



DIPARTIMENTO DI INFORMATICA  
CORSO DI LAUREA TRIENNALE IN INFORMATICA

---

TESI DI LAUREA IN  
“METODI PER IL RITROVAMENTO DELL’INFORMAZIONE”

SVILUPPO DI UN'APPLICAZIONE WEB PER UN SISTEMA DI  
SUPPORTO ALLE DECISIONI PER L'E-PROCUREMENT

RELATORE  
Prof. Pasquale LOPS

LAUREANDO  
Christian GIACOVAZZO

---

ANNO ACCADEMICO 2021/2022



# Indice

<b>Introduzione</b>	<b>5</b>
<b>1. Sistemi basati su dati testuali</b>	<b>8</b>
1.1 Elaborazione del linguaggio naturale	8
1.1.1 Sentence Detection	9
1.1.2 Tokenization	9
1.1.3 Stop Words Removal	10
1.1.4 POS Tagging	10
1.1.5 Lemmatization & Stemming	10
1.1.6 Keyphrase Detection	11
1.1.7 Chunking	11
1.2 Motori di ricerca classici	12
1.2.1 Indice invertito	13
1.2.2 Modello Bag of Words	14
1.2.3 Il Modello Booleano	15
1.2.4 Term Frequency e Inverse Document Frequency	16
1.2.5 Il Vector Space Model	18
1.3 Modelli Semantici Distribuzionali	20
1.3.1 Random Indexing	21
1.3.2 Motori di ricerca semantici	22
<b>2. Progetto SIAP-PAI</b>	<b>24</b>
2.1 Architettura ad alto livello	24
2.1.1 Data Collector	25
2.1.2 Pre Processing	25
2.1.3 Tender Analyzer	25
2.1.4 Service Tools	26
2.2 Open Data	26
2.2.1 Open Data strutturati da ANAC	27
2.3 Bando di gara	28
2.4 Struttura database relazionale	28
2.5 Dati non strutturati	29
<b>3. Semantic Framework e servizi backend</b>	<b>30</b>
3.1 Semantic Framework	30
3.1.1 Motore di Ricerca	33
3.2 Servizi RESTful	34
3.3 Servizi RESTful impiegati	35
3.3.1 NLP Pipeline	36
3.3.2 Estrazione delle phrases	36
3.3.3 Analyze	38
3.3.4 Search Engine e Random Indexing	38
3.3.5 Related Concepts	40
3.3.6 Bandi e documenti simili	40
3.4 Metadati Strutturati	40
3.4.1 Gestione connessione con la base di dati e API REST	41

3.4.2	API REST per l'interrogazione del database	42
<b>4.</b>	<b>Strumenti utilizzati per l'interfaccia</b>	<b>44</b>
4.1	AngularJS	44
4.2	JavaScript	44
4.3	Scripting	45
4.3.1	Ricerca e risultati	45
4.3.2	Ricerca con Related Concepts	48
4.3.3	Ricerca bandi simili	49
4.3.4	Visualizzazione dei dettagli bando	51
<b>5.</b>	<b>Conclusioni</b>	<b>53</b>
	<b>Riferimenti bibliografici e sitografici</b>	<b>56</b>

## Introduzione

I sistemi di supporto alle decisioni sono sistemi informatici che hanno lo scopo di aumentare l'efficacia e l'efficienza dei processi decisionali in qualunque tipo di organizzazione. Il valore aggiunto che questi sistemi apportano al processo decisionale sta nell'offrire agli stakeholder la possibilità di considerare informazioni e conoscenze rilevanti per aree di interesse che non sarebbero disponibili senza di esse, in modo da poter prendere decisioni e organizzare processi in contesti variabili. Data la grande quantità di informazioni e conoscenze prodotte nelle organizzazioni, i processi di ritrovamento e di elaborazione dell'informazione sono svolti in modo automatico; inoltre, tali informazioni sono conservate prevalentemente in formato digitale.

Un sistema di supporto alle decisioni di questo tipo è utile nel contesto dell'e-procurement, ovvero il processo grazie al quale Aziende Private e Pubbliche Amministrazioni acquisiscono beni e servizi da fornitori attraverso internet.

La maggior parte dei sistemi di supporto alle decisioni impiegano risorse informatiche per elaborare informazioni strutturate, ossia dati che descrivono il dominio di interesse solitamente conservati in grandi basi di dati. Tramite tecniche di *data mining*, si estrarre conoscenza precedentemente ignota a partire dai dati, applicando metodi algoritmici e statistici.

Una tipologia alternativa di sistemi di supporto alle decisioni è quella basata sull'estrazione di informazioni e conoscenza a partire dai dati non strutturati, ossia dai comuni testi scritti in linguaggio naturale. I sistemi più diffusi in questa categoria sono i motori di ricerca, che hanno l'obiettivo di reperire e mostrare all'utente i documenti di una collezione che sono

rilevanti rispetto ad un particolare bisogno informativo. Il criterio con il quale questo genere di sistemi valuta la rilevanza tra un documento e un bisogno informativo si basa solitamente sulla valutazione dell'occorrenza nel documento di specifiche parole con cui esso è espresso. È anche possibile applicare ai documenti tecniche di elaborazione più complesse che tengono in considerazione il significato semantico dei termini. Utilizzando queste tecniche è possibile sia creare motori di ricerca più sofisticati, sia progettare e sviluppare servizi basati sulla rappresentazione semantica del testo. Questo genere di servizi dà all'utente la possibilità di esaminare il contenuto dei documenti e fare confronti su di essi sfruttando il processo di analisi automatica. Ad esempio, è possibile individuare i documenti in una collezione che trattano di un tema, individuare i concetti simili tra loro sulla base di come vengono usati all'interno di una collezione, o trovare i documenti di una collezione che parlano di temi simili.

Il presente lavoro di tesi è incentrato sullo studio del funzionamento di un sistema di supporto alle decisioni basato su dati non strutturati, sviluppato nell'ambito del progetto SIAP-PAI sui bandi di gara pubblici in collaborazione con Regione Puglia e InnovaPuglia, sfruttando i servizi offerti da un sistema di gestione documentale e analisi semantica del testo chiamato Semantic Framework. In particolare, è obiettivo della tesi indagare le possibilità di miglioramento dei servizi offerti dal sistema di supporto alle decisioni mediante l'integrazione di informazioni strutturate riguardanti il dominio di interesse.

Il lavoro di tesi è così strutturato: il primo capitolo ha l'obiettivo di illustrare i principi teorici e le tecniche sulla base delle quali è stato sviluppato il Semantic Framework. Nel secondo capitolo viene presentato il progetto SIAP-PAI per il recupero e analisi dei bandi pubblici. Nel terzo capitolo viene presentata struttura e funzionamento del Semantic Framework, e le

scelte progettuali compiute per integrare nel Semantic Framework un modulo di gestione dei dati strutturati relativi al dominio cui appartengono i documenti gestiti e analizzati nella piattaforma. In conformità con l'architettura del sistema, i servizi aggiuntivi sono stati sviluppati secondo il protocollo REST, del quale sono illustrati i principi di funzionamento. Il quarto capitolo illustra gli strumenti e script utilizzati per la visualizzazione delle pagine dell'interfaccia web.

## 1. Sistemi basati su dati testuali

La maggior parte delle informazioni attualmente disponibili in formato digitale, siano esse di pubblico dominio o appartenenti a organizzazioni, sono organizzate in forma non strutturata, ossia sottoforma di testo. Dal punto di vista della rappresentazione della conoscenza, il testo è un formato totalmente diverso rispetto a quello strutturato con cui sono memorizzate le informazioni in una base di dati. Il testo è, infatti, una forma di rappresentazione della conoscenza prodotta da e per le persone, di conseguenza esso va processato in modo adeguato affinché il suo contenuto informativo possa essere acquisito, processato e reso disponibile automaticamente, e non soltanto tramite la lettura. Esistono due livelli di rappresentazione digitale del testo: il primo, più superficiale, è basato solo sulla *forma* del testo; il secondo, più profondo e completo, tiene conto del *significato* delle parole (ad esempio, le parole “fazzoletto” e “tovagliolo” hanno forme diverse ma si riferiscono a concetti simili).

Operando automaticamente sui testi diventa possibile gestire grandi collezioni di documenti, contenenti anche decine di milioni di parole. Tramite i motori di ricerca è possibile quindi reperire documenti utili cercando al loro interno parole chiave significative dal punto di vista della forma (primo livello) o del significato (secondo livello).

### 1.1 Elaborazione del linguaggio naturale

I passi compiuti per preparare il testo ad essere rappresentato in forme appropriate per le operazioni di ritrovamento dell'informazione fanno parte di un processo detto di “elaborazione del linguaggio naturale” o Pipeline di Natural Language Processing (NLP). Esso comprende un



insieme di passi che hanno lo scopo di aumentare l'efficacia delle tecniche di rappresentazione e analisi del testo, eliminando le ridondanze tipiche di questo tipo di rappresentazione della conoscenza e mettendone in risalto la ricchezza informativa. Le operazioni eseguite in questo processo sono alla base sia della rappresentazione "superficiale" del testo, sia di quella "profonda" che tiene conto del significato dei termini.

La pipeline NLP è stata usata all'interno del Semantic Framework illustrato nei prossimi capitoli.

### 1.1.1 Sentence Detection

Il primo passo nel processo di elaborazione del testo è l'individuazione delle frasi. Esse vengono individuate all'interno di un testo considerando che sono solitamente separate dal punto fermo. Tuttavia, è necessario distinguere questo da altri usi del punto che possono essere presenti nei testi considerati (ad esempio, per abbreviare termini, per separare porzioni di un numero, eccetera).

### 1.1.2 Tokenization

Il secondo passo nella pipeline di NLP è la suddivisione delle frasi in parole o "token", eliminando i segni di punteggiatura e i caratteri speciali. Anche in questo caso, l'euristica base con la quale si identificano i token è che essi sono generalmente separati da spazi; bisogna tuttavia tener conto di token in forme particolari, come le date e l'ora, i nomi propri di persona, ecc., che sono composti da più parti che non vanno considerate separatamente.

### 1.1.3 Stop Words Removal

Le *stop words* sono parole molto comuni in una lingua o in un particolare ambito, al punto tale che il loro contenuto informativo è considerato molto basso o nullo; per questa ragione un testo si può rappresentare efficacemente senza comprenderle risparmiando spazio di rappresentazione e tempo di computazione. Per le lingue più comuni esistono liste di *stop words* che vengono usate per filtrare le parole rilevanti nei documenti.

### 1.1.4 POS Tagging

Il Part Of Speech Tagging è l'operazione che associa a ciascun token la sua categoria grammaticale all'interno della frase. Questa operazione non è banale, poiché molto spesso uno stesso termine può avere più di un significato, o perché alcuni elementi sintattici nella frase sono sottintesi nell'uso comune della lingua. La disambiguazione tra i diversi ruoli sintattici e l'individuazione di quello corretto avviene considerando sia le definizioni del termine considerato, che, soprattutto, le parole assieme alle quali occorre e il loro ruolo all'interno della frase.

L'operazione di POS Tagging è particolarmente utile per la costruzione di una rappresentazione della sintassi del testo e per l'individuazione della corretta semantica associata ai termini, nel caso in cui questi abbiano un diverso significato a seconda del loro ruolo sintattico all'interno della frase.

### 1.1.5 Lemmatization & Stemming

Nei documenti sono spesso presenti diverse forme di una parola che hanno dei significati simili (democrazia, democratico, democratizzazione). La lemmatizzazione è un'operazione di semplificazione dei token che

trasforma ciascuno di essi nella sua forma grammaticale di base (ossia il suo lemma) trasformando il suffisso proprio della specifica inflessione presente nel testo in quello della forma base dell'elemento grammaticale. Il processo di trasformazione non è banale, in quanto va identificata correttamente la forma base di ogni token; in particolare, ciò è più difficile nel caso di termini che hanno la stessa forma ma funzioni grammaticali e significati differenti (es. un'ancora, egli àncora, ancòra qui). In questi casi gli algoritmi di lemmatizzazione possono ricondurre il termine al lemma corretto grazie al contesto in cui occorre e al risultato del processo di POS tagging.

Lo stemming è un'operazione di semplificazione dei token più radicale: essa tronca il suffisso di ogni token riducendo il termine alla sua radice. Uno degli algoritmi più usati per compiere questa operazione è l'algoritmo di Porter [1].

### 1.1.6 Keyphrase Detection

Questo passo di analisi del testo consiste nell'identificare i concetti più rilevanti all'interno di un testo, che ne descrivono dunque l'argomento e il contenuto informativo; essi possono essere espressi anche con più di una parola (ad es. "information retrieval"). Un approccio semplice, che impiega una tecnica di apprendimento non supervisionato, si basa sull'identificazione delle coppie, triple o n-uple di termini che co-occorrono con una frequenza significativa [2].

### 1.1.7 Chunking

Il Chunking è un'operazione di analisi sintattica del testo che può essere vista come un'operazione di tokenizzazione più elaborata, o come un parsing più semplice (è, infatti, anche detta "shallow parsing"). Essa

consiste nell'individuare gli elementi grammaticali che sono composti da più termini, come i nomi propri di luoghi o di persone o le forme verbali che comprendono verbi ausiliari o modali. Ciò è utile perché è più appropriato considerare queste forme grammaticali in modo unitario per avere contezza del loro significato; ad esempio, dire che in un testo si parla di Sud Africa è diverso dal dire che si parla di Sud e di Africa.

## 1.2 Motori di ricerca classici

Uno degli strumenti più comuni e più utili che si può costruire su una collezione digitale di documenti, processati attraverso le operazioni appena descritte, è il motore di ricerca. Esso è un sistema che, dato un insieme di parole – detto “query” – che esprime un bisogno informativo dell'utente, ha l'obiettivo di restituire ad esso i documenti della collezione che sono più rilevanti rispetto ai termini della query. Questo task cambia significativamente in relazione al tipo di rappresentazione del testo adottata. Nel caso in cui i documenti siano rappresentati tenendo conto della semantica dei termini al loro interno, infatti, i documenti più rilevanti per una query saranno quelli che contengono parole dal significato più simile a quelle date in input. Nel caso più semplice in cui i documenti siano rappresentati senza tener conto del significato dei termini, il calcolo della similarità tra documenti e query si basa sull'occorrenza delle parole della query nei documenti, ed eventualmente sulla frequenza con cui queste occorrono.

In questo paragrafo sono considerati i motori di ricerca “classici”, ossia basati solo sulla forma dei termini e non sul loro significato. Esistono diversi approcci per creare un motore di ricerca classico; essi differiscono principalmente per il modo in cui sono rappresentati i documenti

all'interno del sistema e per il modo in cui viene calcolata la similarità tra essi e le query dell'utente. Sono presentate di seguito le strutture dati e le assunzioni sulla rappresentazione dei documenti comuni a diversi tipi di motori di ricerca; viene poi brevemente presentato il modello Booleano e il più diffuso Modello a Spazio Vettoriale.

### 1.2.1 Indice invertito

L'operazione più basilare ed intuitiva che un motore di ricerca classico deve compiere per risolvere una query è individuare quali sono i documenti della collezione che contengono le parole presenti in essa. L'insieme delle parole presenti in tutti i documenti di una collezione, associati alla loro frequenza assoluta, è detto "vocabolario" della collezione. Per compiere l'operazione di ritrovamento dei documenti in modo efficiente i motori di ricerca operano su una struttura dati chiamata "indice invertito". Il suo nome deriva dal confronto con il normale indice di un libro, che mette in relazione i capitoli o i paragrafi di un testo con le pagine in cui si trovano, e quindi con le parole contenute in essi. L'indice invertito è, invece, una struttura dati che associa ad ogni parola del vocabolario i documenti nei quali essa compare, consentendo così di individuare i sottoinsiemi di documenti in cui sono presenti tutte o molte delle parole in una query.

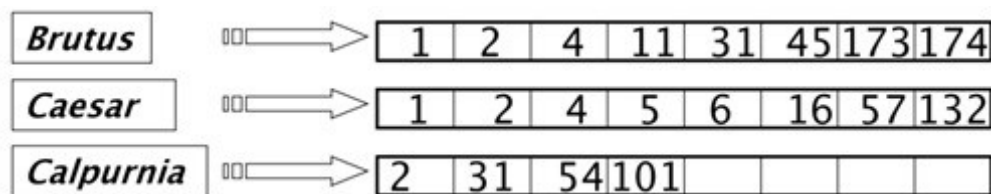


Figura 1.1 Un esempio di indice invertito, nel quale i numeri identificano i documenti della collezione

## 1.2.2 Modello Bag of Words

Il modello Bag of Words è un metodo di rappresentazione di un testo che tiene conto del numero di occorrenze di ogni parola nel testo ma non delle posizioni che ciascuna occupa in esso. Ciò implica che due o più testi che contengono le stesse parole ma in diverso ordine, e che hanno quindi complessivamente significati diversi, saranno rappresentati nello stesso modo. Questo modello si basa sull'assunzione forte che la posizione di una parola nelle frasi del testo in cui compare non sia rilevante per stabilirne il significato e, dunque, per interpretare il contenuto informativo del testo. Sebbene questa assunzione sia errata, e si punti, infatti, a sopperire alle sue mancanze tramite le operazioni di analisi sintattica del testo, come il Chunking e il POS Tagging, il motivo per cui si adotta è che la posizione delle parole in un testo non risulta empiricamente rilevante per il ritrovamento dei documenti attinenti alla richiesta espressa con una query.

Document at index 0		
Document index	(0, 46)	1
	(0, 48)	1
	(0, 81)	1
	(0, 28)	1
	(0, 15)	1
	(0, 27)	1
	(0, 17)	1
	(0, 23)	2
	(0, 59)	1
	(0, 53)	1
	(0, 56)	1
	(0, 6)	1
	Term count	

Figura 1.2 Rappresentazione secondo il modello Bag of Words di un documento identificato dall'indice 0

### 1.2.3 Il Modello Booleano

Adottando una rappresentazione conforme al modello Bag of Words, un documento sarà dunque rappresentato come un multiinsieme di parole, ossia un insieme nel quale gli elementi possono apparire più volte. Il modello booleano [3, 4] implementa nel modo più semplice possibile questa rappresentazione, considerando, cioè, la semplice presenza o assenza di un termine in un documento senza considerarne la frequenza d'uso. Un documento è rappresentato, quindi, come un vettore di variabili booleane, nel quale ogni elemento rappresenta un termine del vocabolario e assume il valore *vero* se la parola appare nel documento, e il valore *falso* altrimenti. Una query per un motore di ricerca costruito secondo questo modello può essere espressa collegando i termini con operatori booleani (es. *termine1 AND termine2 OR termine3*) e la risposta ad essa corrisponde al sottoinsieme di documenti della collezione la cui rappresentazione vettoriale soddisfa le condizioni espresse nella formula logica della query. Ogni documento sarà dunque semplicemente rilevante o non rilevante per una query e non vi sarà associata una misura di rilevanza; per questo motivo, i risultati di una ricerca non vengono restituiti secondo un ordinamento significativo.

Pur essendo un modello che consente di esprimere interrogazioni con precisione, la forma logica delle query costituisce sia un ostacolo sintattico per l'utente, che deve saper esprimere il proprio bisogno informativo in questa forma, sia un vincolo spesso troppo – o troppo poco – stringente nel processo di individuazione dei risultati rilevanti, in quanto i documenti che corrispondono precisamente alla query possono essere molto pochi, quando la query è molto specifica, o troppi, se comprende molti casi ed è espressa tramite un insieme condizioni disgiunte.

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Figura 1.3 Rappresentazione di un insieme di documenti (sulle colonne) secondo il modello Booleano (termini sulle righe)

### 1.2.4 Term Frequency e Inverse Document Frequency

Per arricchire la rappresentazione di un documento utilizzata nel modello booleano si può tenere conto di due parametri relativi ad ogni termine del vocabolario: la frequenza con cui ogni termine appare in ciascun documento e la rarità di un termine rispetto all'intera collezione. L'idea alla base dell'introduzione di queste misure è che un documento sarà più rilevante rispetto ad una query se uno o più termini della query sono molto frequenti nel documento, oppure se uno o più termini che si trovano nella query sono presenti solo in quel documento o in pochi altri. La frequenza di una parola in un documento è detta Term Frequency [5]; matematicamente essa è definita come la frequenza assoluta di occorrenza del termine nel documento. Generalmente, questa misura non viene direttamente impiegata per calcolare la rilevanza di un documento rispetto a una query, poiché porterebbe a far aumentare il peso di un termine in un documento in modo sproporzionato, facendo risultare il documento di una rilevanza rispetto alla query maggiore di quella reale (un documento in cui un termine appare 10 volte non dovrebbe essere 10 volte più significativo di uno in cui esso appare una volta sola). Per questa ragione, si impiega la funzione logaritmo per ridurre l'impatto della crescita del Term Frequency sulla



funzione di scoring; l'operazione applicata è detta *sublinear tf-scaling* ed è definita matematicamente come segue:

$$sub\_tfs(t, d) = \begin{cases} 1 + \log_{10} tf_{t,d}, & tf_{t,d} > 0 \\ 0, & tf_{t,d} \leq 0 \end{cases}$$

dove  $t$  è un generico termine e  $d$  è un generico documento considerato. La rarità di un termine in una collezione è definita matematicamente sulla base del concetto di Document Frequency di un termine, ossia il numero di documenti in cui il termine appare. Questa misura è, ovviamente, inversamente proporzionale alla rarità di un termine; la rarità viene quindi misurata da una misura chiamata Inverse Document Frequency [6], che è

$$idf_t = \log\left(\frac{N}{df_t}\right),$$

definita matematicamente come:

dove  $N$  è il numero di documenti nella collezione e  $df$  è definito come il Document Frequency del termine  $t$ .

Ad ogni termine in un documento può essere associato, dunque, un peso che è direttamente proporzionale sia alla frequenza del termine nel documento stesso, sia alla rarità del termine nella collezione; esso è il prodotto tra Term Frequency e Inverse Document Frequency del termine, ed è chiamato peso *tf-idf* del termine [7]:

$$w_{t,d} = sub\_tfs(w, d) \cdot idf_t$$

### 1.2.5 Il Vector Space Model

Il Modello a Spazio Vettoriale [8] è una tecnica di rappresentazione dei testi nella quale documenti e query sono rappresentati come vettori in uno stesso spazio multidimensionale, nel quale ogni dimensione rappresenta un termine nel vocabolario della collezione. I testi sono quindi rappresentati da entità matematiche di tipo diverso rispetto al modello booleano, e di queste entità vengono sfruttate le proprietà per il calcolo della similarità tra query e documenti. Il modello integra, inoltre, le misure presentate nel paragrafo precedente: infatti, l' $i$ -esimo elemento di un vettore che rappresenta un documento (o una query) ha come valore il peso associato al termine corrispondente rispetto a quel documento, calcolato in funzione della sua Term Frequency ed Inverse Document Frequency come descritto nel paragrafo precedente.

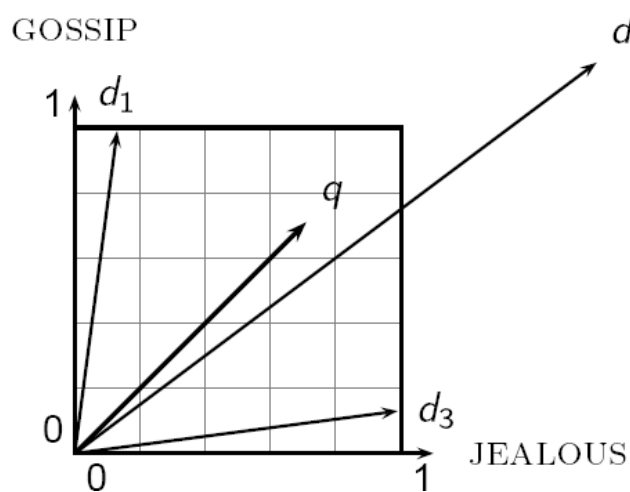


Figura 1.4 Rappresentazione dei documenti come vettori di lunghezza non normalizzata

Nella rappresentazione secondo il Modello a Spazio Vettoriale è possibile comparare documenti e query utilizzando operazioni vettoriali. Una prima possibilità per calcolare la similarità tra due documenti è considerare la loro distanza Euclidea nello spazio multidimensionale. Questa è definita come segue:

$$d(\vec{q}, \vec{p}) = \sqrt{\sum_{i=1}^n (q_i - p_i)^2},$$

dove  $q_i$  e  $p_i$  sono gli elementi in posizione  $i$ -esima dei generici vettori  $\vec{q}$  e  $\vec{p}$ . Usando questa metrica, però, due documenti, contenenti gli stessi termini ma di dimensioni diverse a causa della ripetizione di alcuni termini in uno di essi, saranno considerati diversi perché distanti nello spazio. È necessario, quindi, compiere una normalizzazione della dimensione dei vettori. Questa operazione è compresa nel calcolo della misura di similarità del coseno, che valuta la distanza tra documenti sulla base dell'angolo compreso tra i vettori che li rappresentano. La metrica è definita come segue:

$$\cos(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}| |\vec{d}|} = \frac{\vec{q}}{|\vec{q}|} \cdot \frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|\vec{v}|} q_i d_i}{\sum_{i=1}^{|\vec{v}|} q_i^2 \sum_{i=1}^{|\vec{v}|} d_i^2}.$$

La normalizzazione rispetto alla lunghezza consiste nel dividere il prodotto interno tra i vettori per le loro norme.

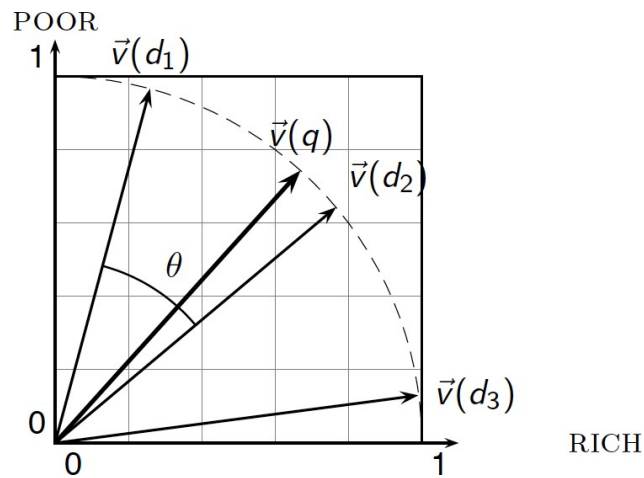


Figura 1.5 Rappresentazione della distanza tra vettori normalizzati basata sull'angolo compreso tra essi

Il Vector Space Model presenta comunque alcune problematiche:

- come in tutte le tecniche di rappresentazione basate sul modello Bag of Words, la posizione dei termini non viene presa in considerazione;
- documenti molto lunghi possono essere penalizzati nel processo di

ritrovamento poiché tenderanno ad avere score di similarità più bassi a causa dell'operazione di normalizzazione delle lunghezze;

- poiché viene usato il prodotto interno per confrontare i vettori, il modello può essere penalizzato dalla sparsità dei dati.

Inoltre, il modello a spazio vettoriale non può rilevare la similarità tra due documenti che abbiano un contenuto semanticamente simile ma contengano parole differenti, poiché la rappresentazione dei documenti è basata solo sulla forma delle parole che contengono e non sul loro significato.

### 1.3 Modelli Semantici Distribuzionali

Rappresentare il testo tenendo conto del significato delle parole ha un impatto significativo sulle operazioni di ritrovamento di documenti rilevanti rispetto a una query. Avere a disposizione una rappresentazione del significato delle parole usate in una collezione di documenti permette di calcolare la similarità tra una query e un documento non soltanto in base alle parole della query che occorrono nel documento, ma anche considerando le parole in esso che hanno un significato simile a quelle date in input.

Per creare una rappresentazione del significato delle parole si utilizzano spesso modelli matematici che le rappresentano come vettori in uno spazio multidimensionale, nel quale quelle che hanno un significato simile sono rappresentate tramite vettori “vicini” tra loro; lo spazio multidimensionale viene detto Word Space o Semantic Space. In questo spazio si può misurare la similarità tra concetti calcolando il coseno dell'angolo compreso tra essi. L'assunzione fondamentale su cui si basano tutti i modelli semantici distribuzionali è che i termini che occorrono frequentemente in contesti simili – e raramente in contesti diversi – hanno un significato simile [9].

Esistono diversi metodi per creare lo spazio vettoriale nel quale rappresentare il significato delle parole; di seguito è presentata la tecnica chiamata Random Indexing.

### 1.3.1 Random Indexing

Il Random Indexing [11] è una delle tecniche di creazione di un Word Space su una collezione di documenti. Essa ha il vantaggio di generare una rappresentazione del significato delle parole senza utilizzare fonti di conoscenza esterne (come dizionari o ontologie) e in modo incrementale, ossia riuscendo ad aggiornare la rappresentazione in modo semplice all'aggiunta di nuovi documenti nella collezione; l'unica operazione di preprocessing del testo necessaria per costruire lo spazio è la tokenizzazione.

La tecnica consiste in due passi principali: il primo consiste nell'assegnare a ogni termine del vocabolario di una collezione un vettore ternario e sparso generato casualmente, ossia un vettore i cui elementi possono assumere i valori -1, 0 e 1, nel quale la maggior parte degli elementi hanno valore 0 e nel quale le posizioni degli elementi non nulli sono scelte in modo casuale. Il secondo passo consiste nel generare un "vettore contesto" per ogni termine nel vocabolario sommando i vettori precedentemente associati alle parole che occorrono frequentemente insieme ad esso nella collezione, in base ad una soglia di vicinanza prefissata. Il vettore contesto di un termine è calcolato secondo la formula seguente:

$$\overrightarrow{cv_i} = \sum_{d \in C} \sum_j \vec{r}_{ij}, \quad -c < j < c, i \neq j,$$

dove  $C$  è la collezione di documenti,  $c$  è la soglia di vicinanza dei termini prefissata che delimita il contesto di ogni parola e  $\vec{r}_j$  è il vettore casuale

assegnato ad ogni parola presente nel contesto. Il vettore contesto di un termine è di fatto la rappresentazione del suo significato nello spazio multidimensionale, ed è un'implementazione algebrica del principio alla base dei modelli semantici distribuzionali enunciato in precedenza.

La creazione di un vettore contesto per ogni parola corrisponde formalmente ad un'operazione di riduzione della dimensionalità della matrice di co-occorrenza dei termini in una collezione. Questa operazione genera uno spazio di dimensionalità ridotta nel quale le distanze tra i vettori che rappresentano le parole restano proporzionali alle distanze che le parole avevano nello spazio originale [10].

### 1.3.2 Motori di ricerca semantici

Usando la tecnica di Random Indexing è possibile rappresentare termini e documenti nello stesso spazio vettoriale multidimensionale. Si può generare, infatti, una rappresentazione vettoriale di un documento tramite la somma pesata di tutti i vettori che rappresentano le parole che contiene, usando come pesi gli indici di Inverse Document Frequency associati ad ogni parola; in questo modo è aumentata la rilevanza nella rappresentazione di un documento dei termini che sono meno comuni nella collezione.

Questa rappresentazione vettoriale di un documento nello stesso spazio in cui sono rappresentati i termini permette di calcolare la similarità semantica tra un termine e un documento o tra due documenti, oltre che tra due termini. Questa possibilità è alla base del funzionamento di un motore di ricerca semantico, ossia che tiene conto del significato dei termini. Se è possibile rappresentare un documento nello spazio vettoriale, infatti, si può rappresentare anche la query di un utente come il vettore somma dei termini presenti in essa. Si può, dunque, calcolare la rilevanza

dei documenti presenti nella collezione rispetto alla query come similarità del coseno tra il vettore che rappresenta la query e quelli che rappresentano i documenti; si può poi, come di consueto, ordinare i documenti in base alla rilevanza rispetto alla query, mostrando i primi  $k$  risultati all'utente.

## 2. Progetto SIAP-PAI

Lo scopo del progetto SIAP-PAI è quello di fornire una soluzione, mediante tecniche di Intelligenza Artificiale, a supporto dei processi gestionale-amministrativi e dell'osservatorio regionale dei contratti pubblici nell'ambito del quadro normativo di settore per l'implementazione degli appalti pubblici.

Questo capitolo ha lo scopo di fornire una panoramica completa dell'architettura del sistema, mediante diversi punti di vista che permettano di rappresentare i diversi aspetti della soluzione proposta. Lo scopo primario è dunque quello di catturare e trasmettere le decisioni architetturali alla base del sistema.

### 2.1 Architettura ad alto livello

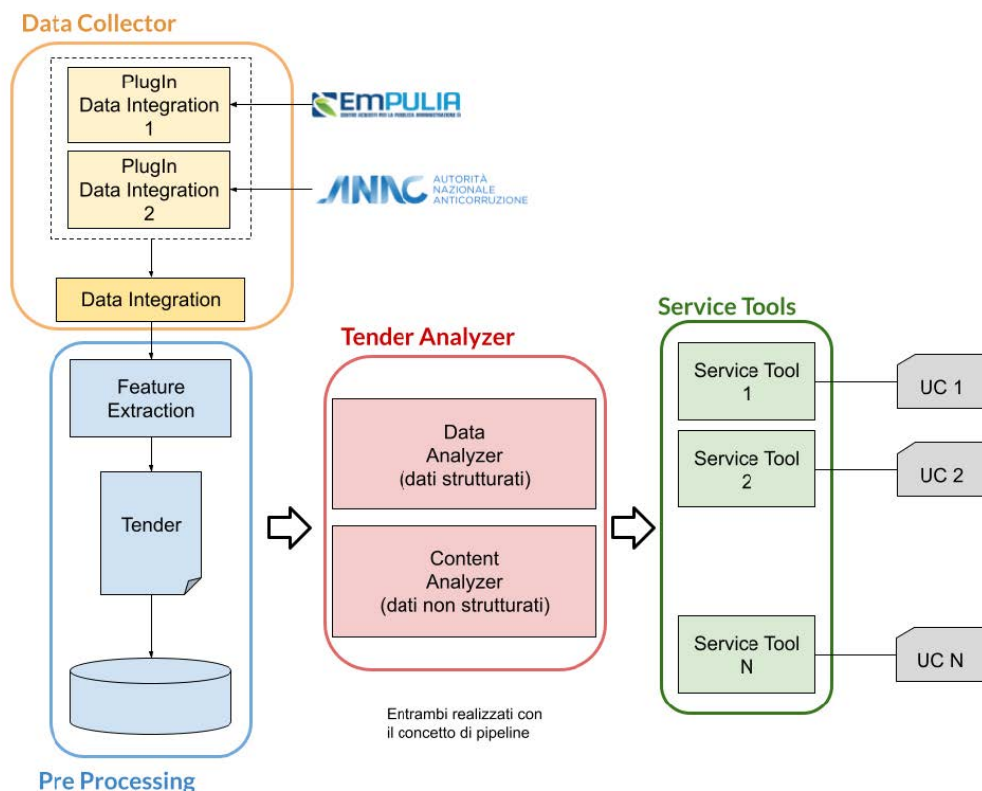


Figura 2.1 Architettura ad alto livello del sistema



La figura mostra lo schema dell'architettura ad alto livello per la creazione di un sistema in grado di fornire supporto nella gestione del ciclo di appalti. L'architettura è suddivisa in quattro diversi moduli:

- Data Collector
- Pre Processing
- Tender Analyzer
- Service Tools

### 2.1.1 Data Collector

Il modulo denominato Data Collector, come suggerisce il nome, si occupa della raccolta dei dati relativi ai bandi e alle gare. Tali dati possono essere estratti da diverse sorgenti di dati: banche dati a livello europeo come quella di TED, Nazionale come SIMOG/ANAC o anche regionale come nel caso di EmPULIA. Allo stato attuale, si è deciso di concentrarsi su SIMOG/ANAC ed EmPulia.

### 2.1.2 Pre Processing

Il modulo di Pre-Processing si occupa di sublimare le informazioni estratte tramite il modulo di Data Collection in entità sulle quali possano essere efficacemente effettuati gli step successivi di analisi. Per questo motivo, è necessario stabilire quali sono le feature di interesse relative ai bandi e successivamente memorizzarle mediante l'ausilio di database o di particolari indici.

### 2.1.3 Tender Analyzer

Il Tender Analyzer ha lo scopo di effettuare le analisi a partire dalle informazioni estratte dalle diverse sorgenti. Data la natura delle sorgenti, questo modulo è suddiviso in due diverse componenti:

- il *Data Analyzer* si occupa di analizzare l'informazione strutturata associata ai bandi: codici, come ad esempio CUP, CIG, date, importi, etc.;

- il *Content Analyzer*, invece, effettua l'analisi dell'informazione non strutturata associata ai bandi. Esempi possono essere rappresentati dagli eventuali allegati inerenti al bando (determine, capitolati, etc.). Il Content analyzer effettua l'analisi testuale utilizzando tecniche di Natural Language Processing (NLP).

Di tutto ciò si occupa il Semantic Framework illustrato nei capitoli successivi.

#### 2.1.4 Service Tools

L'ultimo modulo previsto dall'architettura è rappresentato da una serie di Service Tools. Date le informazioni e le analisi svolte all'interno del modulo precedente, con i service tools vengono effettuate le operazioni specifiche per la realizzazione di particolari casi d'uso, come per esempio la visualizzazione a schermo di specifiche informazioni riguardo un bando o bandi simili a quello ricercato. L'interfaccia, illustrata nei capitoli successivi, si occupa di questo.

### 2.2 Open Data

Con il termine Open Data si introduce un nuovo paradigma nella gestione dei dati tipicamente "nascosti" in applicazioni o basi di dati che vengono resi accessibili a chiunque. L'obiettivo è quello di abbattere, per quanto possibile e ragionevole, le restrizioni tecnologiche e stabilire vincoli legali minimi al riuso dei dati. Una definizione comunemente accettata è la seguente: *"un contenuto o un dato si definisce aperto se chiunque è in grado di utilizzarlo, ri-utilizzarlo e ridistribuirlo, soggetto, al massimo, alla richiesta di attribuzione e condivisione allo stesso modo"*.

Esempi di Open Data sono i formati CSV e Json.

Mentre in generale gli Open Data abbattano le barriere culturali, legali ed economiche al riuso, con i Linked Open Data [16] ci si concentra piuttosto sulla messa a punto di strumenti che permettano di dare ai dati (aperti e

non) un'identità, di renderli collegati tra loro e interoperabili. Il problema dell'interrogazione fra dati di sistemi diversi è reso ancora più complesso dal fatto che ogni sistema si basa tipicamente su infrastrutture eterogenee (diversi linguaggi, formati, protocolli, ecc.). Dunque è possibile che, ad uno stesso oggetto, sistemi differenti assegnino identità differenti. Pertanto, queste definizioni andrebbero allineate per continuare a garantire l'interoperabilità tra sistemi eterogenei. Adottare modelli, tecnologie e standard aperti di Linked Data (e.g., RDF), sfruttando le migliori esperienze maturate nell'ambito del Web Semantico, offre benefici di sicuro interesse per utenti e sviluppatori.

### 2.2.1 Open Data strutturati da ANAC

Da qualche tempo, un numero crescente di amministrazioni, rendendosi conto del valore aggiunto dei Linked Data, si sta facendo carico non solo del lavoro necessario per pubblicarli come dati aperti ma anche di quello di pubblicarli direttamente in modalità "linked". Dalle ricerche effettuate sono emersi diversi enti che operano con dati della Pubblica Amministrazione italiana che mettono a disposizione open data, tra cui il più rilevante è l'ANAC. L'ANAC è un'autorità amministrativa indipendente la cui missione istituzionale è individuata nell'azione di prevenzione della corruzione in tutti gli ambiti dell'attività amministrativa. Fra tutte le sorgenti disponibili, ANAC presenta il più completo numero di elementi rispetto alle altre.

Per rispondere alle esigenze di accedere ai bandi di gara in maniera più efficiente si è sviluppato un applicativo che espone alcune API con lo scopo di renderle accessibili, filtrando i dati a piacimento dell'utilizzatore. Il fine ultimo dell'applicativo è quello di andare a recuperare tutti i dataset resi disponibili dal sito ANAC e salvarli in una base dati adeguatamente strutturata.

## 2.3 Bando di gara

Il bando di gara è lo strumento attraverso il quale è possibile indire una procedura di scelta del contraente, adottato da una stazione appaltante, che intenda aggiudicare un contratto pubblico. È un atto amministrativo di natura generale a rilevanza esterna, con il quale la stazione appaltante rende nota agli operatori economici la propria volontà di pervenire alla conclusione del contratto. In ogni bando di gara ci si può imbattere in diversi termini, tra i quali risulta di particolare importanza il Cig.

Il Cig è un codice adottato in Italia, che permette di identificare un dato contratto sottoscritto con la Pubblica Amministrazione, in seguito ad una gara d'appalto o affidato con una delle altre modalità consentite dal codice dei contratti pubblici. È costituito da una sequenza di 10 caratteri alfanumerici ed è utilizzato anche ai fini della tracciabilità dei flussi finanziari relativi ai contratti pubblici.

## 2.4 Struttura database relazionale

Come già detto, nel modulo di Pre Processing si salvano in un database in maniera strutturata i dati recuperati dal Data Collector. La struttura del database è la seguente.

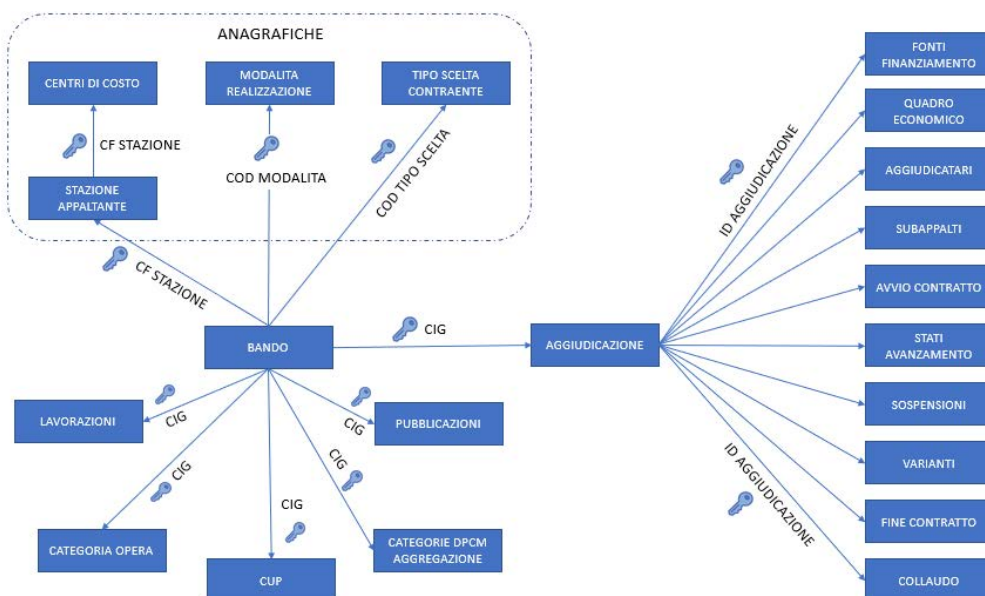


Figura 2.2 Struttura del database

Nel database, il Cig rappresenta la chiave primaria tra le tabelle dove sono stati salvati i vari dati.

## 2.5 Dati non strutturati

Sebbene siano disponibili in letteratura numerosi tool per effettuare analisi testuale, data la natura stessa dei dati presi in esame, occorre concentrarsi su tecniche in grado di processare testi scritti in italiano. Questo, tuttavia, pone un'ulteriore difficoltà data la scarsità di risorse disponibili per la lingua italiana.

I dati presenti in forma non strutturata (disponibili nei formati: doc/docx, pdf, p7m) attualmente sono tutti provenienti dalla piattaforma EmPulia, poiché è l'unica a mettere a disposizione in maniera diretta non solo i metadati, ma anche gli allegati relativi ad un bando. I dati non strutturati estratti da EmPulia, verranno memorizzati separatamente e indicizzati sulla base del Cig del bando a cui fanno riferimento.

Per poter applicare tecniche di NLP (sfruttata dal Semantic Framework) sul testo associato ai bandi è necessario riportare tali file ad un formato contenente esclusivamente testo. Questo obiettivo può essere raggiunto adoperando tecniche di layout extraction e Optical Character Recognition.

### 3. Semantic Framework e servizi backend

#### 3.1 Semantic Framework

In questo paragrafo si forniscono dettagli sul Semantic Framework, in particolare l'architettura del sistema e i suoi componenti.

La figura seguente mostra i componenti principali del nostro framework.

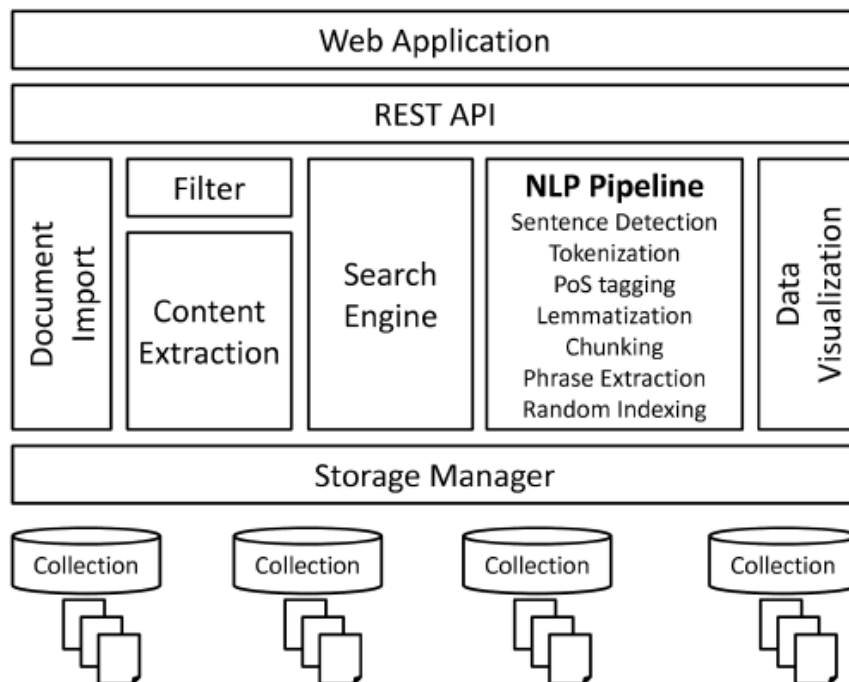


Figura 3.1 Architettura del Semantic Framework

Tutte le funzionalità sono esposte come servizi tramite un'API REST. Il front-end è stato sviluppato come un'applicazione Web che si basa sull'API REST. L'applicazione Web fornisce un facile accesso a tutte le funzionalità. In particolare, l'applicazione Web è stata personalizzata per le esigenze del progetto, al fine di supportare raccolte multiple di documenti e consente la ricerca e analisi su uno o più di essi.

Un componente fondamentale dell'architettura è lo Storage Manager, che si occupa di tutte le operazioni I/O sulle raccolte di documenti. Lo Storage Manager fornisce:

- un livello di rappresentazione astratta sia per i documenti che per le raccolte di documenti;
- accesso alle funzionalità di indicizzazione e ricerca;
- un'interfaccia di comunicazione sia con Content Extraction che con i componenti della NLP pipeline.

Storage Manager archivia i documenti nel file system. I documenti possono essere raggruppati in collezioni (o raccolte), ognuna delle quali ha un nome e una lingua predefinita ed è composta da un insieme di documenti, un indice, un WordSpace e un file contenente le informazioni estratte dalla pipeline NLP. Nel caso di studio in esame, l'unica collezione esistente è "bandi". Inoltre, una collezione può essere pubblica o privata. Una collezione pubblica è accessibile a tutti gli utenti, mentre una raccolta privata è accessibile solo all'utente che l'ha creata. In una raccolta pubblica, è possibile assegnare ad ogni utente diversi privilegi: amministratore, solo lettore, lettore e scrittore.

I documenti possono essere aggiunti a qualsiasi raccolta. In particolare, il sistema è in grado di importare documenti (componente Document Import) da una directory, un singolo file o un CSV. L'importazione da un CSV permette di dividere un documento in più sezioni definite dall'utente. Esempio di sezione sono: titolo, abstract e contenuto. Durante l'importazione, l'utente può selezionare i campi del CSV a cui è interessato. Quando l'utente importa documenti da una directory o da un singolo file, il sistema è in grado di riconoscere il formato del file ed estrarne il contenuto testuale. Il processo di estrazione viene implementato utilizzando Apache Tika [3]. Il toolkit Apache Tika è in grado di rilevare ed estrarre metadati e testo da diversi tipi di file (DOC, PDF, PPT e XLS).

Tutti questi tipi di file possono essere analizzati tramite un'unica interfaccia, rendendo Tika utile per l'analisi dei contenuti. Durante il processo di importazione, il nostro sistema può applicare filtri personalizzati (componente Filtro) in grado di identificare le diverse sezioni di un documento, in modo simile a quanto accade per CSV. I filtri possono essere definiti dagli utenti sfruttando le espressioni regolari per identificare l'inizio e la fine di una sezione del documento. In tal modo, è possibile filtrare le sezioni ridondanti o creare una raccolta contenente solo informazioni specifiche.

Il motore di ricerca fornisce supporto per l'indicizzazione e il recupero dei documenti. Ogni collezione può essere indicizzata utilizzando un classico modello a spazio vettoriale implementato da Apache Lucene [4]. Gli utenti possono eseguire ricerche nella raccolta utilizzando parole chiave e operatori booleani. Inoltre, questa componente supporta anche una ricerca semantica eseguita attraverso uno spazio distributivo in cui i concetti correlati sono rappresentati come punti vicini nello spazio. Lo spazio distributivo, che è il componente chiave della ricerca semantica, è costruito dalla pipeline di NLP attraverso il Random Indexing (RI).

Il componente Data Visualization sfrutta strumenti grafici per l'analisi semantica di documenti in una collezione.

Infine, il cuore della piattaforma è il componente di elaborazione del linguaggio naturale (pipeline NLP), che alimenta sia il motore di ricerca che i componenti di visualizzazione dei dati. La pipeline è in grado di eseguire diverse fasi di elaborazione del testo per l'inglese e l'italiano:

- Sentence Detection: divide un testo in frasi, sfruttando i caratteri di punteggiatura che segnano la fine di una frase.
- Tokenizzazione: divide il testo in token. Ogni token è una parola.



- Part-of-Speech (POS) tagging: identifica il ruolo grammaticale di ogni parola: sostantivo, verbi, aggettivo, avverbio, punteggiatura, preposizione e così via.
- Lemmatizzazione: fornisce il lemma per ogni parola. Il lemma è la forma base di una parola, ad esempio la forma singolare di un sostantivo o la forma infinita di un verbo.
- Chunking: divide un testo in parti di parole sintatticamente correlate, come nomi o gruppi verbali, ma non ne specifica né la struttura interna né il ruolo nella frase principale.
- Phrase Extraction: è in grado di trovare sequenza di parole che identificano un singolo concetto. Esempi sono: Information Retrieval, Document Management, Public Administration.
- Random Indexing: costruisce un WordSpace analizzando una raccolta di documenti. Un WordSpace è uno spazio geometrico in cui le parole sono rappresentate come punti. Se due le parole sono vicine nel WordSpace sono semanticamente correlate.

Il capitolo 1 descrive teoricamente le tecniche sfruttate nella pipeline NLP. Nei paragrafi successivi viene invece illustrato la loro integrazione attraverso servizi REST.

### 3.1.1 Motore di Ricerca

Il motore di ricerca è in grado di recuperare i documenti in base alla query dell'utente. In particolare, il motore di ricerca classifica i documenti calcolando un punteggio di pertinenza tra la query dell'utente e il documento. Il punteggio di rilevanza può essere calcolato utilizzando diversi approcci, nel Semantic Framework implementiamo due metodologie:

- ricerca classica: questo approccio si basa sul classico modello spaziale vettoriale. Questo modello è in grado di recuperare solo i

documenti che contengono almeno una delle parole chiave fornite nella richiesta dell'utente;

- ricerca semantica: questo approccio è in grado di mappare sia i documenti che le query come punti di accesso dello stesso WordSpace usato per rappresentare le parole. Dopo questa mappatura è possibile calcolare la similarità coseno tra ogni documento e la query dell'utente e poi classificare i documenti in base alla loro somiglianza. Questo approccio può recuperare documenti che non contengono la parola chiave fornita nella query.

Il capitolo 1 descrive teoricamente il funzionamento dei due tipi di motore di ricerca.

### 3.2 Servizi RESTful

Lo stile architetturale REST è una tecnica di sviluppo di servizi Web concepita nel 2000 da Roy Fielding [12].

Le architetture REST (ossia REpresentational State Transfert) impiegano il protocollo HTTP per trasferire tutte le informazioni in una comunicazione tra host client e server. Nel protocollo HTTP ogni entità che si può richiedere ad un host server è considerata una “risorsa”, identificabile e localizzabile univocamente tramite il suo indirizzo URL [13]. Ogni risorsa, in un dato momento della comunicazione, si trova in uno “stato”; tramite un servizio REST si può, quindi, o richiedere una rappresentazione dello stato di una risorsa, oppure trasmettere una rappresentazione dello stato che si vorrebbe la risorsa raggiungesse, per richiederne la modifica. Dai concetti di risorsa, stato e rappresentazione di uno stato deriva, dunque, l’acronimo REpresentational State Transfert.

Un servizio web sviluppato secondo questo stile architetturale, dunque, è accessibile a un computer host tramite una semplice richiesta HTTP

all'host server che espone il servizio. A seconda che il servizio richiamato abbia lo scopo di modificare o ottenere informazioni presenti sull'host server, la richiesta HTTP dev'essere formulata tramite i metodi HTTP progettati per ciascuno scopo particolare; ad esempio, il metodo GET è specifico per la richiesta di una risorsa senza modifica, mentre i metodi POST e PUT sono specifici per le modifiche delle risorse.

I vantaggi dell'uso di un'architettura REST sono la sua grande flessibilità, l'indipendenza tra client e server nella comunicazione e la leggerezza dei messaggi scambiati, grazie alla quale si ottengono ottime performance.

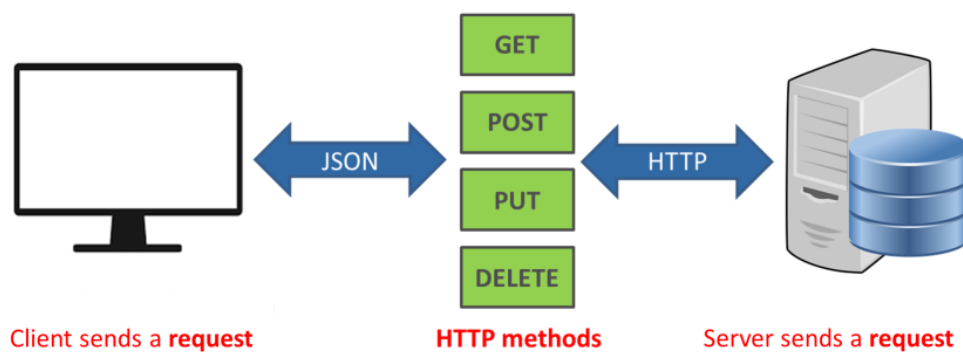


Figura 3.2 Rappresentazione del protocollo REST

### 3.3 Servizi RESTful impiegati

I servizi RESTful messi a disposizione dal Semantic Framework vengono utilizzati sia per la costruzione di un motore di ricerca classico, che per quella di un motore di ricerca semantico. Esse effettuano varie operazioni sui documenti di una collezione, sulla quale gli utenti della piattaforma potranno compiere operazioni di ricerca fornendo in input un insieme di parole chiave.

### 3.3.1 NLP Pipeline

Prima di poter creare il motore di ricerca semantico è necessario effettuare le operazioni di NLP sui documenti della collezione presa in considerazione.

Queste operazioni, già descritte nel capitolo 1, sono:

- Sentences Detection
- Tokenization
- Rimozione delle stopWord
- POS Tagging
- Lemmatizzazione
- Phrase detection

Il semantic Framework fornisce un metodo REST per applicare la Pipeline NLP ad ogni documento. Come parametri il metodo richiede il nome della collezione e l'identificativo di un documento presente in essa su cui effettuare le operazioni di NLP:

`https://{host:port}/ivp/v2/nlp/{collection}/{docid}`

### 3.3.2 Estrazione delle phrases

L'operazione di estrazione delle phrases può essere realizzata attraverso due metodi REST presenti all'interno del Semantic Framework.

Il primo metodo è basato sull'idea che le parole che occorrono molto spesso insieme e molto raramente in contesti diversi possono essere considerate *key phrases*. Queste rappresentano dei gruppi di termini che esprimono un singolo concetto; ad esempio "Artificial Intelligence" è una espressione che denota un particolare ambito di ricerca informatico e viene considerata una *key phrase* poiché i due termini che la compongono assumono un significato diverso da quello che avrebbero se considerati singolarmente.

Questo metodo presenta due vantaggi: non ha bisogno di risorse esterne come dizionari o ontologie, ed è semplice da implementare poiché basato sulla frequenza delle parole.

Tutti i bi-grammi sono valutati attraverso la seguente formula:

$$score(w_i, w_j) = \frac{count(w_i, w_j) - minCount}{count(w_i) \cdot count(w_j)}$$

dove  $w_i$  e  $w_j$  sono due termini che occorrono nella collezione oggetto di analisi, e  $count()$  è la funzione che conta il numero di occorrenze delle parole all'interno della collezione, applicabile sia ad un singolo termine ( $count(w_j)$ ), sia ai bi-grammi ( $count(w_i, w_j)$ ). Inoltre,  $minCount$  è un valore che previene la possibile formazione di phrases con parole non molto frequenti. Sebbene questa tecnica sia in grado di individuare soltanto bi-grammi, ossia *key-phrases* composte da due soli termini, è possibile individuare n-grammi concatenando coppie di bi-grammi. Il metodo REST tramite il quale si può richiamare questo servizio richiede come parametri il nome della collezione sulla quale riconoscere le phrases, una soglia massima di bi-grammi da restituire e un numero che rappresenta la frequenza minima che un bi-gramma deve avere per poter essere restituito come *key-phrase*:

<https://{host:port}/ivp/v2/phrase/{collection}/w2p/{threshold}/{mincount}>

Il secondo metodo di riconoscimento delle phrases impiega un automa a stati finiti in grado di riconoscere sequenze di parole categorizzate all'interno di Wikipedia; si può quindi intuire che questo approccio è utile quando abbiamo liste di concetti predefiniti. Le liste di concetti che vengono prese in considerazione sono formate da n-grammi formati da un massimo di sei parole e un minimo di due; esse verranno utilizzate per la costruzione dell'automa che sarà in grado di riconoscere phrases all'interno di testi. L'elemento fondamentale di questo approccio sono le liste di concetti etichettate da Wikipedia che vengono considerate significative ai fini del riconoscimento delle phrases. Come parametri il metodo REST richiede unicamente il nome della collezione sulla quale effettuare l'estrazione:

<https://{host:port}/ivp/v2/phrase/{collection}/FSA>

### 3.3.3 Analyze

L'analisi statistica delle parole all'interno di grandi testi è in sempre più utilizzata grazie all'aumentare del potere computazionale disponibile. I DSM (Distributional Semantic Models) sono modelli semplici per costruire spazi geometrici di concetti. Essi sono conosciuti anche come Word Space e sono in grado di esaminare grandi testi ed estrarre il contesto d'uso delle parole presenti in essi. Nello spazio risultante la somiglianza semantica tra concetti viene espressa come vicinanza dei punti, visti come parole, nello spazio. In questo modo la somiglianza semantica viene calcolata come il coseno dell'angolo tra i due vettori che rappresentano le parole.

I DSM possono essere costruiti utilizzando diverse tecniche. Un approccio comune è il Latent Semantic Analysis, il quale si basa sulla decomposizione a valore singolare della matrice di co-occorrenza di parole. Tuttavia, esistono molti altri metodi che prendono in considerazione l'ordine delle parole oppure metodi che effettuano predizioni.

Queste operazioni di analisi semantica vengono realizzate attraverso una API REST fornita dal Semantic Framework che richiede come parametri il nome della collezione sul quale effettuare le operazioni e la lingua:

*`https://{host:port}/ivp/v2/analyze/{collection}/{language}`*

### 3.3.4 Search Engine e Random Indexing

Dopo aver effettuato tutte le operazioni sul testo e sulle collezioni si può effettivamente creare il motore di ricerca di tipo semantico.

Lo scopo del search engine è quello di ritrovare documenti che abbiano uno score di rilevanza rispetto alla query molto alto. Lo score di rilevanza può essere calcolato utilizzando varie metodologie:

- Classical search: questo approccio realizza il classico Vector Space Model discusso nel capitolo uno e restituisce documenti che

contengono almeno una delle parole presenti nella query dell'utente.

- Semantic Search: questo approccio si basa sul WordSpace model che viene realizzato attraverso il processo di Random Indexing il quale permette di ottenere i vettori semantici per tutti i termini di una collezione. All'interno del Semantic Framework l'approccio utilizzato per la costruzione del semantic search engine si basa su una semplice ed efficace somma tra i vettori semantici delle parole. Quindi i documenti vengono visti come somma dei vettori di ogni singola parola presente in essi e, durante il processo di ritrovamento, vengono confrontati con la query, anch'essa rappresentata come somma dei vettori associati ai termini che la compongono.

Per la creazione del Search Engine e l'esecuzione Random Indexing per ogni collezione, il Semantic Framework fornisce servizi REST che vengono richiamati solo dopo aver effettuato tutte le operazioni di NLP e estrazione delle phrases dai testi dei documenti delle collezioni.

L'API REST per la creazione del Search Engine richiede come parametro il nome della collezione su cui creare e sincronizzare il motore di ricerca:

*<https://{host:port}/ivp/v2/createSE/{collectionName}>*

Per la creazione del Random Indexing, invece, il servizio REST richiede come parametro il nome della collezione, la grandezza dei vettori mediante i quali saranno rappresentati termini e documenti (vetDim), e la grandezza del vocabolario (vocSize):

*<https://{host:port}/ivp/v2/createRI/{collectionName}/{vetDim}/{vocSize}>*

### 3.3.5 Related Concepts

Dato un concetto, la possibilità di individuare quelli più correlati ad esso è una funzionalità molto importante del Semantic Framework. Infatti, questa è in grado di aiutare l'utente a soddisfare il proprio bisogno informativo estendendo la ricerca con parole molto simili a quelle della query iniziale.

All'interno del Semantic Framework è stato implementato un servizio REST avente come compito quello di restituire una lista di parole simili a quelle della query. Il metodo REST richiede come input il nome della collezione su cui effettuare la ricerca, la grandezza della lista di output contenente i concetti simili e la query dell'utente:

*<https://{host:port}/ivp/v2/simw/{collection}/{topn}/{query}>*

### 3.3.6 Bandi e documenti simili

Una volta creato il motore di ricerca esso verrà utilizzato per effettuare richieste. I risultati delle query saranno i bandi o documenti che avranno un alto score di rilevanza rispetto alla richiesta dell'utente.

Una delle funzionalità più interessanti offerte dal sistema è la possibilità di cercare anche documenti o bandi simili ad un documento o bando ottenuto in seguito ad una query iniziale. Il metodo REST che si occupa di questa feature si comporta in maniera analoga al metodo dei Related Concept. Il metodo richiede infatti come input il nome della collezione su cui effettuare la ricerca, la grandezza della lista di output contenente i concetti simili e l'identificativo del documento rispetto al quale si vogliono cercare documenti simili:

*[https://{host:port}/ivp/v2/simd/{collection}/{topn}/{id\\_documento}](https://{host:port}/ivp/v2/simd/{collection}/{topn}/{id_documento})*

## 3.4 Metadati Strutturati

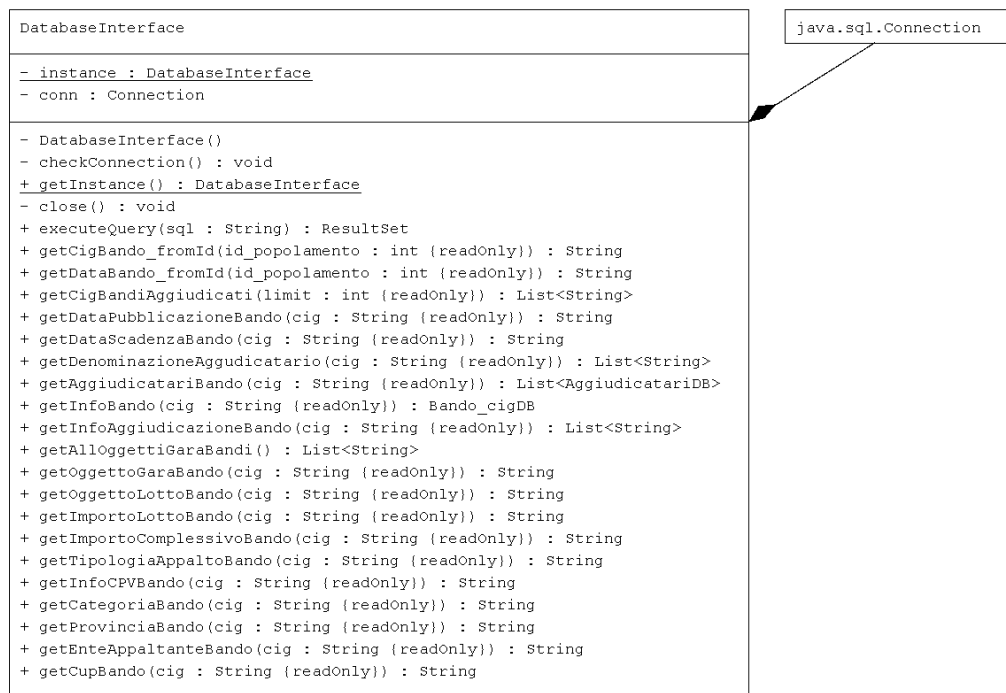
Le informazioni strutturate relative al dominio del progetto SIAP-PAI sono state organizzate all'interno di un database che è stato integrato al Semantic



Framework tramite l'aggiunta di un modulo finalizzato alla gestione dei metadati. Per la costruzione del database è stato impiegato il DBMS (Database Managment System) MySQL. La struttura del database MySQL è identica a quella mostrata in *Figura 2.2*, tuttavia sono state implementate come classi Java solo le tabelle relative a *Bandi* e *Aggiudicatari*, in quanto solo queste di interesse per il progetto.

### 3.4.1 Gestione connessione con la base di dati e API REST

La connessione al database è stata gestita tramite un'unica classe denominata *DatabaseInterface*, la quale, implementando il driver di connessione Java con database MySql, espone metodi che corrispondono alle query di inserimento dati e interrogazione del database utili per integrare i dati strutturati ai servizi del Sistema di Supporto alle Decisioni. I servizi di gestione dei metadati relativi alle collezioni illustrati in precedenza sono stati resi disponibili all'utente tramite API REST, in conformità con l'architettura del Semantic Framework.



*Figura 3.3 Classe di interfaccia con il database*

### 3.4.2 API REST per l'interrogazione del database

Al fine di fornire i dati necessari all'implementazione di servizi avanzati del Sistema di Supporto alle Decisioni, sono stati creati dei metodi REST tramite i quali è possibile ottenere le informazioni contenute nel database relative a singoli Bandi o Aggiudicatari. Tali metodi sono stati implementati nella classe *RestMetadataBandi*.

RestMetadataBandi
<pre>+ get_prova() : Response + get_prova2(text : String {readOnly}) : Response + getCigBando(id_popolamento : int {readOnly}) : Response + getDataBando(id_popolamento : int {readOnly}) : Response + getCigBandiAggiudicati(limit : int {readOnly}) : Response + getDataPubblicazioneBando(cig : String {readOnly}) : Response + getDataScadenzaBando(cig : String {readOnly}) : Response + getDenominazioneAggiudicatario(cig : String {readOnly}) : Response + getAggiudicatariBando(cig : String {readOnly}) : Response + getInfoBando(cig : String {readOnly}) : Response + getInfoAggiudicazioneBando(cig : String {readOnly}) : Response + getAllOggettiGaraBandi() : Response + getOggettoGaraBando(cig : String {readOnly}) : Response + getOggettoLottoBando(cig : String {readOnly}) : Response + getImportoLottoBando(cig : String {readOnly}) : Response + getImportoComplessivoBando(cig : String {readOnly}) : Response + getTipologiaAppaltoBando(cig : String {readOnly}) : Response + getInfoCPVBando(cig : String {readOnly}) : Response + getCategoriaBando(cig : String {readOnly}) : Response + getProvinciaBando(cig : String {readOnly}) : Response + getLuogoLavoriBando(cig : String {readOnly}) : Response + getEnteAppaltanteBando(cig : String {readOnly}) : Response + getCupBando(cig : String {readOnly}) : Response + getInfoPrincipaliBando(cig : String {readOnly}) : Response + getInfoDettaglioBando(cig : String) : Response</pre>

Figura 3.4 Classe che espone le API REST

I metodi REST esposti non sono in rapporto uno ad uno con i metodi presenti nella classe *DatabaseInterface*, che compiono le interrogazioni al database tramite il driver MySQL. Essi, infatti, pur utilizzando i metodi esposti dalla classe, spesso filtrano i risultati delle query esponendo di fatto metodi di accesso ai dati di granularità più fine rispetto a quelli esposti dalla classe di interfaccia diretta. Ciò è stato fatto per fornire all'utente risultati più specifici, adattati alle particolari informazioni che è necessario integrare nei servizi del Sistema di Supporto alle Decisioni. Tutti i dati richiesti tramite API REST

vengono restituiti dai metodi in formato Json, per garantire l'interoperabilità con altri sistemi.

<b>Request</b>	https://localhost:9999/sf/v2/getAggiudicatariBando/85918391D7
<b>Body</b>	[{"id_popolamento":679637,"cig":"85918391D7","ruolo":"MANDANTE","codice_fiscale":"10580381001","denominazione":"LASER S.R.L.","tipo_soggetto":"ATI (RAGGRUPPAMENTI TEMPORANEI DI CONCORRENTI: CONSORZI ORDINARI DI CONCORRENTI)","id_aggiudicazione":2051009,"data_inserimento":1649116800000}, {"id_popolamento":932799,"cig":"85918391D7","ruolo":"MANDATARIA","codice_fiscale":"01278311004","denominazione":"IZI SPA","tipo_soggetto":"ATI (RAGGRUPPAMENTI TEMPORANEI DI CONCORRENTI: CONSORZI ORDINARI DI CONCORRENTI)","id_aggiudicazione":2051009,"data_inserimento":1649116800000}, {"id_popolamento":1542369,"cig":"85918391D7","ruolo":"MANDANTE","codice_fiscale":"05110620589","denominazione":"CLES S.R.L.","tipo_soggetto":"ATI (RAGGRUPPAMENTI TEMPORANEI DI CONCORRENTI: CONSORZI ORDINARI DI CONCORRENTI)","id_aggiudicazione":2051009,"data_inserimento":1649116800000}]

Figura 3.5 Esempio di chiamata al metodo di interrogazione al database

## 4. Strumenti utilizzati per l'interfaccia

Per l'implementazione dell'interfaccia web, sono stati utilizzati diversi tipi di linguaggi di markup e scripting, tra cui AngularJS e JavaScript. Sono evidenziate di seguito le loro principali caratteristiche e documentate le principali funzionalità sviluppate.

### 4.1 AngularJS

AngularJS è un Framework per la creazione di applicazioni composta da un insieme di diverse funzionalità. Esso potenzia l'approccio dichiarativo dell'HTML nella definizione dell'interfaccia grafica e fornisce strumenti per la costruzione di un'architettura modulare e testabile della logica applicativa di un'applicazione.

AngularJS fornisce tutto quanto occorre per creare applicazioni moderne che sfruttano le più recenti tecnologie, come ad esempio le Single Page Application, cioè applicazioni le cui risorse vengono caricate dinamicamente su richiesta, senza necessità di ricaricare l'intera pagina. Esso consente di ampliare la sintassi HTML per estendere i componenti della propria applicazione in modo chiaro e conciso.

### 4.2 JavaScript

JavaScript è il principale linguaggio di programmazione per lo sviluppo di applicazioni web. Sempre più diffuso, tocca ormai gli ambiti mobile, server e desktop. L'enorme diffusione di JavaScript è dovuta principalmente al fiorire di numerose librerie nate allo scopo di semplificare la programmazione sul browser, ma anche alla nascita di framework lato server e nel mondo mobile che lo supportano come linguaggio principale. Le caratteristiche principali di JavaScript sono:

- essere un linguaggio interpretato: il codice non viene compilato, ma eseguito direttamente; in JavaScript lato client, il codice viene eseguito dall'interprete contenuto nel browser dell'utente.
- la sintassi è relativamente simile a quella dei linguaggi C, C++ e Java.
- definisce le funzionalità tipiche dei linguaggi di programmazione ad alto livello (strutture di controllo, cicli, ecc.) e consente l'utilizzo del paradigma object oriented.
- è un linguaggio debolmente tipizzato.
- è un linguaggio debolmente orientato agli oggetti.

In JavaScript, il codice viene eseguito direttamente sul client e non sul server. Il vantaggio di questo approccio è che, anche con la presenza di script particolarmente complessi, il web server non rischia sovraccarichi dato che il lavoro viene svolto dal client. Un rovescio della medaglia è che, nel caso di script particolarmente grandi, il tempo per il trasferimento dalla rete può diventare eccessivamente lungo. Inoltre ogni informazione che presuppone un accesso a dati memorizzati in una base di dati remota deve essere rimandata a un linguaggio che effettui materialmente la transazione, per poi restituire i risultati ad una o più variabili JavaScript.

## 4.3 Scripting

Di seguito sono documentati metodi implementati con AngularJs e JavaScript per la visualizzazione dei risultati.

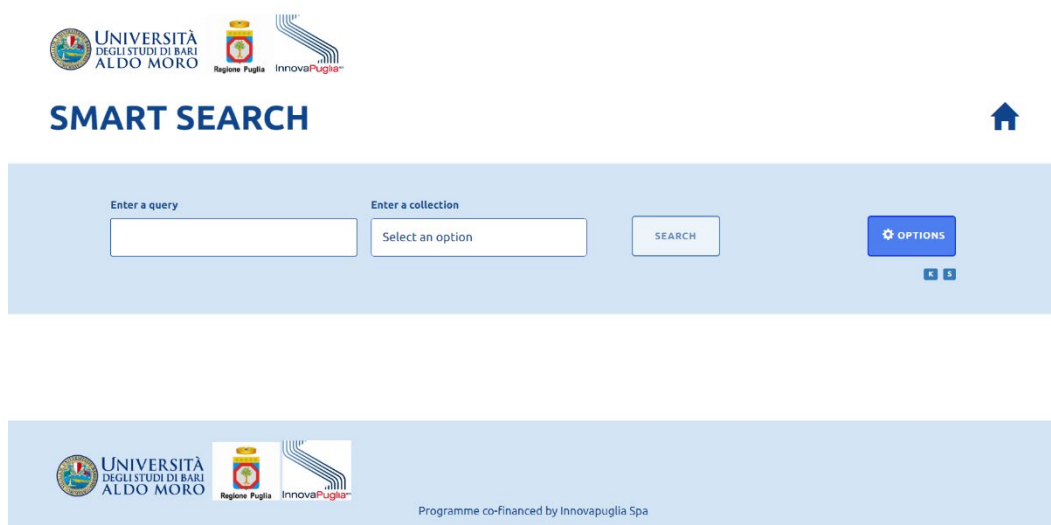
### 4.3.1 Ricerca e risultati

Il compito dell'applicazione web è quello di rendere disponibili le funzionalità della parte back-end all'utente. Per fare ciò gli elementi dell'interfaccia effettuano le chiamate alle API REST del Semantic Framework. Mediante lo scripting in AngularJS, cliccando sul pulsante

*Search* dello Smart Search, viene chiamato il metodo *search()*, tramite la direttiva *ng-submit* presente nella *view* dell'interfaccia.

```
<form method="get" name="filter" class="clear" ng-submit="search()" ng-model="collection">
```

Il metodo *search()*, definito nel controller, inizialmente inizializza tutte le variabile legate ai risultati di ricerca a valori nulli, successivamente, definisce i parametri per la chiamata al servizio REST legata alla ricerca: tipo di ricerca, se semantica, basata sulle keyword, o combinata. Sulla base di questi valori effettua la chiamata al servizio REST passando come parametri la query, la collezione, il numero massimo di risultati da ricevere e il tipo di ricerca.



The screenshot shows the 'SMART SEARCH' interface. At the top, there are logos for the 'UNIVERSITÀ DEGLI STUDI DI BARI ALDO MORO', 'Regione Puglia', and 'InnovaPuglia'. Below the logos, the title 'SMART SEARCH' is displayed in blue, followed by a home icon. The main search area is a light blue box containing a search bar labeled 'Enter a query', a dropdown menu labeled 'Enter a collection' with the option 'Select an option', a 'SEARCH' button, and an 'OPTIONS' button. At the bottom of the page, the same logos are repeated, along with the text 'Programme co-financed by Innovapuglia Spa'.

Figura 4.1 Schermata di ricerca

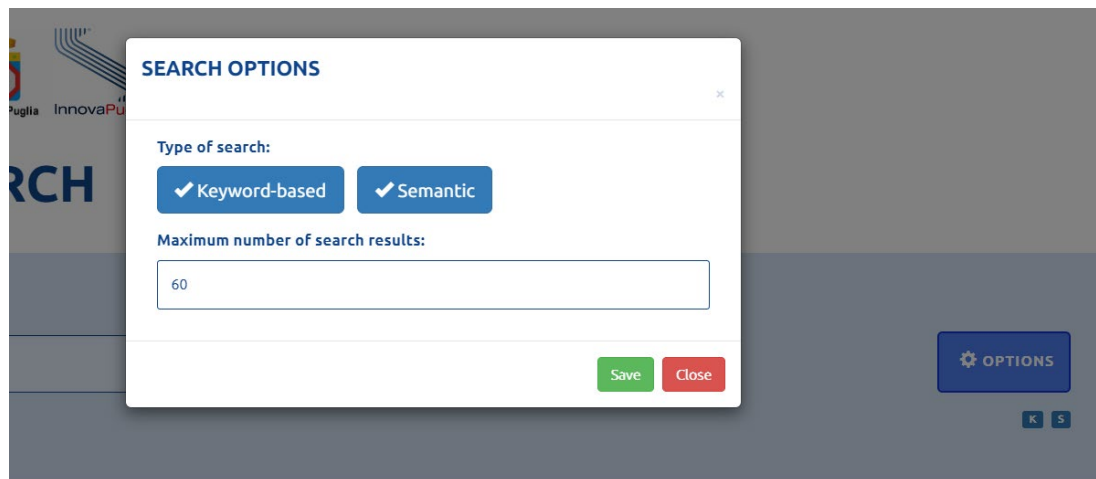


Figura 4.2 Schermata opzioni

I risultati ottenuti, vengono assegnati alla variabile `$scope.results`, filtrata mediante la chiamata del metodo `loadItems()` che definisce il numero di risultati da visualizzare nella *view* dell'applicazione web.

I risultati da visualizzare sono salvati nella variabile `$scope.resultsDisp`, richiamata nella *view* dell'interfaccia mediante la direttiva `ng-repeat`.

```
<div ng-repeat="result in resultsDisp" class="annuaire row-projects">
```

Per una migliore resa grafica, ed una facile lettura dei risultati, si è deciso di far visualizzare inizialmente al massimo 10 risultati, ognuno dei quali con una barra blu indicante il grado di rilevanza con la query. È possibile estendere la visualizzazione dei risultati cliccando sul pulsante *Load more results* (il metodo richiamato è sempre `loadItems()`). Esso, infatti, gestisce anche il numero di risultati da visualizzare: ad ogni sua chiamata, incrementa di una unità la variabile legata al numero di pagine da visualizzare. Questo valore viene poi moltiplicato per la costante che definisce il numero di valori da visualizzare ad ogni step.



<b>OGGETTO:</b> STRUMENTI MICROSOFT OFFICE 365 BUSINESS STANDARD  <b>Cig:</b> 90717748B9 <b>Vincitori bando:</b> <b>Allegato:</b>	<b>Ente appaltante:</b> PAGOPA S.P.A. <b>Provincia:</b> <b>Importo complessivo:</b> € 13.860 <b>Data pubblicazione:</b> 2022-01-20 <b>Data aggiudicazione:</b> 2022-01-21
---	---

Bandi simili

---

<b>OGGETTO:</b> TOOL PER GESTIONE GRAFICI SU MICROSOFT OFFICE  <b>Cig:</b> 8985120396 <b>Vincitori bando:</b> <b>Allegato:</b>	<b>Ente appaltante:</b> PAGOPA S.P.A. <b>Provincia:</b> <b>Importo complessivo:</b> € 2.505 <b>Data pubblicazione:</b> 2022-01-21 <b>Data aggiudicazione:</b> 2022-01-21
--	--


Bandi simili

Load more results

Figura 4.3 Schermata risultati con tasto Load more results

### 4.3.2 Ricerca con Related Concepts

Come già illustrato nei capitoli precedenti, è stata implementata un metodo REST in grado di restituire una lista di concetti correlati alla query inserita come input. Tale lista è utilizzabile per effettuare automaticamente nuove ricerche grazie alla funzione JavaScript `$scope.changeSearch(word)` presente nel controller e richiamata nella pagina html attraverso il comando di AngularJS `ng-click`. La parola selezionata sarà passata come input (`word`) alla funzione, la quale effettuerà una nuova ricerca con quel parametro come query.



## SMART SEARCH

related concepts:

Figura 4.4 Related Concepts



### 4.3.3 Ricerca bandi simili

Per ogni risultato ottenuto dalla ricerca, cliccando sul pulsante *Bandi simili*, l'utente ha la possibilità di visionare i bandi aventi un contenuto molto simile a quello selezionato. Il metodo del *controller* chiamato è *simd(bando, bandoCig, init)*, i cui parametri sono l'id del bando per il quale si vogliono ricercare bandi simili, il cig del bando, e un valore booleano che se vero inizializza l'array *\$scope.goBack*, ovvero l'array utilizzato per tener traccia della navigazione tra ricerche di documenti simili. Il metodo *simd()*, inizialmente inizializza tutte le variabili e i parametri legati alla ricerca in questione, successivamente effettua la chiamata ai servizi REST. I metodi REST chiamati sono:

1. `https://{host:port}/sf/v2/simd/{collection}/{topn}/{id_documento}`
2. `https://{host:port}/sf/v2/getInfoDettaglioBando/{bandoCig}`
3. `https://{host:port}/sf/v2/getAggiudicatariBando/{bandoCig}`

Il primo, come già illustrato nei capitoli precedenti, si occupa di recuperare i documenti o bandi simili ad un documento o bando dato il suo identificativo e la collezione. I risultati sono salvati in un array e successivamente analizzati: viene estratto il cig da ogni risultato, ed usato per effettuare le successive chiamate REST per estrarre le informazioni di nostro interesse dal database dei metadati, ovvero i dettagli del bando e gli eventuali aggiudicatari.

Nel nostro caso, si ottiene un elenco contenente al massimo 10 risultati i quali vengono assegnati all'array *\$scope.simResults* che va ad aggiornare l'elenco dei risultati nella *view* dell'interfaccia mediante la direttiva *ng-repeat*.

```
<table class="table table-striped" ng-if="simResults.length!=0">
  <tr ng-repeat="sim in simResults">
```

Per ogni bando simile è consentito estendere ulteriormente la ricerca richiedendo ulteriori bandi simili a quello di interesse dell'utente. Ciò è

possibile cliccando il pulsante *Bandi simili*, il quale richiama a sua volta il metodo *simd(bando, bandoCig, init)*, documentato in precedenza. Cliccando sul pulsante *Indietro* è possibile navigare tra i risultati tornando alle viste precedenti. In questo caso viene chiamato il metodo *gobackview()* del *controller*, il quale estrae tramite metodo LIFO (Last In First Out) dall'array *\$scope.goback*, contenente lo storico di navigazione tra le ricerche di documenti simili, l'ultimo bando inserito, il quale viene passato come parametro al metodo *simd()* che effettua nuovamente la ricerca.

Così come nella schermata dei risultati principale, anche in questo caso è presente una barra indicante lo score di rilevanza.



Figura 4.5 Schermata bandi simili



Figura 4.6 Schermata bandi simili – tasto indietro

### 4.3.4 Visualizzazione dei dettagli bando

Per ogni risultato, è possibile visualizzare ulteriori informazioni riguardo al bando, gli eventuali vincitori e documenti allegati. La funzione che si occupa di aprire una nuova pagina di visualizzazione per i dettagli è `$scope.goToDetails(bandoCig)`, che prende come parametro il cig del bando selezionato. Successivamente dei metodi REST, dato il cig come parametro, si occupano di recuperare i metadati da visualizzare.

Gli allegati sono recuperati attraverso il metodo REST:

`https://{host:port}/sf/v2/search/{collection}/{topn}/cig/{bandoCig}`

ed è possibile effettuare il download tramite la funzione `$scope.composeAndDownload(bandoCig,bandoTitle)`.

La sezione Ulteriori Informazioni collega ad una dashboard che permette di visualizzare altri dettagli relativi al bando, quali: quadro economico, subappalti, categoria opera, stati di avanzamento ecc...



## DETTAGLI BANDO

**CIG: 788280152B**

Dettagli	Vincitori	Documenti	Ulteriori Informazioni
<h3>DETTAGLI</h3> <p><b>Oggetto:</b> AFFIDAMENTO DEL SERVIZIO DI ASSISTENZA TECNICA(IMAC) E MANUTENZIONE DELLE POSTAZIONI DI LAVORO (PDL) <b>Cig:</b> 788280152B <b>Cup:</b> <b>Importo lotto:</b> € 672285 <b>Tipologia appalto:</b> SERVIZI <b>Cpv:</b> 72611000-6 - SERVIZI DI ASSISTENZA TECNICA INFORMATICA <b>Categoria:</b> FORNITURA DI SERVIZI <b>Classe:</b> <b>Luogo esecuzione lavori:</b> FOGGIA</p>			

Figura 4.7 Schermata dettagli bando

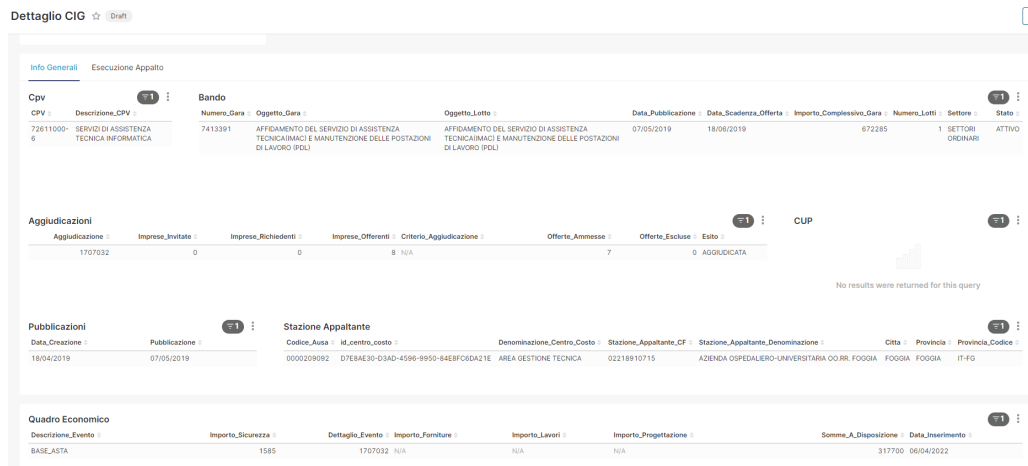


Figura 4.8 Dashboard dettagli bando

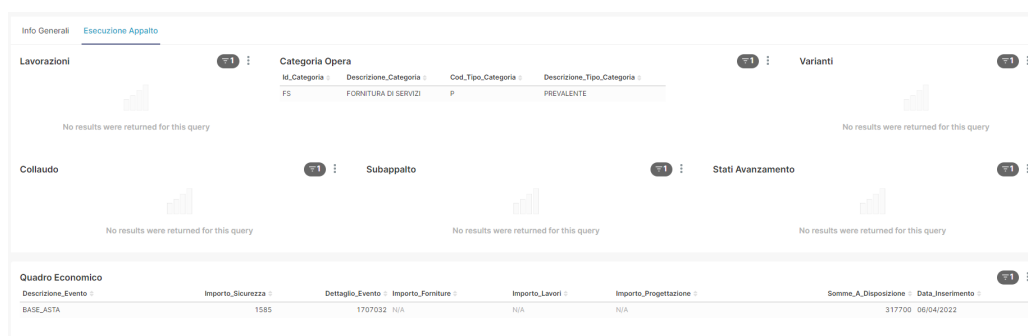


Figura 4.9 Dashboard dettagli bando

## 5. Conclusioni

Il progetto SIAP-PAI ha avuto come obiettivo quello di fornire una soluzione a supporto dei processi decisionali e gestionale-amministrativi nell'ambito dei bandi di gara pubblici, attraverso lo sviluppo del sistema di supporto alle decisioni illustrato nel presente lavoro di tesi. In particolare, il sistema di supporto alle decisioni, basato sul Semantic Framework, offre all'utente la possibilità di acquisire conoscenza nel dominio dei bandi pubblici della Regione Puglia attraverso l'utilizzo del motore di ricerca semantico e delle informazioni strutturate sul dominio integrate nei servizi del sistema.

Lo sviluppo della funzionalità di ricerca semantica come servizio web, nonché la cura dell'interfaccia utente del servizio, ha assunto un ruolo importante nel rendere tale servizio accessibile agli utenti in modo più semplice. Le funzionalità di ritrovamento di concetti simili a quelli presenti nella query dell'utente e la possibilità di esplorare le collezioni attraverso le ricerche svolte a partire dai concetti ritrovati, può supportare notevolmente l'utente nel soddisfare il bisogno informativo espresso tramite una query esplorando le connessioni semantiche presenti in una collezione. La rappresentazione semantica dei testi è stata, inoltre, ultimamente sfruttata nell'implementazione del servizio di *"Bandi simili"*, che permette all'utente di visionare bandi e documenti con contenuto molto simile a quello di interesse. In aggiunta, le funzionalità legate all'applicazione di filtri sulla tipologia di ricerca e al numero di risultati da visualizzare, aiutano l'utente a individuare con facilità e rapidità i risultati più consoni alle proprie esigenze.

Infine, l'integrazione di dati non strutturati relativi al dominio delle collezioni, organizzati in una base di dati relazionale, ha ulteriormente arricchito la disponibilità di informazioni a disposizione dell'utente del

sistema, pur mantenendo flessibile rispetto alle sue esigenze la quantità e il tipo di informazioni di volta in volta mostrate, grazie alla progettazione di una interfaccia utente attenta alle esigenze di usabilità del software.

Gli sviluppi futuri del progetto di tesi sono molteplici, sia dal punto di vista del miglioramento tecnico delle funzionalità del Semantic Framework, che da quello di possibili ulteriori scenari nei quali il sistema potrebbe essere impiegato.

Occorre, innanzitutto, evidenziare che tanto la piattaforma “Semantic Framework”, quanto i servizi del sistema di supporto alle decisioni, offrono funzionalità declinabili alle esigenze di una qualsiasi organizzazione e a qualsiasi dominio applicativo. Le funzionalità di gestione di collezioni di documenti e di analisi semantica del testo offerte dal Semantic Framework sono adatte all’implementazione di molti servizi che necessitino di un’analisi di testi approfondita. Sono servizi che possono essere molto significativi in un sistema di supporto alle decisioni basato su dati testuali sviluppato per un qualsiasi dominio.

È opportuno, tuttavia, che in futuro il sistema sia integrato in modo più efficiente con le fonti documentali e informative relative al dominio dei bandi pubblici. Oltre a semplificare il processo di ritrovamento e aggiunta dei documenti alle collezioni, e di inserimento dei metadati nella base di dati, il sistema potrebbe anche beneficiare della presenza di una maggiore quantità di documenti, che renderebbero le operazioni di analisi dei documenti ancor più significative.

Dal punto di vista dei servizi esposti dal sistema di supporto alle decisioni, è possibile implementare alcuni miglioramenti tecnici. Nel motore di ricerca semantico, ad esempio, può essere integrata una funzione di relevance feedback che permetta all’utente di notificare al sistema quando un documento non è rilevante rispetto alla query che ha svolto; ciò porterebbe al raffinamento della funzione di scoring del motore di ricerca e, dunque, ad ottenere dei risultati di ricerca maggiormente rilevanti.

Inoltre, il modulo di gestione dei metadati potrebbe essere reso indipendente dal dominio, in modo tale da poter gestire le informazioni strutturate attinenti alle collezioni presenti nel Semantic Framework indipendentemente dalla struttura della base di dati nella quale sono memorizzate.

Ulteriormente, in futuro si potrebbe implementare la funzionalità legata alla gestione degli utenti. Essa permetterebbe di creare e gestire un account personale, permettendo all'utente di gestire e salvare le ricerche preferite. Infine, per quanto riguarda l'applicazione web, si potrebbe ottimizzare meglio il codice, utilizzando unicamente gli strumenti messi a disposizione da AngularJS, e la parte sviluppata in Java, per le chiamate ai servizi REST.

## Riferimenti bibliografici e sitografici

[1] Porter, M. F., "An algorithm for suffix stripping", Program, Vol. 14 Issue: 3, pp.130-137, 1980

[2] Landauer, T. K. and Dumais, S. T., "A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge.", Psychological review, vol. 104, no. 2, pp. 211, 1997

[3] Sito di riferimento: <https://tika.apache.org/>

[4] Sito di riferimento: <https://lucene.apache.org/core/>

[5] Luhn, H. P., "A Statistical Approach to Mechanized Encoding and Searching of Literary Information", IBM Journal of Research and Development. 1 (4): 309–317, 1957

[6] Spärck Jones, K. "A Statistical Interpretation of Term Specificity and Its Application in Retrieval". Journal of Documentation. 28: 11–21, 1972

[7] Ramos, J. et al., "Using tf-idf to determine word relevance in document queries," in Proceedings of the first instructional conference on machine learning, 2003

[8] Salton, G., Wong A., Yang C. S., "A vector space model for automatic indexing", in Communications of the ACM, Volume 18 Issue 11, pp. 613-620, Nov. 1975



- [9] Harris, Z., "Distributional structure", Word. 10 (23), pp. 146–162, 1954
- [10] Johnson, W. and Lindenstrauss, J., "Extensions of Lipschitz mappings into a Hilbert space, in Contemporary Mathematics", American Mathematical Society, vol. 26, pp. 189–206, 1984
- [11] Sahlgren M., "An Introduction to Random Indexing," in Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE, vol. 5, 2005
- [12] Fielding R. T., "Architectural Styles and the Design of Network-based Software Architectures", 2000
- [13] Massè, M., "REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces", O'Reilly, 2012
- [14] Sito di riferimento: <https://newsapi.org/>
- [15] Sito di riferimento: <https://oros-git.regione.puglia.it/api-puglia/docs/-/blob/master/RegistroImprese/index.md>
- [16] C. Bizer, T. Heath, e T. Berners-Lee, «Linked data-the story so far», Semantic Serv. Interoperability Web Appl. Emerg. Concepts, pagg. 205–227, 2009

