

POLITECNICO DI MILANO

Facoltà di Ingegneria dell'Informazione

Corso di laurea in Ingegneria Informatica



**PROGETTO E IMPLEMENTAZIONE DI
UN'ONTOLOGIA DI CONTESTO IN UN WIKI
SEMANTICO**

Relatore: Prof. Marco Colombetti

Correlatore: Ing. Davide Eynard

Elaborato di Laurea di:

Giulia Rodà Matr. n° 672235

Anno accademico 2006 - 2007

Ai miei genitori,

a Nicoletta

INDICE

POLITECNICO DI MILANO	1
Facoltà di Ingegneria dell'Informazione	1
Corso di laurea in Ingegneria Informatica	1
.....	1
PROGETTO E IMPLEMENTAZIONE DI UN'ONTOLOGIA DI CONTESTO IN UN WIKI SEMANTICO	1
Anno accademico 2006 - 2007	1
Introduzione	1
Due parole sulla nascita del Web	1
1La Conoscenza come mezzo per il passaggio da Web a Web semantico	4
1.1 Web e Semantica.....	4
1.2 La Semantica e la Conoscenza.....	4
1.3 I linguaggi del Web.....	7
1.3.1 XML.....	8
1.3.2 RDF (Resource Description Framework).....	8
1.3.3 RDFS.....	11
1.3.4 OWL (Web Ontology Language).....	12
CAPITOLO 2	15
2Fusione tra sistema Wiki e Tecnologie del Semantic Web: Wiki Semantico	15
2.1 Sistemi Wiki tradizionali.....	15
2.1.1 Editing via browser.....	17
2.1.2 Sintassi Wiki semplificata.....	17
2.1.3 Forte struttura ipertestuale.....	17
2.1.4 Accesso libero e meccanismo di rollback	18
2.1.5 Editing collaborativi.....	18
2.2 Wiki Semantici.....	19
2.2.1 Tipizzazione/annotazione dei link e tag.....	19
2.2.2 Ricerca semantica.....	20
2.2.3 Reasoning.....	20
2.2.4 Interfaccia utente.....	21

2.2.5 Limiti e problematiche.....	21
2.2.6 I livelli di ontologie in un Wiki Semantico.....	22
CAPITOLO 3.....	24
3 Ambiente di sviluppo e strumenti utilizzati.....	24
3.1 Editor di ontologie: Protégé.....	24
3.2 Graphic OWL (GrOWL).....	28
3.3 Jena.....	29
3.4 Linguaggio di interrogazione SPARQL.....	30
3.5 Pellet: un reasoner per OWL-DL.....	31
3.6 JSPWiki.....	33
3.6.1 Nascita e diffusione.....	34
3.6.2 Peculiarità di JSPWiki.....	35
CAPITOLO 4.....	37
4 Ontologia di contesto.....	37
4.1 I concetti principali.....	37
4.2 Documenti associati e complementari.....	39
4.2.1 La reificazione.....	39
4.2.2 Associazione e Complementarietà.....	39
4.2.3 Propedeuticità e Approfondimento.....	41
4.3 Relazioni per la classificazione di documenti.....	42
4.4 Completamento del modello.....	44
4.5 Esempio di ontologia.....	44
CAPITOLO 5.....	46
5 Implementazione in JSPWiki.....	46
5.1 Classe <i>Ontology.java</i>	48
5.1.1 I metodi principali.....	48
5.1.2 Esecuzione delle query.....	49
5.2 Modifica del motore JSPWiki.....	50
5.2.1 Struttura delle pagine di JSPWiki.....	51
5.2.2 Visualizzazione delle informazioni semantiche.....	51
5.2.3 Creazione di relazioni tra documenti tramite un'interfaccia grafica.....	52
CAPITOLO 6.....	53
6 Conclusioni e sviluppi futuri.....	53

<i>7Bibliografia.....</i>	<i>55</i>
<i>8Ringraziamenti.....</i>	<i>56</i>
<i>Appendice A.....</i>	<i>57</i>
<i>Software utilizzato.....</i>	<i>57</i>

Introduzione

Due parole sulla nascita del Web

È il mese di settembre del 1989, Tim Berners Lee, un giovane consulente al Cern di Ginevra, classe 1955, laureatosi nel 1976 al Queen's College dell'Università di Oxford, redige un breve rapporto, intitolato "Management dell'informazione: una proposta". L'idea di Tim è quella di creare un ambiente di comunicazione universale, decentralizzato, basato sull'ipertesto: il sistema di scrittura che consente di collegare tra loro diversi documenti come se si trattasse di una tela, una trama di tessuto nella quale tutti gli elementi sono tra loro collegati.

Nel mese di agosto del 1991, il progetto World Wide Web diventa pubblico e viene messo a disposizione tramite alcuni gruppi di discussione su Internet. Della nuova tecnologia non viene richiesto nessun brevetto, chiunque può utilizzarla senza pagare alcun diritto al Cern o al suo inventore. È la fortuna di Internet, la caratteristica che ha fatto della Rete il sistema più aperto, più facilmente implementabile nella storia dell'informatica, e che gli ha consentito di diffondersi rapidamente, diventando ciò che i suoi inventori desideravano: una vera Ragnatela Mondiale.

Il Web quindi è nato inizialmente per soddisfare un bisogno: quello di permettere a tutti di caricare i propri contenuti e permettere agli altri di leggerli, eliminando le barriere fisiche. Poi come ogni cosa geniale ha scavalcato le proprie barriere iniziali e ha preso vita propria, cambiando il mondo: il Web ormai non è più uno strumento per "appassionati", ma è diventato parte integrante della vita comune dell'essere umano.

Introduzione

Da quanto detto, upload di contenuti, collaborazione, comunicazione immediata, e concetto di rete sociale sono intrinseci nell'architettura della Rete stessa; nella concezione di Berners-Lee, infatti, il web come lo conosciamo oggi è solo il primo gradino di una scala che prevede al gradino successivo il web semantico.

Con il termine web semantico si intende la trasformazione del [World Wild Web](#) in un ambiente dove i documenti pubblicati (pagine [HTML](#), file, immagini, e così via) sono associati ad informazioni e dati ([metadati](#)) che ne specificano il contesto semantico in un formato adatto all'interrogazione, all'interpretazione e, più in generale, all'elaborazione automatica[WWS].

Con l'interpretazione del contenuto dei documenti che il Web Semantico propugna, saranno possibili ricerche più evolute delle attuali, basate sulla presenza nel documento di parole chiave, ed altre operazioni specialistiche come la costruzione di reti di relazioni e connessioni tra documenti secondo logiche più elaborate del semplice link ipertestuale.

Ed è proprio in questo processo di semantizzazione dei dati divulgati sul Web che si inserisce il presente lavoro di tesi, con l'obiettivo di proporre una possibile strada per reperire informazioni legate fra loro da relazioni semantiche (es. relazioni di propedeuticità), indipendenti dal contenuto dei documenti stessi.

Il primo capitolo introduce il concetto di semantica, la sua applicazione nel campo informatico e la nascita dell'Ingegneria della Conoscenza come basi per lo sviluppo del Web Semantico.

Nel secondo capitolo vengono presentati i sistemi Wiki, in cui si è scelto di sperimentare l'ontologia di contesto, e i vantaggi che la semantica può portare in questo tipo di sistemi.

Il terzo capitolo si focalizza sulla descrizione e il funzionamento dei tool utilizzati per il progetto del modello semantico, per poi illustrare le caratteristiche del motore JSPWiki, in cui è stata implementata l'ontologia.

Introduzione

Il capitolo numero quattro descrive la fase di sviluppo dell'ontologia, mostrando passo passo come si è giunti all'identificazione dei concetti e delle proprietà che li legano.

Nel capitolo cinque viene illustrata la struttura e le modifiche apportate al sistema Wiki, per ottenere un'interfaccia grafica che permetta all'utente l'accesso alle informazioni semantiche sottostanti.

Infine, il sesto capitolo stila un breve bilancio dei risultati raggiunti e dei possibili sviluppi futuri che potrebbe avere questo progetto.

CAPITOLO 1

La Conoscenza come mezzo per il passaggio da Web a Web semantico

1.1 Web e Semantica

Il World Wide Web ha rivoluzionato le comunicazioni e il commercio, ma la maggior parte del suo contenuto è progettata per essere letta da esseri umani, e non per essere manipolato da programmi. I computer possono scorrere le pagine Web alla ricerca di schemi, ma non hanno un sistema adeguato per trattare la semantica.

L'idea del Web semantico è quella di dare una struttura al contenuto significativo della pagine Web, creando un ambiente dove gli agenti software possano svolgere velocemente compiti complessi per i loro utenti.

Per comprendere meglio i vantaggi che derivano dall'integrazione delle tecnologie del Web con funzioni semantiche, è necessario innanzitutto dare una definizione della Semantica e del suo uso in campo informatico.

1.2 La Semantica e la Conoscenza

Il termine semantica (dal greco *semantikos*, “significato”, derivato da *sema*, “segno”) è usato in molte diverse accezioni, tutte attinenti al concetto di “significato” di un “messaggio” in senso ampio. In alcuni casi, esso viene contrapposto a sintassi, intesa in senso ampio come “forma esteriore” del “messaggio”[Sem].

Il concetto di significato, a sua volta, chiama in causa il rapporto fra linguaggio e realtà. Secondo una tradizione antica (che nella storia del pensiero occidentale si può far risalire ad Aristotele), il rapporto fra linguaggio e realtà non è diretto, ma è mediato dai concetti: chi parla l'italiano riesce a collegare la parola "gatto" ai gatti reali proprio perché possiede un concetto di gatto; in altre parole, la relazione fra linguaggio e realtà è mediata dalla mente. La relazione fra queste entità è schematizzata in triangoli:

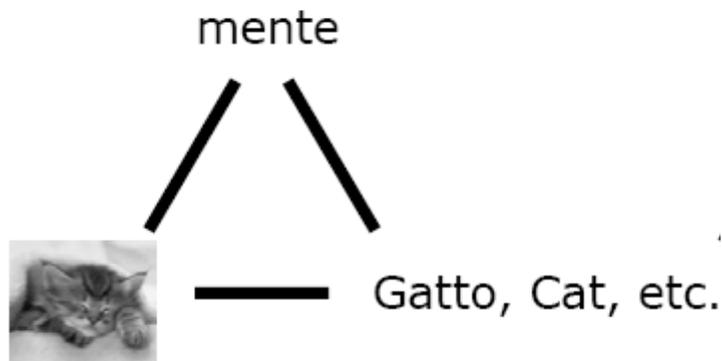


Figura 1.1: Il concetto di "Gatto" rappresentato tramite il triangolo aristotelico.

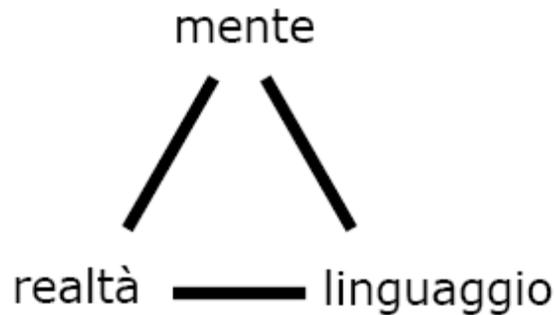


Figura 1.2: Rappresentazione dei concetti tramite i triangoli aristotelici.

Capitolo 1 – La Conoscenza come mezzo per il passaggio da Web a Web semantico

Una volta definita la semantica di un “messaggio”, l'obiettivo è sfruttare le informazioni trovate e tramutarle in conoscenza: in prima approssimazione la conoscenza può essere definita come informazione disponibile per l'azione (Dretske, 1981).

Vari sono i tipi di conoscenze e una prima distinzione può essere effettuata tra:

- conoscenza dichiarativa (knowing that, conoscere)
- conoscenza procedurale (knowing how, saper fare)

Si conoscono le regole della dama e si sa giocare a scacchi.

In inglese la distinzione viene rimarcata utilizzando due verbi differenti: to know e to can, rispettivamente conoscere e essere in grado di. A sua volta, il “conoscere” può essere classificato in tre tipi di conoscenze:

- conoscenze terminologiche (o analitiche): conoscenze legate al lessico di una lingua. Ad esempio, la parola dizionario significa collezione di parole in ordine alfabetico corredate della loro corretta grafia, etimologia e del loro significato.
- conoscenze nomologiche: sono conoscenze di regolarità, legate a leggi generali che governano il mondo. Sono conoscenze sempre valide (la Terra è un pianeta) o riconosciute valide nella maggior parte dei casi (l'acqua bolle a cento gradi).
- conoscenze fattuali: conoscenza di fatti particolari, non legati a conoscenze nomologiche o terminologiche (Silvia gioca a pallavolo) [Col06].

È solo grazie alle loro conoscenze che gli esseri umani possono agire in modo razionale, invece nei calcolatori, nonostante risiedano grandi quantità di

informazioni, solo una piccola parte di queste può essere sfruttata dagli stessi computer per agire: basta pensare che la stragrande maggioranza di file memorizzati nei computer è costituita da documenti in linguaggio naturale!

Ed è proprio in questo contesto che si inserisce l'Ingegneria della Conoscenza, o knowledge engineering, KE (termine coniato per la prima volta da E. Feigenbaum, 1977), intesa come ramo applicativo dell'intelligenza artificiale che si occupa del progetto, della realizzazione e del mantenimento di sistemi basati su conoscenze (knowledge based system, KBS). In generale, un sistema basato su conoscenze è un sistema informatico in grado di sfruttare le informazioni contenute in una base di conoscenze (knowledge base, KB) mediante procedure automatiche di ragionamento.

Affinchè le conoscenze siano sfruttabili da agenti artificiali, è necessario rappresentarle in un formato opportuno, che sia computer readable: la comunità di utenti, incluse alcune organizzazioni dedite all'elaborazione di standard come Internet Engineering Task Force (IETF) e il World Wide Web Consortium (W3C)¹, stanno dirigendo i loro sforzi verso la specifica, lo sviluppo e l'elaborazione di linguaggi adatti alla rappresentazione di una semantica condivisa dei dati sul web.

1.3 I linguaggi del Web

¹ Associazione fondata nell'Ottobre del 1994 dal padre del Web, Tim Berners Lee, con lo scopo di migliorare gli esistenti protocolli e linguaggi per il WWW e di aiutare il Web a sviluppare tutte le sue potenzialità.

1.3.1 XML

Per la costruzione/definizione del Semantic Web si utilizza l'XML, un linguaggio che consente di descrivere semanticamente (e con il dettaglio desiderato) le diverse parti di un documento. Un documento così descritto può poi essere elaborato per usi diversi: estrazione di informazioni secondo criteri specifici, riformulazione più o meno parziale, visualizzazione in funzione delle capacità del terminale.

Sebbene sia un buon modo per specificare informazioni, un documento, ancorché espresso in formato XML, è poco adatto al Web che per sua natura è distribuito e decentralizzato e quindi, informazioni su una particolare entità possono essere localizzate ovunque: la sintassi XML permette infatti di descrivere adeguatamente i contenuti di un documento, ma non definisce alcun meccanismo esplicito per qualificare le relazioni tra documenti. A tal fine non è di aiuto neppure il meccanismo dei collegamenti ipertestuali, reso popolare dall'HTML, perché amorfo, cioè privo della possibilità di descrivere il legame definito.

In altre parole, sebbene in un documento (ad es. una pagina HTML) sia possibile parlare di un Signor Eric Miller ed esprimere semanticamente questo con opportuni tag, è poi difficile capire se due documenti che parlano entrambi di un Signor Eric Miller si riferiscano alla stessa persona con conseguente scarsa qualità dei risultati restituiti dai motori di ricerca.

1.3.2 RDF (Resource Description Framework)

L'evoluzione del Web in Web Semantico inizia con la definizione, da parte del W3C, dello standard Resource Description Framework (RDF), una particolare applicazione XML che standardizza la definizione di relazioni tra informazioni ispirandosi ai principi della logica dei predicati (o logica predicativa del primo ordine FOL) e ricorrendo agli strumenti tipici del Web (ad es. URI) e dell'XML (namespace).

In estrema sintesi, secondo la logica dei predicati le informazioni sono esprimibili con asserzioni (statement in inglese) costituite da triple formate da soggetto, predicato e valore (in inglese spesso identificati come subject, verb e object, rispettivamente).

RDF assegna degli URI (Universal Resource Identifiers) specifici per ogni individuo a cui si riferisce. La figura 1.3 mostra un esempio di grafo RDF tratto dal W3C RDF Primer, in cui viene rappresentata una persona di nome Eric Miller. Quando creiamo i nodi e gli archi di un grafo RDF, un URI usato come nodo, identifica la risorsa rappresentata dal nodo stesso, mentre un URI usato come predicato rappresenta la relazione esistente tra gli oggetti identificati dai nodi connessi.

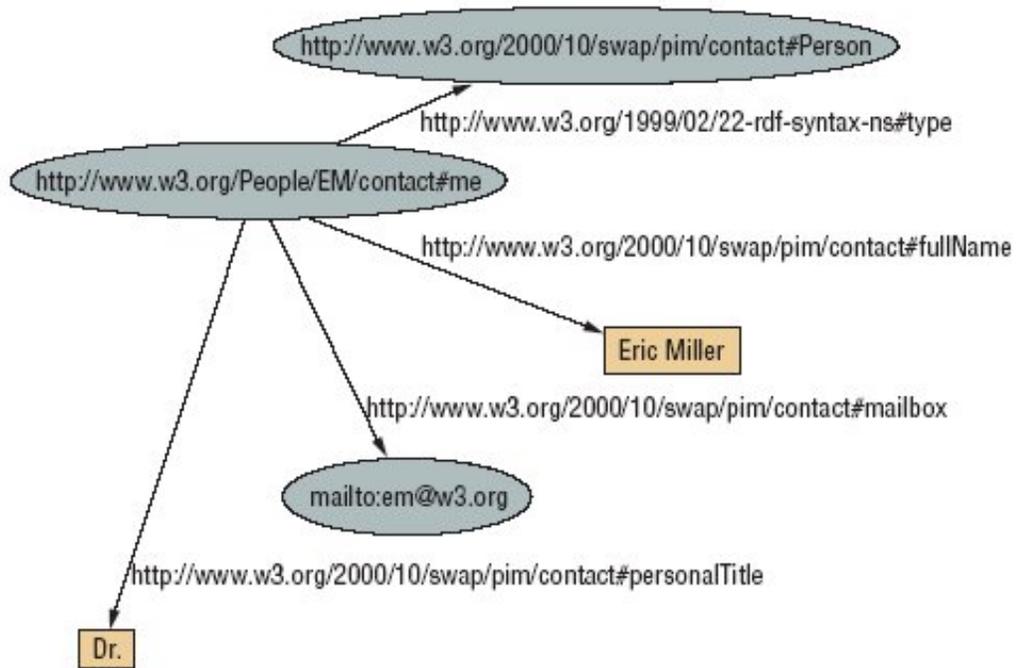


Figura 1.3: Grafo RDF che rappresenta il sig. Eric Miller

La figura 1.3 rappresenta:

- individui - come “EricMiller”, identificato dall'URI `http://www.w3.org/People/EM/contact#me`;
- tipi di risorse - come “Persona”, identificata da `http://www.w3.org/2000/10/swap/pim/contact#Person`;
- proprietà di queste risorse - come “mailbox”, identificata da `http://www.w3.org/2000/10/swap/pim/contact#mailbox`;
- valore delle proprietà- ad esempio “mailto:em@w3.org” è il valore della relazione “mailbox” (RDF usa anche stringhe di caratteri e valori appartenenti ad altri tipi di dati (interi, date,ecc.), come oggetti delle relazioni).

RDF inoltre usa una sintassi chiamata RDF/XML per il passaggio dai grafi al codice XML; un esempio è dato dalla figura 1.4 che rappresenta il codice RDF/XML corrispondente al grafo precedente. Risulta evidente come documenti, quali quello mostrato in figura 1.4, manchino di trasparenza e leggibilità; questo difetto potrebbe inibire verso un'adozione rapida di RDF, tuttavia esistono delle

forme alternative di più facile interpretazione: per esempio la notazione N3² [Paper2].

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:contact="http://www.w3.org/2000/10/swap/pim/contact#">
  <contact:Person rdf:about="http://www.w3.org/People/EM/contact#me">
    <contact:fullName>Eric Miller</contact:fullName>
    <contact:mailbox rdf:resource="mailto:em@w3.org"/>
    <contact:personalTitle>Dr.</contact:personalTitle>
  </contact:Person>
</rdf:RDF>
```

Figura 1.4: Formalizzazione del grafo di figura 1.3 in RDF/XML

1.3.3 RDFS

RDF fornisce un modo per descrivere generiche asserzioni su risorse e proprietà; è spesso necessario indicare il fatto che queste si riferiscono a particolari tipi di risorse ed usano specifiche proprietà. Per descrivere le classi e le relazioni utilizzate

² N3 (noto anche come *N-triples* o *Notation 3*), propone una forma più facile da leggere rispetto ad RDF e l'esempio che stiamo considerando trova la seguente soluzione:

```
<http://www.w3.org/People/EM/contact#me>
<http://www.w3.org/2000/10/swap/pim/contact#fullName>
"Eric Miller"
```

Ciascuna asserzione può essere scritta anche su un'unica linea, mettendo soggetto, predicato e valore l'uno di seguito all'altro.

per costruire il particolare modello RDF si utilizza ancora RDF. Il particolare vocabolario è definito dall’RDF Vocabulary Description Language, altrimenti noto come RDF-Schema, diventato raccomandazione del W3C nel 2004; esso estende RDF prendendone il modello di base e definendo un meccanismo per specificare classi e proprietà e costruire lo specifico vocabolario; in altre parole fornisce un sistema di tipi (semantici) per RDF.

In particolare ricordiamo che in RDFS è possibile esprimere il concetto di sottoclasse `rdfs:subClassOf`, dove una classe è un aggregato di individui che è in relazione `rdf:type` con la risorsa `rdfs:Class`, e anche le proprietà possono essere viste come particolari classi (e come tali avere sottoproprietà ed essere combinate con diversi costruttori).

1.3.4 OWL (Web Ontology Language)

La logica predicativa del primo ordine è estremamente complessa, e RDF ne poteva esprimere una porzione molto ristretta. Non solo: questa logica, se presa complessivamente non è nemmeno computabile, mentre possono essere computabili delle logiche costituite da sottoinsiemi degli operatori della logica del primo ordine. Questi sottoinsiemi della logica formale sono studiati dalle Logiche descrittive, o Description Logics ed una di queste è stata adottata per la formulazione di un nuovo standard, più ricco ed espressivo di RDF: OWL.

Come altri linguaggi precedenti volti agli stessi scopi, inclusi OIL, DAML e DAML+OIL, OWL è stato creato sulla base dei costrutti RDF e RDFS.

Tuttavia OWL offre numerosi nuovi costrutti, due dei quali risultano di notevole importanza: l’equivalenza tra risorse e la relazione inversa. Per equivalenza tra risorse (`owl:equivalentClass`) si intende la possibilità di affermare che due o più risorse rappresentano il medesimo individuo; per inversa (`owl:inverseOf`) si intende la possibilità di dire che se è vero (soggetto, predicato, oggetto), allora è anche vero (oggetto, predicato_inverso, soggetto) [WebS].

Lo standard W3C presenta tre versioni di OWL, a seconda del potere espressivo di cui si necessita:

- OWL-Lite: è il sottolinguaggio sintatticamente più semplice. È adatto per essere usato in situazioni per cui sono necessari vincoli e gerarchie semplici. È computabile (ossia è possibile trovare tutte le soluzioni in un tempo finito) ma poco espressivo;
- OWL-DL: è molto più espressivo di OWL-Lite ed è basato sulle Description Logics (dai cui il suffisso DL) e in quanto tale è computabile e adatto al ragionamento automatico;
- OWL-Full: è il più espressivo tra i sottolinguaggi di OWL, ed è usato per situazioni in cui un alto potere espressivo del linguaggio è più importante della garanzia di decidibilità e di completa computabilità.

OWL è stato sviluppato per la stesura di ontologie, ossia la descrizione di porzioni di realtà in cui vengono presi in considerazione dei concetti (classi), degli individui che soddisfano tali concetti, e delle proprietà (relazioni) esistenti tra le classi del dominio scelto.

Esso è stato ideato allo scopo di essere usato nel contesto del World Wide Web, e tutti i suoi elementi (classi, individui e proprietà) sono definiti come risorse RDF e identificati da URI.

Per il progetto e implementazione dell'ontologia che è oggetto di questo elaborato, si è deciso di utilizzare la versione OWL-DL, in modo da poter usufruire contemporaneamente di buona espressività e di completa decidibilità nella computazione.

Capitolo 1 – La Conoscenza come mezzo per il passaggio da Web a Web
semantico

CAPITOLO 2

Fusione tra sistema Wiki e Tecnologie del Semantic Web: Wiki Semantico

Nella fase di progettazione dell'ontologia di contesto, si è ritenuto opportuno lo svolgimento di una sperimentazione della stessa nell'ambito della tecnologia Wiki, che, per alcune caratteristiche intrinseche che verranno esposte nei paragrafi seguenti, si è rivelato un ambiente aperto e disponibile all'integrazione con funzioni semantiche e metadati. Nei prossimi paragrafi si illustreranno i sistemi Wiki, le loro funzionalità e peculiarità, per poi passare ad introdurre il prodotto della fusione tra Wiki e tecnologie del Semantic Web, ovvero i *Wiki Semantici*.

2.1 Sistemi Wiki tradizionali

Un Wiki può essere visto essenzialmente come una collezione di documenti Web collegati da link ipertestuali. Il primo sito web a cui fu assegnato questo appellativo fu “WikiWikiWeb”, sviluppato da Ward Cunningham, il quale coniò per primo la parola “wiki”, che deriva dall'espressione hawaiana “*wiki wiki*”, il cui significato è “veloce”[WW].

I sistemi Wiki possono essere considerati un esempio di sistemi partecipativi di successo, ossia *social application*, caratterizzate dalla partecipazione di diversi utenti e dalla condivisione di dati e informazioni che si viene a creare fra di essi.

Capitolo 2 – Fusione tra sistema Wiki e Tecnologie del Semantic Web: Wiki Semantico

Non c'è da stupirsi che la tecnologia Wiki abbia riscosso in pochi anni un grande successo presso la comunità, successo dovuto alla facilità di interazione, di recupero e condivisione di dati che essa offre.

Nonostante esista ormai una vasta gamma di sistemi Wiki che differiscono per obiettivi e genere di utenza (i più noti in Internet - Wikipedia³, MediaWiki⁴, MoinMoin⁵, ecc.- ma molti esempi anche all'interno di Intranet aziendali), tutti quanti hanno delle caratteristiche comuni, come illustrato di seguito.

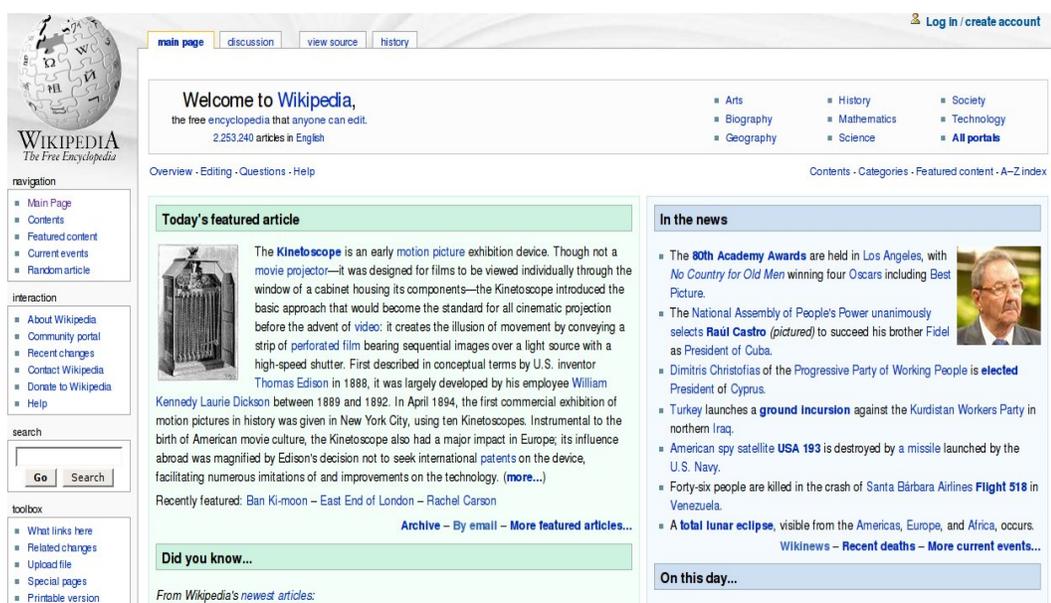


Figura 2.1: Schermata della homepage di Wikipedia.

³ <http://it.wikipedia.org/>

⁴ <http://www.mediawiki.org/>

⁵ <http://moinmoin.wikiweb.de/>

2.1.1 Editing via browser

Si può accedere ai contenuti delle pagine del Wiki tramite una semplice interfaccia browser, senza bisogno di installare software aggiuntivi, e a volte costosi!

Questo approccio consente la modifica delle pagine da qualsiasi luogo l'utente preferisca, e con delle competenze tecniche minime.

2.1.2 Sintassi Wiki semplificata

Le informazioni e i dati sono immessi in un formato semplificato che può variare leggermente da un sistema all'altro (“Wiki Syntax”), ma che comunque si rivela a favore di quegli utenti che non possiedono conoscenze tecniche riguardo a linguaggi di programmazione per il Web, ad es. HTML; per cui all'utente non è richiesta alcuna competenza di questo tipo.

Alcuni sistemi più recenti hanno introdotto tool grafici, quali ad esempio WYSIWYG (What You See Is What You Get), che semplificano ulteriormente la modifica delle pagine, grazie ai quali l'utente può lavorare direttamente con le componenti visuali della pagina.

2.1.3 Forte struttura ipertestuale

Solitamente le pagine appartenenti a un Wiki sono fortemente collegate le une alle altre tramite link. Una delle ragioni è il fatto che la sintassi semplificata dei Wiki permette la creazione di collegamenti fra pagine in modo molto facile e immediato. Per esempio, in molti di questi sistemi un link viene definito tramite la chiusura di una parola tra parentesi quadre, oppure usando il cosiddetto “CamelCase”, in cui ogni lettera iniziale di parole unite è maiuscola (come “CamelCase”).

I link rappresentano lo strumento più importante per la navigazione all'interno del Wiki, e alcuni di questi permettono anche di seguire un link nella direzione opposta rispetto a quella definita; questi ultimi chiamati sono chiamati backlink, o incoming link, e rappresentano i link provenienti da altre pagine web che puntano alla pagina in che si sta visitando .

2.1.4 Accesso libero e meccanismo di rollback

In molti sistemi Wiki non è previsto alcun meccanismo di autenticazione per l'accesso, di conseguenza chiunque può correggere, modificare, completare e perfino cancellare i contenuti delle pagine.

Questo tipo di gestione può sembrare strano, e addirittura pericoloso per la conservazione e attendibilità dei contenuti, ma la pratica ha dimostrato che questo approccio funziona: da una parte, gli utenti con cattive intenzioni sono rari, dall'altra, tutte le modifiche apportate possono essere revocate tramite il meccanismo di *rollback*. Esso consiste nella memorizzazione della versione precedente della pagina ogniqualvolta questa viene aggiornata, in modo da disporre dei contenuti prima della modifica e poterli eventualmente reintrodurre [Paper1].

2.1.5 Editing collaborativi

Tutte le caratteristiche esposte finora, combinate fra loro, fanno del Wiki un sistema ideale per l'editing collaborativo. Quando qualcuno crea una nuova pagina, altri utenti possono contribuire estendendola, correggendola, ecc.. Inoltre, a differenza di altri sistemi per la gestione delle informazioni in un gruppo, i Wiki lasciano agli utenti la completa libertà di scegliere quali contenuti sviluppare, senza strutture rigide o vincoli di altro genere.

L'assenza di restrizioni comporta uno stimolo in più a partecipare, per cui l'utente può:

- avvalersi di una ricerca rapida e semplice per il recupero delle informazioni, grazie alla struttura ipertestuale.
- condividere e migliorare i contenuti.

2.2 I Wiki Semantici

Un “Wiki Semantico” estende un Wiki tradizionale integrandolo con le tecnologie del web Semantico, come RDF, RDFS, OWL, ecc.

Esistono già diversi prototipi di Wiki semantici, come IkeWiki⁶, Semantic MediaWiki⁷, ecc. Alcuni si focalizzano sul contenuto delle pagine, considerando le annotazioni semantiche come un valore aggiunto, altri invece mettono in primo piano le informazioni semantiche della pagine rispetto ai contenuti.

Mentre un Wiki standard è costituito da semplici pagine testuali collegate tra loro mediante link, un Wiki semantico è in grado di generare relazioni tra le pagine interrogando la base di conoscenze che gli viene posta al di sotto.

L’idea principale è di rendere la struttura del Wiki - fortemente “linkata” - processabile da agenti artificiali, interoperabile con altre applicazioni, e ancora più accessibile agli utenti per il reperimento delle informazioni e la creazione di nuove relazioni per inferenza.

2.2.1 Tipizzazione/annotazione dei link e tag

Un primo passo verso un sistema in grado di gestire la semantica dei contenuti potrebbe essere l'utilizzo di *tag*, ovvero etichette che gli utenti possono assegnare ad una pagina, dando vita ad un processo di classificazione dei documenti.

⁶ <http://ikewiki.salzburgresearch.at/>

⁷ http://meta.wikimedia.org/wiki/Semantic_MediaWiki

Un'ulteriore strada è rappresentata dall'annotazione dei link esistenti, per loro natura amorfi, tramite simboli che ne descrivano il significato. Ad esempio: un link tra *Mozart* e *Salzburg* potrebbe essere annotato con “*vissuto in*” o “*nato in*”.

Virtualmente tutti i Wiki semantici permettono di annotare un link assegnandogli un certo tipo. Lo scopo è di dare ai link un significato al di là della sola funzione di navigazione all'interno del sistema.

2.2.2 Ricerca semantica

L'introduzione di una base di conoscenze, sottostante il Wiki, offre agli utenti la possibilità di interrogare la KB. Per esempio, se tutte le pagine che descrivono le opere di Mozart fossero annotate dalla relazione “scritte da Mozart” e questa conoscenza fosse salvata nella KB, allora si potrebbero visualizzare “tutte le opere composte da Mozart” interrogando la base di conoscenze.

Le query, solitamente, sono espresse nel linguaggio SPARQL (di cui si tratterà più avanti), e il Wiki dovrebbe prevedere un'interfaccia semplificata, che renda i passaggi tecnici trasparenti agli utenti, che, come per HTML, potrebbero non possedere le competenze necessarie[Paper1].

2.2.3 Reasoning

Svolgere del reasoning sulle informazioni che si hanno a disposizione vuol dire derivare per inferenza ulteriori informazioni implicite, grazie alle relazioni inserite dall'utente o predefinite nella base di conoscenze.

Per esempio dal fatto che “Mozart” ha composto “Le nozze di Figaro”, un sistema che si avvale del reasoning potrebbe dedurre che “Mozart” è un “Compositore”!

È evidente che il reasoning rappresenta uno strumento indispensabile per la realizzazione di un sistema come un Wiki Semantico; rinunciare a dedurre informazioni per inferenza significherebbe perdere la possibilità di accedere a conoscenze non dichiarate esplicitamente.

2.2.4 Interfaccia utente

Molti Wiki Semantici presentano un'interfaccia utente che dà la possibilità di accedere ai contenuti della base di conoscenze, e per questo leggermente diversa dall'interfaccia di un Wiki tradizionale.

A titolo di esempio, in un sistema di questo tipo ogni pagina può essere arricchita da un blocco a parte dove compaiono le pagine collegate a quella corrente da relazioni semantiche.

Da quanto detto sopra, la navigazione, supportata dalla descrizione del link che si segue, risulta più sofisticata e completa rispetto a quella offerta da sistemi standard.

2.2.5 Limiti e problematiche

A valle di tutto ciò che è stato esposto è necessario tenere presente che il Web Semantico risulta ancora relativamente nuovo per il grande pubblico, per cui anche la tecnologia dei Wiki Semantici è ancora agli albori nell'immaginario collettivo. La recente introduzione della semantica nel panorama dei wiki porta con sé problematiche di varia natura non ancora del tutto risolte. Una di queste problematiche è certamente il proliferare all'interno del testo di link che rischiano di diventare ingestibili da un punto di vista numerico e da un punto di vista qualitativo. Basti pensare al sistema Cyc⁸, che ha oltre 15.000 diversi tipi di link!

⁸ Cyc è un progetto di intelligenza artificiale (IA) che tenta di assemblare una vasta ontologia ed una base di dati contenente le conoscenze del senso comune quotidiano, con lo scopo di creare applicazioni di IA somiglianti al ragionamento umano.

Inoltre la rappresentazione delle azioni, al contrario di quella della conoscenza, è sempre stata relativamente difficile; certi tipi di relazioni (acts-on, carries-out...) potrebbero essere definiti ma sono parecchi gli inconvenienti che questo causerebbe, dal momento che si tratta di relazioni non oggettive.

Nonostante il valore aggiunto che la semantica apporta alla tecnologia Wiki, ad oggi sono stati sviluppati solo un numero esiguo di Wiki Semantici (tra cui ricordiamo Semantic MediaWiki e IkeWiki), proprio a causa dei limiti ancora da superare[WWS].

2.2.6 I livelli di ontologie in un Wiki Semantico

Le ontologie che si prestano a fare da base ad un Semantic Wiki, si rifanno a tre livelli di semantica:

- *ambiente*: un'ontologia sui Wiki - questa ontologia ha senso se interpretata insieme all'ontologia di contesto, che cambia al cambiare del contesto;
- *contesto*: un'ontologia sui processi e sugli attori dell'ambiente editoriale - rappresenta un'ontologia che descrive il contesto in cui ci si muove, in questo caso i documenti del Wiki, rimanendo svincolata dai contenuti di tali documenti;
- *contenuto*: un'ontologia sui particolari contenuti del prodotto editoriale - questo tipo di ontologia viene sviluppata una volta attivo il Wiki, poiché strettamente legata ai contenuti che lo andranno a popolare. Essa può ricevere contributi anche dagli utenti, nel caso in cui essi possano “taggare” le pagine, favorendone la categorizzazione.

Alla luce di queste definizioni, dovrebbero essere più chiare le caratteristiche del tipo di ontologia sviluppata per questo lavoro di tesi, ossia un'ontologia di contesto

Capitolo 2 – Fusione tra sistema Wiki e Tecnologie del Semantic Web: Wiki Semantico

in cui si vogliono definire le relazioni che intercorrono tra le pagine di un Wiki Semantico, dipendenti dalla natura di queste ultime in quanto *documenti*, ma indipendenti dal loro contenuto.

CAPITOLO 3

Ambiente di sviluppo e strumenti utilizzati

Dopo una prima fase di progettazione dell'ontologia, in cui è stata definita la struttura concettuale della base di conoscenze, si è passati alla realizzazione della stessa tramite delle applicazioni e tool specifici, per poi passare all'implementazione nel sistema Wiki.

In questo capitolo verranno presentati tutti gli strumenti software di cui si è fatto uso per il lavoro di tesi.

3.1 Editor di ontologie: Protégé

Protégé è una piattaforma basata sul linguaggio di programmazione Java, è open-source, e offre un ambiente di sviluppo per il progetto di applicazioni e prototipi basati sulla gestione di conoscenze tramite ontologie.

Al suo interno Protégé implementa tutto un insieme di strutture per la modellazione delle conoscenze, e operazioni per la creazione, visualizzazione e manipolazione di ontologie, che possono essere esportate in vari formati, inclusi RDFS, OWL, e XML-Schema[Pr].

Esistono due strumenti principali supportati da Protégé per la creazione e gestione dei modelli:

- l'editor Protégé-Frames;
- l'editor Protégé-OWL.

Quest'ultimo offre agli utenti la possibilità di creare ontologie per il Web Semantico che prevedono l'uso di OWL.

Per dare un'idea di come è possibile usare questo strumento per la creazione di un modello, si riportano ora le operazioni più comuni che è possibile svolgere con Protégé.

Il primo passo nel progetto di un'ontologia è l'identificazione dei concetti coinvolti nel frammento di realtà che si vuole modellare; in Protégé esiste un pannello delle classi in cui è possibile:

- creare dei nuovi concetti, oltre quelli predefiniti owl:Thing e owl:Nothing, a cui si assegna un nome;
- creare classi anonime, derivanti da descrizioni ed espressioni;
- operare l'intersezione, l'unione e il complemento di classi;
- applicare restrizioni alle estensioni di una classe.

L'ultima operazione implica la possibilità di descrivere gli individui di una classe determinandone il tipo e le relazioni a cui essi possono partecipare. In particolare le restrizioni possono essere espresse da:

- quantificazioni esistenziali e universali;
- cardinalità (max, min, equal).

Capitolo 3 – Ambiente di sviluppo e strumenti utilizzati

Nello stesso pannello si trovano le condizioni a cui una classe è soggetta, espresse tramite le restrizioni, e che possono essere “necessarie” oppure “necessarie e sufficienti”; esse devono essere soddisfatte da tutti gli individui della classe selezionata. Al di sotto delle condizioni si trova la sezione delle disgiunzioni (“Disjoints”), in cui possono essere inserite tutte le classi che devono essere disgiunte da quella selezionata. La figura 3.1 mostra un esempio di pannello delle classi.

Un altro pannello dell’applicazione permette la definizione di proprietà, insieme alle caratteristiche che tali relazioni possono possedere, ad esempio la transitività, la simmetria, l’essere inversa di un’altra relazione o funzionale.

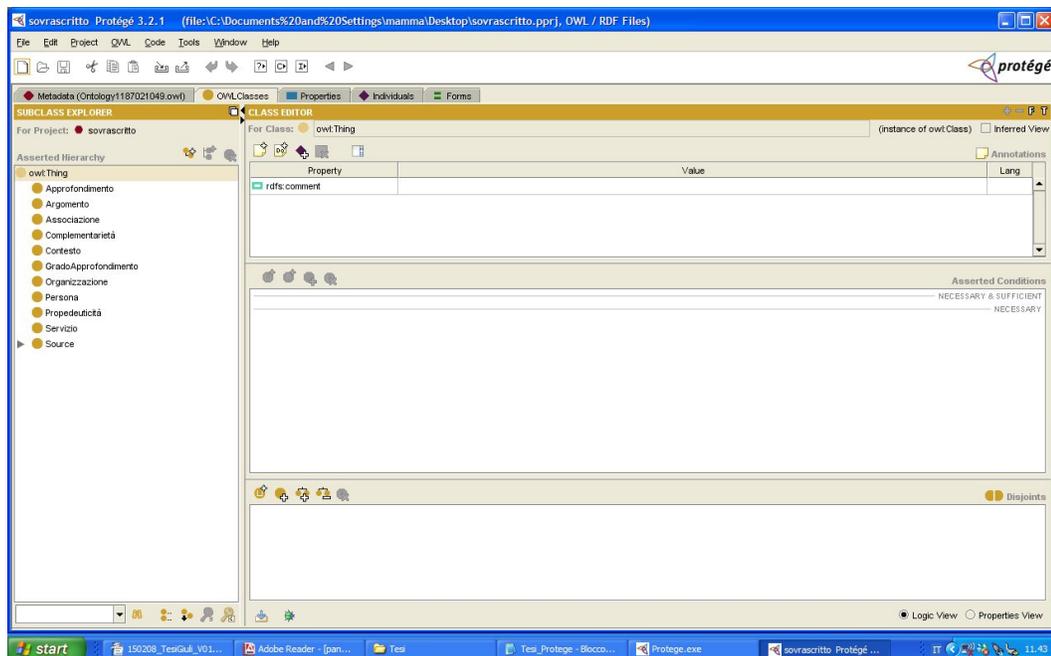


Figura 3.1: Pannello delle classi di Protégé .

Come in OWL, anche in Protégé le proprietà vengono distinte in:

Capitolo 3 – Ambiente di sviluppo e strumenti utilizzati

- ObjectProperty, che mettono in relazione due individui;
- DatatypeProperty, associano un individuo ad un valore di un dominio concreto (string, integer, ecc.).

Protégé supporta anche un terzo tipo di proprietà, “Annotation properties”, usate per associare metadati a classi, proprietà e istanze.

E proprio per le istanze esiste un ulteriore pannello dell'applicazione, del tutto simile agli altri due, usato per creare, manipolare e gestire gli individui.

Infine ricordiamo che Protégé dà la possibilità di importare ontologie già esistenti e adattare alle proprie esigenze.

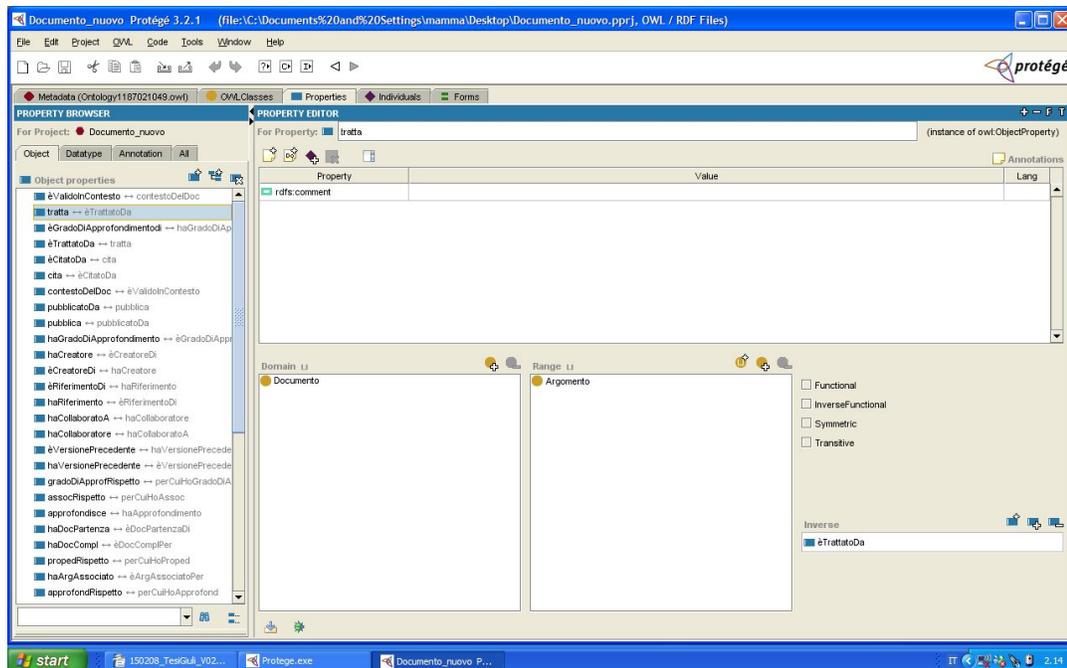


Figura 3.2: Schermata del pannello delle proprietà di Protégé.

3.2 Graphic OWL (GrOWL)

Come si può intuire dal nome stesso, GrOWL⁹ è un'applicazione che permette all'utente la creazione di ontologie a livello grafico.

È stato realizzato dall'ing. Mario Arrigoni Neri presso il Politecnico di Milano, è implementato nell'ambiente di programmazione Java e fa uso della libreria Jena per la gestione dell'ontologia.

GrOWL rappresenta uno strumento alternativo per lo sviluppo di ontologie, che tipicamente vengono sviluppate tramite tool testuali come Protégé, e oltre all'originalità ha il pregio di semplificare la visualizzazione completa dei concetti, degli individui e delle relazioni che li legano, grazie a simboli grafici appositamente studiati per farne intuire il significato. I simboli posti sugli archi delle relazioni, per esempio, lasciano intuire la semantica della relazione tra due individui.

Tutte queste caratteristiche possono essere visualizzate nella figura 3.3, e altre figure simili saranno poi utilizzate nella parte di descrizione dell'ontologia.

⁹ <http://home.dei.polimi.it/arrigoni/GrOWL/>

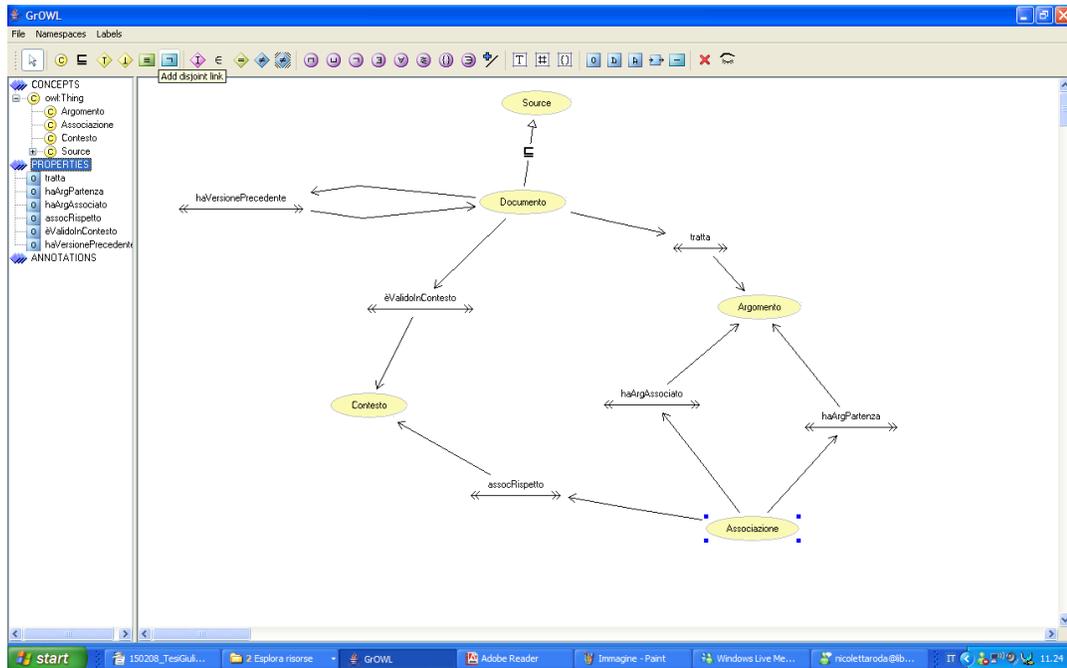


Figura 3.3: Schermata di un'ontologia in GrOWL.

3.3 Jena

Lo sviluppo di applicazioni del Semantic Web ha portato all'implementazione di una serie di tool e librerie per la manipolazione di metadati. Uno di questi tool è appunto Jena, una piattaforma Java, open-source, che permette di manipolare i dati del Web semantico, rappresentati tramite gli standard del W3C, da RDF ad OWL e a SPARQL (illustrato nel paragrafo seguente).

Il corpo centrale di Jena provvede alla gestione di modelli costituiti da triple RDF, tramite classi quali “Model”, “Resource”, ecc., estese poi dalle classi dedite alla manipolazione più specifica di ontologie, tramite i metodi e funzioni offerti dalle classi “OntModel”, “Ontproperty”, e così via. Inoltre implementa anche i

metodi necessari per la gestione di più modelli contemporaneamente, la loro unione e l'importazione.

Un'altra caratteristica è che Jena supporta due metodi per l'applicazione di algoritmi di reasoning alle basi di conoscenze: o tramite un reasoner interno alla libreria stessa, o tramite il collegamento ad un reasoner esterno scelto dall'utente.

3.4 Linguaggio di interrogazione SPARQL

Allo scopo di sviluppare un linguaggio che potesse prelevare informazioni direttamente o indirettamente derivabili da una base di conoscenze, sono stati seguiti diversi approcci. Tra questi, quello che si basa sulla sintassi SQL ad oggi sembra aver avuto maggior successo e una maggiore diffusione. Infatti, nonostante la differenza che intercorre tra una base di dati ed una di conoscenze, la familiarità della sintassi ha sicuramente contribuito a questo successo[Sp].

Il linguaggio SPARQL segue questo filone ed è diventato una raccomandazione del World Wide Web Consortium come linguaggio di interrogazione per il Web Semantico.

Esso si focalizza sull'interrogazione di grafi RDF a livello di triple e viene usato anche con modelli RDF Schema e OWL, ad esempio per recuperare gli individui che presentano determinate caratteristiche.

Le query SPARQL si basano sul meccanismo di “pattern matching” e in particolare su un costrutto, il “triple pattern”, che ricalca la configurazione a triple delle asserzioni RDF fornendo un modello flessibile per la ricerca di corrispondenze.

La figura 3.4 mostra un semplice esempio di query SPARQL in cui le variabili sono prefisse dal simbolo '?' e i campi da restituire sono quelli che seguono il comando `SELECT` (analogamente a SQL).

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?subclass
WHERE {?subclass rdfs:subClassOf ?class .
       ?class rdfs:label 'Source'          }
```

Figura 3.4: Esempio di query Sparql che restituisce tutte le sottoclassi della classe Source.

L'assegnamento di prefissi ai namespace, come

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
```

tende a rendere più leggibile la query, sostituendo, nel corpo della query, il prefisso all'intero namespace.

É da notare che, affinché siano sintatticamente corrette e risolvibili, le query non possono contenere una variabile al posto del predicato.

Da quanto detto SPARQL dà un supporto aggiuntivo, rispetto ai soli metodi messi a disposizione da Jena, per la derivazione di conoscenze da una KB; per questo motivo è risultata utile l'implementazione di alcune query di questo tipo per il lavoro di tesi.

3.5 Pellet: un reasoner per OWL-DL

Pellet è un reasoner completo per la gestione di ontologie scritte in OWL-DL, con buone performance e con qualche caratteristica particolare rispetto ad altri reasoner ad oggi sviluppati (quali Racer, FaCT++...).

É implementato nel linguaggio di programmazione Java ed è un software open source utilizzabile sotto licenza libera, il che rappresenta uno dei fattori che ne

determina la vasta diffusione e uso in un gran numero di progetti, dalla ricerca a svariati sistemi industriali.

Pellet è la prima implementazione di procedure completamente decidibili per OWL-DL (includendo le istanze) ed ha un supporto esteso per il reasoning sugli individui (tra cui le query congiuntive sulle asserzioni), sui tipi di dati definiti dagli utenti, e per il debugging di ontologie.

Esso offre tutte le funzionalità che sono considerate minime e indispensabili per assolvere al compito di reasoner, e che provvedono a svolgere i servizi “standard” di inferenza delle Logiche Descrittive (DL), cioè:

- *controllo di consistenza*, che assicura l'assenza di asserzioni contraddittorie nella Knowledge Base. Nella terminologia delle DL, è l'operazione che controlla la consistenza dell'Abox¹⁰ rispetto alla Tbox¹¹;
- *soddisfacibilità dei concetti*, che controlla se possono esistere istanze di quella classe. Se una classe risulta insoddisfacibile, la definizione di un individuo appartenente all'estensione di quella classe rende l'intera ontologia inconsistente;
- *classificazione*, che costruisce una gerarchia completa dei concetti basandosi sulla relazione di sottoclasse (*rdfs:subClassOf*) esistente tra le classi; questa operazione è molto utile qualora si debba rispondere a query che richiedono un'analisi dell'albero gerarchico dell'ontologia (ad esempio per trovare le sottoclassi di una determinata classe);
- *matching* tra un individuo e il tipo a cui appartiene.

¹⁰ Assertion Box: componente della KB che contiene le asserzioni sugli individui, ad esempio attributi OWL come il tipo, i valori delle proprietà, l'equivalenza, o meno, di asserzioni.

¹¹ Terminological Box: componente che contiene gli assiomi riguardo le classi, ad esempio assiomi per definire le sottoclassi, l'equivalenza tra classi, o la disgiunzione tra classi.

Oltre a queste funzionalità di base Pellet offre la possibilità di risolvere query riguardo gli individui dell'ontologia, scritte in SPARQL, o anche RDQL¹². In una KB complessa e/o di elevate dimensioni può diventare oneroso e difficile anche per gli esperti trovare contraddizioni tra le definizioni dell'ontologia. A differenza di altri reasoner che si limitano a segnalare l'errore senza alcun supporto nella diagnosi e risoluzione del problema, Pellet implementa due servizi di debugging al fine di intercettare e spiegare la presenza nella KB di eventuali inconsistenze.

Le caratteristiche appena esposte, insieme al fatto che Pellet può essere considerato competitivo anche a livello di performance per applicazioni reali, fanno di questo reasoner uno strumento affidabile e completo, motivo per il quale è stato scelto per realizzare questo lavoro di tesi.

Pellet ricopre una funzione molto importante per applicazione che è stata sviluppata, infatti esso rappresenta lo strumento tramite cui è possibile derivare per inferenza e deduzione nuove conoscenze, al di là di quelle esplicitamente asserite nell'ontologia di contesto.

3.6 JSPWiki

Negli ultimi anni il Web ha visto il proliferare di numero cospicuo di sistemi Wiki, dedicati alla gestione dell'informazione negli ambiti più disparati.

Per il presente lavoro di tesi la scelta del sistema in cui implementare e sperimentare l'ontologia di contesto è ricaduta sul sistema JSPWiki.

Il motivo di tale scelta sta nel fatto che JSPWiki è diventato nel tempo un software stabile e diffuso (ha trovato infatti largo impiego nelle intranet aziendali e in istituti di prestigio), ma, ragione ancor più importante, offre la possibilità di

¹²Un altro linguaggio di query per RDF promosso dal W3C.

essere esteso in modo semplice; tra l'altro è stato già adottato per creare Wiki semantici, tra cui ricordiamo Makna¹³.

3.6.1 Nascita e diffusione

JSPWiki è un software wiki scritto in Java secondo lo standard J2EE, e come dice il nome stesso, è realizzato con tecnologia JSP (Java Server Package).

JSPWiki è stato scritto da Janne Jalakanen e rilasciato sotto la licenza libera LGPL. I server Sun Microsystems includono JSPWiki come una delle loro core application.

Grazie alla sua facilità di installazione, molte persone hanno scelto JSPWiki anche per uso privato, per organizzare le proprie informazioni personali. JSPWiki può anche essere impiegato per creare blog, grazie al plug-in integrato ed al supporto per RSS.

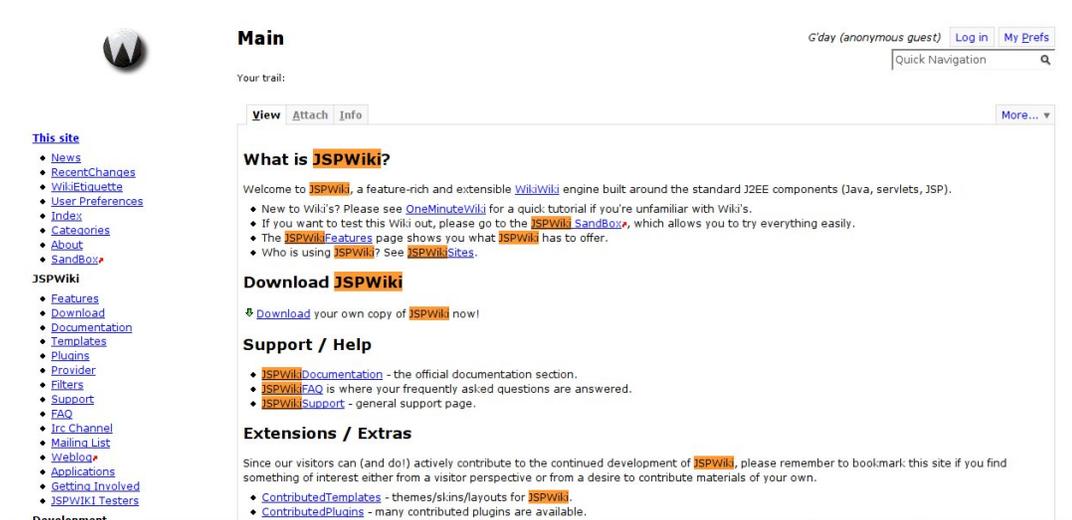


Figura 3.5: Homepage di JSPWiki.

¹³ <http://www.apps.ag-nbi.de/makna/>

3.6.2 Peculiarità di JSPWiki

JSPWiki integra ed estende tutte le funzioni di PHP Wiki¹⁴. Una delle caratteristiche che ha visto JSPWiki vincere il confronto con altri motori wiki, come ad esempio Mediawiki e TWiki¹⁵ è stata la sua sintassi utente estremamente intuitiva. Il linguaggio di mark-up può essere considerato semplice anche per gli utenti più inesperti: ad esempio i link vengono creati semplicemente racchiudendo il nome del link tra parentesi quadre, e ancora le intestazioni si creano ponendo il punto esclamativo in capo alla riga dell'header stesso.

A partire dalla versione 2.3, JSPWiki permette di integrare l'uso di CSS usando i tags `%%`. Per esempio un testo può essere visualizzato in rosso usando il seguente markup: `%% (color:red) RED TEXT %%`. JSPWiki permette inoltre di allegare alle proprie pagine[WJSP].

Tra le altre caratteristiche di JSPWiki ricordiamo:

- *internazionalità*: JSPWiki supporta il sistema unicode UTF-8 come sistema standard di codifica, permettendo quindi l'utilizzo di lingue come il Thai e l'Ebraico;
- *plug-ins*: JSPWiki fornisce una semplice interfaccia per l'integrazione di plug-in, così da permettere agli utenti più esigenti di estendere le funzionalità del Wiki usando Java. Molti plug-in sono disponibili gratuitamente; persino la funzione dei cambiamenti recenti è un plug-in;
- *interfaccia*: JSPWiki permette di personalizzare la grafica e l'interfaccia utente del wiki usando template; numerosi template sviluppati dagli utenti sono disponibili gratuitamente per il download nel Web, a dimostrazione della notevole diffusione di questo motore wiki; tra questi uno ricalca la grafica di Wikipedia. Offre inoltre la possibilità di creare e modificare un

¹⁴ <http://phpwiki.sourceforge.net/>

¹⁵ <http://twiki.org/>

LeftMenu, in cui l'utente può decidere di immettere i collegamenti alle pagine che preferisce;

- *gestione dei file*: i file possono essere allegati direttamente alle pagine del Wiki;
- *WikiWizard*: sono disponibili inoltre due modalità con cui procedere alla modifica o creazione delle pagine, o tramite un “plain editor” o tramite un “WikiWizard editor”, che permette all'utente di definire le caratteristiche del testo in modo visuale, tramite un apposito set di controlli.

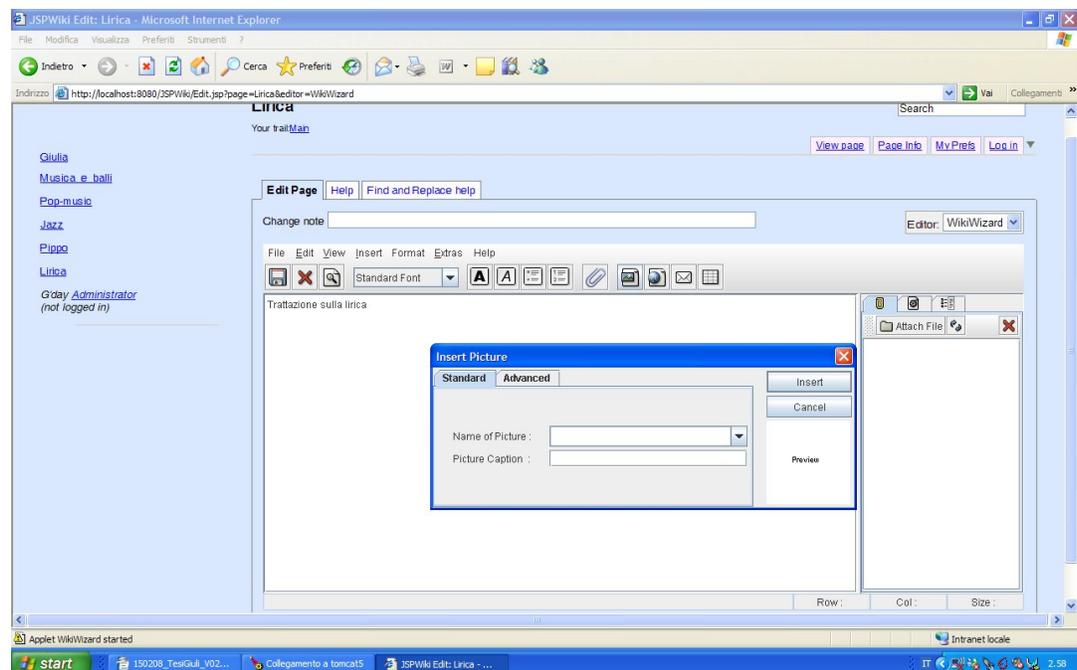


Figura 3.6: schermata dell'editor WikiWizard di JSPWiki.

CAPITOLO 4

Ontologia di contesto

L'ontologia che mi sono proposta di sviluppare rientra nella classe delle ontologie mediane (middle ontology), concernenti un'area conoscitiva ben definita (come ad esempio la chimica organica, il commercio elettronico, la telefonia mobile, i personal computer, gli strumenti musicali e così via), quindi domain dependent, ma non ancora legati a una specifica applicazione e quindi application independent.

Essa tratta la classificazione dei documenti digitali, visti come strumenti per l'apprendimento, la comprensione e l'approfondimento di argomenti.

4.1 I concetti principali

Partendo quindi dalla funzionalità didattica che si vuole dare al documento, sono stati fissati come primi concetti dell'ontologia i seguenti:

- SOURCE: questa classe rappresenta una qualsiasi sorgente di informazione (testi a stampa e elettronici, eventi culturali, ecc.) da cui può essere tratto un documento;
- DOCUMENTO: con la classe “DOCUMENTO” si è voluto indicare un oggetto pubblicato sul web che sia fonte di informazione relativamente ad

uno o più argomenti. Per la definizione appena data, esso risulta essere sottoclasse del concetto SOURCE.

Tra le sorgenti d'informazione troviamo quindi il concetto centrale dell'ontologia, cioè il documento, che è legato ai restanti concetti dell'ontologia tramite relazioni di cui esso stesso è dominio.

Prima di dare una descrizione dei ruoli è utile introdurre alcuni concetti che sono necessari per dare una rappresentazione più completa possibile di un documento:

- ARGOMENTO è la classe che identifica l'oggetto di cui tratta il documento;
- CONTESTO è l'ambiente globale in cui si inserisce l'argomento associato ad un documento e in cui tale argomento ha senso;
- PERSONA, ORGANIZZAZIONE e SERVIZIO sono invece utili per dare informazioni circa la creazione ed elaborazione del documento.

A questo punto le relazioni che si creano spontaneamente tra questi concetti sono:

- *tratta*: associa ad ogni documento almeno un argomento;
- *èValidoInContesto*: mette in relazione ogni documento con i contesti in cui risulta valido;
- *cita*: crea un legame tra due documenti (in questo caso DOCUMENTO è anche codominio della relazione), per cui nel documento, che è soggetto della relazione, appare una citazione del documento che invece è oggetto.

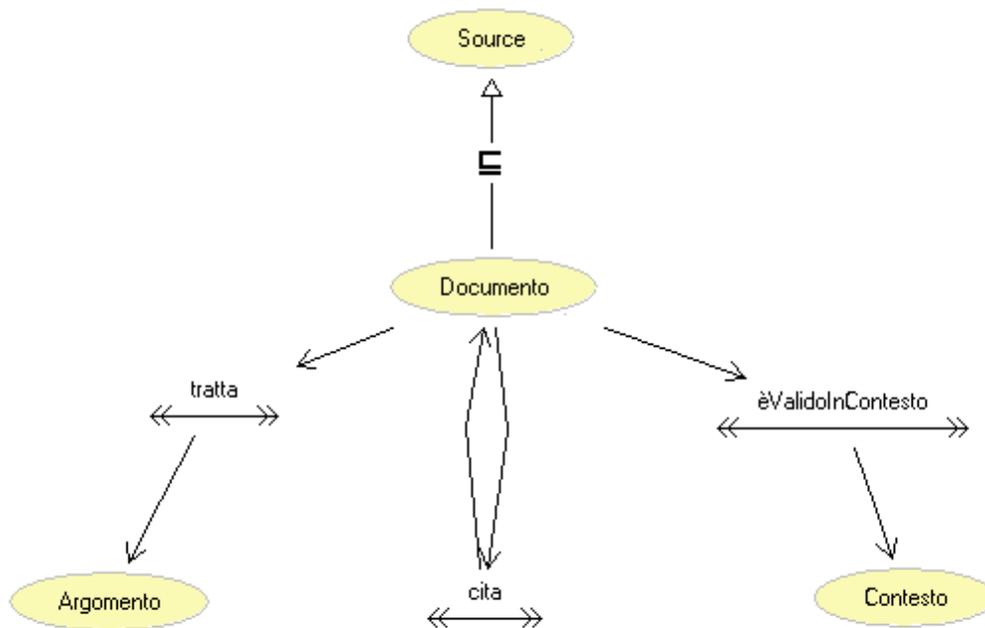


Figura 4.1: Rappresentazione dei concetti principali e delle relazioni che li legano in GrOWL.

4.2 Documenti associati e complementari

4.2.1 La reificazione

Si è ritenuto rilevante esprimere il fatto che un argomento possa essere associato ad un altro argomento rispetto ad un certo contesto; così espressa, la relazione tra gli argomenti risulta essere ternaria, quindi non esprimibile tramite OWL DL, se non applicando il metodo di reificazione, secondo il quale è possibile rappresentare come individuo.

4.2.2 Associazione e Complementarietà

Nel caso considerato ho reificato la relazione di associazione, sostituendola con il concetto ASSOCIAZIONE: ogni individuo nell'estensione di questo concetto può avere un solo argomento di partenza, un solo argomento associato al primo (o di “arrivo”) e un solo contesto per cui vale tale associazione. Tutto ciò nell'ontologia si traduce nelle seguenti relazioni:

- *haArgPartenza*: ASSOCIAZIONE → ARGOMENTO;
- *haArgAssociato*: ASSOCIAZIONE → ARGOMENTO;
- *assocRispetto*: ASSOCIAZIONE → CONTESTO.

Un'altra relazione utile è quella per cui un documento è complementare ad un altro (cioè completa l'informazione portata dal primo documento) rispetto ad uno specifico contesto; è evidente che ci troviamo nella situazione precedente di relazione ternaria, per cui anche in questo caso è stato necessario reificare la relazione ottenendo un nuovo concetto, quello di COMPLEMENTARIETA', che è legato ad uno e un solo documento di partenza, ad uno e un solo documento complementare, e ad un unico contesto tramite le relazioni:

- *haDocPartenza*: COMPLEMENTARIETA' → DOCUMENTO;
- *haDocCompl*: COMPLEMENTARIETA' → DOCUMENTO;
- *complRispetto*: COMPLEMENTARIETA' → CONTESTO.

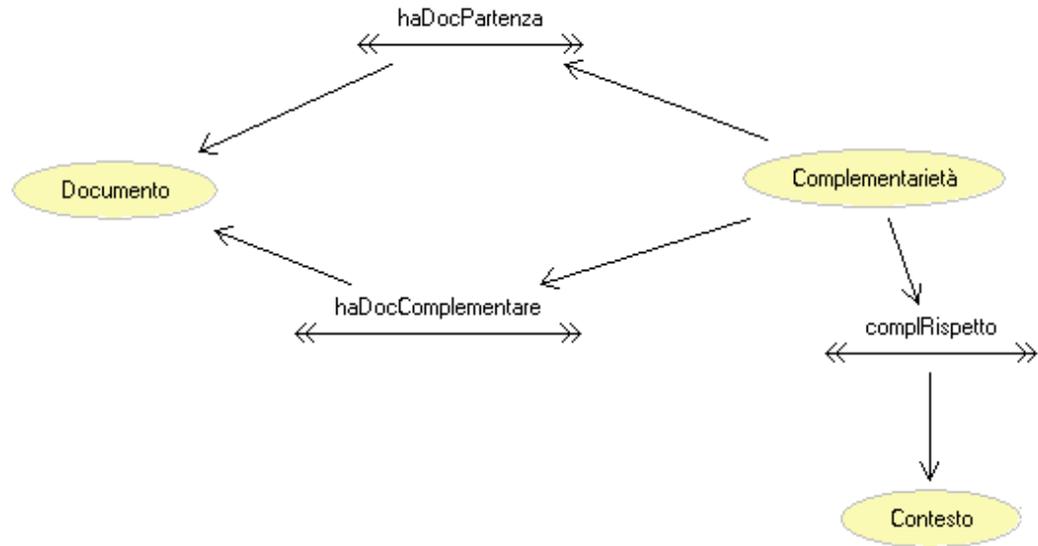


Figura 4.2: Rappresentazione del concetto reificato di Complementarietà e relazioni principali in GrOWL.

4.2.3 Propedeuticità e Approfondimento

Lo sviluppo progressivo della conoscenza di un determinato argomento si inserisce nell'ontologia tramite altre due percorsi caratteristici dell'apprendimento, cioè l'approfondimento dell'argomento e lo studio di conoscenze preliminari per la comprensione del documento in oggetto.

In entrambi i casi ci troviamo di fronte all'associazione di due documenti che risulta valida rispetto ad uno specifico argomento. Infatti, tenendo presente che ogni documento può parlare di più argomenti, non è immediata la relazione che intercorre fra i due documenti ed è necessario specificare l'argomento rispetto al quale tale relazione è valida. Anche in questo caso ritorna utile la reificazione della relazione, che porta all'introduzione di altri due concetti:

- il concetto di APPROFONDIMENTO che rappresenta l'insieme di informazioni aggiuntive contenute in un altro documento, sempre in relazione all'argomento trattato nel documento di partenza;
- il concetto di PROPEDEUTICITA', che permette di reperire, tramite altri documenti, le informazioni preliminari alla comprensione di un determinato argomento di cui tratta il documento in oggetto.

Entrambi i concetti sono legati alle classi DOCUMENTO e ARGOMENTO tramite relazioni del tutto simili a quelle definite per i concetti precedenti.

4.3 Relazioni per la classificazione di documenti

Altre proprietà sono quelle legate alla creazione del documento come:

- *haCreatore*, che associa ad ogni documento una PERSONA, oppure un'ORGANIZZAZIONE oppure un SERVIZIO che sia l'autore del documento;
- *haCollaboratore*, che ha stesso dominio e codominio della relazione precedente, e individua le entità che hanno partecipato alla stesura del documento;
- *pubblicatoDa*, che lega un documento all'entità responsabile della sua pubblicazione (dominio e codominio sono ancora uguali a quelli della relazione precedente);
- *haVersionePrecedente*: lega un documento ad un altro ne che rappresenta la versione precedente; ho definito questa relazione transitiva in modo da riuscire a risalire anche a documenti non immediatamente precedenti.

- *haRiferimento*: esprime il legame tra un certo documento e le fonti da cui è stato tratto, che rientrano nella classe SOURCE.

Infine ho introdotto alcuni ruoli che si distinguono dai precedenti in quanto associano ad un concetto valori costanti, ad esempio numeri interi (con o senza il segno), numeri floating point, caratteri, stringhe, ecc.; gli insiemi di valori di questo genere vengono chiamati *domini concreti* o *domini di dati*.

Queste relazioni associano ad un documento:

- la data di creazione , di tipo *Date* (*haData*);
- un grado di approfondimento per ogni argomento trattato, che può assumere uno dei i tre valori predefiniti: “*low*”, “*medium*” oppure “*high*”; come si può notare, anche in questo caso non è possibile assegnare un valore univoco di approfondimento ad un documento, poiché argomenti diversi dello stesso documento possono essere approfonditi in modo non uniforme. Anche in questo caso si è ricorsi alla creazione di un nuovo concetto (GRADO_APPROFONDIMENTO) e alle relative relazioni, come risultato della reificazione:
 - *gradoDiApprofRispetto*:
GRADO_APPROFONDIMENTO → ARGOMENTO
 - *haValore*: GRADO_APPROFONDIMENTO → STRING
 - *èGradoDiApprofondimentodi*:
GRADO_APPROFONDIMENTO →DOCUMENTO

4.4 Completamento del modello

Per ogni relazione descritta sopra è stato ritenuto utile introdurre la rispettiva proprietà inversa, che non viene riportata in questa trattazione per evitare di appesantire la descrizione delle, già numerose, relazioni, e che comunque risultano abbastanza semplici e intuitive.

Bisogna infine sottolineare che, per ottenere risultati corretti a livello di reasoning sull'ontologia, ogni classe è stata disgiunta dalle classi di pari livello nell'albero gerarchico. Allo stesso modo per gli individui è stata fatta la Unique Name Assumption (UNA), per cui due istanze con nomi diversi rappresentano individui diversi.

4.5 Esempio di ontologia

Per dare un'idea del funzionamento del modello, si propone un piccolo esempio di ontologia con alcuni individui e alcune possibili interrogazioni alla base di conoscenza.

Possiamo ipotizzare di creare un individuo della classe DOCUMENTO che chiamiamo “Jazz”, e con il quale indichiamo un documento che parla di questo genere musicale. Riguardo a tale individuo possiamo asserire che è associato all'individuo “musica”, di tipo ARGOMENTO, tramite la relazione “tratta”.

Alla stessa maniera possiamo definire altri individui della classe DOCUMENTO che trattano lo stesso argomento, ad esempio “Storia della musica Jazz”, che è propedeutico del documento “Jazz” rispetto all'argomento “musica”, oppure “Grandi esponenti della musica Jazz”, anch'esso associato al primo documento.

Per una persona che volesse documentarsi sul genere musicale Jazz, e supponendo di aver aggiunto all'ontologia tanti individui come quelli appena visti,

Capitolo 4 – Ontologia di contesto

si potrebbe interrogare la KB, con la seguente query Sparql, chiedendo di restituire tutti i documenti che sono propedeutici al documento “Jazz”:

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX doc:<http://www.owlontologies.com/Ontology1187021049.owl#>
SELECT ?doc
WHERE {doc:Jazz doc:èIntrodotta ?x.
       ?x doc:propedeuticaIn ?doc.
       ?x doc:propedeuticaIn doc:musica }
```

oppure tramite un'altra query, come quella mostrata sotto, si potrebbero ottenere tutti i documenti associati.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX doc:<http://www.owlontologies.com/Ontology1187021049.owl#>
SELECT ?doc
WHERE {doc:Jazz doc:tratta ?x.
       ?x doc:haAssociazione ?y.
       ?y doc:haArgomentoAssociato ?z
       ?z doc:èTrattatoDa ?doc }
```

Si noti che, se il documento “Jazz” tratta più di un argomento, questa query restituisce tutti i documenti associati rispetto ad ognuno degli argomenti trattati da “Jazz”, e non solo rispetto all'argomento “musica”.

CAPITOLO 5

Implementazione in JSPWiki

Dopo la creazione dell'ontologia si è passati alla fase di implementazione all'interno di JSPWiki, grazie alla quale il sistema può accedere ai contenuti della KB descritta nel capitolo precedente. Il primo passo in questa direzione è stato la creazione di una classe Java che, sfruttando gli strumenti e i metodi messi a disposizione dalla libreria Jena, fosse in grado di manipolare, prelevare e aggiungere informazioni nel file OWL dell'ontologia.

Successivamente si è passati all'adattamento dell'interfaccia delle pagine del Wiki, affinché l'utente potesse visualizzare le informazioni semantiche presenti all'interno della KB, e ne potesse aggiungere delle nuove.

Il sistema complessivo funziona nel modo seguente: l'ontologia è gestita direttamente dalla classe Java `Ontology`; essa è inserita all'interno del motore JSPWiki che crea un oggetto di tipo `Ontology`.

Tramite questo oggetto ogni pagina del Wiki accede alla KB e mostra all'utente i risultati delle operazioni effettuate.

La figura 5.1 dovrebbe chiarire il meccanismo.

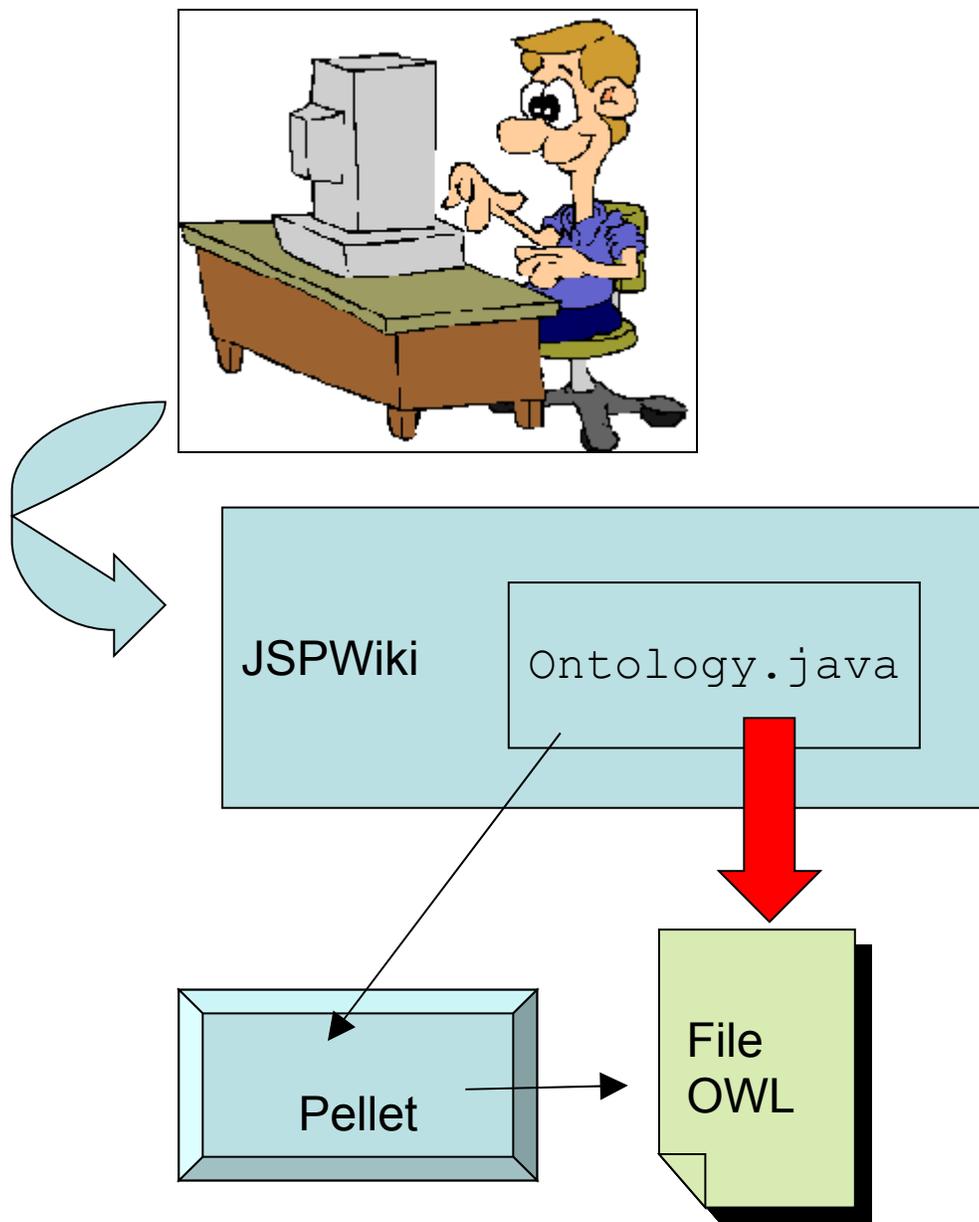


Figura 5.1: Struttura del sistema.

I prossimi paragrafi illustrano più in dettaglio i vari componenti.

5.1 Classe `Ontology.java`

La classe `Ontology.java` può essere considerata il *core* del sistema di gestione dell'ontologia; infatti essa fornisce i metodi richiamati all'interno del motore JSPWiki che si preoccupano del reperimento e manipolazione delle informazioni semantiche.

L'oggetto `Ontology` viene creato al momento dell'inizializzazione dell'oggetto `WikiEngine`; quest'ultimo, che rappresenta l'istanza della classe principale del Wiki, viene creato una sola volta nella sessione di accesso al Wiki, e ciò garantisce che anche l'oggetto che gestisce l'ontologia sia unico per tutta la sessione di lavoro.

`Ontology()` è il costruttore della classe omonima, che provvede a creare il modello dell'ontologia e il collegamento con il reasoner esterno.

Il modello, di tipo `OntModel`, indispensabile per l'accesso ai componenti della KB, viene restituito da un apposito metodo: `getModello()`.

5.1.1 I metodi principali

I metodi principali definiti in questa classe sono tre:

- `infoGenerali(String docName)`, è il metodo con il quale vengono prelevate dalla KB le informazioni generali associate alla pagina del Wiki il cui nome viene passato come parametro.

Queste informazioni, ad esempio gli argomenti trattati nella pagina, o il contesto in cui quei contenuti risultano validi, costituiscono in pratica una classificazione del documento.

- `pagineAssociate(String docName)`, è il metodo che restituisce tutti i documenti nel Wiki e le relazioni per cui vi è un'associazione con la

pagina visualizzata. Anche per questo metodo l'unico parametro è il nome della pagina;

- `aggiungiRelazione(String docName, String relazione, String obj)`.

Questo metodo permette all'utente di associare la pagina corrente `docName` al documento `obj` tramite la proprietà `relazione`.

5.1.2 Esecuzione delle query

I metodi descritti sopra, per restituire i risultati desiderati, devono necessariamente interrogare la KB, andando a prelevare le informazioni asserite o derivate sulla pagina selezionata. Per fare ciò si avvalgono di altri due metodi definiti all'interno della classe `Ontology`:

- il primo si preoccupa di reperire i risultati inerenti a query eseguite su relazioni binarie (come *tratta*, *èValidoInContesto*, ecc.)
- il secondo è un po' più complicato, poiché ha il compito di restituire i risultati per interrogazioni che coinvolgono i concetti reificati. Di seguito è riportato un esempio di query che restituisce i documenti che approfondiscono la pagina `docName` e l'argomento rispetto al quale viene approfondito. La variabile `?x` rappresenta l'individuo di tipo *Approfondimento*.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX doc:<http://www.owlontologies.com/Ontology1187021049.owl#>
SELECT ?doc ?arg
WHERE {doc:docName      doc:approfondisce      ?x .
      ?x                doc:approfondContenutoIn  ?doc .
      ?x                doc:approfondRispetto    ?arg }
```

In entrambi i casi è stato implementato un meccanismo per cui viene riusata la stessa struttura per la query, e vengono sostituite di volta in volta le relazioni coinvolte.

5.2 Modifica del motore JSPWiki

La parte che è stata modificata del motore JSPWiki riguarda essenzialmente l'interfaccia grafica delle pagine. Infatti, le pagine nella versione originale di JSPWiki permettono la gestione dei contenuti, e al più la creazione di un *LeftMenu*, in cui l'utente può aggiungere i link alle pagine che preferisce, senza, però, poter specificare quale relazione intercorre tra la pagina corrente e quelle selezionate a far parte del *LeftMenu*.

Con le modifiche apportate per questo lavoro di tesi si punta ad offrire un supporto a livello di interfaccia utente, da una parte per l'accesso ai metadati associati ad ogni pagina del Wiki semantico, dall'altra per permettere all'utente di contribuire all'arricchimento della KB.

5.2.1 Struttura delle pagine di JSPWiki

Dopo aver deciso la struttura grafica da dare ai nuovi componenti e la posizione in cui inserirli all'interno della pagina, si è passati alla ricerca e analisi dei *template* che gestiscono quella parte della pagina. In JSPWiki i *template* rappresentano il set principale di file HTML e JSP, memorizzati nella directory `JSPWiki/templates/`, che determinano l'aspetto finale delle pagine.

Si è notato che ogni pagina constava di due blocchi principali: uno a sinistra, in cui vengono visualizzati il LeftMenu, il logo, ecc., ed un altro, che occupa la parte restante della pagina, in cui vengono visualizzati i contenuti inseriti dagli utenti e le opzioni per la loro gestione.

5.2.2 Visualizzazione delle informazioni semantiche

Per visualizzare le informazioni semantiche relative ad ogni documento, si è deciso di creare due blocchi distinti da inserire uno sotto l'altro nella sezione sinistra della pagina.

Il codice relativo è stato aggiunto nel file `Favorites.jsp`, che gestisce quella parte della pagina.

I due blocchi compaiono come due trafiletti con header rispettivamente:

Relazioni pagina < nome della pagina> e *Classificazione pagina* <nome della pagina>.

Nel primo vengono visualizzate tutte le pagine del Wiki semantico in relazione con quella corrente e la relativa proprietà che le lega.

Il secondo invece illustra le caratteristiche del documento, in termini di argomenti trattati, citazioni, contesto, ecc.

In figura 5.2 possono si possono osservare i diversi componenti.

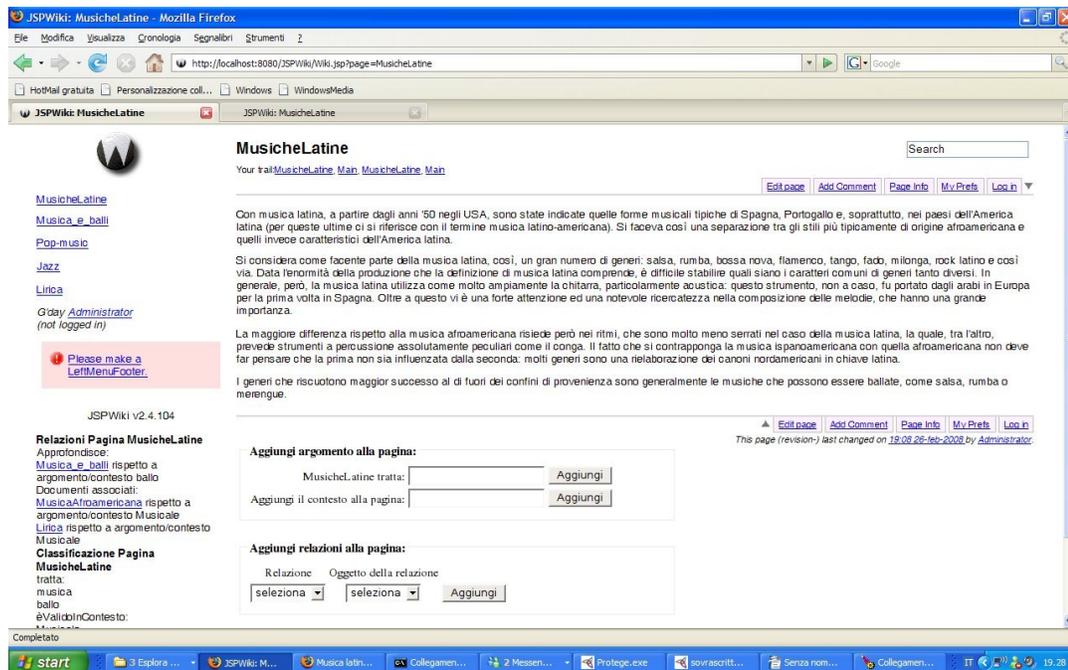


Figura 5.2: Schermata del prototipo di Wiki Semantico realizzato.

5.2.3 Creazione di relazioni tra documenti tramite un'interfaccia grafica

Una modifica ulteriore è stata fatta nel file `Footer.jsp`, in cui, al di sotto dello spazio dedicato ai contenuti, è stata aggiunta la sezione “*Aggiungi relazione*”.

Qui l'utente può scegliere di associare alla pagina corrente altre pagine del Wiki, secondo delle relazioni di associazione predefinite, piuttosto che aggiungere argomenti che vengono trattati nel documento, piuttosto che un nuovo contesto in cui i contenuti hanno senso.

Queste operazioni comportano l'arricchimento della KB di nuovi individui e relazioni fra di essi, che è possibile visualizzare nei due componenti grafici dedicati alle relazioni con altre pagine e alla classificazione del documento.

CAPITOLO 6

Conclusioni e sviluppi futuri

Con il presente elaborato è stata presentata l'implementazione di un'ontologia di contesto all'interno del motore JSPWiki.

La base di conoscenze, posta al di sotto del sistema, ha permesso, per ogni pagina del Wiki, il reperimento di legami impliciti con altre pagine sulla base di relazioni semantiche e deduzioni logiche estrapolate dall'ontologia mediante il supporto di un reasoner. Con queste funzionalità il motore Wiki modificato può essere considerato, a tutti gli effetti, un Wiki semantico, quindi si può dire di aver raggiunto l'obiettivo prefissato in fase di progettazione.

Come si è potuto constatare, il processo di apprendimento di un argomento migliora notevolmente se l'utente è in grado di identificare le relazioni di precedenza con cui sono legati gli argomenti d'interesse, per cui un sistema semantico, come quello proposto in questo lavoro di tesi, migliora la navigabilità e ricerca degli argomenti, dando all'utente un ordine ben preciso da seguire piuttosto che un insieme di link da visitare in modo non lineare.

Inoltre è da notare la semplicità con cui l'interfaccia permette all'utente, non solo il reperimento delle informazioni che introducono o approfondiscono un argomento, ma anche l'introduzione di nuovi legami tra le pagine, che l'utente stesso ritiene utili per l'apprendimento di una determinata materia.

L'ontologia sviluppata può essere considerata, in buona misura, innovativa; in questi anni sono state realizzate ontologie che, tipicamente, sono state incentrate sui contenuti dei documenti, mentre l'ontologia di contesto utilizzata per questo lavoro

di tesi, per come è stata definita e strutturata, si focalizza su relazioni che associano vari documenti fra di loro, non in base ai loro contenuti, ma in quanto documenti. In altre parole questo modello punta a enfatizzare i legami di propedeuticità, approfondimento, complementarità, ecc., che si possono creare tra due documenti, qualsiasi argomento essi trattino.

Questo carattere di generalità rispetto ai contenuti, svincola completamente l'ontologia dall'ambiente Wiki in cui è stata sperimentata per questo lavoro, e la rende adattabile a qualsiasi ambiente in cui vi è una collezione di documenti e pagine.

Questo potrebbe essere un buon punto di partenza per degli sviluppi futuri, che vedano questa ontologia, con opportuni adattamenti rispetto alle esigenze del caso, inserita nell'ambito di altri software o addirittura come base per migliorare la navigabilità dello stesso Web.

In ogni caso è ancora possibile apportare dei miglioramenti al software presentato: il Wiki semantico permette all'utente di inserire nuove relazioni tra le pagine, o nuovi argomenti e contesti, ma l'utente deve comunque attenersi alle relazioni predefinite nella base di conoscenza.

Una prospettiva interessante sarebbe quella di lasciare che gli utenti contribuiscano anche alla definizione di nuove relazioni, facendole diventare parte integrante dell'ontologia nel momento in cui una buona percentuale di utenti le propone come nuove proprietà, il tutto magari supervisionato da un amministratore della base di conoscenze.

In conclusione, con questo lavoro si è voluto mostrare il valore aggiunto che la semantica apporta alla ricerca di informazioni sul Web, e molti altri progetti sono già pronti per costruire passo passo un Web che predilige la *semantica* alla sola *sintassi*.

Bibliografia

- [Col06]M. Colombetti. Appunti delle lezioni di ingegneria della conoscenza.2006.
- [WWS] <http://it.wikipedia.org/wiki/Semantica>.
- [WW] http://en.wikipedia.org/wiki/Main_Page.
- [WebS] http://it.wikipedia.org/wiki/Web_Semantico
- [WJSP]<http://en.wikipedia.org/wiki/JSPWiki#History>
- [Sem] [http://it.wikipedia.org/wiki/Semantica_\(disambigua\)](http://it.wikipedia.org/wiki/Semantica_(disambigua))
- [Sp]Sparql.<http://www.xml.com/pub/a/2005/11/16/introducing-sparql-querying-semantic-web-tutorial.html>
- [Pr]Protégé. <http://protege.stanford.edu/>
- [Gr]GrOWL. <http://home.dei.polimi.it/arrigoni/GrOWL/>
- [JSP]<http://www.jspwiki.org/wiki/JSPWikiDevelopment>
- [Paper1] Sebastian Schaffert and Diana Bischof and Tobias Bürger and Andreas Gruber and Wolf Hilzensauer and Sandra Schaffert. Learning with Semantic Wikis. In Max Völkel and Sebastian Schaffert, editor(s), SemWiki, (206) CEUR-WS.org,2006.
- [Paper2] Nigel Shadbolt and Tim Berners-Lee and Wendy Hall. The Semantic Web Revisited. IEEE Intelligent Systems, (21)3:96-101, 2006.

Ringraziamenti

Ringraziamenti

Appendice A

Software utilizzato

- Protégé. <http://protege.stanford.edu/>
- GrOWL. <http://home.dei.polimi.it/arrigoni/GrOWL/>
- JSPWiki. <http://www.jspwiki.org/>
- Pellet.