

Alma Mater Studiorum - Università di Bologna

Facoltà di Scienze Matematiche Fisiche e Naturali
Corso di Laurea in Scienze e Tecnologie Informatiche

Tesi di Laurea in Programmazione

**Analisi e sviluppo di un engine per
l'interrogazione di endpoints
SPARQL tramite linguaggio naturale**

Autore:

Andrea Mazzotti

Relatore:

Antonella Carbonaro

Sessione II

Anno Accademico 2011/2012

Indice

Introduzione.....	1
1. Evoluzione e Futuro delle Intelligenze Artificiali.....	3
1.1 IBM Watson.....	4
1.2 Siri.....	7
1.3 The Open Cognition Project.....	9
2. Rappresentazione della Conoscenza.....	11
2.1 Web 3.0.....	12
2.2 Linked Data.....	13
2.2.1 303 URIs.....	15
2.2.2 Hash URIs.....	17
2.2.3 Quale strategia utilizzare?.....	19
2.3 RDF.....	20
2.4 Open Graph.....	22
2.5 Knowledge Graph.....	25
2.5.1 Project Glass e Majel.....	29
2.6 Wolfram Alpha.....	30
2.7 Problemi e critiche.....	32
2.7.1 Vastità e ridondanza dei dati.....	32
2.7.2 Indeterminatezza dei concetti (vocabolari).....	32
2.7.3 Inconsistenza e contraddizioni.....	33
2.7.4 Informazioni false.....	34
2.7.5 Privacy	36
3. Endpoints.....	38
3.1 DBpedia.....	39
3.1.1 Faceted Wikipedia Search.....	42
3.1.2 Relation Finder.....	44
3.2 Freebase.....	45
3.2.1 Organizzazione dei contenuti.....	46
3.2.2 Tutto è oggetto.....	46
3.2.3 Metaweb Query Language.....	47
3.2.4 Futuro e monetizzazione.....	49
4. Nuove interfacce:	
Linguaggio naturale.....	50
4.1 NLP-Reduce.....	53
4.2 Querix.....	55
4.3 Ginseng.....	58
4.4 Semantic Crystal.....	60
5. Natural To Artificial.....	62
5.1 Assistente Virtuale: I vantaggi.....	62
5.2 Gestire le differenti funzionalità.....	63

5.3 Strumenti utilizzati.....	65
5.3.1 WordNet.....	65
5.3.2 RelEx.....	66
5.3.3 Sesame.....	66
5.3.4 SPARQL	67
5.4 Database Design.....	68
5.4.1 Relation	70
5.4.2 Domain.....	71
5.4.3 Predicate.....	74
5.5 Patterns.....	76
5.5.1 Un problema finito.....	81
5.5.2 Debugger e crowdsourcing.....	81
5.6 Algoritmo.....	82
5.7 Risultati.....	84
Conclusioni.....	90

Introduzione

Il Web sin dalla sua diffusione a metà degli anni '90, ha radicalmente cambiato la vita quotidiana delle persone ed il modo in cui esse scambiano informazioni. Oggi l'accessibilità dei contenuti è una delle maggiori preoccupazioni degli addetti ai lavori del Web. Carpire immediatamente le informazioni di cui si hanno bisogno infatti è indice di una corretta realizzazione dello strumento in uso. Migliorare l'usabilità della rete significa anche combattere il fenomeno dell'information overloading o dello spam, aiutando gli utenti a filtrare i contenuti di cui non sono interessati, l'ormai sempre presente rumore di fondo che disturba le nostre quotidiane ricerche online. Grazie all'introduzione del Web 2.0 e del pacchetto di tecnologie che ha portato con sé, siamo in grado di fornire informazioni in modo dinamico in base all'utente che le richiede, alle sue preferenze dirette ed anche in base alle sue abitudini. Il livello di adattabilità dei siti agli utilizzatori finali è qualcosa che migliora di mese in mese e ci ha portati a poter disporre di enormi aggregati di informazioni di qualsiasi tipo, in qualsiasi momento. Quello che però risulta ancora oggi difficile, è compiere delle ricerche fra questi dati. Analizzarli, interpretarli e reinterpretarli nuovamente a seconda delle esigenze e dei contesti. Progetti quali Wikipedia riescono con successo a descrivere la realtà che ci circonda, ma delegano ancora all'utente il compito di elaborare i dati, di interpretarli e di combinarli fra loro. Con la presentazione del Web Semantico invece si è discussa soprattutto la possibilità di fornire i dati in un formato ben strutturato, all'interno di ontologie predefinite, dove ogni entità è parte di una gerarchia ed ha delle proprietà che possono essere comuni ad altre. Il bisogno di informazioni strutturate di fatto nasce dalla necessità di poter disporre di motori inferenziali, strumenti in grado di aiutare l'utente a dedurre e comporre informazioni partendo da dati base. Progetti quali DBpedia e Freebase ad oggi rappresentano la maggiore quantità di dati strutturati a disposizione per un ipotetico motore inferenziale.

Gli standard discussi dal World Wide Web Consortium, sono già oggi in grado di rispondere ai principali bisogni di software che vogliono approcciarsi alle informazioni in modo semantico, interpretando, deducendo, al pari di un essere umano, ma con il vantaggio di conoscere già tutte le entità descritte. L'idea alla base del Web Semantico o Web 3.0 è quindi quella di lasciare ad Internet il compito di comporre i documenti, lasciando all'utente il solo piacere di vederseli apparire davanti. Dal 2008 gli sforzi si sono intensificati e vari progetti di rappresentazione semantica delle informazioni hanno preso vita. Alcune soluzioni si sono scontrate con la necessità di trovare nuove forme di interfacce utente, esplorando l'ormai rodato riconoscimento vocale in un nuovo contesto di ricerca dei dati. Governi, istituti di ricerca, scuole, finanza, scienziati e persino aziende cominciano a sperimentare la pubblicazione di dati grezzi utilizzando gli strumenti del Web Semantico. Il principale ed iniziale vantaggio è quello di poter fornire dati potenzialmente accessibili da chiunque. Un formato aperto difficilmente può essere contestato e l'assenza di dati precedentemente lavorati o interpretati, è sinonimo di affidabilità e di fiducia. Non tutti sono comunque disposti a fornire dati attraverso tecnologie tanto trasparenti, ma il fenomeno rimane in continua espansione e la base dati cresce di anno in anno, soprattutto partendo da realtà altrettanto libere, Wikipedia in primis. Tuttavia, se la base del Web Semantico è in continua espansione, lo stesso non vale per gli strumenti che riescono ad utilizzare questo potenziale. Le soluzioni commerciali e più sviluppate riescono in qualche modo ad utilizzare i dati semantici in maniera efficiente, se pensiamo ai recenti Siri e Google Now, ma le aziende dietro di esse spesso si allontanano dallo standard comune, sviluppando linguaggi e standard diversi, che gli utenti sono costretti a scegliere l'uno in esclusione dell'altro. A livello accademico invece si cerca un di punto di forza comune, utilizzando software trasparenti, a licenza libera, che involino il maggior numero di partecipanti. Questo documento analizza lo stato attuale del Web Semantico, delle tecnologie che ne fanno parte e presenta lo sviluppo di una soluzione pensata come collante fra esse, che sia semplice da capire e da utilizzare.

1. Evoluzione e Futuro delle Intelligenze Artificiali

Per intelligenza artificiale si intende genericamente la capacità di un computer di emulare una mente umana. Nel corso degli anni è stato necessario differenziare le IA in forti e deboli. Le intelligenze artificiali di tipo debole sono quelle capaci di sostituire il cervello umano solo per specifici compiti e solo attraverso funzioni ed algoritmi statici pensati in precedenza dai programmatori. In questo ambito possiamo quindi trovare software per l'assistenza ai laboratori di ricerca, chatbots o intelligenze artificiali legate al gioco, tra cui ad esempio Deep Blue di IBM, capace di vincere a scacchi contro un campione mondiale, ma di fatto niente di più di un immenso insieme di euristiche. Le intelligenze artificiali di tipo forte quindi si distinguono per la capacità teorica di evolvere comprendendo i dati e di avere coscienza. La differenza sostanziale è rappresentata dall'effettiva capacità di emulare l'intera mente umana, piuttosto che singole funzioni. Nel corso degli anni l'intelligenza artificiale di tipo forte è stata considerata dai più troppo complessa per poter essere implementata con gli strumenti e le conoscenze attuali, tuttavia le soluzioni cosiddette "deboli" compiono enormi progressi nei loro campi e l'unione di tali tecnologie potrebbe forse portare a risultati concreti nella realizzazione di un'intelligenza artificiale generale. Durante il Techniche festival del 2011, in Guwahati, India, il chatbot Cleverbot è stato votato al 59,3% umano, garantendogli il successo del test di Turing. Risultato sorprendente se si pensa che i veri umani hanno riconosciuto se stessi al solo 63,3%. Il dibattito sull'effettiva capacità di pensiero del bot è comunque ancora aperto. Bisogna innanzi tutto definire cosa significhi *pensare*, ma è abbastanza semplice riconoscere che alla base dell'algoritmo utilizzato da Cleverbot ci sia un vasto insieme di euristiche collezionate negli anni tramite l'interazione con utenti reali.

1. Evoluzione e Futuro delle Intelligenze Artificiali

Cleverbot quindi non è in grado di capire realmente ciò che avviene durante il processo di chat, ma ha una grandissima esperienza in esso. L'impatto con l'utente finale e l'usabilità del software tuttavia sono punti molto importanti per rendere le intelligenze artificiali veramente utili nel contesto quotidiano. Le recenti implementazioni si concentrano infatti sull'interpretazione del linguaggio naturale e sulla realizzazione di veri assistenti virtuali, capaci di funzionare non solo davanti ad esperti del settore, ma anche in mano agli utenti comuni. Androidi da inserire in case di cura per anziani, agende virtuali capaci di organizzare gli appuntamenti degli utenti, navigatori satellitari con percorsi dinamici basati sul traffico reale e sulle abitudini dell'utente, assistenti di laboratorio in grado di aumentare l'accessibilità delle informazioni tecniche o di eseguire piccoli esperimenti di ricerca; il panorama delle intelligenze artificiali si articola di anno in anno e si avvicina sempre di più a contesti reali di vita quotidiana.

1.1 IBM Watson

Sviluppato dal 2006 all'interno del progetto DeepQA di IBM, Watson è un sofisticato sistema di Question Answering (QA) predisposto alla comprensione del linguaggio naturale, alla generazione di ipotesi ed all'apprendimento. Nel Febbraio del 2011 ha partecipato al quiz televisivo Jeopardy!, nel corso del quale ogni concorrente è tenuto a formulare una domanda, sulla base di una risposta fornita che può riguardare un vasto panorama di argomenti. Emulando il successo del Deep Blue, capace di sconfiggere il campione mondiale Garry Kasparov nel 1997, Watson è stato in grado di battere i due campioni in carica di Jeopardy!. Durante la sfida il sistema era scollegato da internet e le informazioni in input sono state fornite sotto forma di testo, nello stesso momento in cui venivano viste dai due concorrenti umani. La forza di Watson è quella di poter eseguire migliaia di algoritmi di analisi del linguaggio, al fine di estrarre un piccolo numero di soluzioni, per poi valutarle sulla base di un database per verificarne la veridicità.

1. Evoluzione e Futuro delle Intelligenze Artificiali

Il successo di Jeopardy! è un traguardo fondamentale per il destino del progetto e per IBM è stata una dimostrazione pubblica della potenza del sistema. Watson è infatti pensato per risolvere problemi concreti all'interno di realtà che hanno a che fare con l'analisi quotidiana di grosse quantità di dati. I principali casi d'uso elencati ufficialmente sono:

- **Diagnosi ed azione** – Assistenza agli operatori delle informazioni che hanno a che fare con un singolo caso per un singolo cliente, nell'individuazione di una condizione tra molte possibilità e nella formulazione di decisioni che ne derivino
- **Contattare il supporto centrale** – Esperienza self-service personalizzata per i clienti, sviluppando dinamicamente profili personali da dati non strutturati
- **Ricerca e scoperta** – Individuazione di studi e fonti di informazioni rare nella costruzione di casi per la ricerca originaria
- **Ottimizzazione dei processi** – Identificazione delle aree di miglioramento nei processi di business attraverso l'analisi di dati non strutturati prodotti dai documenti e dai processi descrittivi
- **Frode e gestione dei rischi** – Individuazione dei primi segnali di frode e gestione del rischio al fine di ridurre la responsabilità complessiva ed i costi per le imprese

Principalmente gli sforzi di IBM sono attualmente orientati alla risoluzione di problemi medici ed all'alta finanza. Nella divisione **Watson: Healthcare** si ha come obiettivo quello di costruire un assistente medico, in grado di spaziare tra i sintomi di un database vasto e sempre in crescita di malattie e di produrre una diagnosi personalizzata sulla base di un profilo storico del paziente.

1. Evoluzione e Futuro delle Intelligenze Artificiali

Watson: An expert diagnostic system

This groundbreaking system is based on the system used on Jeopardy! that can pore through the equivalent of 200 million pages of medical data and formulate a response **in less than 3 seconds**. Systems like this will enable healthcare professionals to make more informed decisions more quickly than ever before.

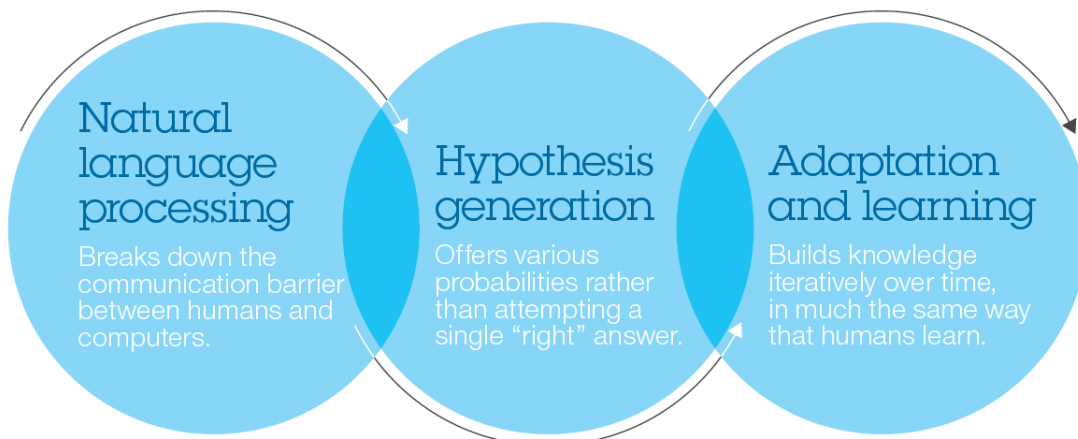


Fig 1.1-1 *Stories of a smarter planet : What makes you sick? - IBM*

Nel settore finanziario invece, **Watson: Finance** si occupa di rispondere a domande del tipo: *“Quali sono le imprese con più probabilità di essere bersaglio di acquisizione nel corso dei prossimi tre mesi?”*¹. Certe valutazioni sono difficilmente stimabili da esseri umani, poiché i dati relativi alle aziende non consistono solo in valutazioni di borsa, ma spesso e volentieri in articoli di giornali, report finanziari ed in generale in forme non strutturate che vanno interpretate.

Non bisogna però pensare ad un'intelligenza artificiale forte, in grado di sostituirsi ad un ricercatore o in generale ad una mente umana. Watson non è HAL di *“2001: Odissea nello spazio”*², piuttosto assomiglia al computer di bordo immaginato da Gene Roddenberry nella serie fantascientifica Star Trek: un assistente in grado di interpretare le richieste dell'utente, di rispondere e di agire in modo affidabile.

1 Perspectives on Watson: Finance <http://www.youtube.com/watch?v=IV6mfWWMIVg>

2 DeepQA FAQ: <http://researchweb.watson.ibm.com/deepqa/faq.shtml#16>

1.2 Siri

Siri nasce come applicazione per iOS sviluppata da Siri, Inc. Nell'Aprile 2010 Apple si mostra interessata, decide di comprare l'azienda, di annullare gli sviluppi per le piattaforme concorrenti e di integrare Siri a partire da iOS 5. L'integrazione consiste nel permettere al software di interagire con tante altre applicazioni, come quelle di posta, previsioni del tempo, messaggistica, appuntamenti, web browser, orologio, musica, mappe. L'applicazione è quindi divenuta ufficialmente ponte tra utente e smartphone, classificandosi a tutti gli effetti come assistente personale intelligente. La funzionalità di riconoscimento vocale permette all'utente infatti di parlare al proprio dispositivo, di fissare appuntamenti, mandare messaggi, fare ricerche via web, ricercare percorsi stradali ed in generale fornire assistenza per un insieme sempre più vasto di operazioni.



Fig 1.2-1 iPhone 4S: A Siri-ously slick, speedy smartphone - *ArsTechnica.com*

1. Evoluzione e Futuro delle Intelligenze Artificiali

L'introduzione di una interfaccia vocale per un assistente intelligente è sicuramente un enorme passo avanti per i dispositivi moderni, sempre più presenti nelle nostre vite ed in situazioni in cui non sempre si ha la comodità di poter usare mouse e tastiera o anche un touchscreen. La volontà di Apple, in linea con la filosofia aziendale, è quella di mantenere le cose semplici e Siri è pensato per rispondere a domande di tipo colloquiale, al pari di una persona comune. In locale quindi l'applicazione è pensata per adattarsi all'utente, al suo accento, al suo modo di esprimersi ed in remoto le domande vengono mandate ed analizzate da un server centrale che sfrutta vari database online e servizi di ricerca, quali Rotten Tomatoes, Wolfram Alpha e Bing Answers. Le difficoltà però corrispondono al bisogno dell'utente di poter far affidamento su Siri, l'assistente deve essere infatti in grado di funzionare nel momento del bisogno, che può corrispondere benissimo ad un locale affollato e rumoroso. I casi d'uso attuali e le condizioni di funzionamento accettabile però limitano ancora troppo l'esperienza, ma sicuramente il progetto è precursore per tutta una serie di agenti intelligenti che arricchiranno l'esperienza utente nel futuro prossimo. I dati di Park Associates infatti rivelano che il 64% degli utenti utilizza l'assistente diverse volte a settimana; di questi il 55% degli utenti è soddisfatta di Siri, il 9% no ed il resto si pone nel mezzo.³ Per una nuova funzionalità di tale portata, il feedback degli utenti è sicuramente incoraggiante. Corretti i problemi di pronuncia e di comprensione dell'audio, Siri potrebbe effettivamente porsi come modello di agente virtuale.

3 Park Associates - <http://www.parksassociates.com/blog/article/and-now-for-the-rest-of-the-siri-story>

1.3 The Open Cognition Project

OpenCog è un framework Open Source per la realizzazione di intelligenze artificiali. Originariamente era basato sui sorgenti del “*Novamente Cognition Engine*” di Novamente LLC distribuiti nel 2008 e ben presto si è affermato nella comunità, vantando fra i sostenitori l'Artificial General Intelligence Research Institute, il progetto Google Summer of Code e nel 2008 anche il Singularity Institute for Artificial Intelligence. OpenCog mette a disposizione l'interfaccia **AtomSpace** per la manipolazione e lo storage di hypergraph. AtomSpace di fatto è un contenitore di grafi ottimizzato alla costruzione di motori di deduzione probabilistica. Ogni Atom ha tipi differenti di **TruthValue** o **AttentionValue**, con i quali l'engine implementato può decretare il valore di verità di una conoscenza espressa nel database o scegliere se mantenere in memoria centrale o secondaria un certo Atom in base alla sua **ShortTermImportance**, **LongTermImportance** o **VeryLongTermImportance**. L'implementazione di una rete logica probabilistica è il punto di forza e la caratteristica centrale di OpenCog. Gli hypergraph sono pensati per essere **Self-modifying, Evolving Probabilistic**. Supportano quindi la modifica, l'aggiunta e l'eliminazione di Atom e di link fra essi attraverso i quali si possono creare associazioni tra nodi come ad esempio:

```
Subset gatto mammifero <1>
```

con la quale si esprime una relazione tra il concetto di gatto e di mammifero, dove 1 è una componente di TruthValue che in questo caso esprime il massimo grado di veridicità.

Ben Goertzel, sviluppatore del primo progetto dal quale OpenCog deriva, utilizza questo database design come implementazione della sua teoria, secondo la quale “*mind is made of pattern*”⁴. Con questo principio in mente, si sono sviluppate quindi diverse funzionalità, quali l'Associazione, l'Allocazione Differenziata a seconda del grado di attenzione, la creazione di Pattern e l'Assegnazione di Credito, come ad esempio la modifica del TruthValue per un Atom.

4 Goertzel B. (1991) *The Hidden Pattern*. Brown Walker Press (June 6, 2006)



Fig 1.3-1 Cane virtuale implementato utilizzando OpenCog

2. Rappresentazione della Conoscenza

Il concetto di intelligenza artificiale forte porta di conseguenza a ragionare sulla rappresentazione della conoscenza. Come può una macchina interpretare le informazioni che quotidianamente trasmettiamo? Come può un motore inferenziale lavorare con efficacia sui tanti dati forniti da Wikipedia? Come può imparare e salvare nuove informazioni che derivano dalle deduzioni compiute? La questione della rappresentazione matematica della conoscenza è di primaria importanza, sia nell'ambito delle intelligenze artificiali, sia nell'ambito del Web Semantico. Rappresentare i dati in modo strutturato significa creare un'ontologia, definire a tavolino delle classi che rappresentino i fenomeni e gli oggetti del mondo reale, che però è composto di una quantità infinita di eccezioni. Inoltre non è ben chiaro come funzioni l'interpretazione e la memorizzazione delle informazioni nel nostro cervello. Gli studi di Richard M. Restak⁵ su pazienti affetti da danni cerebrali, fanno intuire che il cervello organizza le informazioni attraverso categorie. Alcuni soggetti ad esempio pur essendo capaci di descrivere tutte le istanze di una particolare categoria di oggetti, non sono in grado di fare altrettanto per altre categorie. In particolare un paziente in grado di descrivere accuratamente gli oggetti inanimati, rispondeva "*è un animale*" se posto davanti alla foto di un rinoceronte, un maiale o qualsiasi altro animale. Tuttavia egli era capace di descriverne le caratteristiche, per il rinoceronte ad esempio "*Enorme, pesa più di una tonnellata, vive in Africa*". Restak suggerisce anche che i moduli o le categorie possano anche dipendere dai sensi, pazienti in grado di descrivere proprietà olfattive di particolari animali, non erano in grado di descriverne altre visive e viceversa. Le informazioni sembrano quindi essere memorizzate come categorie interconnesse, nelle quali le stesse entità sono collegate fra loro.

5 Richard M. Restak – "*The Modular Brain*"

2. Rappresentazione della Conoscenza

L'idea alla base del Web Semantico è simile e potrebbe quindi fornire una fonte di informazioni molto utile per una intelligenza artificiale che voglia simulare il cervello umano. Il problema fondamentale però rimane quello della distanza intrinseca fra uomo e macchina. Come esseri umani siamo portati a descrivere la realtà così come la percepiamo, ma la percezione è relativa al singolo osservatore, come risolvere quindi questa diversità? Una macchina non è ancora in grado di osservare i fenomeni del mondo reale autonomamente e la nostra rappresentazione è tutto ciò di cui può disporre. Di qui l'importanza di trovare un linguaggio formale standard che sia comune a tutti e che possa essere abbastanza dinamico da assorbire ogni divergenza.

2.1 Web 3.0

Il World Wide Web che iniziò a diffondersi nei primi anni novanta, era composto da tre tecnologie semplici ed essenziali: Identificatori delle pagine (URI), un linguaggio di pubblicazione (HTML) ed un protocollo di trasmissione (HTTP). Ai suoi albori quindi il Web era facilmente accessibile e dai contenuti fondamentalmente aperti. Le informazioni erano statiche, l'HTML puro era consultabile anche da editor di testo, non c'era il bisogno vero e proprio di un browser.

Il Web 2.0 introduce invece i contenuti dinamici ed una vasta schiera di linguaggi di programmazione server e client side. L'utente non ha più a disposizione i dati del sito, ma solo una vista parziale di quello che il sito decide di mostrare. Le informazioni sono nascoste e portate alla luce spesso sotto forma di variabili dai nomi incomprensibili ed indecifrabili. Ogni sito parla la sua lingua e la comunicazione avviene solamente attraverso l'utilizzo di interfacce (API), l'una diversa dall'altra, che vengono messe a disposizione dei programmatori. Nel corso degli anni il Web 2.0 ha offerto funzionalità sempre maggiori, arrivando addirittura a sostenere interi sistemi operativi ⁶ o suite di programmi. Si parla sempre più spesso di nuove web applications, di cloud computing, ma le informazioni sono davvero fruibili?

Interrogandosi sul futuro del Web 3.0 e cercando di definire i suoi standard, Tim Berners-Lee, inventore del Web ha introdotto il concetto di Web Semantico.⁷ Per essere

⁶ eyeOS Cloud Desktop - <http://www.eyeos.org/>

⁷ Victoria Shannon - *A 'more revolutionary' Web* – The New York Times

2. Rappresentazione della Conoscenza

Semantico il Web ha bisogno di poter *capire* le informazioni e di poter trasmettere la propria conoscenza in un linguaggio accessibile.

Il passaggio ad una versione successiva può essere automatico? Si può rappresentare la conoscenza in un formato digitale, con un'evoluzione degli attuali strumenti? O bisognerà piuttosto ritornare sui propri passi per quanto riguarda l'accesso alle informazioni? Le proposte sono tante e da anni si sta cercando di definire uno standard.

2.2 *Linked Data*

Per Tim Berners-Lee Semantic Web significa "*Raw Data!*"⁸ e la struttura nella quale disporli è un grafo. Il concetto di Linked Data esprime la necessità di rappresentare, attraverso gli standard RDF e SPARQL, ogni oggetto reale e virtuale con un URI di riferimento e di collegare questi oggetti tra loro tramite links ad altri URI. Il design utilizza le tre tecnologie standard e fondamentali del Web (HTML, HTTP, URIs), più altri standard semantici per la rappresentazione delle entità. Il fine principale è quello di assegnare ad ogni oggetto del mondo reale e ad ogni documento del mondo virtuale, un indirizzo che inizi con HTTP e che ogni indirizzo restituisca informazioni importanti in un formato aperto, tra le quali relazioni ad altri oggetti, attraverso altri indirizzi HTTP.

Le specifiche dei Linked Data permettono di fatto l'associazione di concetti che fanno parte anche di Dataset (o Database) differenti, come ad esempio:

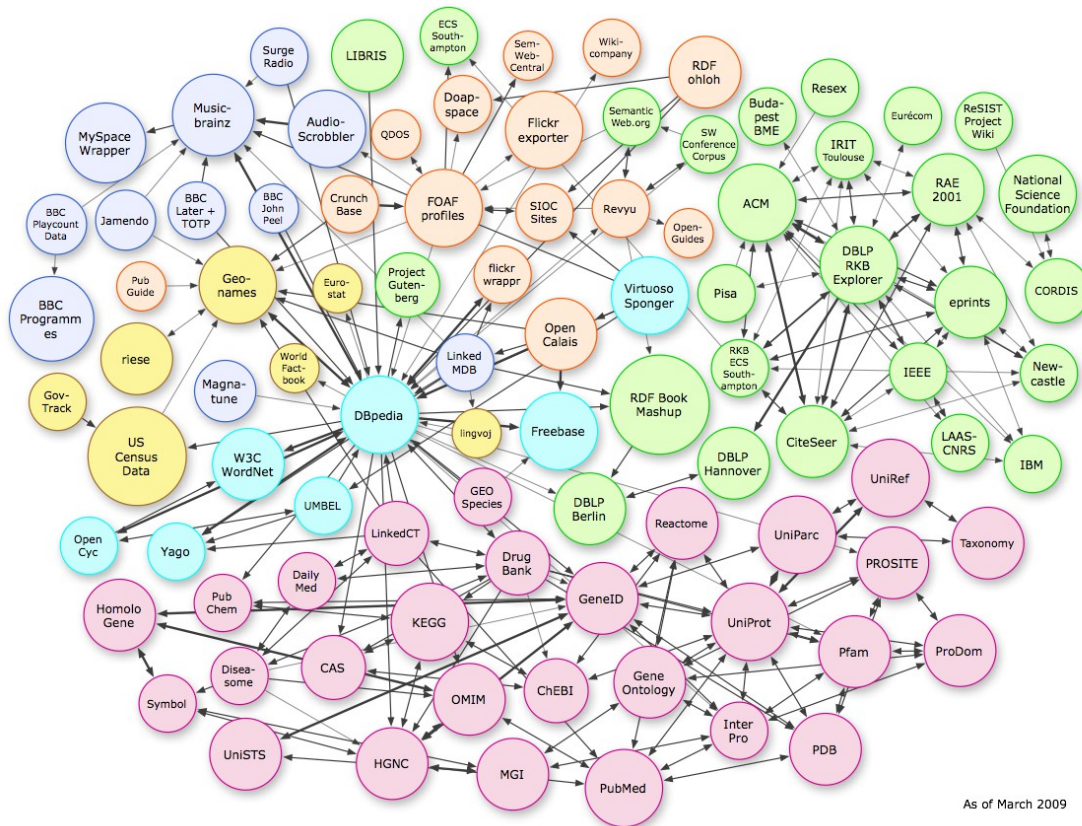
```
Subject: http://data.linkedmdb.org/resource/film/77  
Predicate: http://www.w3.org/2002/07/owl#sameAs  
Object: http://dbpedia.org/resource/Pulp\_Fiction\_%28film%29
```

Questo principio, unito al lavoro comunitario, ha portato alla nascita del progetto *Linking Open Data*⁹, nel quale si sta costruendo un grafo in continua espansione, collegando fra loro oggetti provenienti da Dataset differenti, utilizzando triple RDF.

⁸ TED Talks – *Tim Berners-Lee on the next Web*

⁹ <http://www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData>

2. Rappresentazione della Conoscenza



Una delle principali questioni da affrontare, è quella di differenziare gli URI di oggetti esistenti nel mondo reale, dai documenti Web che li descrivono. La pratica più semplice, sarebbe quella di utilizzare URI differenti per evitare ogni tipo di ambiguità, ma questo tipo di approccio è incoerente con il principio di "Web of Data", nel quale umani e macchine devono poter accedere a dati per loro comprensibili, in quanto si violerebbe il principio di rappresentare un'entità con un indirizzo univoco HTTP. Le due strategie discusse per affrontare questo problema quindi utilizzano il meccanismo di content negotiation¹⁰ di HTTP per poter esprimere il tipo di documento richiesto; HTML nel caso l'utente sia un essere umano o RDF nel caso la richiesta venga fatta da una macchina.

¹⁰ Hypertext Transfer Protocol – HTTP/1.1 - RFC2616

2.2.1 303 URIs

Nella strategia 303 URIs, il server risponde al client con il codice HTTP 303 See Other e con l'URI di riferimento per l'oggetto richiesto. Questo processo è chiamato 303 redirect e permette al client di accedere ai contenuti indirizzati dal nuovo URI.

Prendiamo ad esempio il Dataset di Big Lynx e assumiamo di dover accedere al documento RDF che contiene i dati di Dave Smith. Big Lynx utilizza tre diversi URI di riferimento:

- <http://biglynx.co.uk/people/dave-smith> (URI che identifica la persona Dave Smith)
- <http://biglynx.co.uk/people/dave-smith.rdf> (URI che identifica il documento RDF che descrive Dave Smith)
- <http://biglynx.co.uk/people/dave-smith.html> (URI che identifica il documento HTML che descrive Dave Smith)

Per accedere al documento RDF, il client dovrà inizialmente connettersi al server <http://biglynx.co.uk/> e mandare la seguente richiesta HTTP GET:

```
1 GET /people/dave-smith HTTP/1.1
2 Host: biglynx.co.uk
3 Accept: text/html;q=0.5, application/rdf+xml
```

L'header `Accept:` esprime la preferenza verso il formato RDF piuttosto che HTML. Il server dovrebbe rispondere:

```
1 HTTP/1.1 303 See Other
2 Location: http://biglynx.co.uk/people/dave-smith.rdf
3 Vary: Accept
```

Questo è di fatto un 303 Redirect, che contiene la locazione della risorsa richiesta, nel formato richiesto, nell'header `Location:`

2. Rappresentazione della Conoscenza

Successivamente il client tenterà di accedere al nuovo URI, utilizzando un'altra richiesta HTTP GET:

```
1 GET /people/dave-smith.rdf HTTP/1.1
2 Host: biglynx.co.uk
3 Accept: text/html;q=0.5, application/rdf+xml
```

Il server a questo punto risponderà con il documento RDF richiesto e lo status HTTP 200 (OK) per confermare la correttezza della richiesta.

```
1 HTTP/1.1 200 OK
2 Content-Type: application/rdf+xml
3
4
5 <?xml version="1.0" encoding="UTF-8"?>
6 <rdf:RDF
7   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
8   xmlns:foaf="http://xmlns.com/foaf/0.1/">
9
10 <rdf:Description rdf:about="http://biglynx.co.uk/people/dave-smith">
11 <rdf:type rdf:resource="http://xmlns.com/foaf/0.1/Person"/>
12 <foaf:name>Dave Smith</foaf:name>
13 ...
```

2.2.2 Hash URIs

Una delle critiche alla strategia 303 URI, è che richiede due richieste HTTP per ottenere una singola descrizione di un oggetto. La strategia Hash URI è stata quindi pensata per utilizzare un singolo URI, che però contenga al suo interno una parte speciale, chiamata fragment identifier e separata dal un simbolo hash (#). Come esempio utilizziamo il vocabolario SME (Small and Medium-sized Enterprises) di Big Lynx, identificato dall'URI `http://biglynx.co.uk/vocab/sme/`

All'interno del vocabolario ci sono diverse proprietà, tra le quali `SmallMediumEnterprise` o `Team`, che possono essere richieste utilizzando il fragment identifier alla fine dell'URI:

- `http://biglynx.co.uk/vocab/sme#SmallMediumEnterprise`
- `http://biglynx.co.uk/vocab/sme#Team`

Per prima cosa, il client tronca l'URI rimuovendo l'identificativo (ad esempio `#Team`). Successivamente manda la seguente richiesta HTTP GET al server:

```
1 GET /vocab/sme HTTP/1.1
2 Host: biglynx.co.uk
3 Accept: application/rdf+xml
```

2. Rappresentazione della Conoscenza

Il server risponde mandando l'intero documento RDF/XML:

```
1 HTTP/1.1 200 OK
2 Content-Type: application/rdf+xml;charset=utf-8
3
4
5 <?xml version="1.0"?>
6 <rdf:RDF
7 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
8 xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#">
9
10 <rdf:Description
rdf:about="http://biglynx.co.uk/vocab/sme#SmallMediumEnterprise"
>
11 <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Class" />
12 </rdf:Description>
13 <rdf:Description
rdf:about="http://biglynx.co.uk/vocab/sme#Team">
14 <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-
schema#Class" />
15 </rdf:Description>
16 ...
```

Il documento quindi non contiene solo la proprietà #Team all'interno del vocabolario, ma anche #SmallMediumEnterprise. A questo punto è quindi compito del client trattenere solo la tripla RDF di riferimento a #Team e scartare il resto dei dati.

2.2.3 Quale strategia utilizzare?

Entrambe le strategie descritte hanno vantaggi e svantaggi. In particolare Hash URI ha il vantaggio di ridurre le richieste HTTP necessarie, diminuendo la latenza di accesso alle informazioni, ma d'altra parte al client vengono restituite tutte le informazioni dell'URI non-fragment, che non tengono conto dell'effettivo interesse del client e vengono di fatto scartate. Se il file RDF richiesto ad esempio contiene un grande quantitativo di triple, l'approccio Hash URI porta a trasmettere un grande quantitativo di dati inutili trasmessi al client. L'approccio 303 URI invece permette di essere molto più flessibile e selettivo, ridirigendo il client ad ogni risorsa. La soluzione migliore prevede quindi di utilizzare entrambe le strategie: 303 URI è spesso usato per restituire risorse che sono parte di enormi database, come ad esempio le pagine che esprimono le entità di DBpedia. Hash URI invece è utilizzato per identificare le proprietà all'interno di vocabolari RDF, in quanto sono spesso files esigui contenenti un migliaio di triple e poiché è conveniente per il client ottenere tutto il vocabolario una volta sola, piuttosto che richiedere di volta in volta le singole proprietà.

2.3 RDF

Il modello RDF¹¹, originariamente pensato come un modello per esprimere metadati, rappresenta le informazioni attraverso grafi diretti con nodi ed archi etichettati ed è diventato il principale strumento per la descrizione concettuale dei dati nel web. RDF aspira a diventare *lingua franca* nell'ambito del Web Semantico, mirando a diventare ponte di comunicazione tra linguaggi differenti, in quanto molto semplice da leggere nel formato RDF/XML e concettualmente molto vicino al ormai rodato modello entity-relationship.

In RDF la descrizione di una risorsa è rappresentata in una serie di triple, rappresentate come Subject, Predicate e Object.

<i>Roma</i>	<i>è in</i>	<i>Italia</i>
<i>Subject</i>	<i>Predicate</i>	<i>Object</i>

Il soggetto di una tripla è un identificativo URI che descrive la risorsa. L'oggetto può essere invece una stringa, un numero, una data o anche l'URI di un'altra risorsa. Il predicato nel mezzo esprime la relazione tra soggetto ed oggetto. Anche il predicato è definito da un URI che viene dal *vocabolario*, il quale contiene una collezione di URI che possono essere usati per rappresentare le informazioni di un certo dominio.

11 Graham Klyne and Jeremy J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax - W3C Recommendation, 2004. <http://www.w3.org/TR/rdf-concepts/>

2. Rappresentazione della Conoscenza

Esistono due tipi principali di triple RDF, distinte in Literal Triples e RDF Links:

1. **Literal Triples:** come oggetto utilizzano una stringa, un numero o una data. Sono usate per descrivere le proprietà delle risorse, ad esempio il nome o la data di nascita di una persona. Possono anche essere inclusi i tag di lingua, per esprimere la lingua in cui l'oggetto è descritto, come ad esempio i diversi titoli di un film in lingue diverse.

`http://biglynx.co.uk/people/matt-briggs http://xmlns.com/foaf/0.1/nick "Matty"`

2. **RDF Links:** descrivono le relazioni tra due risorse e consistono di tre link RDF: soggetto, oggetto e predicato. Si differenziano tra internal ed external RDF links. I link interni esprimono relazioni tra risorse di uno stesso data set, come:

`http://biglynx.co.uk/people/matt-briggs http://xmlns.com/foaf/0.1/knows http://biglynx.co.uk/people/dave-smith`

Quelli esterni invece connettono risorse provenienti da sorgenti differenti.

Questo tipo di link è fondamentale nella costruzione del Web of Data, in quanto connettono diversi data set che altrimenti sarebbero isolati fra loro.

`http://biglynx.co.uk/people/matt-briggs http://biglynx.co.uk/vocab/sme#leads http://biglynx.co.uk/teams/production`

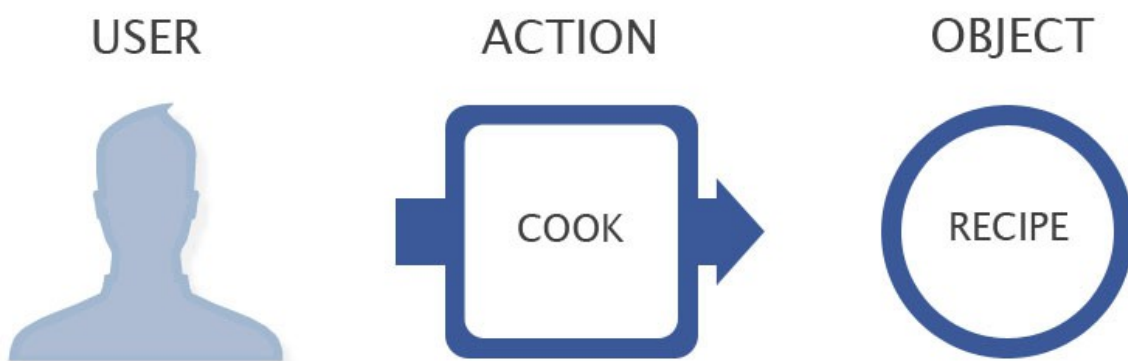
2.4 Open Graph

L'Open Graph protocol permette agli sviluppatori di integrare le loro pagine nel *social graph*. Il concetto di grafo sociale è divenuto popolare durante la conferenza Facebook f8 del 24 Maggio 2007, per spiegare il funzionamento della piattaforma Facebook. Ad oggi il termine identifica il grafo sociale che include tutti gli utenti di internet, definisce le loro proprietà e le relazioni fra essi.

Il protocollo, basato inizialmente su RDFa, prevede l'inserimento dei tag `<meta>` all'interno del `<head>` di una pagina web. I quattro tag obbligatori sono:

- **og: title** - il titolo dell'oggetto come dovrebbe apparire nel grafo
- **og: type** - il tipo dell'oggetto, ad esempio "video.movie". A seconda del titolo è possibile che siano richieste proprietà aggiuntive
- **og: image** - un'immagine che rappresenti l'oggetto nel grafo
- **og: url** - URL canonico dell'oggetto, utilizzato come ID permanente nel grafo

L'obiettivo di Facebook è quindi quello di connettere tra di loro le persone fisiche del Web, le loro proprietà, interessi e collegamenti ad altre entità fisiche del mondo reale o virtuale ad essi collegati.



2. Rappresentazione della Conoscenza

Prendiamo ad esempio le informazioni personali di un utente per quanto riguarda i *likes*:

```
{
  "data": [
    {
      "name": "Let's Build a Goddamn Tesla Museum",
      "category": "Cause",
      "id": "135116486631346",
      "created_time": "2012-08-20T07:55:04+0000"
    },
    {
      "name": "Grooveshark",
      "category": "Interest",
      "id": "103122196394517",
      "created_time": "2012-04-29T07:49:36+0000"
    },
    {
      "name": "Avira",
      "category": "Internet/software",
      "id": "136240249747850",
      "created_time": "2012-04-16T22:18:10+0000"
    },
    .
    .
  ]
}
```

2. Rappresentazione della Conoscenza

La pagina mostra i dati, solamente i dati, dai quali in questo caso è possibile estrapolare le preferenze di un utente e percorrendo l'intero grafo, capire quali affinità ha questo utente con altri, di che nazionalità è la maggior parte degli utenti che utilizza l'antivirus Avira, quali news vengono lette in determinate città, quante delle persone che mangiano da Mc Donalds bevono anche Coca Cola ed in generale è possibile produrre in maniera del tutto dinamica un'analisi sul grafo per ottenere le informazioni desiderate.

Uno degli aspetti non tecnici, ma fondamentali dell'Open Graph, è quindi quello della privacy. Facebook promuove continuamente strumenti di protezione dei propri dati, di controllo del livello di pubblicazione, ma la realtà è che gli utenti non sono veramente in possesso dei propri dati. La loro cancellazione non è trasparente, il rimanere anonimi nemmeno e questo fiume di informazioni è nelle mani di chi detiene il servizio ed i server, ovvero Facebook.

Open Graph però significa anche *Facebook al di fuori di Facebook*, ovvero ogni sviluppatore può decidere di pubblicare i propri dati utilizzando il protocollo sul proprio dominio. L'obiettivo di Facebook è appunto quello di riuscire ad integrare il mondo esterno alla piattaforma, utilizzando i metadati come uno strumento di colonizzazione. Lo svantaggio di una scelta simile è quella di lasciare a terzi uno strumento con il quale offrire servizi compatibili, ma terzi rispetto a Facebook, come ad esempio altri social network quali Diaspora o Google+.

Dal punto di vista dell'utente però, significherebbe poter abbattere le barriere del social networking, eliminando l'obbligo di aderire ad un servizio, piuttosto che ad un altro, poiché meno popolare.


2.5 Knowledge Graph

Il Knowledge Graph è una base di dati semantica in uso da Google per il miglioramento dei risultati di ricerca. Annunciato a Maggio 2012, ha cominciato a comparire tra le ricerche degli utenti offrendo risultati strutturati e legati tra loro semanticamente. L'obiettivo è quello di fornire direttamente le informazioni agli utenti, senza che essi debbano visitare altri siti per estrarle.


Le informazioni contenute dal Knowledge Graph derivano da diverse fonti, quali Freebase, il CIA World Factbook, Wikipedia e Google Maps. Attualmente appare in tre diverse forme:

1. Un form di disambiguazione, nel caso i risultati siano multipli


See results about



[Charlotte Brontë](#)
Charlotte Brontë was an English novelist and poet, the eldest of the three Brontë sisters who survived ...



[Bronte](#)
New South Wales
Bronte is a beachside suburb of Sydney, in the state of New South Wales, Australia. Bronte is located 8 ...




[Bronte, Ontario](#)
Bronte is the community that makes up much of the west end of Oakville, in Ontario, Canada. Twelve Mile ...

2. Rappresentazione della Conoscenza

2. Un form contenente i dati strutturati relativi all'entità e i link per approfondire la ricerca

Charlotte Brontë



Charlotte Brontë was an English novelist and poet, the eldest of the three Brontë sisters who survived into adulthood, whose novels are English literature standards. She wrote Jane Eyre under the pen name Currer Bell. [Wikipedia](#)

Born: April 21, 1816, [Thornton, Bradford](#)






Died: March 31, 1855, [Haworth](#)

Spouse: [Arthur Bell Nicholls](#) (m. 1854–1855)






Movies: [Jane Eyre](#)

Siblings: [Emily Brontë](#), [Anne Brontë](#), [Branwell Brontë](#), [Maria Brontë](#), [Elizabeth Brontë](#)

Books

 <p>Jane Eyre 1847</p>	 <p>Villette 1853</p>	 <p>Shirley 1849</p>	 <p>The Professor 1857</p>	 <p>Poems by Currer, Ellis, an...</p>
---	--	---	--	--

People also search for

 <p>Emily Brontë</p>	 <p>Jane Austen</p>	 <p>Anne Brontë</p>	 <p>Charles Dickens</p>	 <p>George Eliot</p>
---	--	--	---	---

[Feedback](#)

2. Rappresentazione della Conoscenza

3. Una mappa con i luoghi di interesse più vicini alla locazione dell'utente



2. Rappresentazione della Conoscenza

Il Knowledge Graph risponde al bisogno di rispondere a query complesse, ad esempio "Quante donne hanno vinto il premio Nobel negli ultimi 10 anni?"¹², ma anche a domande dirette o informazioni fortemente basate sulla precisa locazione dell'utente, in funzione di Google Now, l'assistente personale concorrente a Siri.

Anche la ricerca su web, ovviamente, risentirà positivamente della nuova tecnologia. In un documento di ricerca di Yahoo! Research¹³, di ispirazione per Google nella creazione del Knowledge Graph, vengono infatti delineate le due principali attività dell'utenza web:

- Un utente può avere in mente un'istanza precisa di un concetto e vuole ricercare o scorrere gli attributi legati ad essa, come ad esempio il trovare il menu o le recensioni di un particolare ristorante.
- Un utente vuole cercare un insieme di istanze di un concetto che soddisfino determinate condizioni, come ad esempio tutti i ristoranti che non siano distanti più di due chilometri da Cesena.

Dalla ricerca di Yahoo! risulta quindi che le informazioni aggregate sono utili alla ricerca, in quanto un utente che cerca una singola istanza di un concetto, è comunque interessato ad avere maggiori informazioni su di esso. Come costruire però un *Web di Concetti*?

- **Estrapolazione delle informazioni** – Questa operazione estrae dati dai documenti, come ad esempio l'indirizzo di un ristorante dal suo sito web o una lista di pubblicazioni da una pagina personale
- **Linking** – Devono essere creati collegamenti tra record esistenti. Ad esempio una recensione al ristorante di riferimento o due libri di uno stesso autore.
- **Analisi** – Questa operazione allega i metadata ai records, ad esempio identificando un dato ristorante ad una particolare categoria di cucina.

12 Introducing the Knowledge Graph - <http://www.youtube.com/watch?v=mmQl6VGvX-c>

13 A Web of Concepts – Yahoo! Research

2.5.1 Project Glass e Majel

Gli approcci di Google verso il mondo della semantica sono vari, ma forse quello più importante riguarda il Project Glass, un progetto di sviluppo che mira a sviluppare un display di realtà aumentata, utilizzando una montatura ad occhiale. Il dispositivo sarà dotato di microfono ed attualmente l'unica interfaccia utente di input è quella vocale. A questo proposito è necessario introdurre Google Assistant, una applicazione di riconoscimento vocale e di assistenza all'utente in sviluppo da tempo e non ancora rilasciata. Il progetto Assistant consiste di tre parti fondamentali:

- Ricavare la conoscenza del mondo in un formato comprensibile dal computer
- Creare un livello di personalizzazione
- Creare un assistente software che possa aiutare nel raggiungimento di obiettivi di vita reale, piuttosto che fornire semplici risultati di ricerca

Inizialmente il progetto era conosciuto come Google Majel, come esplicito riferimento a Majel Barrett-Roddenberry, l'attrice della serie Star Trek che da sempre ha dato vita alla voce dei computers di bordo.

2.6 Wolfram Alpha

The image shows a screenshot of the Wolfram Alpha website. At the top, the logo "WolframAlpha" is displayed in red and orange, with the tagline "computational... knowledge engine" to its right. Below the logo is a search bar containing the query "What is the capital of the most populated country in Europe?". To the right of the search bar are icons for a star and a menu. Below the search bar are several small icons: a keyboard, a camera, a list, and a refresh icon. To the right of these icons are the words "Examples" and "Random". Below the search bar is a box containing the text "Assuming Europe | Use Europe with Russia and Turkey or Europe with Russia instead". Below this box is a large box containing the "Input interpretation" section. This section shows a diagram with three boxes: "most country" on the left, "in Europe" and "by population" in the middle, and "capital city" on the right. Below the "Input interpretation" section is the "Result" section, which displays "Berlin, Germany" and "(Germany)".

Nel 2009 Wolfram Alpha è stato rilasciato da Wolfram Research come *"answer engine"*: un servizio online che risponde direttamente a domande reali, interrogando una base dati strutturata. Attualmente fornisce una sezione per l'esplicitazione dell'input interpretato ed è in grado grazie alla forma strutturata risultante, di dedurre la soluzione sulla base di una conoscenza pregressa, salvata sul knowledge base proprietario ed interno al servizio, oppure estrapolata da documenti esterni.

2. Rappresentazione della Conoscenza

I risultati sono attendibili?

In un articolo dello Spiegel Online¹⁴ viene analizzato il servizio offerto da Wolfram Alpha, ma giunti alle conclusioni, il commento diviene negativo per quanto riguarda la correttezza dei dati forniti, con un occhio di riguardo alla problematica delle fonti da cui essi vengono tratti:

"How can you trust an information processor that sometimes spits out results you don't understand, doesn't explain how it came to these answers and doesn't reveal its sources?"

In un altro articolo¹⁵ di Patrick McCormick il motore online viene messo alla prova su dati medici ed anche in questo caso il feedback finale è negativo, in quanto i dati forniti sono errati, ma soprattutto non è chiara la loro fonte.

Il punto critico del progetto Wolfram Alpha sembra quindi essere quello della propria knowledge base. I risultati sembrano a prima vista corretti, il motore interpreta correttamente molte domande e riesce a dare una risposta, ma queste risposte sono affidabili? I dati sono aggiornati? Da dove vengono? Da chi sono mantenuti ed è possibile sottoporre correzioni? Una azienda privata può realisticamente mantenere un database così vasto senza il contributo della comunità? Sono domande molto delicate a cui Wolfram Research dovrà rispondere per guadagnare la fiducia degli utenti e per garantire l'affidabilità del proprio sistema.

14 Von Konrad Lischka and Matthias Kremp - *"What Google's New Rival Knows – And Doesn't"*

15 Patrick McCormick - *Why You Shouldn't Trust WolframAlpha For Medicine* -
<http://patrickmd.net/blog/2009/05/23/why-you-shouldnt-trust-wolframalpha-for-medicine/>

2.7 Problemi e critiche

2.7.1 Vastità e ridondanza dei dati.

Uno dei problemi di primo impatto del Web Semantico è quello dell'Information Overload. Le informazioni ad oggi disponibili sono tante, ma soprattutto contengono duplicati e dati non rilevanti che è difficile, se non impossibile eliminare. La vastità dei database quindi aggrava gli altri problemi, in quanto i sistemi devono essere abbastanza complessi da ragionare con un numero elevato di concetti, più o meno consistenti. Il concetto di Web Semantico o meglio di Linked Data però è allo stesso tempo una soluzione al problema, in quanto i dati presentati in forma grezza non hanno necessità di essere ripubblicati più e più volte in differenti formati, ma sarebbe possibile esporli direttamente nella modalità preferita, tenendo come base un'unica fonte.

2.7.2 Indeterminatezza dei concetti (vocabolari)

I vocabolari, o prefissi, descrivono le relazioni tra entità. Gli attributi di ogni vocabolario, definiscono i predicati della relazione tra soggetto ed oggetto, nel formato RDF. In base al tipo di entità descritta, si potrà utilizzare uno o più vocabolari. Uno dei più semplici da capire è forse il Friend of a Friend (FOAF¹⁶), che descrive le relazioni tra persone. Fra i suoi attributi possiamo trovare *name*, *gender*, *title*, *homepage*, *weblog*, *friendOf*, *knows* e via dicendo.

Il corretto utilizzo e la comprensione degli attributi è molto importante, poiché in sistemi complessi porta alla generazione di diverse ambiguità di difficile risoluzione da parte di motori inferenziali, ma anche da parte di utenti in carne ed ossa che sottopongono query.

16 Leigh Dodds - *An Introduction to FOAF* - <http://www.xml.com/pub/a/2004/02/04/foaf.html>

2. Rappresentazione della Conoscenza

In database vasti come DBpedia inoltre i vocabolari utilizzati sono molti¹⁷ e spesso le proprietà di un vocabolario sono ridondanti rispetto a quelle di un altro, oppure alcuni attributi con lo stesso nome in diversi vocabolari, hanno significato differente.

Questo capita però anche all'interno di uno stesso vocabolario, considerando **dbprop** ad esempio, di grande uso sull'endpoint di DBpedia, concetti come **birthDate** e **dateOfBirth** hanno significati piuttosto diversi tra loro.

birthDate ritorna infatti una data completa: *1879-03-14*

dateOfBirth invece il giorno di nascita: *14*

Capire questa differenza può essere banale per un essere umano, ma rappresenta un limite per un motore inferenziale che voglia attivare processi di deduzione logica sui dati. I vocabolari inoltre rappresentano le proprietà di un concetto ed i loro elementi devono quindi essere accessibili e comprensibili, sia attraverso una descrizione, sia attraverso un nome che sia significativo e semantico. La ridondanza delle proprietà tra diversi vocabolari inoltre è un problema che va affrontato definendo ed adottando uno standard. La pratica attuale invece è quella di continuare a supportare differenti vocabolari, uno o più per endpoint, in un contesto che si frammenta anno dopo anno sempre di più. Per ovviare a questo problema è stato creato un motore di ricerca per prefissi o vocabolari¹⁸, ma pur semplificando la vita degli sviluppatori, non risolve il problema dell'indeterminatezza degli attributi.

2.7.3 Inconsistenza e contraddizioni

Il maggiore problema dei motori inferenziali e della consistenza di tutto il Web of Data, è l'evitare contraddizioni. Un'informazione contraria ad un'altra può interrompere un processo logico-deduttivo e mandare in errore il sistema. Quale informazione è giusta? Sulla base di cosa?

Un software difficilmente potrà controllare la veridicità delle informazioni e della fonte da cui provengono. I dati sono percepiti ed inseriti da esseri umani, una macchina quindi non può far altro che fidarsi del lavoro della comunità che gestisce il data set. Un valore di *truth* può quindi essere espresso sui concetti forniti, ma non in maniera autonoma dal

17 DBpedia - Predefined Namespace Prefixes - <http://dbpedia.org/sparql?nsdecl>

18 Namespace lookup for RDF developers - <http://prefix.cc/>

2. Rappresentazione della Conoscenza

software, in quanto non ancora in grado di ottenere i dati attraverso l'osservazione diretta dei fenomeni del mondo reale.

Fortunatamente progetti come Wikipedia sono da anni abituati alla costante verifica, revisione e correzione delle informazioni fornite. Il lavoro della comunità, se pubblico, può essere continuamente migliorato con l'obiettivo di mantenere il massimo grado di attendibilità. Questo però implica che i dati debbano necessariamente essere di dominio pubblico, accessibili e di facile correzione per ogni utente che li utilizzi. Nel paragrafo precedente è stato infatti notato come approcci chiusi come quello di Wolfram Research non siano apprezzati dall'utente finale, sempre più sensibile alla correttezza dei dati e che soluzioni quale quella adottata da Freebase, affrontata nel prossimo capitolo, sia in generale preferibile.

2.7.4 Informazioni false

Il termine *Metacrap*, che deriva dall'unione di *metadata* e *crap*, ha cominciato a diffondersi con il saggio di Cory Doctorow del 2001, intitolato "*Metacrap: Putting the torch to seven straw-men of the meta-utopia*"¹⁹. Il documento affronta i sette principali problemi della fondatezza dei metadati:

1. **Le persone mentono:** i metadati vengono usati in un mondo competitivo, popolato da venditori con la necessità di vendere la loro merce, artisti che competono per aumentare l'audience ed in generale utenza in grado di avvelenare il sistema di metadati con informazioni false che dovrebbero avvantaggiarli.
2. **Le persone sono pigre:** mantenere le informazioni non è meno semplice di crearle. Riuscire ad espandere continuamente un data set con nuovi dati e riuscire allo stesso tempo a mantenerli è un'impresa che richiede enormi sforzi ed è sempre soggetta ad errori. Durante il corso degli anni è stato più volte dimostrato che opere di disinformazione siano possibili anche su realtà affermate e diffuse come Wikipedia, dove lo sciacallaggio delle pagine è sempre possibile e viene corretto lentamente in sezioni poco visitate o di scarso interesse.

¹⁹ C.Doctorow - *Metacrap: Putting the torch to seven straw-men of the meta-utopia* - <http://www.well.com/~doctorow/metacrap.htm>

2. Rappresentazione della Conoscenza

3. **Le persone sono stupide:** nella meta-utopia ogni utente deve essere in grado di pubblicare i propri dati in maniera corretta, ma l'esperienza ci porta continuamente di fronte ad errori, anche grossolani, nella classificazione delle informazioni, tra cui errori grammaticali, di punteggiatura, ma anche di semantica.
4. **Conosci te stesso:** le informazioni inserite dagli utenti e che riguardano loro stessi sono spesso difficilmente attendibili. Inoltre è difficile che si riesca ad attribuire un corretto peso alle differenti proprietà dei concetti descritti.
5. **Gli schemi non sono neutrali:** in un contesto ideale, gli esperti di epistemologia discutono a tavolino la mappa delle gerarchie dei concetti e delle loro proprietà, ma nella realtà ogni usufruttore dei metadati può usare la propria notazione e soprattutto creare nuove sub-gerarchie che siano inconsistenti con il resto. Nell'ambito della rappresentazione della conoscenza inoltre, i concetti sono vasti e di difficile classificazione.
6. **La metrica influenza i risultati:** nella valutazione dei concetti, come ad esempio software o prodotti commerciali, avere una metrica comune aiuta ad effettuare i paragoni e a compilare classifiche a seconda delle prestazioni. Come è logico pensare, nella valutazione di un software ad esempio alcuni valori sono spesso intrinsecamente opposti, ad esempio programmi che hanno un alto valore in sicurezza costano di più, oppure alimenti che si conservano per più tempo, sono più dannosi per la salute. Nella realtà della pubblicazione dei metadati però è molto comune la pratica di pubblicare solo i risultati positivi, nascondendo di fatto delle informazioni dannose per chi le pubblica, ma vitali per la fondatezza dei dati.
7. **Esiste più di un modo per descrivere qualcosa:** chi decide cosa è cosa? Un determinato gruppo di utenti può trovarsi di fronte ad cartone animato per bambini, un altro di fronte ad uno show per adulti, ma di fatto osservare la stessa identica cosa. Con la diffusione dei metadati diventa anche obbligatorio mantenere delle gerarchie, classificando ogni singolo concetto, ma se questa classificazione è semplice ed immediata per molte entità, ci sono casi in cui l'interpretazione è differente da persona a persona.

2.7.5 Privacy

Il caso di Facebook per quanto riguarda la privacy è forse quello più eclatante al momento. Nella puntata del 10 Aprile 2011 di Report²⁰ viene mostrato il potere della pubblicazioni annunci della piattaforma, nonché principale fonte di guadagno dell'azienda. In particolare si inizia con il definire il target dell'ipotetico annuncio, filtrando tutte le persone a cui piace la trasmissione Annozero e Presadiretta, poi si continua selezionando in questo insieme solo le donne, successivamente solo le donne che hanno espresso interesse verso persone del loro stesso sesso e per concludere solo quelle che hanno segnalato come posto di lavoro la Rai. Lo strumento online quindi segnala meno di 20 possibili persone per questo tipo di ricerca, dimostrando la capacità di raggiungere risultati estremamente mirati. Se è vero però che i dati sono forniti in maniera anonima, non è escluso che la questione privacy sia così semplicemente risolta. Con lo stesso sistema infatti è possibile identificare le preferenze politiche di un determinato gruppo di persone, basandosi sui loro interessi. Le informazioni sono fornite dagli utenti in un determinato contesto, senza però conoscere il loro utilizzo finale da parte dell'azienda, che trattiene la proprietà dei dati e li vende senza scrupoli per eseguire ricerche di mercato. Il concetto di Linked Data ci porta quindi di fronte ad una tecnologia con un enorme potere, che però l'utenza ancora non concepisce interamente. La protezione dei dati personali forniti dovrà quindi essere punto di interesse nel futuro sviluppo del Web Semantico, la proprietà intellettuale dei dati personali dovrebbe diventare trasparente e soggetta al pieno controllo dell'utente che li inserisce. Il progetto Diaspora²¹, il social network open source e distribuito, ha come obiettivo quello di rendere ogni nodo della rete sociale appartenente all'utente. Nella presentazione del servizio è descritta la possibilità di mantenere un nodo privato, ma connesso alla rete, contenente il proprio profilo e tutte le informazioni ad esso collegato.

20 Stefania Rimini – Il Prodotto Sei Tu – Report (Rai)

21 The Diaspora Project - <http://diasporaproject.org/>

2. Rappresentazione della Conoscenza

Own your data



Host it yourself

Now you don't have to settle for having your data on someone else's server. Since Diaspora is completely free software, you can [grab the code](#) and host it wherever you want. We're constantly making it easier for individuals to host their own pods on Diaspora*.

Fig 2.7.5-1 *Screen parziale della descrizione del progetto Diaspora*

Se quindi Diaspora non è riuscito ad oggi ad ottenere l'interesse degli utenti dei social network, domani attraverso l'uso di standard aperti quali Open Graph e con un occhio verso proprietà dei dati, potrebbe diventare un punto chiave della rete sociale del prossimo futuro.

3. Endpoints

Grazie alla standardizzazione del formato RDF per la rappresentazione dei dati e di SPARQL come linguaggio di query, nel corso degli anni la disponibilità di database si è estesa fino a garantire la pubblicazione di dati, anche tecnici, di vari settori. Dal 2011 al 2012 il numero di endpoints SPARQL disponibili pubblicamente è raddoppiato, arrivando a contare 391 voci, di cui 245 attive a Settembre 2012. Gli argomenti trattati sono molto eterogenei e riguardano informazioni estratte da Wikipedia, database musicali, medici o governativi a seconda dei casi. Verranno citati in lista quelli di maggior rilievo:

Nome	Commento
Bio2RDF	Database biologico gestito da Laval University, Quebec.
DBTune	Database musicale ospitato dal Semantic Web Education and Outreach (SWEO) Interest Group.
DBpedia	In DBpedia sono contenute le informazioni estratte da un parsing semantico di Wikipedia e mantenute dalla comunità.
DailyMed	Un database di natura medica nel quale sono descritte le composizioni chimiche e le interazioni tra medicinali.
data.gov.uk	Progetto del governo britannico per rendere più accessibili i dati al servizio del cittadino.
lobid.org	Linking open bibliographic data. In lobid vengono conservate informazioni di carattere bibliografico.

3. Endpoints

DBLP Bibliography Database	Il database DBLP fornisce informazioni bibliografiche sulle principali riviste di informatica e conferenze.
World Factbook	Contiene le informazioni del CIA Factbook espresse in formato semantico.
Revyu	Revyu.com è un aggregatore di link e metatag associati a recensioni di qualsiasi natura.
LinkedMDB	Linked Movie Data Base è un database semantico che contiene informazioni riguardanti film, attori, registi, serie televisive e tutto ciò che riguarda il settore del cinema.

3.1 DBpedia

DBpedia è un progetto che mira all'estrazione di contenuti strutturati dalle pagine di Wikipedia. Il data set è riempito utilizzando le infoboxes di Wikipedia e sfruttando i template pre-esistenti per determinate categorie di pagine. Il primo approccio infatti prevedeva l'utilizzo di un parser che estraesse automaticamente le proprietà utilizzate in Wikipedia e che creasse le proprietà del namespace <http://dbpedia.org/property/> in base ai nomi utilizzati nelle infoboxes. Tuttavia queste proprietà non seguivano una specifica gerarchia ed i dati estratti in questo modo risultavano molto *disturbati*.

Con il rilascio della versione 3.2 di DBpedia, è stata introdotta un'ontologia creata ad hoc, per coprire le classi ed i template di Wikipedia. I dati sono rappresentati utilizzando proprietà che possano essere utilizzate in entità di tipologie differenti, di modo da ridurre la ridondanza. Questa soluzione è molto più pulita e strutturata della precedente, ma non copre tutte le proprietà ed i tipi di infoboxes di Wikipedia.

3. Endpoints

Con il rilascio della versione 3.5, sono quindi stati adottati tre data set differenti:

- **Ontology Infobox Types:** Questo dataset contiene i `rdf:types` delle istanze che sono state estratte dalle infoboxes
- **Ontology Infobox Properties:** Dataset contenente i valori estratti dalle infoboxes. Queste proprietà possono essere uguali per entità differenti, come ad esempio la proprietà `volume` può essere utilizzata nell'ambito della descrizione di un lago o di un pianeta. L'approccio è utile per evitare inutili ridondanze, ma risulta impossibile poi risalire al tipo di entità partendo dalla proprietà.
- **Ontology Infobox Properties (Specific):** Questo dataset contiene le proprietà che sono state specializzate per l'uso in una specifica classe. Ad esempio la proprietà `height` è specializzata nella classe `Person` per esprimere il valore in centimetri, piuttosto che in metri.

Attualmente l'ontologia adottata copre 359 classi che formano una gerarchia descritta da 1775 differenti proprietà. Inoltre, con la versione 3.7 l'ontologia è un grafo aciclico diretto e non più rappresentata ad albero. Contiene ad oggi 2,350,000 istanze, di cui:

Luoghi	573000
Persone	764000
Lavori	333000
Specie	202000
Organizzazioni	192000

La copertura può sembrare ottima, ma le statistiche²² indicano che solo il 4,35% di tutti i template della versione inglese di Wikipedia è stato mappato. Wikipedia di fatto rappresenta uno dei primi tentativi di portare sul Web dei dati in una forma strutturata. I template sono stati adottati da anni, tuttavia la comunità continua ad ampliarli, modificarli e ad usarli in modi differenti, anche su pagine che rappresentano una stessa classe di concetti. Questo porta a presupporre sia che gli utenti si sentano limitati dal dover descrivere la realtà utilizzando modelli pre-confezionati, sia che determinati gruppi descrivano le stesse cose in modi diversi.

22 DBpedia Mapping Statistics - <http://mappings.dbpedia.org/server/statistics/en/>

3. Endpoints

Anche in DBpedia il problema è sentito, in quanto gli utenti possono inserire nuove proprietà all'ontologia, ma le convenzioni per farlo sono solo consigliate e non specifiche. Non è difficile quindi trovare nel vocabolario proprietà con nomi ambigui, prive di qualsiasi commento e per le quali non sia definito un range di valori o un dominio di utilizzo.

Il progetto DBpedia in ogni caso ha portato alla nascita di strumenti interessanti e precursori al Web Semantico. Il database è liberamente accessibile ed interrogabile tramite query SPARQL, ma anche visitabile tramite browser in formato HTML. Tra le diverse applicazioni possiamo trovare tool di estrazione delle informazioni, che collegano le parole di un testo alle entità relative di DBpedia per una maggiore comprensione del testo, oppure interfacce per la creazione facilitata di query SPARQL, ma anche strumenti grafici per l'immediata consultazione del database. Nello specifico andremo ad analizzare un faceted browser ed un relation finder.

3. Endpoints

3.1.1 Faceted Wikipedia Search

Il Faceted Browser presentato nel progetto DBpedia implementa la tecnica della navigazione sfaccettata, per la consultazione dei dati strutturati estratti dalla versione inglese di Wikipedia. Attraverso questo strumento è quindi possibile eseguire ricerche filtrando i contenuti proposti.

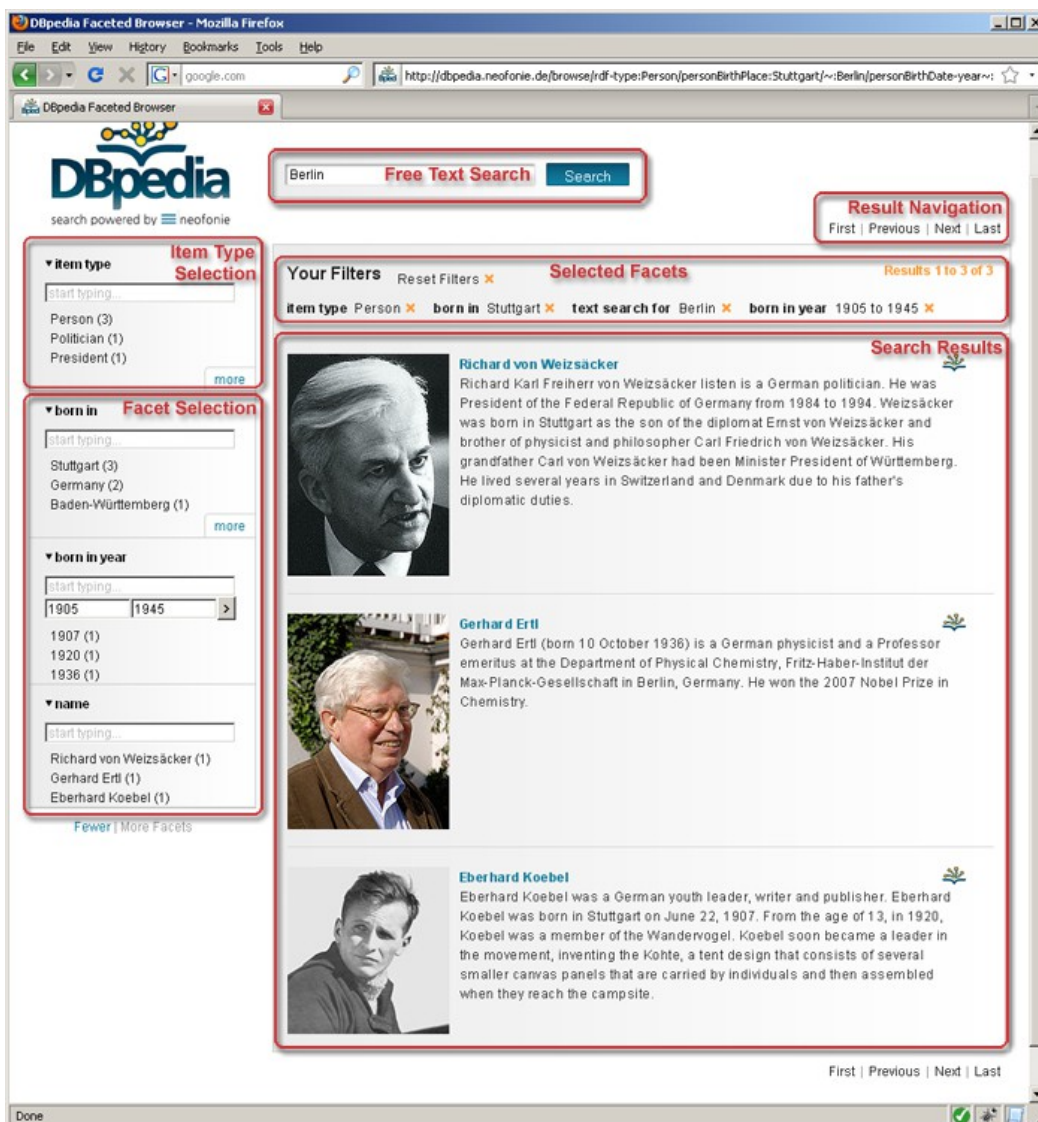


Fig. 3.1.1-1 Screenshot del Wikipedia Faceted Browser

3. Endpoints

L'interfaccia è composta di quattro elementi fondamentali:

1. **Campo ricerca:** è possibile eseguire ricerche di testo libero per ricavare i risultati iniziali.
2. **Selezione sfaccettata e selezione del tipo di entità:** attraverso un menu è possibile filtrare i risultati della ricerca in base alle proprietà più comuni fra essi. Si possono quindi selezionare solo determinate classi di entità, come definire un range di ricerca su particolari proprietà.
3. **Filtri attivi:** i filtri attivati sulla ricerca sono sempre presenti e consultabili dall'utente, che può decidere di disattivarli per procedere ad un percorso di personalizzazione differente dei risultati. Sono indipendenti l'uno dall'altro.
4. **Risultati di ricerca:** i risultati contengono un piccolo abstract per ogni entità e rimandano alle pagine relative di Wikipedia.

Lo strumento permette di creare quindi query composte che altrimenti sarebbe difficile fare. Il suo utilizzo è piuttosto immediato, ma non è sempre detto che una stessa classe di entità abbia proprietà simili sulle quali poter attivare dei filtri, questo poiché il problema della disambiguazione dei dati coinvolge anche i risultati di ricerca del Faceted Browser, che fonda la sua implementazione su template pre-definiti e su di una forma fortemente strutturata.

3. Endpoints

3.1.2 Relation Finder

RelFinder è un'applicazione scritta utilizzando il framework gratuito Adobe Flex. L'applicazione è in grado di estrapolare e visualizzare le relazioni tra più oggetti rappresentati in formato RDF presenti su data set che forniscono un accesso standardizzato SPARQL.

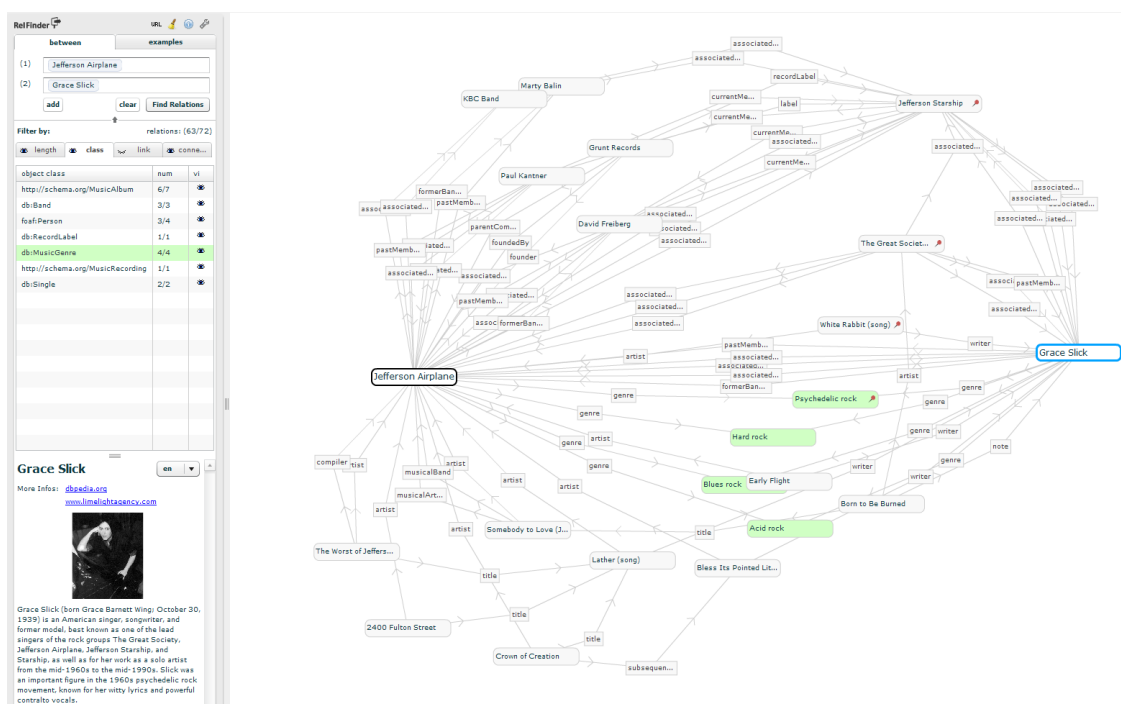


Fig. 3.1.2-1 Screenshot di RelFinder dopo una query

L'interfaccia è composta di quattro parti:

- 1. Campo ricerca:** inizialmente è necessario introdurre due o più oggetti appartenenti ai database, dei quali si vuole trovare l'insieme di relazioni.
- 2. Filtri:** nella sezione dedicata ai filtri attivabili sul risultato, è possibile selezionare solo relazioni che coinvolgono determinate classi (db: MusicGenre piuttosto che foaf: Person), oppure tipologie di collegamenti, quali bandMember, founder, genre, label, ecc.

3. Endpoints

3. **Riquadro informativo:** è possibile selezionare un'entità presente nel grafo per visualizzarne tutte le informazioni disponibili in varie lingue, per avere una rapida visione di cosa il nodo relativo rappresenti. Inoltre è presente un link alla pagina relativa di DBpedia.
4. **Grafo risultante:** i risultati sono presentati sotto forma di grafo diretto modellabile dall'utente.

Così come per il Faceted Browser, anche questo strumento risulta di difficile utilizzo. Diventa necessario infatti escludere buona parte dei risultati per avere una corretta visibilità del grafo risultante e molte delle proprietà che connettono i nodi non sono chiare all'utente. Questo è ovviamente un problema dettato dalla semantica dei vocabolari, ma ciò non toglie che l'interfaccia grafica rimanga di difficile utilizzo per un utente casuale.

3.2 Freebase

Freebase è un database di conoscenza mantenuto e riempito dalla comunità. Le informazioni sono rappresentate principalmente sotto forma di metadati, i quali rendono il database una collezione fortemente strutturata di informazioni estratte da altre fonti o aggiunte personalmente dagli utenti in un contesto collaborativo simile a quello utilizzato da Wikipedia. Metaweb Technologies, l'azienda software che sviluppa il progetto sin dal 2007, è stata acquisita da Google a Luglio 2010. Freebase ad oggi fornisce la principale base di informazioni per il modello Knowledge Graph visto nel capitolo precedente. I dati sono gratuitamente disponibili per usi commerciali e non, ma sono comunque rilasciati sotto licenza appartenente a Metaweb Technologies. Il codice sorgente invece è proprietario e non disponibile pubblicamente.

3.2.1 Organizzazione dei contenuti

Il modello Freebase prevede che ogni soggetto descritto nel database (Topic), abbia particolari proprietà (Properties) e sia appartenente a determinati tipi (Types), i quali fanno parte di specifici domini (Domains). Ad esempio nel soggetto Rita Levi-Montalcini possono essere inclusi vari tipi, per descriverla come senatrice, neurobiologa o ricercatrice. Ogni Type rappresenta soggetti che condividono le stesse particolari Properties. Ad esempio ogni soggetto di tipo città condividerà la proprietà sindaco o numero di abitanti e così via. I soggetti possono essere inseriti in Topic, i quali automaticamente attribuiscono ad esso un set prefinito di proprietà. I Topic sostanzialmente emulano il modello dei Template usato da Wikipedia. È importante però considerare che seppure gli utenti possano inserire nuovi tipi e proprietà, questi faranno parte del common pool solo e soltanto dopo essere stati approvati da un impiegato di Metaweb.

3.2.2 Tutto è oggetto

La particolarità di Freebase consiste nel fatto che ogni entità descritta nel database è prima di tutto un `/type/object`. Ogni object ha quindi un type che lo caratterizza. Un'altra feature interessante è l'utilizzo di `expected_types`, ovvero di sapere a priori quale tipo di entità ci si debba aspettare prima dell'esecuzione di una query. Per un essere umano è scontato aspettarsi un oggetto di tipo `location` a fronte di una richiesta sulla proprietà `place_of_birth` di un soggetto, ma per un motore inferenziale questa informazione è importante, poiché non solo aiuta a migliorare l'accuratezza delle query per proprietà più complesse, ma migliora anche la comprensione delle stesse poiché a priori si conosce quale tipo di risposta verrà fornita.

3. Endpoints

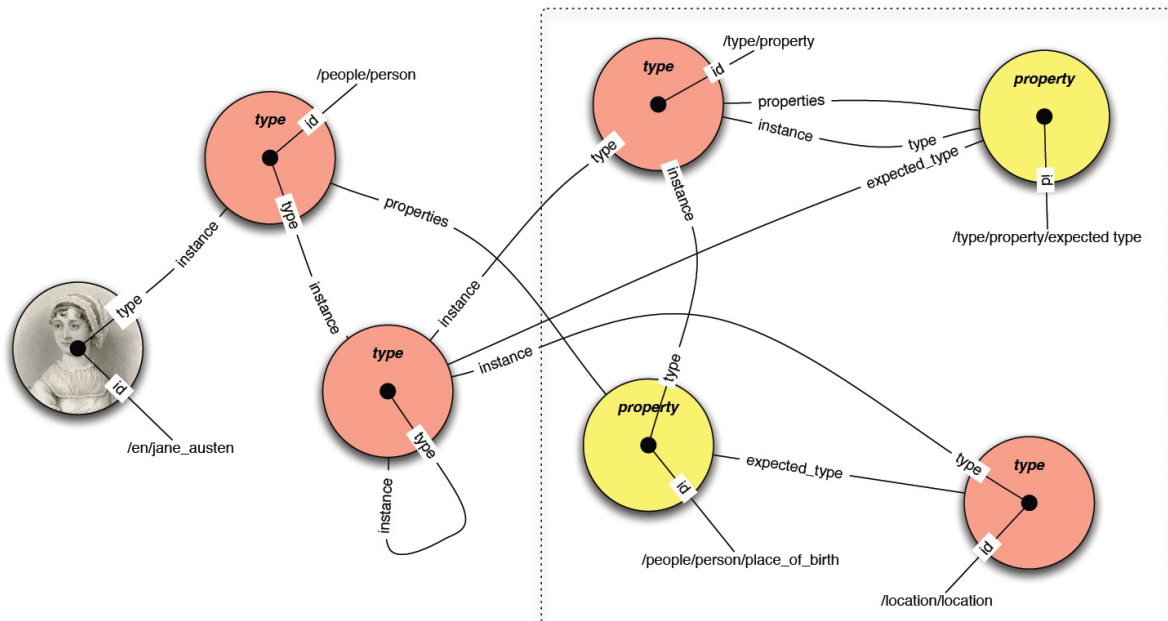


Fig. 3.2.2-1 Freebase Design²³

3.2.3 Metaweb Query Language

MQL nasce come linguaggio di query all'interno del progetto Freebase e rimane ad oggi lo strumento ufficiale per accedere alle informazioni sul database. La sintassi usata è quella del formato JSON²⁴. Vediamo ad esempio la query per risalire ai film diretti da Sofia Coppola:

```
{
  "type" : "/film/director",
  "name" : "Sofia Coppola",
  "film" : []
}
```

²³ *Querying Freebase: Get More From MQL* – Jamie Taylor (May 2011)

²⁴ JSON - RFC 4627

3. Endpoints

Freebase risponderà con un oggetto JSON del tipo:

```
{
  "type" : "/film/director",
  "name" : "Sofia Coppola",
  "film" : [
    "Lick the Star",
    "Lost in Translation",
    "Marie Antoinette",
    "The Virgin Suicides",
    "Somewhere"
  ]
}
```

Sostanzialmente quindi per formulare una query è necessario specificare le informazioni che già si conoscono "type" : "/film/director" e "name" : "Sofia Coppola" nel nostro caso, lasciando senza valori le proprietà che desideriamo ricevere in risposta: "film" : []. Le parentesi quadre di fatto rappresentano un array che Freebase andrà a riempire. La richiesta di valori si può eseguire con quattro espressioni:

- **null**: Se la proprietà contiene un valore, restituisce quel valore. In alternativa viene restituito l'id dell'oggetto.
- **[]**: Come visto nell'esempio, viene restituito un array di valori. Se la proprietà indica un oggetto, come per **null** verrà restituito l'array contenente gli id degli oggetti restituiti.
- **{}**: Se la proprietà contiene un valore, restituisce l'oggetto che rappresenta quel valore. Questo oggetto avrà a sua volta un tipo e delle proprietà. Se la proprietà invece contiene un oggetto, viene restituito l'oggetto comprensivo di id, nome e proprietà.
- **[{}]**: Esattamente come **{}**, ma restituisce un array di oggetti piuttosto che uno singolo.

3.2.4 Futuro e monetizzazione

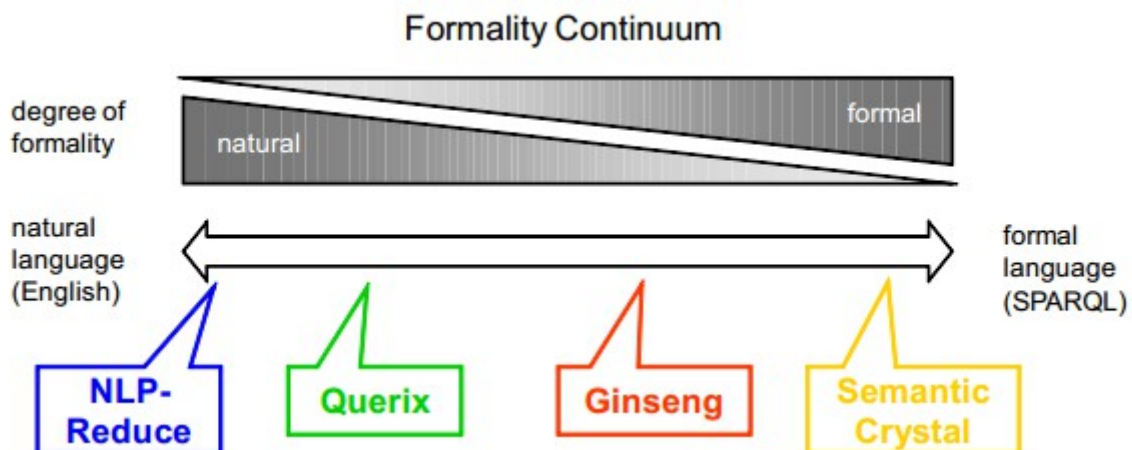
Orientato ai metatag e fortemente strutturato, Freebase è attualmente uno dei database semantici meglio documentati e di facile utilizzo. Nel 2012 sono stati oltrepassati i 20 milioni di Topic e la comunità ogni giorno aggiunge o migliora le voci contenute. Le informazioni sono di libero accesso, sia per attività commerciali che non, ma l'accesso è limitato dall'utilizzo di API proprietarie e dal linguaggio MQL. Il modello di pubblicazione si discosta quindi dalla mentalità open canonica e porta alla nascita di dubbi per quanto riguarda l'effettiva proprietà intellettuale delle voci registrate. I dati appartengono alla comunità? Ad essa sarà sempre garantito il libero accesso? Inoltre non esiste un vero sistema democratico all'interno della gestione dell'ontologia, ma sono gli stessi impiegati di Metaweb a decidere quali proprietà o schemi siano migliori di altri, utilizzando privilegi di cui l'utenza non dispone. Infine essendo il motore basato su codice proprietario, la comunità è impossibilitata a creare un mirror del database su piattaforma interamente libera su modello Wikipedia.

Nonostante le ambiguità però il database è largamente utilizzato data la facilità di utilizzo, l'ottimo standard e la chiara documentazione tecnica. L'uso ad oggi è gratuito, ma si prevede che in futuro possa essere implementato un sistema di advertising e probabilmente un abbonamento per l'accesso alle informazioni prive di pubblicità. Il futuro di Freebase sarà determinato dalle scelte etiche e commerciali, poiché dipende quasi esclusivamente dalla comunità che contribuisce ad arricchirlo, ma che potrebbe decidere di non contribuire più, sia in favore di soluzioni ugualmente efficienti e più libere, sia per via di una cattiva gestione o di pessime scelte riguardanti la proprietà intellettuale delle informazioni inserite.

4. Nuove interfacce: Linguaggio naturale

Il concetto di Web of Data porta al bisogno di nuove interfacce, più complesse e necessariamente più intuitive di quelle fino ad oggi adottate. Eseguire query composte in Google non è semplice con il solo utilizzo informale di parole chiave e soluzioni come il Faceted Browser di DBpedia visto nel capitolo precedente risultano macchinose e di lento accesso per gli utenti. Considerando però i progressi nell'ambito del riconoscimento vocale e della linguistica, ci si può concentrare sull'utilizzo del linguaggio naturale, esprimibile con le regole formali della grammatica e possibilmente comprensibile anche da una macchina che abbia una base dati semantica e che quindi riesca ad associare ad ogni parola il corretto significato.

Nella ricerca condotta da Esther Kaufmann ed Abraham Bernstein dell'Università di Zurigo, Svizzera, si è valutata l'usabilità per gli utenti di interfacce di linguaggio naturale. Nell'esperimento sono stati usati quattro engine di ricerca semantica che utilizzano la lingua naturale in formati più o meno strutturati.



4. Nuove interfacce: Linguaggio naturale

Come possiamo vedere, in ordine di formalità troviamo NLP-reduce, che utilizza domande totalmente non strutturate, che ricordano la ricerca attraverso parole chiave comune ai tradizionali motori di ricerca, Querix che invece necessita di domande espresse in lingua inglese corretta e successivamente Ginseng e Semantic Crystal, che limitano intrinsecamente le possibilità dell'utente in modo da ridurre gli errori della query generata. Le quattro differenti soluzioni sono state sottoposte ad un campione eterogeneo di utenti, ai quali è stato chiesto di rispondere ad un insieme di quattro domande in un tempo limite, utilizzando una dopo l'altra le quattro interfacce, per poi fornire un feedback sulla complessità di utilizzo degli strumenti, sulla fiducia dei risultati forniti e sulla immediatezza delle interfacce di input utilizzate.

	tempo medio per le 4 domande	tempo medio per una domanda	media di query sottoposte	tasso medio di successo	tasso medio di fallimento
NLP-Reduce	2 min 39 sec	23.54 sec	7.94	69.27 %	30.73 %
Querix	4 min 11 sec	29.31 sec	7.75	77.08 %	22.92 %
Ginseng	6 min 06 sec	34.82 sec	11.06	63.54 %	36.46 %
Semantic Crystal	9 min 43 sec	89.53 sec	7.02	54.86 %	45.14 %

I risultati, contrariamente a quanto si possa pensare, vedono Querix avere più successo, con Semantic Crystal all'ultimo posto, seppure utilizzi un formalismo tale da evitare errori nell'interpretazione della query. Per quanto riguarda invece la costruzione delle query, come ci si poteva aspettare, si nota che più il linguaggio è formale, più gli utenti impiegano tempo. Dall'analisi dei log di utilizzo, risulta addirittura che alcuni utenti di Semantic Crystal abbiano iniziato a costruire la query, ma l'abbiano abbandonata in quanto troppo complessa da creare. Il tasso molto basso di successo infatti deriva non dall'imprecisione dell'engine, ma dallo scorretto uso da parte degli utenti.

	risultato medio di usabilità	interfaccia apprezzata di più	interfaccia apprezzata di meno	linguaggio apprezzato di più	linguaggio apprezzato di meno
NLP-Reduce	56.72	12.50 %	25.00 %	18.75 %	25.00 %
Querix	75.73	66.67 %	2.08 %	60.42 %	4.17 %
Ginseng	55.10	6.25 %	12.50 %	16.67 %	12.50 %
Semantic Crystal	36.09	14.58 %	60.42 %	4.17 %	58.33 %

4. Nuove interfacce: Linguaggio naturale

Per quanto riguarda l'usabilità, è stata adottata la *System Usability Scale*²⁵, sviluppata da Jogh Brooke e che misura principalmente i seguenti aspetti:

- **Efficacia:** gli utenti riescono a raggiungere i loro obiettivi?
- **Efficienza:** quante risorse vengono utilizzate dagli utenti per raggiungere quegli obiettivi?
- **Soddisfazione:** l'esperienza è stata soddisfacente?

I valori della scala vanno da 0 a 100.

I risultati dimostrano che Querix è lo strumento più apprezzato, ma in generale che gli utenti siano più a loro agio sottoponendo domande nel formalismo che utilizzano quotidianamente per comunicare nel mondo, ovvero la loro lingua. È interessante anche notare come questa soluzione sia preferibile addirittura a formalismi meno strutturati come quelli adottati da NLP-Reduce, in quanto la pratica di individuare parole chiave per query composte non fa parte del linguaggio o in generale del nostro modo di esprimerci e porre domande ad altre persone. Grazie ai commenti degli utenti inoltre si nota come Querix sia percepito come maggiormente affidabile rispetto ai concorrenti; nonostante gli altri motori rispondano fornendo una lista di entità alla domanda "*How many rivers run through Colorado?*", Querix risponde semplicemente "*There are 10*", ma la risposta in linguaggio naturale è ritenuta più attendibile in quanto si percepisce che lo strumento abbia *inteso* la domanda fornita.

Il rapporto tra gli strumenti in grado di rendere usabile il Web Semantico e gli utenti, dovrà quindi essere di fiducia ed immediato. La ricerca dimostra che è possibile costruire motori semantici in grado di interpretare il linguaggio naturale con un buon grado di successo, ma soprattutto che gli utenti lo preferiscono in quanto non necessita di imparare nuove nozioni e ci si possono costruire con semplicità ed efficacia query composte.

Di seguito analizzeremo più in profondità le quattro soluzioni proposte.

25 J. Brooke - *SUS - A quick and dirty usability scale*

4.1 NLP-Reduce

NLP-Reduce è una interfaccia di linguaggio naturale per l'interrogazione di ontologie, indipendente dal dominio di ricerca. L'engine è definito *naïve* in quanto l'approccio alla risoluzione è semplice e processa la domanda fornita in linguaggio naturale come una serie scollegata di parole. Il procedimento permette quindi query simili a quelle comunemente usate su Google e composte solamente da parole chiave. NLP-Reduce è il meno formale tra le quattro soluzioni presentate. L'interfaccia utente permette di inserire domande complete, parti di frase o anche una serie di parole chiave. Prendiamo il caso della query "Chinese restaurants in San Francisco".

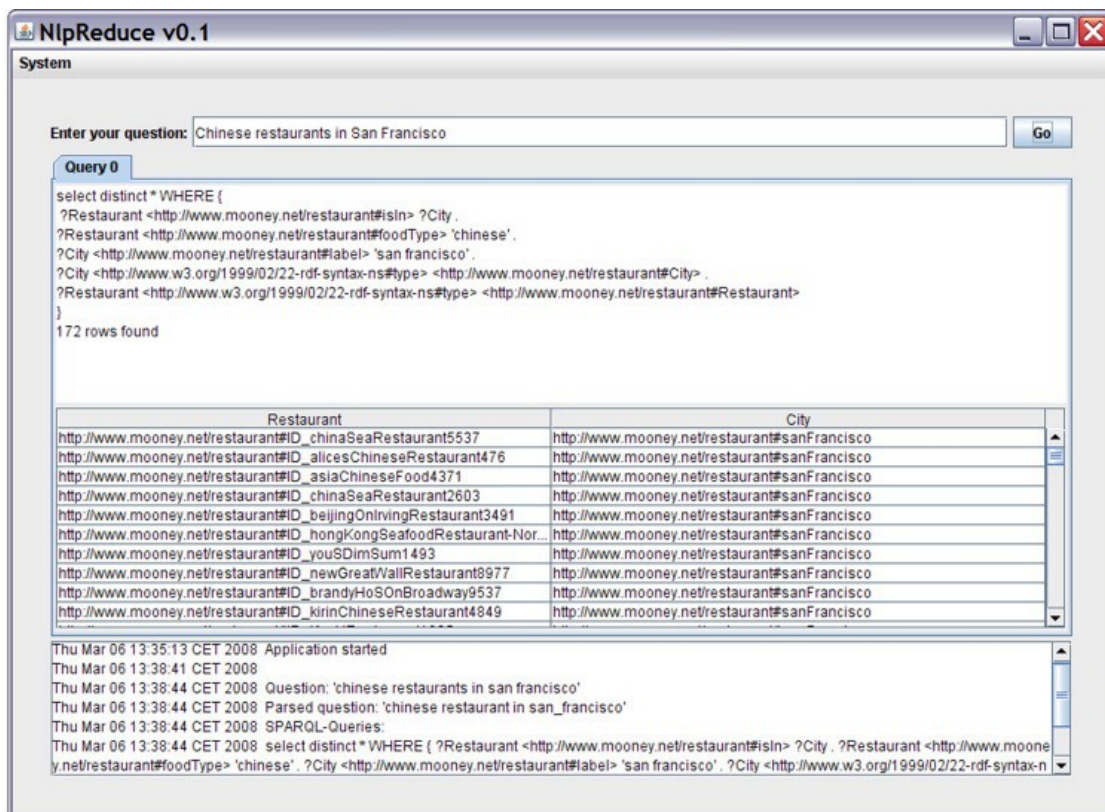


Fig 4.1-1 Finestra principale di NlpReduce

4. Nuove interfacce: Linguaggio naturale

Inizialmente il sistema rimuove dall'input i caratteri speciali, come virgole e punti interrogativi. Le parole poi vengono ridotte alla radice utilizzando una tecnica di *stemming* e vengono passate al modulo Query Generator, componente chiave di NLP-Reduce. Il modulo inizialmente ricerca triple nelle quali siano comprese le parole in input espresse come object, oppure sinonimi sulla base del dizionario semantico WordNet. Nel caso preso ad esempio verranno identificate triple contenenti le proprietà <isIn> o <isInCity> poiché contengono la parola chiave "in" espressa nella query. Successivamente le triple vengono valutate tramite un sistema di ranking ed in secondo luogo vengono identificate le proprietà all'interno della base dati, che possano essere associate alle triple coinvolte nel passo precedente. In questo caso la parola "Chinese" è contenuta in triple contenenti la proprietà <foodType>, restituita come risultato. L'ultima operazione, prevede l'associazione di tutte le altre parole rimaste della domanda in input, con proprietà che si possano combinare. Vengono quindi fornite le triple [<Restaurant> <isIn> 'sanFrancisco'] e [<Restaurant> <foodType> 'chinese']. Quando non rimangono più parole da associare, il sistema prova ad eseguire la query SPARQL generata per mostrare un risultato.

4.2 Querix

Simile a NLP-Reduce, Querix è un'interfaccia alla query di ontologie, indipendente dal dominio di ricerca e basata sulla corrispondenza di patterns. Questa ultima proprietà implica la necessità di una struttura ben precisa delle domande in input ed il formato adottato è quello della lingua inglese, comprensiva di regole grammaticali e di sintassi. A differenza di NLP-Reduce però Querix non risolve autonomamente le ambiguità con un sistema di ranking, ma utilizza una interfaccia apposita per lasciare all'utente la possibilità di chiarire la propria query.

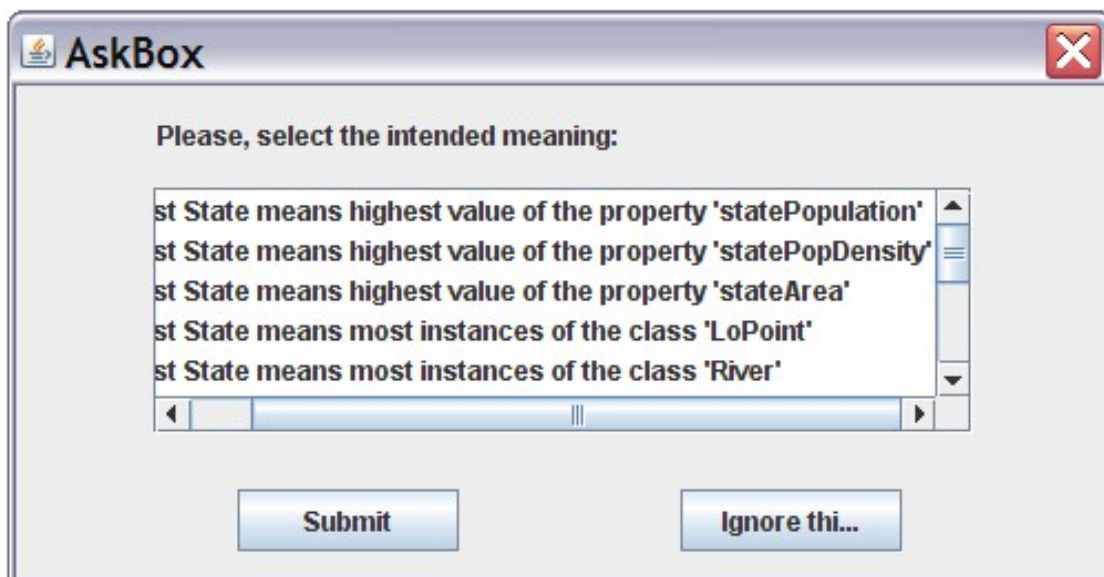


Fig 4.2-1 Form di disambiguazione di Querix

Il sistema inoltre non prevede il riconoscimento automatico del dominio di ricerca, ma è anch'esso delegato all'utente che lo deve esplicitare prima di sottoporre la propria domanda. Inoltre è richiesto che ogni domanda inizi con "Which...", "What ...", "How many ...", "How much ...", "Give me ..." o "Does ...".

4. Nuove interfacce: Linguaggio naturale

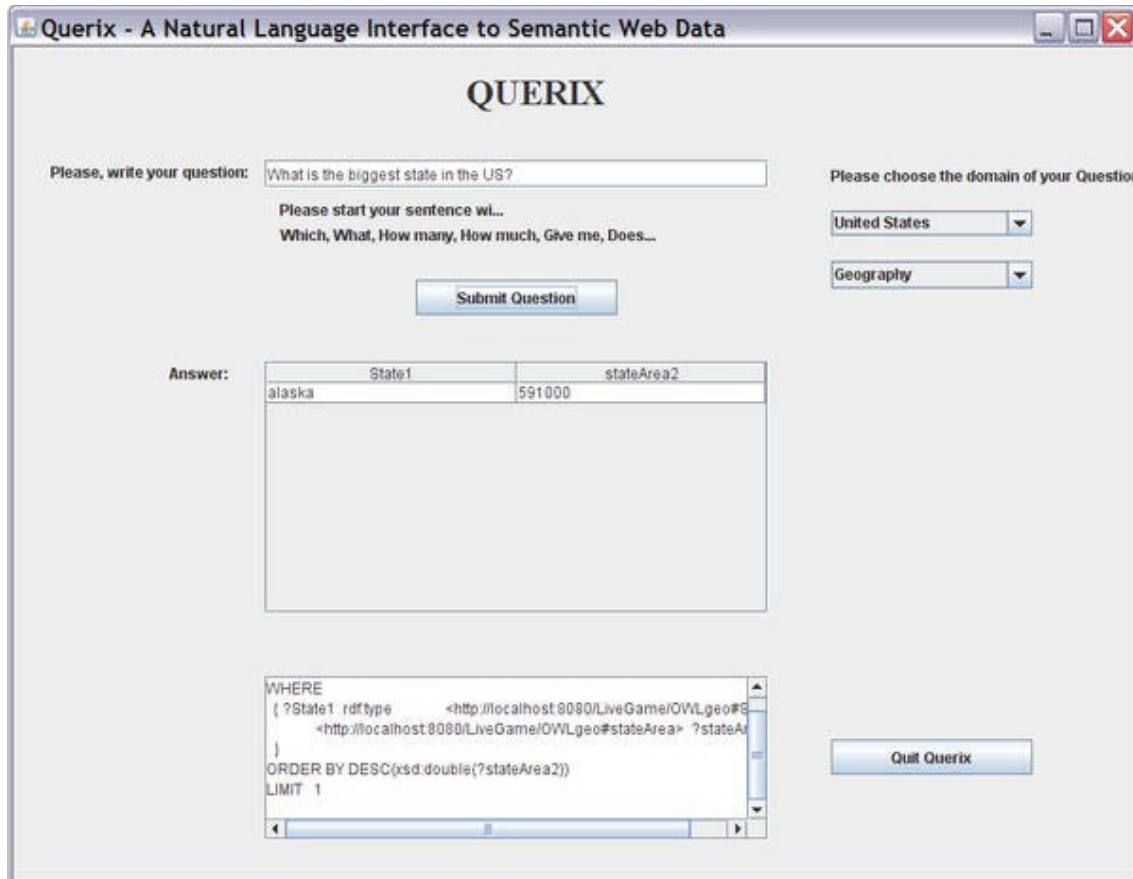


Fig. 4.2-2 Finestra principale di Querix

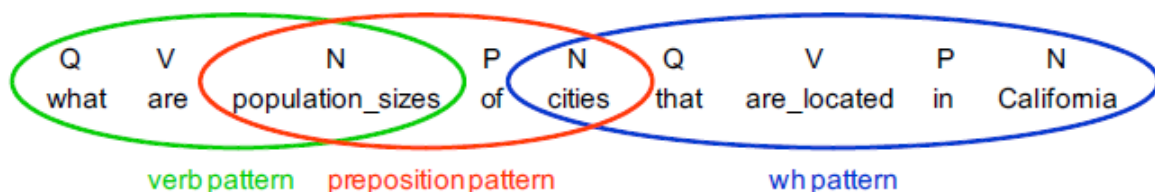
Dato l'utilizzo di un linguaggio formale, Querix integra fra i componenti il Stanford Parser che inizialmente fornisce l'albero della sintassi della domanda di linguaggio naturale. Il modulo *Query Analyzer* utilizza i dati del parser ed estrae una sequenza di categorie *Noun* (N), *Verb* (V), *Preposition* (P), *Wh-Word* (Q) e *Conjunction* (C) con le quali successivamente generare lo scheletro della query. Per esempio, presa la domanda "What are the population sizes of cities that are located in California?", lo scheletro risultante è Q-V-N-P-N-Q-V-P-N. Il secondo componente del modulo è WordNet, che come visto in NLP-Reduce, restituisce i sinonimi per tutti i verbi ed i sostantivi dell'albero generato dal parser.

4. Nuove interfacce: Linguaggio naturale

Q wha t	V are	N population sizes	P of	N cities	Q that	V are located	P in	N California
	be exist	inhabitant citizen number magnitude measurement		town metropolis urban center municipal		situate place site settle		CA Golden State

Secondariamente interviene il modulo Matching Center, chiave di Querix, che cerca una corrispondenza tra lo scheletro generato e le triple dell'ontologia. Per ogni query vengono eseguiti tre passi:

1. Per prima cosa il modulo prova ad associare lo scheletro della query con un piccolo insieme di pattern euristici, ad esempio Q-V-N che rappresenta "what are the population sizes" e N-P-N che rappresenta "population sizes of cities". Questa corrispondenza serve per identificare modelli di tipo soggetto-proprietà-oggetto presenti nella domanda.
2. Successivamente vengono utilizzati i sinonimi forniti da WordNet per migliorare i risultati e permettere all'utente di non essere limitato alla scelta di parole specifiche e limitate al proprio vocabolario.
3. Infine si cerca una corrispondenza tra le triple trovate al punto due e le triple che corrispondano ai patterns identificati nel punto uno. Ulteriormente i risultati vengono filtrati a seconda del dominio espresso all'inserimento della query.



L'ultimo modulo, il *Query Generator*, costruisce la query SPARQL ed utilizzando Jena la sottopone al database per cercare una risposta.

4.3 Ginseng

Ginseng è un'interfaccia di linguaggio naturale guidata attraverso menu. Il nome è appunto l'acronimo di "*a guided input natural language search engine (for the Semantic Web)*". La differenza con le soluzioni viste precedentemente è che Ginseng non utilizza un vocabolario predefinito e non ha bisogno di interpretare le query. Tutte le ontologie sono conservate nella base dati, il vocabolario è chiuso e deve necessariamente essere seguito. L'approccio sicuramente limita le possibilità dell'utente, ma permette la costruzione di query alle quali sia sempre possibile rispondere, inoltre si tenga in considerazione che con l'aumentare delle ontologie disponibili, anche il vocabolario utilizzato è destinato ad aumentare e fornire più possibilità.

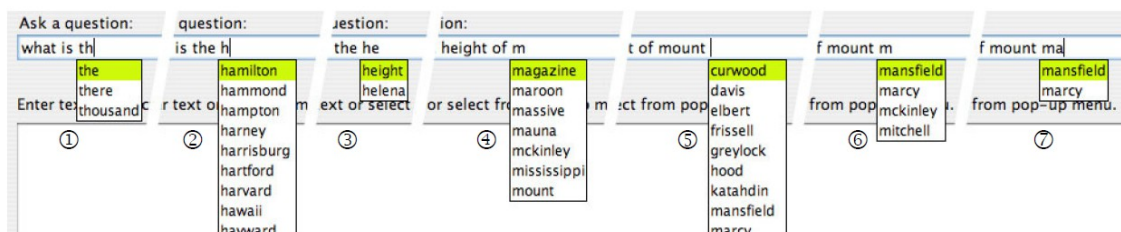


Fig 4.3-1 Ginseng: form di costruzione della query

Sul lato tecnico, Ginseng è composto da quattro moduli fondamentali: un compilatore grammaticale, una grammatica multi-livello generata in maniera parzialmente dinamica, un parser incrementale ed un livello di accesso alle ontologie. L'ultimo modulo è di fatto ARQ, lo SPARQL engine del framework Jena. All'avvio, Ginseng carica in memoria le knowledge base. Per ognuna di esse il Grammar Compiler genera le regole grammaticali necessarie per estendere la parte di grammatica statica utilizzata dall'engine, che contiene regole generiche ed indipendenti dall'ontologia. La grammatica multi-livello è poi utilizzata dal parser in due modi:

1. Inizialmente è specificato un insieme di domande analizzabili, che verranno proposte dinamicamente all'utente in forma di alternative.
2. Secondariamente definisce le regole con cui le query SPARQL devono essere generate.

4. Nuove interfacce: Linguaggio naturale

Il parser incrementale quindi mantiene in memoria una struttura che rappresenta tutte le possibili alternative generate dall'input fornito dall'utente runtime. Questa operazione è essenziale per poter fornire una o più possibili continuazioni della domanda al momento della sua formazione, in modo da guidare l'utente a costruire query sempre analizzabili dall'engine.

```
(1) <START> ::= <OQ> ?
                | SELECT <<OQ>>
                | WHERE { <<OQ>> }

(2a) <OQ> ::= which <subject> <verb>
                | <<subject>>
                | <<subject:1>> <<verb>>

(2b) <OQ> ::= what <subject> <verb>
                | <<subject>>
                | <<subject:1>> ) ( <<subject:1>> <<verb>>

(3) <subject> ::= state
                | ?state
                | <rdf:type> <geo:state> ( type=[<geo:state>] )

(4) <verb> ::= borders <object>
                | -
                | <geo:borders> <<object>> ( domain=[<geo:state>],
                | range=[<geo:state>] )

(5) <object> ::= new york city
                | ?newyorkcity
                | <geo:newYorkCity> ( type=[<geo:city>, <geo:capital>] )

(6) <object> ::=| mississippi
                | ?mississippi
                | <geo:mississippi> ( type=[<geo:river>] )

(7) <object> ::= mississippi
                | ?mississippi
                | <geo:mississippi> ( type=[<geo:state>] )
```

Ginseng quindi rappresenta uno strumento semplice, ma molto potente per la sua capacità di costruire query corrette. La sua implementazione lo rende fortemente legato all'attuale interfaccia, composta di mouse, tastiera e grafica, precludendo la possibilità di estenderlo con un processo di riconoscimento vocale. Questo però non esclude che possa rimanere un ottimo strumento di debug e che i principi alla base dell'engine non possano essere utilizzati per risolvere in maniera semplice ed esplicita ambiguità, per permettere ad un ipotetico motore inferenziale di imparare nuovi pattern di interrogazione della propria base dati.

4.4 Semantic Crystal

Semantic Crystal rappresenta l'interfaccia maggiormente strutturata e restrittiva. Utilizza il linguaggio formale SPARQL, ma ne semplifica l'uso per gli utenti attraverso una finestra grafica in cui è possibile generare grafi. Inizialmente infatti si è pensato all'utilizzo di QGraph, un linguaggio visuale per l'interrogazione e la modifica di database basati su grafi. Tuttavia la soluzione è stata valutata troppo difficile per l'utilizzo da parte di utenti finali ed è stata semplificata adottando un linguaggio simile ad InfoCrystal e maggiormente incentrato sulla rappresentazione grafica, in quanto maggiormente comprensibile.

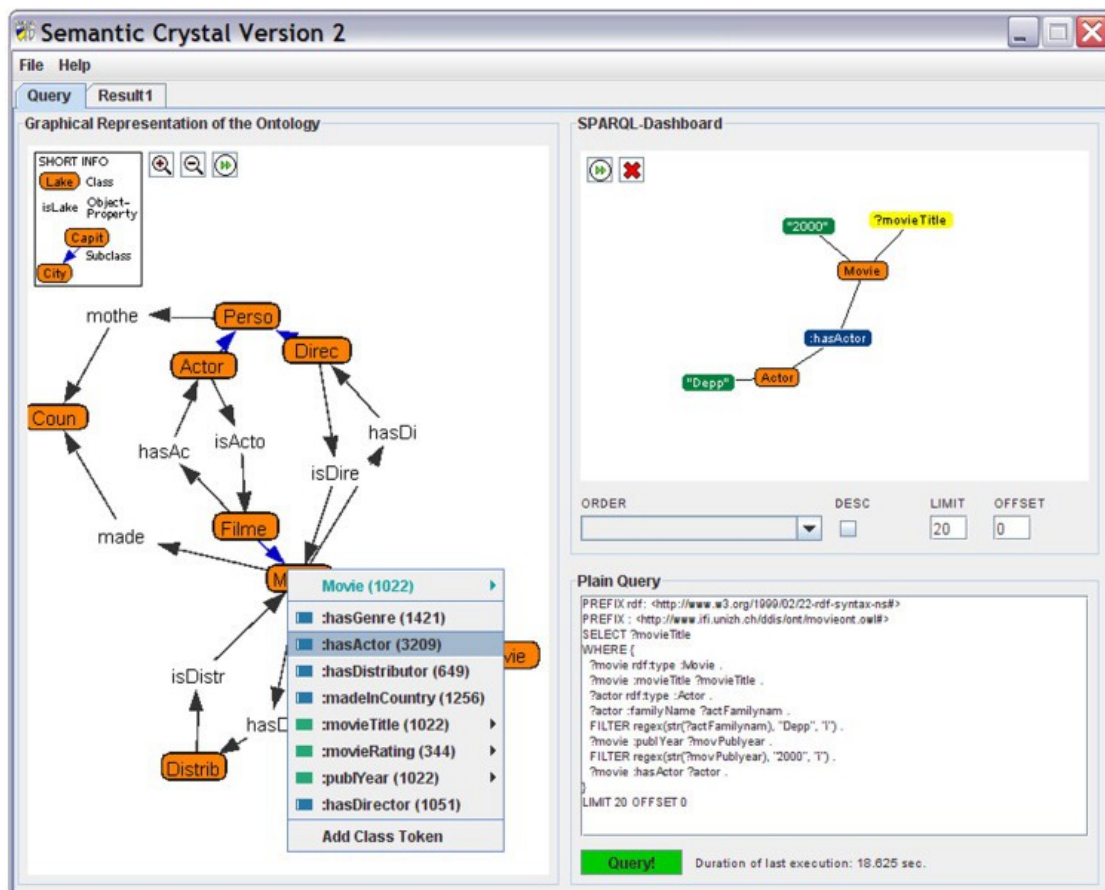


Fig 4.4-1 Semantic Crystal durante la costruzione di una query

4. Nuove interfacce: Linguaggio naturale

Semantic Crystal permette quindi l'interrogazione di ontologie pre-esistenti ed è indipendente dal dominio di ricerca. L'interfaccia utente mette a disposizione la possibilità di creare query SPARQL in modo grafico ed intuitivo. Cliccando sulle entità si possono esplorare le relative proprietà, in modo da comporre query via via più complesse. Come in Ginseng, il framework Jena è utilizzato come layer di accesso alla knowledge base, espressa nel linguaggio OWL. Il modulo *Semantic Crystal XML converter* invece trasforma l'ontologia in un pratico formato basato su XML, di modo che il tool di visualizzazione grafica sia in grado di rappresentarlo nell'interfaccia utente.

5. Natural To Artificial

L'idea di N2A nasce dalla volontà di poter controllare un PC Desktop utilizzando un microfono ed il linguaggio naturale. Da anni siamo abituati a tecnologie simili per l'Accesso Facilitato integrate in alcuni sistemi operativi, ma il difetto di queste soluzioni è che limitano la libertà di esprimersi dell'utente. Si possono sì controllare finestre, aprire applicazioni o scrivere testo utilizzando la propria voce, ma si è costretti ad usare un particolare insieme di comandi in un ordine ben specifico. Il bisogno quindi di imparare questi comandi è un deterrente all'utilizzo della tecnologia, che di conseguenza spesso non è sfruttata dagli sviluppatori, poiché l'utenza è minima.

5.1 Assistente Virtuale: I vantaggi

L'usabilità di un assistente virtuale è limitata esclusivamente dalle funzionalità offerte. Come abbiamo visto nel caso degli smartphones, la soluzione offerta da Apple con Siri è utile in tutti quei casi in cui sia difficile poter interagire con il device attraverso l'uso del touchscreen. Allo stesso modo, nel caso della domotica, si deve poter avere accesso ad un client di gestione, dal maggior numero di stanze presenti nella casa. Gli esempi sono tanti, ma i vantaggi della tecnologia si possono forse riassumere:

- **Assenza di interfacce grafiche:** un software di riconoscimento vocale può essere la soluzione ottimale in cui sia complesso poter utilizzare un'interfaccia o sia difficile implementarne una abbastanza intuitiva per l'utente finale. Essendo inoltre l'interfaccia grafica una considerevole voce di spesa per chi implementa software, l'assenza di essa rappresenterebbe un risparmio.
- **Immediatezza:** il linguaggio naturale non richiede manuali, istruzioni d'uso o corsi di apprendimento. Lo conosciamo, lo utilizziamo tutti i giorni, al contrario di computers, sistemi operativi e programmi che spesso sono ambigui ed emblematici anche per gli utenti più esperti.

- **Accessibilità:** come abbiamo detto, un assistente virtuale che riconosca il linguaggio naturale ha bisogno solo di microfoni posti in modo che la voce dell'utente possa raggiungerli in tutte le occasioni. Non c'è bisogno di ulteriore hardware per interfacciarsi, non esiste necessità di utilizzare le mani su di un touchscreen o altri tipi di dispositivi di input.
- **Multifunzionalità:** un assistente virtuale è in grado di svolgere differenti compiti. Mantenendo l'esempio nel campo della domotica, un software di gestione della propria casa può essere utilizzato per aprire e chiudere finestre, serrande, spegnere ed accendere luci, attraverso i progressi nel campo delle Smart Grid addirittura poter controllare i singoli elettrodomestici. Essendo multifunzione però, l'assistente può anche essere utilizzato per compiti che non riguardino strettamente la casa, ma potrebbe ad esempio riprodurre musica, filmati su determinati schermi, leggere o inviare posta elettronica e documenti, ma anche fornire informazioni di carattere generale prese da endpoints semantici via internet.

5.2 Gestire le differenti funzionalità

Considerato il linguaggio naturale e quindi l'assenza di comandi specifici per compiere determinate azioni, come distinguere se un utente vuole riprodurre un brano musicale o sta cercando di eseguire una ricerca su internet?

L'idea più semplice a cui pensare, è quella di *capire* cosa un utente stia chiedendo. A questo proposito è necessario analizzare la struttura semantica della richiesta e valutarla. Il concept iniziale di N2A prevedeva di utilizzare un sistema di valutazione di determinate combo, estrapolate dalla domanda posta in input.

5. Natural To Artificial

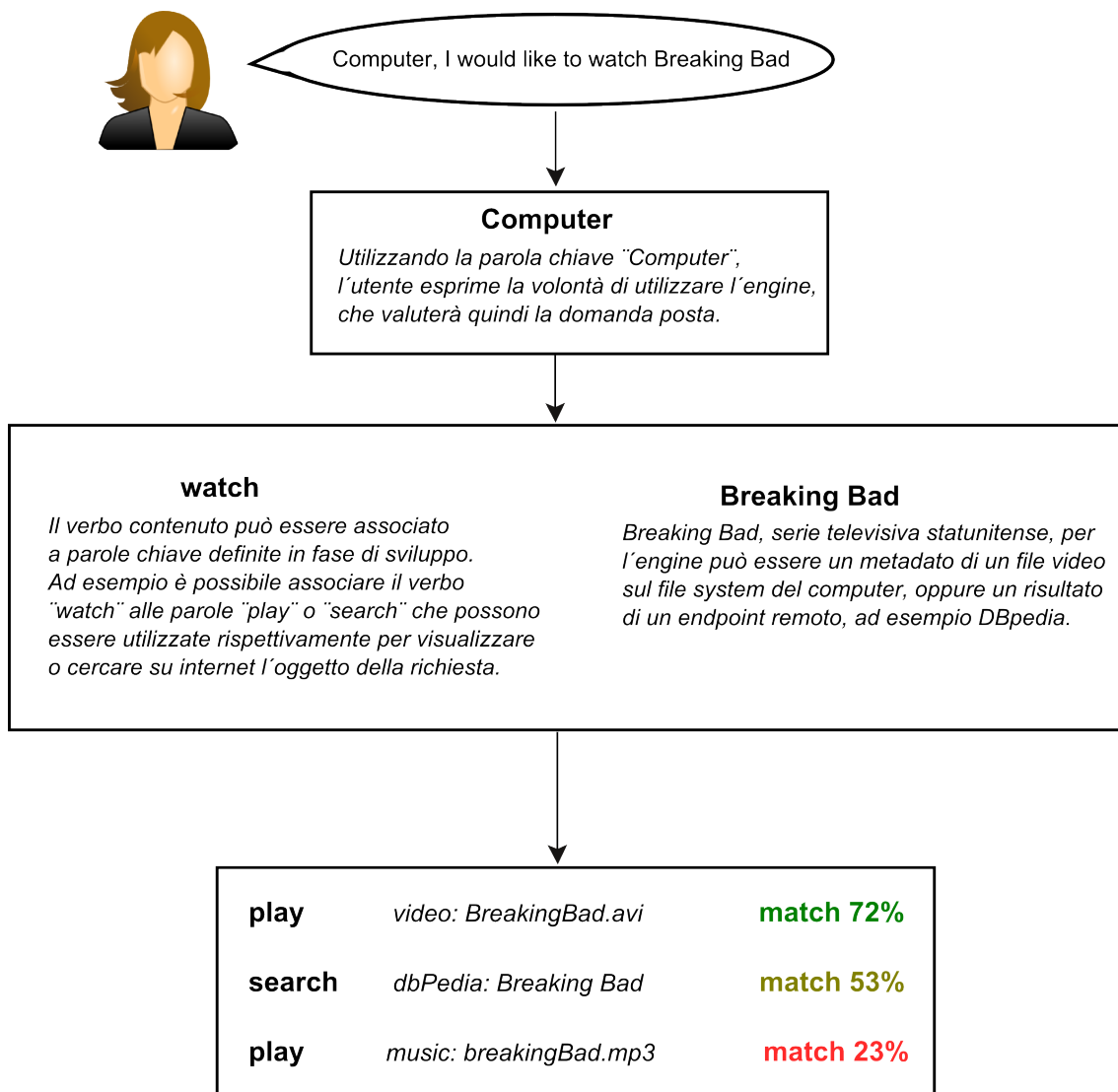


Fig 5.2-1 N2A: Concept iniziale

5. Natural To Artificial

Se il database degli oggetti può essere quindi gestito esternamente, tramite metadati presenti nel filesystem o risultati di ricerca su endpoints remoti, è invece necessario che i predicati associati alle parole chiave siano salvati localmente e che le relazioni con i sinonimi siano modificabili in base all'esperienza dell'utente. In alcuni casi infatti è possibile che due utenti richiedano la stessa funzionalità, utilizzando però verbi differenti o è possibile che alcuni utenti nell'esempio mostrato intendano visualizzare i risultati di ricerca, ma si esprimano comunque utilizzando la parola "*watch*". L'engine quindi ha bisogno di tempo e di esperienza per adattarsi all'utente e migliorare in questo modo il sistema di matching.

Nell'ambito di questo progetto si è scelto di procedere con la sola implementazione della parte riguardante le ricerche su database semantici remoti. Nel concept iniziale questa funzionalità doveva essere attivata con comandi che contenessero la parola *search* o eventuali sinonimi, oppure domande dirette esplicite, espresse tramite *WH Question Words*²⁶.

5.3 Strumenti utilizzati

5.3.1 WordNet

WordNet è un database semantico-lessicale per la lingua inglese nel quale nomi, verbi, aggettivi ed avverbi sono associati l'un l'altro da relazioni semantiche. Il risultato ottenuto è un grafo dove ogni nodo corrisponde ad una parola ed ogni arco è una relazione fra esse. WordNet è uno strumento fondamentale per la linguistica computazionale e l'elaborazione del linguaggio naturale. Il database è liberamente disponibile per la consultazione online, ma anche sotto forma di data files dal quale N2A è in grado di estrarre le informazioni necessarie. In particolare di WordNet si sono salvate le parole e le relazioni semantiche relative ad esse. Le entità Relation del database di N2A provengono interamente dal parsing dei data files di WordNet.

26 English Club – WH Question Words - <http://www.englishclub.com/vocabulary/wh-question-words.htm>

5.3.2 ReEx

L'Open Cognition Project mira alla creazione di un framework open source per lo sviluppo di una intelligenza artificiale. Sponsorizzato dal 2008 dal Singularity Institute for Artificial Intelligence, ha pubblicato recentemente una roadmap²⁷ piuttosto ambiziosa, che prevede nei prossimi 10 anni di arrivare alla realizzazione di una intelligenza artificiale forte, capace di migliorarsi ed apprendere. ReEx²⁸ è l'estrattore di dipendenze semantiche in uso all'interno di OpenCog. Costruito sulla base del Carnegie-Mellon Link Grammar parser, si differenzia dagli altri parser in quanto è incentrato sulla comprensione semantica della frase, piuttosto che alla rappresentazione sintattica del testo. In questo senso è preferibile utilizzarlo per il parsing di domande ed all'interno di motori inferenziali, poiché in grado di fornire un output fortemente semantico e di distinguere domande, da frasi ipotetiche o speculative.

Come ogni altro strumento di OpenCog, di ReEx sono disponibili i sorgenti sotto licenza Apache. Per lo sviluppo di N2A, è stata creata una build apposita sotto forma di libreria .jar

5.3.3 Sesame

Rilasciato sotto licenza BSD, Sesame è un framework open source per l'interrogazione e l'analisi di dati RDF. Supporta SPARQL e SeRQL come linguaggi di interrogazione ed offre la possibilità di memorizzare tuple semantiche su di un database locale per creare sistemi di ragionamento complessi. Nell'ambito del progetto, è stato utilizzato semplicemente per interrogare endpoint SPARQL remoti.

27 OpenCog Foundation – *Development Roadmap* - <http://opencog.org/roadmap/>

28 OpenCog Wiki – *RelEx* - http://wiki.opencog.org/w/RelEx_Dependency_Relationship_Extractor

5.3.4 SPARQL

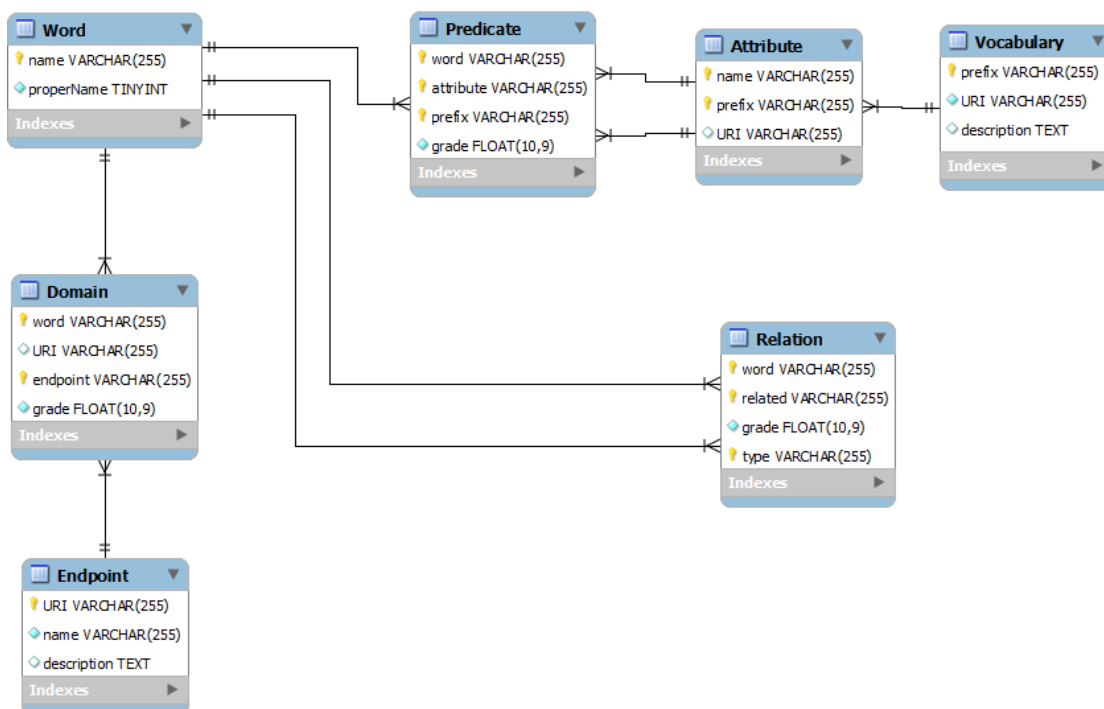
SPARQL è un linguaggio per l'interrogazione di dati RDF su database. Si è diffuso dopo la standardizzazione da parte del Data Access Working Group²⁹, sezione del consorzio W3C, avvenuta nel 2008. SPARQL è un elemento chiave nell'ambito del Web Semantico, permette lo storage di conoscenza e l'estrazione di relazioni su database distribuiti nel web. Il formato RDF permette di identificare triple del tipo *soggetto-predicato-oggetto*, salvate da SPARQL sotto forma di grafi su cui successivamente elaborare le query poste dall'utente. Il consorzio W3C offre una vasta lista di endpoints SPARQL attivi e liberamente interrogabili.³⁰ Gli endpoints spaziano una quantità di argomenti altrettanto vasta, ci sono datasets medici, relativi alla biologia, genetica, dati governativi e generici come nel caso di DBPedia. L'obiettivo di N2A è quello di integrare tutti gli endpoint disponibili, per avere il più vasto parco di triple RDF disponibili, nel quale cercare le risposte alle domande dell'utente. Ogni entità all'interno dell'engine, potrà essere caratterizzata ad uno o più Domini di appartenenza, legati all'endpoint relativo.

29 SPARQL Working Group Homepage: <http://www.w3.org/2001/sw/DataAccess/homepage-20080115>

30 W3C – *Currently Alive SPARQL Endpoints* -<http://www.w3.org/wiki/SparqlEndpoints>

5.4 Database Design

L'entità fondamentale dell'engine è Word, alla quale sono collegati i concetti di Domain, Predicate e Relation con un grado di verità di relazione espresso in logica fuzzy, ovvero capace di variare nell'intervallo 0-1 dei numeri reali. Attraverso il grado di verità, è possibile scartare su base statistica le relazioni che hanno più successo nel risultato delle query, da quelle che portano a più fallimenti. Un sistema di feedback è in grado di incrementare o decrementare questo valore per le entità considerate nella costruzione della query.



Word

name: chiave primaria, definisce la parola vera e propria

properName: booleano che indica se la parola è un nome proprio di cosa o persona

Domain

word: referenza a Word (*name*)

endpoint: referenza ad Endpoint (*URI*)

grade: grado di verità della relazione

URI: riferimento virtuale alla parola considerata, nell'endpoint espresso dalla relazione

Endpoint

URI: riferimento all'endpoint SPARQL remoto

name: nome dell'endpoint

description: descrizione dell'endpoint (opzionale)

resources: riferimento al prefisso da utilizzare per la ricerca di parole sconosciute

Predicate

word: referenza a Word (*name*)

attribute: referenza ad Attribute (*name*)

prefix: referenza a Vocabulary (*prefix*)

grade: grado di verità della relazione

Attribute

name: nome dell'attributo

prefix: referenza a Vocabulary (*prefix*)

URI: riferimento diretto all'attributo

Vocabulary

prefix: nome del vocabolario

URI: indirizzo del vocabolario

description: descrizione

Relation

word: referenza a Word(*name*)

relation: referenza a Word(*name*)

grade: grado di verità della relazione

type: tipo della relazione (Sinonimo, Contrario, Metonimia, Parte di, ecc.)

5.4.1 Relation

L'entità Relation esprime un concetto di correlazione, ovvero una certa parola è legata ad un'altra, attraverso un grado ed un tipo di relazione. Le istanze di questa entità rappresentano le informazioni estratte dal dizionario semantico WordNet. I gradi di correlazione sono stati scelti arbitrariamente e sono riportati nella seguente tabella:

Nouns	
Antonym	0
Synonymous	0.80
Hypernym	0.70
Instance Hypernym	0.60
Hyponym	0.75
Instance Hyponym	0.75
Member holonym	0.75
Substance holonym	0.70
Part holonym	0.70
Member meronym	0.75
Substance meronym	0.70
Part meronym	0.70
Attribute	0.50
Derivationally related form	0.60
Domain of synset	0.40
Member of this domain	0.30
Verbs	
Antonym	0
Hypernym	0.70
Hyponym	0.75
Entailment	0.70
Cause	0.65

5. Natural To Artificial

Also see	0.60
Verb Group	0.50
Derivationally related form	0.60
Domain of synset	0.40
Adjectives	
Antonym	0
Similar to	0.80
Participle of verb	0.80
Pertainym (pertains to noun)	0.80
Attribute	0.50
Also see	0.60
Domain of synset	0.40
Adverbs	
Antonym	0
Derived from adjective	0.80
Domain of synset	0.40

Si consideri che pur essendo i valori iniziali decisi in modo arbitrario, essi possono cambiare durante il funzionamento dell'engine in maniera del tutto dinamica, sulla base dell'esperienza con l'utente.

5.4.2 Domain

L'entità Domain rappresenta il particolare dominio collegato ad una parola. Per dominio si intende l'endpoint nel quale l'oggetto definito da quella parola è contenuto. L'entità nasce priva di istanze e viene riempita automaticamente da N2A durante il corso del tempo. Quando l'engine non conosce su quale endpoint è descritto un determinato soggetto, prova a cercarlo su tutti quelli disponibili.

5. Natural To Artificial

Un esempio concreto può essere rappresentato dalla parola **nitrogen**. Sappiamo che è un elemento chimico, che rappresenta un oggetto concreto del mondo reale, ma su quale dei tanti endpoints questo concetto è descritto? L'unica possibilità per l'engine è quella di ricercarlo ricorsivamente ed attribuire ad un risultato un grado di verità predefinito. Non si può con certezza sapere a priori se l'utente ha intenzione di cercare informazioni di carattere generale, descritte da endpoints quali DBpedia, oppure se necessita informazioni più dettagliate e tecniche, presenti probabilmente in un endpoint usato per natura prettamente scientifica. Si può comunque presupporre legando il grado al risultato della query. La query infatti conterrà uno o più predicati associati al soggetto e questi potrebbero essere soddisfatti solo da uno degli endpoints, che verrà scelto come dominio preferito per questa particolare istanza. Ricercare quindi la categoria di appartenenza dell'azoto, piuttosto che la sua configurazione elettronica, porterà ad istanze Domain differenti. In base all'esperienza ed all'update del grado di verità legato alla parola, successivamente N2A potrà essere in grado di ricercare più spesso sul database scientifico nel caso l'utente sia interessato più spesso ad informazioni tecniche, piuttosto che DBpedia o simili nel quale le informazioni sono divulgative.

Tuttavia, ricercare le entità negli endpoints non è semplice, poiché ognuno di essi implementa il proprio standard. Per ricercare un entità in un endpoint tramite SPARQL è necessario eseguire una query:

```
PREFIX rdfs:<http://www.w3.org/2000/01/rdf-schema#>
SELECT * WHERE {
    ?subject rdfs:label "Nitrogen"@en }
```

Non è detto però che tutti gli endpoints utilizzino la proprietà `rdfs:label` che serve appunto per esprimere una determinata risorsa in modo comprensibile ad un essere umano.

3.6 `rdfs:label`

`rdfs:label` is an instance of `rdf:Property` that may be used to provide a human-readable version of a resource's name.

A triple of the form:

`R rdfs:label L`

states that L is a human readable label for R.

The `rdfs:domain` of `rdfs:label` is `rdfs:Resource`. The `rdfs:range` of `rdfs:label` is `rdfs:Literal`.

Multilingual labels are supported using the language tagging facility of RDF literals.

Tab 5.4.2-1 *RDF Vocabulary Description Language 1.0: RDF Schema - W3C Recommendation 10 February 2004*

Sull'endpoint di BioGateway infatti le proteine hanno nome espresso con la proprietà `ssb:name`. SSB rappresenta quale vocabolario? A quale URI è possibile raggiungerlo? Come è possibile sapere preventivamente in quale modo sono espressi i nomi delle entità in un particolare endpoint che non segue lo standard?

Su Freebase ad esempio le stesse proprietà hanno differenti nomi, ovvero si deve accedere all'oggetto `/type/object/name`.

L'utilizzo di proprietà differenti per esprimere gli stessi concetti sui vari endpoints rappresenta quindi un limite di N2A poiché incapace di supportare pienamente tutti i database. Si dovrebbe legare ogni vocabolario a determinati endpoints e per ognuno specificare determinate regole iniziali per l'associazione dei nomi presenti nel vocabolario.

5.4.3 Predicate

L'entità Predicate rappresenta l'associazione tra parole ed attributi dei vocabolari. Le istanze sono predefinite al momento dell'inizializzazione dell'engine, ma è necessario caricarle a mano. Non ci sono regole ben precise sulla loro nomenclatura e spesso si usa dividere le parole tramite lettere maiuscole, come `<influencedBy>` o `<sameAs>`, ma questa è solo una convenzione che ricalca quelle usate dai linguaggi di programmazione. Inoltre, come visto nel capitolo 2 nel paragrafo relativo ai vocabolari, si trovano spesso ambiguità e non è possibile associare automaticamente e con accuratezza gli attributi a parole del linguaggio naturale.

Questa entità quindi risente sia dell'ambiguità e della ridondanza degli attributi, sia dell'ambiguità in fase di associazione manuale alle parole. Chi determina quali parole sia meglio associare a determinate proprietà? La necessità di compiere un inserimento arbitrario, porta di conseguenza a scelte spesso discutibili. L'attributo `<placeOfBirth>` deve essere legato alla parola *place*, alla parola *birth* oppure ad entrambe? Secondo quale regola?

Le proprietà nei vocabolari inoltre possono mettere a disposizione dei commenti per facilitare la loro comprensione agli utilizzatori, ma non sempre questi commenti sono presenti e non sono pensati per essere interpretati da una macchina. Se però il formato RDF prevede di seguire il facile principio di **soggetto-predicato-oggetto**, perché gli attributi non possono essere espressi così come sono stati pensati, ovvero come predicati?

Il design di N2A quindi potrebbe essere migliorato per prevedere l'associazione di più parole combinate, ad un singolo attributo. Mantenendo l'esempio di `<placeOfBirth>`, potremmo associarlo all'espressione "*place of birth*", composta dalle parole *place*, *of* e *birth* oppure a "*birth place*", composta dalle parole *birth* e *place* poste in sequenza.

What is the **place of birth** of Albert Einstein?

What is the **birth place** of Albert Einstein?

5. Natural To Artificial

Le frasi sarebbero ugualmente comprese e sarebbe più facile per il parser eliminare porzioni di frase, associandole direttamente a specifici attributi, senza doverle trattare come query composte.

Perché quindi non utilizzare come convenzione, quella di associare espressioni del linguaggio comune a determinati attributi? Per facilitare il funzionamento dei motori di Natural Language Processing, ma soprattutto per avvicinarli all'utente finale.

L'utilizzo di `rdfs:label` prevede già questo tipo di approccio, ma è piuttosto raro che un attributo abbia etichette multiple o etichette che lo avvicinino al linguaggio naturale. Inoltre, come già evidenziato nel paragrafo precedente, non è detto che un attributo abbia in primo luogo la proprietà `rdfs:label`.

5.5 Patterns

I patterns rappresentano il punto di incontro tra l'engine e l'utente, attraverso la lingua inglese espressa in un formato logico e grammaticale corretto. Allo stesso modo in cui un compilatore è in grado di comprendere ed elaborare la semantica di un linguaggio di programmazione, N2A è in grado di comprendere ed elaborare le richieste in input solo e soltanto grazie ai patterns conosciuti.

Il formato .patterns implementato, utilizza due informazioni fondamentali dell'estrattore di dipendenze RelEx:

Constituents

ADJP	Adjectival Phrase
ADVP	Adverbial Phrase
NP	Noun Phrase
PP	Prepositional Phrase
PRT	Particle
QP	Quantifier Particle
S	Clause
SBAR	Subordinate Clause
SINV	Subject Inverted
TOP	Sentence
VP	Verb Phrase
WHADVP	Wh-Adverb Phrase
WHNP	Wh-Noun Phrase
WHPP	Wh-Prepositional Phrase

Relations

RelEx relation	Formal description	Example Relation	Example Sentence
_amod	attributive adjectival modifier	_amod(door, locked)	The locked door fell open.
_advmod	adverbial modifier	_advmod(run, quickly)	Jim runs quickly.
_appo	appositive of a noun (appositional modifier)	_appo(bird, robin)	The bird, a robin, sang sweetly.
_date_day	day of month	_date_day(December, 3rd)	It happened on December 3rd, 1990.
_date_year	year modifier	_date_year(December, 1990)	It happened on December 3rd, 1990.
_expl	expletive/filler subject	_expl(be, there)	There is a place we can go.
_iobj	indirect object	_iobj(give, quarterback)	The linebacker gave the quarterback a push.
_measure_distance	unit of distance measure	_measure_distance(a way, foot)	The boy is 4 feet away.
_measure_per	unit of repetition.	_measure_per(times, day)	Take these 4 times a day.
_measure_size	unit of size measure	_measure_size(tall, feet)	The boy is 4 feet tall.
_measure_time	unit of time measure.	_measure_time(old, years)	The birthday boy is 12 years old

5. Natural To Artificial

_nn	prenomial modifier of a noun	_nn(line, goal)	He stood at the goal line.
_obj	direct object, also used for passive nominal subject.	_obj(eat, sandwich)	He ate the sandwich.
_parataxis	parataxis	_parataxis(leave, say)	The guy, John said, left early in the morning.
_pobj	object of preposition	_pobj(next_to, house)	The garage is next to the house.
_poss	possessive or genitive modifier of a noun (gen)	_poss(hand, John)	John's hand slipped.
_predadj	predicative adjectival modifier	_predadj(Smith, late)	Mr. Smith is late.
_psubj	subject of preposition	_psubj(next_to, garage)	The garage is next to the house.
_quantity	numeric modifier	_quantity(dollar, three)	He lost three dollars.
_quantity_mod	quantity modifier	_quantity_mod(three, almost)	He lost almost three dollars.
_quantity_mult	quantity multiplier	_quantity_mult(hundred, three)	He lost three hundred dollars.
_subj	subject of a verb	_subj(be, he) _subj(do, one)	He is the one that did it.
_that	implied preposition "that"	_that(know, angry)	He knew I was angry.
_to-be	adjectival complement (acomp)	_to-be(smell, sweet)	The rose smelled sweet.
_to-do	clausal complement (ccomp/xcomp)	_to-do(like, row)	Linas likes to row.

5. Natural To Artificial

Ogni domanda inserita in input verrà prima di tutto analizzata da ReEx, che restituirà un pattern del tipo:

```
(S What (S (VP is (NP (NP the capital) (PP of (NP Italy))))))
```

Successivamente il riconoscitore di pattern di N2A andrà a leggere ogni file `.pattern` a sua disposizione, per cercare un match con quello dato in input. Il match è calcolato sulla base della corrispondenza dei Constituents.

Un pattern conosciuto che corrisponde a quello preso in esempio, può essere:

```
(S (S (VP (NP (NP ?predicate) (PP (NP ?subject)))))
```

È possibile anche l'utilizzo del wildchar '*' per aumentare i casi simili presi in esame. All'engine in questo modo è stato esplicitamente detto quali sono le parole che corrispondono al Soggetto ed al Predicato della query che dovrà sottoporre all'endpoint SPARQL, secondo la logica RDF:

<i>Italy</i>	<i>has capital</i>	<i>Rome</i>
<i>Subject</i>	<i>Predicate</i>	<i>Object</i>

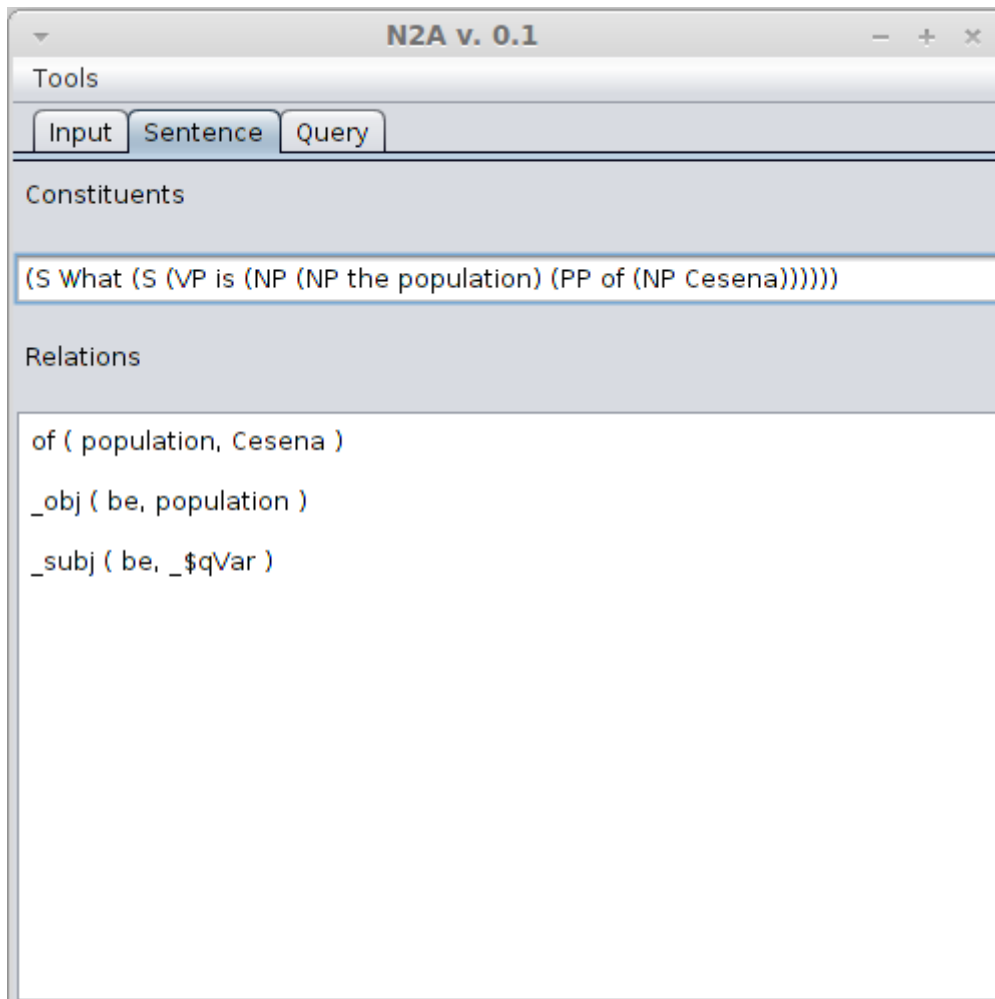
Inoltre è possibile migliorare il risultato attraverso la definizione delle Relazioni nel pattern conosciuto. Se queste sono presenti e corrispondono a quelle della domanda di input, possono funzionare da trigger e dare risultati più sicuri.

In questo esempio:

```
Input: of ( capital, Italy )  
Pattern: of( ?predicate, ?subject)
```

L'utilizzo delle Relations però non è stato implementato poiché del tutto ridondanti nell'analisi di domande non composte.

5. Natural To Artificial



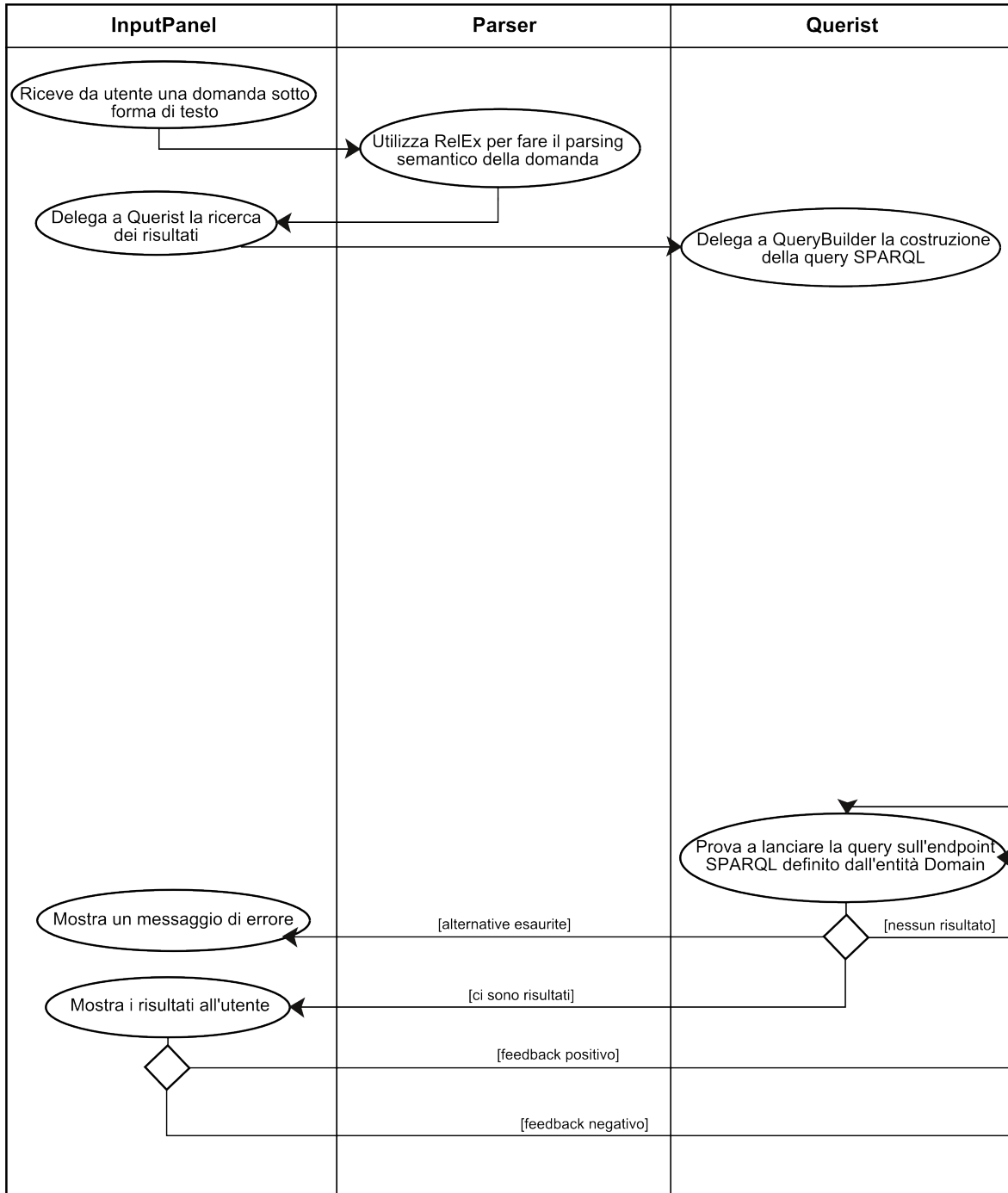
5.5.1 Un problema finito

L'utilizzo dei patterns può sembrare eccessivo, troppo esplicito e contrario alla logica di un motore inferenziale, che dovrebbe giungere da solo agli stessi risultati. La grammatica di una lingua però è finita, composta da regole che rimangono sempre le stesse e tendono ad aumentare molto lentamente nel corso dei secoli e delle abitudini delle persone. In questo senso è quindi molto più semplice pensare ad un "*compilatore di inglese*", piuttosto che ad un motore inferenziale soggetto a contraddizioni, ambiguità della sintassi e regole auto-definite dall'utente. Il limite di query non strutturate è evidente in motori di ricerca quali Google, che soggetti ad un risultato fortemente statistico, non sono sempre in grado di elaborare query composte. I pattern potranno essere molti, ma rappresentano un problema finito, al contrario di quello affrontato da un motore completamente inferenziale.

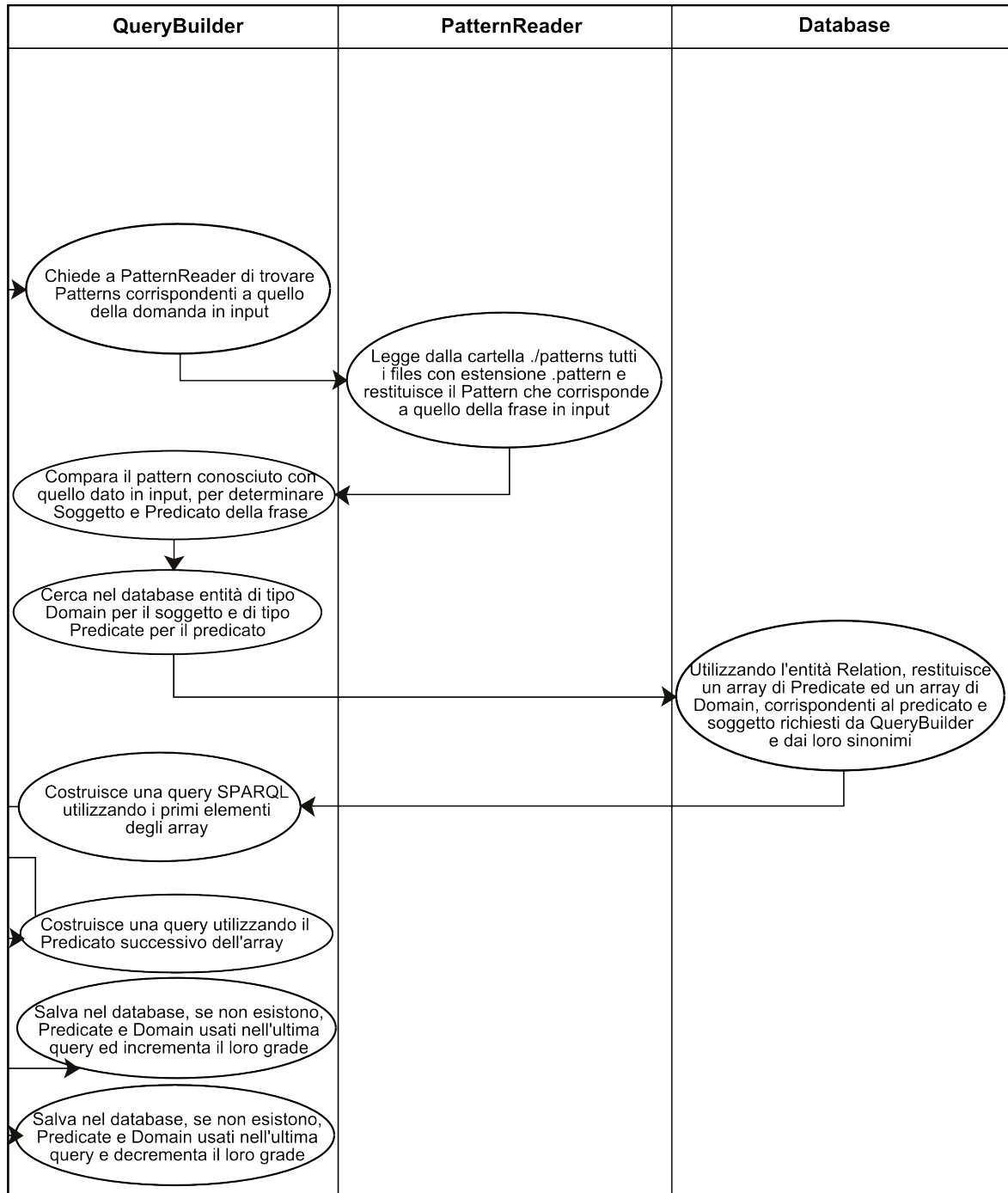
5.5.2 Debugger e crowdsourcing

Il problema dell'inserimento dei patterns inoltre può essere affrontato con un approccio statistico, sulla base dell'esperienza di centinaia di migliaia di utenti, in un'ottica crowdsourced, dove ognuno mette a disposizione il proprio contributo. Il formato infatti è del tutto aperto e semplice da utilizzare. Si potrebbe infatti integrare un debugger dal funzionamento simile a Ginseng, il motore visto nel capitolo 4. In questo modo un utente sarebbe guidato nella creazione della query, ma implicitamente anche alla costruzione di un nuovo pattern.

5.6 Algoritmo



5. Natural To Artificial



5.7 Risultati

Per effettuare i primi test, è stato inserito un sottoinsieme del Vocabolario utilizzato da DBpedia nell'ontologia Country.

```
SELECT ?prop ?title WHERE {  
  ?country a <http://dbpedia.org/ontology/Country> .  
  ?country ?prop [] .  
  ?prop rdfs:label ?title .  
}  
  
ORDER BY DESC(COUNT(DISTINCT ?country))
```

Gli Attributi sono stati associati alle parole come Predicati manualmente, sulla base del loro significato. Il valore del grado di verità è stato inserito arbitrariamente, dando preferenza al match 1:1 con le parole associate.

Un piccolo esempio:

Word	Attribute	Prefix	Grade
anthem	anthem	dbprop	0.50
area	areaTotal	dbprop	0.80
authority	unitaryAuthority	dbprop	0.50
borough	borough	dbprop	0.50
capital	capital	dbprop	0.80
census	census	dbprop	0.50
code	areaCode	dbprop	0.50
income	perCapitaIncome	dbprop	0.80
land	areaLand	dbprop	0.50
language	officialLanguage	dbprop	0.50

5. Natural To Artificial

La domanda in input da prendere in considerazione sarà:

*What is the **population** of Cesena?*

Per testare però la capacità dell'engine di variare tra diversi sinonimi, sbaglieremo di proposito la domanda, sostituendo l'attributo **population**, con la parola correlata **people**.

La domanda posta sarà quindi:

*What is the **people** of Cesena?*

Dopo l'invio della domanda, N2A delega a RelEx, il lavoro di parsing. L'output generato è:

```
(S What (S (VP is (NP (NP the people) (PP of (NP Cesena))))))
```

La classe PatternReader quindi si occuperà di leggere tutti i files con estensione .pattern contenuti nella cartella ./patterns, alla ricerca di una corrispondenza con i Constituents della domanda in input.

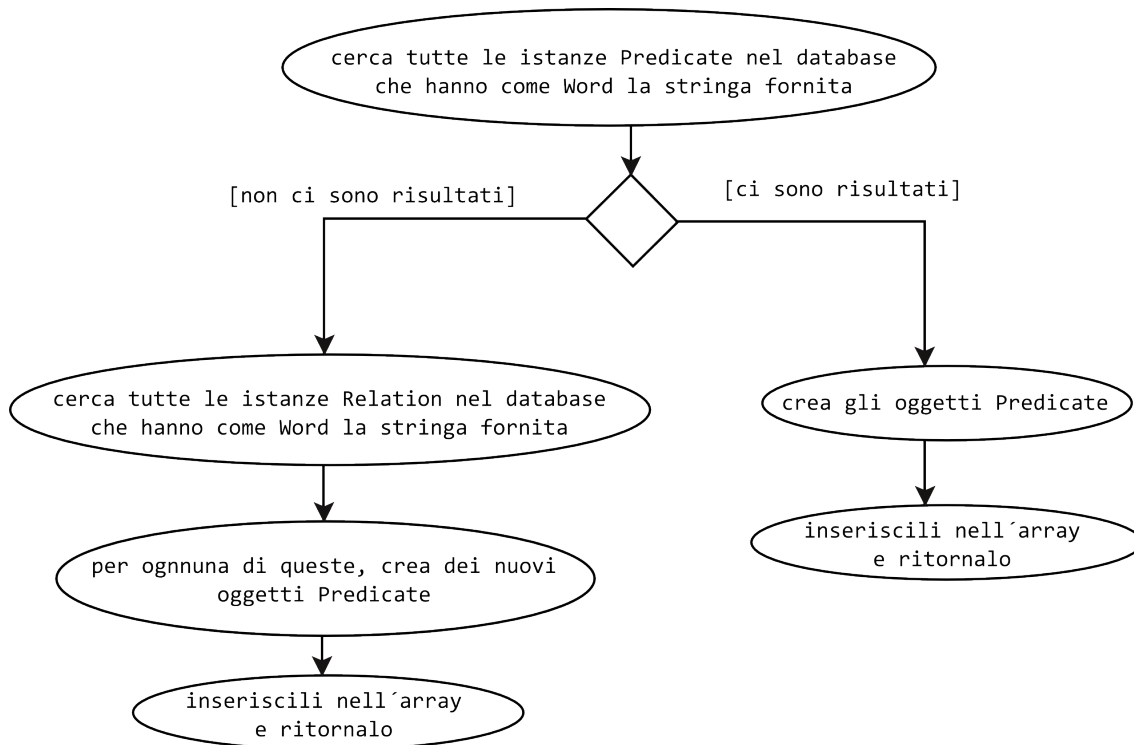
Il pattern trovato è:

```
(S (S (VP (NP (NP ?predicate) (PP (NP ?subject)))))
```

A questo punto la classe QueryBuilder delega alla classe Database la costruzione di un array di Predicates, fornendogli la stringa corrispondente a ?predicate nel pattern. In questo caso people.

5. Natural To Artificial

Database costruisce l'array secondo questo algoritmo:



L'algoritmo così concepito ha bisogno però di ulteriori controlli. Attualmente non è gestito l'evento in cui **people** ritorni degli attributi che non corrispondono però al vero attributo ricercato dall'utente. La ricerca di parole nell'entità Relation verrebbe quindi saltata. Per includerla si può pensare ad un sistema di dimezzamento del grado per ogni Relation trovata che restituisca nuovi attributi ad esempio. Si consideri però il fatto che la frase in input è comunque scorretta e va contro al principio dell'engine, secondo il quale le domande necessitano di essere poste in una forma logica corretta.

In questa istanza Database restituisce un array di 12 Predicate, costruiti sulla base delle parole correlate, ma come word di riferimento quella data in input, cioè **people**. Questo, in caso di successo dell'elaborazione della query, permetterà di inserire nel database un nuovo predicato.

5. Natural To Artificial

Il successo è attualmente dettato dal feedback diretto dell'utente.

word	attribute
people	areaLand
people	largestCountry
people	smallestCountry
people	populationTotal
people	populationMetro
people	populationRural
people	populationUrban
people	populationMetroDensity
people	populationRuralDensity
people	populationTotalRanking
people	populationUrbanDensity
people	populationAsOf

Lo stesso procedimento è implementato per la determinazione del Domain a partire dal ?subject inserito. In questo caso però è stata introdotta una forzatura per ricercare Domain di subjects sconosciuti. La forzatura consiste nell'utilizzare il prefisso **:<http://dbpedia.org/resource/>** per tentare di ricercare fra i risultati di DBPedia, una risorsa che abbia lo stesso nome del ?subject inserito. L'engine potrebbe autonomamente ricercare ricorsivamente l'entità su tutti gli endpoint a disposizione, ma questo attualmente non è semplice per via della diversità di implementazione tra un endpoint ed un altro.

Successivamente QueryBuilder costruisce una query del tipo:

```
SELECT * WHERE { subject predicate ?object }
```

Sostituendo al soggetto la parola **:Cesena** ed al predicato l'attributo **dbprop:areaLand**

5. Natural To Artificial

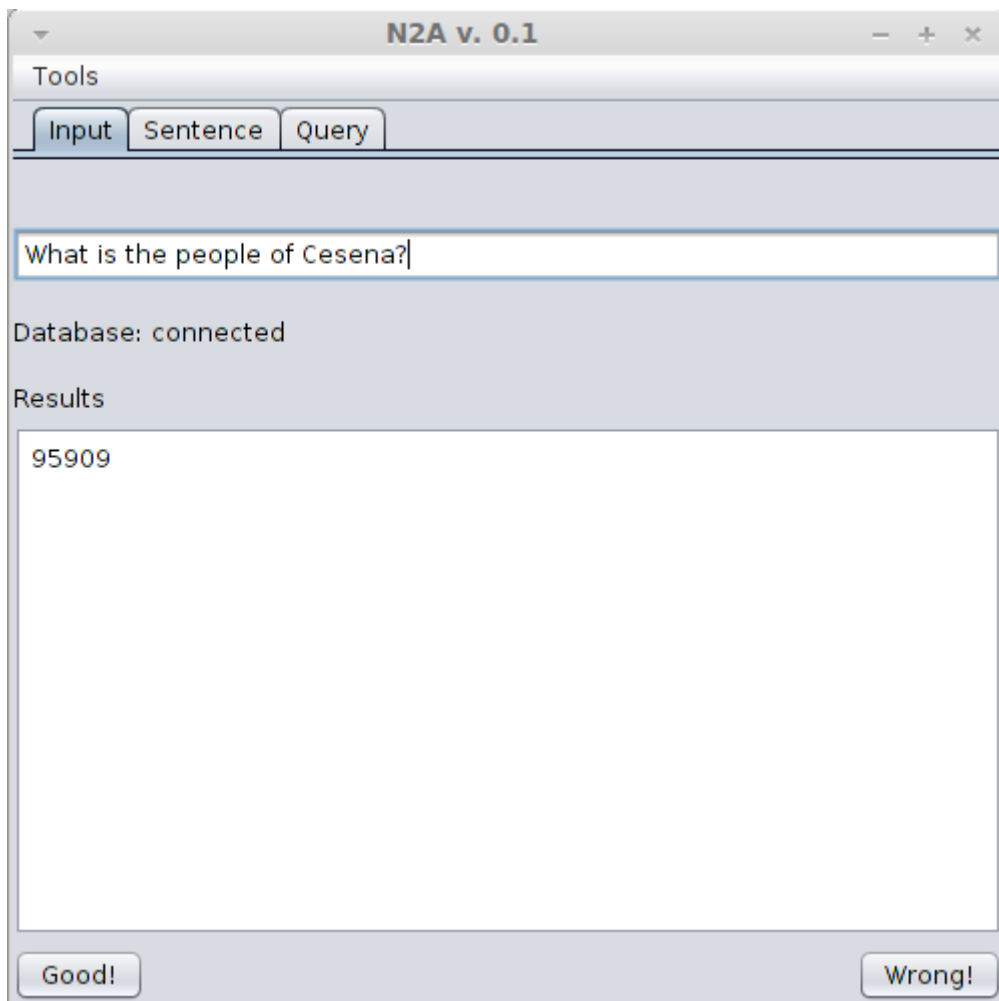
La classe Querist, incaricata di eseguire la query all'endpoint remoto, in caso di assenza di risultati, chiede alla classe QueryBuilder di offrire un'altra query SPARQL, ciclando gli attributi disponibili fino a quando non saranno esauriti, determinando l'assenza totale di risultati per la domanda posta.

In questo caso, il primo attributo che restituisce risultati è populationTotal.

```
PREFIX :<http://dbpedia.org/resource/>  
PREFIX dbprop:<http://dbpedia.org/property/>  
SELECT * WHERE { :Cesena dbprop:populationTotal ?object }
```

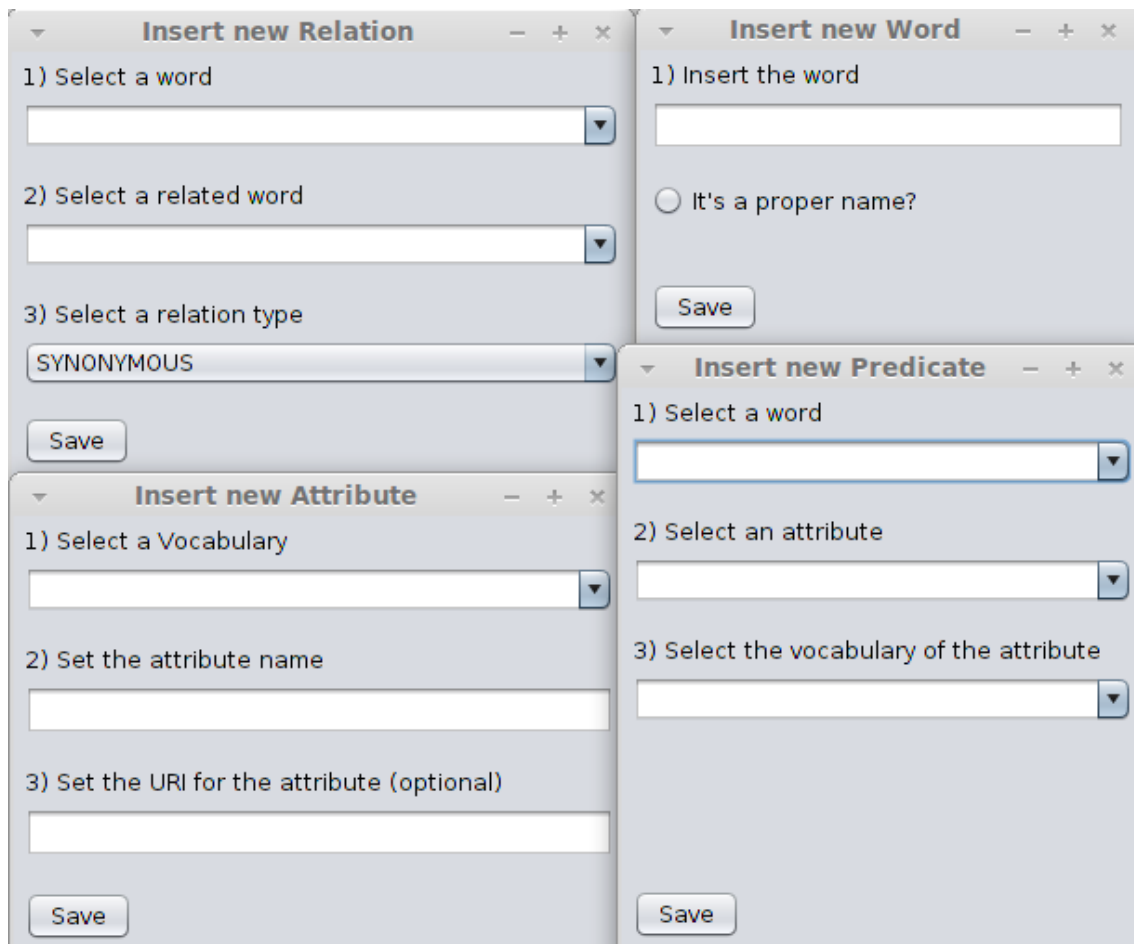
La query costruita quindi può ritornare il risultato e l'utente decidere se è corretto oppure no. In caso di risultato corretto, l'engine inserirà il nuovo Predicato `people:populationTotal` nel database ed incrementerà il suo grado di verità che di default è inserito a 0.5. In caso di risultato non corretto invece, è stato scelto di inserire comunque il Predicato nel caso un risultato sia stato mostrato, per poi abbassarne il grado di verità.

In questo caso il valore ritornato è numerico. Nel caso invece di vere e proprie entità dell'endpoint, è ritornato l'indirizzo di riferimento. In questi casi, con l'integrazione di un browser, si potrebbe direttamente mostrare la pagina con tutti i dettagli per l'entità fornita, piuttosto che una semplice risposta.



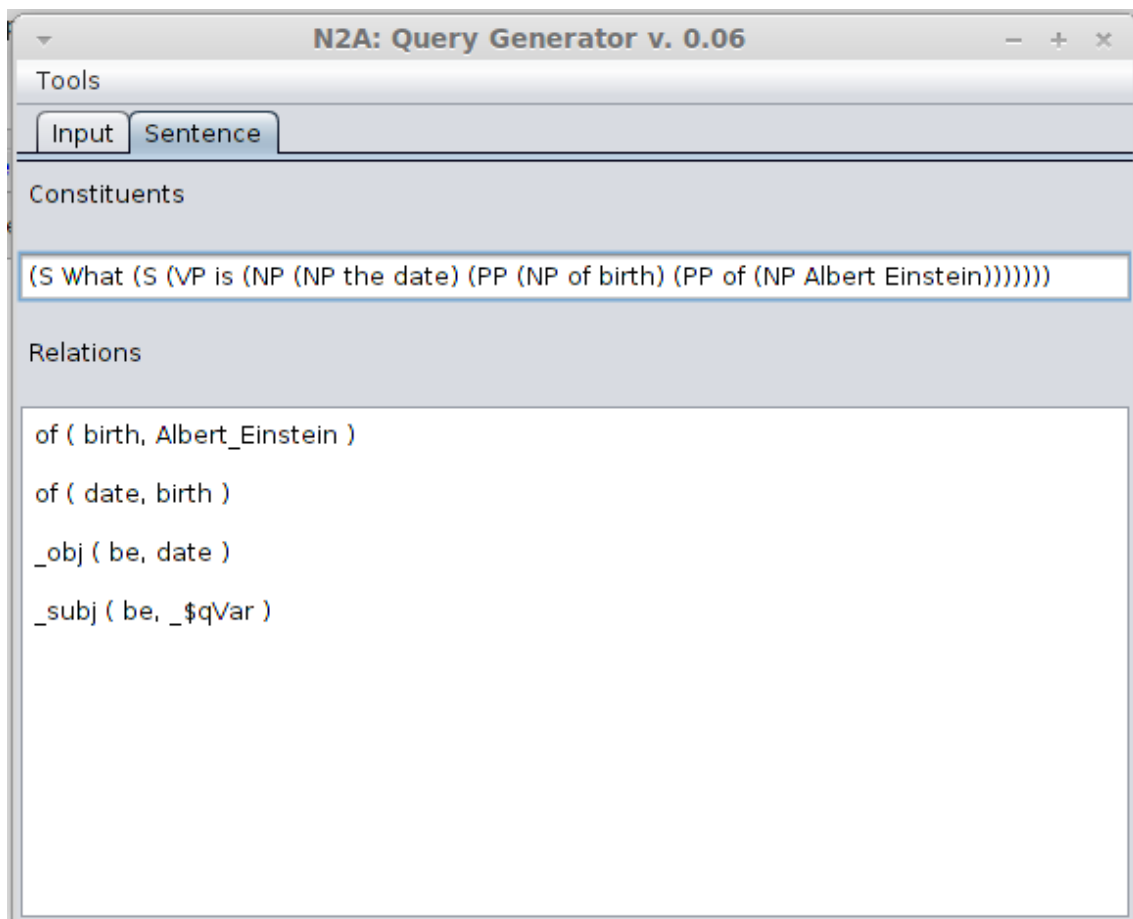
Conclusioni

Il progetto Natural To Artificial ha portato alla realizzazione di un engine scritto in Java di facile utilizzo. Attualmente N2A mette a disposizione dell'utente dei tool grafici per il riempimento del database. Una finestra per la consultazione dei risultati forniti dall'estrattore di dipendenze ReEx ed una finestra dove viene visualizzato l'endpoint sul quale la query è stata lanciata e la query SPARQL stessa.



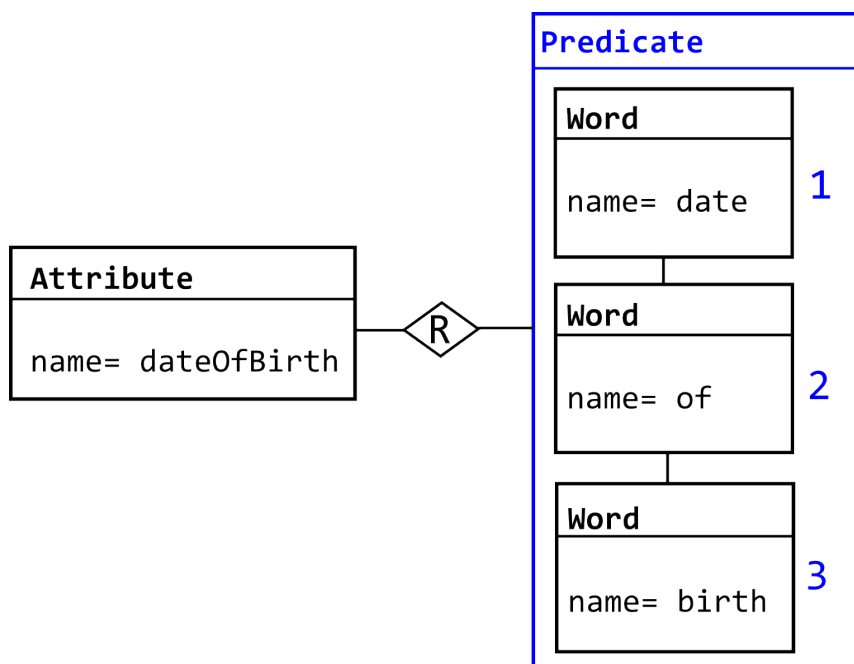
Conclusioni

L'obiettivo del progetto consisteva nella realizzazione di un engine di ricerca semantica che non si basasse su dati locali salvati sotto forma di triple RDF, ma che comprendesse il senso della frase in linguaggio naturale data in input, riuscendo a scandire soggetto e predicato di essa. Il modello di database applicato è stato ideato prendendo ad esempio gli sviluppi interni all'Open Cognition Project. L'entità Word è al centro del design del database e ciò significa che la comprensione dell'input avviene tramite l'attribuzione di un preciso significato ad ogni parola. Sia nel caso si tratti di un soggetto descritto da uno degli endpoints semantici presenti in rete, sia nel caso faccia riferimento ad un particolare attributo di qualche vocabolario.



Conclusioni

I risultati dimostrano che è possibile utilizzare SPARQL ed il formato RDF per permettere al linguaggio naturale di accedere agli strumenti del Web Semantico. Questo può avvenire anche utilizzando svariati endpoints remoti, che però implementino uno stesso standard. Come dimostrato nell'esempio nel capitolo precedente, N2A è in grado di comparare parole del linguaggio naturale ed attributi presenti nei vocabolari, di modo da risolvere automaticamente le ambiguità. Ulteriori funzionalità possono essere aggiunte lato client, con l'implementazione di un debugger per la costruzione guidata di query e l'inserimento automatico di nuovi patterns, ma anche con un miglioramento del database design, di modo che si possano associare non solo singole parole agli attributi, ma interi predicati formati da una composizione ordinata di verbi, sostantivi ed aggettivi. Ad esempio:



Lato server invece è importante incentivare il legame tra linguaggio naturale ed il formato di rappresentazione delle informazioni. Il Web Semantico infatti ad oggi offre un set di tecnologie dall'ottimo potenziale. Gli standard adottati inoltre permettono già di implementare soluzioni efficaci nell'ambito della rappresentazione delle informazioni e della loro elaborazione.

Conclusioni

Strumenti come RDF e SPARQL sono in costante miglioramento e mettono a disposizione una completa e dettagliata documentazione. I problemi relativi alla privacy o alla veridicità dei contenuti sono presenti, ma quotidianamente affrontati e discussi già nel contesto attuale del Web 2.0. Soluzioni di crowdsourcing democratico e feedbacks sono state adottate con successo da portali quali Wikipedia. Lasciare agli utenti il controllo dei dati porta quindi per esperienza a risultati concreti e di grande importanza. Tuttavia si sente l'assenza di software altrettanto aperti e liberi quali MediaWiki, senza il quale Wikipedia non avrebbe avuto l'estensione attuale. Il bisogno di definire ontologie in modo democratico e funzionale attraverso gli utenti è una questione una certa entità. Essi hanno il compito sia di migliorare lo standard, sia di trovare soluzioni che rappresentino il sapere collettivo al di là del maggior numero possibile di ambiguità. L'eterogeneità degli standard adottati di fatto limita l'effettiva preparazione degli sviluppatori, che sono costretti a dover gestire una torre di Babele di linguaggi, sempre più simile a quella rappresentata dalle vastissime API adottate dal Web 2.0. Freebase infatti ha sicuramente ottime funzionalità ed un supporto di livello commerciale, ma costringe chi vuole usarlo a privarsi di tutti gli altri endpoints che seguono lo standard RDF/SPARQL o ad implementare una soluzione differente per le due diverse basi di dati, con tutti i costi di sviluppo e di manutenzione del software. Il limite per le aziende quindi consiste anche nel fatto che poche persone conoscono tutto, la loro rarità le rende risorse molto costose ed i singoli progetti dipendono spesso da poche persone, con il rischio che queste se ne vadano, mandando a monte il progetto stesso e l'investimento dell'azienda. D'altra parte, convincere le aziende a pubblicare i dati in modo trasparente è difficile, poiché ad oggi non c'è molto da guadagnare, se non la fiducia virtuale degli utenti. Il futuro del Web Semantico e dei motori di interpretazione del linguaggio naturale che vogliono usare i dati da esso forniti, dipende con molta probabilità dalla capacità di spronare gli utenti ad adottare gli standard.

Questa motivazione può essere spinta da un'argomentazione etica, ma dovrebbe essere anche aiutata da soluzioni reali. Sotto quest'ottica Natural To Artificial verrà rilasciato sotto licenza Open Source, per dare la possibilità a chiunque voglia di contribuire con lo sviluppo e di migliorare il progetto, integrandolo anche in motori di riconoscimento vocale e controllo del desktop. L'idea per il futuro è quella di supportare lo standard e le convenzioni dichiarate dal World Wide Web Consortium, evitando scrupolosamente soluzioni differenti.

Conclusioni

Selezionare con tale criterio le fonti di informazioni porta a drastiche perdite di dati, ma è più importante in questa fase contribuire alla costruzione di una base dati comunitaria, libera ed indipendente. L'approccio dei Patterns inoltre dovrà essere migliorato, per consentire alla comunità di inserirne di nuovi o di ottimizzarne il formato, attraverso form di costruzione guidata della query o di un più efficace utilizzo del parser semantico ReEx. La costruzione di un portale per la memorizzazione di dati RDF e la strutturazione delle informazioni, è altrettanto importante. DBpedia è un'ottima base dalla quale partire, ma non ha ancora sviluppato strumenti di crowdsourcing democratico. Inoltre è bene promuovere con la stessa importanza e gli stessi meccanismi, un vocabolario che sia comune a differenti endpoints e possa unire quindi non solo diverse entità, ma addirittura diverse ontologie, poiché esse nel modello semantico non sono altro che classi aggiungibili nel grafo dei Linked Data allo stesso modo delle entità stesse. Un vocabolario comune può agire da collante ed allo stesso tempo può ottenere un maggior supporto, una maggiore attenzione alla ambiguità degli attributi ed alle proprietà relative ad essi. Proprietà oggi considerate *ornamentali*, come i commenti e le etichette, ma di fondamentale importanza per l'avvicinamento del formato RDF al linguaggio naturale e di conseguenza all'utente finale.

Bibliografia

1. Ben Goertzel, Itamar Arel, Cassio Pennachin - *CogBot - An Integrative Cognitive Architecture Aimed at Emulating Early Childhood Intelligence in a Humanoid Robot*
2. *WordNet 3.0 Reference Manual: Format of WordNet database files* , 2012
3. Bob DuCharme - *Learning SPARQL*, 2011
4. Dean Allemang, James Hendler - *Semantic Web for the Working Ontologist: Modeling in RDF, RDFS and OWL*, 2008
5. Amit Sheth, Miltiadis Lytras - *Semantic Web-Based Information Systems: State-of-the-Art Applications*, 2007
6. Bhavani Thuraisingham - *XML Databases and the Semantic Web*, 2002
7. Thomas B. Passin - *Explorer's Guide to the Semantic Web*, 2004
8. Dr John Davies, Professor Dieter Fensel, Professor Frank van Harmelen - *Towards The Semantic Web: Ontology-driven Knowledge Management*, 2003
9. John Davies, Rudi Studer, Paul Warren - *Semantic Web Technologies: Trends and Research in Ontology-based Systems*, 2006
10. Bo Leuf - *The Semantic Web: Crafting Infrastructure For Agency*, 2006
11. Grigoris Antoniou, Frank van Harmelen - *A Semantic Web Primer* , 2004
12. Steffen Staab, Heiner Stuckenschmidt (Eds.) - *Semantic Web and Peer-to-Peer: Decentralized Management and Exchange of Knowledge and Information*, 2006
13. H. Peter Alesso, Craig F. Smith - *Thinking on the Web : Berners-Lee, Godel, and Turing*, 2006
14. Kuldeep Kumar, Sandeep Kumar - *Some Observations on Semantic Web Service Processes, Tools and Applications*, 2009
15. Eric Prud'hommeaux, Andy Seaborne - *SPARQL Query Language for RDF*, 2007
16. IBM - *Stories of a smarter planet - What makes you sick?*

17. Matthew Ikle, Ben Goertzel - *Grounding Possible Worlds Semantics in Experiential Semantics*
18. Ben Goertzel, Hugo de Garis, Cassio Pennachin, Nil Geisweiller, Samir Araujo, Joel Pitt, Shuo Chen, Ruiting Lian, Min Jiang, Ye Yang, Deheng Huang - *OpenCogBot: Achieving Generally Intelligent Virtual Agent Control and Humanoid Robotics via Cognitive Synergy*
19. Christian Bizer, Tom Heath, Tim Berners-Lee - *Linked Data - The Story So Far*
20. Esther Kaufmann, Abraham Bernstein - *How Useful Are Natural Language Interfaces to the Semantic Web for Casual End-Users?*
21. Nilesh Dalvi, Ravi Kumar, Bo Pang, Raghu Ramakrishnan, Andrew Tomkins, Philip Bohannon, Sathiya Keerthi, Srujana Merugu - *A Web of Concepts*
22. Rasmus Hahn, Christian Bizer, Christopher Sahnwaldt, Christian Herta, Scott Robinson, Michaela Burgle, Holger Duwiger, Ulrich Scheel - *Faceted Wikipedia Search*
23. Jamie Taylor - *Querying Freebase: Get More From MQL*, 2011
24. H. Blau, N. Immerman, D. Jensen - *A Visual Language for Querying and Updating Graphs*
25. Esther Kaufmann, Abraham Bernstein, and Renato Zumstein - *Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs*
26. Abraham Bernstein, Esther Kaufmann, Christian Kaiser, Christoph Kiefer, - *Ginseng: A Guided Input Natural Language Search Engine for Querying Ontologies*
27. Esther Kaufmann - *Talking to the Semantic Web – Natural Language Query Interfaces for Casual End-users*, 2009
28. Esther Kaufmann, Abraham Bernstein, Lorenz Fischer - *NLP-Reduce: A “naïve” but Domain-independent Natural Language Interface for Querying Ontologies*
29. Vivek Anandan Ramachandran, Dr. Ilango Krishnamurthi - *NLION: Natural Language Interface for querying ONtologies*