POLITECNICO DI MILANO – POLO REGIONALE DI COMO
FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE - CORSO DI LAUREA IN INGEGNERIA INFORMATICA

ELABORATO FINALE DI LAUREA DI I LIVELLO

PROTOTIPO DI NAVIGATORE WEB SEMANTICO AD INTERFACCIA VOCALE

Bozzolan Matteo 651512

Frigerio Mattia 653835

Relatore: Prof. Thimoty Barbieri

Sommario

Indice delle figure

1. SINTESI

L'attuale navigazione vocale sul web si basa essenzialmente su due modalità che consentono di esplorare i contenuti delle pagine in modo sequenziale (dall'alto verso il basso) o gerarchico (in base alla loro struttura). Entrambi gli approcci presentano notevoli carenze e difetti, risultando inefficienti e scomodi all'uso. Per questo motivo ci si è posti l'obiettivo di sviluppare una nuova modalità di navigazione che risolvesse i problemi riscontrati. L'idea è stata quella di utilizzare un approccio semantico, ovvero che sfruttasse i contenuti delle pagine dal punto di vista del significato.

Il progetto sviluppato permette di navigare un sito web tradizionale in modo semantico per mezzo di un'interfaccia completamente vocale.

L'introduzione della semantica nella navigazione è resa possibile grazie all'uso di un'ontologia che contiene in modo strutturato tutta l'informazione relativa al dominio di conoscenza proprio del sito considerato. L'applicazione, tramite un confronto con l'ontologia, attribuisce un significato ai contenuti della pagina che si sta visitando. In questo modo è possibile individuare gli argomenti trattati al suo interno, che vengono quindi presentati all'utente. La navigazione avviene tramite la scelta degli argomenti da parte dell'utente, in base alla quale l'applicazione presenta i relativi contenuti.

Il punto cardine di questo approccio è che la mappatura tra argomenti e contenuti non è predefinita, ma viene valutata contestualmente alla navigazione. Grazie a questa caratteristica, il prototipo sviluppato può essere applicato ad un qualsiasi sito web, purché si disponga di un'ontologia adeguata.

2. BACKGROUND

Il progetto che si vuole realizzare affonda le sue radici in due grandi aree tematiche: da un lato il *Web Semantico* e dall'altro l'*accesso vocale.*

Per quanto riguarda il Web Semantico ed in particolare la *navigazione semantica* gli argomenti più rilevanti sono:

- le ontologie
- RDF
- OWL

Per quanto riguarda l'accesso vocale invece

- le VUI
- VoiceXML

2.1. Web Semantico

Quando si parla di web semantico si intende proporre un web che possieda delle strutture di collegamenti più espressive di quelle attuali. Il termine *Semantic Web* è stato proposto per la prima volta nel 2001 da Tim Berners Lee. Da allora il termine è stato associato all'idea di un web nel quale agiscano agenti intelligenti: applicazioni in grado di comprendere il significato dei testi presenti sulla rete e perciò in grado di guidare l'utente direttamente verso l'informazione ricercata, oppure di sostituirsi a lui nello svolgimento di alcune operazioni. Un agente intelligente dovrebbe essere una applicazione in grado di svolgere operazioni come la prenotazione di un aereo per Parigi con arrivo in centro città prima delle 13.00. Il tutto estraendo informazioni da siti che definiscono l'aeroporto di Parigi in modo diverso (Paris, Charles de Gaulle, Orly) e deducendo, senza che sia

specificato nella query, che un arrivo per le 13.00 in centro implichi un arrivo in aeroporto diverso a seconda dell'aeroporto effettivamente selezionato.

Internet è un insieme di testi, un insieme molto vasto di documenti che descrivono dei contenuti. Per la verità questa non è una grossa novità, vasti corpi di testi sono esistiti fin dall'antichità (biblioteche). La novità di Internet (o meglio degli ipertesti che lo compongono) sta nel fatto che questi testi possono richiamarsi l'uno con l'altro, in modo molto rapido. Il link è l'elemento nuovo che l' HTML ha saputo proporre. Ai suoi esordi Internet era costituito unicamente di testi e indici ipertestuali di testi. Col tempo le cose si sono evolute aggiungendo molto velocemente contenuti, funzionalità e applicazioni che hanno dato vita a quello che è l'attuale Web, con la sua molteplicità di linguaggi e di strutture intrinseche.

Per un utente comunque questo rimane quasi del tutto nascosto. L'utente si orienta nel web grazie a due cose: la sua esperienza di navigazione e la capacità di evocazione che possono avere parole o espressioni chiave. L'esperienza è un aspetto molto importante di cui tutti ci serviamo: impariamo che determinati contenuti si possono reperire sotto determinati portali, impariamo che l'aspetto di un sito può dirci qualche cosa sul genere (formale o informale) delle informazioni. L'esperienza tuttavia è una cosa che viene spontaneamente e non è molto legata ad aspetti tecnici, al codice e alle applicazioni che costituiscono un sito.

Molte cose sono state scritte sul Web Semantico, come se fosse una tecnologia sostitutiva per il Web di oggi. In effetti il Web Semantico si realizza mediante cambiamenti incrementali, aggiungendo descrizioni machine-readable o meglio machine-understandable ai dati e ai documenti già presenti sul Web. Con descrizioni e con modi per connetterle, confrontarle e metterle in contrasto, è possibile costruire applicazioni, strumenti, motori di ricerca, agenti, tutto senza nessuna apparente modifica alle pagine Web.

Questa proposta ha affascinato molto la comunità informatica. Il W3C ha infatti attivato immediatamente un gruppo di lavoro e le università hanno aperto numerosi programmi di ricerca legati a questi temi. Si sono imposti subito degli standard, il più famoso dei quali è certamente *RDFS*, un linguaggio in sintassi XML per definire e esprimere ontologie. La *Semantic Web Activity* del W3C si basa sul lavoro svolto in altre W3C Activity come la XML Activity. Il suo obiettivo principale è quello di sviluppare tecnologie standard basate su *XML* che supportino la crescita del Semantic Web.

Alla base, *XML* fornisce un insieme di regole per creare vocabolari che possano portare struttura a dati e documenti sul Web. XML dà regole chiare per la sintassi; XML Schema quindi fornisce un metodo per comporre vocabolari XML. XML, pur fornendo una potente e flessibile sintassi per la

creazione di documenti strutturati, non permette di imporre limiti semantici sul significato di questi documenti.

RDF - Resource Description Framework - è un modo standard per fare semplici descrizioni. Quello che *XML* è per la sintassi, *RDF* lo è per la semantica: un insieme chiaro di regole per fornire semplici informazioni descrittive. *RDF Schema* fornisce quindi un metodo per combinare queste descrizioni in un singolo vocabolario. Come passo successivo serve un modo per sviluppare vocabolari specifici per un soggetto o dominio. Questo è il ruolo delle ontologie che hanno assunto come linguaggio di riferimento l'*OWL* (*Web Ontology Language).*

Risulta quindi evidente che l'enorme struttura che sta alla base del Web Semantico deve organizzarsi in maniera complessa. La gerarchia della semantica deve necessariamente tenere conto del fatto che ogni applicazione, pensata per uno specifico dominio applicativo, utilizza quasi necessariamente una sua propria ontologia. Nasce quindi la difficoltà di collegare fra loro le varie ontologie, conciliando le diverse semantiche utilizzate. L'operazione di collegare ontologie che definiscono stessi oggetti con una semantica differente è chiamata *Mapping.*

Un primo approccio potrebbe far pensare di risolvere il problema attraverso la definizione di una *Ontologia Globale* universalmente riconosciuta e condivisa. Questa soluzione richiederebbe un unico *mapping*: ogni diverso schema dovrebbe mapparsi sull'ontologia globale che funzionerebbe così come una lingua universale, traduzione universale dei vari linguaggi usati. Questo tipo di soluzione, si baserebbe su transazioni di comunicazione punto a punto e comporterebbe un accordo totale, ad ogni livello, del significato della semantica usata.

Ovviamente questa visione è praticamente improponibile. L'accordo su tutta la conoscenza non è pensabile e qualora fosse anche raggiunto sarebbe incapace di dare risposta a quei naturali mutamenti del dominio descritto, che inevitabilmente sorgono nel tempo. La via che si deve seguire deve essere perciò diversa. L'idea diffusa è quella che si lavorerà su un'architettura di tipo federativo.

Per capire come possa funzionare un'organizzazione federativa è necessario focalizzare alcuni punti cruciali, che derivano dai seguenti interrogativi:

- che tipo di mapping si vuole realizzare?

  Il *mapping* può essere di due tipi automatico o manuale. Osservando lo scenario attuale con realismo si può dire che la soluzione che dovrebbe diffondersi sarà, almeno in fase iniziale, un compromesso tra queste due alternative. In una prima fase dei tool leggeranno grosse quantità di dati e realizzeranno dei cluster che un esperto dovrà poi analizzare e validare.

- che tipo di transazioni vanno applicate?

Per quanto riguarda le tipologie di *transazioni* sono possibili due strade. Il problema è quando scegliere come mappare le diverse ontologie che definiscono i dati d'interscambio. Il servizio che si occupa delle transizioni potrebbe conoscere a priori come mapparle oppure potrebbe calcolarlo al momento. Se da un lato conoscere a priori quali ontologie usare è un vantaggio a livello computazionale, dall'altro permettere di eseguire dei calcoli al momento lascia spazio ad un set di risultati simili con lievi sfumature tra loro. Alla capacità cioè di riportare ad una stessa struttura strutture non identiche ma che presentano un sufficiente grado di vicinanza.

- che accordo deve esserci a priori (standard)?

Riguardo al problema degli *standard*, il modello adottabile potrebbe essere quello dell'*interlingua*. Il presupposto è che alcuni gruppi di ontologie relative a domini e a scopi simili o collegati tra loro parzialmente, condividono una stessa ontologia, dette appunto *interlingua*. Questa soluzione permette di ridurre drasticamente il numero di mappature perché un gruppo di n di ontologie non dove essere mappato (n * n-1)/k volte ma solo una volta per ogni singola ontologia (n volte), riducendo drasticamente l'ordine di complessità del problema. L'idea è quella di mappare ogni ontologia con l'interlingua, la quale fornirebbe l'aggancio con tutte le altre. Questo tipo di soluzione possiede anche il vantaggio di essere in buon accordo con la prassi con la quale si costruisce la collaborazione tra organizzazioni. Inizialmente alcuni gruppi di organizzazioni si accordano per creare un standard condiviso, quando questo standard si consolida e afferma altre realtà si associano allo standard. Questo modello di organizzazione dell'architettura deve esser immaginato replicato in modo iterato, nel senso cioè che un gruppo di interlingue possono a loro volta collegarsi ad una interlingua in grado di accordare la loro semantica.

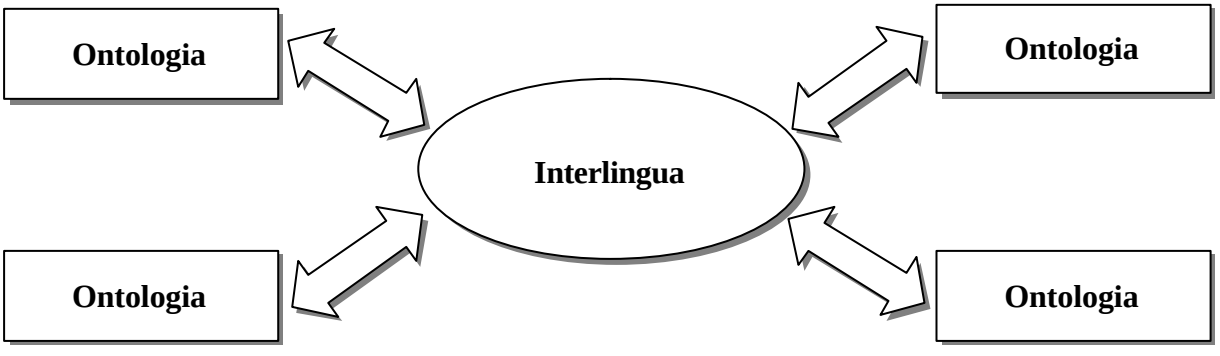

Figura 1 - Modello dell'interlingua per la condivisione di ontologie

In uno scenario come quello descritto si può capire facilmente l'importanza che avranno i *Web Services.* È infatti indispensabile possedere un livello che sia in grado di gestire la scelta dell'ontologia giusta a seconda dell'operazione da eseguire, del dominio da esplorare. L'uso dei *Web Services* per organizzare la modularità apre anche la strada ad un nuovo importantissimo problema.
E' possibile avere viste di diversa profondità sulle ontologie? Se a certi livelli abbiamo applicazioni che debbono mappare ontologie diverse, e quindi operare delle computazioni, non sarà sempre utile considerare l'intera ontologia. Potremmo voler trattare unicamente con il livello più superficiale di questa ontologia. In pratica la definizione, la profondità con cui considerare l'albero dell'ontologia potrebbe variare in base alle necessità di computazione.

2.2. Ontologie

2.2.1. Definizione di ontologia

Le definizioni di ontologia che si possono trovare nella letteratura riguardante l'intelligenza artificiale sono molteplici e spesso una ne contraddice un'altra. Per dare un'idea per quanto possibile generale si può dire che un'ontologia è una descrizione formale di:

- concetti all'interno di un dominio di riferimento (*classi* o *concetti*)
- proprietà di ciascun concetto che descrivono caratteristiche ed attributi del concetto stesso (*ruoli* o *proprietà*)
- restrizioni sui ruoli (*facets* o *role restrictions*)

Un'ontologia insieme ad un insieme di istanze di classi (*individuals*) costituisce una base di dati. In realtà c'è una sottile linea che indica quando termina l'ontologia ed inizia la base di dati.
Un'ontologia definisce inoltre un vocabolario comune per l'interscambio di informazioni all'interno di un dominio specifico, rendendo possibile anche l'interpretazione da parte di un calcolatore di concetti di base e relazioni tra di essi.

L'elemento fondamentale delle ontologie sono le *classi*, le quali descrivono i concetti presenti nel dominio di riferimento. L'esempio tipico che si cita è quello dei vini.
La classe astratta "vini" rappresenta tutti i vini ed ogni particolare vino (quel vino specifico con quella particolare bottiglia sul tavolo di fronte a noi durante la cena di ieri sera) rappresenta un istanza di questa classe. Ogni classe può avere delle *sottoclassi* che rappresentano concetti che sono più specifici rispetto alla superclasse a cui appartengono. Per esempio si può suddividere la classe dei vini in bianchi, rossi e rosé piuttosto che in frizzanti o non frizzanti. Gli slot delle proprietà descrivono delle caratteristiche di una classe o di una sua istanza (ad esempio, per restare all'esempio dei vini, il gusto, la corposità, il livello zuccherino, il produttore, ecc.).

2.2.2. Perché sviluppare un'ontologia

I motivi principali che possono spingere a realizzare un'ontologia sono principalmente:

- *condividere il sapere comune sulla struttura delle informazioni tra persone o agenti software*

  E' in genere il goal più comune nello sviluppo di ontologie e può essere ben spiegato con un ulteriore e classico esempio. Consideriamo alcuni differenti siti web che contengono informazioni mediche o che forniscono servizi di e-commerce di strumenti medicali. Se questi siti condividono e sono pubblicati sulla base della stessa ontologia, allora degli agenti software possono estrarre ed aggregare informazioni da questi siti e usarle per rispondere alle query di utenti piuttosto che come dati di input per altre applicazioni.

- *rendere possibile il riuso del dominio della conoscenza*

  Un esempio esplicativo può essere quello del concetto di tempo. Molti modelli legati a domini differenti tra loro necessitano infatti di poter rappresentare la nozione di tempo. Questa include a sua volta alcune nozioni di base come quelle di intervallo di tempo, di istante temporale, di misura di tempo, ecc. Se un determinato gruppo di ricercatori sviluppa in dettaglio questa ontologia, altri potrebbero in seguito semplicemente riutilizzare questa all'interno dei propri domini di conoscenza. Inoltre se si deve costruire un'ontologia piuttosto grossa, si possono integrare altre ontologie preesistenti che mappano porzioni limitate del dominio in questione,

piuttosto che usare un'ontologia generica ed estenderla per descrivere il proprio particolare dominio.

- *stabilire assunzioni esplicite sul dominio considerato*

  Questo aspetto mira a semplificare i cambiamenti da apportare alle assunzioni fatte sul dominio in seguito ad una variazione della conoscenza. Ad esempio le assunzioni espresse in codice di un certo linguaggio di programmazione sono spesso difficili da individuare, da comprendere ma soprattutto da modificare, in particolar modo per chi non ha esperienze approfondite di programmazione. Inoltre specifiche esplicite sul dominio di conoscenza sono utili per i nuovi utenti che vogliono capire ed imparare che cosa significano i termini nel dominio.

- *separare la conoscenza del dominio dalla conoscenza operativa*

  E' un altro uso comune di un'ontologia. Ad esempio si può descrivere una procedura di configurazione di un prodotto a partire dai suoi componenti di base secondo determinate specifiche e implementare una funzione che svolga queste operazioni indipendentemente dal prodotto e dai componenti stessi. Così si può poi sviluppare un'ontologia di componentistica per PC e applicare l'algoritmo per creare degli ordini di produzione di PC. Allo stesso modo si può riutilizzare lo stesso algoritmo per gestire gli ordini di ascensori semplicemente a patto di fornire al programma un'ontologia di componentistica per ascensori.

- *analizzare la conoscenza*

  E' possibile compiere questo passo una volta disponibile una descrizione dichiarativa dei termini dell'ontologia. L'analisi formale dei termini è resa estremamente preziosa se si attende sia al riuso delle ontologie che alla loro estensione.

2.3. RDF e RDFS

E' chiaro che l'automatizzazione del Web risulta praticamente impossibile restando ancorati alla sua architettura originaria, in cui tutte le informazioni erano in linea di principio *machine-readable*, ma non *machine-understandable*. La soluzione al problema sembra quindi venire dai *metadati.* L'uso efficace dei *metadati*, tuttavia, richiede che vengano stabilite delle convenzioni per la *semantica*, la *sintassi* e la *struttura*. Le singole comunità interessate alla descrizione delle loro risorse specifiche definiscono la semantica dei *metadati* pertinenti alle loro esigenze.
La sintassi, cioè l' organizzazione sistematica dei *data element* per l'elaborazione automatica, facilita lo scambio e l' utilizzo dei *metadati* tra applicazioni diverse.
La struttura invece può essere vista come un vincolo formale sulla sintassi, per una rappresentazione consistente della semantica.
RDF (*Resource Description Framework*) è lo strumento base per la codifica, lo scambio e il riutilizzo di metadati strutturati, e consente l' interoperabilità tra applicazioni che si scambiano sul Web informazioni machine-understandable. Alcuni tra i settori più significativi nei quali *RDF* può essere utilizzato sono:

- descrizione del contenuto di un sito Web, di una pagina o di una biblioteca digitale
- implementazione di *intelligent software agent*, per lo scambio di conoscenza e un utilizzo migliore delle risorse Web
- classificazione di contenuti per applicare criteri di selezione
- descrizione di un insieme di pagine che rappresentano un singolo documento a livello logico

Bisogna precisare che l'*RDF* non descrive la semantica, ma fornisce una base comune per poterla esprimere, permettendo di definire la semantica dei tag *XML*.
RDF è costituito da due componenti:

RDF Model and Syntax: definisce il *data model RDF* e la sua codifica *XML*
RDF Schema (RDFS): permette di definire specifici *vocabolari* per i metadati.

2.3.1. RDF Data Model

RDF fornisce un modello per descrivere le risorse. Tutte le risorse hanno delle proprietà (o anche attributi o caratteristiche). *RDF* definisce una risorsa come un qualsiasi oggetto che sia identificabile univocamente mediante un Uniform Resource Identifier (URI). Il data model RDF è molto semplice, ed è basato su tre tipi di oggetti:

- *Resources*

  Qualunque cosa descritta da un'espressione *RDF* viene detta risorsa (*resource*). Una risorsa può essere una pagina Web, una sua parte piuttosto che un elemento *XML* all' interno del documento sorgente. Una risorsa può anche essere un' intera collezione di pagine web, un oggetto non direttamente accessibile via Web (per es. un libro, un dipinto, etc.). Le risorse sono sempre individuate da un URI.

- *Properties*

  Una *property* è un aspetto specifico, una caratteristica, un attributo o una relazione utilizzata per descrivere una risorsa. Ogni proprietà ha un significato specifico, definisce i valori ammissibili, i tipi di risorse che può descrivere e le sue relazioni con altre proprietà. Le proprietà associate alle risorse sono identificate da un nome, e assumono dei valori.

- *Statements*

  Una risorsa, con una proprietà distinta da un nome e un valore della proprietà per la specifica risorsa, costituisce un *RDF statement*. Uno *statement* è quindi una tupla composta da un *soggetto* (risorsa), un *predicato* (proprietà) e un *oggetto* (valore). L'oggetto di uno *statement* (cioè il *property value*) può essere un'espressione (sequenza di caratteri o qualche altri tipi primitivi definito da XML) oppure un' altra risorsa.

Graficamente, le relazioni tra Resource, Property e Value vengono rappresentate mediante *grafi etichettati orientati*, in cui le risorse vengono identificate come nodi (graficamente delle ellissi), le proprietà come archi orientati etichettati e i valori corrispondenti a sequenze di caratteri come rettangoli. Una rappresentazione grafica di una generica descrizione RDF può essere la seguente:

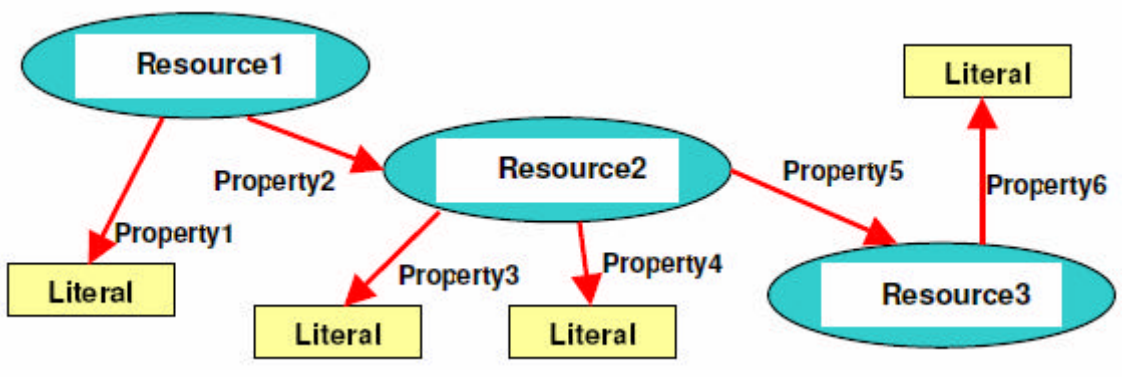

Figura 2 - Esempio di rappresentazione grafica di una descrizione RDF

Un insieme di proprietà che fanno riferimento alla stessa risorsa viene detto descrizione (*description*).

2.3.2. RDF Schema

Il *data model RDF* permette di definire un modello semplice per descrivere le relazioni tra le risorse, in termini di proprietà identificate da un nome e relativi valori. Tuttavia *RDF data model* non fornisce nessun meccanismo per dichiarare queste proprietà, né per definire le relazioni tra queste proprietà ed altre risorse. Tutto ciò è reso possibile da *RDF Schema* che permette di definire dei vocabolari ovvero l'insieme delle proprietà semantiche (in generale specifiche per una particolare comunità). *RDF Schema* permette quindi di definire significato, caratteristiche e relazioni di un insieme di proprietà, compresi eventuali vincoli sul dominio e sui valori delle singole proprietà. Inoltre, implementando il concetto di classe e sottoclasse, consente di definire gerarchie di classi, con il conseguente vantaggio che agenti software intelligenti possono utilizzare queste relazioni per svolgere i loro compiti.

2.4. OWL

OWL (*Web Ontology Language*) è un linguaggio per definire ontologie strutturate basate sul Web che permettano maggiore integrazione ed interoperabilità di dati tra applicazioni. I primi ad adottare questo standard (o meglio i suoi prototipi) sono bioinformatici e comunità mediche, gruppi industriali e governi.
OWL mette insieme la ricerca di diversi gruppi che stavano sviluppando linguaggi per esprimere espressioni ontologiche sul web . Esso trae le sue origini da due grandi sforzi di ricerca:

- una bozza di linguaggio noto come *DARPA* (*Agent Markup Language Ontology* o *DAML-ONT*)
- *Ontology Interface Layer* (*OIL*), sviluppato da ricercatori europei con il supporto della Commissione Europea.

Da qui un gruppo di ricercatori ad hoc forma il *Joint US/EU committee on Agent Markup Languages* e pubblica una nuova versione di questo linguaggio che unisce *DAML* con *OIL* dando vita al *DAML+OIL* che è proprio il linguaggio che, in seguito all'intervento di standardizzazione del W3C, prende il nome di *OWL*.
OWL si basa su *RDF Model and Schema* e aggiunge rispetto a questi un vocabolario più ampio per descrivere proprietà e classi: tra le altre, relazioni tra classi (ad esempio disgiunzione), cardinalità (ad esempio "esattamente uno"), uguaglianza, tipizzazione più ricca di proprietà, caratteristiche di proprietà (ad esempio simmetria) e classi enumerate.
OWL rappresenta un passo significativo sulla strada che dovrà portare dall'attuale Web al Web Semantico. I Web attuale è pensato principalmente per un browser utilizzato da una persona. Senza un uomo che legga e capisca il testo con le relative figure e i contenuti multimediali, la gran parte delle informazioni presenti sul Web sarebbero praticamente inutilizzabili. Lo scopo del Web Semantico è di rendere tutte queste informazioni interpretabili da un'agente software oltre che da un umano. OWL ovvia a questo inconveniente utilizzando URI per nominare le risorse e il linking fornito da RDF per aggiungere alle ontologie le seguenti capacità:

- possibilità di essere distribuite tra più sistemi
- scalabilità per le necessità del Web
- compatibilità con gli standard Web per quanto riguarda l'accessibilità e l'internazionalizzazione
- riusabilità ed estendibilità

2.4.1. Le tre versioni di OWL

Le ontologie realizzate in sintassi OWL possono essere catalogate in tre diverse specie o sottolinguaggi, ognuno dei quali con uno specifico grado di espressività:

- *OWL-Lite*

  E' il sottolinguaggio sintatticamente più semplice e meno espressivo. Esso è stato pensato per essere utilizzato in situazioni che presentano una gerarchia delle classi e vincoli piuttosto semplici.

- *OWL-Full*

  E' il più espressivo dei tre sottolinguaggi ed è pensato per essere utilizzato in situazioni dove l'altissima espressività è più importante che non la garanzia di completezza computazionale o di sicura decidibilità del linguaggio. E' quindi impossibile un ragionamento inferenziale automatizzato sulle ontologie basate su *OWL-Full.*

- *OWL-DL*

  Rappresenta un compromesso tra i due precedenti sottolinguaggi ed è basato sulle *Description Logics* (da cui deriva il suffisso *DL*). Le *DL* sono un frammento logico decidibile del primo ordine (ovvero la loro computazione termina in un tempo finito) e supportano quindi un ragionamento automatizzato. E' inoltre possibile computare automaticamente un classificazione gerarchica e controllare le inconsistenze eventualmente presenti nell'ontologia.

In generale *OWL-DL* può essere considerato un'estensione di *OWL-Lite* e *OWL-Full* un estensione di *OWL-DL*.

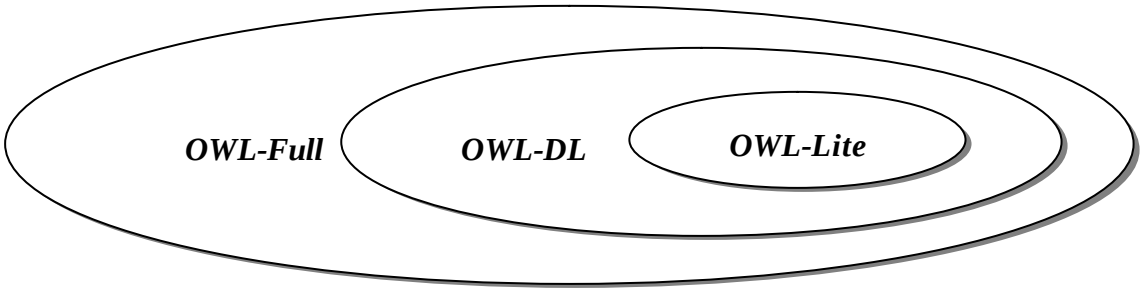

Figura 3 - Relazioni tra le tre tipologie di linguaggio OWL

2.5. VUI

Una VUI (Voice User Interface) è la via attraverso la quale un sistema di riconoscimento del parlato può interagire con un utente, nello stesso modo in cui, quotidianamente, una persona comunica con altre persone.

I lontani progenitori delle VUI sono le GUI (Graphic User Interface), create in gran parte per rendere invisibili all'utente le funzionalità delle applicazioni a livello basso e per fornire invece un ambiente user-friendly molto intuitivo e basato sulla capacità di osservazione. Negli ambienti completamente vocali tutto questo non esiste: la voce e i suoni hanno le stesse funzioni che le immagini, la grafica e le animazioni hanno nelle GUI.

Il riconoscimento delle parole e dei comandi può in un certo senso essere associato al *link* delle tradizionali applicazioni web. Come quando si clicca su un link o su un pulsante, il browser porta ad un'altra pagina o compie una determinata operazione, così quando un comando vocale viene riconosciuto dal sistema, viene presentata all'utente l'informazione associata o viene svolta una determinata operazione.

In una VUI l'interazione è interamente basata sulla sola voce ed è per questo che si avvicinano molto al più naturale e diffuso modo di comunicazione umana: il parlato.

Nella tabella che segue si riporta un confronto schematico e sintetico delle principali differenze tra una GUI e una VUI, ponendo l'attenzione su tre degli aspetti più importanti dal punto di vista dell'utente: lo sforzo mentale, la navigazione e l'accesso ai contenuti e la gestione degli errori.

Argomento	**GUI**	**VUI**
Sforzo mentale	Offrono spesso grandi quantità di informazioni, ma gli schemi di impaginazione e navigazione sono ricorrenti e così facilitano l'accesso diretto alle informazioni desiderate. È l'utente a decidere per quanto tempo visualizzare il documento e leggere i contenuti, oppure se rileggere le pagine già visitate, adattandosi perfettamente alle proprie esigenze	Sono temporanee, nel senso che i messaggi ascoltati si dimenticano in fretta e la ripetizione è un processo difficoltoso e che comunque richiede del tempo. L'informazione nelle applicazioni vocali è effimera esattamente come qualsiasi conversazione tra persone e, insieme al fatto che l'interlocutore è solitamente distratto con lo sguardo, rende difficile comunicare grandi quantità di dati.

Navigazione e accesso ai contenuti	Usano comunemente barre di navigazione, collegamenti auto-descrittivi, e tasti “Avanti” e “Indietro” che rendono semplice capire dove ci si trova e dove si può andare. Possono offrire grandi quantità di opzioni e contenuti sfruttando richiami permanenti sullo schermo (come barre di navigazione, icone o impaginazioni opportune) oltre a elementi visivi accattivanti (grafici, animazioni o colori)	Non è semplice ricordare la propria posizione nell’applicazione: ascoltare la lenta lettura di un testo o la declinazione di opzioni offre maggiori occasioni di distrazione. È richiesta un’attenta progettazione: la navigazione deve essere sempre coerente, elementare e chiara, altrimenti l’utente perderà in breve l’orientamento e navigherà “alla cieca” nel tentativo di trovare un menu che lo riporti dove desiderava. È difficile poi non leggere i contenuti non interessanti, date le limitate possibilità di controllo diretto.
Gestione dell’errore	Se si sbaglia la navigazione si può sempre tornare indietro alla pagina in cui si è commesso l’errore e cambiare la scelta. Per quanto riguarda gli errori di digitazione ci si può correggere come un comune editor di testi prima dell’invio dei dati o comunque ritornando alla pagina precedente	Gli errori non sono dovuti solo all’utente, ma anche al sistema stesso che può interpretare erroneamente anche l’input corretto. È necessario quindi che le applicazioni siano progettate per facilitare la correzione e minimizzare i rischi d’errore

Le motivazioni che portano alla scelta di un’interazione di tipo vocale sono essenzialmente due:

- l’accesso vocale come parte della multimodalità

  L’espressione parlata è la modalità in generale più utilizzata nella comunicazione tra persone. Questo rende più semplice per un utente imparare delle operazione relative a certi servizi con una attivazione vocale, piuttosto che tramite degli input da mouse e tastiera (almeno pur un utente senza specifiche abilità nell’uso del calcolatore). Inoltre come modalità sia di input che di output, il parlato presenta molteplici vantaggi. In primo luogo un’interazione di tipo vocale non interferisce con altre azioni come ad esempio guidare una vettura, essendo completamente

hands and eyes free. Secondariamente tale tecnologia si prestano ad essere integrate facilmente in applicazioni nativamente basate sul suono, come il broadcasting radio o le applicazioni musicali piuttosto che i messaggi di voice-mail. Terzo, ma non ultimo, è il fatto che la tecnologia TTS (*text-to-speech*) implica che le informazioni possono essere trasmesse molto facilmente all'utente finale. Inoltre, con un appropriato design, i comandi vocali possono essere strutturati in modo da essere facilmente imparati e ricordati dall'utente. Essi non occupano un spazio proprio, non necessitano di una collocazione spaziale su uno schermo e non sono legati a nessun tipo di supporto fisico per la loro attuazione.

- l'accesso vocale come mezzo per abbattere le barriere elettroniche

  Rendere il web accessibile a tutti gli utenti significa permettere l'accesso alle informazioni anche a persone con disabilità fisiche (quali limitazioni motorie e visive) o a chi dispone di strumenti hardware e software limitati[1].

L'aspetto più critico di ogni VUI è la *user experience.* Quando si parla a qualcuno per la prima volta ci si aspetta che l'interlocutore comprenda ciò che gli viene detto. Per ogni persona alla quale si rivolge la parola, non è necessario un training su cosa dire, quando parlare o quando stare in ascolto, ma semplicemente si assume di dialogare con una persona che utilizza lo stesso linguaggio e nello stesso modo. Questi processi comunicativi fanno parte naturalmente di tutti i linguaggi e di tutte le culture. Se gli attori di un dialogo seguono queste regole implicite della comunicazione verbale, allora ogni persona può facilmente comunicare con un'altra. Una buona VUI dovrebbe quindi seguire queste regole comunicative di base, rendendo ovviamente la tecnologia trasparente all'utente e creando un'interazione tra la voce e i dati il più possibile naturale e vicino al dialogo quotidiano.
Ovviamente, insieme al motore di riconoscimento del parlato, la VUI è l'elemento più importante di un'applicazione vocale. Le interfacce utente di tipo vocale devono essere semplici da controllare e devono rispondere ad alcuni requisiti fondamentali, tra cui il rispetto delle regole basilari della comunicazione, l'immediatezza nell'uso (senza necessità di training) e l'appropriata corrispondenza alle richieste effettuate.

[1] Secondo le direttive WAI (Web Accessibility Initiative) del W3C

Per rendere possibile tutto ciò i punti principali su cui deve puntare un'applicazione vocale, dal punto di vista del motore di sintesi e riconoscimento del parlato) sono i seguenti:

- ✍ variabilità dei modi di parlare

  Persone differenti parlano la stessa lingua in modi leggermente differenti l'uno dall'altro e necessariamente non pronunceranno una parola in modo uguale. L'interpretazione di questa variabilità ha dato vita ad un'analisi molto complessa che deve gestire le pause naturali, la velocità del flusso delle parole e i cambiamenti di volume della voce.

- ✍ potenza di calcolo e continuità del riconoscimento del parlato

  Le tecniche di riconoscimento del parlato sono molto intensive dal punto di vista computazionale ed inoltre devono avvenire in tempo reale. Questo richiede un potere di calcolo piuttosto elevato (che non era commercialmente disponibile fino a molti anni or sono). Quando una persona parla a velocità naturale, è difficile distinguere (soprattutto per un processamento real-time) a quali parole sono associati certi particolari suoni percepiti, poiché spesso, nel modo naturale di parlare, non ci sono pause tra le parole (o almeno non tra tutte), se non in presenza della punteggiatura.

- ✍ estrazione del significato

  Poche applicazioni di riconoscimento del parlato sono veramente in grado di determinare il significato delle parole. La qualità dell'interpretazione del parlato dipende dall'abilità del motore di riconoscimento di scegliere il *matching* migliore tra i suoni intercettati e la lista delle possibili parole ammesse dalla grammatica. Un passo successivo e più avanzato è quello di estrarre da queste parole un significato. A causa delle diverse pronunce e dei diversi e numerosi termini che sono usati per esprimere lo stesso concetto, una comprensione completa di tutti i fattori umani è veramente critica per interpretare in modo adeguati i significati.

- rumore di fondo

  Il rumore di fondo (dovuto sia a fattori ambientali, in base alla rumorosità del luogo in cui ci si trova, sia a fattori tecnici, in base alle caratteristiche delle apparecchiature di trasduzione del segnale) è in generale difficile da filtrare. Lo sviluppo di microfoni sempre migliori ha notevolmente aiutato, ma il vero obiettivo da raggiungere è quello di isolare la voce da tutti i fattori estranei, come ad esempio il brusio, la musica,ecc.

Oltre a questi aspetti, legati solamente al motore di riconoscimento e sintesi del parlato, il buon funzionamento di un'applicazione vocale dipenderà anche, come già detto, dalla struttura della sua interfaccia. Queste due componenti lavorano in stretta relazione tra loro per poter dar luogo ad una user experience piacevole ma soprattutto efficiente, rendendo la navigazione il più semplice possibile.
Il motore di sintesi e riconoscimento del parlato è responsabile della conversione di suoni udibili in parole e frasi (e viceversa), mentre la VUI attribuisce a queste un significato e guida l'utente attraverso un'esperienza interattiva.

Gli attuali approcci di progettazione di una VUI sono essenzialmente due:

- VUI basate sui menu (*menu-based*)

  Sono quelle più tradizionali in cui, per la natura gerarchica che è insita nella VUI, il sistema accetta solamente una lista limitata di risposto in base al punto della struttura in cui si trova l'utente. Esse offrono un percorso di tipo step-to-step per lo svolgimento delle funzioni di base dell'applicazione. Tale fatto deriva necessariamente dal fatto che tali interfacce nascono per le applicazioni telefoniche, nelle quali la comunicazione è di tipo touch-tone (DTMF). La transizione da questo tipo di tecnologia a quella del riconoscimento del parlato ha spesso avuto come prodotto un design limitativo delle interfacce, poiché la comunicazione via tastiera a toni è molto più sistematica e metodica rispetto alle modalità con le quali le persone interagiscono tra loro.

✍ VUI colloquiali (*conversational*)[2]

Con una VUI di tipo colloquiale l'utente parla con un'applicazione usando un linguaggio del tutto naturale e spontaneo ed è la modalità che più rispecchia le aspettative di una persona che si accinge all'uso delle tecnologie vocali. Con una VUI colloquiale non è necessario alcun periodo di training e l'utente può raggiungere i suoi obiettivi comunicativi velocemente ed in modo efficiente. Il grado di soddisfacimento di una VUI di questo tipo è nettamente più elevato rispetto a quello di una basata sui menu. Ciò è dovuto al fatto che un sistema basato sui VUI colloquiali è, in linea generale:

- *facile da usare*: non è necessario seguire una gerarchia rigida, ma basta dire quello che si vuole e quando si vuole, abbreviando così il tempo di raggiungimento degli obiettivi da conseguire.
- *naturale*: le frasi possono essere dette in linguaggio quotidiano, aumentando così il consenso dell'utente.
- *efficiente*: si possono conseguire più obiettivi ed in minor tempo, aumentando la produttività operativa.
- *interpersonale*: la conversazione naturale può essere usata per accedere al sistema e controllarlo. A sua volta il sistema parla in modo chiaro e facile da capire. In questo modo si riduce drasticamente il forte distacco esistente tra la persona e la macchina, spesso troppo asettica e necessariamente impersonale.
- *controllato dall'utente*: la voce dell'utente ha la facoltà di controllare in ogni momento le azioni del sistema, eliminando così il rischio di disperdersi nei sottomenu.
- *sensibile al contesto*: una VUI colloquiale fornisce degli aiuti appropriati alle necessità impellenti dell'utente, aumentando la sua capacità di risolvere istantaneamente gli eventuali problemi che gli si possono presentare.

.

[2] Secondo le direttive proposte dall' *International Engineering Consortium*

2.6. VoiceXML

2.6.2. La storia

VoiceXML (*Voice eXtensible MarkupLanguage*) conosciuto anche come *VXML*, è il linguaggio, derivato da *XML*, utilizzato per gestire applicazioni vocali per mezzo delle quali la comunicazione uomo-computer (o meglio utente-sistema) non si realizza solo per mezzo di un interfaccia grafica (links, bottoni, etc, accessibili per mezzo di mouse e tastiera) ma anche e soprattutto mediante un interfaccia vocale (*VUI*). *XML* e' un linguaggio che permette di creare e gestire documenti e dati sul web mediante la creazione di vocabolari arbitrari formalmente definiti come *schema*. Un *XML Schema* può definire un documento particolare, un altro un' equazione matematica e così via. *VoiceXML* non e' altro che un *XML Schema* i cui scopi sono quelli di rendere i contenuti e le informazioni Internet consultabili ed accessibili a mezzo input/output vocali. In pratica *VoiceXML* permette di gestire un vero e proprio dialogo vocale tra utente e sistema, senza la necessità di intermediazioni visive o di puntatori. Una delle prime società che si è interessata allo sviluppo di tali interfacce è stata IBM che qualche anno fa sviluppò un apposito linguaggio che chiamò *SpeechML*. Quasi parallelamente anche AT&T e Lucent Technologies ne definirono dei propri denominati con lo stesso nome *PML* (*Phone Markup Language*) anche se diversi tra di loro e Motorola un ulteriore che denomino *VoxML*. Ben presto, come sempre accade in questi casi, tutti si accorsero che era necessario unire gli sforzi per dar vita ad un unico linguaggio: nacque cosi' il *VoiceXML* ed un' organizzazione, il VoiceXML Forum, sempre per iniziativa di AT&T, IBM, Lucent e Motorola, atta a sviluppare e promuovere il nuovo linguaggio standardizzato dal W3C con le 3 specifiche che si sono susseguite nel tempo:

- VoiceXML 0.9 del 17 Agosto 1999
- VoiveXML 1.0 del 7 Marzo 2000
- VoiceXML 2.0 del 24 Aprile 2002 (la più recente)

Al contrario dell'*HTML* o di altri linguaggi pensati per pagine Web strettamente visuali che mancano completamente di controlli sull'interazione tra utente e applicazione necessarie per un'interfaccia di tipo vocale, il *VoiceXML* è stato appositamente concepito per fornire il completo

controllo sul *dialogo parlato*. L'applicazione e l'utente interagiscono dialogando: l'applicazione formula delle richieste (*prompt*) all'utente e l'utente risponde.
In particolare un documento *VoiceXML* fornisce:

- prompt vocali (sintesi del parlato)
- output di file audio
- riconoscimento di parole e frasi
- riconoscimento di toni DTMF (visto che il suo scopo principale è quello di fornire applicazioni Web via telefono)
- registrazione di input vocali
- controllo del flusso del dialogo
- controlli telefonici (inoltro di chiamata e chiusura della linea)

2.6.2. Concetti base

VoiceXML è un linguaggio di *markup* basato su sintassi *XML*. Il documento, per essere fruito dall'utente, deve essere elaborato da un apposito interprete (come del resto avviene anche con l'*HTML* e i *browser*) al quale sono delegate tutte le operazioni da eseguire su ciascuno dei tag. L'ordine di esecuzione del documento è di tipo sequenziale dall'alto verso il basso (dal primo all'ultimo tag nell'ordine in cui si presentano).
Un documento VoiceXML è strutturato in blocchi identificati da un nome univoco che permette di effettuare dei salti da uno all'altro.
I *tag* principali possono essere suddivisi in tre categorie funzionali:

- *Tag* di *dialogo* verso l'utente

  Questi *tag* hanno la funzione di presentare all'utente delle informazioni o dei messaggi. Questi messaggi sono presenti sotto forma di contenuto all'interno dei tag. Il contenuto è poi letto dall'interprete VoiceXML all'atto dell'elaborazione dei tag stessi.
  Appartengono a questa categoria i tag <*block*> e <*prompt*>.

Esempio

```
<vxml version=”1.0”>
   <form id=”main”>
        <block>
             Questo è il messaggio di esempio letto all’utente.
        </block>
   </form>
</vxml>
```

✍ *Tag* di *input* di informazioni da parte dell’utente

Questi tag hanno la funzione di ottenere informazioni in ingresso da parte dell’utente. Sull’ingresso ricevuto viene effettuato un riconoscimento del parlato in base a particolari regole definite appositamente. Appartengono a questa categoria <*choice*>, <*option*> *e* <*field*>.

Esempio

```
<vxml version=”1.0”>
   <menu id=”main”>
        <prompt>
             Scegli tra acqua o vino.
        </prompt>
        <choice next=”acqua.vxml”> acqua </choice>
        <choice next=”vino.vxml”> vino </choice>
   </menu>
</vxml>
```

Il tag <record> rappresenta un caso particolare in quanto registra semplicemente in un file audio quanto ricevuto in ingresso dal microfono, senza effettuare nessun tipo di riconoscimento vocale.

✍ *Tag* per lo svolgimento di *operazioni*

Appartengono a questa categoria tutti i tag che permettono di svolgere operazioni di qualsiasi tipo che non richiedano nessun tipo di interazione con l'utente. Tra le varie operazioni che possono essere eseguite ci sono ad esempio il salto ad altri form (<*goto*>), assegnamento di variabili *(*<*assign*>) e valutazione di condizioni (<*if*>, <*elseif*>).

Esempio

```
<vxml version="1.0">
   <var name="variabile"/>
   <form id="main">
        <field name="scelta">
             <grammar> acqua | vino </grammar>
             <prompt>
                  Scegli tra acqua o vino.
             </prompt>
             <filled>
                  <if cond="scelta=='acqua'">
                       <assign name="variabile" expr="'acqua'"/>
                       <goto next="acqua.vxml"/>
                  <elseif cond="scelta=='vino'"/>
                       <assign name="variabile" expr="'vino'"/>
                       <goto next="vino.vxml"/>
                  </if>
             </filled>
        </field>
   </form>
</vxml>
```

2.6.3. Grammatiche

Una grammatica identifica differenti parole o frasi che un utente può pronunciare e, opzionalmente, specifica come interpretare valida un'espressione in termini di valori per variabili di input. Le grammatiche possono spaziare da una semplice lista di parole fino ad insieme complesso di frasi.
Ad esempio ciascun campo in un *form* deve avere una grammatica che specifica le risposte possibili che può dare un utente per quello specifico campo, oppure un intero *form* può avere una grammatica che specifica come riempire le variabili multiple di input da una singola frase dell'utente o ancora ciascuna scelta in un menu ha una grammatica che specifica gli input che l'utente può dare per effettuare una scelta.
Una grammatica contiene una o più *rules* che specificano l'*input matching*, solitamente con una regola specificata come la *root rule* della grammatica, che permette di usare nell'applicazione VoiceXML la grammatica di cui fa parte senza dichiarare esplicitamente da quale regola partire. In alcune grammatiche c'è esattamente una sola regola di livello superiore che può essere usata come radice della grammatica, come ad esempio nel caso di una grammatica di tipo booleano di tipo Si/No. Al contrario una grammatica più complessa può avere regole multiple che possono essre usate come punto di partenza. Ad esempio se si considera una grammatica per il riconoscimento degli animali marini, ci può essere una regola che individua tutti gli animali. Questa regola può a sua volta essere composta da regole che individuano degli insiemi più piccoli di animali. La grammatica permette di specificare una di queste sottoregole come punto di partenza, invece di usare ogni volta l'intera grammatica. L'uso delle grammatiche all'interno dei documenti *VoiceXML* è specificato dal tag <*grammar*>.
Le grammatiche utilizzabili sono di tre tipi:

- grammatiche *built-in*

  Sono grammatiche presenti nativamente nell'interprete *VoiceXML* e si possono utilizzare direttamente senza nessun tipo di implementazione aggiuntiva.

- Grammatiche dell'*applicazione*

  Sono grammatiche definite in fase di sviluppo dell'applicazione. Esse possono essere definite ex-novo per l'applicazione che si vuole realizzare oppure possono far parte di una libreria

generica di grammatiche che possono essere usate in altre applicazioni. Queste grammatiche possono essere a loro volta:

- inline

  La definizione di una grammatica *inline* appare direttamente nell'elemento <*grammar*> del documento

- esterne

  Se il tag <*grammar*> ha un valore per entrambi gli attributi *src* ed *expr*, allora si è in presenza di una grammatica esterna. La definizione della grammatica avviene in un file esterno che deve essere specificato tramite un URI (*src*) oppure tramite un JavaScript che valuta tale URI (*expr*). Tale processo prende il nome di *referencing* della grammatica. In questo caso la radice della grammatica può essere definita a priori oppure si può delegare all'interprete di stabilire da quale regola partire. Il tag non può contenere elementi figli al suo interno. L'attributo opzionale type definisce invece il tipo MIME della grammatica, ossia il formato della grammatica stessa. I tipi di grammatiche attualmente più diffusi sono i seguenti:

Tipo	**Grammatica**
application/srgs+xml	XML Speech Grammar
application/srgs	ABNF Speech Grammar
application/x-nuance-gsl	Nuance GSL
application/x-nuance-dynagram-binary	Nuance Grammar Object
application/x-jsgf	Java Speech Grammar Format

3. MOTIVAZIONI E OBIETTIVI

3.1. La situazione attuale della navigazione vocale

I limiti principali dell'attuale navigazione vocale delle pagine web sono molteplici. Tralasciando per un attimo i gli aspetti più propriamente legati alle tecnologie del riconoscimento vocale, il problema principale è dato dal tipo di approccio alla navigazione.

In un ambiente grafico l'utente ha fin dall'inizio una visione d'insieme della pagina che intende visitare ed ha contemporaneamente davanti a sé i contenuti, i link, le immagini che gli permettono di muoversi nella pagina e reperire le informazioni desiderate. La navigazione è affidata completamente all'utente che in ogni istante può scegliere, con un semplice click, in che punto della pagina spostarsi, quale pagina visitare, quale operazione svolgere, ecc. Ovviamente con un interfacciamento di tipo visuale l'utente non ha la necessità di conoscere a priori quanto contenute nelle pagine che sta visitando, in quanto un semplice colpo d'occhio gli permette di effettuare una sommaria analisi che gli permetta di orientarsi nel cammino che sta per intraprendere.

In un ambiente totalmente vocale tutto ciò non è possibile. L'utente, infatti, non conosce a priori né la struttura della pagina, né i suoi contenuti ed è quindi difficile, già di per sé, capire cosa potere cercare, mentre è del tutto impossibile sapere dove cercarlo.

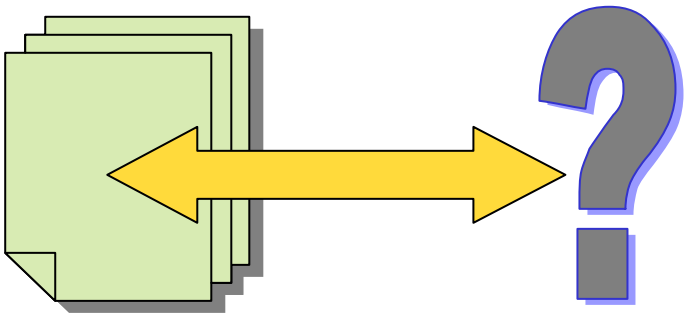

Figura 4 - Come sapere cosa contiene una pagina ad accesso vocale e dove sono i contenuti?

Se un utente accede per la prima volta ad un generico sito la situazione è proprio quella sopra descritta, in particolare per quanto riguarda la seconda condizione. Qualora l'utente conoscesse, per ipotesi, di cosa parla la pagina (per pregresse esperienze di navigazione o semplicemente perché gli è stato reso noto da qualcuno), sicuramente non potrebbe dedurre da questo dove cercare le informazioni desiderate o evincere la struttura del sito ed i possibili cammini navigazionali da seguire.
L'esperienza della navigazione non può essere condotta in prima persona dall'utente, come avviene nell'interfacciamento di tipo visuale, ma deve essere gestita da un apposito *navigatore*, che in linea generale, dovrebbe essere in grado di:

- presentare le informazioni contenute in una pagina

  E' la funzionalità di base di un qualsiasi tipo di navigatore e permette la fruibilità delle informazioni contenute nelle pagine che si vogliono visitare. Ovviamente nel caso dell'interfacciamento vocale questo aspetto assume un ruolo ancora più decisivo, in quanto l'unico mezzo attraverso il quale raggiungere i contenuti è il navigatore stesso, che dovrà occuparsi anche dell'effettiva presentazione degli stessi.

- permettere di muoversi da un punto all'altro della pagina

  La navigazione non è un semplice scorrimento sequenziale di tutta l'informazione contenuta in una pagina, ma presuppone di potersi muovere da un punto all'altro della pagina, spesso in maniera del tutto casuale, altre volte in modo molto mirato, ora leggendo, per esempio, tutto un articolo, ora solo i titoli dei brani presenti. Tutto questo è insito nel concetto stesso di *browsing* (letteralmente "sbirciare, dare un'occhiata") che presuppone un approccio non totale all'informazione, ma parcellizzato e mirato in base alle necessità contingenti.

- effettuare delle ricerche mirate

  Spesso l'obiettivo non è quello di navigare tra le informazioni, ma effettuare una ricerca mirata a reperire solo e soltanto quanto richiesto. Ciò in generale avviene sia nei motori di ricerca che nei normali browser di pagine web tramite ricerca per parole chiave. I risultati sono ottenuti in tempi molto brevi, ma spesso la ricerca non fornisce all'utente quanto richiesto, dando luogo a risultati parziali, non desiderati ed equivoci.

- raggiungere altre risorse tramite link

  Le pagine sono collegate tra loro tramite dei collegamenti ipertestuali, che sono stati la grande novità introdotta dall'Html. Un navigatore deve consentire di poter raggiungere tutti i link presenti e le risorse ad essi collegati. Se in un ambiente visuale tutto ciò risulta estremamente facile, immediato e affidato completamente al controllo dell'utente, nel caso vocale deve essere gestito ancora una volta dal navigatore, che si deve occupare sia della presentazione dei collegamenti, sia della loro attuazione mediante dei comandi vocali.

- fornire tutte le classiche funzionalità utili alla navigazione

  Le funzionalità di base che deve fornire un navigatore sono quelle della navigazione tradizionale, opportunamente ricondotte ed adattate al caso vocale. Nella tabella seguente si confrontano alcune funzionalità visive rilevanti rispetto a dei possibili equivalenti nell'ambito vocale.

Ambiente visuale	**Ambiente vocale**
Point and click	Comandi vocali
Tasto "Indietro"	Comando vocale, come "Go back" o "Indietro", che ritorna al dialogo precedente
Tasto "Pagina iniziale"	Comando vocale, come "Go home" o "Menu", che porta l'utente al dialogo iniziale
Layout dello schermo, colori, grafica e stile	Audio pre-registrato, voci sintetizzate, sesso della voce e avvisi sonori
Finestre di errore o avviso	Avvisi sonori, voci sintetizzate o audio pre-registrato che pronunciano messaggi di errore o avviso
Sezioni di aiuto statiche o aiuto contestuale	Messaggi vocali di aiuto inseriti nei dialoghi
Moduli di input, liste di selezione e bottoni *radio*	Moduli VXML con campi e variabili per l'acquisizione di dati
Indicatori di avanzamento e attesa	Suoni di attesa, musiche, o messaggi vocali di avanzamento e attesa

I navigatori vocali attualmente disponibili e diffusi si basano essenzialmente su due tipi classici di navigazione, ovvero:

- navigazione *sequenziale* (tipo *screen reader*)

  E' senza dubbio la più semplice sia dal punto di vista concettuale che da quello tecnico-implementativo. Una navigazione di questo tipo consiste di fatto nel "leggere" all'utente i contenuti delle pagine nell'ordine in cui essi si presentano, dall'alto verso il basso. Ovviamente un approccio di tal genere, che può essere pur indicato ed efficiente in taluni casi (come la semplice lettura di un file di testo, quale ad esempio un e-book), risulta particolarmente inefficace se applicata al caso specifico della navigazione. Infatti la navigazione è tutt'altro che una semplice lettura sequenziale di un documento. In tal modo si perde ogni possibilità di interazione e di conseguenza viene meno proprio il concetto stesso di navigazione. Con un approccio di questo tipo l'utente è costretto ad ascoltare la lettura dell'intera pagina per poter sapere di cosa parla, se contiene l'informazione desiderata o semplicemente per scoprire che la pagina in questione non è di nessun interesse.

- navigazione *gerarchica* (basata sulla struttura del documento)

  La navigazione non avviene per semplice lettura della pagina da cima a fondo come nel caso precedente, ma si basa sulla struttura gerarchica della pagina . Tutti i documenti ipertestuali hanno infatti la possibilità di essere strutturati a vari livelli mediante tag specifici dell'html (<h1>,<h2>,…,<h5>,<p>). Una navigazione di questo tipo permette di muoversi attraverso l'albero della struttura del documento, aumentando o diminuendo di volta in volta il livello di profondità, ovvero scendendo nei sottoparagrafi o risalendo ai paragrafi di livello più alto. Ovviamente la navigazione risulta più efficiente rispetto al caso sequenziale, poiché consente di muoversi tra le sezioni della pagina al livello di struttura desiderato. Tuttavia se la struttura è piuttosto complessa e a molti livelli, il rischio è quello di perdersi durante la navigazione, oppure scoprire che, dopo una lunga esplorazione fino ai livelli minimi della struttura, l'informazione cercata non è presente.

Una terza tipologia di navigazione, ancora in fase di sviluppo a livello prototipale, è quella *semantica*, basata cioè sul significato dei contenuti e dei loro collegamenti. I navigatori basati su questo nuovo approccio sono tipicamente per un tipo di navigazione visuale.
Gli esempi più noti in letteratura riguardano in particolare:

- navigatori di ontologie alle quali sono collegate delle risorse mappate in modo statico[3]

  La navigazione ha inizio con la scelta di una determinata ontologia di dominio. A questo punto si ha la possibilità di scegliere una particolare relazione, fra quelle disponibili, attraverso la quale visualizzare l'albero dell'ontologia. Navigando all'interno dell'albero ci vengono presentati quei documenti della base di conoscenza associati a precisi concetti selezionati.

- navigatori degli indici semantici di siti[4]

  La navigazione semantica non avviene su tutto il sito, ma solamente a livello di indice. Ad esso è infatti collegata un'ontologia che permette di navigarlo semanticamente e, come nel caso precedente, sulle varie voci dell'indice sono mappate in modo diretto e statico tutte le risorse raggiungibili.

[3] Si veda ad esempio il prototipo sviluppato dall'Università degli Studi di Milano, di Lecce e di Cagliari, raggiungibile all'indirizzo http://ra.crema.unimi.it/navigator
[4] Si veda ad esempio il progetto dell'Università di Helsinki relativo ai musei finlandesi raggiungibile all'indirizzo http://www.cs.helsinki.fi/group/seco/museums (disponibile solo in finlandese, con un breve tutorial in inglese)

3.2. Gli obiettivi della navigazione vocale semantica

L'obiettivo è quindi quello di trovare una modalità di navigazione più pratica ed efficiente che:

- si avvicini al modo di operare quotidiano dell'utente
- sia il più possibile user-friendly
- risolva i problemi delle attuali modalità di navigazione

Il navigatore da sviluppare deve quindi soddisfare le seguenti esigenze che sono emerse in una prima analisi generale del problema:

- la ricerca delle informazioni deve essere demandata completamente all'applicazione

  Chi si occupa del reale reperimento delle informazioni non è l'utente, ma l'applicazione che gestisce la navigazione. Essa riceve in ingresso una richiesta da parte dell'utente e sulla base di questa lo conduce lungo un percorso navigazionale ottimale, proponendogli infine i risultati individuati.

- i contenuti di interesse dell'utente devono essere facilmente raggiungibili

  Tutti i contenuti devono essere raggiungibili sulla base del loro significato e la loro individuazione deve risultare il più possibile semplice, efficiente ed immediata. Come già detto il reperimento delle informazioni è affidato al navigatore, riducendo così lo sforzo operativo continua dell'utente, che altrimenti sarebbe costretto ad ascoltare tutta l'informazione contenuta nelle pagine ed estrarre personalmente mediante un ragionamento logico deduttivo i contenuti desiderati.
- il reperimento delle informazioni desiderate deve essere preciso e deve ridurre la quantità di informazioni superflue, indesiderate e non pertinenti

  Il fatto stesso che non viene letto tutto il contenuto della pagina, ma solamente quelle parti che contengono l'informazione richiesta, riduce drasticamente la quantità di testo letto all'utente.

Inoltre, poiché le ricerche avvengono sulla base della semantica delle parole, i risultati devono essere esattamente quelli desiderati, eliminando così la gran parte dei possibili risultati equivoci, indesiderati e non pertinenti a quanto richiesto.

- l'interazione tra utente e applicazione deve avvenire il più possibile tramite linguaggio naturale

Uno dei grossi problemi che si riscontano nell'interazione vocale tra uomo e macchina è che l'utente spesso non può esprimersi naturalmente così come farebbe dialogando con un'altra persona. Questo è principalmente dovuto al fatto che la struttura delle VUI delle applicazioni vocali è tipicamente strutturata secondo dei menu rigidi, che hanno ben poco a che fare con il linguaggio naturale. L'utente è quindi impossibilitato nell'esprimersi in modo colloquiale, né tanto meno può esprimere richieste formulate senza schemi linguistici rigidi che possano essere compresi dall'applicazione che gestisce il flusso del dialogo. L'obiettivo è quindi quello di mettere l'utente il più possibile a suo agio, consentendogli, ove possibile, di instaurare un dialogo del tutto naturale con l'applicazione, riducendo così anche il grosso distacco emozionale che necessariamente si crea tra uomo e macchina.

Ricollegandosi ad uno dei problemi affrontati nel paragrafo precedente, con una navigazione di tipo classico l'utente non può effettuare le tipiche operazioni di *browsing* così come letteralmente inteso, ossia non può effettuare un' analisi se pur sommaria dei contenuti di una pagina per comprendere in pochi istanti quali sono i suoi contenuti principali. Questa operazione di interpretazione e di analisi, del tutto naturale per un utente ad accesso visuale, dovrebbe quindi essere svolta dal navigatore, in modo da fornire all'utente una descrizione generale della pagina, presentandogli i concetti principali in essa contenuti. In tal modo, anche ad un primo accesso ad una qualsiasi pagina, un utente è in grado di avere una visione panoramica almeno sui suoi contenuti.

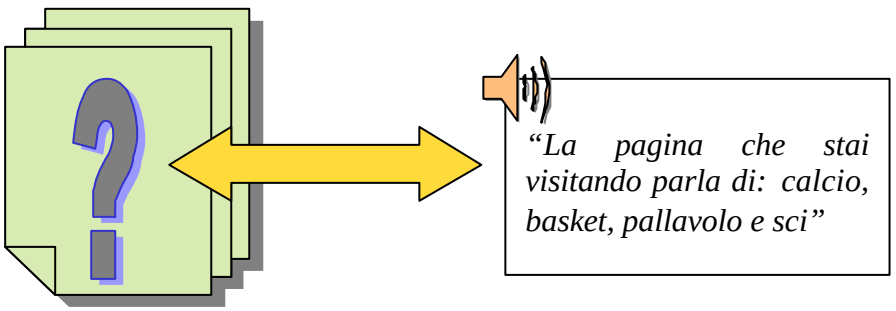

Figura 5 - Analisi dei contenuti principali di una pagina da parte del navigatore

Una considerazione finale, ma di fondamentale importanza, è il fatto che il navigatore è concepito per essere utilizzato su qualsiasi sito progettato in modo tradizionale ed anche già realizzato. L'approccio adottato non è rivolto alla progettazione di siti semantici e semanticamente navigabili, ma diametralmente opposto. Dato un qualsiasi sito e una ontologia (non necessariamente mappata in modo diretto sul sito) che descriva l'informazione in esso contenuta, il navigatore deve essere in grado di attribuire una semantica alle informazioni e, sulla base di questa, condurre l'utente durante la navigazione.

4. SPECIFICHE E PROGETTAZIONE

4.1. Analisi dei requisiti funzionali

Per poter usufruire di funzionalità semantiche nella navigazione, verrà utilizzata un'*ontologia* (ovvero una collezione strutturata e gerarchica di concetti e di relazioni che intercorrono tra di essi) che raccoglierà tutta la conoscenza necessaria per la descrizione dei contenuti su cui si vuole effettuare la navigazione. Dato un qualsiasi documento sarà possibile quindi stabilire le corrispondenze con i concetti presenti nell'ontologia. In questo modo sarà possibile attribuire al documento una propria semantica.

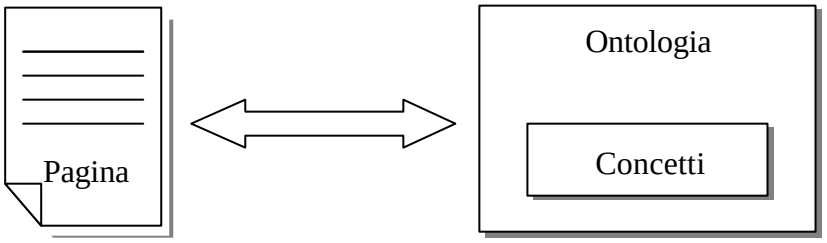

Figura 6 - Corrispondenza tra pagine e concetti dell'ontologia

Tuttavia se da una parte risulta semplice stabilire le corrispondenze di un documento con i concetti dell'ontologia, dall'altra è difficile stabilirle con le relazioni tra concetti. Spesso, infatti, una relazione in sé non è individuabile all'interno di un documento generico ed inoltre non è sufficiente individuare i concetti coinvolti nella relazione per ipotizzare la presenza della relazione stessa.
Perciò la navigazione dovrà basarsi esclusivamente sui concetti senza considerare le relazioni (almeno così come si presentano all'interno dell'ontologia).

Per quanto riguarda l'idea iniziale di possibilità di espressione dell'utente in linguaggio naturale, bisogna considerare che se da un lato è possibile che le richieste vengano formulate liberamente, dall'altro risulta piuttosto complessa la loro analisi semantica necessaria per comprendere il reale significato della richiesta. Per questo motivo l'idea è quella di analizzare e semplificare la richiesta espressa dall'utente individuando in essa i concetti presenti nell'ontologia e le loro eventuali

relazioni per poi poter effettuare tutte le operazioni necessarie sull'ontologia in modo da poter produrre i risultati richiesti.

Inoltre il navigatore ipotizzato inizialmente funzionerebbe solamente nel caso in cui l'utente conoscesse a priori ciò di cui parlano le pagine da visitare. Si ritiene quindi necessario fornire all'applicazione due modalità differenti di navigazione semantica:

- tramite una richiesta dell'utente vengono individuati i concetti dell'ontologia presenti nella richiesta i quali vengono poi cercati nei contenuti della pagina; i risultati vengono poi presentati all'utente in modo sequenziale.

- l'utente non effettua nessuna richiesta, ma l'intera pagina viene analizzata, individuando tutti i concetti dell'ontologia presenti in essa; l'utente può quindi navigare i contenuti della pagina scegliendo tra i concetti individuati (che saranno presentati secondo la gerarchia delle classi OWL).

Tale navigatore si presta ad essere applicato a pagine che:

- si riferiscono ad un ambito specifico e ristretto in cui i concetti presenti siano ragionevolmente limitati in numero (mantenendo comunque un certo grado di strutturazione)

- presentino una sufficiente quantità di contenuti effettivamente navigabili (ossia brani di testo di una certa ampiezza che siano in relazione con i concetti dell'ontologia). Tutte le parti "piccole" di testo o non in relazione con l'ontologia dovrebbero perciò essere ignorate (ad esempio i titoli non costituiscono, in quest'ottica, dei contenuti rilevanti ai fini della navigazione).

Funzionalità richieste all'applicazione

L'applicazione mette a disposizione dell'utente un'interfaccia completamente vocale attraverso la quale egli può accedere alle funzionalità di base della navigazione web. L'utente può scegliere la pagina da visitare e navigare attraverso i collegamenti ipertestuali (link) presenti. L'interfaccia vocale permette di navigare i contenuti di una pagina in modo semantico, cioè in base al loro significato e non in modo sequenziale o gerarchico come avviene tradizionalmente.

4.2 Architettura software

4.2.1. Architettura client-server

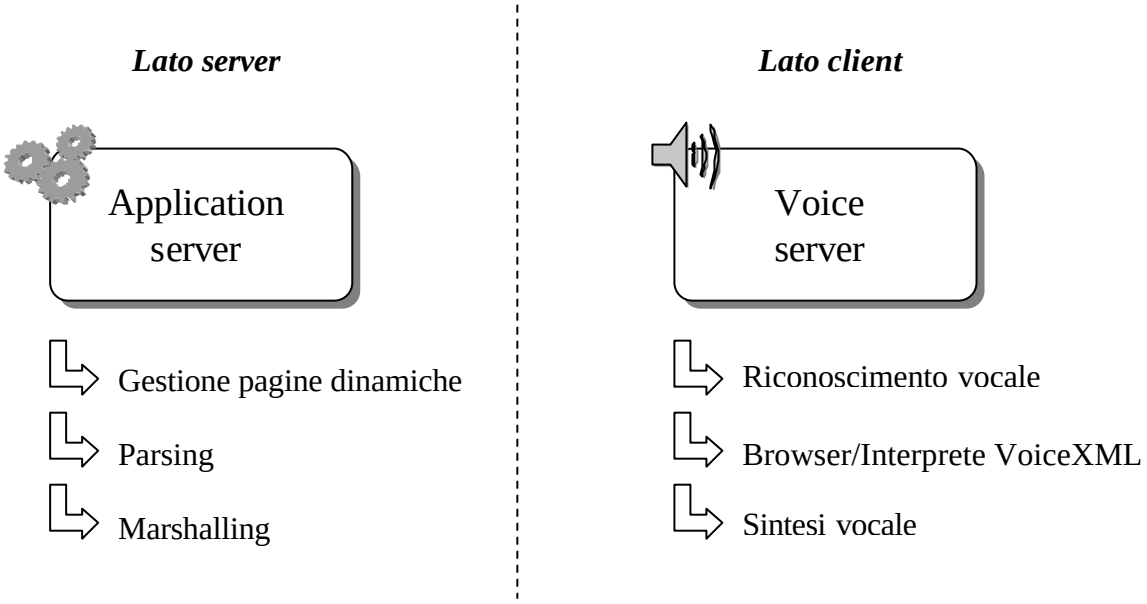

Figura 7 - Funzionalità dell'architettura client-server

4.2.1.1. Lato server

L'applicazione risiede completamente su un *application server* che si occupa dell'esecuzione delle varie pagine dinamiche (*Java Server Pages e Servlet*).
Su di esso risiedono inoltre un *parser* e un *marshaller* utilizzati per tutte le conversioni delle pagine web in *Java* (*parsing*) e viceversa (*marshalling*).

4.2.1.2. Lato client

Nonostante l'intera esecuzione avvenga a lato *server*, è necessario che a lato *client* sia presente un voice server che permetta di usufruire dell'interfaccia puramente vocale. Infatti il *server* invia al *client* delle pagine *VoiceXML* che necessitano di essere interpretate e quindi "lette" dal *voice server* all'utente che ne ha fatto richiesta. In particolare il voice server si deve occupare di:

1. riconoscimento vocale
2. browsing/interpretazione delle pagine *VoiceXML*
3. sintesi vocale

4.2.2. Funzionamento dell'applicazione

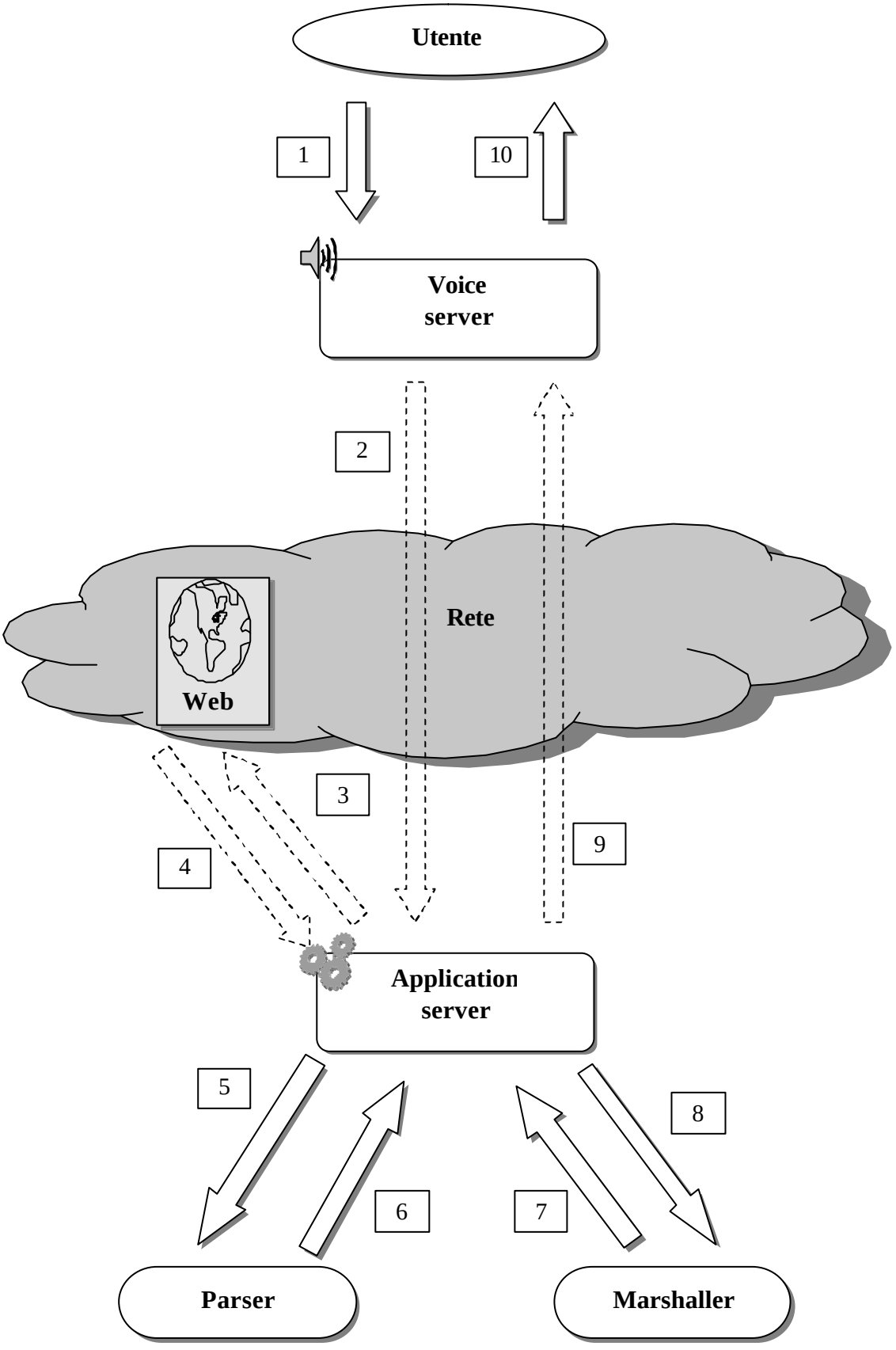

Figura 8 - Architettura dell'applicazione

L’utente dialoga con un’interfaccia vocale e richiede una pagina web (1). Tramite il voice server, la richiesta viene inoltrata attraverso la rete fino all’*application server* (2) che si occupa del reperimento della risorsa web necessaria (3-4). A questo punto la pagina *Html* viene inviata al parser (5) e convertita in un oggetto Java (6) sul quale l’applicazione svolge tutte le operazioni necessarie. Terminate tali operazioni il server invia un altro oggetto java al *marshaller* (7), il quale lo converte in una pagina *VoiceXML* (8). L’applicazione ritorna la pagina prodotta al Client (9). Infine il voice server la interpreta e la presenta all’utente (10).

4.2.3. Descrizione delle classi

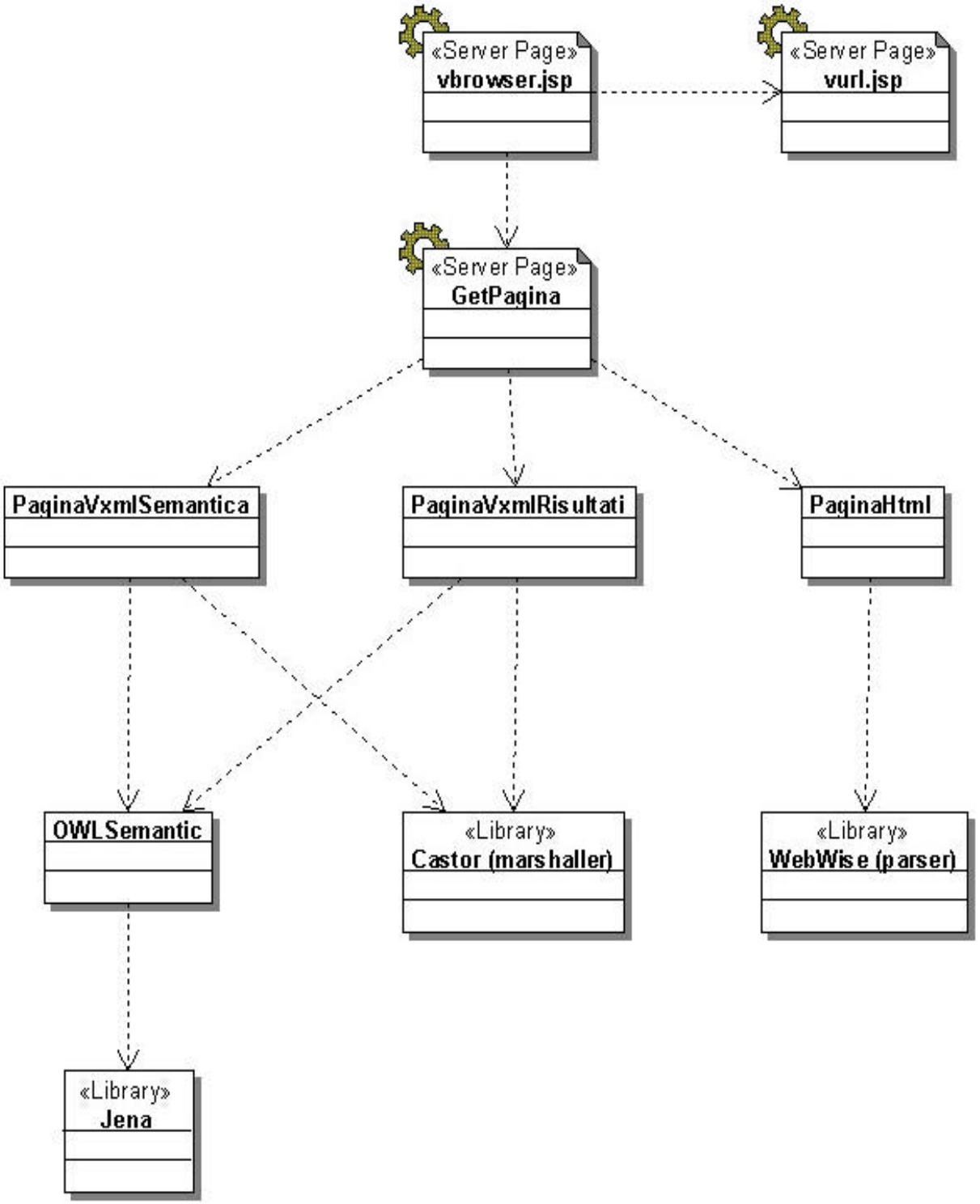

Figura 9 - Class diagram

L’applicazione è costituita da una pagina principale (*vbrowser.jsp*), che ha la funzione di interfacciamento con l’utente. *Vbrowser.jsp* è inoltre la pagina che viene mandata in esecuzione all’avvio.

Una seconda pagina jsp (*vurl.jsp*) è utilizzata per gestire l’inserimento degli url.

La servlet *getPagina* si occupa della richiesta della pagina Html che l’utente ha scelto di visitare e della selezione della classe appropriata in base al tipo di navigazione semantica selezionata (*paginaVxmlSemantica* e *paginaVxmlRisultati*).

La classe *paginaHtml* (istanziata dalla servlet) crea, attraverso il *parser*, un oggetto Java che costituisce la rappresentazione gerarchica della pagina Html.

Le due classi *paginaVxmlSemantica* e *paginaVxmlRisultati* elaborano semanticamente le informazioni estratte dall’oggetto creato dal *parser* e costruiscono un ulteriore oggetto java che, dato in ingresso al *marshaller*, viene trasformato nella pagina VoiceXML finale che viene letta all’utente.

La classe *OWLSemantic*, utilizzando le funzioni offerte dalla libreria Jena, costituisce l’interfaccia tra l’applicazione Java e l’ontologia Owl.

4.2.4. Casi d'uso dell'applicazione

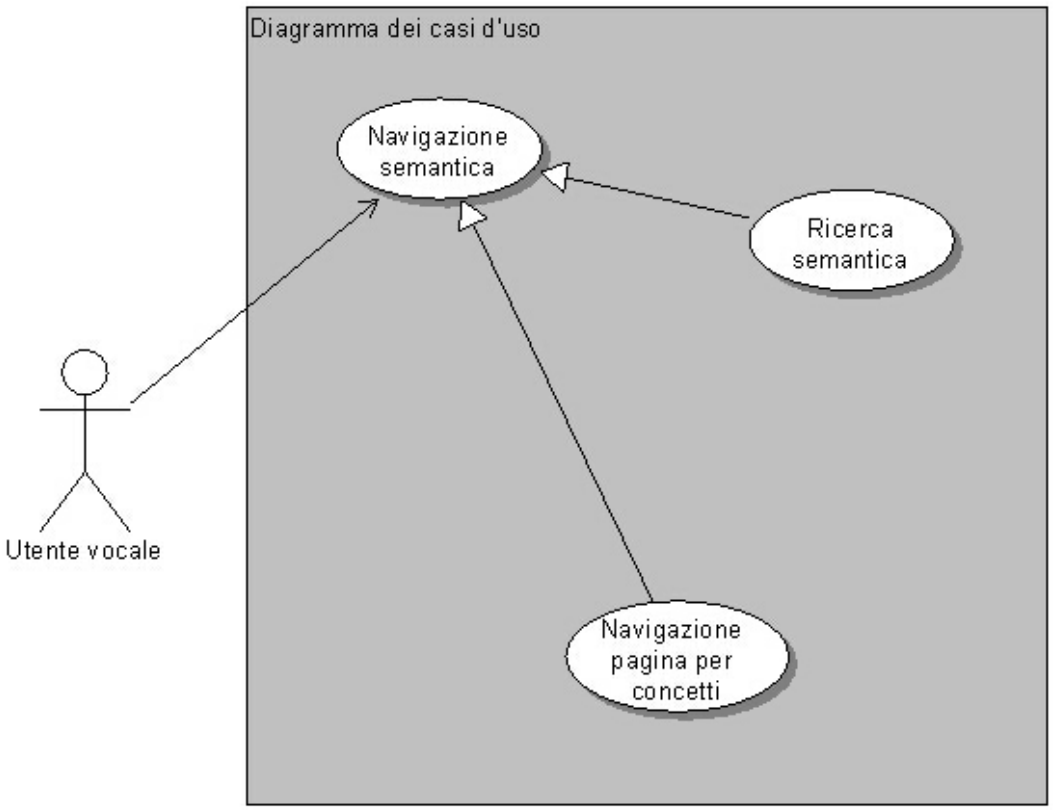

Figura 10 - Use Case diagram

Il caso d'uso principale dell'applicazione consiste nella navigazione semantica di una pagina html. Come già visto nella fase di analisi tale navigazione si presenta in due differenti modalità, pertanto il caso d'uso principale è una generalizzazione dei due casi d'uso specifici:

- *Ricerca semantica*

  L'utente esprime tramite una richiesta vocale i termini della ricerca che vuole effettuare all'interno della pagina che sta visitando. Successivamente ascolta i contenuti della pagina corrispondenti alla richiesta effettuata.

- ✍ *Navigazione della pagina per concetti presenti*

  All’apertura della pagina l’utente naviga l’elenco dei concetti presenti. Può inoltre scegliere un concetto e ascoltare i contenuti ad esso inerenti.

Di seguito si riportano due *sequence diagram* che descrivono in modo più dettagliato i due casi d’uso specifici sopra citati.

4.2.4.1. Ricerca semantica

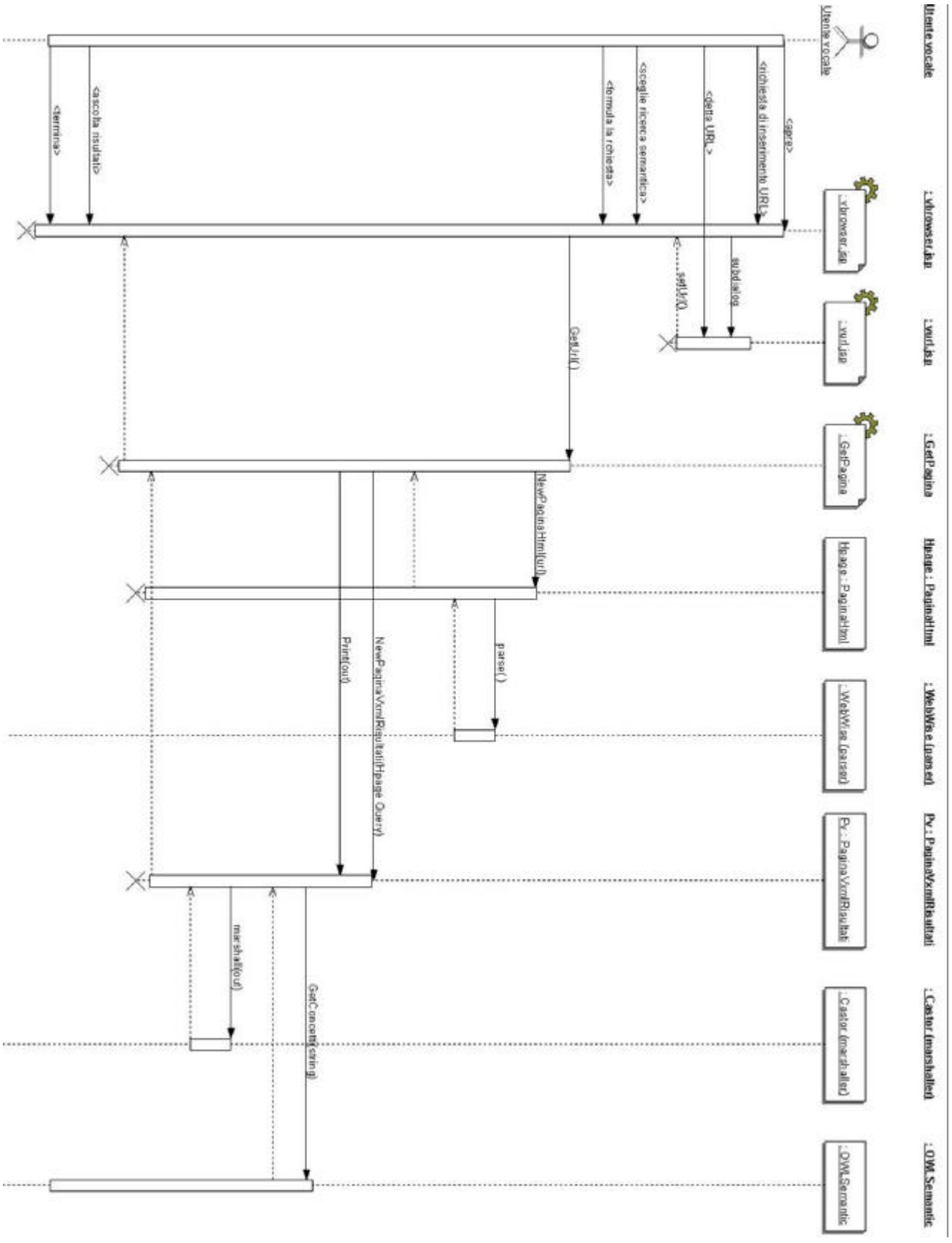

Figura 11 - Sequence diagram della ricerca semantica

Si riporta ora una breve e sintetica spiegazione del diagramma appena presentato ponendo l'attenzione soprattutto sull'utente e sulle classi o oggetti coinvolti nelle interazioni, senza entrare nel dettaglio delle singole operazioni che caratterizzano ogni messaggio di sequenza:

All'avvio dell'applicazione l'utente apre la pagina *vbrowser.jsp* e richiede l'inserimento dell'indirizzo della pagina che intende visitare.
Attraverso un *subdialog* (che avvia *vurl.jsp*) detta l'*url* della pagina prescelta.
A questo punto, dopo aver settato l'indirizzo, sceglie di effettuare una ricerca semantica e formula la sua richiesta.
Vbrowser.jsp manda in esecuzione la servlet *getPagina* passandole i termini della ricerca.
Viene creato così l'oggetto *hpage* che parsa la pagina Html e la trasforma in Java.
Successivamente viene istanziata anche la classe *PaginaVxmlRisultati* che dialogando con *OWLSemantic* costruisce l'oggetto finale che viene mandato in ingresso al *marshaller.*
A questo punto viene ritornata a *vbrowser.jsp* la pagina VoiceXML finale.
L'utente ascolta i risultati prodotti e termina infine l'applicazione.

4.2.4.2. Navigazione della pagina per concetti

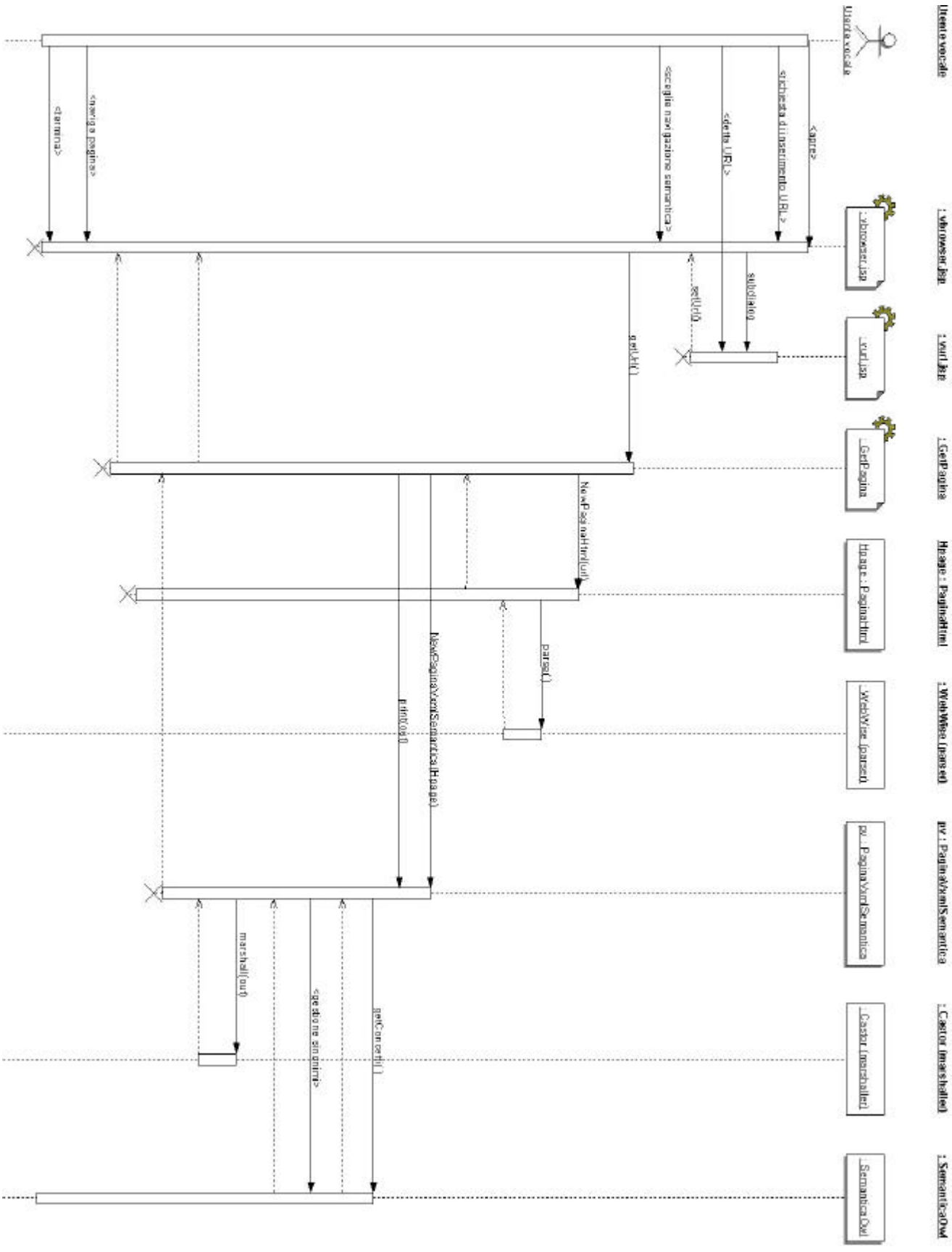

Figura 12 - Sequence diagram della navigazione per concetti

Si riporta ora una breve e sintetica spiegazione del diagramma appena presentato ponendo l'attenzione soprattutto sull'utente e sulle classi o oggetti coinvolti nelle interazioni, senza entrare nel dettaglio delle singole operazioni che caratterizzano ogni messaggio di sequenza:

All'avvio dell'applicazione l'utente apre la pagina *vbrowser.jsp* e richiede l'inserimento dell'indirizzo della pagina che intende visitare
Attraverso un *subdialog* (che avvia *vurl.jsp*) detta l'*url* della pagina prescelta.
A questo punto, dopo aver settato l'indirizzo, sceglie di effettuare una navigazione della pagina per concetti presenti in essa.
Vbrowser.jsp manda in esecuzione la servlet *getPagina* passandole l'*url* inserito.
Viene creato così l'oggetto *hpage* che parsa la pagina Html e la trasforma in Java.
Successivamente viene istanziata anche la classe *PaginaVxmlSemantica* che dialogando con *OWLSemantic* costruisce l'oggetto finale che viene mandato in ingresso al *marshaller.*
A questo punto viene ritornata a *vbrowser.jsp* la pagina VoiceXML finale.
L'utente naviga vocalmente la pagina prodotta e termina infine l'applicazione.

4.3. Metodologia di progettazione e realizzazione dell'ontologia

Il processo di sviluppo di un'ontologia richiede fondamentalmente di:

- definire le classi dell'ontologia
- organizzare le classi in una gerarchia tassonomica di sottoclassi-superclassi
- definire gli slot delle proprietà e descrivere i valori che possono assumere
- immettere i valori degli slot
- creare delle istanze

Una premessa fondamentale è che, sebbene la progettazione di una ontologia possa sembrare simile ad un tipo di progettazione orientata agli oggetti, sviluppare una ontologia è differente dal creare classi e relazioni nella programmazione object-oriented. Infatti mentre nell'ottica dell'object-orientation si concentra l'attenzione attorno ai metodi di una classe (un programmatore effettua un design delle classi basandosi sulle proprietà operative di una classi), per progettare un'ontologia è necessario prendere decisioni sulla base delle proprietà strutturali di una classe.

In generale non esiste un metodo univoco e soprattutto universalmente "corretto" per sviluppare un'ontologia. La metodologia qui illustrata[5] (ed in seguito utilizzata per il design dell'ontologia usata nel progetto) segue un approccio di tipo iterativo: si parte con un primo passo piuttosto grossolano di definizione dell'ontologia per poi raffinare sempre più l'evoluzione della stessa scendendo via via ad un livello di dettaglio maggiore.
In primo luogo si possono individuare tre regole fondamentali che spesso possono aiutare, ma che non vogliono essere affatto dogmatiche:

1. *Non c'è un modo univocamente corretto di modellare un dominio, ma ci sono sempre varie alternative possibili. La migliore soluzione dipende quasi sempre dal tipo di applicazione che si vuole realizzare e dalle possibili estensioni che potrebbe avere*
2. *Lo sviluppo di un'ontologia è necessariamente un processo iterativo*

[5] La metodologia adottata è quella proposta dall'Università di Stanford *(Ontology Development 101)*

3. *I concetti dell'ontologia devono essere il più possibile vicini agli oggetti (fisici o logici) e alle relazioni del dominio di interesse. Nella descrizione del dominio compariranno quindi di preferenza nomi (oggetti) o verbi (relazioni).*

In generale il decidere per che cosa andrà utilizzata l'ontologia e quanto dettagliata o generica sarà è ciò che deve guidare le decisioni della modellazione durante tutto il suo iter. Tra tutte le alternative possibili, occorre determinare quella che meglio funziona per il compito a cui deve assolvere, che risulta più intuitiva, più estendibile e mantenibile. Bisogna inoltre ricordare che un'ontologia è un modello della realtà del mondo e i concetti presenti in essa devono quindi riflettere in un certo senso questa realtà. Dopo aver definito una versione prototipale dell'ontologia, bisognerebbe valutarla e correggerne gli eventuali sia usandola in applicazioni reali che discutendone con esperti del dominio considerato. Questo processo di design iterativo deve continuare per l'intero ciclo di vita dell'ontologia.

Il suddetto processo si può scomporre operativamente in 7 passi principali e consequenziali:

1. determinare il dominio e lo scopo dell'ontologia
2. considerare il riuso di ontologie preesistenti
3. enumerare i termini più importanti
4. definire le classi e la loro gerarchia
5. definire le proprietà delle classi (slot)
6. definire le restrizioni sugli slot
7. creare le istanze

Di seguito verranno esplicitati i sopraccitati 7 step facendo esplicito riferimento al caso specifico del progetto realizzato.

4.3.1. Determinazione del dominio e dello scopo dell'ontologia

Definire il dominio e lo scopo di un'ontologia significa in poche parole rispondere ad alcune domande di base come ad esempio:

- *Quale ambito di conoscenza deve coprire l'ontologia?*
- *Per che cosa andrà utilizzata l'ontologia?*
- *A quale tipo di richieste dovranno dare una risposta le informazioni contenute nell'ontologia?*
- *Chi dovrà usare l'ontologia?*
- *Chi dovrà mantenerla?*

Una base di conoscenza basata sull'ontologia deve essere in grado di rispondere a questa breve lista di domande, dette *domande di competenza*. Queste domande saranno molto utili in seguito per verificare se l'ontologia realizzata contiene abbastanza informazione.
Queste domande di competenza sono solamente, come già detto, un profilo generale e non necessitano di essere del tutto esaustive.
Le risposte possono cambiare durante il processo di design, ma in ogni istante possono aiutare a limitare lo scopo del modello.

Rispondiamo alle domande precedenti per quanto riguarda il caso dell'ontologia realmente sviluppata durante il progetto.
L'ontologia deve coprire in generale il campo della *musica* e ed in particolare andrà utilizzata per gestire un archivio di articoli relativi al Conservatorio di Musica G. Verdi di Como e a tutte le iniziative musicali ad esso connesse. Le informazioni presenti nell'ontologia dovranno permettere agli utenti di tale archivio di reperire gli articoli i cui contenuti rispondano alle richieste specificate di volta in volta. Ad esempio delle richieste tipo a cui dovrà essere in grado di fornire delle risposte possono essere: richiesta di articoli in cui si parla di un certo compositore, piuttosto che di un particolare concerto o di tutti i concerti per un determinato strumento, ecc.
L'ontologia verrà usata non direttamente dagli utenti ma verrà gestita tramite un'applicazione che gestirà tutte le operazioni di navigazione attraverso le varie pagine che costituiscono l'archivio. Ipoteticamente (ma realisticamente) la manutenzione potrebbe essere affidata ad un esperto di musica (o comunque non dovrebbe mancare il supporto di tale figura per poter costruire un'ontologia corretta).

4.3.2. Considerazione del riuso di ontologie preesistenti

E' pressoché sempre apprezzabile considerare ciò che qualcuno ha già fatto e verificare se si possono rifinire ed estendere risorse esistenti per il proprio particolare dominio e compito. Riusare delle ontologie potrebbe essere necessario se il proprio sistema necessita di interagire con altre applicazioni che utilizzano già particolari ontologie o vocabolari controllati. Molte ontologie sono disponibili in formato elettronico e possono essere facilmente importate in un ambiente integrato di sviluppo di ontologie. Il formalismo con il quale sono espresse le ontologie spesso non costituisce un grosso problema, grazie al fatto che molti sistemi di rappresentazione della conoscenza possono importare ed esportare ontologie. Inoltre la traduzione da un tipo di formalismo ad un altro non è solitamente particolarmente complesso.

Nel caso considerato, non si sono individuate in letteratura delle ontologie già sviluppate che si confacessero a quanto ritenuto necessario.
Inoltre, poiché il progetto da realizzare vuole essere soltanto un prototipo, si è ritenuto più utile e adeguato realizzare un'ontologia *ad hoc* che mappasse esattamente tutta (per quanto possibile) l'informazione contenuta negli articoli reali presi come esempio.
A questo proposito si segnala che l'ontologia è stata costruita sulla base degli articoli tratti dall'archivio del Conservatorio di Musica G. Verdi di Como relativi alla rassegna stampa 2003/2004 e reperibili all'indirizzo:
http://www.conservatoriocomo.it/archiv/archiv-review03-04.html
In particolare sono stati analizzati in dettaglio i primi 52 articoli presenti (fatta esclusione per quelli in formato non testuale, ma di immagini).

4.3.3. Enumerazione dei termini più importanti

E' utile scrivere una lista di tutti i termini di cui si vuole parlare o che si vogliono spiegare ad un ipotetico utente dell'ontologia. In particolare è utile chiedersi:

Quali sono i termini di cui si vuole parlare?
Quali proprietà devono avere questi termini?
Cosa si vuole dire di questi termini?

Inizialmente è importante stendere una lista comprensiva dei termini senza preoccuparsi delle coincidenze con i concetti che rappresentano, con le relazioni tra i termini o di qualsiasi proprietà che i concetti devono avere.

Si riporta a titolo di esempio una parziale enumerazione di tipo *brain storming* di alcuni dei termini individuati al termine della fase di analisi:

musicisti, compositori, nomi propri, periodi storici, appartenenza ad una corrente musicale, correnti musicali, strumento musicale, concerto, orchestra, secoli, tipi di musica, esecutori, ente, associazione, rassegna, coro, concorso, collocazione storica, conservatorio, violino, formazione, percussioni, docente, secolo, Romanticismo, musica elettronica, pianoforte, allievo, musica sacra, concorso, seminario, *ecc.*

I successivi due passi (la definizione delle classi e delle gerarchie e la definizione delle proprietà dei concetti) sono strettamente interconnessi e risulta difficile dire se compiere prima l'uno piuttosto che l'altro. Tipicamente si creano poche definizioni di concetti nella gerarchia e successivamente si continua con la descrizione delle proprietà di questi e via di seguito aggiungendo di volta in volta nuovi concetti.

4.3.4. Definizione delle classi e della loro gerarchia

Ci sono vari approcci nello sviluppo di una gerarchia di classi:

- *Top-down*

  Il processo di sviluppo parte con la definizione dei concetti più generali del dominio ed in seguito si esegue una specializzazione di questi.

- *Bottom-up*

Il processo inizia con l'individuazione delle classi più specifiche, le foglie della gerarchia, con il conseguente raggruppamento di queste classi in concetti più generali (superclassi).

- *Una combinazione delle due precedenti*

  Il processo di sviluppo è una combinazione degli approcci top-down e bottom-up. Si definiscono dapprima i concetti più salienti e poi li si generalizzano e specializzano in modo appropriato a seconda dei casi. Nel caso proposto in figura ad esempio, si parte dall'individuazione dei vini principali (Chianti, Pinot nero, ecc.) per poi da un lato scendere di profondità specializzando le tipologie individuate (riportando le sottotipologie di ciascun vino) e dall'altro salire di livello generalizzando in classi più astratte (i vini bianchi, rossi e rosati).

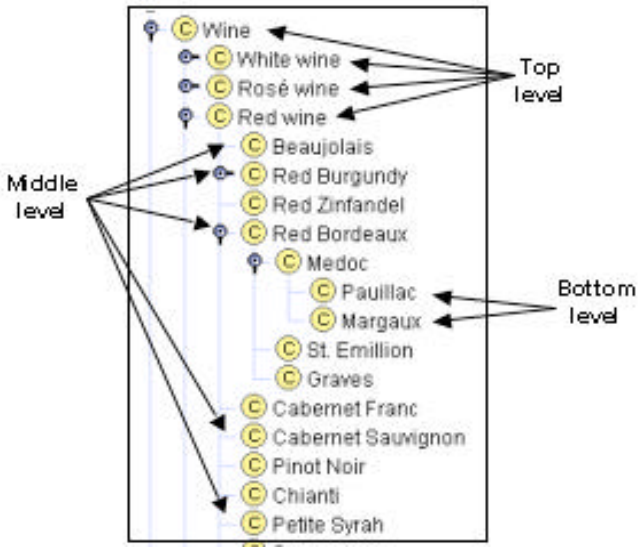

Figura 13 - Esemplificazione dei livelli di gerarchia

Nessuno di questi tre metodi è a priori il migliore. L'approccio da scegliere dipende in modo molto marcato dal personale punto di vista del dominio. Se uno sviluppatore ha una visione sistematica e "dall'alto" del sistema, sarà più semplice ed efficiente una strategia di tipo top-down. L'approccio combinato risulta spesso il più semplice, poiché i concetti di livello intermedio tendono ad essere quelli più descrittivi del dominio.

In generale, qualsiasi sia il metodo scelto si parte generalmente dalla definizione delle classi. Dalla lista creata al passo 3 si scelgono i termini che descrivono oggetti aventi un'esistenza indipendente piuttosto che termini che descrivono questi oggetti. Questi termini diverranno classi dell'ontologia. Si organizzano poi le classi in una tassonomia gerarchica chiedendosi se, essendo un'istanza di una classe, un oggetto sarà necessariamente un'istanza di certe altre classi. Se una classe A è superclasse di B, allora ogni istanza di B è anche istanza di A. In altre parole la classe B rappresenta un concetto che è un "*kind of*" A.

Una volta stesa la lista di tutti i termini considerati rilevanti si è proceduto all'individuazione delle classi principali, che nello specifico sono:

Genere_musicale, Corrente_musicale, Periodo_storico, Compositore, Musicista, Strumento-formazione, Evento ed *Ente*.

Le precedenti classi sono in grado di contenere tutta l'informazione relativa al dominio considerato. In particolare sono state così scelte in quanto non necessitano (almeno in questa fase prototipale) di nessun tipo di strutturazione gerarchica, in quanto tutte le classi possono appartenere ad un unico livello. Questo approccio deriva dalla scelta di avere come concetti dell'ontologia solamente delle istanze individuali e non delle classi astratte, che vengono invece usate solamente per una classificazione di primo livello. Tutto ciò è dovuto al fatto che nell'OWL Lite e DL esiste una netta separazione tra classi e istanze, che vincola le scelte strutturali. Questo problema è in realtà risolto dall'OWL Full, che permette ad un concetto di essre contemporaneamente classe ed istanza.Tuttavia l'OWL Full non può ancora essere impiegato per lo sviluppo di un'ontologia da utilizzare all'interno di una reale applicazione, in quanto non è attualmente supportato da nessun software (soprattutto perché non assicura tempi finiti di computazione).

4.3.5. Definizione delle proprietà delle classi (slot)

Le classi da sole non riescono solitamente a fornire abbastanza informazioni per rispondere alle domande di competenza enunciate al passo 1. Quindi una volta definite alcune classi, occorre descrivere la struttura interna dei concetti.
Ovviamente, dopo aver individuato le classi principali, molti dei termini che rimangono nella lista stesa al passo 3 sono dei buoni candidati per diventare proprietà di queste classi.

Per ciascuna proprietà nella stabilire quale classe descrive. Queste proprietà diventano *slot* delle classi. In generale ci sono molti tipi proprietà degli oggetti che possono diventare slot nell'ontologia:

- Proprietà *intrinseche* (nell'esempio dei vini potrebbe essere il gusto)
- Proprietà *estrinseche* (ad esempio il nome o l'area di provenienza di un vino)
- *Parti,* se l'oggetto è strutturato; queste possono essere sia fisiche che astratte (ad esempio le varie portate di una cena)
- *Relazioni* con altri individui; sono relazioni tra membri individuali di una classe con altri oggetti (ad esempio il produttore di un vino, che rappresenta una relazione tra un vino e un'azienda vinicola, e l'uva con cui il vino è stato prodotto).

In aggiunta alle proprietà identificate inizialmente, è necessario aggiungere alle classi tutti gli slot che ne derivano. Una sottoclasse eredita inoltre tutti gli slot della superclasse.

✍ *sameAs*

Questa proprietà nativa dell'OWL è stata utilizzata nell'ontologia per gestire il problema dei sinonimi a livello di individuals. Si è scelto infatti di introdurre nell'ontologia alcuni dei sinonimi più ricorrenti dei concetti principali individuati inizialmente (il concetto principale, non essendo un sinonimo, non presenta la proprietà). In tal modo alcuni concetti, se pur differenti dal punto di vista morfologico e grafico, possono facilmente essere considerati equivalenti dal punto di vista semantico. Ad esempio i tre concetti *coro*, *gruppo_corale* e *complesso_vocale* sono stati considerati come sinonimi e tra questi *coro* è quello principale.

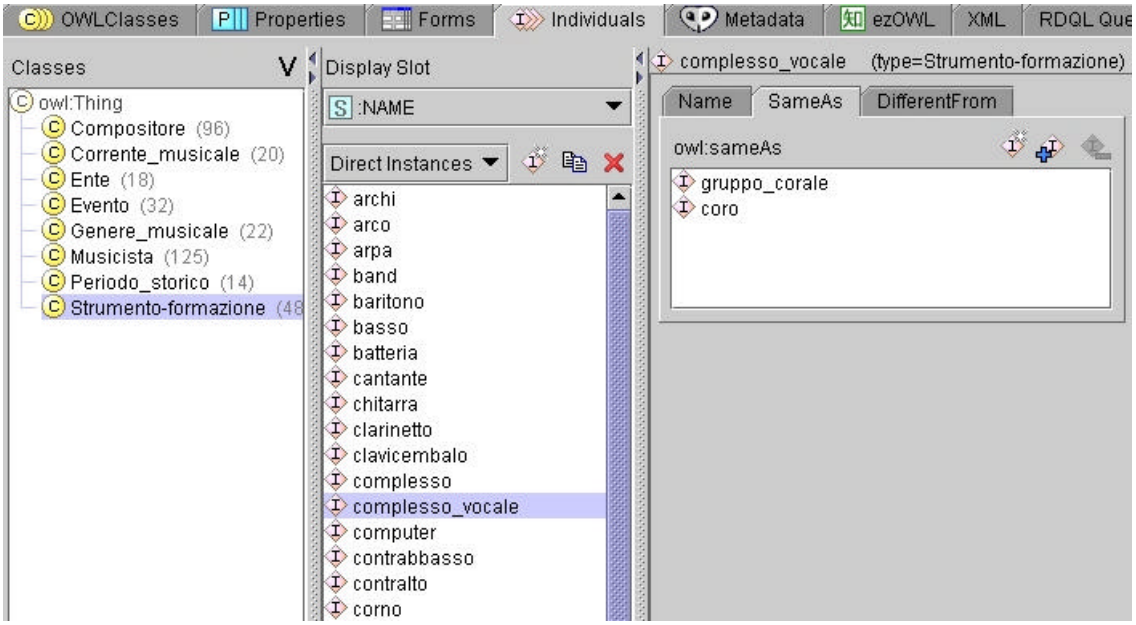

Figura 14 – Proprietà sameAs

✍ *seeAlso*

Questa proprietà nativa di RDF è stata utilizzata nell'ontologia per fornire una minima strutturazione gerarchica a certi concetti individuati, visto che la gerarchizzazione è possibile solo a livello di classi e non di istanze. In tal modo si è ovviato al problema di dover atomizzare e separare alcuni concetti in classi autonome. L'esempio è quello di *cantante* che presenta 6 proprietà *seeAlso* relative alle 6 sottotipologie che questo concetto può avere.

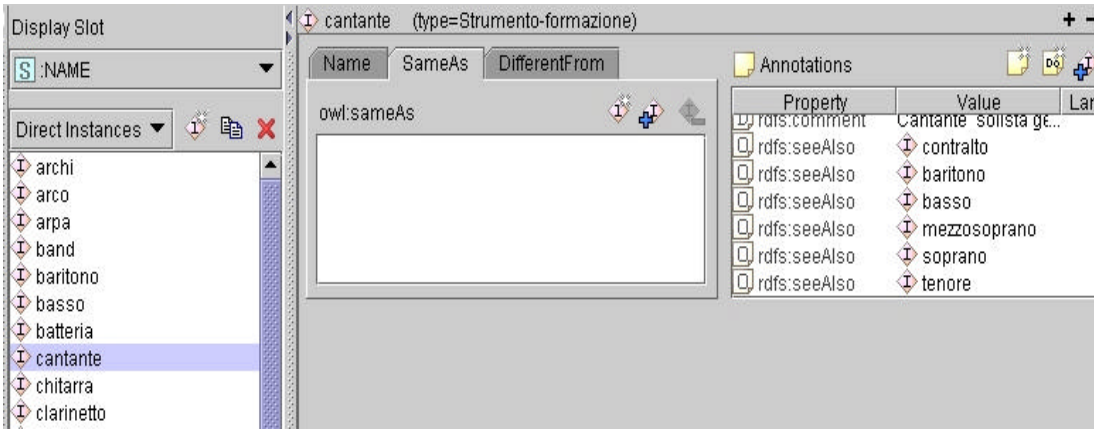

Figura 15 - Proprietà seeAlso (screenshot esemplificativo)

4.3.6. Definizione delle restrizioni sugli slot

Gli slot possono avere diversi tipi di restrizioni che descrivono i tipi di valori, i valori ammessi, la cardinalità e altre caratteristiche che i valori possono avere.

Cardinalità

La cardinalità definisce quanti valori può assumere uno slot. Alcuni sistemi distinguono solo tra cardinalità singola o multipla, mentre altri consentono di specificare un valore minimo e uno massimo di cardinalità per descrivere in modo più accurato il numero dei valori degli slot.

Tipo di valori

Una restrizione sul tipo di valori descrive quali tipi base possono essere inseriti negli slot. I tipi più comuni sono:

- *String* (il tipo più semplice e comunemente usato, costituito da una stringa di testo)
- *Number* (descrive i valori in termini numerici; spesso sono usati i tipi più specifici come *Float* o *Integer*)
- *Boolean* (è un semplice flag si-no)
- *Enumerated* (specifica una lista predefinita di valori dello slot)
- *Instance* (consente la definizione di relazioni tra istanze individuali; gli slot con tipo di valori *Instance* deve definire anche una lista di classi dalle quali possono provenire le istanze)

Dominio e range

Le classi ammesse per uno slot di tipo Instance sono definite come il *range* dello slot stesso. Le classi alle quali è collegato uno slot o una classe le cui proprietà sono descritte da uno slot sono chiamate *dominio* dello slot. Spesso il dominio di uno slot è proprio la stessa classe a cui è connesso. Per questo motivo non è necessario definire separatamente il dominio.

Per quanto riguarda le cardinalità non sono stati espressi dei vincoli in quanto l'ontologia è stata sviluppata in linguaggio OWL Lite, che supporta solamente cardinalità di tipo uno a uno (*single values*).

Tutte le tre proprietà, essendo delle relazioni a livello di istanza, ammettono necessariamente valori di tipo *individual*.

Per le due proprietà predefinite *seeAlso* e *sameAs* non si sono espressi dei vincoli su range e dominio, mentre *localizzazione_storica* ha un dominio definito dalle classi *Genere_musicale*, *Compositore* e *Corrente_musicale* (ovvero le tre classi su cui può essere definita la proprietà) e ha come range le istanze della classe *Periodo_storico* (ovvero la classe alla quale le tre precedenti possono essere relazionate tramite la proprietà stessa).

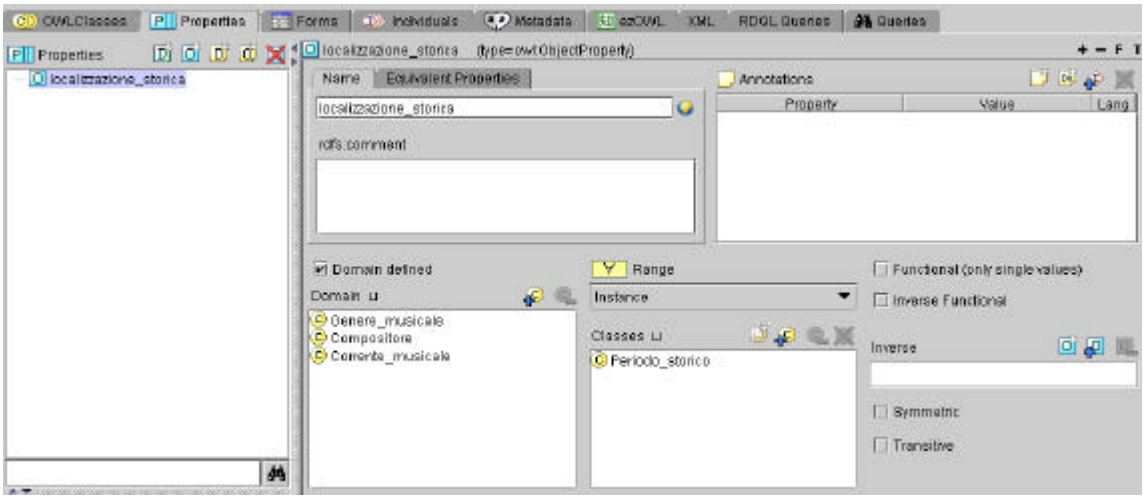

Figura 16 - Dominio e range della proprietà localizzazione_storica (screenshot esemplificativo)

4.3.7. Creazione delle istanze

L’ultimo passo consiste nel creare tutte le istanze individuali delle classi. Per definire ogni istanza di una classe occorre:

- Scegliere una classe
- Creare le istanze
- Per ogni istanza riempire i campi degli slot

Per ognuna delle classi create si è provveduto all'inserimento di tutte le istanze individuate all'interno degli articoli di riferimento e di altre, seppur non presenti, ma ritenute utili al fine di costruire un'ontologia efficiente e il più possibile completa.
Per le istanze aventi delle proprietà si è provveduto anche all'inserimento di tutti i dati necessari. Di seguito si riportano alcuni esempi di istanze create per alcune delle classi dell'ontologia.

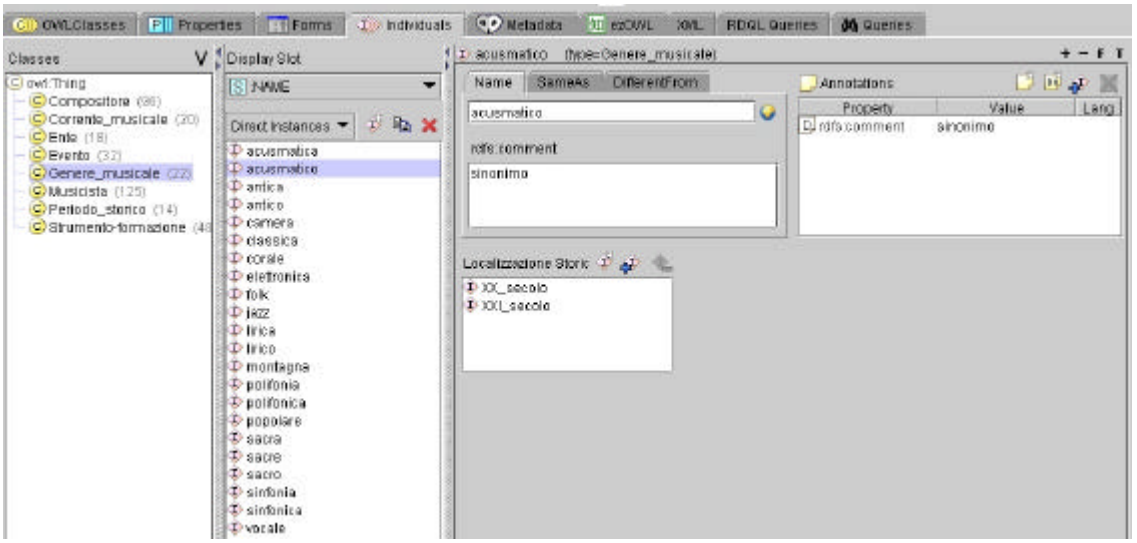

Figura 17 - Istanze della classe Genere_musicale

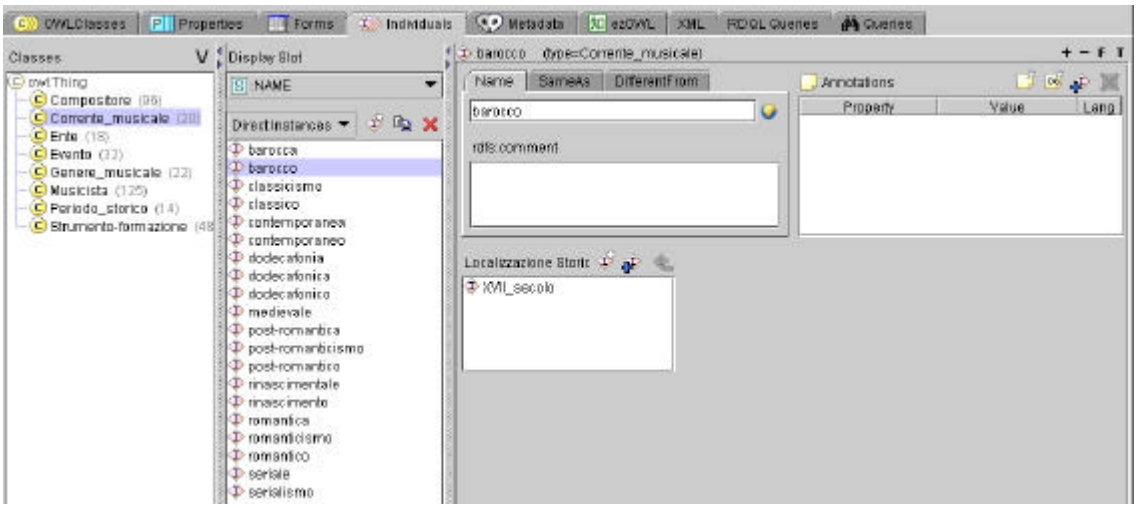

Figura 18 - Istanze della classe Corrente_musicale

5. DESCRIZIONE DELL'IMPLEMENTAZIONE

5.1. Software e linguaggi utilizzati

L'applicazione realizzata è un insieme di classi, servlet e pagine web sia statiche sia dinamiche che risiedono completamente all'interno di un application server. Il tutto è stato implementato in linguaggio Java a partire da un'applicazione esistente: il navigatore web vocale *Violino*, basata sull'architettura *Violin*, realizzata per fornire un framework di supporto all'implementazione di sistemi multimodali. Tale applicazione effettuava una conversione delle pagine HTML in VoiceXML per permetterne la navigazione in modo sequenziale o gerarchico. Il navigatore "Violino" è stato sfruttato per quanto riguarda il trattamento in Java dei formati HTML e VoiceXML. In particolare sono state riutilizzate alcune classi che si occupano dell'interfacciamento con il parser WebWiseFree e il marshaller CastorXML.
L'application server utilizzato durante le prove è il server Jakarta Tomcat versione 5.0.19.
Per quanto riguarda il supporto vocale è stato invece utilizzato l'IBM WebSphere Voice Server SDK versione 3.1.1.
Per quanto riguarda la realizzazione dell'ontologia è stato utilizzato il software Protege 2.1 ed i relativi plugin per il linguaggio OWL ed in particolare il plugin standard protege.owl e ezOWL (un tool grafico per la creazione e la gestione di ontologie). Per gestire l'ontologia all'interno delle classi Java sono state utilizzate le API Jena 2.1.

5.2. Funzionalità base

L'implementazione realizzata permette di navigare il sito di prova tramite le modalità di navigazione semantica progettate in precedenza, cioè la navigazione semantica "per concetti" e la "ricerca semantica". Oltre a queste due modalità il prototipo include la possibilità di effettuare una conversione dell'*HTML* e di navigare la pagina in modo sequenziale. Tale funzionalità non è stata implementata appositamente, ma è stata presa da un'applicazione già esistente (il già citato navigatore "Violino") con lo scopo di permettere un confronto tra le varie modalità di navigazione durante l'analisi dei risultati, in modo da valutarne le differenze in termini di vantaggi e svantaggi.

Il prototipo propone quindi tre modalità distinte di navigazione della pagina visitata. Durante la navigazione sarà possibile in qualsiasi momento cambiare la modalità.
Per quanto riguarda le due modalità di navigazione semantica bisogna dire che la navigazione della pagina "per concetti" permette di navigare la totalità dei contenuti della pagina (ed è quindi confrontabile con la navigazione sequenziale), mentre la "ricerca semantica" è uno strumento utile solamente per cercare determinati argomenti all'interno dei contenuti. Per questo motivo quest'ultima modalità non è confrontabile con le altre due e non può essere considerata una vera e propria "navigazione" della pagina. Tuttavia entrambe le modalità di navigazione semantiche, come del resto la navigazione sequenziale, permettono di svolgere le stesse funzionalità base della navigazione web, cioè la lettura (in questo caso sarebbe più indicato "ascolto") dei contenuti e la navigazione dei collegamenti ad altre pagine. La navigazione delle pagine indicate dai collegamenti avviene allo stesso modo della pagina in cui ci si trovava precedentemente: ad ogni nuova pagina si può scegliere la modalità di navigazione, che può essere sempre cambiata in qualsiasi momento.

5.3. Realizzazione del sito di test

Il prototipo dell'applicazione è stato realizzato con lo scopo di valutare le possibilità e i vantaggi effettivi ottenibili con le metodologie di navigazione semantica ipotizzati nella fase di progettazione. Per questo motivo si è pensato di concentrarsi su uno specifico caso di test attraverso il quale sia possibile analizzare al meglio i risultati ottenuti. E' stato quindi ideato un sito di prova della tipologia individuata precedentemente, cioè con quantità di contenuti rilevante e riguardante un ambito ristretto in termini di argomenti. Come punto di partenza per la costruzione dell'esempio è stato preso l'archivio degli articoli del Conservatorio di Musica G. Verdi di Como (che può essere consultato all'indirizzo http://www.conservatoriocomo.it/archiv/index.html). L'ontologia che sta alla base del funzionamento dell'applicazione è stata creata in base ai contenuti presenti in tali articoli, come già precedentemente discusso. Anche il sito di test naturalmente è stato costruito utilizzando come contenuti un certo numero degli articoli presenti sul sito del Conservatorio. La versione finale dell'esempio è costituita da alcune pagine contenenti gli articoli scelti. Sono state realizzate sia pagine che presentano articoli su argomenti vari (come è il caso di un archivio in cui gli articoli sono inseriti in ordine temporale) sia pagine che raccolgono articoli su un determinato argomento.

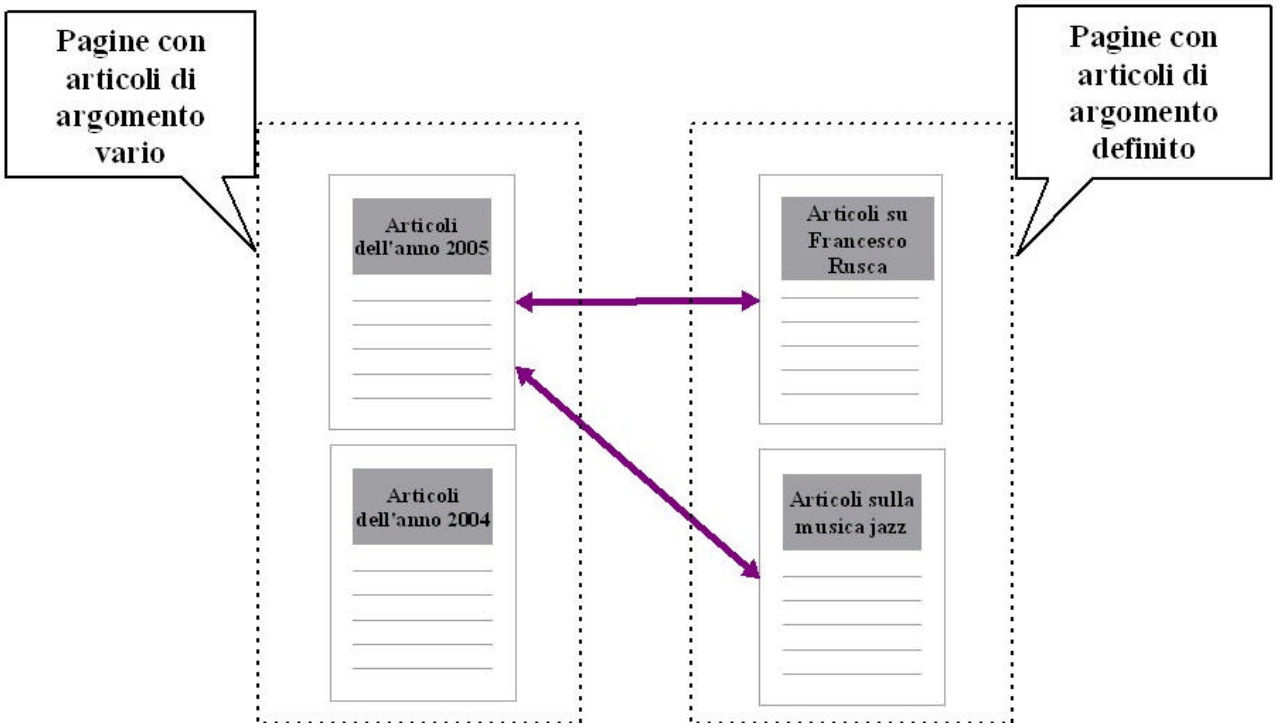

Figura 19 - Tipi si pagine e struttura del sito di test

5.4. Funzionamento dell'applicazione

5.4.1. Avvio dell'applicazione

Per avviare l'applicazione l'utente deve semplicemente eseguire una request per la pagina di avvio all'interno di un ambiente vocale, cioè tramite un voice server.

All'avvio l'applicazione legge un messaggio di benvenuto e informa l'utente sui comandi globali del navigatore. Questi sono particolari comandi che l'utente può dare in qualsiasi momento durante l'esecuzione delle pagine vocali. Essi generano un evento che interrompe la normale esecuzione del VoiceXML e permette di eseguire alcune operazioni speciali.

Il primo di questi comandi è il comando "menu". Quando l'utente chiama questo comando l'applicazione torna al menu principale dal quale, come si vedrà in seguito, è possibile cambiare la modalità di navigazione della pagina. Questo comando deve essere dato anche all'avvio per iniziare la navigazione del sito di prova.

Il secondo comando è il comando “termina”. Questo comando permette all’utente di far terminare l’esecuzione dell’applicazione. Quando l’utente chiama questo comando l’applicazione si chiude.
Oltre a questi due è presente anche un altro comando che, pur non essendo globale, è il caso di introdurre a questo punto della descrizione. Il comando in questione è il comando “ferma”. Questo comando può essere chiamato ogni volta che l’applicazione sta leggendo un brano. Chiamando questo comando l’utente può interrompere la lettura del brano in questione e saltare al brano successivo. Il comando “ferma” è presente solamente nelle modalità di navigazione semantica, poiché non era presente originariamente nel navigatore “Violino”.

5.4.2. Menu principale e scelta della modalità di navigazione

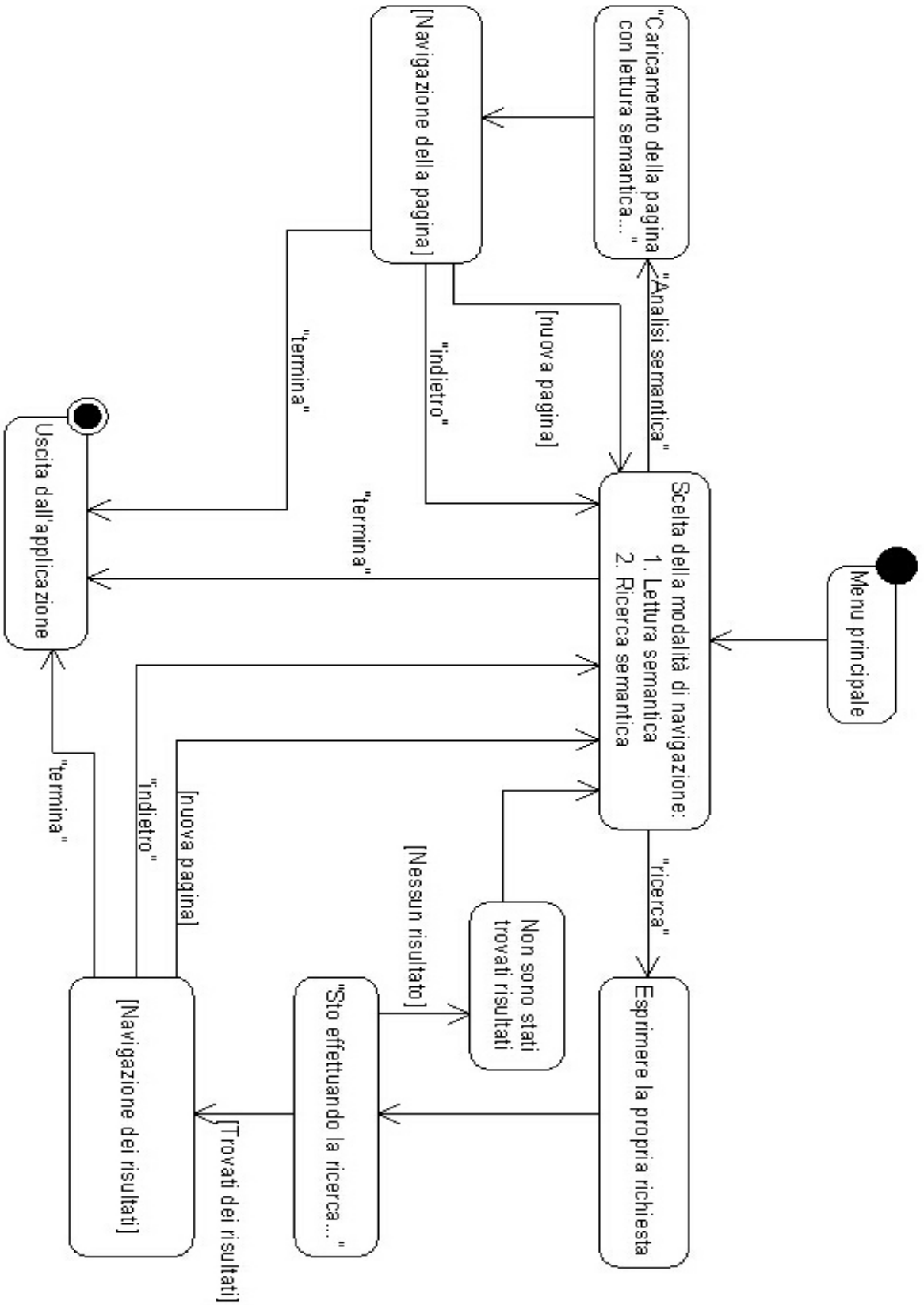

Figura 20 - Mappa navigazionale relativa al menu principale

Dal menu principale l'utente può sceglierla modalità di navigazione che vuole effettuare sulla pagina che sta visitando attualmente. Naturalmente all'avvio dell'applicazione si tratta della home page del sito di prova, ma, come precedentemente spiegato, a questo menu è possibile accedere in qualsiasi momento semplicemente chiamando il comando "menu". L'applicazione descrive le modalità di navigazione e l'utente può fare la propria scelta.
Se l'utente sceglie di effettuare una lettura semantica della pagina l'applicazione passa direttamente al caricamento della pagina e avvisa l'utente con un messaggio di caricamento. Infatti, per questa modalità di navigazione non è necessario alcun input da parte dell'utente in questa fase. L'utente può quindi navigare la pagina con la lettura semantica come si vedrà in seguito. Al termine della navigazione l'utente può scegliere di terminare l'applicazione oppure di ritornare al menu principale per selezionare una nuova modalità. Se invece si è scelto, durante la navigazione della pagina, di navigare un collegamento l'applicazione tornerà al menu principale in cui si potrà scegliere la modalità di navigazione della nuova pagina.
Se l'utente sceglie, invece, di effettuare una ricerca semantica all'interno della pagina l'applicazione chiederà di esprimere una richiesta. A questo punto l'utente potrà esprimere la propria richiesta. Quando la richiesta sarà stata acquisita l'applicazione provvederà ad effettuare la ricerca all'interno della pagina, avvisando l'utente con un messaggio di caricamento. A questo punto se non saranno stati trovati risultati l'applicazione avviserà l'utente e poi tornerà al menu principale. Se saranno stati trovati dei risultati l'utente potrà navigarli, come si vedrà in seguito. Alla fine della navigazione dei risultati l'utente può scegliere di terminare l'applicazione oppure di ritornare al menu principale per selezionare una nuova modalità. Se invece si è scelto, durante la navigazione della pagina, di navigare un collegamento l'applicazione tornerà al menu principale in cui si potrà scegliere la modalità di navigazione della nuova pagina.

Si riporta ora un esempio che mostra l'interazione tra l'applicazione e l'utente riferito all'avvio dell'applicazione e al menu principale:

Applicazione: Benvenuto nel prototipo di navigatore semantico ad interfaccia vocale.
Per iniziare, dire: menu.
Per sapere la lista dei comandi principali, dire: aiuto.

Utente: menu

A: Caricamento menu principale...

A: Scelta della modalità di navigazione.

Per effettuare una lettura semantica della pagina, dire: 1.

Per definire una ricerca semantica all'interno della pagina, dire: 2.

Per effettuare una lettura sequenziale, dire: 3.

5.4.3. Navigazione della pagina con lettura semantica

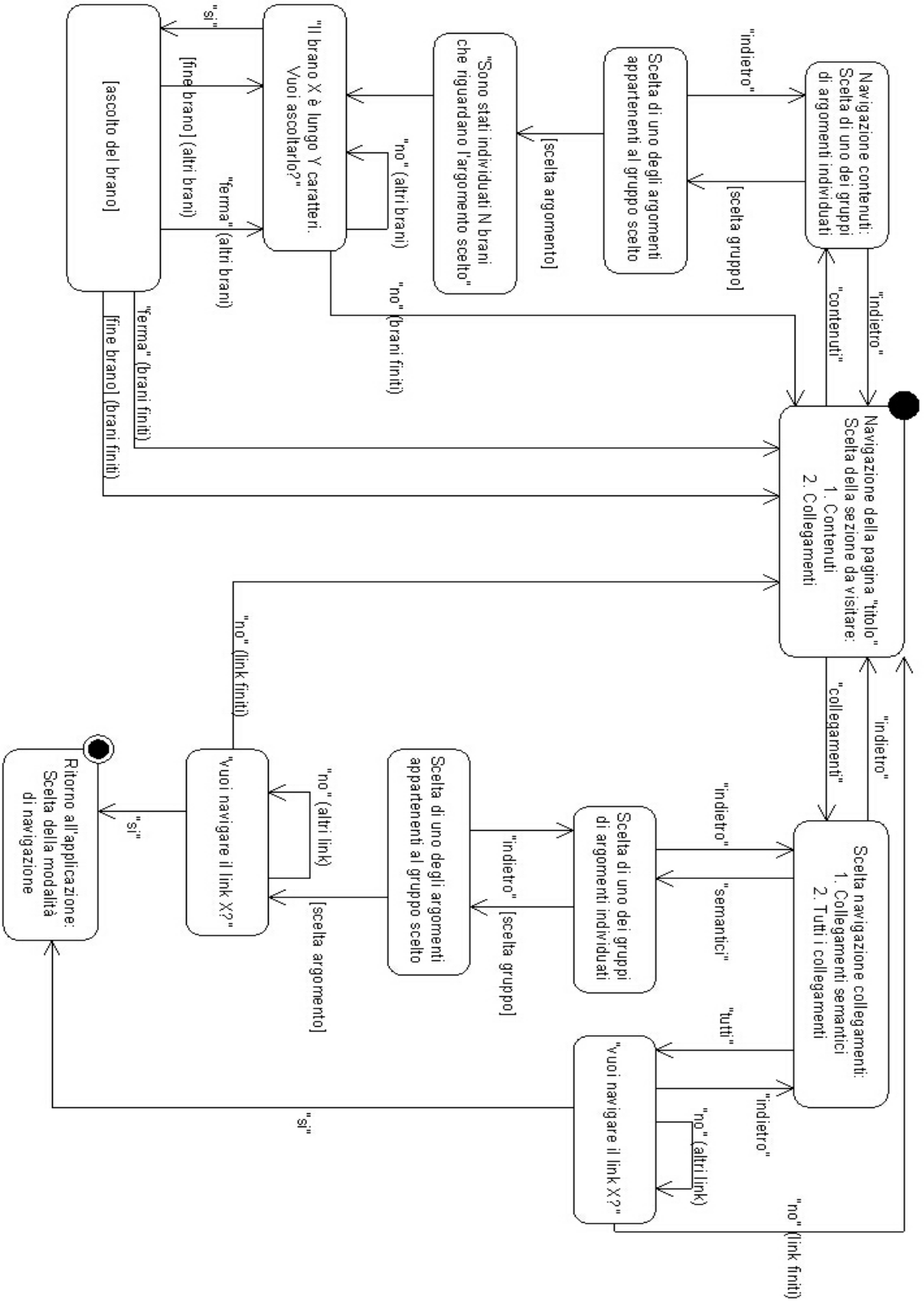

Figura 21 - Mappa navigazionale relativa alla navigazione semantica della pagina

Se l'utente ha precedentemente scelto di navigare la pagina con la lettura semantica dopo la fase di caricamento l'applicazione lo informerà del titolo della pagina. Inoltre verrà proposta la scelta di navigare i contenuti oppure i collegamenti, poiché con le modalità di navigazione semantica essi vengono navigati separatamente.

Se l'utente sceglie di navigare i contenuti l'applicazione farà l'elenco dei "gruppi di argomenti" che sono stati individuati nella pagina. Tali "gruppi di argomenti" sono le categorie in cui possono essere divisi tutti gli argomenti (che sarebbero i concetti dell'ontologia) presenti all'interno della pagina. L'utente deve scegliere uno dei gruppi oppure può tornare al passo precedente, cioè alla scelta di navigare i contenuti oppure i concetti.

Se sceglie uno dei gruppi l'applicazione proporrà l'elenco di tutti gli argomenti individuati nella pagina appartenenti al gruppo selezionato. L'utente potrà quindi scegliere uno degli argomenti oppure scegliere di tornare indietro al menu di scelta dei "gruppi di argomenti".

Se l'utente sceglie uno degli argomenti appena elencati l'applicazione lo informerà riguardo al numero di brani che riguardano l'argomento selezionato.

Dopodichè l'applicazione informerà l'utente riguardo la lunghezza del primo brano individuato (l'informazione avviene attraverso il numero di caratteri di cui è composto il brano). Inoltre propone all'utente la scelta di farsi leggere un determinato brano oppure di saltarlo.

Se si sceglie di farsi leggere un brano l'applicazione comincerà con la lettura. Come scritto in precedenza l'utente ha la possibilità di invocare il comando "ferma" e di interrompere la lettura del brano. Nel caso che l'utente chiami tale comando l'applicazione si comporterà come se avesse raggiunto la fine del brano.

Se l'utente ha invece scelto di non farsi leggere il brano si ricade ancora una volta nella stessa situazione. In pratica il brano può venire letto fino alla fine, può venire letto parzialmente e poi interrotto oppure può venire saltato. In tutti e tre i precedenti casi l'applicazione si troverà nella stessa situazione.

Infatti, a questo punto potrebbero esserci altri brani che ancora devono essere presentati all'utente. L'applicazione in questo caso informerà l'utente che il secondo brano ha una certa lunghezza e proporrà la scelta di leggerlo oppure di saltarlo, rieseguendo la stessa sequenza appena descritta. Il processo continuerà finchè sono presenti altri brani da proporre all'utente.

Quando invece non ci saranno più brani che riguardano il particolare argomento in questione in ognuno dei tre casi precedentemente descritti (si è raggiunta la fine dell'ultimo brano, la lettura dell'ultimo brano è stata interrotta oppure si è scelto di saltare l'ultimo brano) l'applicazione tornerà all'inizio della navigazione, rileggendo il titolo della pagina e riproponendo la scelta tra i contenuti e i collegamenti.

Come è stato spiegato quando l'utente seleziona un argomento gli vengono presentati tutti i brani che riguardano l'argomento scelto. E' importante notare che ogni brano riguarda di solito un certo numero di argomenti. Per questo motivo un particolare brano può essere presentato all'utente più volte quando vengono scelti diversi argomenti di cui tale brano tratta. Perciò l'utente potrebbe continuamente trovarsi ad ascoltare gli stessi brani quando seleziona argomenti diversi. Per evitare questa situazione l'applicazione tiene traccia dei brani che sono stati già letti, perciò la volta successiva che dovrà presentare all'utente uno di questi brani lo informerà del fatto che il brano in questione è già stato letto. In ogni caso chiederà se l'utente desidera rileggerlo un'altra volta.

Se l'utente ha scelto inizialmente di navigare i collegamenti l'applicazione gli proporrà una scelta ulteriore. Chiederà infatti se l'utente desidera navigare i collegamenti "semantici" oppure se desidera navigare tutti i collegamenti della pagina. Questa scelta è richiesta dal fatto che, a differenza dei contenuti, i collegamenti presenti all'interno di una pagina web sono costituiti generalmente da una breve frase che funge da descrizione della pagina collegata. A causa della sua brevità, però, è difficile individuarne la semantica (tramite il confronto con i concetti dell'ontologia). Perciò, per evitare che una buona parte dei collegamenti della pagina non siano effettivamente navigabili viene proposta all'utente la possibilità di navigare tutti i collegamenti. Bisogna però notare che tale navigazione avviene senza semantica, poiché i collegamenti vengono esposti all'utente in sequenza.

A questo punto se l'utente ha scelto di navigare i collegamenti "semantici" si procederà in modo analogo al caso della navigazione dei contenuti. Infatti verranno elencati all'utente i "gruppi di argomenti". Dopo che l'utente ne avrà selezionato uno verranno elencati gli argomenti appartenenti al gruppo scelto.

Quando l'utente avrà selezionato un argomento l'applicazione inizierà a proporre i collegamenti che riguardano l'argomento scelto. Verrà letto il testo del collegamento e verrà proposta la scelta di navigarlo.

Se l'utente sceglie di non navigarlo l'applicazione continuerà a proporre tutti gli altri collegamenti che riguardano l'argomento selezionato. Quando i collegamenti in questione saranno terminati l'utente ritornerà alla scelta iniziale di navigare i contenuti o i collegamenti.

Se l'utente sceglie invece di navigare il collegamento l'applicazione termina la navigazione della pagina attuale e ritorna al menu di scelta in cui l'utente potrà selezionare la modalità di navigazione che intende utilizzare per la nuova pagina indicata dal collegamento.

La navigazione di tutti i collegamenti è esattamente identica a quella dei collegamenti semantici e si differenzia solamente per il fatto che non viene proposta nessuna scelta degli argomenti e l'applicazione inizia subito a leggere i link all'utente.

Si riporta ora un esempio di navigazione della pagina con lettura semantica:

Applicazione: Benvenuto nel prototipo di navigatore semantico ad interfaccia vocale.
Per iniziare, dire: menu.
Per sapere la lista dei comandi principali, dire: aiuto.

Utente: menu

A: Caricamento menu principale...

A: Scelta della modalità di navigazione.
Per effettuare una lettura semantica della pagina, dire: 1.
Per definire una ricerca semantica all'interno della pagina, dire: 2.
Per effettuare una lettura sequenziale, dire: 3.

U: uno

A: Caricamento pagina con lettura semantica, attendere prego...

A: Stai navigando la pagina “Archivio del Conservatorio”. Scelta della sezione da visitare:
per navigare i contenuti della pagina, dire: 1;
per navigare i collegamenti, dire: 2.

U: uno

A: All'interno della pagina sono stati individuati i seguenti gruppi di argomenti:
Musicista, Compositore, Corrente musicale, Ente, Genere musicale, Evento, Periodo storico, Strumento-formazione.
Scegliere uno dei gruppi.

U: Corrente musicale

A: Gli argomenti appartenenti al gruppo “Corrente musicale” sono i seguenti:
Rinascimento, romanticismo.
Scegliere uno degli argomenti oppure dire ‘indietro’ per tornare all'indice dei gruppi.

U: romanticismo

A: “Romanticismo” è presente in 2 brani.

A: Il brano 1 è lungo 1730 caratteri.
Vuoi ascoltarlo?
U: si
A: [L’applicazione legge il testo del brano]
A: Il brano 2 è lungo 2143 caratteri.
Vuoi ascoltarlo?
U: no
A: [Ritorno al menu iniziale]
A: Stai navigando la pagina “Archivio del Conservatorio”. Scelta della sezione da visitare:
per navigare i contenuti della pagina, dire: 1;
per navigare i collegamenti, dire: 2.
U: due
A: Per navigare i collegamenti semantici, dire: 1.
Per navigare tutti i collegamenti, dire: 2.
U: due
A: Il collegamento 1 è il seguente:
“Visualizza altri articoli su Francesco Rusca.”
Vuoi navigarlo?
U: no
A: Il collegamento 2 è il seguente:
“Vai agli articoli del periodo Luglio-Dicembre 2004.”
Vuoi navigarlo?
U: si
A: [Ritorno al menu di selezione]
A: Scelta della modalità di navigazione.
Per effettuare una lettura …
(…)

5.4.4. Navigazione della pagina tramite ricerca semantica

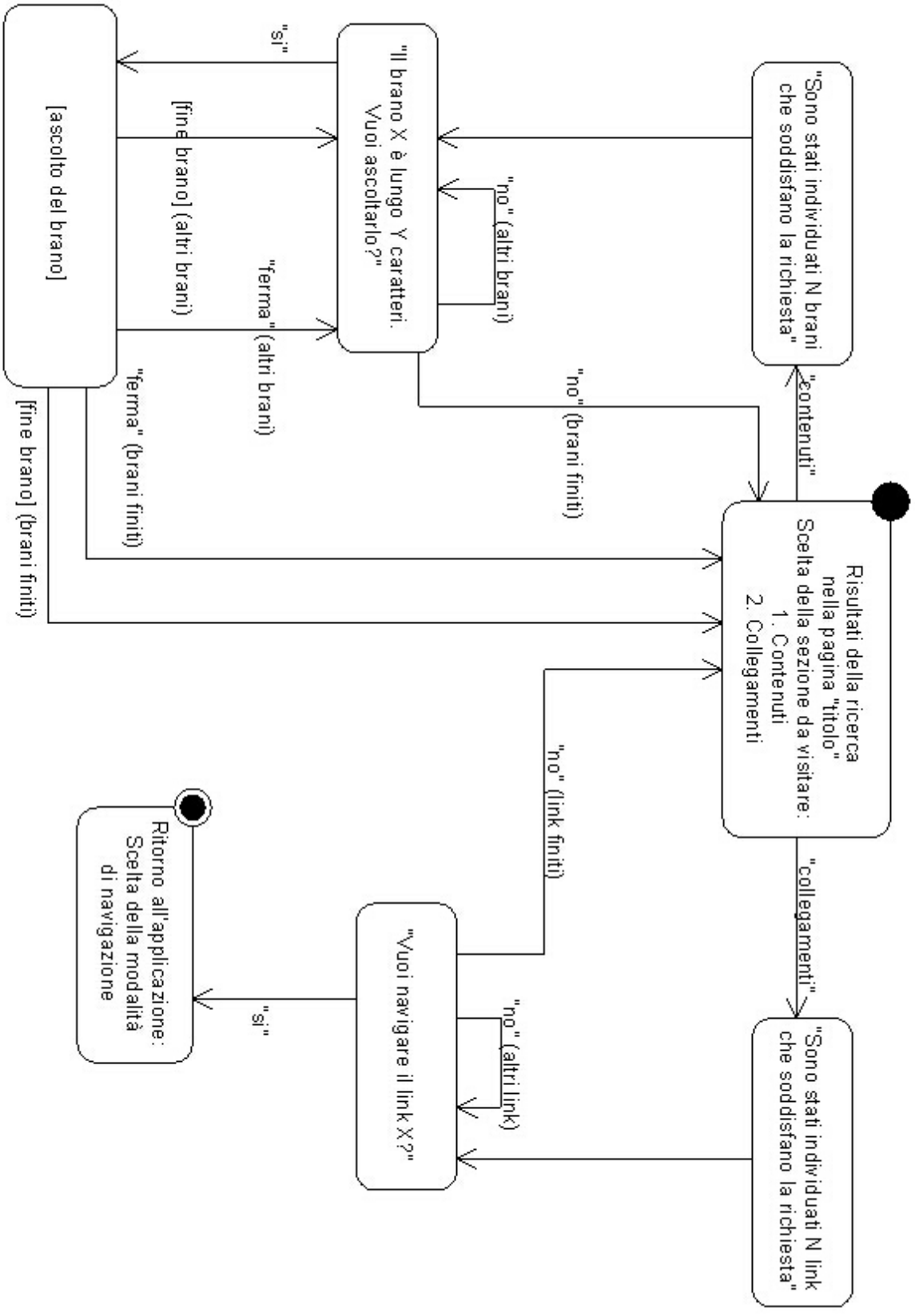

Figura 22 - Mappa navigazionale relativa alla ricerca semantica all'interno della pagina

La navigazione dei risultati di una ricerca semantica avviene in modo molto simile alla navigazione della pagina con lettura semantica.

All'inizio l'applicazione legge all'utente il titolo della pagina all'interno della quale è stata effettuata la ricerca. Dopodichè chiede all'utente se vuole navigare i risultati all'interno dei contenuti oppure all'interno dei collegamenti.

Se l'utente sceglie di navigare i brani che soddisfano la ricerca l'applicazione lo informa sul numero dei brani individuati e poi li propone in sequenza esattamente come avveniva nella navigazione con lettura semantica.

La navigazione dei collegamenti avviene allo stesso modo. I vari collegamenti che soddisfano la ricerca vengono letti all'utente, che ha la possibilità di scegliere se navigarli.

Si riporta ora un esempio di navigazione della pagina con ricerca semantica:

Applicazione: Benvenuto nel prototipo di navigatore semantico ad interfaccia vocale.
Per iniziare, dire: menu.
Per sapere la lista dei comandi principali, dire: aiuto.

Utente: menu

A: Caricamento menu principale...

A: Scelta della modalità di navigazione.
Per effettuare una lettura semantica della pagina, dire: 1.
Per definire una ricerca semantica all'interno della pagina, dire: 2.
Per effettuare una lettura sequenziale, dire: 3.

U: due

A: Esprimere la propria richiesta...

U: Ricerca qualcosa su Francesco Rusca

A: Sto effettuando la ricerca, attendere prego...

A: Ho effettuato la ricerca nella pagina "Archivio del Conservatorio".
Ho trovato 4 risultati.

A: Il brano 1 è lungo 1510 caratteri.
Vuoi ascoltarlo?

U: no

A: Il brano 2 è lungo 1730 caratteri.
Vuoi ascoltarlo?

U: si

A: [L’applicazione legge il brano]

U: termina

A: Termine dell'applicazione... Arrivederci!

6. VALUTAZIONE DEI RISULTATI, CONCLUSIONI E SVILUPPI FUTURI

6.1. Correttezza del funzionamento

In questa sezione l'applicazione realizzata verrà valutata dal punto di vista della correttezza di funzionamento. Si cercherà, cioè, di analizzare se il funzionamento reale dell'applicazione corrisponde con quello prefissato, ovvero se i risultati ottenuti sono quelli attesi.
Verrà valutata l'elaborazione delle pagine HTML sia per quanto riguarda le funzioni di estrazione dei contenuti rilevanti che per quanto riguarda l'analisi semantica degli stessi e l'individuazione delle corrispondenze con i concetti presenti nell'ontologia.

6.1.1. Estrazione dei contenuti rilevanti dalla pagina HTML

La prima funzione importante dell'applicazione che occorre testare è quella che si occupa dell'individuazione dei contenuti rilevanti per la navigazione semantica all'interno di una qualsiasi pagina HTML. Questo problema apparentemente semplice è complicato dal fatto che nel formato HTML i contenuti possono essere inseriti nella pagina attraverso vari modi. Il numero dei tag che possono contenere informazioni importanti che occorre estrarre è abbastanza alto e dato che ogni tag possiede proprietà differenti può risultare non immediato il metodo da impiegare per l'individuazione di tutti i contenuti all'interno di una pagina.
Per poter valutare l'effettiva validità del metodo impiegato per l'estrazione dei contenuti sono state effettuate varie prove utilizzando pagine HTML formattate in modi differenti. I casi più semplici, quelli formati con i contenuti inseriti nei tag classici e ad un solo livello di profondità, non hanno creato nessun problema. I casi più complessi, con i contenuti inseriti in tag non "canonici", ad alto livello di profondità e con tag di formattazione particolare, hanno invece richiesto un certo numero di correzioni nel procedimento di estrazione. Effettuate tutte le correzioni l'estrazione dei contenuti si è rivelata efficace anche nelle pagine HTML malformate, cioè con usi impropri dei tag di formattazione. Durante le prove nessun brano di contenuto è stato ignorato.

6.1.2. Elaborazione dei contenuti dal punto di vista semantico

Una volta estratti i contenuti rilevanti di una pagina è necessario effettuare un'analisi semantica. In pratica l'applicazione dovrebbe individuare le corrispondenze tra i contenuti della pagina e i concetti presenti nell'ontologia. Questa operazione avviene sia durante l'analisi per permettere la navigazione semantica, sia per effettuare le ricerche all'interno della pagina. A questo livello, mettendo da parte per il momento i problemi legati alla gestione dei sinonimi, di cui parleremo in seguito, l'importante è che l'applicazione riesca ad individuare tutte le corrispondenze tra i contenuti e i concetti dell'ontologia.
Per valutare l'efficacia di questa operazione basta semplicemente confrontare il risultato dato dall'applicazione con quello atteso, visto che è molto semplice individuare le corrispondenze confrontando "manualmente" un brano e l'ontologia.

Prendiamo per esempio il seguente brano:

"…La novità dell'anno è sicuramente la collaborazione con la storica Stagione concertistica del Carducci, dopo il ripristino dello storico organo pneumatico Mascioni del 1913 progettato personalmente da Marco Enrico Bossi. Il binomio Carducci-Conservatorio prenderà il via il prossimo 17 febbraio con un concerto di Massimo Nosetti e del gruppo di Archi del Conservatorio dedicato all'Organo da concerto nel Settecento, da Haendel a Mozart; a seguire , lunedì 28 febbraio, L'organo in sala da concerto con Cristina Rubin, soprano e Marco Rossi all'organo e un trittico solistico nelle domeniche 5, 19 e 26 giugno. Immancabile risulta la serie di concerti dedicati alla coralità nelle sue diverse forme Polyphoniae. Dopo le edizioni che hanno affrontato aspetti specifici del repertorio e della prassi esecutiva corale, il settimo ciclo sarà dedicato a concerti e momenti di incontro sulla coralità in senso stretto nelle sue varie forme…"

All’interno di questo brano sono presenti i seguenti concetti dell’ontologia:

“concerto, Carducci, organo, conservatorio, Nosetti, archi, settecento, Haendel, Mozart, Rubin, soprano, Rossi, Polyphoniae”.

Se costruiamo una pagina di test che contenga solamente il brano precedente è semplice controllare se i risultati ottenuti dall’applicazione sono esatti. L’applicazione propone all’utente la seguente struttura degli argomenti:

```
Musicista:
     Nosetti
     Rubin
     Rossi
Compositore:
     Haendel
     Mozart
Ente:
     Carducci
     conservatorio
Evento:
     concerto
     Polyphoniae
Periodo storico:
     settecento
Strumento-formazione:
     organo
     archi
     soprano
```

Si può notare che tutti i concetti sono stati individuati correttamente.

6.1.3. Gestione dei sinonimi

Quando si analizza una pagina HTML per poter effettuare una lettura semantica non è sufficiente individuare tutte le corrispondenze tra contenuti e concetti dell'ontologia. Infatti ogni concetto può trovarsi in varie forme all'interno della pagina e se non venisse effettuato nessun tipo i valutazione l'applicazione ritornerebbe all'utente tutte le forme individuate di un particolare concetto, quando in realtà l'argomento è solamente uno. E' necessario ritornare all'utente solamente il concetto principale. Questo però crea un problema: infatti anche se viene ritornato solo il concetto principale le corrispondenze dei suoi sinonimi non devono essere perse. Perciò l'applicazione deve essere in grado di riferire correttamente le corrispondenze dei sinonimi al concetto principale.
Per valutare la correttezza di questo procedimento bisogna effettuare un controllo sui risultati generati dall'applicazione durante l'analisi di un brano contenente un concetto che nell'ontologia è presente come sinonimo.

Prendendo per esempio il brano:

"…Quattordici appuntamenti aspettano il pubblico per la serie Sabato in musica 2005, confermata presso l'Auditorium di via Cadorna nell'orario con inizio alle 18.15. Docenti del Conservatorio, ex allievi in carriera e guppi proporranno incontri con La sonata nel Novecento, Schubert, Brahms, Mendelssohn, Piazzolla, Gerswhin, Rota, Morricone, Williams. Tre cicli di melodie per voce e pianoforte, Band and Rhapsody ispiratrici, Big Band Jazz, Il sassofono classico, Omaggio a Fernand Sor, Reubke, una vita breve ma intensa, Le boef sur le toit, Schumann: Frauenliebe e Dichterliebe , Franz Schubert: La Trota, La Variazione: tecnica amata in tutte le epoche. Ce n'è per tutti i gusti, rigorosamente ad ingresso libero…"

notiamo che esiste una corrispondenza con il concetto "sassofono". Questo concetto nell'ontologia è un sinonimo di "sax". Se l'applicazione esegue correttamente le sue funzioni nella pagina vocale presentata all'utente dovrà essere presente non il concetto "sassofono", ma il concetto "sax". Inoltre il brano precedente dovrà essere indicato come contenente il concetto

“sax”. In pratica, anche se il brano non contiene il concetto “sax”, ma un suo sinonimo, l’applicazione deve essere in grado di individuare correttamente la corrispondenza.

Se effettuiamo un test sul brano precedente, in effetti, l’applicazione ritorna:

…

```
Strumento-formazione:
     pianoforte
     sax
```

…,

cioè la corrispondenza è come dovrebbe essere e il sinonimo non viene presentato all’utente.

6.1.4. Ricerche semantiche

Il procedimento con cui l’applicazione effettua una ricerca semantica è praticamente lo stesso attraverso il quale viene realizzata l’analisi semantica della pagina. Infatti una volta estratti dalla richiesta i termini della ricerca (attraverso un metodo che verrà discusso successivamente), l’applicazione procede semplicemente ad individuarne le corrispondenze con i contenuti della pagina. Da valutare, invece, è il procedimento attraverso il quale vengono trattati alcuni concetti, che generano ricerche con più termini. Tali concetti possiedono un attributo che fa riferimento ad altri concetti simili o che hanno lo stesso valore semantico ai fini della ricerca.

Un ottimo esempio è costituito dal concetto “cantante”. Se l’utente vuole effettuare una ricerca semantica che riguarda i cantanti, ovviamente l’applicazione la riferirà proprio al concetto “cantante”. Tuttavia tale concetto fa riferimento ad altri concetti dell’ontologia, come ad esempio “soprano”, “basso” o “baritono”, che ai fini della ricerca hanno lo stesso valore in quanto sono, appunto, cantanti.

Per valutare se una ricerca del concetto “cantante” valuta anche questi concetti aggiuntivi, basta effettuare una ricerca in una pagina di prova i cui brani abbiano delle corrispondenze con uno o più di questi concetti aggiuntivi, ma che non abbiano nessuna corrispondenza con il semplice concetto “cantante”.

Come esempio si può usare il brano seguente:

"…«La vita senza la musica sarebbe un errore»: tale citazione citabile di Nietzsche ha costituito il fil rouge di un programma vario che ha avuto come denominatore comune la cantabilità strumentale. Il diavolo, però, ci ha messo la coda, poiché i tre interpreti sono diventati due per l'indisposizione del soprano Cristina Rubin. Sicché il clarinettista Carlo Dell'Acqua e la pianista Claudia Bracco, hanno dovuto reinventare il programma, cercando di mantenersi in linea con il motto-traccia, determinante per la compilazione del programma. Diremmo che ci sono riusciti, andando a indagare brani di autori poco conosciuti, ma degni di essere ascoltati…"

La ricerca del concetto cantante, in effetti, trova una corrispondenza nel brano precedente, che contiene però non il concetto "cantante" in sé, ma il concetto "soprano", a cui il concetto "cantante" fa riferimento. Questo procedimento particolare utilizzato nelle ricerche semantiche risulta quindi effettivamente efficace.

6.2. Analisi qualitativa del funzionamento

In questa sezione l'applicazione verrà valutata dal punto di vista della qualità. Verranno cioè valutate le caratteristiche positive e negative del suo funzionamento, sotto vari punti di vista o riguardo a particolari aspetti.
Verrà effettuato, dove possibile e significativo, un confronto tra la modalità di navigazione semantica proposta dall'applicazione e la navigazione sequenziale sulle stesse pagine, mettendo in evidenza alcuni particolari aspetti.
Verranno poi valutate alcune caratteristiche dell'applicazione dal punto di vista dell'usabilità, in particolar modo riguardo alla comodità e alla semplicità di utilizzo.
Inoltre verranno valutate le qualità e i difetti della modalità di ricerca semantica.

6.2.1. Velocità computazionale

Nonostante l'applicazione realizzata sia soltanto un prototipo è bene valutare brevemente anche la sua velocità computazionale. In pratica si tratta di valutare il tempo impiegato dall'applicazione per ottenere la pagina HTML da analizzare, estrarne i contenuti rilevanti, individuare le corrispondenze con i concetti dell'ontologia e costruire la pagina VoiceXML finale che verrà presentata all'utente. Bisogna innanzitutto trattare separatamente l'analisi della pagina per una lettura semantica e la ricerca semantica all'interno della pagina. Infatti, come si vedrà tra poco i tempi di computazione sono sostanzialmente differenti.

Prima di discutere del tempo necessario alla computazione della pagina per permettere una navigazione semantica bisogna specificare che l'applicazione non utilizza nessun metodo per evitare di ricomputare la stessa pagina ad ogni richiesta, perciò le richieste già effettuate in precedenza non variano il tempo di computazione delle richieste successive. Dai test effettuati si nota che l'applicazione impiega una certa quantità di tempo (di solito tra i dieci e i venti secondi) per elaborare la pagina finale presentata all'utente. Sono stati effettuati vari test con pagine HTML di lunghezza variabile (variando il numero dei brani). Il tempo impiegato dall'applicazione ad eseguire tutte le operazioni necessarie varia in minima parte, anche triplicando il numero dei brani. Aumentando il numero dei brani aumentano proporzionalmente anche le operazioni di ricerca delle corrispondenze con i concetti dell'ontologia. Tuttavia questo particolare fattore sembra avere scarsa influenza sul tempo impiegato per elaborare la pagina finale, perciò il fattore che causa tempi così alti anche nel caso di pagine di piccole dimensioni è da ricercarsi altrove. Effettuando alcuni test senza generare la pagina VoiceXML finale si può notare che l'applicazione svolge tutte le operazioni in tempi relativamente brevi. Dunque il passaggio che causa una lunga elaborazione è il passo finale, ovvero la generazione della pagina VoiceXML da parte del marshaller. Infatti tale pagina è piuttosto lunga e contiene una grande quantità di menù e operazioni VoiceXML. Questa estrema complessità pesa molto sul marshaller che deve poi costruire la pagina finale partendo da un albero Java con un alto livello di profondità.

Per quanto riguarda la ricerca semantica all'interno della pagina si nota che i tempi di attesa sono relativamente brevi. Questo perché l'applicazione deve elaborare i contenuti della pagina in cui si effettua la ricerca solamente con un numero molto ristretto di concetti (i termini della ricerca). Per questo motivo il numero di nodi VoiceXML di cui sarà formata la pagina finale sarà estremamente ridotto rispetto al caso della navigazione con lettura semantica.

6.2.2. Velocità ed efficacia di navigazione dei contenuti

Come discusso in precedenza uno dei problemi legati alla lettura sequenziale di una pagina consiste nel fatto di doverla ascoltare interamente per individuare i contenuti che interessano. Questo causa, tra le altre cose, una notevole perdita di tempo, in quanto l'utente deve ascoltare anche i contenuti che non sono di suo interesse. Un aspetto che è importante valutare è quindi il tempo necessario all'utente per raggiungere i contenuti di interesse per mezzo della navigazione semantica. Rispetto alla navigazione sequenziale, tramite la quale l'utente accede immediatamente ai contenuti della pagina, è necessario che l'utente navighi gli argomenti che sono stati individuati nella pagina ed effettui una scelta prima di poter ascoltare i contenuti. Questa caratteristica della navigazione semantica richiede perciò all'utente un tempo maggiore dedicato alla navigazione della pagina che nella navigazione sequenziale non è presente. Tuttavia il tempo richiesto all'utente per navigare gli argomenti è ridotto, poiché si tratta solamente di effettuare due scelte e, dato che gli argomenti sono organizzati in modo gerarchico, si riduce anche il tempo necessario per ascoltare quali sono gli argomenti prima di poter effettuare la propria scelta. Inoltre il poco tempo perso per selezionare l'argomento interessato indirizza l'utente direttamente ai contenuti che lo interessano. In questo modo si guadagna tempo, perché l'utente individua subito e solamente i brani che vuole ascoltare, e si evita di dover ascoltare inutilmente anche i contenuti che non interessano. Selezionando un argomento infatti l'utente è sicuro che l'applicazione leggerà solo i brani utili.
Naturalmente la navigazione semantica non permette di ascoltare in una volta sola tutti i contenuti della pagina; per una navigazione del genere è più utile la semplice navigazione sequenziale, poiché l'uso della semantica non introduce in questo particolare caso vantaggi di sorta.

6.2.3. Comodità e semplicità d'uso

E' importante valutare come si comporta l'applicazione dal punto di vista dell'utilizzo da parte dell'utente, ovvero cercare di capire in che modo l'utente accede all'applicazione e la utilizza. Un aspetto estremamente rilevante della questione consiste nel considerare la semplicità di impiego dell'applicazione da parte dell'utente. Questo aspetto deve essere analizzato sia nel caso di utilizzo in condizioni normali che nel caso in cui l'utente inizia ad utilizzare l'applicazione per la prima volta. Quest'ultimo caso deve essere trattato per primo, in quanto è importante considerare se le modalità di utilizzo dell'applicazione sono facili da apprendere.

Per effettuare un'analisi in quest'ottica sono stati effettuati dei test facendo provare l'applicazione ad alcune persone che non ne conoscevano il funzionamento. E' significativo notare che l'applicazione è stata testata da persone con una discreta esperienza nel campo della navigazione web. Questo fatto è importante per evitare difficoltà di valutazione dei risultati, in quanto, nel caso di persone senza esperienza di navigazione web, non sarebbe stato possibile determinare in che misura questa inesperienza avrebbe influito sulle eventuali difficoltà di apprendimento.

Dai test effettuati è emerso, come era facile aspettarsi dato che l'applicazione propone modalità di navigazione leggermente differenti da quelle comuni, che gli utenti necessitano di alcune spiegazioni sul funzionamento della navigazione attraverso la lettura semantica della pagina. In pratica il metodo che deve essere utilizzato per tale modalità di navigazione (cioè la scelta dei gruppi di argomenti e successivamente degli argomenti stessi) risulta leggermente anomalo e non propriamente di immediata comprensione e deve pertanto essere spiegato più approfonditamente. Inoltre dalle prove è emerso che la ricerca semantica richiede un po' di pratica per poter essere effettuata efficacemente, perché comporta una certa difficoltà da parte dell'utente capire in che modo esprimere la richiesta. Tuttavia la modalità di navigazione tramite ricerca semantica e le problematiche ad essa legate vengono discusse ed analizzate nel prossimo paragrafo.

In definitiva si può dire che l'apprendimento dei metodi di utilizzo dell'applicazione da parte dell'utente non è immediato e comporta qualche difficoltà, soprattutto nelle prove iniziali.

Dopo che l'utente ha imparato come usare l'applicazione è possibile valutare la facilità d'uso in condizioni normali. Dalle prove effettuate è emerso che il normale utilizzo dell'applicazione (effettuare ricerche in una pagina, navigarla con la lettura semantica tramite la scelta degli argomenti, navigare i collegamenti ad altre pagine) non presenta particolari difficoltà. Una volta imparate le operazioni necessarie (che si riducono ad un numero ristretto), la navigazione del sito di prova è stata effettuata in maniera semplice e di facile attuazione. Gli unici problemi che sono sorti nelle prove sono stati causati dal sistema di riconoscimento vocale, che può richiedere di esprimere un comando più volte affinché esso sia riconosciuto, e dal sintetizzatore vocale che legge la pagina, che di solito legge un brano in modo poco naturale che spesso risulta di difficile ascolto.

In definitiva, in base alle prove effettuate, è possibile valutare che l'uso dell'applicazione risulta semplice una volta imparati i metodi di navigazione, ma comporta alcune difficoltà proprio per l'apprendimento di tali metodi.

6.2.4. Efficacia della ricerca semantica

La ricerca semantica è un aspetto dell'applicazione che presenta una certa complessità. Infatti l'applicazione dovrebbe permettere all'utente di esprimere la propria richiesta in qualsiasi modo ed eseguirla sempre correttamente. Tuttavia per permettere una tale caratteristica sorgono non poche difficoltà. L'implementazione di questa particolare tipologia di ricerca utilizzata nell'applicazione presenta alcune limitazioni e i risultati effettivamente ottenuti devono essere attentamente valutati. Inizialmente l'applicazione doveva riconoscere all'interno della richiesta dell'utente solamente i termini presenti in qualche forma all'interno dell'ontologia. In questo modo l'utente avrebbe potuto esprimere la richiesta in qualsiasi modo e con qualsiasi sintassi senza impedire il corretto riconoscimento degli effettivi termini della ricerca. Naturalmente questo tipo di implementazione si traduce nell'impossibilità di effettuare ricerche complesse. Infatti l'utente può solamente esprimere nella richiesta gli argomenti che vuole siano presenti nei risultati. Non è possibile cioè esprimere una richiesta del tipo: "Brani che contengono l'argomento A ma non l'argomento B", poiché l'applicazione in questo caso effettuerebbe la ricerca: "argomento A *e* argomento B". Tuttavia è stato ritenuto sufficiente, almeno nel caso di questo prototipo, ripiegare su un'implementazione del genere, senza cercare soluzioni più complesse, soprattutto perché tali soluzioni avrebbero reso più difficoltosa l'analisi dell'efficienza della richiesta, oltre che della sua efficacia.

Durante i test effettuati si è cercato di esprimere le ricerche in vari modi per poter valutare l'efficacia dell'implementazione utilizzata. Tuttavia è emerso un problema piuttosto rilevante. Poiché il numero dei concetti dell'ontologia è elevato, a causa della scarsa precisione del motore di riconoscimento vocale, l'applicazione tende a riconoscere termini dell'ontologia anche se non presenti nella richiesta dell'utente. In pratica qualsiasi cosa dica l'utente l'applicazione rileva la parola, all'interno dell'ontologia, la cui pronuncia si avvicina di più a ciò che l'utente ha detto. In pratica questa scarsa precisione del riconoscimento vocale invalida quasi totalmente l'utilizzo di una grammatica a riconoscimento parziale. Per ovviare a questo inconveniente senza dover indicare una regola precisa di espressione della richiesta è stato inserito un certo numero di parole, non presenti nell'ontologia, che ipoteticamente l'utente potrebbe usare nell'espressione della sua richiesta. Questo insieme di parole aggiuntive (all'interno della grammatica) è stato ampliato effettuando i test sulla ricerca. Tuttavia, poiché ogni persona può utilizzare modi diversi per esprimere una stessa richiesta, si è notato che, in teoria, è sempre possibile esprimere tale richiesta in modo che l'applicazione non riesca a riconoscerla in modo corretto. Purtroppo risulta difficoltoso valutare i risultati di questa metodologia poiché gran parte dell'efficacia della ricerca semantica

dipende dal motore di riconoscimento vocale. La richiesta deve essere infatti espressa in modo chiaro scandendo sufficientemente le parole.
Nonostante le difficoltà è stato comunque possibile valutare che, se la richiesta viene riconosciuta correttamente dal motore di riconoscimento vocale, essa può essere espressa in linguaggio quasi naturale e, come già visto, i risultati ritornati dall'applicazione sono corretti. La modalità di ricerca semantica avrebbe dunque una buona efficacia se l'applicazione disponesse di un buon sistema di riconoscimento vocale.

6.3. Conclusioni

Dall'analisi dei risultati è possibile trarre alcune conclusioni riguardo all'applicazione progettata e realizzata soprattutto in base agli obiettivi prefissati.
Si può dire che l'applicazione sviluppata propone effettivamente una nuova modalità di navigazione vocale per pagine web. Questa nuova modalità presenta alcune migliorie rispetto alle normali modalità di navigazioni web vocali.
In primo luogo, come già discusso, permette di navigare una pagina web con minori sprechi in termini di tempo. Infatti, dato che l'utente naviga ascolta solamente i contenuti che lo interessano, si evita di dover ascoltare l'intera pagina, obbligati a sentirsi leggere anche la parti di testo per le quali non si ha nessun interesse.
Questo particolare aspetto risolve, almeno in parte, uno dei problemi più rilevanti delle modalità tradizionali di navigazione, ovvero il fatto che tali modalità richiedono all'utente un altissimo livello di attenzione. Infatti per poter navigare una pagina web fruttuosamente con le metodologie tradizionali l'utente deve riuscire, durante l'ascolto dell'intera pagina, a mantenere una certa attenzione in modo da riuscire a seguire la lettura dei contenuti. Un tale livello di attenzione è possibile solamente per pagine molto brevi. Solitamente l'utente non riesce a seguire tutta la lettura e finisce inevitabilmente col deconcentrarsi e perdersi alcune informazioni presenti nella pagina. Tramite la modalità di navigazione con lettura semantica, invece, l'utente deve ascoltare in successione solo una piccola parte dei contenuti, letti inoltre separatamente. Oltretutto gli vengono letti solo i contenuti che egli stesso ha scelto. In questo modo il livello di attenzione richiesto è più facilmente mantenibile da parte dell'utente. Si può dire quindi che l'attenzione dell'utente è

piuttosto focalizzata rispetto alla navigazione tradizionale, permettendo quindi una navigazione più efficace.

Un altro aspetto positivo della modalità di navigazione progettata è il maggior coinvolgimento dell'utente nella navigazione. Infatti nella navigazione vocale tradizionale l'utente si trova di fronte ad un percorso navigazionale prestabilito. Si trova cioè a dover ascoltare la pagina secondo un percorso che non ha la possibilità di cambiare. L'interazione dell'utente è ridotta al minimo e consiste per lo più nella possibilità di saltare la lettura di alcuni passi troppo lunghi (a priori, senza poter sapere nulla sui contenuti). Con la modalità di navigazione progettata, invece, è l'utente a definire il percorso di navigazione, scegliendo tra gli argomenti presenti nella pagina. La navigazione della pagina, quindi, è gestita in prima persona dall'utente, che non si trova più a dover solamente ascoltare in modo passivo la lettura della pagina. Questa nuova modalità si avvicina di più rispetto alle precedenti al consueto concetto di navigazione, che considera l'utente parte attiva.

Finora sono state discusse le migliorie apportate dall'applicazione realizzata rispetto alle modalità di navigazione vocale precedenti. Tuttavia esistono anche altri aspetti positivi legati più strettamente all'aspetto semantico. Infatti, utilizzando un approccio di tal genere, l'applicazione svolge alcune operazioni che nella navigazione tradizionale (non solamente vocale) sono effettuate mentalmente dall'utente. Questa caratteristica è visibile in due aspetti differenti nelle due modalità di navigazione progettate.

Nella navigazione con lettura semantica l'applicazione, tramite l'analisi semantica della pagina, riesce a descriverne i contenuti all'utente, mentre nella navigazione tradizionale è l'utente che deve individuare gli argomenti leggendo i vari brani.

Nella modalità di ricerca semantica invece l'applicazione effettua all'interno della pagina una ricerca di particolari contenuti che sono di interesse per l'utente. Nella navigazione tradizionale è invece l'utente che deve individuare tra tutti i contenuti della pagina quelli che lo interessano.

Queste particolarità facilitano la navigazione all'utente che può così evitare di dover dedurre gli argomenti trattati all'interno di una pagina leggendola in modo più o meno rapido e lasciare, invece, che tali operazioni vengano svolte in automatico dall'applicazione.

6.4. Sviluppi futuri

Alcune considerazioni vanno espresse sul sistema di supporto vocale. Infatti l'applicazione sarebbe più funzionale se il riconoscimento e la sintesi vocale fossero migliori. Innanzitutto un riconoscimento vocale più efficiente migliorerebbe la comodità di utilizzo dell'applicazione, evitando all'utente di dover spesso ripetere i comandi. Inoltre le ricerche semantiche risulterebbero di conseguenza più efficaci e sarebbe possibile sviluppare metodologie per permettere di effettuare ricerche più complesse. Per quanto riguarda invece la sintesi vocale, un sintetizzatore che producesse un tono di voce più vicino a quello umano aumenterebbe notevolmente l'usabilità dell'applicazione in termini di:

- facilità di comprensione da parte dell'utente dei brani ascoltati
- interazione con l'utente più amichevole

Dal punto di vista degli sviluppi futuri ci sono varie considerazioni che possono essere fatte, sia riguardo l'applicazione sviluppata che l'approccio utilizzato per la metodologia di navigazione semantica.
Innanzitutto il prototipo di applicazione realizzato può essere sviluppato in maniera da includere nella navigazione semantica tutta l'espressività e complessità del linguaggio OWL. Si possono aprire molte strade per la navigazione semantica utilizzando anche tutti gli attributi e le relazioni presenti in tale linguaggio. La grande espressività dell'OWL Full potrebbe inoltre permettere una navigazione sempre più potente e razionale, a patto di risolvere il problema della computabilità.
Avendo a disposizione un linguaggio più espressivo, si potrebbe pensare ad altri metodi di navigazione semantica, ad esempio una navigazione che permetta di spostarsi all'interno dei contenuti, senza dover tornare all'inizio per scegliere un argomento. Inoltre sarebbe possibile effettuare un'analisi più complessa dei contenuti della pagina, riuscendo ad esempio ad individuare quali sono gli argomenti principali trattati e quali, invece, gli argomenti trattati solo brevemente o superficialmente.
L'approccio utilizzato per permettere la navigazione semantica può essere utilizzato anche al di fuori dell'applicazione realizzata ed in altri ambiti applicativi. Si può pensare ad esempio ad un sito progettato e realizzato appositamente per la navigazione semantica. In tal caso il sito sarebbe costituito da un'ontologia (che ne descrive i contenuti) e da un insieme di contenuti (raccolti, ad

esempio, in una base dati). Le pagine del sito, vocali o visive, potrebbero essere generate dinamicamente a partire dai contenuti tramite la loro analisi semantica. In un sito del genere si potrebbero navigare addirittura tutti i contenuti in modo globale, si eviterebbero i problemi legati alla conversione tra linguaggi (HTML e VoiceXML) e si oltrepasserebbero i cammini navigazionali prestabiliti dalla struttura delle pagine, necessariamente presenti nel caso di una navigazione semantica applicata ad un sito progettato secondo criteri tradizionali.

Per quanto riguarda, infine, la navigazione web su più siti per mezzo dell'applicazione realizzata, sarebbe molto utile poter utilizzare librerie di ontologie OWL attraverso le quali poter analizzare siti di argomenti differenti tra loro. In questo modo si potrebbe riferire ogni sito all'ontologia o alle ontologie che descrivono i suoi contenuti e permetterne così la navigazione semantica.

7. BIBLIOGRAFIA

ONTOLOGIE, OWL, RDF E WEB SEMANTICO

1. Sean Bechhoferl, Ian Horrocks1, Peter F. Patel-Schneider (2003), *Tutorial on OWL*, ISWC, Sanibel Island, Florida, USA .

2. Matthew Horridge, Holger Knublauch, Alan Rector, Robert Stevens, Chris Wroe, *A Practical Guide To Building OWL Ontologies Using The Protege-OWL Plugin and CO-ODE Tools Edition 1.0.*

3. Stefano Campanini (2004), *Introduzione a OWL,*Webbit 2004, Padova.

4. Protege (2000), *The Protege Project*, http://protege.stanford.edu.

5. Electronics and Telecommunications Research Institute, Tutorial di ez*OWL - plugin di Protege per la gestione grafica di ontologie OWL*, http://iweb.etri.re.kr/ezowl/index.html

6. Helena Lindström, Rob Howard, *Introduction to OWL, the Web Ontology Language.*

7. Natalya F. Noy and Deborah L. McGuinness (2002), *Ontology Development 101: A Guide to Creating Your First Ontology,* Stanford University, Stanford.

8. Janet Daly (2003), *Il World Wide Web Consortium Pubblica le Candidate Recommendation del Web Ontology Language*, http://www.w3.org/2003/08/owl-pressrelease.html.it.

9. Smith, Welty, McGuinness (2004), *OWL Web Ontology Language Guide W3C Recommendation 10 Feb 2004*, http://www.w3.org/TR/owl-guide.

10. Dean Schreiber, *OWL Web Ontology Language Reference W3C Recommendation 10 Feb 2004*, http://www.w3.org/TR/owl-ref

11. Oreste Signore (2002), RDF *per la rappresentazione della conoscenza*, Ufficio Italiano W3C presso il C.N.R. - Istituto CNUCE, Pisa.

12. Navigli Roberto (2004), *Linguaggi per il Web Semantico*, Dipartimento di Informatica dell' Università La Sapienza – Roma

13. Sourceforge, *An introduction to RDF and Jena API*, http://jena.sourceforge.net/tutorial/RDF_API/index.html

14. Sourceforge, *A programmer's introduction to RDQL*, http://jena.sourceforge.net/tutorial/RDQL/index.html

15. Hewlett-Packard Development Company, *RDQL – RDF Data Query Language*, http://www.hpl.hp.com/semweb/rdql.htm

16. Berners-Lee T., Hendler J., Lassila O. (2001), *The Semantic Web*, Scientific American, http://www.scientificamerican.com/2001/0501issue/0501bernerslee.html

17. Mauro Iannucci, Massimo Imperiali, *Semantic Web*, http://online.infomedia.it/riviste/dev/90/articolo23/articolo.htm

18. Paolo Ceravolo (2003), *L'architettura del Semantic Web*, http://pro.html.it

19. Paolo Ceravolo (2003), *Cos'è e a cosa serve il Web Semantico*, http://pro.html.it

VUI e VoiceXML

20. W3C Note (2000), *Voice eXtensible Markup Language (VoiceXML) version 1.0*, http://www.w3.org/TR/2000/NOTE-voicexml-20000505

21. BeVocal Inc. (2004), *VoiceXML Tutorial*, Mountain View, CA 94043.

22. BeVocal Inc. (2004), *Grammar reference*, Mountain View, CA 94043

23. VoiceXML Italian User Group (2004), *VoiceXml Tutorial*, http://www.vxmlitalia.com/tutorial.html

24. VOXEO, *VoiceXML Development Guide Version 2.1*, http://www.vxml.org/tutorialhome.htm

25. W3C Recommendation (2004), *Speech Recognition Grammar Specification Version 1.0*, http://www.w3.org/TR/2004/REC-speech-grammar-20040316

26. James A. Larson et al (2005), *Ten Guidelines for Designing a Successful Voice User Interface*, SpeechTEK 2004, http://www.speechtechmag.com

27. International Engineering Consortium (2002), *The benefits of a conversational Voice User Interface in a voice portal*, http://www.iec.org

28. Heather Du (2002), *Vocal Information Retrieval: A System Prototype and User Interface Design Issues*, Department of Computer and Information Sciences, University of Strathclyde.

29. Loquendo, *VoiceXML Tutorial* , http://www.loquendocafe.com/vxml_intro.html

30. Loquendo, *Grammar Tutorial* , http://www.loquendocafe.com/indexgrammar.html

31. Loquendo, *VUI Tutorial* , http://www.loquendocafe.com/indexvui.html

32. Loquendo, *VoiceXML Reference* , http://www.loquendocafe.com/vxml_reference1.html

33. C. J. Kennedy (2004), *Voice Browsing: how two great ideas go great together*, Wireless Development Network, httt://www.wirelessdevnet.com.

VARIE

34. Thimoty Barbieri, Antonio Bianchi, Licia Sbattella (2004), *Multimodal communications for vision and hearing impairments*, Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milano.

35. Rossi Davide (2004), *Prototipo di navigatore vocale di pagine Internet*, Progetto d'esame di Informatica Grafica 2004, prof. Thimoty Barbieri, Politecnico di Milano – Polo di Como.

36. Conservatorio di Musica G.Verdi di Como (2003-2004), *Archivio articoli della rassegna stampa*, http://www.conservatoriocomo.it/archiv/index.html.

37. W3C, *WAI - Web Accessibily Initiative*, http://www.w3.org/WAI

8. **RINGRAZIAMENTI**

Si ringraziano con riconoscenza:

il prof. Thimoty Barbieri, per la disponibilità e l'aiuto nella preparazione di questo lavoro,

Davide Rossi, per aver permesso l'uso del progetto da lui realizzato,

il Conservatorio G. Verdi di Como, per aver reso disponibili gli articoli dell'archivio dell'istituto.