Enginyeria tècnica en Informàtica de Sistemes

ONTO-TRAUMA

Ontologia d'aprenentatge en l'àrea de la cirurgia traumatològica

Miquel Lladó i Puigdemont
TFC – XML i Web semàntica
1º semestre del 2011-12
Consultor: Josep Anton Pérez Braña

1. ÍNDEX

2. Introducció 2
2.1. Justificació del treball 2
2.2. Objectius 2
2.2.1. Objectius generals 3
2.2.2. Objectius específics 3
2.3. Enfocament i planificació 3

3. XML i XML-Schema 4
3.1. Orígens del XML 5
3.2. Sintaxi del XML 5
3.3. Espai de noms en XML 7
3.4. DTD i XML-Schema 7
3.4.1. DTD 7
3.4.2. XML-Schema 8

4. RDF i RDF-Schema 9
4.1. RDF 9
4.2. RDF-Schema 9

5. Ontologies i OWL 11
5.1. Ontologies 12
5.2. OWL 13
5.2.1. Subllenguatges d'OWL 14

6. Aplicació pràctica 16
6.1. Protégé 16
6.1.1. Instal·lació 16
6.1.2. Funcionalitats 17
6.2. Disseny de la ontologia 23
6.3. Wiki 31
6.4. Wiki Semàntica 32
6.5. Ontowiki 32
6.5.1. Instal·lació 33
6.5.2. Funcionalitats 35
6.6. Semantic MediaWiki 37

7. Conclusions i línies de futur 38

8. Glossari de termes 40

9. Bibliografia 43

Annex 1. Enunciat 45

Annex 2. Instal·lació Semantic MediaWiki 46

2. INTRODUCCIÓ

2.1 Justificació del treball

L'evolució que ha tingut la web en aquests últims anys ha estat tan gran que s'ha hagut de modificar la forma de tractar la informació. Fins fa poc, la informació s'introduïa a la web i a partir d'aquí es tractava, però el fet que aquest volum d'informació augmentés molt considerablement dia a dia, va provocar que la informació s'hagués de tractar d'una forma diferent per evitar una saturació, d'una forma estructurada. És aquí on entra la Web semàntica.

La Web semàntica introdueix significat a la informació, fent que aquesta es pugui tractar d'una forma més eficient. Inclús introdueix el concepte que hi hagi un intercanvi d'informació entre les pròpies aplicacions que la tracten. En definitiva, permet el intercanvi d'informació entre màquines.

Aquest treball pretén treballar en aquest camp, aprofundint en les possibilitats tecnològiques i algunes de les diferents eines i aplicacions que se n'han derivat, per acabar dotant de significat i semàntica els documents i facilitar-ne el processament automàtic.

2.2 Objectius

Els objectius principals del projecte són: estudiar els conceptes bàsics de la web semàntica, conèixer la representació i estructura d'alguns llenguatges de informació de la web i aplicar l'ús d'ontologies[1] en la representació del coneixement. Per assolir i consolidar aquests objectius, es realitzaran casos pràctics d'aquestes tecnologies, implementant una aplicació web senzilla.

Aquests objectius però, es podrien diferenciar en uns objectius generals de tot projecte de final de carrera i els objectius específics del treball especificat (Annex 1 – Enunciat).

[1] ONTOLOGIA: En metafísica, estudi del ser, de les entitats, i de les seves relacions.

2.2.1 Objectius generals

L'objectiu final d'aquest projecte és la realització d'un treball que serà la culminació de totes les assignatures cursades en els estudis de l'Enginyeria Tècnica en Informàtica de Sistemes. Com a tal, serà una síntesi de les assignatures que s'han impartit en aquesta enginyeria. El objectius generals els podríem enumerar com:

- A partir d'un problema o enunciat, analitzar, planificar i estructurar un projecte per donar solució a aquest projecte.
- Fer ús de tots els coneixements que s'han adquirit al llarg d'aquests estudis per desenvolupar el projecte, fins a trobar-ne una solució viable i factible.
- Elaborar una memòria del projecte.
- Elaborar una presentació virtual del projecte i resultats.

2.2.2 Objectius específics

Els objectius més específics que es busquen alhora de realitzar aquest projecte serien:

- Conèixer el llenguatge XML, RDF i OWL i quina relació hi ha entre ells.
- Crear una ontologia , mitjançant el programa lliure PROTEGE, de un enunciat establert (veure ANNEX 1).
- Crear una wiki a partir de la ontologia que hem dissenyat, i veure com es relaciona la wiki amb la ontologia.

Per aconseguir aquests objectius caldrà aprendre i saber utilitzar programari específic com el PROTEGE, que hem mencionat anteriorment, i el ontowiki i/o el Semantic mediaWiki.

És dóna per entès que la realització d'aquest projecte implicarà un objectiu indirecte que serà la realització d'un projecte complex, des de l'anàlisi, la cerca d'informació, l'estructuració, la realització i finalment la presentació, sigui per la via que sigui.

2.3 Enfocament i planificació

L'objectiu final d'aquest treball és la realització d'un cas pràctic: la realització d'una wiki semàntica a partir d'una ontologia que haurem dissenyat.

Per arribar a aquest objectiu, primer haurem d'aprofundir en la seva base teòrica. Començarem tractant el llenguatge XML[2], que és un llenguatge estructurat que dóna

[2] XML: En anglès " Extensible Markup Lenguage" ,Llenguatge de marques extensible .

una sintaxi extensible, sense restriccions, per la creació de llenguatges Web. Ampliarem aquest estudi amb el XML-Schema, que permet imposar-hi restriccions, afegint vocabulari i estenent el tipus de dades.

A continuació ens centrarem en el RDF[3], que ja introdueix una semàntica als documents XML. RDF-Schema, a part, afegeix classes i propietats al llenguatge RDF. Finalment, a partir d'aquests dos llenguatges, estudiarem el OWL[4], que ens ajudarà a la creació d'ontologies basades en Web.

Un cop s'hagi aprofundit en aquesta base teòrica, es dissenyarà una Ontologia amb l'ajuda del programari lliure PROTÉGÉ, i finalment, es crearà una petita web semàntica utilitzant eines com la Ontowiki.

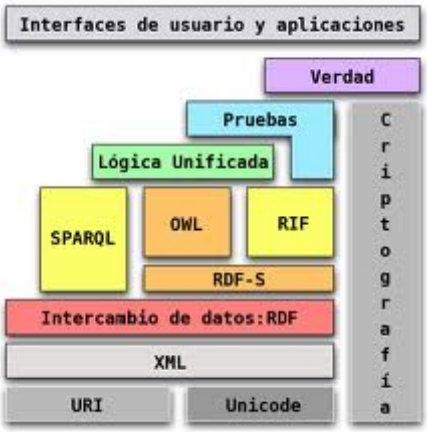

Figura 1: Pila semàntica

3. XML I XML-SCHEMA

XML és un llenguatge basat en el marcatge, aquest fet és bàsic perquè hi hagi un bon intercanvi de informació a la web. Es caracteritza per estructurar la informació que tenim a la web, facilitant d'aquesta forma, un intercanvi i reutilització de dades i documents en aquest medi. És tan important que ha arribat a ser l'estàndard a la web.

La paraula que defineix més el llenguatge XML és "extensible", és a dir, que no està limitat a un vocabulari o utilització en particular sinó que es pot adaptar als propòsits als que es vulgui aplicar.

[3] RDF: En anglès "Resource Description Framework", llenguatge de descripció del W3C.
[4] OWL: En anglès "Ontology Web Lenguage", llenguatge de marques per publicar i compartir dades usant la WWW.

3.1 Orígens del XML

El XML va ser creat per el W3C[5] com una evolució del SGML[6] i per acabar el problema que generava les mancances del HTML[7].

Els principals problemes que presenta HTML són:

- És un llenguatge pensat per mostrar-se per navegador, no per disseny o impressió.
- No pot mostrar contingut dinàmic.
- L'estructura està molt barrejada amb el disseny. No és exigent amb la estructura lògica de la pàgina.
- El número de etiquetes és limitat.

Per solucionar totes aquestes mancances es crea el XML, un metallenguatge (permet definir llenguatges de marcat segons les necessitats) obert a llenguatges i S.O. que es dissenya per estructurar i descriure dades de forma que puguin ser compreses per diferents aplicacions. Un metallenguatge extensible. En definitiva, és un metallenguatge que proporciona una sintaxi a la creació de vocabularis web i, facilita el intercanvi i reutilització de dades i de documents estructurats.

3.2 Sintaxi del XML

El documents XML es caracteritzen per estar composats de 2 elements: les entitats i els tags (etiquetes). Les entitats disposen del contingut del document i els tags serveixen per estructurar i descriure aquestes entitats. Per diferenciar les marques de les entitats, les delimitarem amb uns caràcters especials que anomenem *delimitadors*, aquests solen ser els signes "<" o ">".

```
<NOM_ETIQUETA> Contingut <\NOM_ETIQUETA>
```

Fig.2 Exemple estructura simple XML.

Després d'una etiqueta d'inici " <NOM_ETIQUETA> " hi haurà el contingut i tot seguit la etiqueta de final o tancament " <\ NOM_ETIQUETA> ". Aquestes etiquetes també poden tenir atributs que proporcionaran informació addicional sobre elles mateixes o sobre el contingut.

```
<NOM_ETIQUETA nom_atribut="valor_atribut"> Contingut <\NOM_ETIQUETA>
```

[5] W3C: World Wide Web Consortium. Consorci internacional que genera la recomanacions per la www.
[6] SGML: Standard Generalized Markup Language.
[7] HTML: HyperText Markup Language.

Fig 3. Exemple estructura simple XML amb atribut.

També es poden afegir comentaris al document utilitzant els delimitadors “<!-->” i “<-->” com mostra el següent exemple:

```
<!—Això és un comentari -->
```

Fig 4. Exemple comentari XML

Un cop hem definit les etiquetes bàsiques cal definir la estructura del document. Aquest és composa d’un pròleg i el contingut en si del document. El pròleg conté informació sobre la versió XML del document i opcionalment, pot contenir informació sobre el tipus de codificació emprada, tipus de declaracions XML,...

El contingut del document està organitzat jeràrquicament en forma de arbre invertit, de forma que etiquetes definides poden incloure altre etiquetes. Aquestes però, sempre han de estar ben estructurades. Així per exemple si tenim una estructura com:

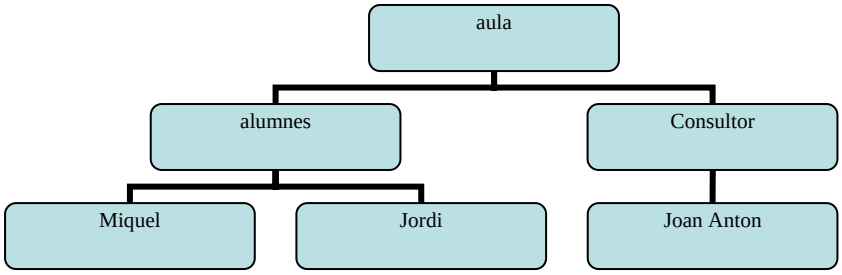

Fig 5. Exemple estructura arbre “aula”

Una possible codificació XML seria:

```
<?xml version=”1.0”?>
<!—exemple codificació simple XML-->
<aula>
        <alumnes>
                <nom> Miquel <\nom>
                <nom> Jordi <\nom>
        <\alumnes>
        <consultors>
                <nom email=”janton@uoc.edu”>  Joan Anton <\nom>
        <\consultors>
<\aula>
```

Fig 6. Exemple codificació XML “aula”.

3.3 Espai de noms en XML

L'estàndard del espai de noms en XML apareix el 1999 i proporciona les regles per associar noms d'elements i atributs de XML amb una col·lecció de noms d'elements i atributs associats a un URI[8]. L'objectiu de l'espai de noms era que és pogués construir documents XML reutilitzant elements d'altres documents, sense que hi hagués problemes de duplicitat de noms. Es pot donar el cas que diferents documents tinguin elements amb el mateix nom, però amb diferent significat. Aquest estàndard permet referir-se inequívocament a un element d'un vocabulari determinat.

Per definir un espai de noms cal utilitzar la següent notació:

```
<nom_element xmlns:prefix="url_del_espai_de_noms">
```

Fig 6. Exemple espai de noms

Per crear elements o atributs que es troben en l'espai de noms de l'exemple 6, utilitzarem el nom que haurem definit per a referenciar-ho. En l'exemple, utilitzarem *prefix*.

3.4 DTD i XML-Schema

Tots els documents XML han de complir una norma bàsica. A part de ser ben estructurats, han de ser vàlids. Per complir-ho han e seguir les normes bàsiques d'uns dels dos estàndards que existeixen: els documents DTD[9] o bé, els documents XML-Schema. El W3C va recomanar en primera instància la utilització de les DTD fins la aparició de XML-Schema, des de llavors ha estat l'estàndard proposat per aquest organisme.

3.4.1 DTD

Les DTD estableixen les normes que ha de seguir un document perquè sigui vàlid, és a dir, especifica la estructura que ha de seguir i quines restriccions existeixen. Una DTD indica els elements que poden aparèixer al documents, quins atributs pot utilitzar, quin ordre segueixen els seus elements fills, si poden estar buits o no,....

Les DTD es poden definir dins el propi document o bé es poden definir en un altre document i fer-ne referència des del document XML, aquest últim sistema fa que aquesta DTD pugui ser utilitzada per molts altres documents XML.

[8] URI: en anglès "Uniform Resource Identifier", identificador uniforme de recursos.
[9] DTD: en anglès "Document Type Definition"; definició de tipus de document.

```
<!DOCTYPE nom_DTD SYSTEM “ url dtd”>
```

Fig 7. Exemple definició externa DTD

El problema de les DTD és que tenen una sèrie d’inconvenients:

- Tenen una sintaxi pròpia diferent a la de XML.
- No suporten el espai de noms.
- Ofereixen tipus de dades limitats

Per millorar el sistema de validació de les DTD, es va treballar en els seus punts febles i es va dissenyar un nou sistema: el XML-Schema.

3.4.2 XML- Schema

A partir de mitjans del 2001, XML-Schema desbanca a les DTD com a estàndard per les definicions de tipus de documents i dades, i com no, per la validació de documents XML. La nova solució resolt els problemes que tenien les DTD, ja que com hem dit, aquest últim es va dissenyar tenint en compte totes les mancances que presentaven les DTD.

Les principals característiques per les quals XML-Schema queda com a nou estàndard són les següents:

- Usa la sintaxi de XML, és a dir, utilitza el mateix llenguatge per les definicions que pel document en si.
- Permet utilitzar el espai de noms (namespaces).
- Augmenta significativament el nombre de tipus de dades permeses, a part que permet crear-ne de noves.
- Permet usar conjunts d’elements, fent que els elements no tinguin un ordre fix a l’estructura.
- Són extensibles.

Amb aquest nou pas augmenta significativament el tractament de les dades però encara ens fa falta un element molt important. Aquestes dades estan molt més estructurades però no estan dotades de sentit semàntic. És a dir, encara ens faltar relacionar dades i continguts segons els seu significat. Si s’aconsegueix aquest punt, es podrà relacionar els elements segons el seu significat i facilitarem d’aquest mode, l’automatització de processos en el tractament d’aquestes dades.

4. RDF i RDF-SCHEMA

4.1 RDF

El RDF és llenguatge per a la descripció de recursos que es caracteritza per aportar significat als documents XML.

La sentencia RDF es caracteritza per tenir la estructura bàsica d'una oració, és a dir, d'estar formada per un subjecte, un objecte i un predicat que serà el que relacionarà el subjecte i l'objecte. A diferencia de les oracions del llenguatge, enlloc d'estar formades per paraules, estan formades per URIs. Per exemple podríem tenir la següent tripleta:

<http://joanmas.cat> <http://agradar.exemple/termes/agradar> <http://www.w3.org/people/Berners-Lee/weaving>

Fig 8. Exemple tripleta RDF

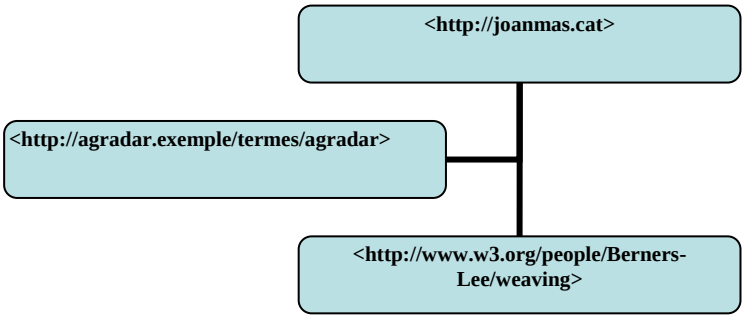

Fig 9. Esquema figura 8

En aquest exemple tenim el subjecte, que serà el primer URI, en aquest cas en *Joan Mas*. El segon URI seria el predicat, que seria agrada. I el tercer URI seria el objecte, en aquest cas el llibre "*Weaving de web*" de Berners-Lee. Per tant, relacionaríem el subjecte i l'objecte amb el verb *agradar*, de tal manera que tindríem que en Joan Mas li agrada "Weaving de web".

Un cop tenim definida la estructura bàsica cal definir una sintaxi per emmagatzemar i transferir aquesta informació. Hi ha varis models però el més utilitzat és el RDF/XML que ens permetrà aprofitar tots les recursos existents per XML. L'exemple de la figura 8 ens quedaria de la següent manera:

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:predicat="http://agradar.exemple/termes/agradar">

    <rdf:Description rdf:about="http://joanmas.cat"
          predicat:llibre=" http://www.w3.org/people/Berners-Lee/weaving"/>

</rdf:RDF>
```

Fig 10.Exemple RDF/XML

Com podem veure, escriure en RDF no és una cosa trivial. Per treballar en aquest llenguatge, el més normal és que la informació es tregui dels milions de bases de dades que tenim a la web. Quan tenim la informació guardada en una base de dades, és relativament senzill que una màquina accedeixi a aquesta informació, és a dir, que hi hagi un processament automàtic de la informació.

En aquest punt veiem una de les grans utilitats del RDF, poder publicar aquestes bases de dades a la web.

Amb el RDF hem aconseguit establir unes relacions entre els diferents elements de què disposem, ara ens faltaria afegir informació semàntica a aquestes relacions, establint una jerarquia entre aquestes. En aquest punt s'introdueix el RDF-Schema.

4.2 RDF-Schema

RDF-Schema es va crear partint de RDF, per dotar de un vocabulari per treballar amb les classes i jerarquies. Amb RDF-Schema podem crear jerarquies, especificant propietats i relacions entre elles, i restriccions als elements, cosa que fins ara no podíem fer. RDF-Schema també es codifica amb el llenguatge XML.

Les primitives de RDF-Schema s'especifiquen amb el prefix *rdfs*, a diferencia de les de RDF que ho feien amb el prefix *rdf*. A continuació descriurem les primitives més importants que utilitza:

PRIMITIVA	TIPUS	DESCRIPCIÓ
Resource	rdfs	Tots els recursos descrits en sentències RDF.
Literal	rdfs	Representa la classe dels valors literals com enters, strings,...
Class	rdfs	Correspon al concepte genèric de *type*. Els membres de la classe s'anomenen instàncies de la classe.
type	rdf	S'usa per definir que un element pertany a una determinada classe.

subClassOf	rdfs	S'usa per crear jerarquies. Defineix que una classe es una subclasse d'una altra.
Property	rdf	Defineix els recursos que són propietats RDF
subPropertyOf	rdfs	Defineix una propietat que és una especialització d'una altra.
range	rdfs	Estableix que els valors d'una propietat determinada són instàncies d'una o diverses classes especificades.
domain	rdfs	Indica que els recursos d'una determinada propietat són instàncies d'una o diverses classes especificades.

Hem definit les primitives més importants tots i que n'hi han d'altres com *comment* (per posar comentaris), *label* (per posar etiquetes),...

Podem veure un diagrama com exemple de com RDF-Schema utilitza el vocabulari per descriure classes i propietats:

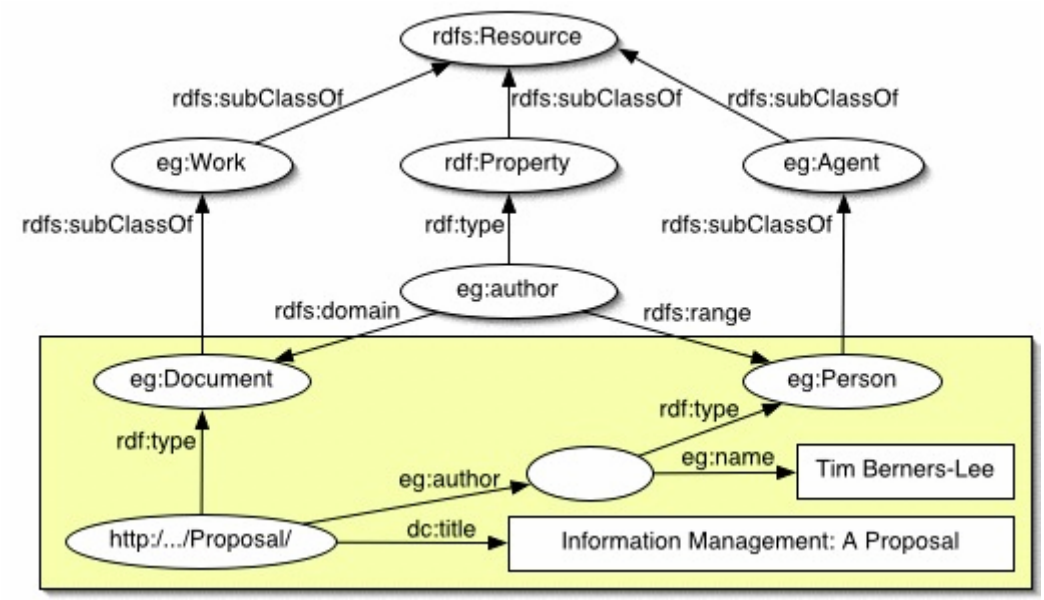

Fig 11. Exemple extret de : http://www.w3.org/TR/2002/WD-rdf-schema-20021112/#ch_class

En aquest exemple es poden veure diferents relacions de propietat, com jerarquies: trobem subclasses com que un document és una subclasse de treball o que una persona és una subclasse de agent. També es distingeixen relacions de propietat, com que un document pertany a un autor. O també relacions de rang, en què un autor està dins el rang de persona.

5. ONTOLOGIES I OWL

Fins ara, tots el punt que hem anat tractat s'han dirigit per tenir la suficient base teòrica per el disseny de la ontologia proposada en el cas pràctic. En aquest apartat es tractaran les ontologies pròpiament dites i el llenguatge OWL, que és un llenguatge que permet definir ontologies a la web.

Cada vegada és més necessària una web semàntica, evolució lògica de la web, que ens estructuri la informació, ens hi afegeixi significat i la faci extensible. La web ha evolucionat cap a aquest punt fent que sigui viable la interacció de les màquines amb els humans i amb la pròpia web. Què ens aporten les ontologies en aquests marc? Doncs, ens aporten semàntica a les dades i ens les relacionen, perquè els processos de tractament d'informació puguin ser automatitzats.

5.1 Ontologies

Ontologia prové dels mots grecs *Onto* (ser, estar) i *logia* (ciència, estudi), és a dir, és l'estudi de ser, de les entitats, i de les relacions entre elles. En definitiva, ve a definir els termes que s'usen per descriure i representar una àrea de coneixement.

En informàtica, defineix els vocabularis perquè les màquines els puguin processar. És bàsic que aquest vocabularis estiguin ben definits i relacionats entre ells de forma inequívoca per poder-ho fer. La IA[10] va desenvolupar les ontologies als anys 80 per facilitar la compartició i reutilització del coneixement.

Les ontologies estan formades per classes. En aquestes classes hi ha de constar una descripció formal i explícita de la classe, les seves propietats i les restriccions que tindran aquestes propietats. Un cop especificades i relacionades totes les classes, resulta un mapa conceptual del domini donat.

Existeixen diferents aplicacions on es poden utilitzar ontologies com:

- Cerca a Internet: permeten tenir la informació estructurada perquè la cerca sigui més eficient.
- Biomedicina: és un dels camps on s'ha treballat més en ontologies darrerament. S'han creat molts recursos per compartir coneixement entre professionals, des de ontologies en dominis com les relacions entre proteïnes i gens, fins a simples tractament d'informació bàsica hospitalària.

[10] IA: Intel·ligència Artificial.

- Biblioteques digitals: Ens permeten estructurar de forma més eficient el nostre catàleg digital, i fer-lo més accessible i entenedor.
- Comerç electrònic: Es poden usar ontologies en la descripció d'un domini, en aquest cas comercial, i així disposar de descripcions d'usuaris, tendències i relacions entre ells per poder fer estudis de mercat i facilitar l'agrupació d'usuaris (clustering) amb similars gustos o preferències.

Les ontologies requereixen d'un llenguatge lògic i formal per representar-se, el OWL per exemple.

5.2 OWL

OWL és un llenguatge per representar ontologies desenvolupat per W3C. És basa en el DAML-OIL[11] i és una extensió del RDF i RDF-Schema. A partir del febrer del 2004 és la recomanació del W3C.

Té com a objectiu facilitar un model de marcat construït sobre RDF i codificat en XML. Utilitza referències URI per a fixar els noms i amplia el vocabulari que oferia RDF i RDF-Schema per descriure propietats i classes. La gran diferència amb el que hem vist fins ara és que OWL és un llenguatge dissenyat per processar informació automàticament, no només per presentar-la als humans.

Les ontologies d'OWL tenen les següents característiques:

- Capacitat de ser distribuïble per varis sistemes.
- Escalabilitat segons les necessitats de Web.
- Compatibilitat amb els estàndards de la Web per proporcionar accessibilitat i internacionalització.
- Obertura i extensibilitat.

El llenguatge OWL està format per tres subllenguatges amb nivells d'expressivitat i complexitat creixent depenent les necessitats. Cada subllenguatge és una extensió del seu predecessor, tal com es mostra a la figura següent:

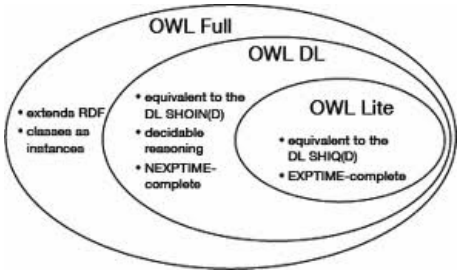

Fig 12. Subllenguatges d'OWL

[11] DAML-OIL: llenguatge semàntic de marcat per a recursos Web.

5.2.1 Subllenguatges d'OWL

El llenguatge OWL es divideix en tres subllenguatges segons la seva complexitat i expressivitat, ordenats de menor a major tenim:

- **OWL Lite**: Dissenyat per a usuaris amb els requisits més bàsics com ara establir classificacions jeràrquiques i restriccions de cardinalitat (només permet valors 0 i 1). També es caracteritza per la seva facilitat d'integració i compatibilitat amb altres models ontològics i per la seva simplicitat per al desenvolupament de programari.

- **OWL DL:** Dissenyat per usuaris que necessiten màxima expressivitat en temes semàntics i alhora, necessiten que els seus dissenys conservin la completesa computacional, és a dir, totes les seves conclusions seran tractades en un temps finit.

- **OWL FULL:** És el més complet dels tres i el que els engloba a tots els altres. Està dissenyat per als usuaris que necessiten màxima expressivitat en temes semàntic sense garanties computacionals. És l'únic subllenguatge que permet estendre el vocabulari predefinit del sistema, això comporta algun inconvenient, com que no tingui un programari completament compatible amb les seves funcionalitats.

A continuació descriurem les característiques del llenguatge OWL Lite, i després els afegits que incorporaran OWL DL i OWL FULL..

OWL LITE:

En aquest punt presentarem una petita descripció de les característiques de OWL Lite, aquest subllenguatge només utilitza alguna de les característiques que ofereix el llenguatge OWL, i està molt relacionat amb el RDF. Alguna de les característiques que relacionen RDF amb OWL Lite podrien ser:

- **Class:** serveix per definir un grup de individus que comparteixen unes mateixes propietats. Podem definir una classe ***thing***, que seria una superclasse que inclouria totes les altres i ***nothing***, una classe sense instàncies o sigui una subclasse de totes les altres.
- **rdfs:SubClassOf:** per crear jerarquies de classe.
- **rdf:Property:** les propietats serviran per establir relacions entre individus o entre individus i valors. Les primeres, que relaciones instàncies de classe entre si, les anomenem: ***ObjectProperty*** ; i les que relaciones instàncies de classe amb valors: ***DatatypeProperty***.

- **rdfs:SubPropertyOf:** per fer subpropietats d'altres propietats.
- **rdfs:Domain:** Un domini redueix els individus als que pot aplicar-se una propietat. Com que es refereix a la propietat, ho podem considerar com una restricció global.
- **rdfs:Range:** serveix per limitar els individus que pot tenir una propietat com a valor.

Les següents característiques estan relacionades amb l'equivalència entre propietats:

- **equivalentClass:** serveix per definir classes equivalents, en aquest cas, les instàncies de seran les mateixes per classes equivalents. És útil per notar sinònims.
- **equivalentProperty:** per definir dos propietats com equivalents.
- **sameAs:** defineix dues instàncies com iguales.
- **differentFrom:** per definir una instància com diferent de les altres.

En OWL Lite també és possible incloure restriccions de classe. Per indicar que hi ha elements restringits usarem la sintaxi **owl:Restriction**. Ara si usem la sintaxi **owl:onProperty**, indicarem la propietat restringida.

En quant a restriccions, podrem tenir-ne que ens indicaran els valors que podem utilitzar: **AllValuesFrom** o **SomeValuesFrom**. O bé la quantitat de valors que podem utilitzar: **minCardinality** (indica el mínim d'elements amb què es pot relacionar) o el contrari, **maxCardinality**.

OWL DL i OWL FULL:

OWL DL i OWL FULL, utilitzen el mateix llenguatge però OWL DL té alguna restricció que no existeix en OWL FULL. Una de les restriccions que existeixen és que OWL DL requereix una separació per tipus, és a dir, una classe no pot ser in individu o una propietat i una propietat no pot ser una classe o un individu. Això ens implica que OWL DL no permet restriccions en elements del llenguatge, al contrari que OWL FULL.

A part, OWL DL, obliga que totes les propietats que s'utilitzen, o siguin ObjectProperty o bé DatatypeProperty.

A part de la sintaxi que hem descrit per OWL Lite, en aquest punt s'hi afegeix el següent:

- **OneOf:** per crear classes enumerades.
- **hasValue:** per valors de propietat, és adir, un individu podria ser el valor d'una propietat.
- **disjointWith:** Una classe pot ser totalment disjunta a una altre (ex:mascle, femella).
- **UnionOf, ComplementOf, IntersectionOf:** per classes booleanes.

6. APLICACIÓ PRÀCTICA

En aquest apartat es crearà una wiki a partir d'una ontologia que haurem dissenyat prèviament. Per fer-ho s'utilitzarà un programari específic: el Protégé, per crear la ontologia i Ontowiki per crear la wiki.

Abans de procedir a les especificacions de disseny pròpiament dites, farem una breu introducció a aquest programari específic pel disseny d'Ontologies, el Protégé.

6.1 Protégé

Protégé és un entorn gràfic gratuït pensat per la creació i edició d'ontologies i bases de coneixement, va ser creat per la Universitat de Stanford amb col·laboració amb la de Manchester.

Està pensat per treballar a base de plug-ins, això fa que sigui molt extensible. Un d'aquests plug-ins permet utilitzar el llenguatge OWL per crear ontologies. Altres plug-ins, com veurem més endavant, serveixen per visualitzar gràficament el nostre treball.

Està escrit en java, això implica que es pugui utilitzar en diferents plataformes independentment del sistema que s'utilitzarà. L'únic requisit és tenir instal·lada una plataforma pel desenvolupament de software en java com el JDK. Utilitza una interfície gràfica molt entenedora i fàcil d'usar, ja que és una aplicació que utilitza molt els swings[12] que ofereix el llenguatge java.

Protégé enregistra els projectes en forma de base de dades, obrint la possibilitat a reutilitzar el gran ventall de bases de dades relacionals que existeixen actualment. També té la possibilitat de exportar els projectes en altres formats, a part del OWL, com el format RDF.

6.1.1 Instal·lació

Per a la instal·lació d'aquest programari, cal accedir a la pàgina web que hi té destinada la Universitat de Stanford, http://protege.stanford.edu/download/download.html . En aquesta pàgina caldrà registrar-se en cas que no si estigui, i ja es pot descarregar la versió que s'esculli. Cal destacar que també ens dóna la opció d'instal·lar-nos la plataforma per Java si és que no es disposa d'ella.

[12] Swing: Biblioteca gràfica en el entorn Java

Actualment hi ha possibilitat de descarregar-se dues versions del programa, la 3.X i la 4.0. Després de estudiar les dues versions de què es disposa , http://protegewiki.stanford.edu/wiki/Protege4Migration#Side_by_Side_Comparison, per aquest treball es decideix instal·lar la versió 3.4.7 ja que permet treballar en RDF i OWL, i suporta el llenguatge SPARQL.

6.1.2 Funcionalitats del programa

En aquest apartat descriurem les funcionalitats bàsiques del programa Protégé, que ens permetran dissenyar la ontologia de l'aplicació pràctica. El primer que ens permet quan creem un nou projecte és escollir el tipus que s'utilitzarà en el treball i el URL de la Ontologia:

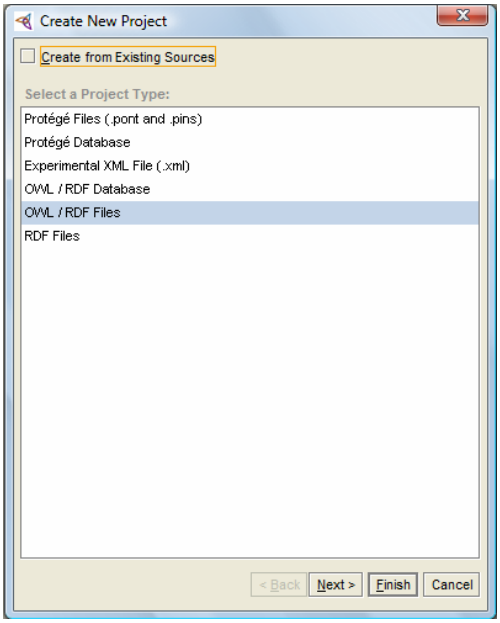

Fig.13: Finestra selecció tipus

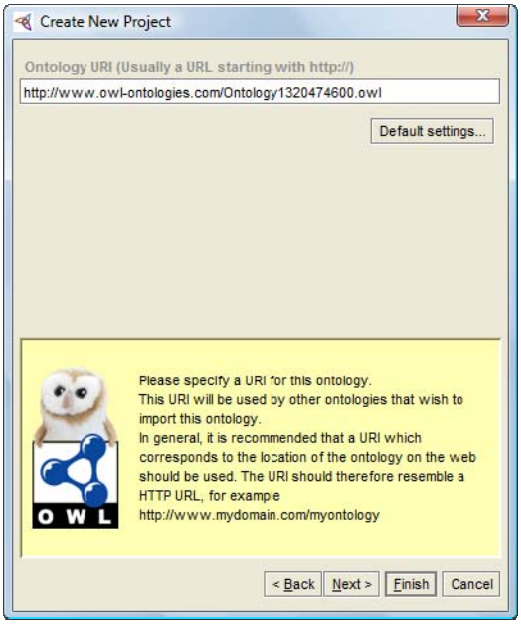

Fig. 14: Finestra selecció URI

En el nostre cas s'escull el tipus OWL/RDF files ja que la nostra ontologia la dissenyem en el llenguatge OWL. El URI escollit serà : www.owl-ontologies.com/Ontology1321475933.owl. Tot seguit s'escull el subllenguatge d'OWL, en el nostre cas i després de fer un estudi de la ontologia a crear, ens decantem per el OWL-DL, ja que amb el subllenguatge més petit no en faríem prou alhora de traspassar la ontologia a una wiki.

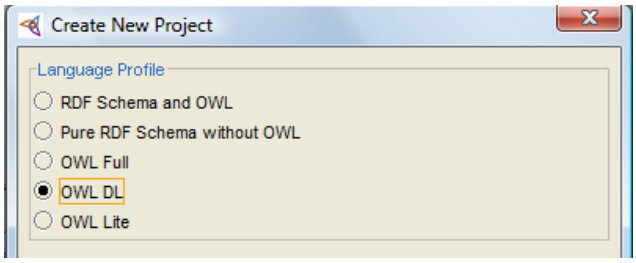

Fig. 15:Finestra selecció Subllenguatge

Finalment s'escull si es vol treballar de forma lògica o amb propietats, en aquest treball usem aquesta darrera opció.

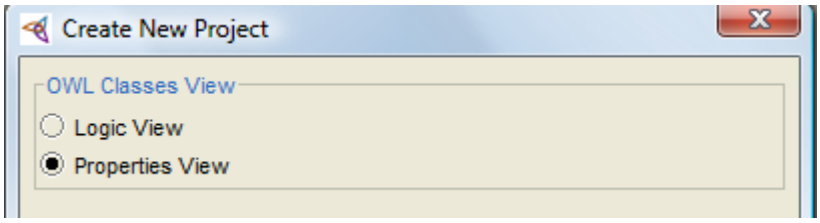

Fig. 16: Finestra de selecció de vista

Un cop tenim el projecte definit, es comença la creació de la ontologia. El programa Protégé disposa d'una interfície molt clara on es poden diferenciar diferents pestanyes per les classes, propietats (Properties) i instancies (Individuals).

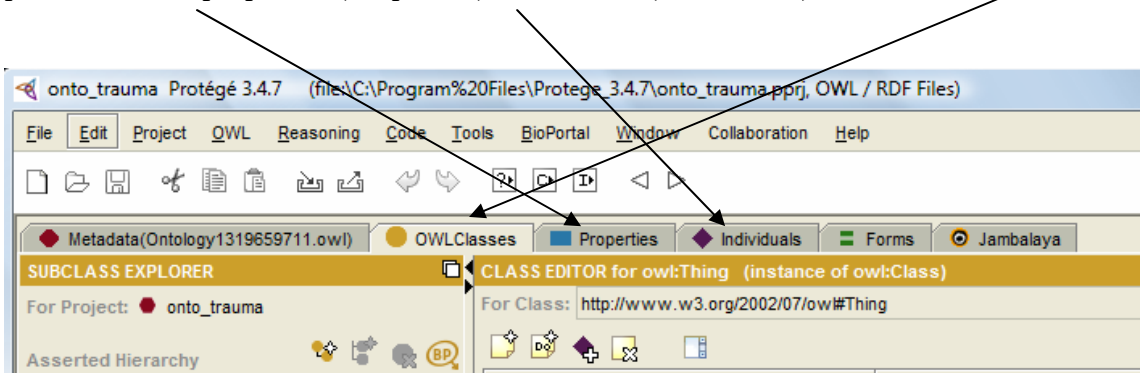

Fig. 17: Vista de les diferents pestanyes.

- **<u>CLASSES:</u>**

En la pestanya de les classes hi trobem l'editor de classe i el *Subclass Explorer*. Aquest últim ens servirà per crear les classes i les subclasses, mentre que l'editor s'usa per introduir comentaris, les condicions i establir les disjuncions entre classes.

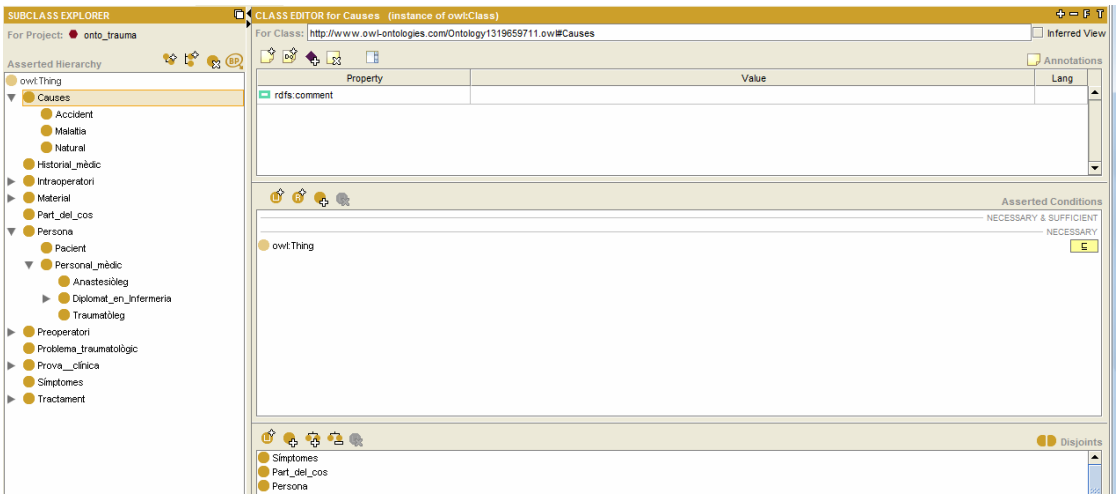

Fig. 18: Finestra de la pestanya de classes

Per crear les classes i subclasses tenim les següents icones al *Subclass Explorer*:

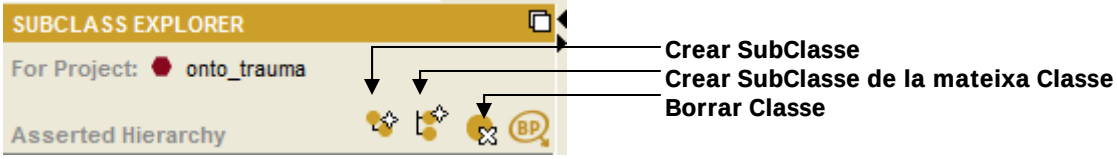

Fig. 19: Icones per crear classes i suclasses

- **<u>PROPIETATS:</u>**

A la pestanya de propietats és on es defineixen les propietats de la nostra ontologia. Com ja es va definir en apartats anteriors, les propietats relacionen un domini amb un rang, és a dir, funcionen com un verb del llenguatge i relacionen el subjecte (Domini) amb un predicat (Rang).

En aquesta pestanya tenim dos apartats diferenciats: una és el *Property Browser*, que serveix per crear propietats i subpropietats. En aquesta apartat es diferencien les ObjectProperties amb les DataProperties en dues pestanyes diferents.

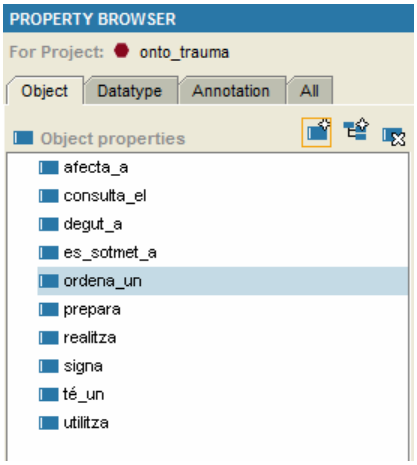

Fig. 20: Finestra ObjectProperties

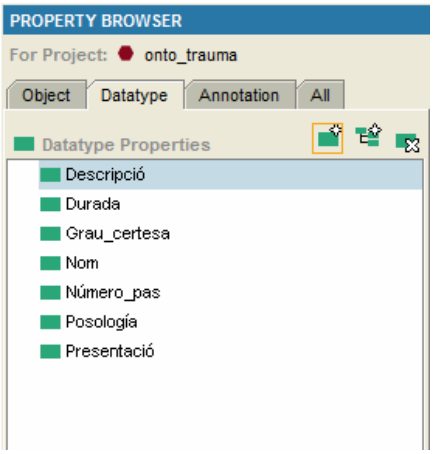

Fig. 21: Finestra DataProperties

I l'altra l'editor de propietats que permet relacionar les propietats amb els seus rangs i dominis.

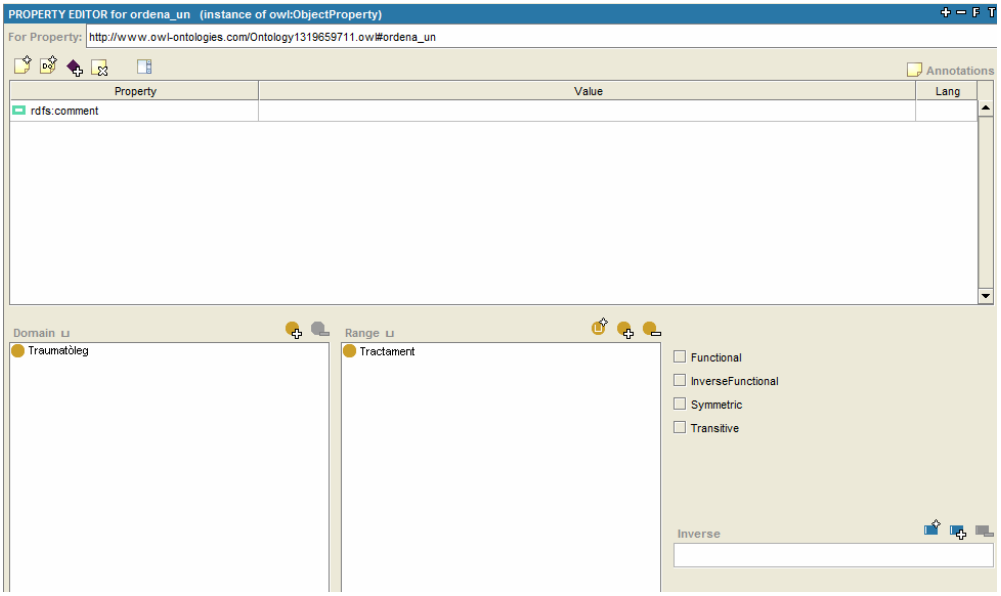

Fig. 22 : Editor de propietats

- **<u>INSTÀNCIES:</u>**

Finalment la pestanya d'Instàncies ens servirà per definir les possibles instàncies de cada classe:

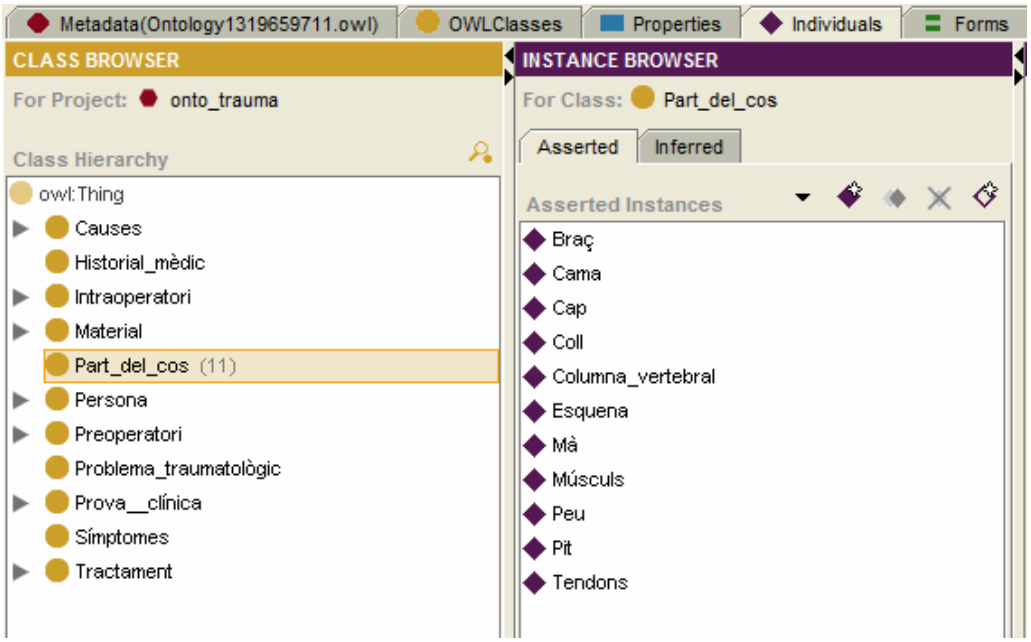

Fig. 23: Pestanya Instàncies.

A part de les funcionalitats bàsiques per crear una ontologia, Protégé disposa d'utilitats per realitzar, per exemple, la exportació del nostre projecte a altres formats, de la forma que es mostra a la figura següent:

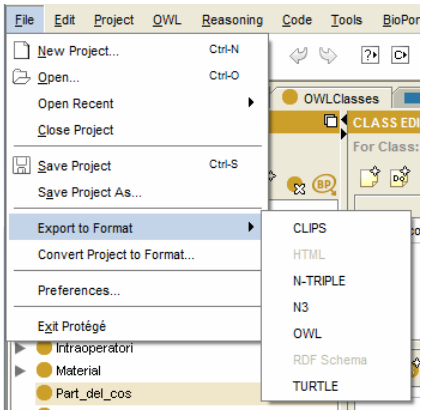

Fig. 24: Exportació a altres formats

O la introducció de nous plug-ins accedint al menú Projects->Configure i seleccionar els plug-ins que volem incorporar al projecte:

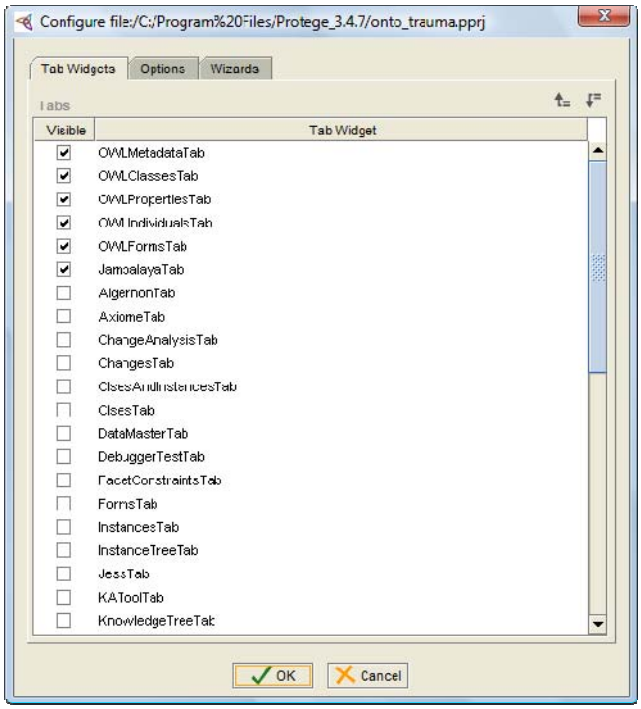

Fig. 25: Finestra de selecció de plug-ins

6.2 Disseny de la ontologia

Alhora de dissenyar la nostra ontologia, el primer que fem un cop tenim l'enunciat (Annex 1) és fer un anàlisi per poder definir les classes i les subclasses de la base de coneixement que volem tractar, en el nostre cas, una operació de traumatologia. Quan cal i com es realitza. Per fer aquest anàlisi, estudiarem les diferents tasques que s'ha de realitzar i qui les realitza. Utilitzarem diagrames de casos d'ús per aquest anàlisi. També caldrà tenir en compte en quin ordre cal realitzar aquestes tasques, un diagrama d'activitats ens ajudarà en aquest punt.

Diferenciem dos apartats principals en el problema: la diagnosi, quan es detecta un problema i el tractament, quan es soluciona.

La diagnosi es caracteritzarà per tenir el següent flux: una o unes causes provoquen un problema traumatològic a una part del cos del pacient, aquest problema generarà uns símptomes. El pacient, un cop té el problema traumatològic, va a la visita del traumatòleg, que després d'un examen físic, ordena un tractament per solucionar el problema.

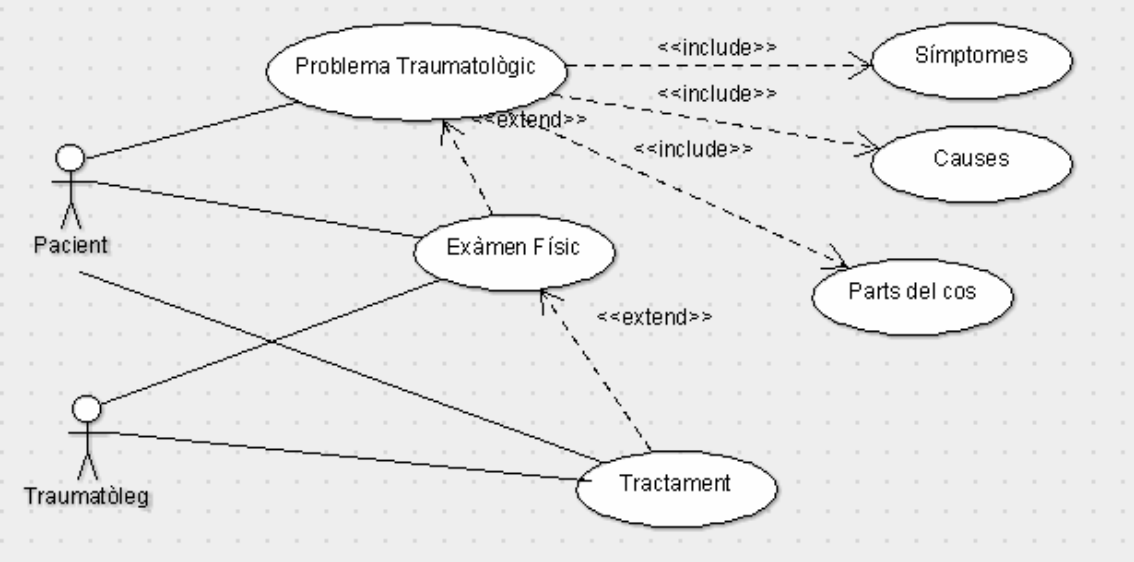

Fig. 26: Diagrama dels casos d'ús de la detecció de problema

El tractament pot ser conservador o quirúrgic. En cas que sigui conservador, el pacient segueix el tractament i si es soluciona es dóna l'alta al cas. En cas que s'hagi de seguir un tractament quirúrgic, diferenciarem dues fases en aquest procés: el preoperatori i el intraoperatori. Realment hi hauria una altra fase, el postoperatori, però aquesta fase queda fora de l'abast d'aquest treball.

Si ens centrem en el preoperatori, diferenciarem dues tasques principals: la consulta amb el traumatòleg i la consulta amb el anestesiòleg, totes dues necessàries i prèvies a la intervenció pròpiament dita.

Cal destacar que és necessari que el pacient signi el consentiment informat en la consulta preoperatòria.

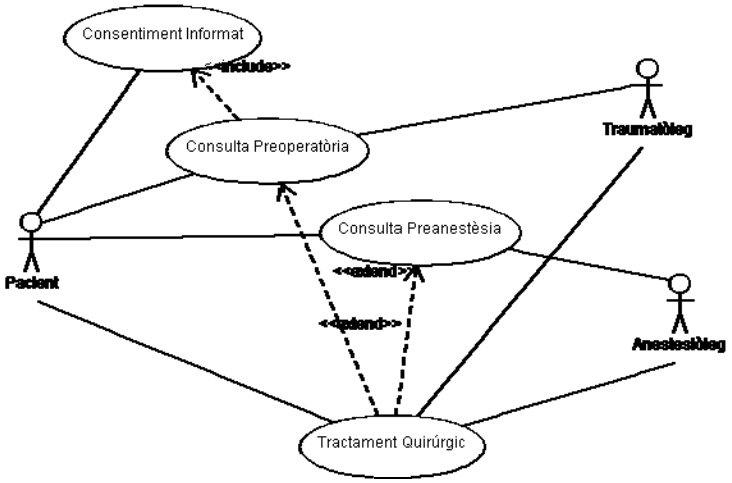

Fig. 27: Diagrama dels casos d'ús del preoperatori

El intraoperatori també tindrà les seves tasques i actors. Aquí caldrà tenir en compte qui realitza cada tasca. Les resumim en el següent diagrama de casos d'ús:

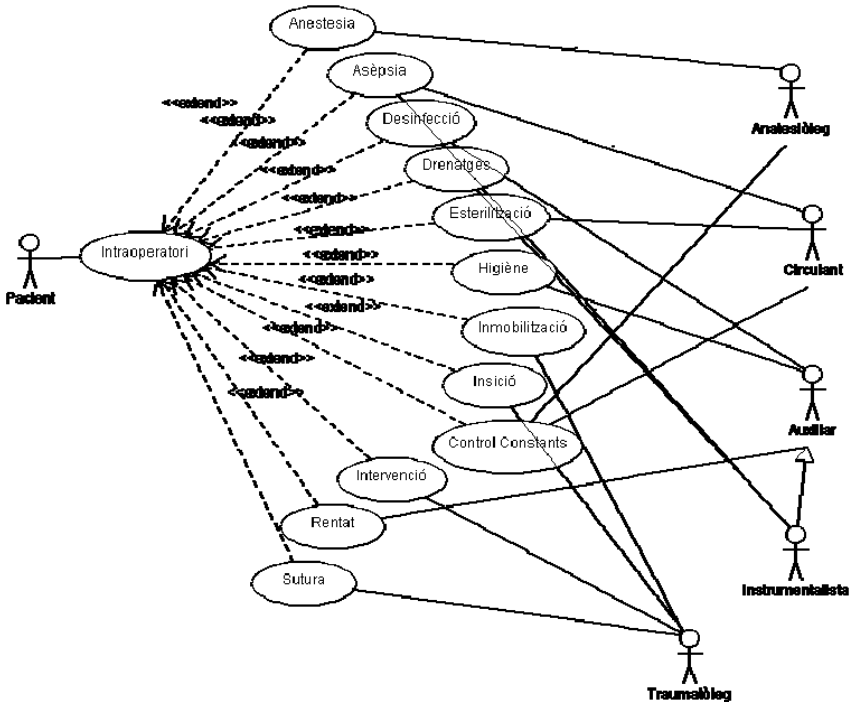

Fig. 28: Diagrama dels casos d'ús del intraoperatori

Un cop tenim definides totes les tasques necessàries i qui les realitza, cal ordenar aquestes tasques segon l'ordre en què s'executen. Ens ajudarem d'un diagrama d'activitats per definir tot el procés:

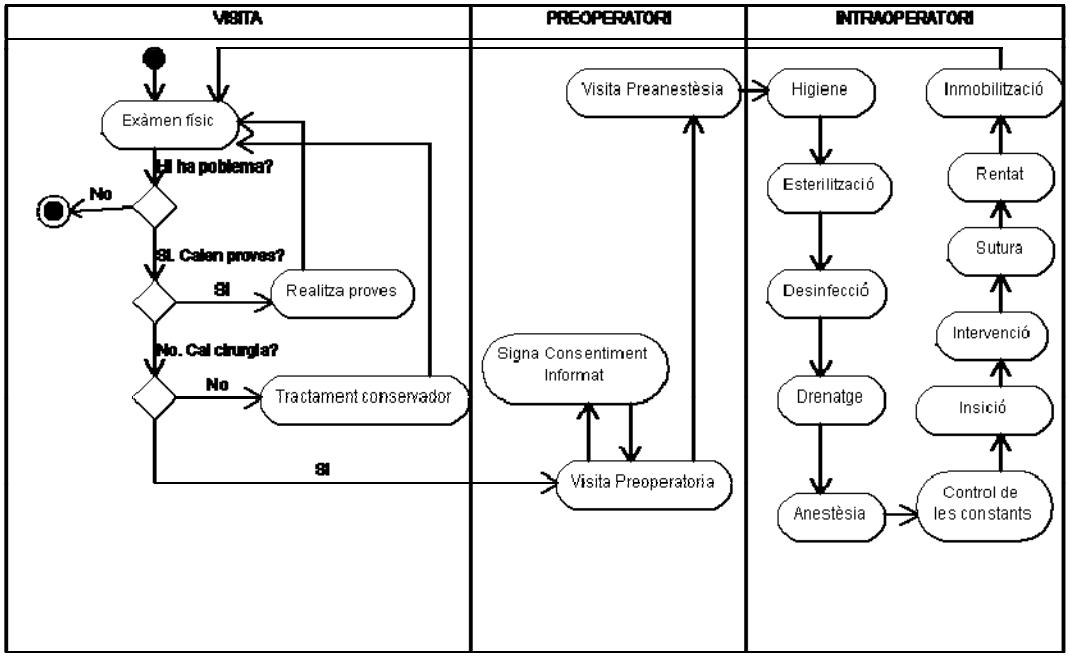

Fig. 29: Diagrama d'activitats

Tenint en compte aquests punts podem definir les classes que formaran la ontologia:

- **Persona:** serà la classe per definir tots els actors que participen a la ontologia, diferenciarem dues subclasses: el **Pacient** i el **PersonalMedic**. El personal mèdic el diferenciarem en **Traumatolegs**, **Anestesiolegs** i **Infermer** (**Circulant**, **Instrumentalista** o **Auxiliar**).

- **ProblemaTraumatologic:** és la classe que defineix el problema en si i que servirà per relacionar part de la ontologia.

- **Causes:** defineix les causes que han provocat el problema traumatològic, dintre aquesta classe trobarem tres possibles subclasses: **Accident**, **Malaltia** o **Natural**.

- **HistorialMedic:** creem una classe 'Historial mèdic' perquè serà una classe que pertany al pacient i s'utilitzarà pel personal mèdic en el seu treball.

- **PartsDelCos:** és la classe per definir la part del cos on hi ha el problema. Diferenciarem diferents subclasses com: **Ossos**, **Musculs**, **Organs**, **Nervis** o **Tendons**.

- **ProvaClinica:** és la classe per definir les proves que ordenarà el traumatòleg un cop es faci la visita. Constarà d'altres subclasses segons el tipus de prova de què es tracti: **ExamenFisic** , **Radiografia** o **Analitica**.

- **Simptomes:** classe que engloba els símptomes que pot patir el pacient.

- **Material:** la classe material la dividirem en instrumental, necessari per les intervencions, taula, implants (perquè n'hi ha de varis tipus i no ho considero instrumental) i carros, ja que en l'enunciat s'especifica que existeixen tres tipus de carros diferents. Pel que fa als implants, hem fet 4 subclasses segons la zona on es col·loca l'implant: **Cadera**, **Columna**, **Genoll** o **Trauma**, aquest últim implica el material necessari per una placa en un fèmur, per exemple, és a dir, cargols, plaques,... que considerem implants perquè de fet, és material extern que s'implanta al pacient.

- **Tractament:** aquesta classe serveix per diferenciar si s'ha d'intervenir quirúrgicament o no. És el tractament que segueix el pacient per solucionar el problema. En cas que no calgui intervenció quirúrgica, diferenciarem quin tipus de tractament conservador es realitza: medicació (oral o intravenosa) o rehabilitació .

- **Preoperatori:** un cop es sap que cal intervenció quirúrgica cal una classe per definir la primera part de la intervenció, els passos previs a l'entrada al quiròfan. En aquesta classe diferenciarem la consulta amb l'anestesista de la del traumatòleg, on es signa el consentiment informat.

- **Intraoperatori:** la classe intraoperatori inclourà totes les etapes de la operació quirúrgica.

A continuació, i gràcies al plug-in JAMBALAYA, adjuntem un esquema de totes les classes i subclasses del disseny.

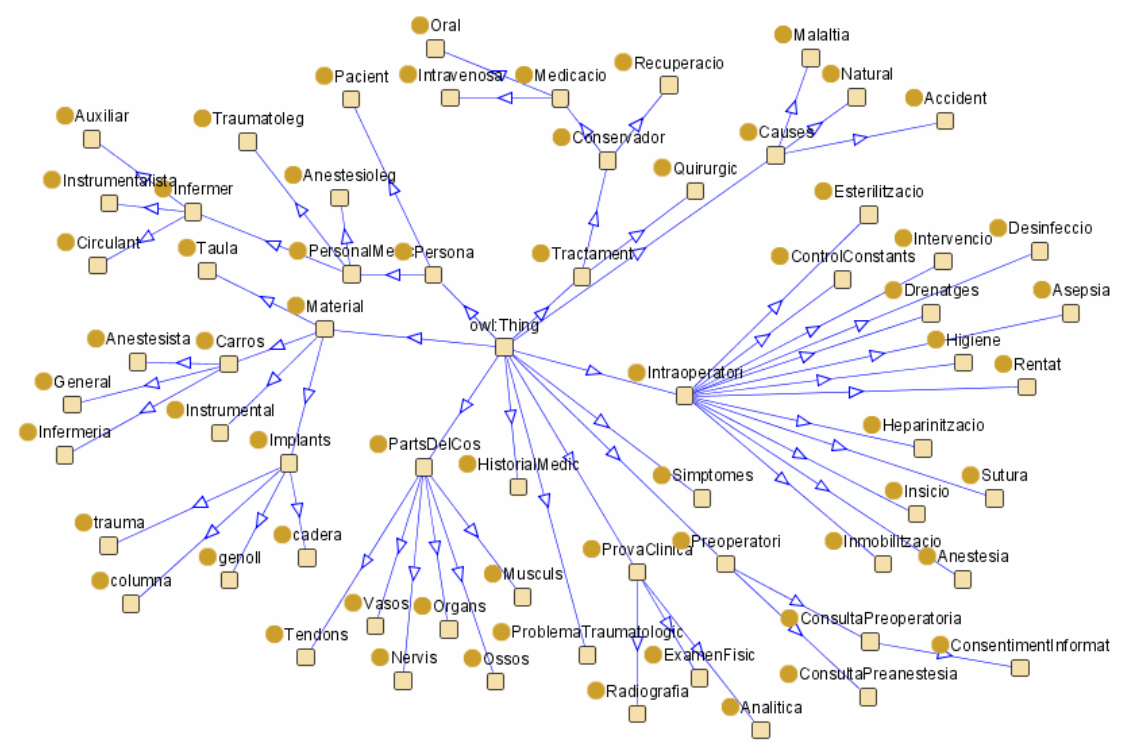

Fig. 30: Esquema de les classes i subclasses.

Un cop tenim les classes jerarquitzades, cal definir les propietats per relacionar-les, entre elles o amb dades concretes, és a dir, cal definir les ObjectProperties i les DataProperties. El mètode que s'ha seguit és un mètode semàntic, a partir d'una classe que utilitzarem de subjecte (Domini), es buscaran possibles enllaços (atributs) per relacionar-los amb els predicats (Rangs) fins que tenim totes les relacions de la ontologia.

Les Object Properties ens serviran per poder especificar qui realitza cada tasca i quin ordre han de seguir en la ontologia.

D'aquesta manera tenim:

OBJECTPROPERTIES:

DOMINI	PROPIETAT	RANG
• ProblemaTraumatologic	• degut	• Causes
• ProblemaTraumatologic	• afecta	• PartDelCos
• Anestesioleg	• consulta	• HistorialMedic
• Traumatoleg		

• Pacient	• essotmet	• ProvesCliniques
		• Tractament
		• Preoperatori
		• Intraoperatori
• Traumatoleg	• ordena	• Tractament
• Instrumentalista	• prepara	• Material
• Traumatoleg	• realitza	• ExamenFisic
		• ConsultaPreoperatoria
		• Inmobilitzacio
		• Insicio
		• Sutura
		• Intervenció
• Pacient	• signa	• ConsentimentInformat
• Pacient	• te	• Simptomes
		• HistorialiMedic
		• ProblemaTraumatologic
• Preoperatori	• anteriora	• Intraoperatori
• ProvaClinica	• nocalcirugia	• Conservador
• ProblemaTraumatologic	• provoquen	• Simptomes
• ProvaClinica	• calcirugia	• Quirurgic
• ProvaClinica	• necessariaper	• Preoperatori
• Anestesioleg	• interveen	• ConsultaPreanestesia
		• Anestesia
		• ControlConstants
• Circulant	• socupade	• Asepsia
		• Esterilitzacio
		• ControlConstants
• Instrumentalista	• esresponsablede	• Rentat
		• Higiene
		• Deseinfeccio
		• Drenatges
		• Asepsia
• Auxiliar	• participaen	• Rentat
		• Higiene
		• Desinfeccio
• PersonalMedic	• utilitza	• Material

Per crear ObjecteProperties anotades anteriorment, hem utilitzat sinònims per diferenciar les tasques que realitza cada actor. Així per exemple, l'anestesiòleg '*intervé*' en tasques, de la mateixa manera que el traumatòleg '*realitza*' tasques o l'auxiliar '*participa*' en unes altres. D'aquesta manera evitem que diferents actors se'ls hi assignin tasques que no el hi pertoquen.

També hem utilitzat verbs per lligar tota la ontologia. Diem que un pacient '*té*' un problema traumatològic, aquest li '*provoca*' uns símptomes. També especifiquem que aquest problema '*és degut*' a unes causes i '*afecta*' una part del cos, per exemple. D'aquesta manera anem lligant tota la ontologia seguint les especificacions que hem detallat en els diagrames anteriors.

En quant a les DataProperties s'ha buscat diferents atributs que ens serveixen per especificar cada domini. De cada classe que tenim detallada, s'han buscat possibles atributs que ens serveixin per definir-la millor i completar-ne el significat. D'aquesta manera tenim que com que ens interessa saber quina durada té un tractament, per exemple, creem una DataProperty amb aquest nom i que ens indicarà mitjançant un enter, la durada del tractament.

La taula ens quedarà de la següent manera:

DATAPROPERTIES:

DOMINI	PROPIETAT	RANG
• Simptomes • ProblemaTraumatologic	• descripció	• String
• Conservador	• durada	• Int
• Causes	• graucertesa	• Boolean
• Persona • ProblemaTraumatologic	• nom	• String
• Persona	• codi	• String
• Taula	• materialdeconstruccio	• String
• Intraoperatori	• numeropas	• Int
• HistorialMedic	• numero	• Int
• Accident	• lloc	• String
• ConsentimentInformat	• signants	• String
• Analitica • ExamenFisic • HistorialMedic	• informe	• String
• Radiografia	• imatge	• Any
• PersonalMedic	• categoria	• String
• Tractament • ProvaClinica • Preoperatori • Intraoperatori	• dataprevista	• DateTime
• Medicacio	• posologia	• String
• Instrumental	• presentacio	• Embossat en doble capa de paper • Doble capa paper sense teixir en una base metàlica

		• caixa tancada amb els corresponents filtres d'esterilització

Cal explicar diferents consideracions que he fet alhora de crear la taula anterior, com per exemple, en el últim quadre tenim la propietat *graucertesa*, que equival a 1 quan se sap del cert la causa del problema i 0 quan no.

També tenim El *numeropas* que afecta el domini *Intraoperatori*, i serveix per indicar el lloc de l'etapa dintre el procés, és a dir, si és el número 1 serà l'etapa que es realitza en primer lloc, 2 en segon, i així progressivament. Aquesta és una manera per ordenar tot el procés de l'intraoperatori.

L'atribut *presentacio* serveix per indicar com està embossat l'instrumental necessari per la intervenció.

Altres DataProperties serien per exemple, *lloc* que ens informa del lloc on es va produir l'accident o l'atribut *dataprevista* , com el seu propi nom indica, la data fixada per la realització de la prova, tractament o operació.

Tenim altres atributs, però són tan trivials que en quadre anterior queden suficientment explicats.

Finalment introduïm les instàncies per acabar de perfilar la ontologia. Primer introduirem diferents malalties que poden provocar problemes traumatològics, com ara: bursitis, fàscies plantar, fractures, hèrnies discals,...

I també s'introdueixen els noms de les diferents subclasses de les parts del cos on hi pot haver el problema, d'aquesta manera, anirem introduint instancies en totes les classes que es pugui, perquè quan es vulgui fer una simulació, aquesta sigui el més complerta possible.

Amb tot, ens quedaria el següent esquema extret amb JAMBALAYA on s'indiquen les objectproperties que hem definit:

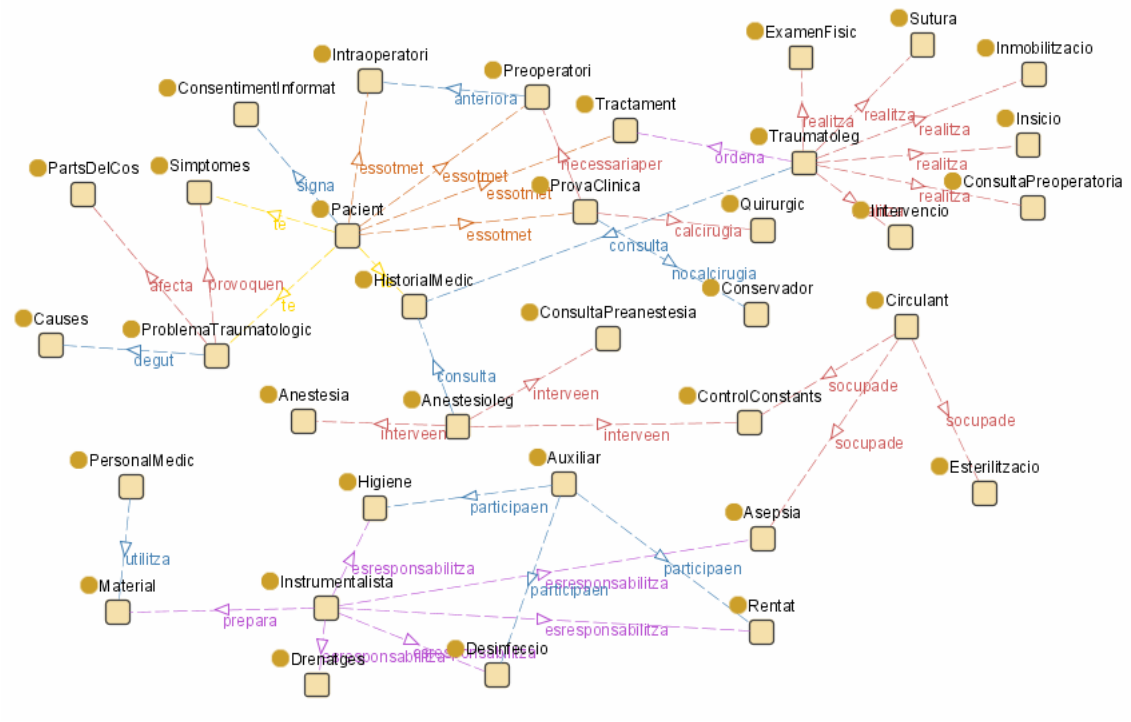

Fig. 31: Esquema bàsic de la ontologia

6.3 Wiki

Un cop tenim dissenyada la nostra ontologia, aquesta serà utilitzada per crear una wiki. Per fer-ho, utilitzarem un programari específic. Abans però, caldria explicar el que entenem per una wiki.

Segons la wikipèdia, "un wiki és un lloc web col·laboratiu que pot ser editat des del navegador pels usuaris", això significa que els usuaris poden crear, modificar, afegir o esborrar la informació d'una web de forma molt interactiva. Tenim molta facilitat alhora d'editar i actualitzar les pàgines.

Les wikis han esdevingut darrerament, una eina molt important per empreses i universitats com a webs i intranets per gestionar bases de coneixement. Des d'aquestes webs es gestionen projectes que poden ser modificats, ampliats o corregits per qualsevol usuari registrat.

Cal destacar que actualment existeixen moltes wikis que es poden actualitzar sense una revisió prèvia. Cada vegada menys, perquè actualment , en quasi totes existeix una pàgina específica amb els canvis recents, que enumera les edicions més recents dels articles o els canvis realitzats últimament. Molts wikis utilitzen aquests canvis per filtrar la informació i desfer els canvis realitzats malintencionadament.

A part de la seva facilitat de edició, una wiki es caracteritza per ser una web relacionada mitjançant enllaços interns amb altres pàgines amb contingut relacionat. Aquesta característica és molt important pel nostre treball, ja que utilitzarem aquesta

característica per relacionar classes mitjançant els atributs que prèviament haurem definit.

L'evolució lògica de les wikis són les wikis semàntiques. Si una wiki conté text més o menys estructurat i hipervíncles no classificats, una wiki semàntica permet treballar amb la informació d'una forma més complexa ja que utilitza metadades de les pàgines i de les seves relacions. Afegeix semàntica al contingut de la wiki.

6.4 Wiki Semàntica

Podem definir una wiki semàntica com una base de coneixement. Si una wiki tradicional estructura la informació enllaçant els textos amb hyperlinks sense considerar el significat de la informació, una wiki semàntica ens estructura la informació a partir de una informació addicional que conté el text, unes metadades, que afegeixen significat a les paraules o el relacionen amb altres paraules.

Aquesta habilitat o facilitat que ofereixen les wikis semàntiques faciliten la comprensió del significat del text, tant, que fins i tot s'obre a la comprensió de la informació per les màquines.

Actualment existeixen llenguatges que són utilitzats per la creació de wikis semàntiques, com el RDF o el OWL. O llenguatges especialitzats ens cerques d'informació, sobretot en wikis semàntiques, com l'SPARQL.

Pel que fa al software especialitzat en el treball amb wikis semàntiques, tenim software específic com l'Ontowiki o el Wagn, o extensions semàntiques al software específic per wikis, com Semantic MediaWiki que és ofereix semàntica a la MediaWiki.

En la nostra aplicació pràctica utilitzarem el software Ontowiki que ens permet usar la Ontologia que hem creat com a base de la nostra wiki.

6.5 Ontowiki

Ontowiki és una aplicació desenvolupada per la universitat de Leipzig, basada en pOWL[13]. Aquesta aplicació desenvolupa les prestacions d'una ferramenta wiki per permetre la generació d'ontologies de forma col·laborativa. Facilita una presentació visual de una base de coneixement com a un mapa d'informació. Les bases de coneixement són considerades com mapes d'informació, a través dels quals, els usuaris accedeixen a la informació i els permeten elaborar els seus propis mapes de forma interactiva.

[13] http://powl.sourceforge.net

És una aplicació basada en web, escrita en PHP[14] i que utilitza una base de dades per guardar la informació. En la nostra aplicació pràctica, utilitzarem el paquet Virtuoso com a base de dades.

Per la instal·lació d'Ontowiki haurem de tenir en compte una sèrie de requisits (http://code.google.com/p/ontowiki/wiki/DeploymentRecommendations). Els més importants serà tenir un Sistema Operatiu Ubuntu, Apache i PHP configurats, i una base de dades relacionada, en el nostre cas el paquet Virtuoso.

A continuació detallarem el procés d'instal·lació del programari que s'ha seguit:

6.5.1 Instal·lació

Primer instal·lem Apache2

```
sudo apt-get install apache2
```

activem el mode-rewrite

```
sudo a2enmod rewrite
```

activem els permisos per poder escriure a la carpeta web

```
sudo chmod 777 /var/www
```

i activem el servei Apache

```
sudo service apache2 start
```

tot seguit ens descarreguem[15] el paquet Virtuoso i el descomprimim, a continuació executem el script de configuració

```
sudo ./configure --prefix=/opt/virtuoso-opensource-6.1.0
```

Cal tenir instal·lats tots els paquets que necessita Virtuoso : autoconf, automake, libtool, flex, bison, gperf, gawk, m4, make i OpenSSL, per instal·lar-ho farem:

```
sudo aptitude install gawk per ex.
```

Compilem i instal·lem:

```
sudo make && make install
```

el configurem:

[14] Llenguatge de programación interpretat, dissenyat originalmente pel disseny de pàgines web dinàmiques.
[15] http://sourceforge.net/projects/virtuoso/files/virtuoso/6.1.0/virtuoso-opensource-6.1.0.tar.gz/download

```
echo export PATH=\"\$PATH\":/opt/virtuoso-opensource-6.1.0/bin/
> /etc/profile.d/virtuoso.sh
```

I l'executem:

```
cd /opt/virtuoso-opensource-6.1.0/var/lib/virtuoso/db

sudo virtuoso-t -f &
```

Per provar el seu funcionament podrem entrar a http://localhost:8890 des del servidor. Un cop tenim el paquet Virtuoso configurat descarregarem i instal·lem l'aplicació Ontowiki des de http://code.google.com/p/ontowiki/downloads/list. I comencem la configuració d'Ontowiki.

Creen una carpeta anomenada Ontowiki a : <prefix>/var/lib/virtuoso. Copiem l'arxiu <prefix>/var/lib/virtuoso/db/virtuoso.ini a la carpeta que hem creat anteriorment. Obrim aquest fitxer amb un editor i busquem la opció "DirsAllowed", en aquesta opció hi afegim els directoris que utilitzarà ontowiki: /temp, /var/www, etc.

També caldrà configurar l'ODBC, ja que les connexions amb la Base de Dades l'utilitzen. Primer modifiquem els fitxer odbcinst.ini i odbc.ini que trobarem a la carpeta /etc.

Al fitxer odbcinst.ini hi afegirem la línia:

```
Driver = /lib/virtodbc.so
```

Aquest fitxer caldrà haver-lo copiat a la carpeta /lib si no li teníem. Al fitxer odbc.ini hi haurà de quedar:

```
[ODBC Data Sources] VOS = Virtuoso

[VOS] Driver = /lib/virtodbc.so

      Description = Virtuoso Open-Source Edition

      Address = localhost:1111
```

Finalment acabem de configurar l'Ontowiki. Copiem el fitxer config.ini-dist com a config.ini i l'obrim en un editor. Cal que hi tinguem:

```
store.backend = virtuoso
store.virtuoso.dsn = VOS
store.virtuoso.username = "dba"
store.virtuoso.password = "dba"
```

Habilitarem el mode Debug afegint:

```
debug = yes
```

i ens assegurem que AllOverwrite està en All al fitxer httpd.conf. Després ja hi podrem accedir en un navegador amb http://localhost/ontowiki.

6.5.2 Funcionalitats

Com ja s'ha especificat, Ontowiki és una aplicació que s'obre des d'un navegador. Un cop a dins, caldrà accedir a la nostra ontologia per obrir la base de coneixement amb la que voldrem treballar. Per accedir al nostre fitxer .owl que s'ha creat des de Protégé, cal accedir al Menú Edit de 'Knowledge Bases' i aqui seleccionar l'opció 'Create Knowledge Base'. Cal especificar l'URI on varem crear la ontologia i el fitxer .owl, tal com es mostra en la figura següent:

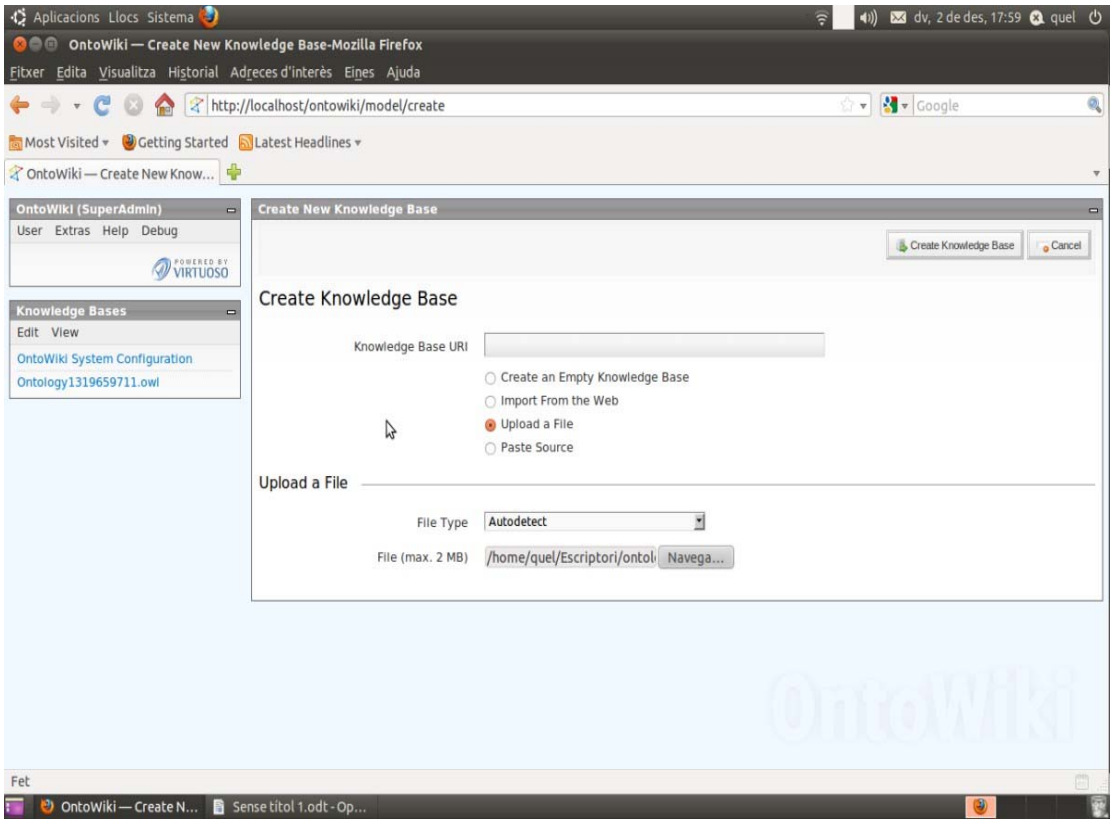

Fig. 32: Pantalla de creació de la Base de Coneixement.

Un cop accedim a la nostra base de coneixement observem que tenim una finestra de navegació, on podem seleccionar treballar amb classes o propietats. Dintre de cada opció hi trobarem les instancies corresponents. També trobem un quadre a la dreta de la pantalla on hi haurà els links per poder navegar dins la nostra wiki basada en la

ontologia que hem dissenyat. En aquest quadre, a partir d'una propietat, podrem accedir directament al seu domini, al rang o a les seves subclasses si en té.

En el quadre principal hi trobem les instancies, l'historia, una altra pestanya anomenada 'community' on hi trobarem totes les aportacions que podran fer els usuaris i el source o codi.

Totes aquests elements podran ser editats per ampliar la nostra base de coneixement, inclús es pot ampliar directament des de la mateixa wiki. Amb tot això ja tenim el prototip de la nostra wiki en funcionament.

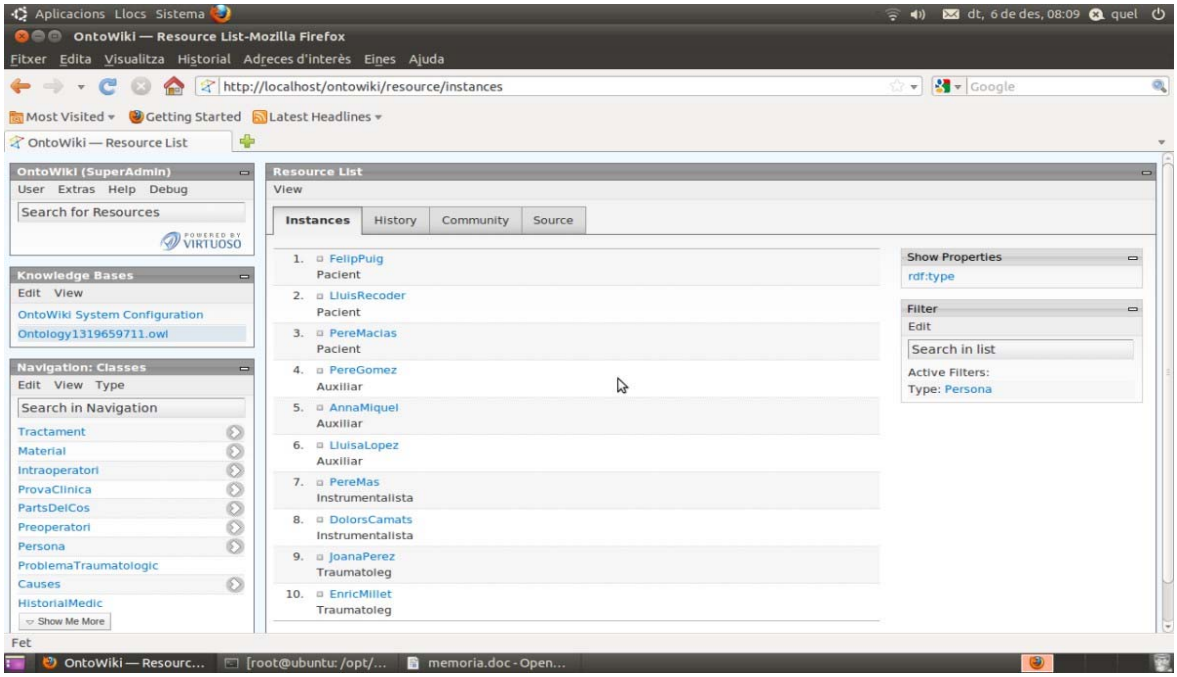

Fig. 33: Wiki originada a partir de la nostra ontologia.

Ontowiki ens aporta altres visions per navegar o visualitzar la informació de la nostra base de coneixement. Ontowiki usa el RDF per representar la informació. No obstant, per a facilitar mostrar la informació a les persones permet crear diferents vistes de la informació, com representacions tubulars o mapes. Si la informació va destinada a ser processada per màquines, pot usar RDFa, informació enllaçada o interfícies amb SPARQL.

Encara que ens hàgim decidit per Ontowiki per crear la nostra wiki a partir la ontologia desenvolupada, cal destacar que hi ha altre mètodes per crear una wiki a partir d'una ontologia, com usar la extensió Semantic MediaWiki de MediaWiki. A continuació explicarem unes quantes característiques d'aquesta extensió, encara que no s'hagi utilitzat pel desenvolupament de la nostra wiki.

6.6 Semantic MediaWiki

Semantic MediaWiki és una extensió de MediaWiki. La Semantic MediaWiki (SMW) ajuda en la cerca, organització i avaluació de la nostra wiki. A part afegeix els tags que permeten poder-la compartir més fàcilment. La SMW és una extensió de MediaWiki que possibilita que els textos de la wiki puguin ser compresos per màquines i no només per persones. Parlem d'extensió perquè necessita d'un altre programa per funcionar. En el cas de la SMW, dependrà de MediaWiki pel seu funcionament.

SMW afegeix marques addicionals al text de la wiki que ens permet afegir-hi notacions semàntiques. Això ajudarà a tenir estructurada molt millor la wiki, simplificar-la i sobretot, facilitar el tractament d'informació que altres usuaris puguin realitzar.

La SMW es caracteritza per:

- **Generar llistes automàticament**: normalment, les wikis tendeixen a tenir moltes llistes de diferents elements. Amb aquesta eina, i gràcies a la informació semàntica que contindrà, aquestes llistes es podran generar molt més fàcilment de forma automàtica.

- **Visualització de la informació:** els diferents formats de visualització faciliten et treball i la comprensió de la informació per part de l'usuari.

- **Millora de l'estructura de dades:** tendeix a fer més robustes les estructures d'emmagatzematge de la informació.

- **Cerca d'informació:** els usuaris poden millorar la cerca de informació específica amb diferents extensions.

- **Consistència per diferents llenguatges:** millora la consistència del llenguatge i evita redundàncies per wikis que poden suportar varis idiomes.

- **Integració de la informació:** la informació que tenim a la wiki pot ser ampliada fàcilment via importació de dades de múltiples tipus, aquesta és fàcilment integrable a la nostra wiki.

Cal tenir en compte que, tot que tan l'Ontowiki com la Semantic MediaWiki són dos eines que ens permeten afegir semàntica a la nostra wiki, tenen alguna diferencia entre elles. La Semantic MediaWiki ens ofereix moltes de les funcionalitats d'Ontowiki, inclús en millora molts aspectes, com pot ser la edició o navegació.

Un dels aspectes perquè s'usa més SMW és per la facilitat d'edició. Només cal destacar que la Wikipèdia es basa en la seva tecnologia.

No obstant, té alguns punts febles. Cal destacar que no disposa de editor d'ontologies, cosa de què si disposa Ontowiki, per exemple. Tant Ontowiki com la extensió SMW tenen la possibilitat d'importar ontologies, però només Ontowiki té la capacitat per editar-les.

Si volem fer una còpia de seguretat de la nostra wiki, podem utilitzar l'aplicació Wikiteam[16]. Wikiteam és un script GPV v3 escrit en phyton que descarrega tot el contingut d'una wiki que usi el sistema Mediawiki. A part d'obtenir tot l'estat actual de la wiki, pot recuperar l'historial de canvis, imatges i fitxers.

Un cop descarregat el cripta només cal executar la comanda:

```
Phyton dumpgenerator .py –api=http://wikidomain.com/path/to/api.php –
xml --images
```

Des del directori on volem guardar la nostra wiki. Cal anotar que:

- api: és la ruta a la API[17] de la wiki, normalment és la URL de la wiki, seguida de api.php.
- xml: serveix per recuperar les pàgines del historial.
- Images: recupera les imatges i els fitxers de la nostra wiki.

Aquesta aplicació ens permet guardar la nostra wiki en format XML.

7. CONCLUSIONS I LÍNIES DE FUTUR

El desenvolupament d'aquest treball ha permès a l'autor, l'estudi de varis temes relacionats amb la Web, concretament amb el tema del tractament de la informació que es fa a la Web.

La Web presenta un greu problema alhora del tractament de la informació. Durant anys, les dades de la Web han anat creixent exponencialment sense cap control, això ha provocat que cada cop sigui més difícil processar aquesta informació correctament. Les cerques d'informació són molt poc eficients, hi ha una gran quantitat de informació menyspreable, etc. El desenvolupament de la Web Semàntica treballa en aquest context, el problema però, és que encara falta conscienciar molts col·lectius perquè treballin en el mateix sentit.

[16] http://code.google.com/p/wikiteam/
[17] Del inglès Application Programming Interface,

En l'elaboració d'aquest treball m'he familiaritzat amb eines que faciliten tot el tractament de dades a la Web. S'ha descobert el món de les ontologies, un sistema molt vàlid pel processament de la informació i la seva estructuració. És un sistema ideal per crear una base de coneixement d'un domini específic. Tot i això, crec que encara és un sistema molt embrionari i falta que s'apliqui a molts més camps.

També hem treballat el tema dels wikis. Hem vist que és una eina ideal per compaginar-se amb les ontologies. Gràcies a les aplicacions que hem utilitzat, hem pogut importar la nostra ontologia a una wiki que hem creat. Amb aquesta importació, la nostra ontologia ha servit d'estructura bàsica de la wiki, amb tots els enllaços ja creats.

És important destacar la importància, cada vegada més gran, que estan agafant el món de les wikis en la Web actual. Estan proporcionant la interacció necessària perquè la Web deixi de ser una simple presentació de dades, i agafa el sentit d'una font de coneixement interactiva.

Amb tot això, crec que es disposa dels fonaments i les eines necessàries per millorar el tractament de la informació, però que aquest canvi serà un procés lent perquè encara falta molta implicació dels col·lectius relacionats amb el seu desenvolupament.

Com a possibles línies de futur per aquest treball, destacaria els següents punts:

- La creació de la wiki utilitzant Semantic Mediawiki, instal·lant les extensions necessàries per la importació d'ontologies. Podria ser una eina per comparar més profundament les diferents eines existents per la creació de wikis.

- Ampliar la ontologia amb el postoperatori. S'ha desenvolupat la base de coneixement fins el intraoperatori, però no cal oblidar que existeix un període de recuperació que també forma part del mateix domini.

- Un cop veiem que és un procés vàlid per una operació de traumatologia, es podria ampliar a altres camps de la medicina perquè pugui ser una eina de treball per professionals del sector.

8. GLOSSARI

- **Apache.** Servidor web, mantingut i desenvolupat per una comunitat de desenvolupadors sota la tutela de l'Apache Software Foundation.

- **API (*Application Programming Interface*).** Conjunt de funcions i procediments o mètodes, a la programació orientada a objectes que ofereix certa biblioteca per seu utilitzat per un altre Software com una capa d'abstracció.

- **DataProperty.** En el llenguatge OWL se les anomena a les relacions entre classes i valors determinats.

- **DAML+OIL**. Successor dels llenguatge DAML i OIL i combina el millor de tots dos. Va ser substituït pel llenguatge OWL.

- **DTD (*Document Type Definition*).** És una declaració en un document XML que especifica les restriccions en la seva estructura. Sol estar emmagatzemat en un fitxer independent, però podria estar inclòs en el arxiu del document.

- **Espai de noms (*NameSpace*).** És un estàndard del W3C que proporciona elements i atributs identificats de manera unívoca dins del d'un document XML. Si cada vocabulari té el seu espai de noms, tenim la possibilitat de compartir semàntiques utilitzant elements iguals evitant possibles ambigüitats del llenguatge.

- **HTML (HiperText Markup Language).** Llenguatge de marques dissenyat per estructurar textos i relacionar-los en forma d'hipertext.

- **Jambalaya.** Plug-in de Protégé per la visualització gràfica de les ontologies.

- **JDK (*Java Development Kit*).** Software que ofereix les eines per la creació de programes en Java.

- **ObjectProperties.** En el llenguatge OWL se les anomena a les relacions entre classes.

- **Ontologia.** És una estructura jerarquitzada d'elements que conté les entitats més importants d'un domini i les relacions entre elles.

- **Ontowiki.** Aplicació que desenvolupa les prestacions d'una ferramenta wiki per permetre la generació d'ontologies de forma col·laborativa. Facilita una presentació visual de una base de coneixement com a un mapa d'informació.

- **OWL (*en anglès Web Ontology Language*).** Llenguatge de marques impulsat pel W3C, dissenyat per la publicació i compartició de dades a Internet utilitzant Ontologies.

- **PHP.** Llenguatge de programació interpretat, dissenyat per la creació de pàgines web dinàmiques.

- **Protégé.** Entorn gràfic gratuït dissenyat per la creació i edició d'ontologies i bases de coneixement.

- **RDF (*Resource Description Framework*).** Desenvolupat pel W3C, especifica un model de metadades basat en el concepte de convertir les declaracions del recursos en el mes semblant a oracions típiques del llenguatge (subjecte-predicat-objecte).

- **RDF-Schema.** Extensió del RDF que descriu les jerarquies de classes i les seves propietats i relacions entre elles.

- **Semantic MediaWiki.** Aplicació que serveix per proporcionar semàntica a una wiki. Per instal·lar-se cal primer disposar una eina pel desenvolupament de wikis, com per exemple la Ontowiki.

- **SGML (*Standard Generalized Markup Language*).** Norma que estableix les regles d'etiquetatge del documents sense imposar-ne cap en particular.

- **SPARQL.** Llenguatge estàndard per la consulta de grafs RDF. Tecnologia clau pel desenvolupament de la Web Semàntica.

- **Tag.** Marca amb tipus que delimita una regió en els llenguatges basats en XML.

- **Ubuntu.** Sistema operatiu que utilitza com a nucli Linux i està basat en Debian.

- **URI (*Uniform Resource Identifier*).** Cadena curta de caràcters que identifica inequívocament un recurs.

- **URL (*Uniform Resource Locator*).** Cadena de caràcters que identifica inequívocament cadascun dels recursos d'informació disponibles a Internet.

- **Virtuoso.** Software que combina les funcionalitats d'una base de dades tradicional amb d'altres molt més específiques.

- **Wiki.** Lloc web col·laboratiu que pot ser editat des del navegador pels usuaris, això significa que els usuaris poden crear, modificar, afegir o esborrar la informació d'una web de forma molt interactiva.

- **W3C (*Word Wide Web Consortium*).** Consorci internacional que produeix recomanacions per la www. Creada el 1994 per Tim Berners-Lee.

- **XML(eXtensible Markup Lenguage).** Nascut com a successor del HTML, permet separar l'estructura del document del seu contingut. Es basa en etiquetes per separar els diferents elements del document.

- **XML-Schema.** Evolucionat de les DTDs, permet descriure els continguts permesos en un document XML.

9. BIBLIOGRAFIA

Llibres

- **Heather Williamson.** XML : manual de referència; traducción, Ángel Moreno Blázquez, Pilar Moreno Díaz ; revisión técnica Jesús Sánchez Allende
- **Dean Allemang, James Hendler** .Semantic Web for the working ontologist : effective modeling in RDFS and OWL
- **Antoniou G., van Harmelen F. (2004).** A semantic web primer. The MIT Press.
- **Arroyo S., et al. (2004):** Semantic Aspects of Web Services in Practical Handbook of Internet Computing. Munindar. Edited by P. Singh. Chapman Hall and CRC Press, Baton Rouge. 2004.
- **Domingue, John; Fensel, Dieter; Hendler, James A**. Handbook of Semantic Web Technologies. Ed Springer Reference. ISBN 978-3-540-92912-3. On-line: http://www.springer.com/computer/database+management+%26+information+retrieval/book/978-3-540-92912-3
- **Ismael Sanz, Ernesto Jiménez-Ruiz.** Maig del 2007. http://krono.act.uji.es/publications/techrep/Book-Chapter-Protege-report2007.pdf
- **Horridge Mathhew i altres**: A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools Edition 1.0. 2004.

Pàgines web

- W3 Consortium. Organisme encarregat de les bases de la Web semàntica i els diversos llenguatges implicats. http://www.w3.org, Octubre del 2011.
- Protege Ontology Library: http://protege.cim3.net/cgi-bin/wiki.pl?ProtegeOntologiesLibrary, Octubre del 2011.
- Manual Protégé: http://protege.stanford.edu/doc/users.html, Octubre del 2011.
- XML: http://en.wikipedia.org/wiki/XML, Octubre del 2011.
- XML_Schema: http://en.wikipedia.org/wiki/XML_Schema_%28W3C%29, Octubre del 2011.
- Web semàntica: http://infomesh.net/2001/swintro/, Octubre del 2011.
- RDF Schema: http://www.w3.org/TR/2000/CR-rdf-schema-20000327/
- Definicions ontologia: http://es.wikipedia.org/wiki/Ontolog%C3%ADa_%28inform%C3%A1tica%29, Octubre del 2011.

- OWL: http://www.w3.org/2004/OWL/, Octubre del 2011.
- Ontowiki: http://ontowiki.net/Projects/OntoWiki, https://github.com/AKSW/OntoWiki, http://code.google.com/p/ontowiki/wiki/Welcome, Novembre del 2011.
- Virtuoso: https://github.com/AKSW/OntoWiki/wiki/VirtuosoBackend, Novembre del 2011, http://wiki.toeska.cl/index.php/Instalaci%C3%B3n_Content_Compass_v1, Novembre del 2011.
- Wiki: http://es.wikipedia.org/wiki/Wiki, Novembre del 2011.7
- Semantic MediaWiki: http://www.semantic-mediawiki.org, http://infosofia.com.ar/2008/07/01/instalacin-y-configuracin-de-un-wiki-semntico/

Altres TFC

- **Natàlia Reichardt Berini**: Creació d'una Ontologia Universal. 2004-2005
- **Felix Abarca Corrales**: XML i Web Semàntica. 2004-2005
- **Judit Cabezas Cabiol**: Guia per visitar Barcelona, 2011
- **José V. Grimalt Aranda**: Creació d'una Ontologia universal. 2004-2005

ANNEX 1. Enunciat

Títol: "ONTO-TRAUMA"

Ontologia d'aprenentatge en l'àrea de cirurgia traumatològica

Descripció del treball
Disseny i avaluació d'una Ontologia d'aprenentatge en l'àrea de cirurgia traumatològica

Antecedents
En una operació quirúrgica hi ha diferents actors. Principalment, tenim el professional doctor que realitza la intervenció, el professional infermer que s'encarrega que tot el material quirúrgic que s'utilitzarà i el pacient. Existeixen altres professionals, com per exemple, el professional doctor especialista en anestesia i professionals infermers de suport.

Les diferents operacions que es realitzen, normalment, tenen un conjunt de passos diferenciats i en un ordre concret i cadascun dels passos necessita un material específic.

En tot moment, és clar, pot sortir qualsevol inconvenient que pot dificultar l'operació.

Objectius del treball

L'objectiu d'aquest TFC és dissenyar una ontologia que pugui recollir tota la informació per poder desenvolupar una operació quirúrgica. Concretament, ens centrarem en l'àrea de cirurgia traumatològica i únicament en els actors del professional doctor que realitza l'operació i el professional infermer encarregat del material quirúrgic.

Els requeriments concrets que ha de complir l'ontologia es proporcionaran al inici del projecte.

Un cop creada l'ontologia en llenguatge OWL, s'haurà de fer un petit prototip que permeti utilitzar l'ontologia per crear una wiki semàntica. Una wiki semàntica és una wiki que utilitza una ontologia (o esquema conceptual) per etiquetar els seus continguts. D'aquesta manera es pot representar informació de domini que sigui interpretable tant per humans com per programes informàtics.

ANNEX 2. Instal·lació Semantic MediaWiki

Semantic MediaWiki és una extensió de MediaWiki, per tant, per poder-la instal·lar caldrà tenir-la instal·lada abans. MediaWiki és un software lliure creat en php, per la creació o edició de wikis. Treballa sobre una base de dades, concretament sobre MySQL o PostgreSQL, que li serveix de suport per l'emmagatzematge i tractament de la informació.

Tenint en compte aquests punts caldrà tenir instal·lada una d'aquestes bases de dades, en aquest projecte ens hem decantat pel MySQL. Tot seguit instal·lem MediaWiki mitjançant l'aplicació de Apache Subversion. Ens situem al directori que prèviament haurem creat: /var/www/mediawiki i fem:

```
svn checkout
http://svn.wikimedia.org/svnroot/mediawiki/branches/REL1_17/phase3
```

tot seguit configurem seguint http://www.mediawiki.org/wiki/Manual:Configuration.

Per instal·lar la extensió Semantic MediaWiki, caldrà descarregar-nos la extensió Validator i la extensió de SMW pròpiament dita fent:

```
svn checkout
http://svn.wikimedia.org/svnroot/mediawiki/trunk/extensions/Validator/
```

i

```
svn checkout
http://svn.wikimedia.org/svnroot/mediawiki/trunk/extensions/SemanticMe
diaWiki/
```

des del directori extensions de mediawiki. Després, afegirem les comandes

```
require_once( "$IP/extensions/Validator/Validator.php" );
 include_once(
"$IP/extensions/SemanticMediaWiki/SemanticMediaWiki.php" );
 enableSemantics(www.owl-ontologies.com/Ontology1321475933.owl');
```

al final del fitxer LocalSettings.php que trobem al directori mediawiki. Després podrem accedir-hi des de /localhost/mediawiki.