PFC
Creació d'un plug-in de Protegé per a podar Ontologies

Berta Morera Lizandra

Consultor: Felipe Geva Urbano

Responsable de la assignatura: Jordi Casas Roma

Índex

1 Introducció

La primera definició d'ontologia la trobem al camp de la filosofia, quan al segle XVII Rodolfo Goclenio la definia per primer cop. Els estudis sobre aquest concepte van portar a ubicar les ontologies al camp de la metafísica, dins de la filosofia. Al camp de la informàtica, el primer cop en parlar d'ontologies va ser la segona meitat del segle XX, concretament al camp de la intel·ligència artificial, i des de llavors han estat moltes les investigacions i avenços assolits en aquest camp.

Una de les aplicacions principals de les ontologies és la Web semàntica. Actualment la Web és un conjunt caòtic d'informació: texts, llibres, propaganda comercial, àudio, vídeo, etc.. I la seva recuperació actualment es fa mitjançant paraules contingudes en els texts o mitjançant anotacions. La Web semàntica proposa un sistema per associar semànticament dades relacionats en un determinat context i per això utilitza les ontologies ja que permeten categoritzar les dades per aquest context.

La creació d'una ontologia des de zero és un procés en el qual s'ha d'invertir gran quantitat de temps i esforços. Tanmateix, en molts casos es podria reutilitzar una ontologia ja existent i eliminar les parts que no són d'interès per a usuaris i dissenyadors, deixant les parts que sí són d'interès. Aquest procés s'ha de realitzar mitjançant un algorisme de poda, el qual primer seleccionarà els conceptes d'interès directe i indirecte per a procedir després a l'eliminació de les parts que queden fora del domini de l'ontologia que es vol obtenir, tot mantenint la seva correctesa.

Aquest projecte de fi carrera mostra en primer lloc, els orígens de les ontologies, la situació actual i les línies de futur, i per l'altra costat descriu com aplicar un algorisme de poda a una ontologia ja existent mitjançant un plug-in desenvolupat en JAVA i que es crida des de l'entorn de treball facilitat per Protégé.

2 Objectius del projecte

Tal i com s'ha dit, la construcció d'ontologies des de zero és una feina a la que s'hi ha dedicar temps i esforç i que en molts casos es podria simplificar partint d'una altra ontologia ja creada i provada. Al mercat existeixen eines que permeten la reutilització d'ontologies existents però o bé no són públiques o bé no són de prou qualitat com per a ser utilitzats en tots els casos.

L'objectiu d'aquest treball de fi de carrera té una doble besant. D'una banda, una part més abocada a la investigació on es tracta de fer un estudi de l'estat de l'art de les ontologies, les seves aplicacions, i les orientacions que s'estan donant en el desenvolupament d'ontologies. I una besant més tècnica on, vistes les mancances pel que fa a eines d'optimització i reaprofitament d'ontologies, es detalla la creació d'una API en java per a l'editor Protegé, que permeti aplicar un algorisme de poda sobre una ontologia existent i que doni com a resultat una ontologia amb prou qualitat.

Per la consecució d'aquests objectius es disposa de material didàctic sobre OWL i Protégé disponible en la definició del pla docent de l'assignatura, enllaços per descarregar Protégé i enllaços amb exemples de creació d'APIs per a Protegé, i per últim l'algorisme de poda que es farà servir en el plug-in.

Els objectius es poden concretar doncs, en:

- Definició de l'estat de l'art
- Estudi de mercat de diferent productes de codi obert o propietari que permetin optimitzacions i reutilització d'ontologies
- Aprenentatge del llenguatge OWL utilitzat per la creació d'ontologies
- Aprenentatge del l'editor Protegé com a entorn de desenvolupament de les ontologies amb el llenguatge OWL.
- Aprenentatge de la creació d'APIs per a Protegé a través d'exemples existents
- Estudi de l'API de poda ja existent
- Ampliació de les funcionalitats de l'API de poda

3 Abast del projecte

L'abast del projecte va des de l'estudi de les ontologies i les eines que hi ha al mercat, i que permeten no només la seva creació sinó també les que permeten la seva optimització i reutilització, fins la creació d'un plug-in per a Protegé que permetrà la poda d'una ontologia.

4 Resultats esperats

De la realització del projecte s'espera obtenir d'una banda una visió de l'estat actual de les ontologies i les eines i recursos que existeixen per la seva gestió. I d'altra banda un plug-in per a Protegé que permeti la creació de noves ontologies a partir d'ontologies ja existents.

5 Estat de l'art de les ontologies

5.1 Definicions

De la filosofia a la computació

El terme ontologia té el seu origen en la filosofia grega. Les seves bases van ser establertes als textos escrits per Aristòtil, tot i que no va ser fins al segle XVII que Rodolfo Goclenio va fer la que es considera la primera definició formal del terme en l'obra *Lexicon philosophicum, quo tanquam clave philosophiae fores aperiuntur,* on exposà que l'ontologia és la filosofia de l'ens.

Més endavant, Leibniz, en la seva obra *Introductio ad Encyclopaediam arcanam* la definí com la "ciència del que és i del no res, de l'ens i del no ens, de les coses i dels seus modes, de la substància i de l'accident". L'any 1962 Jean Le Clerc a la seva obra *Ontologia sive de ente in genere* li donà un caire més tècnic i Christian Wolff la popularitzà definint-la com "ciència de l'ens en general, en quant que ens".

Aquestes definicions contribuïren a situar l'estudi de les ontologies en la branca de la filosofia anomenada metafísica, i que se centra en l'estudi de la naturalesa i organització de la realitat, és a dir el que existeix. Les ontologies en filosofia s'ocupen de l'establiment d'aquelles categories o modes generals de ser que tenen les coses partint de l'estudi profund de les seves propietats, estructures i sistemes. En com els ens poden ser classificats de diferents formes dins d'unes jerarquies, i subdividits segons les similituds i diferències que presenten.

En informàtica no es comença a parlar d'ontologies fins mitjans dels anys 80 del segle passat . Els primers estudis es realitzaren en el camp de intel·ligència artificial, i l'any 1993 Gruber donà la definició d'ontologia com "l'especificació explícita i formal d'una conceptualització compartida" [1]. A aquesta definició cal afegir a més que una ontologia defineix un vocabulari comú a l'àrea mitjançant un conjunt de termes, relacions entre termes i les regles per a combinar termes i relacions entre termes.

La definició donada en el camp de la informàtica és equivalent a la donada en el camp de la filosofia, tot i així la primera té com a finalitat la conceptualització en la comunicació i l'intercanvi d'informació entre diferents sistemes i entitats, i és precisament en aquest punt on difereixen ambdues definicions.

Una altra diferència és que, tot i que en filosofia una ontologia es definia com una explicació sistemàtica de l'Existència, en els sistemes basats en el coneixement el que existeix és el que pot ser representat i es representa mitjançant un formalisme declaratiu.

Història

Durant la segona meitat del segle XX, els filòsofs van debatre sobre els mètodes possibles per construir ontologies, sense arribar a aconseguir construir-ne cap. Paral·lelament, en el camp de la informàtica, es van construir algunes ontologies grans i robustes com WordNet[2] y Cyc[3], i amb un debat relativament petit sobre com havien de ser construïdes.

Des de la segona meitat dels anys 70 del segle XX, investigadors en el camp de la intel·ligència artificial reconeixen que la captura del coneixement és la clau per a construir sistemes d'intel·ligència artificial grans i sòlids. Aquests investigadors argumenten que es poden crear ontologies com a models de computació que permeten raonaments automatitzats reals.

Més endavant a la dècada dels 80, és quan des de la branca de intel·ligència artificial es comença a parlar d'ontologies. Aquest terme es referia tant a les teories per modelar el món, com a un component dels sistemes de coneixement.

A principi dels anys 90 Tom Gruber[4] publica *Toward Principles for the Design of Ontologies Used for Knowledge Sharing*[5] i dóna una definició d'ontologia que ha esdevingut un estàndard. Segons Gruber una ontologia és una "especificació formal i explícita d'una conceptualització compartida". En aquesta definició trobem els conceptes següents:

- Conceptualització, es refereix a un model abstracte d'algun fenomen del món del que s'identifiquen els conceptes que són rellevants
- Explícit, fa referència a la necessitat d'especificar els diferents conceptes que conformen l'ontologia.
- Formal, implica que l'especificació s'ha de presentar en un llenguatge de representació formalitzat
- Compartida, és a dir, que l'ontologia ha de reflectir el coneixement acceptat, com a mínim, pel grup de persones que l'han d'utilitzar.

Segons Gruber, les ontologies sovint són equiparables a les jerarquies taxonòmiques de les classes, definicions i relacions entre classes. La diferència està en que les ontologies no es poden limitar a aquestes formes, ni poden ser definides amb terminologies conservadores que no afegeixen cap coneixement del món. Per tant, per especificar una conceptualització, es necessita definir axiomes que forcin les possibles interpretacions dels termes definits.

A finals dels anys 90, el 1997, Weigand[6] dóna una definició més concreta d'ontologia afirmant que "una ontologia és una base de dades que descriu els conceptes del món o d'un domini, algunes de les seves propietats i com els conceptes es relacionen entre ells"

Tipus d'ontologies

Podem definir diferents tipus classificacions d'ontologies. Així, atenent a l'univers de domini sobre el que actuen distingim:

- Ontologies de domini: aquestes ontologies modelen un domini o subdomini específic de la realitat, com ara la medicina, les aplicacions militars, etc.
- Ontologies genèriques: amb aquestes ontologies es representen conceptes generals aplicables a un ampli rang d'ontologies de domini. Fan servir un glossari intern per definir els termes i descripcions d'objectes associats ja que són utilitzats per diversos conjunts de dominis.
- Ontologies representacionals: a la que s'especifiquen les conceptualitzacions que

existeixen en els formalismes de representació del coneixement.

A més d'aquests tres tipus, Guarino (1998)[7] hi afegeix les *task ontologies*, creades per a una activitat o tasca específica. Es diferencien de les ontologies de domini en que la seva aplicació és en un domini més concret com per exemple la venda de productes o la diagnosi de malalties.

Van Heist [8] proposa un classificació d'acord amb la quantitat i tipus de conceptualització, hi distingeix els següents tipus:

- Terminològiques: especifiquen els termes que són utilitzats per representar el coneixement. S'acostumen a utilitzar per unificar vocabulari a un univers de domini determinat.
- D'informació: especifiquen l'estructura d'emmagatzemament de bases de dades.
- De modelat del coneixement: especifiquen conceptualitzacions del coneixement.

Per últim es pot fer una classificació en funció de la usabilitat i la reusabilitat. Tenint en compte que la relació entre les dues característiques és inversament proporcional, de manera que com més reutilitzable és una ontologia menys usable és, i a l'invers.

Components d'una ontologia

Independentment de l'univers de discurs i del llenguatge en que estiguin expressades, les ontologies tenen components comuns. Aquests són:

- Individus: són instàncies o objectes
- Classes: conjunts, col·leccions, conceptes o tipus d'objectes
- Atributs: aspectes, propietats, característiques o paràmetres que els objectes poden tenir
- Relacions: són les maneres com les classes i individus es relacionen
- Restriccions: descripcions formals de què ha de ser vertader perquè una afirmació sigui acceptada com una entrada
- Regles: afirmacions amb la forma de les sentències *If – then* que descriuen les inferències lògiques que poden ser dibuixades des d'una asserció en un formulari particular
- Axiomes: afirmacions (incloent regles) en forma lògica que considerats junts conformen el tot que l'ontologia descriu en el seu domini d'aplicació.

Passes en la creació d'una ontologia [9]

Abans d'especificar les passes en la creació cal remarcar que no existeix una metodologia que ens doni una única ontologia resultant per un mateix domini, sinó que hi poden haver diversos resultats depenen de la persona que els hagi dissenyat i tots ells poden ser igualment vàlids

- El primer pas en la creació d'una ontologia és la determinació del domini de context i l'abast de l'ontologia
- En segon lloc s'ha de verificar si l'ontologia ja existeix i, en cas afirmatiu, veure la possibilitat de reutilitzar-la d'acord als propis requeriments.

- En tercer lloc cal identificar els conceptes rellevants del domini de context mitjançant un mètode d'abstracció d'informació del coneixement.
- D'acord amb els conceptes s'han de determinar les classes, la jerarquia de les classes i el modelat propi del paradigma orientat a objectes.
- Un cop establerta la jerarquia de classes, s'ha de donar la definició de les classes (atributs i característiques principals) i definir els aspectes d'interès relacionats amb els atributs (cardinalitat, etc.)
- Finalment, s'han de crear les instàncies.

RDF

El Resource Data Foundation (RDF) és un estàndard per l'intercanvi de dades al Web. RDF té característiques que faciliten la fusió de dades fins i tot entre esquemes diferents, tot mantenint un model de dades simple. RDF té una semàntica formal que ofereix una base de dependències pel raonament sobre el significat d'una expressió i a més té un vocabulari extensible basat en URIs.

Un dels seus principals avantatges és permetre especificar semàntica per les dades mitjançant una serialització d'XML, el RDF-SYNTAX[10]. RDF i XML són per tant complementaris, tot i que XML no és l'única possibilitat per representar la sintaxis de RDF.

A més d'oferir la possibilitat de definir metadades, RDF fa servir *esquemes* per definir col·leccions de classes, generalment d'un domini específic. Els esquemes organitzen les classes de forma jerarquitzada i ofereixen extensibilitat a través del refinament de les seves subclasses. A més és possible utilitzar esquemes ja existents per definir-ne de nous mitjançant modificacions incrementals sobre els primers.

L'estàndard RDF permet la representació del nom de les propietats i els seus valors. Les propietats es poden entendre com atributs de recursos i correspondre als parells atribut-valor clàssics, però també representen les relacions entre recursos. Per tant un model RDF es correspon a un diagrama entitat-relació i, utilitzant la terminologia del disseny orientat a objecte, podem dir que els recursos es corresponen a objectes i les propietats a variables d'instància.

Un model de dades està compost de tres tipus bàsics d'objectes:

- Recursos: és tot el que es pugui descriure mitjançant expressions RDF. Un recurs pot ser una plana web, una part, un element XML o fins i tot un conjunt de planes.
- Propietats: una propietat és un aspecte concret, característica, atribut o relació utilitzat per descriure un recurs. Cada propietat té un significat específic, defineix valors permesos, el tipus de recurs que pot descriure i la seva relació amb d'altres propietats.
- Estaments: un estament és la unió entre un recurs, una propietat i el valor d'aquesta propietats. Aquestes tres parts s'anomenen respectivament: subjecte, predicat i objecte. L'objecte d'un estament pot ser un altre recurs o un literal.

OWL

El llenguatge d'Ontologies Web (OWL) està dissenyat per ser utilitzat per a processar el contingut de la informació al Web enlloc de representar únicament aquesta informació. Igual que RDF, OWL

permet representar explícitament termes i relacions entre termes, la diferència està en que OWL ofereix millors mecanismes d'interoperabilitat.

OWL estén RDF per permetre relacions complexes entre diferents classes RDF, i donar millor precisió en les restriccions de classes i propietats.

Abans d'OWL s'havien utilitzat altres llenguatges per desenvolupar eines i ontologies, però presentaven la limitació de que eren només aplicables a una comunitat específica. OWL, recolzat en la utilització de RDF, dota a les ontologies de les següents capacitats:

- Capacitat de ser distribuïdes a través de diversos sistemes
- Escalabilitat a les necessitats del Web
- Compatibilitat amb els estàndards Web d'accessibilitat i internacionalització
- És obert i extensible

OWL presenta tres subllenguatges, cadascun d'ells amb un grau d'expressivitat més elevat que l'anterior:OWL Lite, OWL DL, y OWL Full.

- OWL Lite: està dissenyat per a usuaris que necessiten una classificació jeràrquica i restriccions simples.
- OWL DL: està dissenyat per a usuaris que esperen una màxima expressivitat tot garantint que totes les conclusions siguin computables, i que tots els càlculs es resolguin en un temps finit. OWL DL inclou totes les construccions del llenguatge d'OWL, però només poden ser usats sota certes restriccions (per exemple, una classe pot ser subclasse d'altres classes, però no pot ser mai instància d'una altra classe).
- OWL Full: està adreçat a usuaris que esperen la màxima expressivitat i llibertat sintàctica que ofereix RDF sense garanties computacionals. Per exemple, OWL Full permet que una classe sigui considerada simultàniament com una col·lecció de classes individuals i com una classe individual. OWL Full permet una ontologia per augmentar el significat del vocabulari preestablert.

5.2 Camps d'aplicació de les ontologies

Web semàntica

Sens dubte, aquest és el camp on té més rellevància l'aplicació de les ontologies i és aquí des d'on s'ha donat més impuls el seu estudi.

L'estudi de les possibilitats de classificació de la informació havia estat tradicionalment en mans de documentalistes i arxivers, és a partir dels anys 90 quan comencen a prendre pes les investigacions en el desenvolupament de tecnologies i mètodes que permetin l'organització i la gestió de la informació documental.

Principalment hi ha tres problemes en processar i accedir a la informació a Internet:

- En primer lloc el fet de que el Web sigui un sistema descentralitzat, això canvia l'escenari tradicional per al que estaven preparades les disciplines clàssiques

vinculades a la documentació i la recuperació d'informació.
- En segon lloc, quan l'usuari extreu informació del Web es troba amb l'anomenat *adversarial information retrieval*, i que consisteix en la recuperació d'informació que no té res a veure amb el que es cerca per culpa de generadors de continguts que preparen o versionen els seus continguts per enganyar als motors de cerca d'informació i aconseguir aparèixer a un lloc elevat dins la llista de consultes més populars. La motivació és rebre visites per anunciar-se o vendre algun producte. Aquest també és un aspecte mai contemplat en la cerca clàssica d'informació.
- Per últim, la dificultat que pot tenir per l'usuari el tractament de la informació amb marcatges html. Per exemple una persona és capaç de trobar i interpretar un títol encara que el tipus de lletra sigui diferent del que està acostumada a veure, en canvi si a ordinador no se li indica el nom de la etiqueta com a títol serà incapaç d'identificar-lo com a tal.

Als darrers anys el W3C ha elaborat més de 80 especificacions per la implantació d'una infraestructura que permeti la gestió documental de recursos nous i heterogenis. I s'ha proposat el disseny d'eines que permetin conèixer, comparar i combinar recursos web amb diferent estructura: les ontologies. Els principals mitjans amb els que es volen assolir els objectius de la web semàntica són:

- Incorporació de l'estàndard XML al web per a la codificació de pàgines amb la finalitat de que aquestes transportin una càrrega semàntica.
- Incorporació de metadades a les pàgines i llocs Web en un format que sigui compatible amb l'estructura general www, amb diverses categories de pàgines i que sigui interoperable entre diferents sistemes informàtics. Per això s'aplica la norma RDF
- Especificació de conceptes de diversos dominis del coneixement mitjançant un sistema d'ontologies, el qual fa servir un llenguatge basat en lògica simbòlica que permet ser eventualment interpretat per un ordinador, l'OWL.

Gestió del coneixement

Les ontologies permeten evolucionar els mètodes tradicionals de classificació basats en classificacions i tesaures. En els darrers anys s'han aplicat eines de mineria de dades, o datamining, i han estat de gran utilitat ja que permeten l'anàlisi de grans volums d'informació amb finalitats estadístiques, pressa de decisions i definició de models de comportament. Tot i així la mineria de dades no permet obtenir un coneixement semànticament estructurat, com el que ofereixen les ontologies que permeten representar el coneixement sobre un determinat domini.

Sistemes de recomanació de consultes

També anomenats sistemes de filtrat d'informació podrien assimilar-se als sistemes de difusió d'informació mitjançant els quals, i d'acord al perfil dels usuaris subscrits al servei, es generen periòdicament unes alertes que notifiquen a aquests usuaris de l'existència de recursos que s'adeqüen als seus interessos.

En el Web aquests sistemes apliquen tècniques de filtrat d'informació que faciliten l'accés dels usuaris a la informació que necessiten. L'origen d'aquests sistemes es remunta a principis de la

dècada dels noranta i utilitzen tècniques de filtrat d'informació a partir de les facilitats que ofereix l'HTML I XML. En l'actualitat s'estan dedicant molts esforços als sistemes de recomanació semàntic que basen el seu funcionament en l'aplicació d'ontologies per utilitzar la informació semàntica de les característiques categòriques d'un element.

Hipertext

Les ontologies poden arribar a oferir la possibilitat de superar les limitacions pròpies de l'hipertext, com ara texts amb autonomia funcional [11]

Teleeducació (e-learning)

La teleeducació, entesa com la formació a distància utilitzant el Web, requereix de la creació de continguts que sovint comença des de zero i que representa per tant un cost molt alt, ja que la formació via Internet no deixa lloc a la improvisació per part del professor i això obliga a tenir uns continguts ben definits i estructurats.

Una manera d'amortitzar els continguts és la seva reutilització. El procés de reutilització passa per estructurar un curs en una o més peces d'altres cursos que ja s'hagin dut a terme. És en aquest punt on són aplicables les ontologies. El desenvolupament pot potenciar :

- La independència, ja que estructura els cursos i permet una fàcil separació lògica.
- La interoperabilitat, ja que l'ontologia és independent de la plataforma i pot servir de llenguatge comú que permeti conversions entre unes plataformes i altres.
- Ofereix característiques de durabilitat i qualitat a través de les dates de creació i caducitat.
- Permet reconèixer als autors dels cursos.

Comerç electrònic

En els darrers anys s'han introduït les ontologies també en l'àmbit del comerç electrònic. Autors com Farsani y Nematbakhsh (2006) [12] suggereixen una metodologia per recomanacions personalitzades en el context del comerç electrònic. Es tracta d'un procediment per recomanar productes a clients potencials. L'algoritme proposat es basa en el modelat d'informació sobre productes i usuaris amb OWL (Ontology Web Language).

El procés s'inicia amb la classificació de productes i consumidors mitjançant OWL, això facilita l'anàlisi de les similituds producte-client. En una segona fase, es seleccionen consumidors actius tenint en comte recomanacions anteriors (el sistema no recomana a un client si el número de les seves recomanacions anteriors no sobrepassa un llindar determinat). La classificació de productes i clients s'utilitza per a crear una matriu d'avaluacions productes-clients. L'algoritme recomana algun dels productes d'entre les classes definides basant-se en el número d'avaluacions en la matriu.

Aplicacions a CMS

Un sistema de gestió de continguts és una aplicació informàtica que proporciona una estructura de suport per la creació i administració de continguts digitals. Per definició els gestors de continguts han de manegar molta informació i en formats diversos, en aquest context les ontologies proporcionen interoperabilitat entre bases de dades i a més permeten descriure aspectes i relacions lògiques per definir taxonomies, sistemes de navegació i adaptació de continguts.

5.3 Estudi de mercat de les eines relacionades

En el conjunt d'eines relacionades en la creació de les ontologies descriurem en primer lloc els llenguatges. La característica comuna, és que tots ells estan basats en llenguatges de marques.

D'entre els llenguatges que existeixen actualment es destaquen els llenguatges ontològics tradicionals com Ontolingua, KIF, OCML, Flogic, LOOM. I també llenguatges per la Web estandaritzats per el consorci de la W3C com XML, XOL, SHOE, DAML+OIL, OWL i SWRL.

A continuació es dóna una breu descripció d'alguns dels llenguatges utilitzats en la creació d'ontologies:

- OWL (Ontology Web Language), ha estat reconegut per W3C com a estàndard per la creació d'ontologies. Va ser desenvolupat com a continuació de RDF i RDFS i és el primer projecte de llenguatge ontològic que inclou OIL, DAML i DAML+OIL. El que es pretén és que OWL sigui utilitzat en tot el World Wide Web. Tots els seus elements (classes, propietats i individus) estan definits com a recursos RDF i identificats per URIs.
- SWRL (Semantic Web Rule Language), estén OWL per tal d'incloure Horn-like rules i inclús permet combinar-les amb una base de coneixement OWL. A més proporciona un nivell més alt d'abstracció que amplia la d'OWL. Les regles que es proposen fan referència a la implicació entre un antecedent (body) i un conseqüent (head). El significat s'ha de llegir com: “sempre que les condicions especificades a l'antecedent es compleixin, les condicions del conseqüent s'han de donar també”. Un antecedent buit implica la veritat absoluta (true), i un conseqüent buit implica la falsedat de l'antecedent (false). Existeix una sintaxi XML per representar aquestes regles basada en RuleML i la sintaxi de representació XML d'OWL
- KIF, és una sintaxis per la lògica de primer ordre i es basa en S-expresions. La lògica de primer ordre (o lògica de predicats, o càlcul de predicats) consisteix a separar cada sentència en subjecte i predicat. El predicat modifica o defineix les propietats del subjecte. En la lògica first-order un predicat només es pot referir a un únic subjecte.
- Rule Interchange Format (RIF) i F-Logic, combina ontologies i regles.
- Common Algebraic Specification Language, és un llenguatge d'especificacions basat en lògica. S'aplica a les especificacions d'ontologies per donar modularitat i mecanismes per estructurar les dades.
- Common logic, correspon a l'estàndard ISO 24707, i és una especificació per a la

família de llenguatges d'ontologies que poden ser traduïts d'uns als altres amb precisió.
- CycL, és el llenguatge dissenyat pel projecte Cyc. Aquest és un projecte d'intel·ligència artificial que té l'objectiu d'unir una ontologia comprensiva i una base de dades de coneixement general per tal de permetre a les aplicacions d'intel·ligència artificial realitzar raonaments del tipus humà.
- DOGMA (Developing Ontology-Grounded Methods and Applications) adopta l'aproximació d'orientació a la realitat per donar un alt nivell d'estabilitat semàntica.
- Gellish, és un llenguatge que inclou regles per la seva pròpia extensió i d'aquesta manera integrar una ontologia amb un llenguatge d'ontologia.
- IDEF5 és un mètode d'enginyeria per desenvolupar i mantenir un domini d'ontologies de forma usable i acurada.
- SADL té un subconjunt de l'expressivitat d'OWL i utilitza el llenguatge anglès introduït via un Plug-in d'Eclipse.
- OBO, és el llenguatge utilitzat per les ontologies relacionades amb la biologia i la biomèdica.

A més dels llenguatges, en el procés de creació i gestió d'ontologies són necessàries altres eines que permetin estructurar la informació en conceptes semàntics i afegir informació addicional. Aquestes són les anomenades eines d'anotació. Segons els resultats publicats pel projecte Roda[13], el qual té com a objectiu l'avaluació d'eines d'anotació, podem distingir tres tipus d'eines:

1. **Eines de creació d'ontologies**, són els editors d'ontologies i permeten la codificació d'una ontologia en un determinat llenguatge. Aquestes eines permeten definir l'estructura a través de la qual es classificarà la informació. Algunes eines d'aquest tipus són:

   - Apollo: permet modelar el coneixement a través d'uns principis bàsics com ara classes, instàncies, funcions, relacions, etc. La interfície d'usuari té una arquitectura oberta i està escrit en llenguatge de programació JAVA.
   - Link Factory: permet construir sistemes de terminologia corporativa capaços d'extreure valor significatiu de gran quantitat de dades no estructurades emmagatzemades a bases de dades de contingut corporatiu.
   - OILED: és un editor d'ontologies utilitzat per la construcció d'ontologies que fa servir DAML+OIL
   - OntoEdit Free and Professional versions: Permet crear, navegar i modificar ontologies. Acompleix els estàndards del W3C i ofereix moltes interfícies exportables a la majoria de llenguatges de representació d'ontologies.
   - Ontolingua Server: Proporciona un entorn de col·laboració distribuït per a navegar, crear, editar, modificar i utilitzar ontologies.
   - OpenKnoME: és la base dels motors de coneixement topThing. És un sistema de gestió del coneixement i un motor d'ontologies.
   - Protégé-2000: permet el desenvolupament d'ontologies i sistemes basats en el coneixement. És una eina de codi obert que va ser creada per la Universistat de Stanford en Java i utilitzant una arquitectura extensible. Les aplicacions creades amb Protégé són emprades en la resolució de problemes i presa de decisions a dominis particulars. Protégé suporta dues formes de modelar ontologies: amb frames i amb OWL. A més les ontologies poden ser exportades a diversos formats com RDF Schema, OWL i XML Schema.
   - SymOntoX: software que emmagatzema i gestiona un domini d'ontologia.
   - WebODE: eina per modelar el coneixement utilitzant ontologies. Proporciona la màxima flexibilitat i interoperabilitat amb altres aplicacions de negocis.
   - WebOnto: està compost per un applet de JAVA i un servidor web customitzat que permet als usuaris navegar i editar models de coneixement sobre el web.

2. **Eines d'anotació externa,** que permeten associar meta informació a pàgines que estan presents al Web. La meta informació s'emmagatzema en un repositori extern a la pàgina que està destinat específicament a mantenir les anotacions. Aquests repositoris acostumen a ser bases de dades RDF. La metainformació pot estar basada o no en una ontologia. En aquesta categoria trobem editors de text i servidors d'anotacions, per exemple:

   - COHSE[14]: és un projecte d'investigació sobre mètodes que milloren la qualitat, consistència i amplitud de documents web enllaçats mentre es recuperen i mentre es creen. Utilitza tres tecnologies: un servei de raonament d'ontologies que s'utilitza per representar un sofisticat model conceptual de termes documentals i les seves relacions; un servidor obert d'enllaços hipermèdia basat en Web; i la integració d'aquests dos per poder enllaçar documents via metadata descrivint els seus continguts.
   - Annotea[15]: projecte que pretén millorar l'entorn col·laboratiu de la W3C a través de l'us d'anotacions compartides. Una anotació pot ser un comentari, una nota, una explicació o qualsevol text que es vulgui adjuntar a un document Web externament.
   - Annozilla: projecte dissenyat per veure i crear anotacions associades a una pàgina web, com defineix el projecte Annotea del W3C.
   - Yawas (Yet Another Web Annotation System): programa que permet destacar parts d'un document web.
   - Annotation System: sistema basat en tres arquitectures vinculades: un client que pot realitzar i cercar anotacions; un servidor d'aplicacions que gestiona les peticions del client; i un servidor de base de dades per emmagatzemar les anotacions realitzades pels usuaris.
   - Trellis Web: entorn interactiu que permet als usuaris afegir observacions, punts de vista i conclusions.

3. **Eines d'autor**, permeten informació estructurada, meta informació, a la mateixa pàgina Web partint d'una ontologia ja definida prèviament. Normalment s'utilitzen els llenguatges XML o RDF, i per la definició de les ontologies s'acostuma a utilitzar DAML o OWL. Alguns exemples d'aquestes eines són:

   - MnM: eina d'anotacions basada en ontologies que permet anotar pàgines Web amb continguts semàntics de forma automàtica o semiautomàtica. Integra un navegador amb un editor d'ontologies i proporciona unes APIs obertes per enllaçar MnM amb servidors d'ontologies i per integrar MnM amb eines d'extracció d'informació. Treballa amb diferents llenguatges d'ontologies com RDF, DAML+OIL i OCML
   - OntoAnnotate: permet ampliar la base del coneixement de documents mitjançant la navegació Web. L'usuari pot aprendre fets individuals que corren directament a la base de coneixement, i amb ells, les seves associacions i relacions. OntoAnnotate pot ser utilitzat juntament amb les anotacions dels documents per proveir els documents de contingut semàntic, fent-los directament accessibles i interpretables com continguts en la Web Semàntica. En resum, permet a cada usuari crear documents amb continguts semàntics capaços de ser entesos per les màquines.
   - OntoMat-Annotizer: ajuda a l'usuari a crear i mantenir ontologies basades en DAML+OIL, per exemple per crear instàncies, atributs i relacions.
   - SHOE Knowledge Annotator: és un llenguatge de representació del coneixement basat en HTML. Afegeix informació semàntica a les pàgines Web utilitzant etiquetes. Distingeix dues categories d'etiquetes: les que s'utilitzen per la construcció d'ontologies; i les que s'utilitzen per anotar documents Web per subscriure'ls a una o més ontologies, declarar entitats de dades i realitzar

afirmacions sobre les entitats segons les normes escrites per les ontologies
- SMORE: permet etiquetar els documents en RDF utilitzant ontologies Web associades als elements i termes específics.
- Melita: eina d'anotació de text semi-automàtica basada en ontologies que té integrada una màquina d'extracció d'informació. Implementa una metodologia amb el propòsit de gestionar per als usuaris tot el procés d'anotació.
- GATE (General Architecture for Text Engineering): es tracta d'una infraestructura de components de software reutilitzable que s'utilitza per donar suport al processament del llenguatge humà.

A banda de les eines descrites al projecte RODA també es poden trobar al mercat altres plataformes per el desenvolupament d'ontologies, com per exemple:

- Gecosoft: aquesta plataforma esta composta per dues eines de software principals
  - Un editor del coneixement que permet automatitzar el procés de construcció de coneixement en forma de Mapes Conceptuals i que obté la formalització en el llenguatge OWL dels mapes construïts
  - Un servidor de Coneixement que automatitza els processos de gestió de persistència, col·laboració, accés i integració del coneixement que és generat i que ha estat compartit en diferents moments per diferents usuaris.

  Aquestes eines es troben integrades a partir d'un conjunt de mòduls relacionats amb el treball col·laboratiu que faciliten la comunicació entre ambdues.

- Sesame és una eina de codi obert desenvolupada com a API de Java i que proporciona als desenvolupadors d'aplicacions un conjunt d'eines molt útil per fer qualsevol cosa amb RDF. Proporciona un entorn de treball per l'emmagatzemament, consulta i raonament amb RDF i RDF Schema i pot ser usat com a base de dades RDF i RDF Schema o com a llibreria de Java per aplicacions que necessiten treballar internament amb RDF.

- Jena és un framework de codi obert per desenvolupar aplicacions en Java amb tecnologies de la Web Semàntica. Té dos components principals:
  - un motor d'inferència basat en regles d'entorn per generar models RDF, RDF Schema i OWL
  - un intèrpret i servidor SPARQL

  i a més diversos serveis d'emmagatzemament com adaptadors a bases de dades relacionals.

  En un principi Jena va ser creat per manipular metadades des d'aplicacions Java, però en la actualitat existeixen dues versions, una que principalment proporciona suport a RDF i capacitat de raonament limitat; i l'altra que inclou una API per la manipulació d'ontologies i que a més suporta el llenguatge OWL.

  En resum Jena permet gestionar tot tipus d'ontologies, emmagatzemar-les i realitzar consultes sobre elles.

5.4 Casos d'us

En els darrers anys, fruit de la necessitat d'organitzar la informació al Web, han sorgit nombrosos estudis per establir estàndards en les comunicacions dins d'un domini.

El grup de treball OWL del consorci W3C, ha identificat els principals casos d'us d'ontologies al Web i els ha descrit al document *Casos d'Us i Requisits* [15]. L'estudi ha estat realitzat sobre unes

25 aplicacions implementades, i en sistemes en us (utilitzant llenguatges d'ontologies Web prematurs).

Aquest grup de Treball els ha classificat en sis àrees principals:

- Portals Web

  Són llocs Web que proporcionen informació sobre un tema concret proporcionant als usuaris la possibilitat de consultar i rebre noticies, parlar, crear una comunitat o trobar enllaços a d'altres llocs que proporcionin informació relacionada. Les ontologies permeten fixar regles de categorització amb la finalitat de millorar les cerques.

  Un dels secrets de l'èxit d'un portal és la informació que conté, la qual és introduïda pels propis membres de la comunitat que han d'indexar els seus documents a algun subtema i han d'establir etiquetes amb informació per tal de que els continguts puguin ser utilitzats en sindicacions. Tanmateix, un simple índex no proporciona suficient agilitat en les cerques. Per permetre una sindicació de continguts més intel·ligent, els llocs web poden definir ontologies i d'aquesta manera aconseguir que les cerques donin uns resultats que serien impossibles utilitzant els sistemes de recuperació convencional.

  Exemples de portals basats en ontologies són: Ontoweb[17] o The Open Directory Project[18].

- Col·leccions multimèdia

  En les col·leccions multimèdia les ontologies permeten proveir d'anotacions semàntiques a imatges, àudio, i objectes no textuals. Idealment, les ontologies haurien de capturar coneixement addicional sobre el domini, que pot ser utilitzat per millorar la recuperació d'imatges.

  Les ontologies multimèdia poden ser de dos tipus: media-especific o content-especific. Les primeres poden tenir taxonomies de diferents tipus media i descriure les seves propietats. Les segones poden descriure el subjecte del recurs. Aquestes ontologies no són específiques pels continguts multimèdia per tant poden ser reutilitzades per d'altres documents dins el mateix domini.

- Administració de Llocs Web Corporatius

  Habitualment els llocs corporatius tenen multitud de pàgines per mostrar informació dels seus productes, casos d'estudi, procediments corporatius, etc. Les ontologies poden ser utilitzades per a indexar aquests documents i proporcionar millors mitjans de recuperació que les taxonomies clàssiques que utilitzen cerques a través de paraules clau.

- Documentació de Disseny

  És utilitzat sobre grans plecs de documentació utilitzats en enginyeria. Aquesta documentació pot ser de diferents tipus i està organitzada seguint una estructura jeràrquica tot i que les estructures difereixen entre elles.

  Les ontologies poden ser utilitzades per a construir un model d'informació que permeti l'exploració de tota la informació (elements, relacions entre elements, propietats dels elements i enllaços a documentació que descriu i defineix aquests elements)

- Agents intel·ligents i serveis

  Es refereix a agents que tenen la capacitat d'entendre i integrar diversa informació provinent de diferents recursos.

Un exemple seria un agent per planificar activitats socials, aquest agent recolliria els gustos i preferències de l'usuari i proposaria una activitat per cada vespre. Aquest tipus d'agents requereixen d'ontologies de domini que representin els restaurants, hotels, etc., així com ontologies de servei per representar els termes utilitzats als serveis reals.

- Computació Ubiqüista

  El paradigma de la computació ubiqüista es caracteritza pel canvi de la maquinària de computació dedicada per les capacitats embegudes al nostre entorn diari. Les seves característiques són: mides reduïdes, manejables i aparells wireless. El fet d'estar tant estesos, i la seva naturalesa basada en wireless fa que requereixin d'arquitectures de xarxa per un suport automàtic ad hoc a la configuració. Una altra raó per desenvolupar una configuració automàtica és que aquesta tecnologia esta orientada a donar suport a consumidors ordinaris.

  Els serveis que es poden oferir i els mecanismes de capacitat de descripció necessiten d'una representació d'esquemes ad hoc que depèn fortament d'estandarditzacions.

  Els llenguatges ontològics són utilitzats per descriure les característiques dels aparells, el mode d'accedir-hi, la política establerta pel propietari pel seu us, i d'altres restriccions i requeriments.

Finalment cal destacar que tant en el desenvolupament dels llenguatges ontològics com en les eines de creació i administració d'ontologies, tots els esforços van cap a la utilització d'OWL i de fet la majoria dels sistemes que actualment utilitzen DAML, OIL, I CAML+OIL (que són els llenguatges predecessors i en els que es basa OWL) s'estan migrant a OWL. A més eines de creació d'ontologies, com Protégé que és àmpliament utilitzada, disposa de suport per OWL.

6 Algorisme de poda

6.1 Descripció

La creació d'una ontologia és una feina feixuga que implica la inversió de molt de temps i esforços si es comença des de zero. És per això que en el procés de creació sempre es té en consideració la reutilització d'una ontologia existent.

Com més general sigui l'ontologia de la qual es parteix, més parts quedaran fora del domini de l'ontologia que es vol aconseguir. Per tant és necessària la utilització d'algun mecanisme que elimini les parts de l'ontologia inicial que són irrellevants al domini de l'ontologia que es vol crear. Aquest mecanisme és l'algorisme de poda[18].

L'algorisme de poda és utilitzat per esborrar tant automàticament com sigui possible les parts que són irrellevants pel propòsit de l'ontologia que es vol crear. L'algorisme està compost de dues fases:

- Fase de selecció
- Fase de poda

6.1.1 Fase de selecció

Es tracta de cercar els conceptes d'interès directe (CoI) en el sistema d'informació donat, entenent que un concepte és d'interès directe si els usuaris i dissenyadors estan interessats en la seva representació a la base d'informació del sistema, o en inferir la seva informació.

Quan els requeriments funcionals del sistema d'informació estan especificats formalment, els conceptes d'interès directe es poden extreure automàticament. Els detalls del procés d'extracció depenen del mètode i el llenguatge utilitzat en l'especificació, però en general l'especificació formal d'un sistema d'informació consisteix en:

- Una signatura (nom, paràmetres i resultats). Els tipus de paràmetres i resultats seran els tipus de les entitats definides a l'ontologia resultant.
- Un conjunt de precondicions. Són expressions booleanes que impliquen conceptes definits a l'ontologia resultant, i que defineixen les condicions que s'han de satisfer si l'operació s'executa correctament.
- Un conjunt de postcondicions. Igual que les precondicions, són condicions booleanes que defineixen les condicions que s'han de satisfer, en aquest cas, un cop s'ha dut a terme l'execució.

Partint d'aquestes definicions, els conceptes d'interès directe quedarien definits com:

- Els tipus de paràmetres i els resultats de les operacions de sistema.
- Els conceptes que apareixen a les pre o postcondicions

Tot i així, no sempre és possible obtenir l'especificació formal dels requeriment, o si s'obté, aquesta pot ser incompleta. En aquests casos els dissenyadors poden definir explícitament el conjunt CoI o afegir conceptes nous a aquells determinats per les especificacions funcionals.

A més dels conceptes d'interès, a l'ontologia original hi poden haver conceptes que generalitzen a aquests però que no són d'interès directe. Aquests conceptes s'han de tenir en compte ja que poden participar en restriccions que afecten instàncies del grup de CoI tot i que, com ja es veurà, més endavant alguns poden ser eliminats un cop es verifica que no són necessaris per mantenir la correctesa de la nova ontologia.

Al conjunt de conceptes CoI i les seves generalitzacions, l'anomenem conjunt de conceptes d'interès generalitzat, G(CoI). Formalment

$$G(CoI) = \{c \mid c \in CoI \vee \exists sub\ (\text{IsA+}(sub,c) \wedge sub \in CoI)\}$$

6.1.2 Fase de poda

Un cop trobats els conceptes d'interès directe del sistema d'informació donat, cal identificar i eliminar de l'ontologia original (O_0) els conceptes que no pertanyen a aquest grup, per això l'algorisme de poda utilitzat en aquesta memòria proposa les següents fases:

- Poda dels conceptes i restriccions irrellevants
- Poda de supertipus innecessaris
- Poda de camins de generalització innecessaris
- Eliminació dels elements orfes

Poda dels conceptes i restriccions irrellevants

Consisteix a determinar el conjunt de conceptes d'interès directe generalitzat, *G(CoI).* Aquest conjunt és el format pels conceptes d'interès directe i les seves generalitzacions. Un cop determinat *G(CoI)*, s'eliminaran de l'ontologia inicial O_0 els elements que no en formin part i que són els anomenats conceptes irrellevants. Formalment s'han d'eliminar els conceptes

$$\text{ConceptesIrrellevants} = \{c \mid c \text{ és un concepte} \wedge c \in O_0 \wedge c \notin G(CoI)\}$$

Podar cada un d'aquests conceptes implica trobar les relacions de generalització en les quals participa i eliminar-les. També poden ser eliminats els supertipus però no els

subtipus que seran analitzats en la fase següent.

De forma similar s'han d'eliminar les restriccions que no són rellevants perquè són aplicades a conceptes que ja no formen part del conjunt de conceptes d'interès directe o indirecte. Formalment ho definim de la següent manera:

RestriccionsIrrellevants = {*ic* | *ic* is a constraint ∧ *ic* ∈ O_0 ∧ ∃c (c ∈ CC(ic) ∧ c ∉ G($\mathcal{CoI}$)}

En aquest pas cal tenir en compte que alguns llenguatges permeten representar diverses restriccions unides en una sola. En aquests casos s'ha de separar la restricció de la resta per no eliminar totes les restriccions d'integritat.

Poda de pares innecesaris

Un cop realitzada la fase anterior ja es disposa del conjunt de conceptes d'interès directe, les seves generalitzacions i les seves restriccions d'integritat més rellevants per l'ontologia final. Tot i així encara hi poden haver conceptes que, tot i no ser rellevants per l'ontologia final, no s'han eliminat per ser supertipus de conceptes d'interès directe.

El pas següent serà determinar quins són els conceptes potencialment innecessaris i, en cas de confirmar-se que ho són, eliminar-los. Per això cal en primer lloc identificar els elements que són estrictament necessaris, aquests són els que han estat definits com a elements d'interès directe i els que són utilitzats per alguna de les restriccions d'integritat:

ConceptesNecessaris = $\mathcal{CoI}$ ∪ *CC(O1)*

La resta d'elements són potencialment innecessaris:

ConceptesPotencialmentInnecessaris = *G*($\mathcal{CoI}$) – *ConceptesNecessaris*

Un cop determinats aquests grups de conceptes, es podran esborrar pares del grup de *ConceptesNecessaris* si aquests pertanyen al grup de *ConceptesPotencialment-Innecessaris* i a més no són fills d'algun dels conceptes que pertanyen al grup de *ConceptesNecessaris:*

ParesInnecessaris = {c | c ∈ ConceptesPotencialmentInnecessaris ∧
¬∃c' (c' ∈ ConceptesNecessaris ∧ IsA+(c,c'))}

Altrament, si esborréssim un concepte potencialment innecessari però que tingués un pare necessari, provocaríem la pèrdua de la relació de generalització entre el seu pare necessari i els seus fills, també necessaris. El fill no heretaria els tipus de relacions definits pel pare i es generaria una ontologia incorrecte.

En conclusió, si un concepte potencialment innecessari no té un pare que pertany al grup de conceptes necessaris, significa que no participa d'un camí de generalització rellevant i pot ser eliminat.

Poda de camins de generalització innecessaris

El següent pas en el procés de poda és la comprovació de que per cada un dels conceptes hi ha només els camins de generalització necessaris, ja que en ocasions hi pot haver més d'una generalització entre conceptes però no totes ser necessàries. Es tractarà doncs, d'eliminar els camins de generalització que, en cas d'eliminar-se, no produeixen cap pèrdua a l'ontologia final.

Un camí de generalització entre dos conceptes és potencialment redundant si cap dels conceptes intermedis és:

- Estrictament necessari, és a dir que no forma part del conjunt de conceptes d'interès directe, ni és utilitzat per alguna de les restriccions d'integritat rellevants
- Super o subtipus d'una altra relació de generalització

En resum, una generalització serà potencialment redundant si existeixen altres generalitzacions entre el mateix parell de conceptes, i cap dels conceptes intermedis és estrictament necessari ni participa en una altra generalització. En aquest cas es poden eliminar els conceptes que intervenen al camí de generalització i totes les relacions de generalització a les quals intervenen.

Eliminació dels elements orfes

El darrer pas és l'eliminació de les instàncies de l'ontologia, els classificadors de les quals han estat eliminats a les passes anteriors del procés de poda. Quan una instància d'una classe és eliminada, tots els valors de les seves propietats així com les relacions *sameAs* són també eliminades. Formalment el conjunt d'instàncies a eliminar és

$$\text{InstànciesOrfes} = \{i \mid i \text{ is an individual} \wedge i \in O3 \wedge \neg \exists c \, (\, c \in O3 \wedge \mathit{InstanceOf}(i,c) \,)\}$$

6.2 Consideracions per l'aplicació de l'algorisme al plug-in de Protégé

6.2.1 Selecció dels conceptes d'interès

Tal i com s'ha dit, el primer pas de l'algorisme de poda consisteix a la selecció dels paràmetres d'interès, els quals es poden obtenir a partir dels requeriments funcionals o bé especificar-los de forma explícita. Així el dissenyador de l'ontologia pot seguir un procediment totalment manual (guiat o no) o automàtic. En un procediment automàtic és el sistema qui determina a partir de la informació de que disposa, quins són els conceptes d'interès. Aquesta informació la pot determinar a partir de conceptes d'interès introduïts prèviament; a partir de dades que no són considerades conceptes (p.e. Tipus d'entitats, tipus de relacions...); o bé recursos externs.

El desenvolupament del Plug-in no contempla la possibilitat de definir conceptes de forma automàtica ja que Protégé no ofereix la possibilitat d'especificar requeriments funcionals, per tant no hi ha possibilitat de fer una selecció automàtica dels conceptes. En canvi s'ha

donat la flexibilitat de poder triar entre un procediment d'introducció totalment manual, o bé guiat pel sistema:

- Selecció desassistida: el dissenyador tria els conceptes necessaris sense cap assistència per part del sistema.

- Selecció assistida: el sistema dóna suport al dissenyador proposant-li els conceptes per què trii els que són d'interès per el seu domini.

6.2.2 Eliminació dels elements orfes

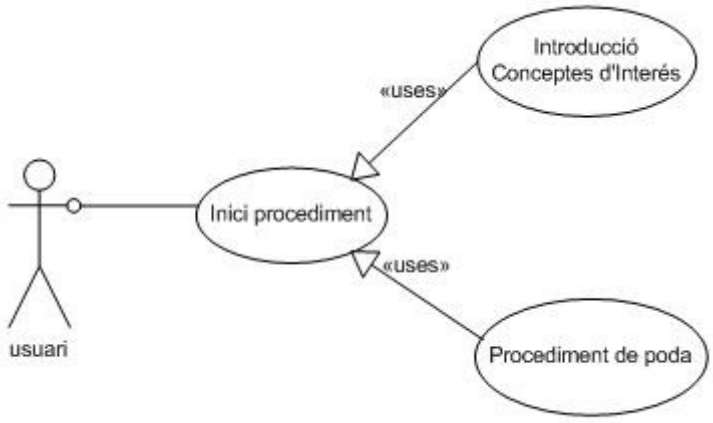
Tot i que l'algorisme defineix aquesta fase com una fase independent, el desenvolupament per a l'API de Protégé és innecessari ja que la pròpia aplicació elimina les instàncies de les classes o restriccions eliminades.

7 Anàlisi de l'API de java

En tractar-se d'un plug-in, els models descrits en aquest document defineixen les funcionalitats que ofereix la pestanya concreta que s'ha desenvolupat i que representa una petita part de les utilitats que ofereix Protégé. Cal remarcar que el plug-in només pot funcionar com a part de l'eina Protégé i no de forma independent o en un altre eina de desenvolupament d'ontologies.

7.1 Model de casos d'us

El model de casos d'us consta d'un únic actor que és l'usuari final que utilitza l'API i tres cassos d'us. Existeix una relació d'inclusió (*Include*) entre els casos d'us *Inici procediment* i *Introducció conceptes d'interès* i *Procediment de poda* ja que els dos últims no tenen sentit per si sols, i només poden donar-se si són des del primer.



A continuació es detallen els casos d'us definits pel procés de poda:

Cas d'us "Inici procediment"

1. Resum de la funcionalitat: permet a l'usuari iniciar el procediment de poda.
2. Flux d'esdeveniments principals:
    i. L'usuari tria la pestanya *Prunning*
    ii. L'usuari indica els conceptes que són rellevants en el domini de la seva ontologia (cas d'us *Introducció de conceptes d'interès*)
    iii. L'usuari inicia el procés de poda (cas d'us *Procediment de poda)*
3. Flux d'esdeveniments alternatius:

i. L'usuari no introdueix cap concepte d'interès directe

1) El sistema mostra les característiques de l'ontologia carregada

2) Torna al punt *ii* del flux d'esdeveniments principals sense iniciar el procés de poda.

ii. Els paràmetres introduïts no es troben al sistema

1) El sistema mostra un missatge d'error

2) Torna al punt *ii* del flux d'esdeveniments principals sense iniciar el procés de poda per tal de que l'usuari modifiqui els conceptes

iii. L'usuari anul·la el procés de poda

1) Torna al punt *i* del flux d'esdeveniments principals

4. Pantalla:

- Pestanya Prunnig

Dades mostrades:

– Desplegable amb els conceptes de l'ontologia carregada (classes i propietats definides)

– Informació del sistema. Informació d'ajuda en el cas de que l'usuari l'hagi sol·licitat, informació sobre els errors si hi ha hagut, o bé informació sobre les característiques de l'ontologia original i la final.

Dades sol·licitades:

– Introducció de conceptes d'interès.

Cas d'us "Introducció de conceptes d'interès "

1. Resum de la funcionalitat: aquest cas d'us està inclòs al cas d'us *Inici procediment* i permet a l'usuari introduir els conceptes que són d'interès directe en el domini de la seva ontologia. L'usuari podrà introduir cada un dels conceptes de forma assistida o no assistida. Per la introducció assistida disposarà d'un desplegable amb el conjunt classes i propietats de l'ontologia original i un botó *Afegir* per incorporar-les a la taula de conceptes. Si l'usuari no vol introduir els conceptes de forma assistida pot escriure'ls directament a la taula de conceptes. En qualsevol cas els conceptes introduïts a la taula poden ser modificats o esborrats independentment de com s'hagin introduït.
2. Flux d'esdeveniments principals:
   i. L'usuari introdueix els conceptes d'interès directe directament a la taula (introducció no assistida) o a través del desplegable (introducció assistida)
   ii. El sistema valida l'existència dels conceptes a l'ontologia original
3. Flux d'esdeveniments alternatius:
   i. Els conceptes que s'han introduït no corresponen a cap classe ni propietat de l'ontologia original
      1) S'atura el procés
4. Pantalla:

Aquest cas d'us està inclòs al cas d'us *Inici procediment* i no té pantalla pròpia

Cas d'us "Procediment de poda"

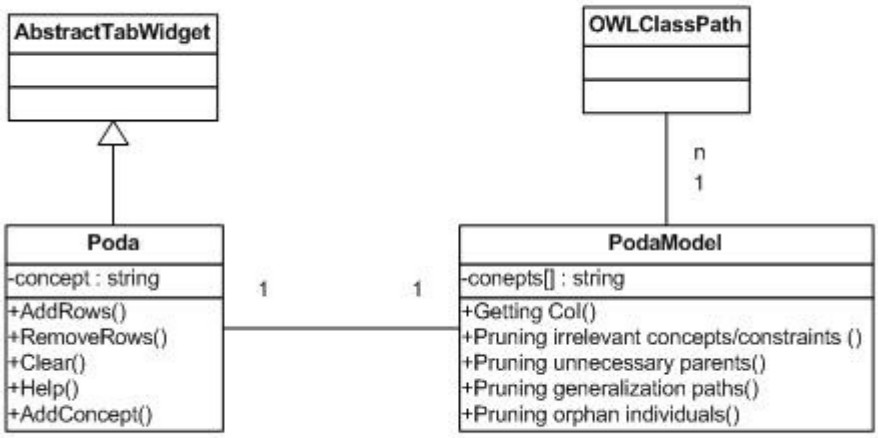
1. Resum de la funcionalitat: aquest cas d'us està inclòs al cas d'us *Inici procediment* i executa les funcions que conformen el procediment de poda.
2. Flux d'esdeveniments principals:
    i. Serialització dels conceptes d'interès directe
    ii. Eliminar conceptes irrellevants del model
    iii. Eliminar pares innecessaris
    iv. Eliminar camins de generalització innecessaris
    v. Eliminar elements orfes
    vi. Mostrar resultats: elements inicials, elements finals i elements eliminats
3. Flux d'esdeveniments alternatius:
    i. Error en l'execució d'alguna de les passes
        1) S'atura el procediment i es desfan els canvis realitzats si hi ha hagut
        2) Es mostra la fase que ha donat error a l'àrea de notificació d'informació de la pestanya
4. Pantalla:

    Aquest cas d'us esta inclòs al cas d'us *Inici procediment* i no té pantalla pròpia

7.2 Diagrama estàtic d'anàlisi (model conceptual de dades)

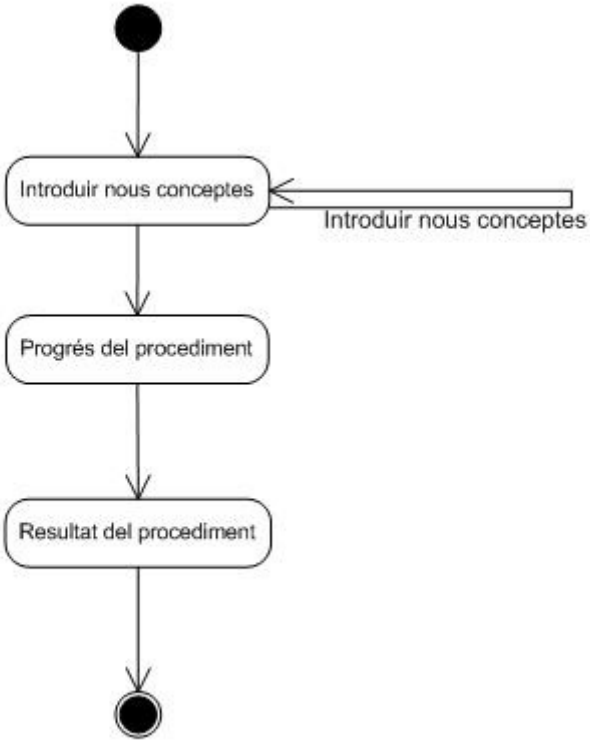
El model conceptual definit a partir del model de casos d'us conté els atributs i operacions que l'API utilitzarà per podar una ontologia donada.

El resultat és un model senzill ja que l'API representa una part d'un model més ampli que és el Protégé. Està format per tres classes: la classe Poda que conté les operacions de la capa d'usuari i és una generalització de la classe AbstractTabWidget; la classe PodaModel que conté els mètodes per executar l'algorisme de poda; i la classe OWLClassPath que conté mètodes relacionats amb els camins d'una classe.



7.3 Interfície gràfica d'usuari

Atesa la simplicitat gràfica de l'api, s'utilitza la mateixa pestanya tant per a introduir dades com per a mostrar el resultat de les operacions realitzades, s'ha definit cada canvi durant el procés com un estat en el diagrama.



L'esbós de la pestanya queda tal i com es mostra a la imatge següent.

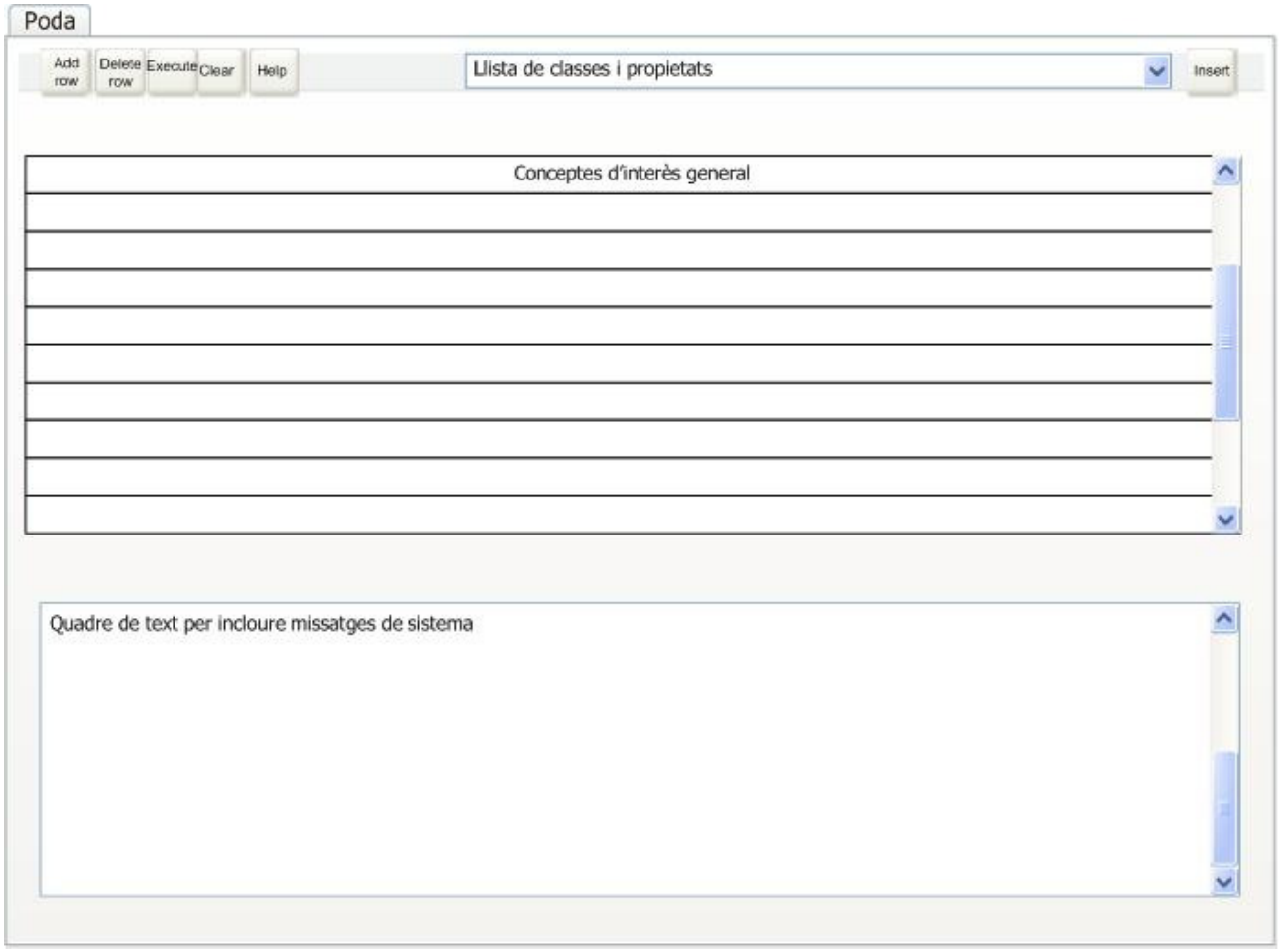
La pestanya disposarà d'un menú, seguit d'una taula per permetre la introducció dels conceptes d'interès i un espai per la sortida d'informació del sistema.

El menú contindrà els controls següents:

- Botó *Add row*, afegeix una fila a la taula de conceptes d'interès
- Botó *Remove row*, elimina una fila a la taula de conceptes d'interès
- Botó *Execute*, inicia el procediment de poda
- Botó *Clear*, neteja les files que hi pugui haver a la taula i la informació que hi hagi a l'àrea de text deixant la pestanya com quan s'accedeix per primer cop.
- Desplegable de conceptes, control a on es mostren totes les classes i les propietats de l'ontologia carregada i permet la selecció d'una d'elles.
- Botó *Insert*, copia la el concepte seleccionat al desplegable de conceptes a la taula de conceptes d'interès.

La taula amb els conceptes d'interès permetrà la modificació de les seves cel·les independentment de si el valor s'ha introduït directament o si s'ha copiat des del desplegable.

Per últim, l'àrea de text mostrarà l'ajuda del sistema, informació sobre les característiques de l'ontologia original i la resultant en cas de que l'execució sigui satisfactòria i els missatges d'error.


Poda
Add row
Delete row
Execute
Clear
Help
Llista de classes i propietats
Insert
Conceptes d'interès general
Quadre de text per incloure missatges de sistema

8 Disseny de l'API

8.1 Requeriments tècnics

Requeriments tècnics:

- JDK 1.4 o superior
- Protégé 3.4.4 o superior
- Afegir al classpath: protege.jar, looks.jar, unicode_panel.jar i protege-owl.jar

8.2 Disseny de la pestanya

Tal i com s'ha dit, la nova API és una extensió de la classe AbstractTabWidget. La nova pestanya ha d'incloure una zona per la introducció dels conceptes d'interès i una àrea per mostrar els missatges del sistemes. A més hi haurà d'haver un menú amb els botons: Add row, delete row, Execute, Clear i Help, un desplegable que mostri les classes de l'ontologia i un botó per copiar el concepte seleccionat al desplegable a la llista. Quan un concepte és a la taula, tant si ha estat escrit directament a la taula, com si prové del desplegable, l'usuari pot modificar-lo.

Els controls definits anteriorment requereixen de la importació del paquet javax.swing. Per definir la taula s'utilitzarà una *JTable* i per les operacions sobre aquesta es definirà una taula temporal *DefaultTableModel*. Per l'àrea de text que mostrarà els missatge del sistema s'utilitzarà control *JTextArea* amb scroll. Pel que fa al menú, s'utilitzarà el control *JToolBar* i sobre aquest s'afegiran els botons, definits com *JButtons*. Per últim per mostrar les classes i propietats s'utilitzarà un control *JComboBox*.

El funcionament serà el següent: el primer cop que entri l'usuari, el *JComboBox* mostrarà els conceptes obtinguts del model inicial amb els mètodes `getUserDefinedOWLNamedClasses` i `getUserDefinedOWLProperties` que pertanyen a `OWLModel`. Mitjançant un botó que estarà a continuació del combo podrà afegir el concepte que ha seleccionat a la taula. Una altra possibilitat per introduir conceptes a la taula és escriure'ls directament sobre la taula.

Amb els botons del menú podrà fer la resta d'accions. Els botons *Add row* i *Delete row* seran per operar únicament sobre la *JTable* i li permetran afegir files o esborrar-les. El botó *Clear* actuarà sobre la *JTable* esborrant totes les files i sobre *JText* deixant l'àrea en blanc. El botó *Help* mostrarà informació d'ajuda per l'execució a l'àrea definida pel control *JText*. El botó *Execució* iniciarà el procediment de poda.

El primer pas en l'execució de l'algorisme consistirà a recollir els conceptes que l'usuari ha introduït a la taula i guardar-los a una llista. Per això s'utilitzarà una variable de tipus `ArrayList`. Amb la llista de conceptes d'interès, es validarà que aquests estiguin dins l'ontologia carregada a Protégé, en cas de no existir, es mostrarà un missatge d'error per l'àrea de sortida de dades i s'aturarà l'execució.

La interfície que ens permetrà manipular el model inicial, és `OWLModel`. Així utilitzant el mètode `GetOWLModel` s'obtindrà l'ontologia carregada i es validarà que els conceptes de la llista introduïda per l'usuari existeixen al model.

Amb els conceptes d'interès directe validats i guardats a la variable `ArrayList`, iniciarem el procediment de poda. Pel primer pas *Esborrar conceptes irrellevants del sistema*, utilitzarem els mètodes `getUserDefinedOWLNamedClasses` i `getUserDefinedOWLProperties` de la interfície `OWLModel` que retornen les classes i propietats del model carregat. D'aquesta manera anirem revisant l'ontologia inicial i guardant els conceptes d'interès directe i els conceptes que són necessaris per tractar-se de generalitzacions, subtipus o supertipus i sense els quals el model final seria incorrecte.

Amb la llista de conceptes d'interès directe i conceptes necessaris, passarem al segon pas del procediment, *Esborrar parents innecessaris.* Per això s'obtindran les classes i propietats que, tot i no ser d'interès, poden ser necessàries per tractar-se de pares de conceptes d'interès. Per cada una d'aquestes classes i propietats es comprovarà que si té una superclasse aquesta no forma part dels conceptes d'interès directe ni és subclasse d'un concepte d'interès directe. Sí es compleixen aquestes condicions o bé si no té una superclasse, la classe es pot eliminar.

Pel tercer pas *Esborrar camins de generalització innecessaris*, s'haurà de comprovar que existeix més d'un camí de generalització i a més que les classes intermèdies no són estrictament necessàries. Per comprovar que la classe disposa de més d'un camí de generalització s'utilitzarà l'operació `getSuperclassCount` i per obtenir el nom s'utilitzarà `getNamedSuperclasses` de la classe `OWLNamedClass`. Si s'acompleixen aquestes condicions s'esborrarà el camí de generalització.

El darrer pas del procediment, *Esborrar elements orfes*, no cal implementar-lo perquè el propi Protégé elimina les instàncies de classes o restriccions eliminades.

9 Descripció de la instal·lació i utilització de l'API

Per la correcte execució de l'API és necessari incloure al classpath les classes que es llisten a continuació o iniciar el Protégé des del fitxer RunProtege.bat:

- protege-owl.jar
- looks.jar
- protege.jar
- xml-apis.jar
- antlr-2.7.5.jar
- arq.jar
- arq-extra.jar
- axis.jar
- commons-discovery-0.4.jar
- commons-lang-2.0.jar
- commons-logging-1.1.1.jar
- concurrent.jar
- edtftpj-1.5.2.jar
- ekitspell.jar
- icu4j_3_4.jar
- iri.jar
- jcalendar.jar
- jdom.jar
- jena.jar
- jep-2.4.0.jar
- json.jar
- junit.jar
- kazuki.jar
- log4j-1.2.12.jar
- lucene-core-2.3.1.jar
- orphanNodesAlg.jar
- owlsyntax.jar
- stax-api-1.0.jar
- swingx-1.0.jar
- swrl-jess-bridge.jar
- wstx-asl-3.0.0.jar

- xercesImpl.jar

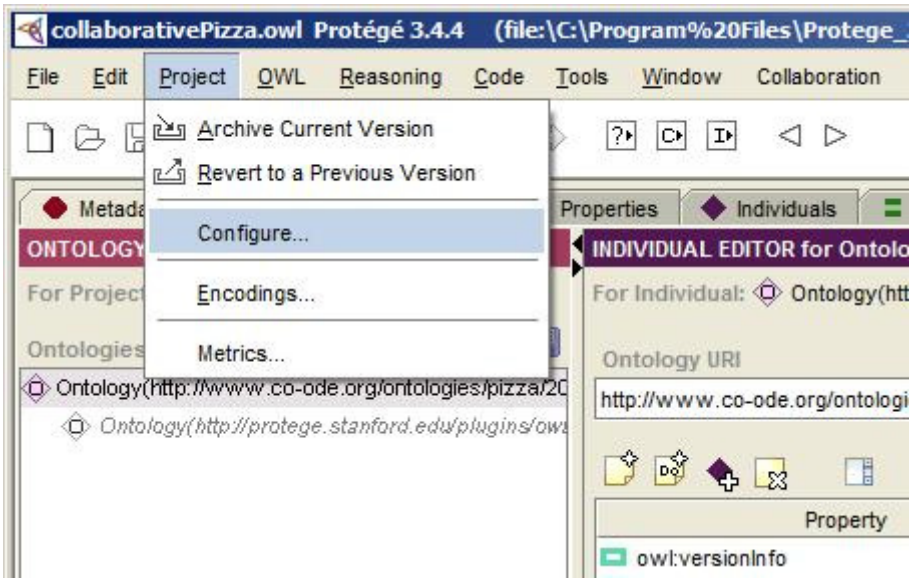
i els arxius propis de Protégé:

- lax.jar
- looks-2.1.3.jar
- protege.jar

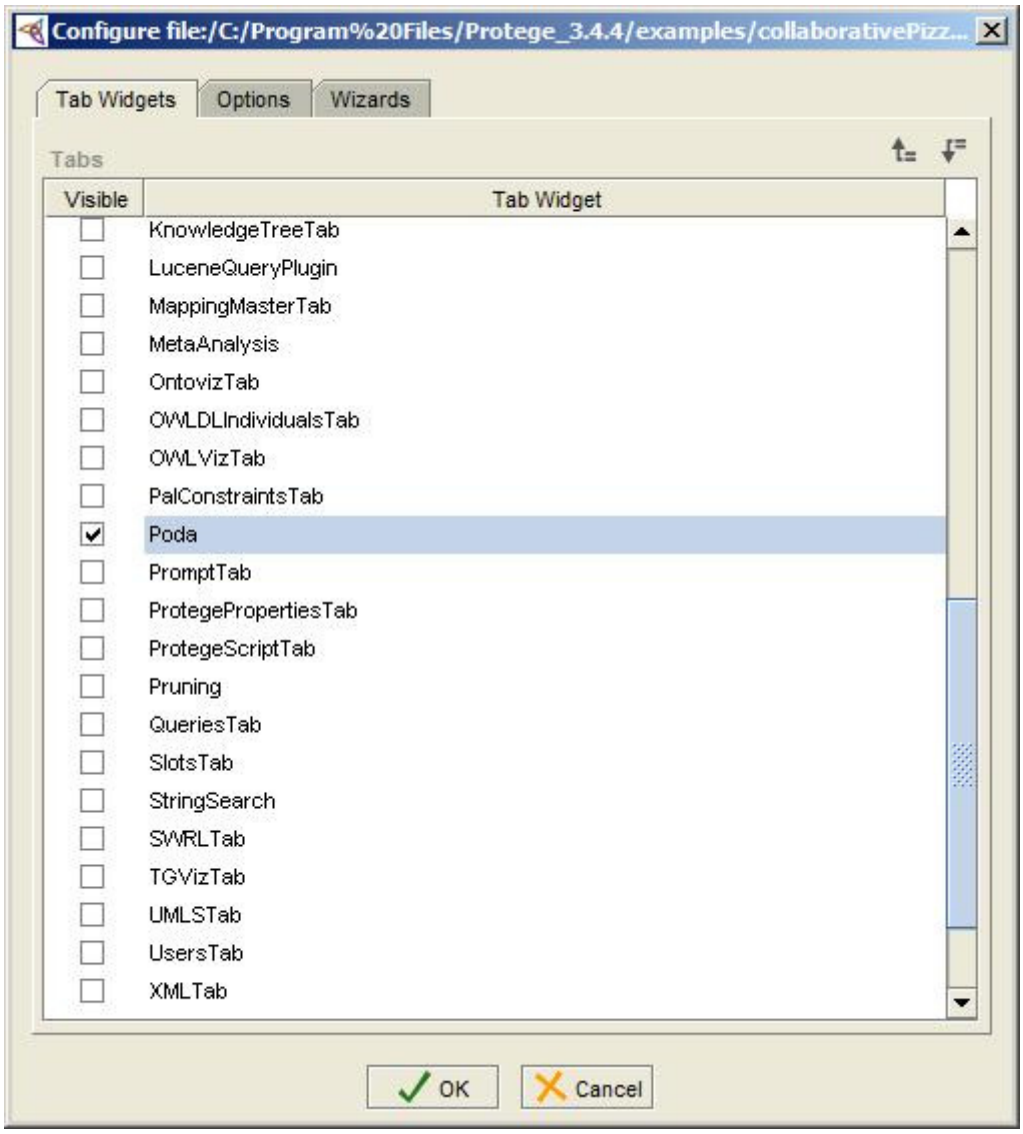
El plugin esta inclòs en un fitxer comprimit que s'ha de desempaquetar al directori Plugins de la instal·lació de Protégé. Un cop descomprimit cal comprovar que s'ha guardat a la ruta:

\\Protege_3.4.4\plugins\prune

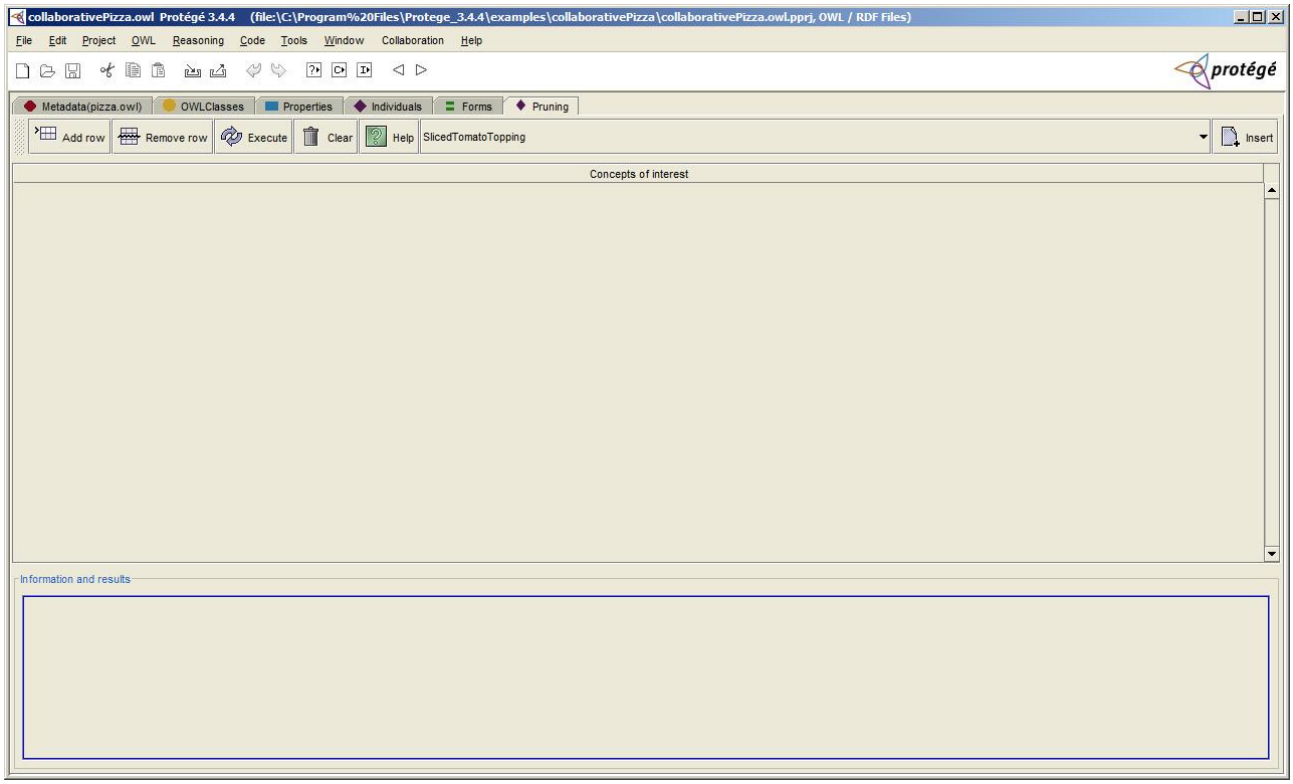
Per disposar de la nova pestanya, caldrà obrir el Protégé des del fitxer executable, .bat i carregar la ontologia que es vol podar. Un cop carregada s'ha d'accedir al menú Project, i després a l'opció *Configure*, tal i com es veu a la imatge següent:



La opció de Configure mostrarà la llista de les API's carregades i les disponibles. Entre elles ha d'aparèixer la nova API, Pruning. S'ha de seleccionar i prémer el botó OK



Després de prémer el botó OK, i sense haver de reiniciar la nova pestanya Prunning ja és visible al Protégé.


collaborativePizza.owl Protégé 3.4.4 (file:\C:\Program%20Files\Protege_3.4.4\examples\collaborativePizza\collaborativePizza.owl.pprj, OWL / RDF Files)
File Edit Project OWL Reasoning Code Tools Window Collaboration Help
protégé
Metadata(pizza.owl) OWLClasses Properties Individuals Forms Pruning
Add row Remove row Execute Clear Help SlicedTomatoTopping Insert
Concepts of interest
Information and results

10 Conclusions

De la besant més teòrica dels objectius fixats en aquest projecte se'n desprèn, que tot i que les ontologies tenen els seus orígens en la filosofia, més concretament en el camp de la metafísica, la seva aplicació en el camp de la informàtica ha donat lloc a nous estudis que les han fet evolucionar i que han permès en poc temps, poc més de cinquanta anys, passar de les primeres definicions a la formalització d'ontologies grans i la seva aplicació efectiva.

Gran part del impuls que s'ha donat al seu estudi, ha estat motivat pel creixement exponencial de la informació a Internet i la necessitat d'organitzar i gestionar de forma eficaç aquesta informació. Així si tradicionalment es deia que 'la informació és poder', amb l'arribada d'Internet la preocupació ja no és tenir informació sinó saber trobar-la ella, gestionar-la i explotar-la.

Tradicionalment, aquestes operacions s'han realitzat mitjançant mineria de dades o bé mitjançant la utilització d'etiquetes i cerques sobre els textos. Però aquestes tècniques presenten limitacions, i no permeten obtenir un coneixement semànticament estructurat ni la conceptualització del domini que proporcionen les ontologies. Aquestes capacitats han permès que les ontologies es puguin aplicar a diversos camps, sobretot en el desenvolupament de la web semàntica. En aquest sentit cal destacar els esforços del consorci W3C en l'elaboració d'especificacions per la implantació d'una infraestructura que permeti la gestió documental de recursos nous i heterogenis i la recomanació d'utilitzar OWL com a llenguatge per definir les ontologies.

Però encara queda molt per fer en el camp de les ontologies i, tot i que els avenços han estat importants durant els darrers anys, encara manquen eines que afavoreixin la reusabilitat d'ontologies donant resultats correctes. Un exemple en són eines que facilitin la poda d'ontologies, la qual ha donat lloc a aquest projecte. Actualment al mercat existeixen eines que permeten aquesta operació però o bé són de codi propietari o bé són de codi obert però no són fiables. El plug-in per a Protégé que ha estat objecte d'aquest projecte ve a solucionar aquesta mancança mitjançant l'aplicació de l'algorisme de poda proposat per Jordi Conesa Caralt a la seva tesi doctoral *Pruning and Refactoring Ontologies in the Development of Conceptual Schemas of Information Systems*.

Part de la tesi doctoral defineix un algorisme per la poda d'ontologies que permet obtenir, a partir d'una ontologia més general, una nova ontologia correcta i aplicable al domini de context per el qual es vol utilitzar. L'algorisme de poda es divideix en dues fases, la fase de selecció de conceptes i la fase de poda, i la segona fase en quatre passes. A partir de la fase de selecció, cada una de les passes que conformen la fase de poda té com a punt de partida el conjunt de conceptes obtingut de la passa anterior; i el conjunt de conceptes resultant de cada passa serveix com a punt de partida de la passa següent. D'aquesta manera, d'un conjunt de conceptes inicial més general, cada passa el redueix gràcies a la poda d'elements innecessaris fins obtenir una ontologia que conté només els elements d'interès i els estrictament necessaris per què l'ontologia final sigui correcte.

L'adaptació de l'algorisme a l'entorn proporcionat per Protégé, ha provocat alguns canvis en la definició de l'algorisme degut a la impossibilitat tècnica d'aplicar-los sobre Protégé. Així, la fase de selecció dels conceptes d'interès, prèvia a la fase de poda i que estableix el conjunt de conceptes que desencadena la resta de l'algorisme, no permet en el plug-in desenvolupat l'extracció de conceptes d'interès de forma automàtica que és una de les possibilitats descrites per l'algorisme. El motiu és que Protégé no permet l'especificació de requeriments funcionals necessària per a l'extracció automàtica. Tanmateix sí que s'ha implementat la possibilitat de triar conceptes de

forma totalment manual mitjançant la introducció dels conceptes a la taula i de forma guiada a través d'un desplegable que conté les classes i restriccions de l'ontologia original.

Una altra modificació respecte a la definició original de l'algorisme ha estat la reducció de la fase de poda de quatre passes a tres. La passa no implementada ha estat la poda dels elements orfes. En aquest cas el propi Protégé elimina els elements que són instàncies de classes o restriccions eliminades. Tot i així, la supressió d'aquesta passa no afecta al resultat i és transparent a l'usuari.

Per la implementació de l'algorisme s'ha pres com a punt de partida el desenvolupament d'un plug-in ja creat en un projecte de fi de carrera anterior. El nou plug-in incorpora millores en la fase de selecció de conceptes d'interès permetent que a més de la introducció manual de conceptes, es pugui fer una incorporació guiada a partir d'un desplegable amb la llista de classes i restriccions. A més s'ha millorat la usabilitat de la pestanya afegint una taula per la introducció dels conceptes en lloc de l'àrea de text lliure; la ubicació de tots els botons necessaris per actuar sobre el plug-in, junts a la part superior; i per últim la incorporació d'una àrea per informar d'errors, resultats i ajuda a la mateixa pantalla evitant així la utilització de finestres.

Tanmateix han quedat alguns punts pendents o que es podrien millorar:

- La llista de conceptes sempre els mostra tots. De manera que encara que s'hagi triat un concepte i no tingui sentit tornar-lo a seleccionar al desplegable, continua apareixent. Tot i així la tria d'un element repetit no provoca cap error ja que internament es valida si existeix aquesta repetició i en cas afirmatiu, s'ignora el concepte.
- No es valida que un concepte que s'ha introduït escrivint directament a la taula, existeixi a l'ontologia original. Tot i que aquest requeriment es va definir a l'anàlisi, finalment no s'ha implementat
- El sistema d'introducció de conceptes pot millorar-se, ja que tot i ser més àgil la selecció des d'un desplegable, aquest sistema obliga a la introducció individual de conceptes que en ontologies grans pot resultar farragós. En futures versions es podria incorporar un control que permetés multiselecció o una segona pantalla des de la que es poguessin seleccionar diversos conceptes a l'hora.
- Tot i haver afegit alguns canvis en la implementació de l'algorisme, la seva execució requereix molt de temps. Caldria veure si és possible incorporar millores per fer-lo més eficient.

Finalment, relacionat amb l'elaboració del projecte s'ha hagut de destriar informació d'entre tota la que hi ha al web (que no sempre és correcte) fent evident la necessitat de l'aplicació de les ontologies en les cerques. A més ha calgut aprendre no només el concepte d'ontologia, sinó la creació d'aquestes i el funcionament de l'eina Protégé. A un nivell més tècnic també s'ha hagut d'apendre la programció amb el paquet Swing de Java.

11 Referències

[1] GRUBER, T. R.. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5 199-200, 1993.
[2] http://wordnet.princeton.edu/
[3] http://www.cyc.es/
[4] GRUBER, T. R.. A Translation Approach to Portable Ontology Specifications. Knowledge Acquisition, 5 199-200, 1993.
[5] http://tomgruber.org/writing/onto-design.htm
[6] Weigand, H.. "Multilingual Ontology-Based Lexicon for News Filtering –The TREVI Project", 138-159. (1997)
[7] GUARINO, N. "Understanding, Building, and Using Ontologies" en Kwnoledge Acquisition Workshop 1996.
[8] VAN HEIJST, G., SCHEREIBER, A.T. Y WIELINGA, B.J. "Using Explicit Ontologies in KBS Development" en International Journal of Human and Computer Studies, 1996
[9] Noy N. McGuinness D. “Ontology Development 101: A Guide to Creating your first Ontology”
[10] RDF/XML Syntax Specification (Revised), Dave Beckett, Editor, W3C Recommendation, 10 February 2004, http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/ . Latest version available at http://www.w3.org/TR/rdf-syntax-grammar/ .
[11] Adolfo Vasquez Rocca - http://adolfovrocca.bligoo.com/content/view/215944/EL-HIPERTEXTO-Y-LAS-NUEVAS-RETORICAS-DE-LA-POSTMODERNIDAD-Por-Adolfo-Vasquez-Rocca.html
[12] Farsani, H.K. y Nematbakhsh, M.A. (2006). "A Semantic Recommendation Procedure for Electronic Product Catalog". International Journal of Applied Mathematics and Computer Sciences, v. 3, pp. 86-91.
[13] http://roda.ibit.org/herramientas.cfm

[14] http://cohse.cs.manchester.ac.uk/

[15] Annotea project, http://www.w3.org/2001/Annotea/

[16] OWL Working Group - “OWL Web Ontology Language Use Cases and Requirements” http://www.w3.org/TR/2004/REC-webont-req-20040210/

[17] Ontoweb, portal basat en ontologies - http://www.ontoweb.org/

[18] The Open Directory Project - Portal basat en ontologies – http://www.dmoz.org

[19] Jordi Conesa Caralt - Pruning and Refactoring Ontologies in the Development of Conceptual Schemas of Information Systems (Oct 2007)

11 Bibliografia

http://www.iula.upf.edu/materials/**070223pedraza.pdf**

http://dipahurb.googlepages.com/AnteProyecto_DianaPaolaHurtado.pdf

http://dspace.upv.es/xmlui/bitstream/handle/10251/1828/tesisUPV2719.pdf?sequence=1

www.mdpi.com/journal/futureinternet

http://www.web.upsa.es/spdece08/contribuciones/176_Fermoso_Sanchez_Sicilia_LOMOWL.pdf

http://protege.stanford.edu/publications/ontology_development/ontology101-es.pdf

http://www.hipertext.net/web/pag286.htm

http://www.um.es/docencia/pastor/wp-content/uploads/2009/10/ibersid-2009-exposicion.pdf

http://leaenbinario.blogspot.com/2009/05/herramientas-para-trabajar-con.html

http://www.wshoy.sidar.org/index.php?2005/12/09/30-ontologias-que-son-y-para-que-sirven

http://www.web.upsa.es/spdece08/contribuciones/176_Fermoso_Sanchez_Sicilia_LOMOWL.pdf

http://www.w3.org/2003/08/owlfaq

http://www.w3.org/Submission/SWRL/

http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/

http://www.w3.org/2007/09/OWL-Overview-es.html