf:ID="amic">
  <rdf:type rdf:resource="&owl;SymmetricProperty" />
  <rdfs:domain rdf:resource="#Persona" />
  <rdfs:range rdf:resource="#Persona" />
</owl:ObjectProperty>

<Persona rdf:ID="Josep">
  <amic rdf:resource="#Rosa" />
</Persona>
```

Exemple 75.

FunctionalProperty: Les propietats poden ser declarades per a tenir un valor únic. Si una propietat és un *FunctionalProperty* llavors no té més que un valor per a cada individu. A *FunctionalProperty* la cardinalitat mínima de la propietat és el zero i la seva cardinalitat màxima és 1.

Per exemple, *patroPrimari* pot ser declarat per a ser un *FunctionalProperty*. D'això es pot deduir que cap individu pot tenir més que un patró primari. Malgrat tot, això no implica que cada Persona ha de tenir almenys un patró primari.

```
<owl:ObjectProperty rdf:ID="patroPrimari">
  <rdf:type rdf:resource="&owl;FunctionalProperty" />
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="empleat">
  <owl:inverseOf rdf:resource="#patroPrimari" />
</owl:ObjectProperty>
```

Exemple 76.

- ***InverseFunctionalProperty***: Les propietats poden ser declarades per a ser inverses funcionals. Si una propietat és inversa funcional llavors l'invers de la propietat és funcional. Així l'invers de la propietat té al menys un valor per a cada individu (propietat inequívoca).

  Per exemple, *teNombreSeguretatSocial* pot ser declarat per a ser invers funcional (o inequívoc). L'invers d'aquesta propietat (que podria ser *nombreSeguretatSocial*) té un valor per a qualsevol individu en la classe de nombres de la Seguretat Social. Així el nombre de la Seguretat Social de qualsevol persona és l'únic valor per a la seva propietat *nombreSeguretatSocial*. D'això es pot deduir que no poden haver dos instàncies diferents de *Persona* amb el mateix nombre de la Seguretat Social. També es pot deduir que si dues instàncies de Persona tenen el mateix nombre de la Seguretat Social, llavors aquelles dues instàncies es refereixen al mateix individu.

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

<owl:ObjectProperty rdf:ID="nombreSeguretatSocial">
  <rdf:type rdf:resource="&owl;InverseFunctionalProperty" />
  <owl:inverseOf rdf:resource="#teNombreSeguretatSocial" />
</owl:ObjectProperty>
```

Exemple 77.

5.5.1.4 OWL Lite Restriccions de Propietat

OWL Lite permet a les restriccions ser col·locades com a propietats que poden ser usades per les instàncies d'una classe. Aquest tipus (i les restriccions de cardinalitat, veure apartat 5.5.1.5) és usat dintre del context d'un *owl:Restriction*. L'element *owl:onProperty* indica la propietat restringida. Les dues restriccions següents limiten quins valors poden ser usats mentre que les restriccions de la secció (apartat 5.5.1.5) limiten quants valors poden ser usats.

- ***allValuesFrom***: La restricció *allValuesFrom* és declarada sobre una propietat respecte a una classe. Això vol dir que aquesta propietat sobre aquesta classe particular té associada una restricció de rang local. Així si una instància de la classe és relacionada amb la propietat d'un segon individu, llavors el segon individu pot deduir-se com una instància de la classe restricció de rang local.

  Per exemple, la classe *Persona* pot tenir una propietat *teFilla* restringida per tenir *allValuesFrom* de la classe Dona. Això vol dir que si l'individu *Marta* de la classe *Persona* és relacionat per la propietat *teFilla* a l'individu *Rosa*, llavors d'això es pot deduir que *Rosa* és una instància de la classe *Dona*.

```
<owl:Class rdf:ID="Persona">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#teFilla" />
      <owl:allValuesFrom rdf:resource="#Dona" />
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Exemple 78.

- ***someValuesFrom***: La restricció *someValuesFrom* és declarada sobre una propietat respecte a una classe. Una classe particular pot tenir una restricció contra una propietat en la qual almenys un valor per a aquella propietat és d'un cert tipus.

  Per exemple, la classe *SemanticWebPaper* pot tenir una restricció *someValuesFrom* contra la propietat *hasKeyword* que declara que algun valor per a la propietat *hasKeyword* deuria ser una instància de la classe *SemanticWebTopic*.

```
<owl:Class rdf:ID="SemanticWebPaper">
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasKeyword" />
      <owl:someValuesFrom rdf:resource="#SemanticWebTopic" />
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Exemple 79.

5.5.1.5 OWL Lite Restriccions de Cardinalitat

OWL Lite inclou una forma limitada de restriccions de cardinalitat. Les restriccions de cardinalitat a OWL (i *OWL Lite*) s'esmenten com restriccions locals, ja que són declarades sobre propietats respecte a una classe particular. És a dir les restriccions obliguen la cardinalitat d'aquella propietat sobre les instàncies

d'aquella classe. Les restriccions de cardinalitat *OWL Lite* són limitades perquè només permeten declaracions sobre cardinalitats de valor 0 o 1 (no permeten valors arbitraris per a la cardinalitat, com en el cas d'OWL DL i OWL Full).

- ***minCardinality***: Una restricció *owl:minCardinality* descriu una classe de tots els individus que tenen com a mínim N valors semànticament diferents (individus o valors de dades) per a la propietat afectada, on la N és el valor de la restricció de cardinalitat.

  L'exemple següent descriu una classe dels individus que tenen com a mínim dos pares.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#tePares" />
  <owl:minCardinality
rdf:datatype="&xsd;nonNegativeInteger">2</owl:minCardinality>
</owl:Restriction>
```

  Exemple 80.

- ***maxCardinality***: Una restricció owl:maxCardinality descriu una classe de tots els individus que tenen com a màxim N valors semànticament diferents (individus o valors de dades) per a la propietat afectada, on la N és el valor de la restricció de cardinalitat.

  L'exemple següent descriu una classe dels individus que tenen com a màxim dos pares.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#tePares" />
  <owl:maxCardinality
rdf:datatype="&xsd;nonNegativeInteger">2</owl:maxCardinality>
</owl:Restriction>
```

  Exemple 81.

- ***cardinality***: Descriu una classe de tots els individus que tenen exactament N valors semànticament diferents (individus o valors de dades) per a la propietat afectada, on N és el valor de la restricció de cardinalitat.

  El següent exemple descriu una classe d'individus amb exactament 2 pares.

```
<owl:Restriction>
  <owl:onProperty rdf:resource="#tePares" />
  <owl:cardinality
rdf:datatype="&xsd;nonNegativeInteger">2</owl:cardinality>
</owl:Restriction>
```

  Exemple 82.

5.5.1.6 OWL Lite Intersection Classe

OWL Lite conté un constructor d'intersecció però limita el seu ús.

intersectionOf: Uneix una classe per a una llista de descripcions de classe. Una declaració *owl:intersectionOf* descriu una classe per a la qual l'extensió de classe conté amb precisió a

aquells individus que són els membres de l'extensió de classe de totes les descripcions de classe en la llista.

En aquest exemple el valor de *owl:intersectionOf* és una llista de dues descripcions de classe (dues enumeracions), ambdues descriuen una classe amb dos individus. El resultat de la intersecció és una classe amb un individu, *Jordi*, ja que és l'únic individu que és comú a ambdues enumeracions.

```
<owl:Class>
  <owl:intersectionOf rdf:parseType="Collection">
    <owl:Class>
      <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Jordi" />
        <owl:Thing rdf:about="#Josep" />
      </owl:oneOf>
    </owl:Class>
    <owl:Class>
      <owl:oneOf rdf:parseType="Collection">
        <owl:Thing rdf:about="#Jordi" />
        <owl:Thing rdf:about="#Joan" />
      </owl:oneOf>
    </owl:Class>
  </owl:intersectionOf>
</owl:Class>
```

Exemple 83.

5.5.1.7 OWL Datatypes

OWL usa mecanismes RDF per a valors de dades. Mirar la de Guia d'OWL *secció sobre datatypes*[25] ([]) per a una descripció més detallada dels tipus de dades encastats d'OWL agafats en gran part d'*XML Schema datatypes*.

5.5.1.8 OWL Lite Header Information

OWL Lite suporta nocions d'inclusió d'ontologia i relacions i adjunció de la informació a ontologies. Mirar la *Referència de OWL*[26] per a detalls i la *Guia d'OWL*[27] per a exemples.

5.5.1.9 OWL Lite Annotation Properties

OWL Lite permet anotacions sobre classes, propietats, individus i capçaleres d'ontologia. L'ús d'aquestes anotacions està subjecte a certes restriccions. Mirar la *secció sobre Anotacions en la Referència de OWL*[28] per més detalls.

5.5.1.10 OWL Lite Versioning

[25] http://www.w3.org/TR/owl-guide/Datatypes1
[26] http://www.w3.org/TR/owl-ref/
[27] http://www.w3.org/TR/owl-guide/
[28] http://www.w3.org/TR/owl-ref/Annotations

RDF ja té un petit vocabulari per a descriure la informació sobre versions. L'OWL amplia considerablement aquest vocabulari. Mirar la *Referència d'OWL*[29] per més detalls.

5.5.2. OWL DL

Amplia les següents característiques d'OWL Lite:

- Limitacions: les classes no són instàncies ni tipus, els tipus no són instàncies ni classes.
- RDF ⊄ OWL DL.

Class Axioms	**Boolean Combinations of Class Expressions**
oneOf, dataRange *disjointWith* *equivalentClass* (aplicat a expressions de classes) *rdfs:subClassOf* (aplicat a expressions de classes)	*unionOf* *complementOf* *intersectionOf*
Arbitrary Cardinality	**Filler Information**
minCardinality *maxCardinality* *cardinality*	*hasValue*

5.5.3. OWL Full

Amplia les següents característiques d'OWL DL:

- Llibertat sintàctica d'RDFS sense garanties computacionals.
- RDF ⊂ OWL Full.

5.5.3.1 Descripció de Llenguatge Incremental d'OWL DL i OWL Full

Tant l'OWL DL com l'OWL Full utilitzen el mateix vocabulari encara que l'OWL DL sigui subjecte a algunes restriccions. L'OWL DL requereix separació de tipus (una classe no pot ser també un individu o propietat, una propietat no pot ser també un individu o una classe). Això implica que les restriccions no poden ser aplicades als mateixos elements de llenguatge d'OWL (cosa que és permesa en l'OWL Full). A més, l'OWL DL requereix que les propietats siguin *ObjectProperties* o *DatatypeProperties*:

[29] http://www.w3.org/TR/owl-ref/Header

DatatypeProperties són relacions entre instàncies de classes i literals RDF i *XML Schema datatypes*, mentre *ObjectProperties* estan relacionats entre instàncies de les dues classes.

oneOf: (classes enumerades): Les classes poden ser descrites per l'enumeració dels individus que constitueixen la classe. Els membres de la classe són exactament el joc d'individus enumerats; ni més, ni menys. Per exemple, la classe *daysOfTheWeek* pot ser descrita simplement enumerant als individus diumenge, dilluns, dimarts, dimecres, dijous, divendres, dissabte. D'això es pot deduir la cardinalitat màxima (7) de qualsevol propietat que té *daysOfTheWeek* com la seva restricció *allValuesFrom*.

hasValue: (valors de propietat): Una propietat pot requerir de tenir a un cert individu com un valor. Per exemple, els casos de la classe de *dutchCitizens* poden ser caracteritzats com aquella gent que té *theNetherlands* com un valor de la seva nacionalitat. (El valor de nacionalitat, *theNetherlands*, és un cas de la classe de Nacionalitats).

disjointWith: Les classes poden ser declarades per a ser inconnexes l'una de l'altre. Per exemple, l'Home i la Dona poden ser declarats per a ser classes inconnexes. D'aquesta declaració *disjointWith*, es pot deduir una inconsistència quan un individu és declarat per a ser un cas d'ambdós i de manera similar es pot deduir que si un és un cas d'Home, llavors no és un cas de Dona.

unionOf, complementOf, intersectionOf (Combinacions booleanes): L'OWL DL i L'OWL Full permeten combinacions Booleanes arbitràries de classes i restriccions: *unionOf*, *complementOf*, i *intersectionOf*. Per exemple, usant *unionOf*, es pot declarar que una classe conté les coses que són *USCitizens* o *DutchCitizens*. Usant *complementOf*, es podria declarar que els nens no són *SeniorCitizens*. La ciutadania de la Unió europea podria ser descrita com la unió de la ciutadania de tots els Estats membres.

minCardinality, maxCardinality, cardinality (cardinalitat plena): Mentre en l'OWL Lite, les cardinalitats són restringides a almenys, en la majoria o exactament 1 o 0, l'OWL Full permet a declaracions de cardinalitat per a nombres sencers arbitraris no negatius. Per exemple la classe de DINKS ("Dual Income, No Kids") restringiria la cardinalitat de la propietat *hasIncome* a una cardinalitat mínima de dues (mentre la propietat *hasChild* hauria de ser restringida a la cardinalitat 0).

6. Conclusions

La web està en plena expansió i a mida que augmenta el nombre d'usuaris i el nivell de coneixement d'aquests, també creix el nivell de necessitats i d'exigència. Això fa que el que fins fa poc temps era molt bo, útil, novedós, etc., ara sigui molt criticat per no arribar a les noves expectatives creades.

Si a això li afegim el fet que al mon de la informàtica continuen havent diferents estàndards (plataformes, sistemes operatius, navegadors web, etc.) i això no sembla que es vagi ha acabar, la creació de noves tecnologies vàlides per a tothom sembla una cosa quasi utòpica.

En aquest sentit i amb el concepte de web semàntica explicat en aquest document, sembla que la tendència sigui la necessitat de trencar aquesta heterogeneïtat i convergir cap a un concepte/tecnologia únic, que sigui multiplataforma i conseqüentment capaç d'avançar en bé de tots.

El consorci W3C està fent un molt bon paper en aquest sentit i ja sona com una mena d'estàndard de la comunitat informàtica mundial per tot el que envolta a les tecnologies web. Això si, tot això és a costa de multitud d'eines/llenguatges diferents moltes vegades difícils d'aprendre i en constant evolució, la qual cosa fa que sigui necessari un gran esforç per part dels informàtics per tal d'*estar al dia* i sempre es tingui la molesta sensació d'estar desfasat.

7. Annex 1. Dublin Core

Dublin Core Metadata Initiative (DCMI) és una organització dedicada a promocionar l'estandardització d'un vocabulari comú per descriure els recursos de les pàgines web.

A continuació s'enumeren algunes de les característiques de Dublin Core:

- Dublin Core Metadata Initiative: http://dublincore.org/
- *Namespace*: http://purl.org/dc/elements/1.1/
- Conjunt d'elements bàsics amb el seu significat compartit
  - Contingut: Coverage, Description, Type, Relation, Source, Subject, Title
  - Propietat Intel·lectual: Contributor, Creator, Publisher, Rights
  - Instanciació: Date, Format, Identifier, Language
- Cada element bàsic admet una sèrie de qualificadors
  - Refinament d'elements
    Exemple: Date.created, Description.tableOfContents
  - Esquema de codificació
    Exemples: Identifier.URI, Date.DCMIPeriod

8. Annex 2. Ontologies per la web

En aquest apartat es pot veure la evolució de les ontologies utilitzades per la web.

- SHOE (*Simple HTML Ontology Extensions*) Univ. Mayland, 1996.
  Permet definir **ontologies** en documents HTML.
  Objectiu = Facilitar cerques i anotacions de documents.
- XOL (*XML Ontology exchange Language*).
  Objectiu = Intercanvi de definicions d'**ontologies** entre sistemes.
- OIL (*Ontology Inference Layer*).
  Utilitza sintaxi RDF(S) i afegeix primitives de representació del coneixement en marcs.
  Es basa en l'ús de *description logics*.
- DAML (*DARPA Agent Markup Language*).
  Projecte americà de creació de llenguatge per a **ontologies**.
- DAML-OIL. Projecte conjunt.
  Serà la base d'OWL.
- OWL (*Web Ontology Language*).
  Estàndard desenvolupat en el consorci Web.

9. Annex 3. Schemes i Namespaces

Quan s'escriu una frase en llenguatge natural s'utilitzen paraules que tenen la intenció de transmetre un significat inequívoc. En el cas d'RDF és molt important que tant l'escriptor com el lector d'una sentència (declaració) entenguin el mateix significat per als termes utilitzats, com p.e. **Creator, approvedBy, Copyright**, etc.

A RDF el significat s'expressa a través d'un esquema. Un esquema defineix els termes que s'utilitzaran a una sentència (declaració) RDF i li atorgarà significats específics.

Un esquema és un lloc on es documenten (s'expliquen) les definicions i restriccions d'ús de les propietats. Per evitar confusions entre definicions independents del mateix terme, RDF utilitza la facilitat dels *namespace* d'XML.

10. Bibliografia

- http://www.w3.org/XML/: És el document de partida i normatiu pel que fa a XML.
- http://www.w3.org/TR/2004/REC-xml-20040204/: És la normativa oficial del W3C pel que fa a XML.
- http://www.elcel.com/products/xmltools.html: Conté un *XML-Validator* a partir de DTD.
- http://www.w3.org/XML/Schema: És el document base pel que fa a XML-Schema. Inclou enllaços a moltes eines i tutorials d'XML-Schema.
- http://www.w3.org/TR/2001/WD-xmlschema-formal-20010320/: Document normatiu de l'XML-Schema 1.0.
- http://www.w3schools.com/schema/default.asp: Tutorial d'XML-Schema.
- http://www.w3.org/RDF/Metalog/docs/sw-easy.html: W3C. The Semantic Web Made Easy.
- http://www.w3.org/2001/sw/: W3C. Semantic Web.
- http://www.w3.org/TR/REC-rdf-syntax/: W3C. Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation 22 February 1999.
- http://www.w3.org/TR/rdf-schema/: W3C. RDF Vocabulary Description Language 1.0: RDF Schema.
- http://www.w3.org/DesignIssues/RDFnot.html: Tim Berners-Lee. What a semantic can represent.
- http://www.w3.org/TR/rdf-syntax-grammar/: RDF/XML Syntax Specification.
- http://www.bib.uc3m.es/~mendez/publicaciones/7jc99/rdf.htm: DRF: un modelo de metadatos flexible para las bibliotecas digitales del próximo milenio.
- http://www.ontology.org: Ontology.org – Enabling virtual business.
- http://www.semanticweb.org: SemanticWeb.org.
- http://www.w3.org/TR/2003/PR-rdf-primer-20031215: RDF Primer.
- http://www.w3c-es.org/Prensa/2004/nota040210.html: El consorcio World Wide Web publica las recomendaciones RDF y OWL.
- http://www.w3c-es.org/Prensa/2004/nota040210.html: El consorcio World Wide Web publica las recomendaciones RDF y OWL.
- http://www.w3.org/TR/2004/REC-owl-features-20040210/: Visió general d'OWL. Des d'aquí es pot anar a la resta d'enllaços que conformen tot allò relacionat amb OWL.
- http://xml.coverpages.org/ni2004-01-08-a.html: Parla essencialment d'OWL-S
- http://www.w3.org/TR/2004/REC-owl-semantics-20040210/: OWL Web Ontology Language Semantics and Abstract syntax.

- http://www.w3.org/TR/2003/NOTE-owl-xmlsyntax-20030611/owl-xmlsyntax.html: OWL Web Ontology Language XML Presentation syntax.
- http://www.w3.org/PICS/: Platform for Internet Content Selection.
- http://www.w3.org/P3P/: Platform for Privacy Preferences Project.
- http://es.dublincore.org/documents/dces/index.shtml: Es poden veure tots els elements del Dublin Core.
- http://www.w3.org/TR/owl-ref/: OWL Web Ontology Language Reference.
- http://www.w3.org/TR/owl-guide/: OWL Web Ontology Language Guide.