Le ontologie sono uno strumento sempre più diffuso tra gli sviluppatori Web per i vantaggi che offrono nella condivisione delle informazioni. In questo mini-corso di tre lezioni cercheremo, attraverso un approccio pragmatico basato su esempi, di introdurre il lettore alla realizzazione di ontologie ed al loro utilizzo nell'ambito di applicazioni Java. In questa prima puntata mostreremo in particolare come definire una semplice ontologia OWL, introducendo anche concetti elementari di modellazione della conoscenza.

Ontologie OWL: Teoria e Pratica

(prima puntata)

di Nicola Capuano

La parola *Ontologia* non ha un unico significato: nell'ambito della filosofia, esso è legato alla teoria dell'esistenza che si occupa di dare risposte a domande del tipo: cos'è l'esistenza? Quali proprietà possono spiegarla? Come queste proprietà la spiegano? Nell'ambito dell'Intelligenza Artificiale, invece, lo stesso termine indica la *specificazione di una concettualizzazione* [1] ovvero una descrizione formale di un insieme di concetti e delle relazioni che intercorrono tra essi. In quest'ultima accezione, il termine *Ontologia* sta diventando sempre più comune tra gli utenti e gli sviluppatori del Web.

Il Web, come è attualmente concepito, assomiglia ad un enorme continente di cui non esiste la mappa. Le uniche possibilità che abbiamo di recuperarvi risorse utili sono basate su ricerche sintattiche, che affidano la loro efficacia alla corrispondenza tra parole chiave. Con le ontologie, di contro, è possibile realizzare mappe semantiche del Web interpretabili sia dall'uomo che dalla macchina. L'approccio è quello *bottom-up*: non un'enorme mappa imposta dall'alto (ad esempio le categorie di Yahoo) ma tante piccole mappe interconnesse, realizzate da singoli utenti, a cui interessa formalizzare un dominio o, più semplicemente, aggiungere semantica al proprio sito Web.

Ma il *Semantic Web* (ovvero il Web esteso attraverso l'uso di ontologie) è solo una delle possibili applicazioni delle ontologie. Esistono numerosi altri campi di utilizzo tra cui: knowledge management, data integration, content management, e-commerce, information filtering, problem solving, ecc. Ogni volta che occorre condividere informazioni – indipendentemente dalle tecnologie utilizzate, dall'architettura delle informazioni e dal dominio di applicazione – è possibile (e spesso conveniente) ricorrere alle ontologie. Lo hanno già capito alcune aziende (Mondeca, Ontoprise, TopQuadrant, ecc.) che hanno fatto della realizzazione di soluzioni ontology-based il loro core business.

Scopo di questa serie di tre articoli è di fornire un'introduzione "pragmatica" all'uso delle ontologie. Impareremo ad utilizzare il linguaggio ormai più diffuso per l'espressione di ontologie: *OWL*, definito nel 2004 dal World Wide Web Consortium (gli ideatori, tra l'altro, di HTML, XML e RDF). Utilizzeremo l'editor open source *Protégé* per la definizione di ontologie OWL, che impareremo ad utilizzare all'interno di applicazioni Java attraverso il framework open source *Jena*. Utilizzeremo, inoltre, il software *Racer* sia per verificare la consistenza dell'ontologia sia per farvi inferenza.

Cominceremo col realizzare una semplice ontologia OWL introducendo, tra l'altro, alcuni concetti elementari di modellazione della conoscenza. Unico requisito per sperimentare gli esempi che verranno esposti è l'installazione della versione 3.0 *full* di Protégé, reperibile al sito protege.stanford.edu, nella sezione *Download*.

Il linguaggio OWL

OWL è rilasciato in tre diverse versioni, con complessità e potere espressivo crescenti. *OWL-Lite* è la versione sintatticamente più semplice, attraverso cui è possibile definire gerarchie di classi e vincoli poco complessi. *OWL-DL* è la versione intermedia, offre un potere espressivo più elevato e mantiene la completezza computazionale (tutte le conclusioni sono computabili) e la decidibilità (tutte le computazioni si concludono in un tempo finito). *OWL-Full* offre la massima espressività, senza offrire alcuna garanzia circa completezza e decidibilità.

Ogni versione del linguaggio include ed estende la precedente. In altre parole, un'ontologia OWL-Lite è sempre valida anche in OWL-DL (ma non il contrario); un'ontologia OWL-DL è sempre valida anche in OWL-Full (ma non il contrario). Questi articoli, in particolare, si concentreranno su OWL-DL, così chiamato per la sua corrispondenza con le Logiche Descrittive [2].

I componenti principali di un'ontologia OWL sono tre: individui, proprietà e classi. Gli *individui* rappresentano gli oggetti nel dominio di interesse, le *proprietà* sono relazioni binarie (ovvero che collegano due oggetti per volta) tra individui, le *classi* sono gruppi di individui. La **Figura 1** mostra un semplice esempio dove 7 individui (*Andrea*, *Milano*, *Napoli*, *Nicola*, *Roma*, *Salerno* e *Tonia*) sono raggruppati in 2 classi (*Città* e *Persone*) e relazionati attraverso 3 tipi di proprietà (*haFratello*, *haMoglie* e *viveInCittà*). Gli individui sono rappresentati come piccoli tondi pieni, le classi come ovali vuoti e le proprietà come archi direzionati.

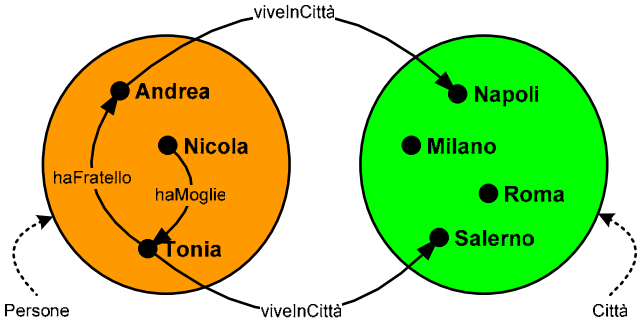

Figura 1. Rappresentazione schematica di alcune classi, proprietà ed individui.

Un'ontologia costruita sull'esempio citato ci direbbe che *Nicola*, *Tonia* ed *Andrea* sono *Persone* mentre *Napoli*, *Milano*, *Roma* e *Salerno* sono *Città*. *Nicola* ha per moglie *Tonia* che vive a *Salerno*. *Tonia*, a sua volta, ha un fratello che si chiama *Andrea* e vive a *Napoli*. Le classi OWL possono essere organizzate in gerarchie di superclassi e sottoclassi dette *tassonomie*. Nell'esempio precedente, dalla classe *Persone* è possibile derivare la sotto-classe *Uomini* e la sotto-classe *Donne*. Dire che *Uomini* è sotto-classe di *Persone* significa affermare che tutti gli *Uomini* sono *Persone*.

Dopo questa brevissima introduzione agli elementi fondanti di OWL, siamo pronti a realizzare la nostra prima ontologia con Protégé. Il dominio scelto ad insindacabile giudizio dell'autore è quello della birra.

Definiamo una tassonomia con Protégé

Lanciamo Protégé e creiamo un nuovo progetto prestando attenzione a selezionare il formato corretto: *OWL Files* (Protégé è in grado di gestire anche formati diversi da OWL). Per creare un nuovo progetto, all'avvio di Protégé è sufficiente selezionare l'opzione *OWL Files* dalla dialog box che appare e premere il pulsante *New*. In alternativa è possibile scegliere l'opzione *New Project* dal menu *File*. Ciò fatto, compare la GUI di Protégé (**Figura 2**).

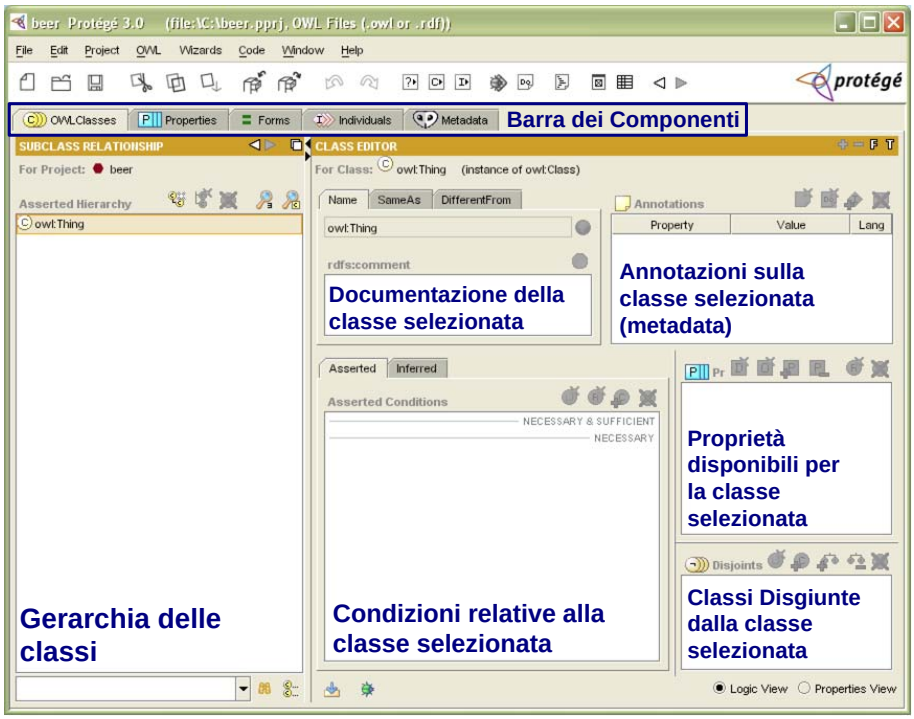

Figura 2. L'interfaccia utente di Protégé con il pannello delle classi attivo.

Il controllo più importante della GUI è la *Barra dei Componenti*, che permette di navigare tra i diversi elementi (classi, proprietà, individui, ecc.) dell'ontologia in fase di definizione. Il pannello inizialmente visualizzato è quello delle classi che consente, sulla sinistra, di definire la tassonomia e, sulla destra, di caratterizzare la classe selezionata. L'ontologia vuota contiene la sola classe *owl:Thing* da cui vengono derivate tutte le altre classi.

Per inserire una nuova classe nell'ontologia è sufficiente posizionarsi sulla classe di cui si vuole creare una sotto-classe e premere il pulsante *Create Subclass*. Ciò fatto, occorre scegliere il nome della nuova classe nel pannello *Name* della sezione *Documentazione*. La **Figura 3** mostra come creare la classe *Birra* come sotto-classe di *owl:Thing*.

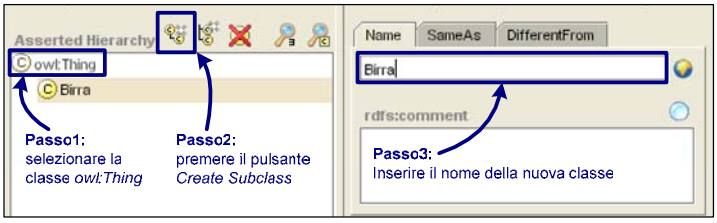

Figura 3. Creare una nuova classe.

Creiamo allo stesso modo anche le sottoclassi *Lievito*, *Luppolo* e *Malto* di *owl:Thing*. Vogliamo ora dichiarare che le quattro classi "sorelle" (ovvero derivate dalla stessa superclasse) *Birra*, *Lievito*, *Luppolo* e *Malto* sono tra loro *disgiunte*, ovvero che un individuo appartenente ad una non può contemporaneamente appartenere ad un'altra di esse (un lievito non può essere un luppolo, un malto non può essere una birra e così via). OWL non fa infatti alcuna assunzione sulla disgiunzione tra classi ed assume che le classi possano sovrapporsi, se non dichiariamo esplicitamente il contrario.

La disgiunzione può essere dichiarata selezionando una qualsiasi delle quattro classi sorelle e premendo il pulsante *Add all Siblings* nella sezione *Classi Disgiunte* (vedi **Figura 4**). Selezionando l'opzione *Mutually between all Siblings* dalla dialog box che appare ci assicuriamo, inoltre, non solo che la classe attuale sia disgiunta da tutte le sorelle, ma che anche ogni altra sorella sia disgiunta dalle altre.

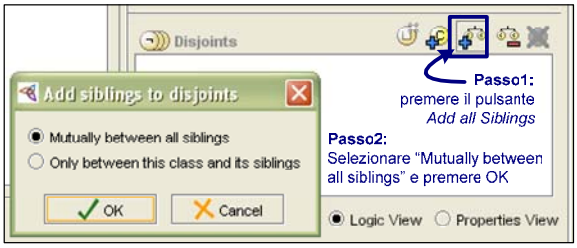

Figura 4. Rendere disgiunte tutte le classi derivate dalla stessa super-classe.

Aggiungiamo ora ulteriori sottoclassi, fino ad ottenere la tassonomia di **Figura 5**. Per farlo più velocemente possiamo ricorrere al wizard *Create Groups of Classes* che si trova nel menu *Wizards*. Assicuriamoci sempre che le classi allo stesso livello siano tra loro mutuamente disgiunte, applicando il procedimento testé descritto.

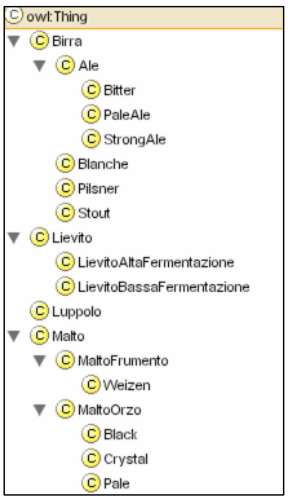

Figura 5. La tassonomia ottenuta.

Introduciamo alcune proprietà

Come dicevamo, le proprietà sono relazioni binarie tra individui. OWL ammette due tipologie principali di proprietà: le proprietà *object* che collegano individui ad individui e le proprietà *datatype* che collegano individui a valori di tipi ammissibili per XML e/o RDF. Per definire le proprietà in Protégé è necessario accedere all'omonimo pannello, mostrato in **Figura 6**, attraverso la barra dei componenti.

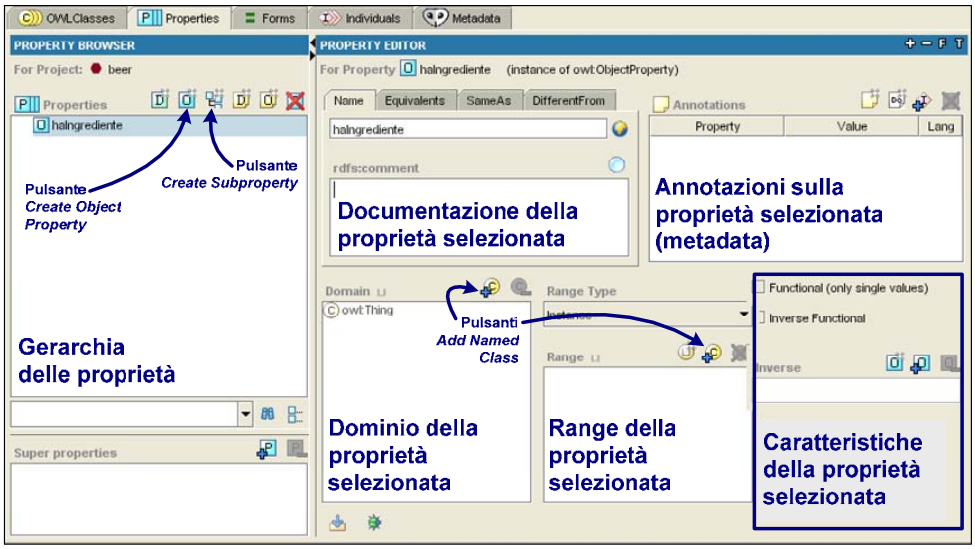

Figura 6. Il pannello *Properties* di Protégé.

Anche questo pannello è suddiviso in due sezioni principali: la parte sinistra mostra la gerarchia delle proprietà mentre la parte destra consente di caratterizzare la proprietà selezionata. Anche le proprietà possono infatti essere organizzate in gerarchie, dove ogni sotto-proprietà specializza la proprietà da cui è derivata.

Inserire una nuova proprietà è del tutto simile ad inserire una nuova classe. L'unica differenza è che è possibile creare una proprietà senza antenati. Proviamo a creare ad esempio la proprietà *haIngrediente* e le sottoproprietà *haLuppolo*, *haMalto* e *haLievito*. Per creare la proprietà padre è necessario utilizzare il pulsante *Create Object Property* (**Figura 6**) mentre per le figlie è necessario posizionarsi su *haIngrediente* ed utilizzare il pulsante *Create Subproperty*. I nomi delle proprietà vanno inseriti nel pannello *Name* della sezione *Documentazione*.

Il *dominio* di una proprietà è la classe (o l'insieme di classi) ai cui individui appartenenti si può applicare la proprietà. Il *range* (o codominio) di una proprietà è invece la classe (o l'insieme di classi) i cui individui appartenenti possono essere valori della proprietà. In altre parole, una proprietà collega individui appartenenti ad un dominio ad individui appartenenti ad un range. Le sezioni *Dominio* e *Range* del pannello *Properties* hanno la funzione di raccogliere queste informazioni. Per inserirvi una nuova classe è necessario utilizzare il pulsante *Add Named Class* (**Figura 6**) e scegliere la classe da inserire dalla dialog-box che appare.

Nella nostra ontologia, ad esempio, la proprietà *haIngrediente* ha come dominio la classe *Birra*. Anche le proprietà *haLuppolo*, *haMalto* e *haLievito* hanno come dominio la classe *Birra*, ma non è necessario specificarlo, dato che il dominio per le tre proprietà è ereditato dalla super-proprietà *haIngrediente*. Viceversa è conveniente stabilire il range delle tre proprietà figlie rispettivamente nella classe *Luppolo*, *Malto* e *Lievito*. Ciò facendo, dichiariamo che se ad un individuo della classe *Birra* viene associato un altro individuo attraverso la proprietà *haLuppolo*, quest'ultimo deve essere della classe *Luppolo* o di una delle sue sottoclassi. Lo stesso dicasi per le proprietà *haMalto* e *haLievito* rispetto alle classi *Malto* e *Lievito*.

Le proprietà OWL possono avere diverse caratteristiche, tra cui la transitività, la simmetria e la funzionalità. È possibile inoltre dichiarare che alcune proprietà sono inverse di altre. Non approfondiremo per il momento questi argomenti.

Creiamo alcune restrizioni sulle classi

Un tipico utilizzo delle proprietà è nella definizione di restrizioni di classe, ovvero di restrizioni all'insieme di individui che possono appartenere ad una classe. Le restrizioni possono essere di vario tipo. Ci focalizzeremo in prima istanza sulle "restrizioni di quantità". Tali restrizioni sono composte da un quantificatore, da una proprietà e da un argomento (o *filler*). Le restrizioni che utilizzano il quantificatore esistenziale (∃) sono dette restrizioni esistenziali. Le restrizioni che utilizzano il quantificatore universale (∀) sono dette restrizioni universali.

La “restrizione esistenziale” in OWL identifica l’insieme di individui che, per una data proprietà, hanno *almeno* relazioni con individui di una specifica classe. Ad esempio una restrizione del tipo ∃ *haMalto MaltoOrzo* specifica la classe di tutti gli individui che hanno almeno una relazione di tipo *haMalto* con un individuo della classe *MaltoOrzo*. Ciò non esclude gli individui che hanno relazioni di tipo *haMalto* anche con individui esterni alla classe *MaltoOrzo* (vedi **Figura 7**). La “restrizione universale”, invece, identifica l’insieme di individui che, per una data proprietà, hanno *al più* relazioni con individui di una specifica classe. Ad esempio una restrizione del tipo ∀ *haMalto MaltoOrzo* specifica la classe di tutti gli individui che hanno relazioni di tipo *haMalto* solo con individui della classe *MaltoOrzo*, più tutti gli individui che *non* hanno relazioni di tipo *haMalto* (vedi, ancora, **Figura 7**).

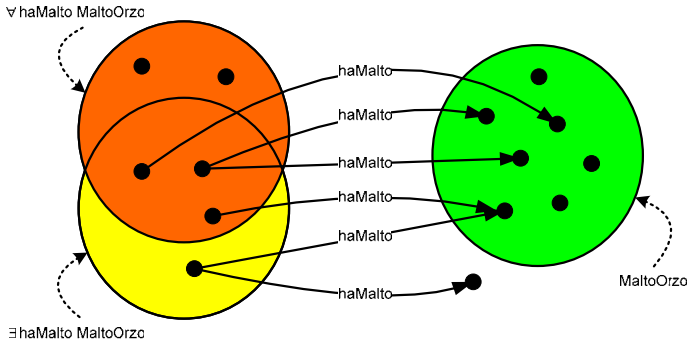

Figura 7. Esempi di restrizioni esistenziali ed universali.

Vediamo ora come applicare queste nuove nozioni in un caso concreto legato all’universo birrario. Le birre sono ottenute da una miscela di malti (eventualmente uno solo) tra cui, obbligatoriamente, del malto d’orzo. Alcune birre, come ad esempio le *Blanche*, hanno anche malto di frumento. Altre birre, tra cui le *Ale*, le *Pilsner* e le *Stout* hanno solo malto d’orzo. Come fare ad esprimere queste restrizioni?

In primo luogo occorre dichiarare che una condizione necessaria perché un individuo appartenga alla classe *Birra* è che tra gli ingredienti ci sia almeno un malto d’orzo, ovvero che l’individuo abbia almeno una relazione di tipo *haMalto* con un individuo della classe *MaltoOrzo*, ovvero che l’individuo appartenga alla classe ∃ *haMalto MaltoOrzo*. ∃ *haMalto MaltoOrzo* è dunque una restrizione (condizione necessaria) per la classe *Birra*.

Per inserire questa restrizione nella nostra ontologia è necessario tornare al pannello delle classi attraverso la *Barra dei Componenti***,** selezionare la classe *Birra*, selezionare la riga *Necessary* nella sezione *Condizioni* e premere il pulsante *Create Restriction* (vedi **Figura 8**). Ciò fatto compare la palette *Create Restriction*, nella quale occorre selezionare la proprietà *haMalto* dall’elenco delle proprietà, ∃ dall’elenco delle restrizioni e inserire il nome della classe *MaltoOrzo* nel filler (eventualmente aiutandosi con l’elenco che si ottiene premendo il pulsante *Insert Class*). Premendo *OK*, la restrizione viene creata ed aggiunta alla sezione restrizioni della classe *Birra*.

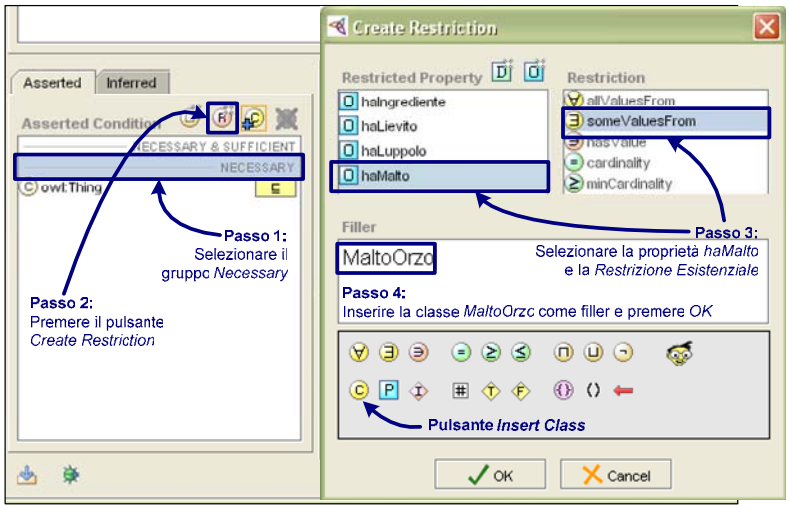

Figura 8. Creare una nuova restrizione.

Allo stesso modo, per dichiarare che una *Blanche* deve avere tra gli ingredienti almeno un malto di frumento, occorre aggiungere la restrizione ∃ *haMalto MaltoFrumento* alla classe *Blanche*. Invece, per dire che le *Ale*, le *Pilsner* e le *Stout* hanno solo malti d'orzo, occorre aggiungere la restrizione ∀ *haMalto MaltoOrzo* per le classi *Ale*, *Pilsner* e *Stout*.

La **Figura 9** riassume le condizioni della classe *Birra* e delle sue sottoclassi dopo le ultime operazioni. Come si può notare, le condizioni sono suddivise tra "necessarie e sufficienti" (che non tratteremo ora), "necessarie" ed "ereditate". Tra le condizioni necessarie è automaticamente inserita anche la super-classe: la prima condizione necessaria perché un individuo appartenga ad una classe è infatti che esso appartenga anche alla sua super-classe o, meglio, all'insieme delle sue super-classi (OWL supporta l'ereditarietà multipla). Ogni sottoclasse, inoltre, eredita tutte le condizioni delle super-classi che vengono automaticamente inserite nel gruppo *Inherited*.

Per esercitarci proviamo ad aggiungere altre restrizioni per dichiarare, ad esempio, che le *Pilsner* sono birre fatte con lieviti a bassa fermentazione mentre le *Ale*, le *Blanche* e le *Stout* sono fatte con lieviti ad alta fermentazione. Possiamo dichiarare, inoltre, che le *Stout* sono fatte con almeno un malto di tipo *Black*, mentre le *Bitter* e le *Pilsner* sono fatte solo con malti di tipo *Pale*.

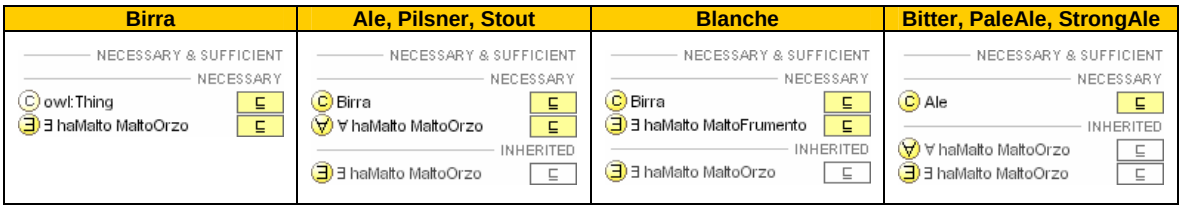

Figura 9. Sommario delle condizioni sulla classe *Birra* e sulle sue sotto-classi.

Una volta terminato, non dimentichiamo di salvare il nostro lavoro selezionando la voce *Save Project* dal menu *File* e scegliendo un nome adeguato per la nostra ontologia (ad esempio "birra.owl"). Oltre al file con estensione *owl*, Protégé crea un ulteriore file con estensione *pprj* che include informazioni aggiuntive relative al progetto, necessarie all'editor alla successiva riapertura. Non approfondiremo in questa sede la sintassi del linguaggio OWL e quindi il contenuto del file generato. Se siete interessati a questo argomento, troverete tutte le informazioni necessarie nella sezione del sito del World Wide Web Consortium dedicato ad OWL (vedi il riquadro dei Riferimenti).

Conclusioni

In questo articolo abbiamo imparato a creare ed a salvare una semplice ontologia in OWL. Nella prossima puntata proseguiremo la trattazione delle ontologie e delle loro applicazioni informatiche. In particolare verranno introdotte varie tipologie di proprietà e si considereranno ulteriori metodi di definizione delle classi (OWL ne supporta ben 6). Si imparerà inoltre a creare individui appartenenti a classi e ad usare un reasoner sia per il controllo di consistenza che per la classificazione automatica dell'ontologia. Nella terza ed ultima puntata, invece, impareremo ad utilizzare le ontologie nell'ambito di applicazioni Java.

L'ontologia della birra utilizzata come esempio in questo articolo è allegata come file sorgente all'indirizzo ftp.infomedia.it; può essere importata in Protégé selezionando la voce *Build New Project* dal menu *File*. L'importazione può avvenire anche direttamente da Internet fornendo l'URL dell'ontologia da importare piuttosto che il nome del file sul computer locale. Alcune letture utili a chi voglia approfondire l'argomento dell'editing di ontologie OWL con Protégé sono riportate nella bibliografia (in particolare [3] e [4]).

Bibliografia

[1] T.R. Gruber – "A Translation Approach to Portable Ontology Specification", Knowledge Acquisition 5, 1993.

[2] AA. VV. – "The Description Logic Handbook", Cambridge University Press, 2003.

[3] N. F. Noy, D. L. McGuinnes – "Ontology Development 101: A Guide to Creating Your First Ontology", http://www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.pdf, 2001.

[4] M. Horridge, H. Knublauch, A. Rector, R. Stevens, C. Wroe1 – "A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools", http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf, 2004.

[5] http://www.w3c.org/2004/OWL – *la sezione del sito del World Wide Web Consortium dedicata a OWL.*

[6] http://www.w3c.org/2001/sw – *la sezione del sito del World Wide Web Consortium dedicata al Semantic Web.*

[7] http://protege.stanford.edu – *il sito di Protégé ospitato dalla Stanford University.*

[8] http://pear.cs.umbc.edu/swoogle – *un motore di ricerca per reperire ontologie sul Web.*