Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Studio e progettazione di un linguaggio ad alto livello per la costruzione automatica di basi di conoscenza AIML

AgneseAugello, MariaVasile, Giovanni Pilato, Giorgio Vassallo,
Salvatore Gaglio

RT-ICAR-PA-04-08 **giugno 2004**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

Studio e progettazione di un linguaggio ad alto livello per la costruzione automatica di basi di conoscenza AIML

AgneseAugello[2], Maria Vasile[2], Giovanni Pilato[1],
Giorgio Vassallo[2], Salvatore Gaglio[1,2]

Rapporto Tecnico N.8: ***Data:***

RT-ICAR-PA-04-08 **giugno 2004**

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sezione di Palermo Viale delle Scienze edificio 11 90128 Palermo
[2] Università degli Studi di Palermo Dipartimento di Ingegneria Informatica Viale delle Scienze 90128 Palermo

Sommario

La ricerca nel campo dell'elaborazione del linguaggio naturale si propone lo sviluppo di sistemi che siano in grado di comprendere, manipolare e generare espressioni linguistiche. Questo processo è tuttavia ostacolato dalle ambiguità che caratterizzano il linguaggio naturale. Sono stati tentati diversi approcci al fine di superare tali limiti: da un lato sono stati sviluppati sistemi che analizzano in dettaglio la struttura sintattica e semantica del linguaggio, dall'altro lato sono stati sviluppati semplici sistemi per sostenere un dialogo più o meno coerente con l'utente. Tra questi ultimi hanno avuto notevole diffusione i chat-bot, agenti software in grado di sostenere attraverso interfacce multimediali una conversazione in linguaggio naturale con un interlocutore umano, e tra questi in particolare il chat-bot ALICE. L'abilità discorsiva di un chat-bot si basa essenzialmente sulla sua base di conoscenza, che quindi dovrebbe essere pertinente ed esaustiva rispetto al contesto di utilizzo. In particolare la base di conoscenza di ALICE è costituita da moduli domanda-risposta, chiamati categorie e descritti mediante il linguaggio di mark-up AIML.

La scrittura delle categorie richiede un enorme spreco di tempo, sia in termini di ricerca delle fonti di informazioni, sia in termini di composizione delle categorie stesse nel rispetto delle regole definite dal linguaggio AIML.

Per tale motivo è stato creato l'*Alicebot Learning Language* (ALL), un linguaggio che consente la generazione di applicazioni interattive per la costruzione di categorie AIML. Tra le funzionalità di tale linguaggio assume particolare importanza la possibilità di utilizzare Internet come sorgente dei contenuti informativi dei chat-bot, la possibilità di poter analizzare in maniera automatica i documenti contenenti le *Frequently asked questions* (FAQ), che per natura sono strutturati come un elenco di coppie domande/risposte ed ancora la possibilità di ALL di estrarre da un documento soltanto le informazioni pertinenti ad un dato contesto, sfruttando le nuove tecnologie di recupero dell'informazione.

1. Introduzione

La ricerca nel campo dell'elaborazione del linguaggio naturale (NLP: Natural Language Processing) si propone lo sviluppo di sistemi che siano in grado di comprendere, manipolare e generare espressioni linguistiche.

Le potenziali applicazioni di tale area di ricerca sono innumerevoli, tuttavia per potere comprendere il linguaggio naturale, un sistema deve potere essere in grado di costruire rappresentazioni formali delle frasi che gli vengono fornite, sulle quali poter compiere, in modo automatico, inferenze e ragionamenti.

Questo processo è ostacolato dalle ambiguità che caratterizzano il linguaggio naturale, per tale motivo l'area dell'elaborazione del linguaggio naturale è stata descritta come 'AIhard'; da questo punto di vista realizzare un sistema competente quanto un uomo nella comprensione del linguaggio naturale richiederebbe la risoluzione del 'problema dell'Intelligenza Artificiale (AI: Artificial Intelligence) [1].

Diversi approcci sono stati tentati al fine di superare tali limiti: da un lato sono stati sviluppati semplici sistemi per sostenere un dialogo più o meno coerente con l'utente, dall'altro lato sono stati sviluppati sistemi che analizzano in dettaglio la struttura sintattica e semantica del linguaggio.

Tra i sistemi più semplici hanno avuto una notevole diffusione i sistemi noti con il nome di chat-bot. Chat-bot è un termine relativamente recente nato per indicare agenti software in grado di sostenere attraverso interfacce multimediali una conversazione in linguaggio naturale (to chat), con un interlocutore umano.

Il funzionamento dei chat-bot si basa essenzialmente sul riconoscimento, all'interno delle frasi dell'utente, di schemi fissi o parole chiave, a partire dalle quali si costruiscono le risposte.

Tali sistemi furono creati nel tentativo di superare un test proposto in un articolo del 1950 da A.Turing conosciuto come test standard di Turing (STT)[2], in base al quale un interlocutore umano deve stabilire tramite un terminale se sta dialogando con un altro uomo o con una macchina.

I chat-bot possono essere utilizzati in diverse aree di applicazione tra le quali l'interazione tra utenti ed applicazioni software, l'interazione tra utenti e siti web, le applicazioni di intrattenimento, la creazione di docenti virtuali in applicazioni educative quali le piattaforme e-learning.

Fra i software realizzati per la creazione di chat-bot particolare interesse viene riservato al progetto ALICE (Artificial Linguistic Computer Entity) della Lehigh University (Pennsylvania)[5]. Per potere implementare il dialogo con l'utente ALICE fa uso di una base di conoscenza costituita da moduli domanda-risposta, chiamati categorie e descritti mediante *l'Artificial Intelligence Mark-up Language* (AIML)[6].

Risulta evidente come l'abilità discorsiva di un chat-bot si basi essenzialmente sulla sua base di conoscenza, che quindi dovrebbe essere pertinente ed esaustiva rispetto al contesto di utilizzo.

Tuttavia la scrittura delle categorie richiede un enorme spreco di tempo, sia in termini di ricerca delle fonti di informazioni da cui estrarre le categorie, sia in termini di composizione delle categorie stesse nel rispetto delle regole definite dal linguaggio AIML.

Per tale motivo è stato creato l'*Alicebot Learning Language* (ALL), un linguaggio che consente la generazione di applicazioni interattive per la costruzione di categorie e file AIML.

Tra le funzionalità di tale linguaggio particolare importanza assume la possibilità dell'utilizzo di Internet come sorgente dei contenuti informativi dei chat-bot; in particolare viene fornita all'utente la possibilità di poter analizzare in maniera automatica i documenti contenenti le *Frequently asked questions* (FAQ), che per natura sono strutturati come un elenco di coppie domande/risposte.

ALL inoltre si accosta alle nuove tecnologie di recupero dell'informazione per permettere ad un utente di estrarre da un documento soltanto le informazioni pertinenti ad un dato contesto. Per tale motivo è stata utilizzata la tecnologia di analisi della semantica latente, una tecnica che consente, mediante l'elaborazione di un vasto insieme di documenti, di estrarre il contenuto semantico delle parole analizzate e di inferire in tal modo le relazioni latenti esistenti fra i termini in esame.

2. I Chat-bot ALICE

Chat-bot è un termine relativamente recente nato per indicare agenti software in grado di sostenere attraverso interfacce multimediali una conversazione in linguaggio naturale (to chat), con un interlocutore umano.

I chat-bot rappresentano un approccio concreto al problema dell'interazione uomo-macchina e come tali, un elemento fondamentale nel processo di creazione di software intelligenti in grado di emulare il comportamento umano.

Il funzionamento dei chat-bot si basa essenzialmente sul riconoscimento, all'interno delle frasi dell'utente, di schemi fissi o parole chiave, a partire dalle quali si costruiscono le risposte.

L'interlocutore mediante un terminale fornisce la domanda ed il chat-bot, analizzando la sua base di conoscenza, restituisce la risposta più adatta, ovvero quella che realizza la corrispondenza (to match) migliore con la richiesta dell'utente.

Fra i software realizzati per la creazione di chat-bot particolare interesse viene riservato al progetto ALICE (Artificial Linguistic Computer Entity) [5].

ALICE fa uso di una base di conoscenza costituita a partire da una collezione di documenti contenenti moduli domanda-risposta. Tali moduli sono descritti mediante *l'Artificial Intelligence Mark-up Language* (AIML)[6] un linguaggio di mark-up basato su XML. Il generico modulo domanda-risposta in AIML viene chiamata *categoria* ed il tag utilizzato per descriverla è <*category*>.

Ogni categoria è composta da una domanda, da una risposta e da un contesto opzionale. La domanda è chiamata *pattern*, il tag usato per descriverla è <*pattern*> e contiene un'espressione che può consistere di frasi o porzioni di frasi in linguaggio naturale, singole parole e simboli wildcard (*, _) utilizzati per indicare una corrispondenza con una qualsiasi parola[1].

La risposta è detta *template*, il tag usato per descriverla è <*template*> e contiene una espressione che può consistere di parole in linguaggio naturale e tag AIML.

I tag AIML possono trasformare la risposta in un piccolo software che può salvare dati, attivare altri programmi, dare risposte condizionali o richiamare ricorsivamente altre categorie.

I due tipi di contesto opzionale sono i tag <*that*> e <*topic*>, che consentono di spostare la conversazione su un determinato argomento (<*topic*>) o di tenere traccia di quanto detto durante la conversazione (<*that*>).

Le categorie sono essenzialmente di tre tipi:

- atomiche
- predefinite
- ricorsive

Le categorie atomiche costituiscono il tipo più semplice di categoria; il pattern non contiene wild-card ed il template è una semplice frase in linguaggio naturale.

Le categorie predefinite permettono al chat-bot di rispondere quando il pattern corrisponde parzialmente alla domanda; ciò viene ottenuto con l'inserimento dei simboli di wild-card all'interno del pattern.

---

[1] I simboli wildcard (*,_) indicano entrambi una generica parola ma possiedono un diverso livello di precedenza nell'algoritmo di pattern-matching utilizzato.

Infine le categorie ricorsive sono caratterizzate dall'avere il tag <*srai*> all'interno del template, che esegue la chiamata ricorsiva ad un'altra categoria. Tali categorie permettono di rendere verosimile il dialogo con l'utente implementando forme di:

- sinonimia: associando la stessa risposta a pattern diversi che esprimono lo stesso concetto;
- riduzione simbolica: riconducendo forme grammaticali complesse a forme più semplici;
- correzione grammaticale: riconducendo le espressioni inserite scorrettamente dall'utente ad espressioni grammaticalmente corrette.

Per migliorare l'efficienza della ricerca dei pattern e per avere una rappresentazione compatta in memoria, il software AIML memorizza tutte le categorie in un grafo gestito da un elemento del sistema denominato Graphmaster. I rami del grafo corrispondono alle parole che costituiscono il bagaglio lessicale del chat-bot, i percorsi dalla radice ad un nodo foglia corrispondono ad una domanda dell'utente e puntano alla risposta corrispondente alla domanda.
Il meccanismo di dialogo di Alice si basa su un algoritmo di ricerca chiamato *pattern matching* che cerca la corrispondenza (*matching*) della domanda con i pattern della base di conoscenza del chat-bot. La base di conoscenza dei chat-bot inoltre è realizzata attraverso un processo ciclico di apprendimento supervisionato chiamato *targeting* che coinvolge utente, chat-bot e supervisore del chat-bot (*botmaster*). Il procedimento consiste nello scorrere i documenti che memorizzano le conversazioni del chat-bot; le domande dell'utente che hanno attivato risposte errate o incomplete costituiscono degli obiettivi (targets) che offrono lo spunto per la creazione di nuove categorie. Il supervisore stabilisce quali tra questi targets diventeranno nuove categorie.

3. LSA

L'analisi della semantica latente è una teoria ed un metodo che permette, per mezzo di computazioni statistiche applicate ad un grande insieme di documenti, di inferire e rappresentare il significato delle parole in essi contenute, in base al contesto in cui le parole vengono utilizzate [4].
L'analisi della semantica latente si basa su un principio: il significato di una parola può essere definito statisticamente a partire da un grande insieme di contesti in cui è presente la parola [7]. Con il termine contesto ci si può riferire ad una frase, ad un paragrafo, o all'intera pagina di un documento. L'insieme dei contesti in cui una data parola può o non può comparire fornisce un insieme di vincoli mutui che determina la similarità di significato della parola rispetto ad altre parole o ad insiemi di documenti.

Le parole ed i documenti analizzati, vengono rappresentati da vettori all'interno di uno spazio semantico parole-documenti; la procedura matematica utilizzata per questo scopo è la decomposizione ai valori singolari (SVD: *Singular Value Decomposition*), una tecnica dell'algebra matriciale che permette di rappresentare le parole ed i documenti come vettori all'interno di uno spazio di grandi dimensioni [8].

La dimensione dello spazio viene scelta opportunamente in modo da eliminare il dettaglio informativo che viene considerato come 'rumore' e catturare le relazioni nascoste (da qui il termine 'latente') tra le parole e tra le parole e gli stessi documenti; la dimensione massima che può essere scelta è determinata dal numero dei documenti analizzati.

La scelta della dimensione ottimale è il punto cruciale dell'applicazione di questo metodo: una dimensione troppo piccola può far perdere informazione utile, mentre una dimensione troppo grande può trattenere l'informazione inutile.

E' stato dimostrato che le rappresentazioni del significato delle parole e dei documenti che possono essere ottenute con la tecnica di analisi della semantica latente, possono simulare una varietà di fenomeni cognitivi umani quali ad esempio l'acquisizione della conoscenza, la classificazione delle parole e la comprensione dell'informazione contenuta in un testo [9].

E' importante notare che per applicare questa tecnica non vengono usati dizionari o basi di conoscenza, non viene analizzato l'ordine delle parole, e non vengono nemmeno usati analizzatori sintattici o morfologici.

Un aspetto che differenzia l'analisi della semantica latente da altri metodi riguarda i dati in ingresso: questa tecnica infatti, riesce ad indurre le rappresentazioni delle parole e dei documenti a partire da associazioni tra espressioni 'unitarie' di significato, come le parole ed i documenti in cui esse compaiono e non da associazioni tra parole successive; l'efficienza della tecnica dipende dalla potente analisi matematica su cui essa si basa.

I sistemi classici di recupero dell'informazione riassumono il contenuto informativo dei documenti e delle richieste d'informazione dell'utente in un insieme di termini indice e recuperano quei documenti che effettuano una corrispondenza lessicale con la richiesta d'informazione, ossia quei documenti che contengono i termini indice contenuti nella richiesta d'informazione.

Tale approccio è semplice ma solleva molte questioni; la maggior parte della semantica di un documento o di una richiesta d'informazione viene persa nel momento stesso in cui il testo che esprime tale semantica viene rimpiazzato da un insieme di termini. Ciò è dovuto principalmente alla varietà di parole che normalmente possono essere usate per descrivere un concetto.

Tra i problemi che un sistema di recupero d'informazione deve affrontare nell'analizzare testi in linguaggio naturale, risultano evidenti quelli che derivano dalla sinonimia, ossia la possibilità di

esprimere lo stesso concetto con diversi termini, e dalla polisemia, ossia la possibilità di utilizzare uno stesso termine con differenti significati in contesti differenti.

A causa di questi problemi, con una strategia di reperimento basata solo sulla corrispondenza lessicale tra il documento e la richiesta d'informazione, si rischia di ottenere un numero troppo alto o basso di risultati: a causa della sinonimia molti documenti rilevanti possono essere scartati perché non contengono gli stessi termini della richiesta d'informazione, a causa della polisemia possono essere restituiti documenti che, pur contenendo gli stessi termini della richiesta d'informazione, non sono in realtà pertinenti alla richiesta stessa.

Il significato di un testo si deduce dai concetti che in esso vengono descritti piuttosto che dalle parole utilizzate. Per questo motivo nasce la necessità di accostarsi al problema con un approccio diverso. L'analisi semantica latente può essere applicata per questo scopo, e l'applicazione della tecnica finalizzata al recupero d'informazione viene chiamata indicizzazione basata sulla semantica latente (LSI: Latent Semantic Indexing).

L'indicizzazione basata sulla analisi della semantica latente utilizza la tecnica di decomposizione ai valori singolari (SVD) per ridurre le dimensioni dello spazio semantico parole-documenti e per cercare di risolvere i problemi dovuti alla polisemia e alla sinonimia che costituiscono un grosso ostacolo ai sistemi automatici di reperimento d'informazione.

L'indicizzazione basata sulla analisi della semantica latente si basa sull'assunzione che la variabilità della scelta delle parole nel descrivere un concetto oscura parzialmente la struttura semantica del documento. Riducendo la dimensione dello spazio termini-documenti viene eliminato il dettaglio informativo e possono essere rivelate le relazioni semantiche nascoste.

Tale metodo analizza statisticamente l'utilizzo delle parole nell'intera collezione di documenti e riesce a posizionare vicini nello spazio quei documenti che pur non avendo termini in comune sono correlati semanticamente.

L'applicazione della tecnica coinvolge l'esecuzione delle seguenti attività:

- Raccolta e pre-elaborazione dei documenti;
- Costruzione della matrice di co-occorrenze parole-documenti;
- Pesatura della matrice;
- Decomposizione ai valori singolari della matrice.

L'applicazione della indicizzazione basata sulla semantica latente richiede la raccolta di un insieme sufficientemente ampio di documenti, dove per documento si intende l'intero testo, un paragrafo, o anche una singola frase.

Un numero elevato di documenti permette di ottenere una maggiore completezza, poiché aumenta il grado di copertura dello spazio dei termini, ed una maggiore precisione, poiché se aumenta il

numero di documenti aumenta la possibilità di trovare le stesse parole in contesti diversi; una conseguenza si riscontra nel fatto che le relazioni indotte sono più significative.

I documenti raccolti vengono sottoposti ad una fase di pre-elaborazione che permette di eliminare la punteggiatura, gli spazi e le parole comuni e quindi di ottenere i termini significativi per i documenti.

Successivamente viene costruita una matrice A di dimensioni $m \times n$, in cui le m righe corrispondono agli m termini indice estratti dall'insieme di documenti e le n colonne corrispondono agli n documenti; il generico elemento $a_{i,j}$ della matrice indica il numero di occorrenze del termine i-esimo all'interno del documento j-esimo.

La matrice A è molto sparsa, poiché il numero di termini contenuti in un generico documento è generalmente di gran lunga minore del numero di termini presenti nell'intera collezione di documenti.

Per incrementare o decrementare l'importanza relativa dei termini all'interno del singolo documento e lungo l'intera collezione di documenti si utilizza uno schema di pesatura; ogni valore di occorrenza viene sostituito da un valore che rappresenta il peso di ciascun termine[10], dato da:

$$d_{ij} = L_{ij} \cdot G_i \cdot N_j$$

dove:

- L_{ij} è il 'Peso Locale', ossia il peso del termine i-esimo nel documento j-esimo;
- G_i è il 'Peso Globale', ossia il peso del termine i-esimo nell'insieme dei documenti;
- N_j è il 'Fattore di Normalizzazione'.

In letteratura esistono vari schemi di pesatura che combinati opportunamente tra loro influiscono in modo determinante sull'efficacia della tecnica di indicizzazione con l'analisi semantica latente[11].

La matrice ottenuta viene sottoposta ad una delle più importanti decomposizioni ortogonali derivanti dall'algebra lineare numerica: la decomposizione ai valori singolari (SVD). Tale decomposizione è comunemente usata nella soluzione dei problemi lineari ai minimi quadrati vincolati e non vincolati, per la stima del rango di una matrice, per stimare i coefficienti nell'analisi di correlazione canonica ed in un vasto insieme di applicazioni quali ad esempio il reperimento di informazione e la tomografia di riflessione sismica[12].

In particolare la SVD troncata ad una dimensione minore del rango della matrice può essere utilizzata per generare una approssimazione di rango $k<<r$, dove r è il rango della matrice.

La decomposizione ai valori singolari della matrice A troncata ad una dimensione k, con $k<<r$ è data da:

$$A_k = U \cdot \Sigma \cdot V^T$$

con U di dimensioni $m \times k$, V di dimensioni $n \times k$ e Σ di dimensioni $k \times k$. A_k è una approssimazione della matrice A che permette di evidenziare informazioni importanti sulla struttura della matrice di partenza.

4. Il linguaggio A.L.L.

A.L.L. (*Alicebot Learning Language*) è un linguaggio appositamente studiato per ampliare la base di conoscenza di chat-bot basati sulla tecnologia ALICE.

La creazione dei file AIML che costituiscono la base di conoscenza di un chat-bot Alice, richiede innanzitutto la scelta di contenuti informativi, ossia la scelta degli argomenti riguardanti il campo in cui verrà impiegato il chat-bot. Gli sviluppatori di Alice hanno creato una grande collezione di file AIML che permettono ai chat-bot di sostenere una semplice conversazione piuttosto generica; tuttavia nel caso in cui si vuole creare un chat-bot finalizzato a dialogare su un determinato argomento, occorre fare riferimento a fonti più ampie e specifiche. Inoltre bisogna tenere conto che il fattore che influenza in modo cospicuo l'abilità discorsiva del chat-bot è il numero di categorie utilizzato, più elevato è il numero di categorie più il chat-bot appare "preparato" su un determinato argomento.

Volendo affrontare il processo di creazione dei file AIML con un approccio top-down i problemi che si pongono in prima analisi sono principalmente due:

- reperimento e modalità di utilizzo delle informazioni: occorre fare riferimento a fonti informative non strutturate, testi, documenti in linguaggio naturale, da cui estrarre la conoscenza che andrà ad incrementare il grado di preparazione del chat-bot, inoltre occorre prevedere un modo per sfruttare opportunamente i contenuti reperiti generando a partire da essi le categorie AIML.
- creazione delle categorie secondo le regole del linguaggio AIML: ogni nuova categoria deve essere memorizzata con la giusta formattazione, è necessario inserire i tag di marcatura <category> …..</category> per distinguerla dalle altre e <pattern>….</pattern> <template> …</template> per distinguere le varie parti del dialogo, e separare quindi il pattern utilizzato per effettuare il matching dell'input dell'utente dal template che costituirà la corrispondente risposta fornita dal chat-bot.

Queste considerazioni mettono in evidenza le difficoltà connesse ad un approccio "manuale" al problema: il lavoro di creazione dei files AIML può essere estenuante e ripetitivo.

Tra i tools esistenti che offrono un supporto automatico alla creazione dei files AIML emerge PandoraWriter[13] reperibile all'indirizzo http://www.pandorabots.com/, il sito offre un servizio di hosting per chat-bot : l' utente può creare , gestire, personalizzare e addestrare chat-bot A.L.I.C.E. . In tale sito l'utente può inserire un dialogo in linguaggio naturale che riproduce una possibile conversazione con il chat-bot e il software automaticamente la converte in file AIML. E' inoltre possibile effettuare il download dei file creati e l'upload, testandoli direttamente sui chat-bot in esecuzione sul server di Pandorabot.

4.1 Soluzione Proposta

A.L.L. è stato progettato per consentire all'utente di realizzare applicazioni che possano costituire un supporto qualificato ed efficiente alla realizzazione di files AIML. E' un linguaggio semplice ed immediato: ha una sintassi molto simile ai comuni linguaggi di programmazione per cui l'utente che ha un minimo di familiarità con essi può utilizzarlo facilmente acquisendo in modo immediato la padronanza dei costrutti e della struttura del programma in genere. La sintassi del linguaggio prevede una certa flessibilità nei costrutti: in molti casi è il linguaggio a risalire, in base al contesto, alla sintassi precisa dei costrutti e ad adattarsi di conseguenza, permettendo all'utente di omettere ad esempio l'indicazione dei parametri in qualche punto del programma. Consente di realizzare applicazioni complesse anche a partire da semplici programmi: ad esempio è possibile realizzare applicativi per la connessione e il reperimento delle informazioni direttamente da Internet e per la selezione di contenuti informativi in modo "intelligente" applicando le nuove tecniche di reperimento dell'informazione messe a disposizione dall'analisi della semantica latente (LSA). A.L.L. è rivolto non solo a chi opera nel settore ma anche ad utenti inesperti che desiderano personalizzare il proprio chat-bot, infatti prescinde da una specifica conoscenza del linguaggio AIML.

Il progetto che ha portato alla realizzazione di A.L.L. prevede l'utilizzo di un compilatore on-line, e la possibilità di testare i file AIML creati aggiungendoli alla base di conoscenza del chat-bot disponibile sul sito.

Vengono di seguito illustrate le diverse funzionalità del linguaggio A.L.L. che consentono di implementare varie modalità di creazione di files AIML:

4.1.1. Inserimento manuale dei contenuti

A.L.L. include istruzioni primitive che consentono di convertire direttamente l'input dell'utente in categorie AIML, ottenendo delle funzionalità analoghe a quelle previste da PandoraWriter.

Questo consentirà di realizzare un' applicazione interattiva mediante la quale il dialogo viene inserito dall'utente e progressivamente convertito in AIML (Figura1).

```
// Esempio prog 1.all
var
        category categoria;
        boolean continua;
        string temp;
MAIN
begin
        newAIMLFile();
        continua = true ;
        given (continua) {
        setQuestion(categoria);
        setAnswer(categoria);
        updateFile( AIMLTranslate(categoria) );
        continua = confirm ("inserire una nuova categoria AIML?");};
        closeAIMLFile ();
end
```

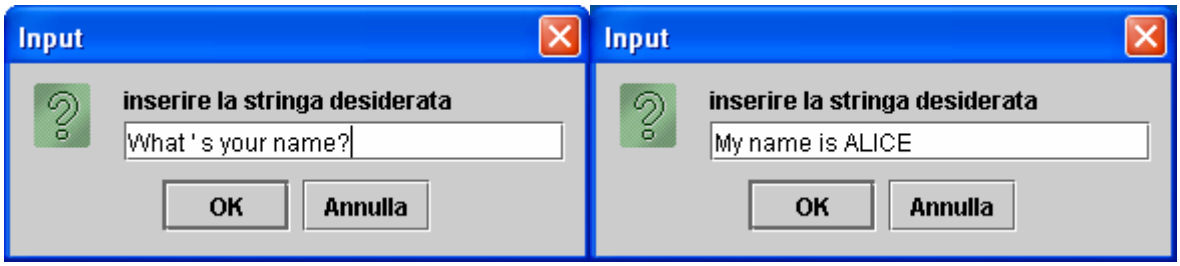

Figura 1: Programma ALL per l'inserimento manuale di categorie

4.1.2. Ricerche on-line su argomenti specifici

A.L.L. permette di costruire applicativi per la connessione e la ricerca on-line di testi che riguardano argomenti specifici necessari all'addestramento dei chat-bot.

Le primitive del linguaggio che realizzano questa funzionalità si basano sulle API del noto motore di ricerca GOOGLE . Può essere costruita in locale una cartella contenente dei files di testo su un preciso argomento mediante una ricerca sul web.

L'esecuzione del programma consentirà di visualizzare l'interfaccia mostrata in Figura2, utilizzando la quale l'utente potrà inserire l'argomento della ricerca, visualizzare la lista di url ottenuti come risultato, visualizzare informazioni specifiche relative ad un particolare url come il

titolo della pagina e il sommario, e scaricare in locale il contenuto testuale di una pagina che può risultare interessante ai fini dell'addestramento.

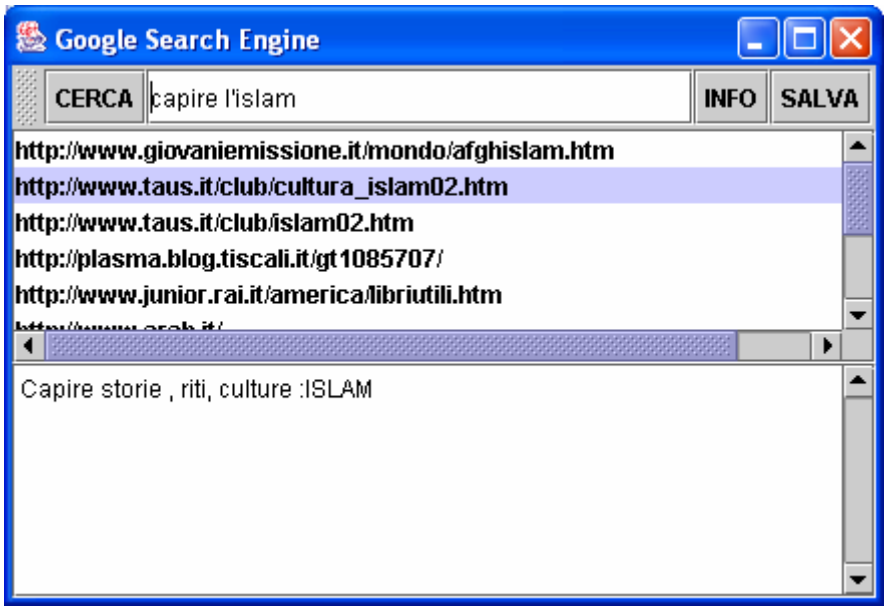

Figura 2: Esecuzione di un programma ALL che permette di scaricare documenti da Internet

4.1.3. Conversione di FAQ in categorie AIML

Con il termine FAQ (*Frequently Asked Questions*) si indica una classe di documenti reperibili sul web che forniscono le risposte a domande comunemente rivolte dagli utenti su argomenti specifici. La struttura domanda-risposta delle FAQ suggerisce la possibilità di implementare un metodo che possa convertire in modo automatico una FAQ in una categoria AIML.

Tuttavia osservando la struttura di un documento contenente FAQ ci si pone immediatamente il problema di estrarre solo la coppia domanda-risposta scartando quelle porzioni di testo che possono essere la premessa , l'indice delle domande, e i riferimenti finali .

Per la realizzazione di una istruzione primitiva che potesse consentire il *parsing* automatico delle FAQ sono state analizzati centinaia di documenti FAQ riguardanti argomenti diversi scaricati dal sito USENET[14]; l'analisi ha portato ad una serie di considerazioni non banali:

La struttura di documento di FAQ è flessibile e dipende dalle scelte personali di chi scrive le FAQ. E' possibile individuare dei marcatori di FAQ che delimitano la coppia domanda risposta, che separano la domanda dalla risposta, o che indicano l'inizio della domanda, ma l'ordine dei marcatori e le loro diverse combinazioni non consentono di applicare un metodo totalmente automatico per l'estrazione delle coppie domanda-risposta. I marcatori individuati nelle FAQ di USENET possono esser ricondotti a dei tipi precisi mostrati nella Tabella1:

Tabella 1: Tipi di marcatori usati nelle FAQ

Tipo marcatore	Rappresentazione grafica
Alpha Markers	"Q:", "A:", "Subject:", "Section:"
Alphanumeric Makers	"[1-0]", "1)", "10.2a", "VIII."
Symbolic Markers	"**********", "==========".

Sono stati condotti diversi studi a riguardo, in particolare nell' articolo "Mining Free Text for Structure"[15], gli autori hanno messo in evidenza le difficoltà riscontrate nella realizzazione di un algoritmo di supporto ad un software chiamato FAQ FINDER, che individua la coppia domanda-risposta cercando di trovare una logica nell'ordine con cui si susseguono i vari marcatori; i risultati ottenuti da FAQ FINDER sono ottimi se paragonati alle difficoltà , esso infatti fallisce solo nel 12% dei casi.

A.L.L. include istruzioni primitive di supporto all'estrazione dei moduli domanda-risposta da un documento contenente FAQ.

In particolare è possibile visualizzare l'interfaccia interattiva di Figura3, utilizzando la quale l'utente può creare una cartella locale di microdocumenti che rappresentano le coppie domanda-risposta estratti da un documento contenente FAQ.

L'utente può visualizzare il documento di FAQ nell'apposita area di testo ed attraverso il pulsante "TROVA_MARCATORI" può attivare una procedura automatica per il ranking dei marcatori .

L'euristica utilizzata è quella del conteggio dei marcatori; questi vengono ordinati in senso decrescente di occorrenze e visualizzati nella lista a campi selezionabili visualizzata in alto . L'utente può scegliere tra i marcatori indicati quello che verrà utilizzato per la scomposizione tramite il pulsante "EXTRACT_FAQ" del testo in microdocumenti. I microdocumenti ottenuti possono essere visualizzati ed eventualmente modificati e salvati in un opportuna directory.

Il programma prevede un ulteriore istruzione che consente di caricare e di visualizzare in successione tutti i microdocumenti salvati precedentemente in modo da dare l'opportunità all'utente di scegliere in modo definitivo le sezioni della FAQ che diventeranno categorie AIML.

```
// Esempio prog 3.all
var
MAIN
Begin
        mode(local);
        showInteractiveFaq() ;
        showInteractiveCategory ();
end
```

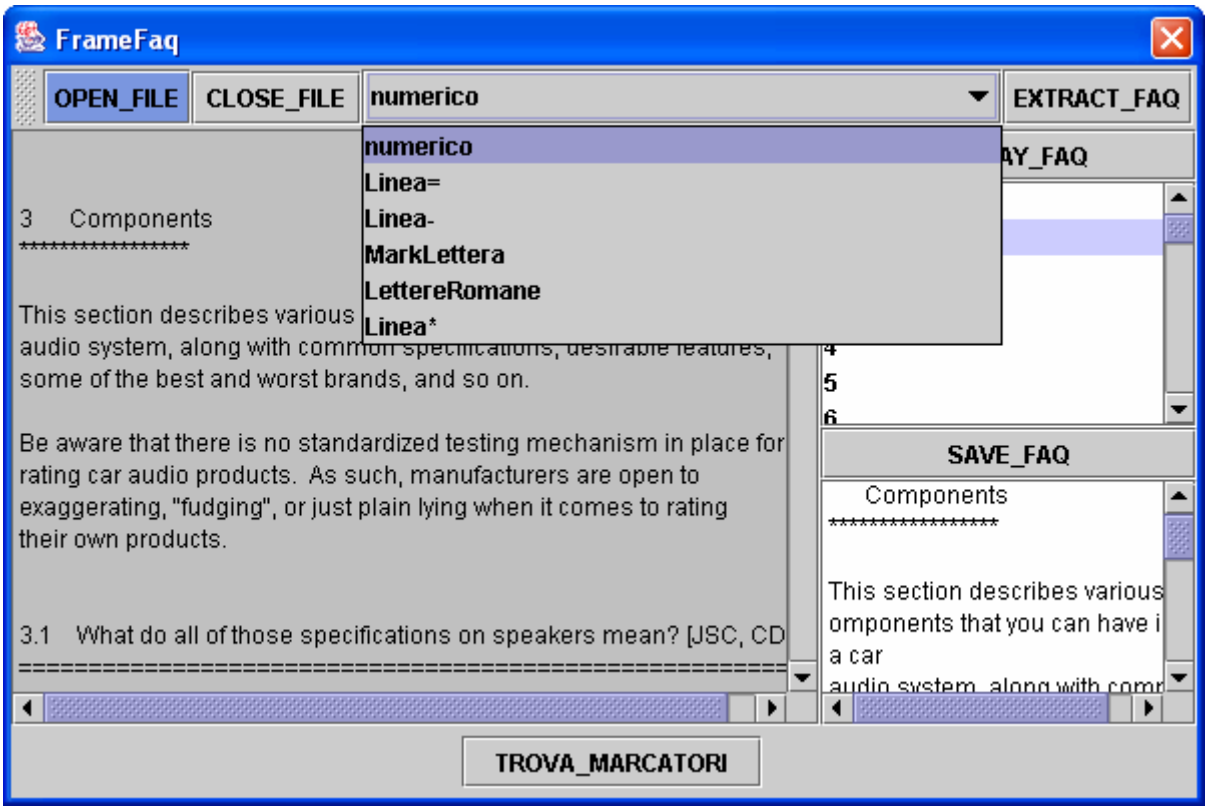

Figura 3: Esecuzione di un programma ALL che permette di estrarre automaticamente le coppie domanda risposta da un documento di FAQ

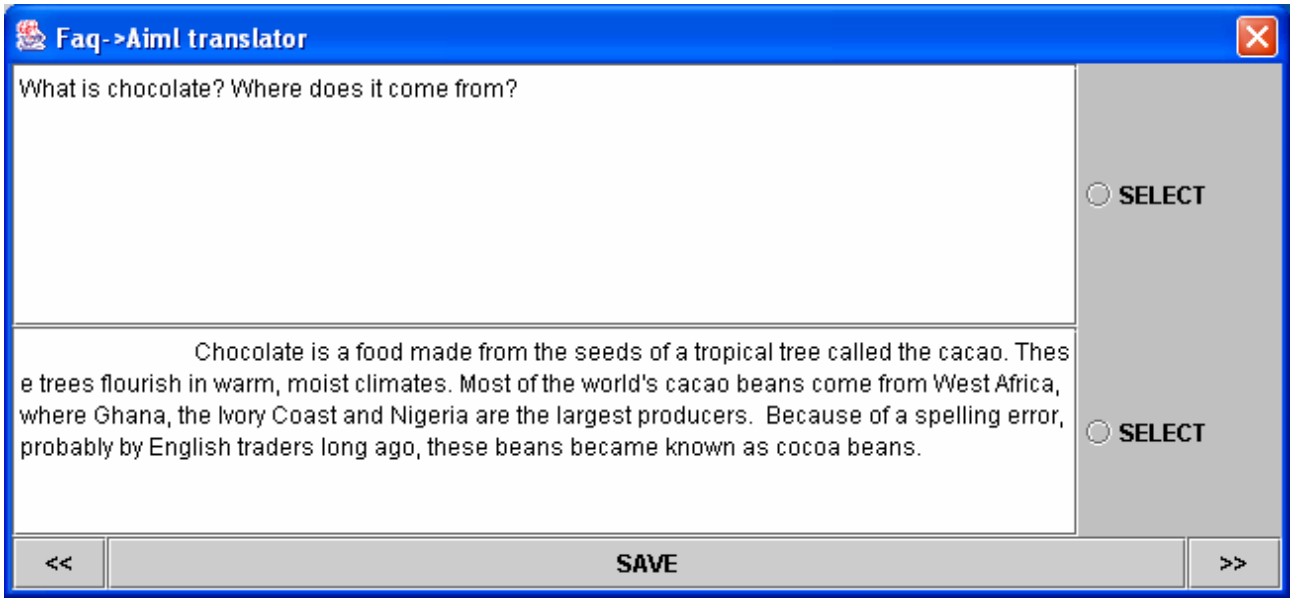

Figura 4: Finestra in cui vengono visualizzate le coppie domanda-risposta estratte dal documento

4.1.4. Estrazione dei contenuti da file di testo e database semantico

Utilizzando A.L.L. è possibile creare semplici applicazioni che costruiscono le parti del dialogo estraendo le informazioni necessarie da file di testo locali che trattano argomenti specifici.

A.L.L. comprende istruzioni per :

- Estrarre tutte le frasi da un file di testo e memorizzarle in una lista: per frase si intende la sequenza di parole comprese tra una coppia di segni di punteggiatura appartenenti all'insieme{ ".","?" } . Questa funzionalità è stata realizzata implementando un analizzatore lessicale , che analizza il file in ingresso con lo scopo di individuare quelle porzioni di testo che corrispondono a frasi di senso compiuto. L'analizzatore grazie all'uso delle espressioni regolari è in grado di riconoscere sigle, abbreviazioni, marcatori di elenco puntato.
- Estrarre frasi che contengono determinate parole: La primitiva del linguaggio che realizza tale funzionalità si basa sulla stessa tecnica vista sopra , ma consente di prelevare da un testo solo le frasi di senso compiuto che contengono parole appartenenti ad una lista passata come parametro. In questo modo è possibile indicizzare il testo rispetto ad un insieme di termini fornito dall'utente, dando la possibilità di sfruttare le potenzialità di un database semantico di parole, costruito applicando la tecnica della LSA. In questa nuova ottica un utente che desidera specializzare un chat-bot su un argomento specifico ed ha a disposizione un file di testo come risorsa informativa può effettuare una interrogazione al database per ottenere le parole semanticamente correlate alla parola-chiave che costituisce l'argomento di interesse, assicurandosi in tal modo la possibilità di individuare in un testo solo le frasi "significative" per l'argomento. Tenendo conto che l'obiettivo è quello di

ottenere file AIML si può pensare di abbinare alle frasi estratte altre frasi proposte dall'utente, per la realizzazione dei moduli domanda- risposta. L'esecuzione del programma genera una successione di finestre di dialogo che consentono all'utente di :

- selezionare la cartella e il nome del nuovo file AIML;
- inserire la parola-chiave su cui vuole focalizzare l'attenzione;
- estrarre cinque parole dal database semantico che sono correlate con la parola data;
- estrarre dal file di testo le frasi che contengono le parole appartenenti alla lista ottenuta;
- visualizzare un interfaccia grafica attraverso la quale le frasi estratte vengono proposte all'utente nella sezione "Template:" che può modificarle e inserire da tastiera nella sezione " Pattern:" le frasi che meglio si abbinano a quelle date per la costruzione del file AIML. In figura è presentata questa interfaccia le cui funzionalità saranno esposte con maggior dettaglio nella sezione manuale utente.

```
// Esempio  prog 4.all
var
        list_of  frasi;
        list_of parole;
        string argomento ;
MAIN
begin
        newAIMLFile();
        argomento = read ("inserire la parola-chiave argomento della ricerca:");
        parole = getSimilarWords(argomento,5);
        frasi = selectSentences(parole);
        showInteractiveAIML (frasi);
end
```

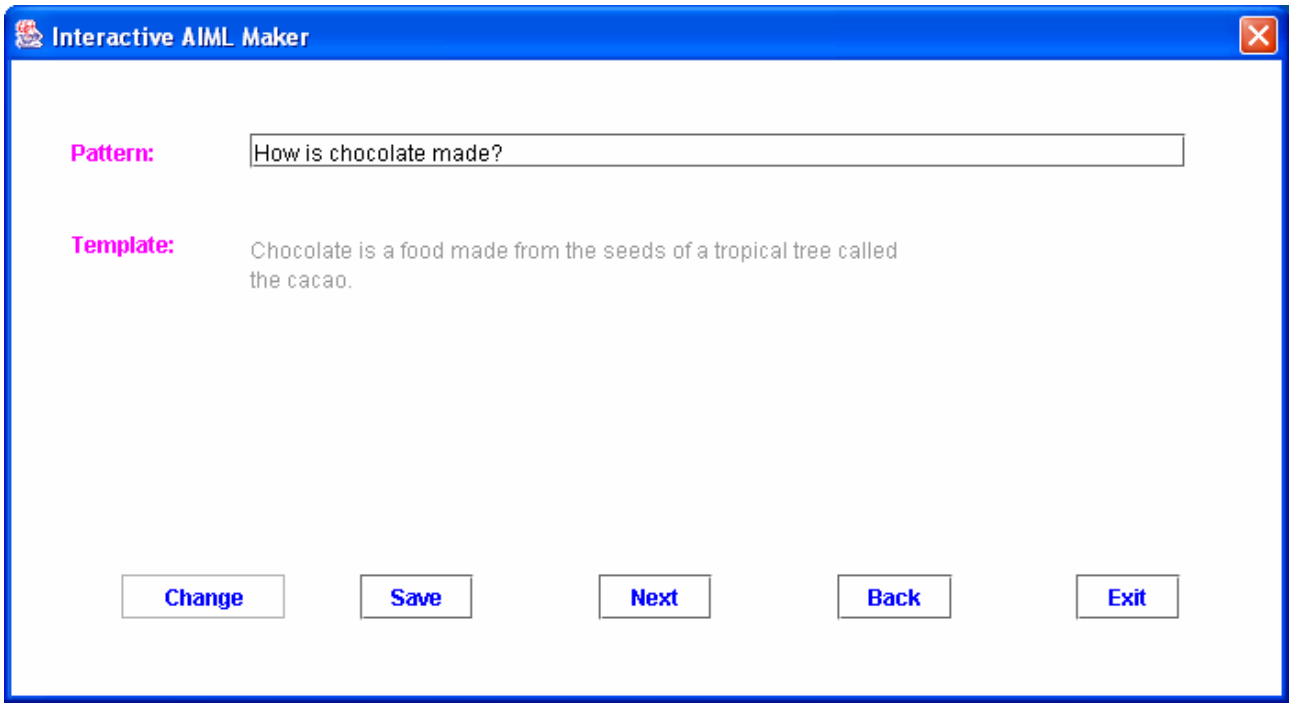

Figura 5: Esecuzione di un programma ALL che permette di estrarre da un documento le frasi contenenti le parole correlate semanticamente ad un determinato argomento

5. Conclusioni

Il linguaggio A.L.L. è stato creato per semplificare il processo di creazione di categorie AIML per i chat-bot con tecnologia ALICE. In questo lavoro sono state create primitive e strutture dati idonee sia per l'inserimento manuale di porzioni di dialogo da trasformare in categorie AIML, sia per sfruttare le tecniche di trattamento dei testi e di reperimento dell'informazione per semplificare la scelta dei contenuti informativi estraendoli automaticamente da file di testo.

6. Appendice: Grammatica del linguaggio A.L.L.

< *programma* >*::=* < *variabili* > <*funzioni*>< *main_program*>

<*variabili*>*::=* ***ε*** | **var** | **var** <*lista_variabili*>

<*lista_variabili*>*::=*<*dichiarazioni_variabili_tipo*>|<*lista_variabili*>
<*dichiarazioni_variabili_tipo*>

<*dichiarazione_variabili_tipo*>*::=* <*tipo*> <*lista_variabili_tipo*> *;*

<*lista_variabili_tipo*>*::=* <*id_var*> |<*lista_variabili_tipo*>, <*id_var*>

<*id_var*>*::=* **[a-zA-Z][a-zA-Z0-9"_"]***

<*funzioni*>*::=* ***ε*** | <*definizione_funzione*> |<*funzioni*> <*definizione_funzione*>

<*definizione_funzione*>*::=***SUB** <*tipo_funzione*> <*id_funzione*>
(<*lista_parametri*>***)*** **<***var_globali*> <*blocco_istruzioni*>

<*tipo_funzione*>*::=* **void** | <tipo>

<*id_funzione*>::= **[a-zA-Z][a-zA-Z0-9"_"]***

<*lista_parametri*>*::=* ***ε*** |<*tipo*> <*id_var*>|<*lista_parametri*>, <*tipo*> <*id_var*>

<*tipo*>*::=***"index"| "real" |"boolean" |"string" |"list_of" | "category"**

< *main_program*>*::=* **MAIN** <*blocco_istruzioni*>

< *blocco_istruzioni* >*::=***begin** <*contesto*> <*seq_ istruzioni* > **end**

<*seq_ istruzioni* >*:=* **ε**|<*istruzione*> *;*|<*seq_istruzioni*> <*istruzione*>*;*

<*istruzione*> ::= <*fun_call*>
|<*primitiva*>
|<*espressione*>
|<*costrutto_given*>
|<*selezione*>

|<*return*>

<*return*>*::=* **return | return (**<*espressione*>***)***

<*contesto*> ::= **mode (**<*scelta*>***);***

<*scelta*>*::=* **“local” |”remote” | “input” |”output”**

<*costrutto_given*>*::=* **given <***insieme*> ***{*****<***seq_istruzioni*>***}***

<*insieme*>***::=[*** <*operando*>, <*operando*> ***]|*** ***[***<*id_var*> **] | (** <*espressione*> **)**

<*selezione*>*::=***if** *(*<*espressione*>*)* ***{*** <*seq_istruzioni*>***}***
|**if** *(*<*espressione*>*)* ***{*** <*seq_istruzioni*>***}*** **else** ***{*** <*seq_istruzioni*>***}***

<*primitiva*> *::=* <*IOprimitiva*>
|<*TEXTprimitiva*>
|<*CATEGORYprimitiva*>
|<*LISTprimitiva*>
|<*GOOGLEprimitiva*>
|<*STRINGprimitiva*>
|<*GUIprimitiva*>
|<*LEXprimitiva*>
|<*DBprimitiva*>

<*IOprimitiva*>*::=* **read ()**
| **read (**<*operando*>**)**
| **write ()**
|**write (**<*operando*>**)**
| **confirm ()**
| **confirm (**<*operando*>**)**
| **chdir (**<*scelta*>**)**

<*TEXTprimitiva*>*::=* **newAimlFile ()**
|**updateFile (**<*operando*>**)**
|**closeAimlFile ()**

<*CATEGORYprimitiva*>*::=* **getAnswer (**<*operando*>**)**
|**getQuestion (**<*operando*>**)**
| **setAnswer (**<*operando*>**)**
| **setAnswer (**<*operando*>,<*operando*>**)**
|**setQuestion (**<*operando*>**)**

|**setQuestion (**<*operando*>,<*operando*>**)**
|**AimlTranslate (**<*operando***)**

<*LISTprimitiva*>*::=* **length (**<*operando*>**)**
|**addElement (**<*operando*>**)**
| **addElement (**<*operando*>,<*operando*>**)**
|**getElement (**<*operando*>,<*operando*>**)**
|**setElement(**<*operando*>,<*operando*>**)**
| **setElement(**<*operando*>,<*operando*>,<*operando*>**)**

<*GOOGLEprimitiva*>*::=* **googleQuery ()**

<*STRINGprimitiva*>*::=* **equal (**<*operando*>,<*operando*>**)**
| **equal (**<*operando*>**)**
| **modify ()**
| **modify (**<*operando*>**)**

<*GUIprimitiva*>*::=* **showInteractiveAiml (**<*operando*>**)**
| **showInteractiveFaq ()**
|**showInteractiveCategory()**

<*LEXprimitiva*>*::=* **extractSentences ()**
|**selectSentences(**<*operando*>**)**
|**extractQuestion (**<*operando*>**)**

<*Dbprimitiva*>*::=****getSimilarWords*** **(**<*operando*>,<*operando*>**)**

<*fun_call*>*::=*<*id_par*> ***( )***|< *id_par*> ***(***<*lista_par_call*> ***)***

<*id_par*>*::=***[a-zA-Z][a-zA-Zo-9"_"]***

<*lista_par_call*>*::=*<*espressione*>|<*lista_par_call*> <*espressione*>

<*espressione*>*::=*<*espressione*> <*op_binario*> <*espressione*>
| <*operando*>
| *!* <*espressione*>
| *! (* <*espressione*> *)*
| <*espressione*> ++
| <*id_var*> = <*espressione*>

<*op_binario*>*::=* <*op_rel*>| <*op_log*>| <*op_arit*>

*<op_arit> ::= - | + |/ |**

<op_log> ::= && | ||

<op_rel> ::= == | != | < | > | <= | >=

<operando> ::= <id_var>
|<cost_intera>
| <cost_stringa>
| <cost_boolean>
| <primitiva>
|<fun_call>
| (<espressione>)

<cost_intera> ::= [0-9]+

<cost_stringa>::= <stringa>

*<cost_boolean>::= “**true**” | “**false**”*

<cost_reale>::= [0-9]+ “.”[0-9]+

7. Bibliografia

1. N.J. Nilsson. Artificial Intelligence: a New Synthesis. Morgan Kaufman Publisher, Inc., 1998.
2. A.Turing. Computing machinery and intelligence. Mind, 1950.
3. B.Ribeiro-Neto R.Baeza-Yates. *Modern Information Retrieval*. Addison Wesley, 1999.
4. S.T. Dumais, T.K. Landauer. A solution to plato's problem: the latent semantic analysis theory of acquisition, induction and representation of knowledge. Psychological Review, 1997.
5. Artificial linguistic computer entity (A.L.I.C.E.) http://alice.sunlitsurf.com/alice/about.html
6. Artificial intelligence markup language (A.I.M.L.) http://alice.sunlitsurf.com/alice/aiml.html
7. Z. Harris. Structure distributionnelle. La grammaire,lectures, 1975.
8. F.T. Luk, G.H. Golub and M.L.Overton. A block lanczoz method for computing the singular values and corresponding singular vectors of a matrix. ACM Transactions on Mathematical Software, 7:149–169, 1981.
9. P.W. Foltz T.K.Landauer and D. Laham. An introduction to latent semantic analysis. Discourse Processes, 25:259–284, 1998.
10. G.W. Furnas; S.Deerwester; S.T.Dumais; T.K.Landauer and R.A.Harshman. Indexing by latent semantic analysis. The American Society for Information Science, 1990.
11. T.G. Kolda E.Chisholm. New term weighting formulas for the vector space method in information retrieval, 1999.
12. M.W.Berry. Large scale singular value computations. International Journal of Supercomputer Application, 1992.
13. PandoraWriter. http://www.pandorabots.com/
14. www.usenet.org
15. Vladimir A. Kulyukin, Utah State University, USA e Robin Burke, California State University, Fullerton, USA . Mining Free Text for Structure, 2002.