UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

FACOLTÀ DI SCIENZE FISICHE, INFORMATICHE E
MATEMATICHE

LAUREA TRIENNALE
in Informatica

Realizzazione di una Funzionalità di Ricerca Testuale in un Sito Web Dinamico in Tecnologia Lasso

Candidata:
Rupinder Kaur

Relatore:
Prof. Riccardo Martoglia

Anno Accademico 2017/2018

Parole Chiavi:

1. Sito Web
2. Funzione di Ricerca
3. Lasso
4. Filemaker
5. Ricerca Testuale

Ringraziamenti

- *Ringrazio il mio relatore prof. Riccardo Martoglia per avermi dato la disponibilità e i consigli utili per la stesura della tesi, anche nell'ultimo momento.*
- *Ringrazio i miei genitori, la mia sorella, mio marito che mi hanno dato il supporto emotivo per superare l'ansia pre-laurea.*
  Ringrazio soprattutto i miei genitori che hanno dovuto superare più ostacoli di me per farmi arrivare a questo punto dove sono adesso.

Indice

INTRODUZIONE **1**

CAPITOLO 1: TEORIA **5**

1.1 TEORIA REALIZZAZIONE DEL SITO WEB 5

1.2 MODELLO CLIENT-SERVER 6

1.3 MOTORE DI RICERCA 8

1.4 MOTORI DI RICERCA GIÀ DISPONIBILI 10

1.4.1 Ricerca Personalizza di Google 10

1.4.2 Ricerca Personalizzata di Microsoft: Bing 11

1.5 MOTORI DI RICERCA SVILUPPATI AD HOC 12

1.6 TIPI DI RICERCA 14

1.6.1 Ricerca Full-Text 17

CAPITOLO 2: TECNOLOGIE UTILIZZATE **19**

2.1 DBMS 19

2.1.1 Tipi di Database 22

2.1.2 Cosa fa un DBMS 24

2.2 WEB SERVER 24

2.2.1 IIS: Internet Information Services 27

2.2.2 Apache HTTP Server 27

2.3 SERVER SIDE IMPLEMENTATION 28

2.3.1 Lasso 30

2.3.1.1 Le Caratteristiche del Linguaggio 32

2.3.1.2 Lasso Runtime 35

2.4 ALTRE TECNOLOGIE ACQUISITE 35

2.4.1 HTML 35

2.4.2 SQL 38

2.4.3 JQuery / Ajax 40

CAPITOLO 3: PROGETTAZIONE **43**

3.1 BREVE INTRODUZIONE 43

3.2 CONFIGURAZIONI UTILI 44

3.3 SPECIFICA DELL'AZIENDA 47

3.4 PROGETTAZIONE 48

CAPITOLO 4: IMPLEMENTAZIONE **50**

4.1 DB CONNECTION 50

4.2 ESTRAZIONE DATI DAL DATABASE 52

4.3 GRAFICA 55

4.3.1 Visualizzazione Risultati 56

CONCLUSIONI **61**

BIBLIOGRAFIA E SITOGRAFIA **62**

Introduzione

Gli avanzamenti nella tecnologia della rete Internet e la digitalizzazione dell'informazione hanno facilitato la nostra vita, rendendo reperibile la maggior parte dei prodotti del nostro bisogno in un *click*. Ha completamente cambiato il nostro modo di vivere diventando la fonte primaria di informazioni e rivoluzionando il modo in cui le imprese operano e comunicano con i loro clienti.

Per acquisire nuovi clienti è necessario essere visibili sui mezzi che le persone utilizzano per reperire le informazioni sui prodotti/servizi che intendono acquistare: nel mondo di oggi la maggioranza delle persone si rivolge a Internet, in particolare ai Motori di Ricerca come Google, Yahoo o Bing. Il 93% delle decisioni d'acquisto inizia con una ricerca su un Motore di Ricerca, il 60% di tutte le ricerche effettuate è svolto tramite Google, il motore di ricerca più usato al mondo!

Per essere visibili nel mondo online, le aziende si dispongono dei siti web sempre più efficienti, in modo da garantire un servizio soddisfacente alla propria clientela. Il sito web è il contenitore di tutte le informazioni che un'azienda vuole fornire al pubblico generico. Per le aziende di medio-grande dimensioni, questo contenitore è talmente vasto che non può contenere tutte le informazioni in una singola pagina. Questo è il motivo per cui le aziende si optano per i siti web multi-pagine attraverso dei collegamenti ipertestuali inseriti nelle pagine.

Il lavoro svolto in questa tesi riguarda al tirocinio svolto nell'azienda Multitraccia sc. Il progetto formativo proposto dall'azienda era nell'implementazione di uno script di ricerca nel linguaggio *Lasso,* analizzando la struttura delle pagine web e il modo con cui quest'ultime sono state salvate nel database creato con la tecnologia *Filemaker*. L'azienda si opera nel settore informatico, offrendo i servizi di tipo: creazione dei software, assistenza software, Desktop Publishing al Multimedia, Networking e Internet Solutions.

Il sito web di Multitraccia segue la tecnica multi-pagine, le cui pagine sono raggiungibili attraverso degli appositi link inseriti nella pagina home. All'inizio del tirocinio, il sito web dell'azienda si disponeva di una casella di ricerca che però era stata disattivata, per cui non forniva alcun risultato all'utente. L'obiettivo finale di questo tirocinio era di far funzionare questa casella di ricerca, eseguendo la ricerca in tutte le pagine del sito e di mettere in risalto la sezione[1] della pagina home in cui è stato trovato la parola cercata attraverso dei link ancóra.

In questa tesi cercheremo di mettere in risalto l'importanza della funzione di ricerca e l'implementazione di essa in linguaggio Lasso: linguaggio con cui è stato implementato il sito web di Multitraccia.

La tesi è strutturata in due parti, ogn'una di esse contiene due capitoli:

- Parte Teorica:
    - Capitolo 1: In questo capitolo si cerca di definire l'importanza della funzione di ricerca, l'usabilità del modello Client-Server, le modalità di ricerca disponibili ecc.
    - Capitolo 2: In questo capitolo si definiscono a livello teorico tutte le tecnologie utilizzate in questo progetto.

[1] Le pagine del sito web sono suddivise in 5 sezioni principali: Menu, Servizi, Progetti, Chi Samo e Contatti

- Parte della Realizzazione:
    - Capitolo 3: In questo capitolo si affrontano la progettazione del problema e tutte le scelte effettuate.
    - Capitolo 4: In questo capitolo è stato implementato quanto scritto nel capitolo precedente della progettazione.

Parte I

Parte Teorica

Capitolo 1: Teoria

In questo capitolo viene introdotta brevemente la storia della nascita dei siti web, il modello Client-Server su cui si basa il concetto di ricerca, i vari strumenti di ricerca già esistenti pronti ad essere integrati nel sito web invece di implementare uno script di ricerca e i vantaggi dell'implementazione del proprio script di ricerca ecc..

1.1 Teoria Realizzazione del Sito Web

Anche se la storia di Internet va indietro agli anni '50 quando i primi progetti di Internet furono presentati, la vera rivoluzione dell'Internet ha raggiunto il boom dopo il lancio di *www*[2]. Il www o w3 è uno dei principali servizi di Internet che permette agli utenti consultare un insieme di contenuti connessi fra loro tramite i link. Tutto questo è possibile oltre che dall'uso efficiente dei Protocolli di rete, anche dalla facilità d'uso e l'efficienza dei **motori di ricerca** del Web Browser**.**

[2] www: World Wide Web annunciato da CERN nel 1991

I contenuti principali del Web sono costituiti da testo e grafica, seguendo le regole definite dal W3C[3]. I contenuti del web sono un insieme di siti web che a loro volta contengono un insieme di pagine web raggiungibili mediante gli appositi link.

Strutturalmente i siti web sono categorizzati come:

- **Siti web Statici**: contengono le pagine web statiche. Ogni aggiornamento del sito web va fatta a mano modificando direttamente il codice HTML del sito.
- **Siti web Dinamici**: sono formati da pagine web dinamiche contenente le informazioni che possono variare nel tempo, dando un aspetto più realistico della realtà attuale.

La dinamicità delle pagine web richiede un'alta interazione fra l'utente e il sito. Un esempio più comune di questa interazione è quando il sito web si relaziona con il Database da cui prende i contenuti, e in base alla variazione del database varia anche l'aspetto del sito web.

La ricerca web multi-pagine implementata nel nostro caso presso l'azienda Multitraccia si basa su un sito web dinamico di cui i contenuti sono ospitati sul database realizzato con il software Filemaker.

1.2 Modello Client-Server

Prima di analizzare la ricerca nel sito web in questione, è opportuno conoscere l'importanza della ricerca non limitando solo in una risorsa del web ma nel contesto generale, il modo in cui viene effettuato, i motori di ricerca disponibili ecc.

[3] W3C: Il World Wide Web Consortium è un'organizzazione non governativa internazionale che ha come scopo quello di sviluppare tutte le potenzialità del World Wide Web.

L'informazione ospitata su internet è talmente vasta che per fare una ricerca del contenuto d'interesse si può perdere una svariata quantità di tempo. Per questo motivo i motori di ricerca che conosciamo, utilizzano gli algoritmi di ricerca sempre più efficienti, in modo da visualizzare il contenuto d'interesse all'utente finale in minor tempo possibile. Gli algoritmi di ricerca decidono i criteri d'importanza con cui vengono presentati i risultati all'utente finale.

Fare una ricerca di uno specifico sito web d'interesse non è altro che un'azione di ricerca nel complesso dei dati ospitati nel gigante database di Internet. Lo strumento principale da cui parte l'utente è un software installato sulla propria macchina che offre un'interfaccia grafica per creare un'interazione fra l'utente e internet. Il termine tecnico di questo software è il *"Web Browser"*. Per citarne alcuni esempi di browser più comuni si può parlare di Chrome, Internet Explorer, Firefox ecc. I Browser Web hanno il compito di implementare le funzionalità non solo dal lato client per il protocollo HTTP[4] in modo da scaricare dalla rete locale le risorse (contenuto ipertestuale) a partire dal loro URL, ma si occupano anche della visualizzazione di tali risorse che sono in sotto forma di pagine HTML.

Nell'architettura di rete Client-Server di Internet, web browser rappresenta il client che fa le richieste di risorse web ai vari Web-Server e Application Server, che ospitano rispettivamente siti web e applicazioni web. Un web browser si appoggia sempre a un motore di ricerca per raggiungere i siti web; una volta che quest'ultimo ha dato in output i risultati desiderati, cliccando su URL[5] desiderato viene eseguita in background una risoluzione tra URL e IP. La risoluzione tra URL e IP avviene attraverso l'interrogazione di vari DNS[6] database. Una volta che si conosce l'IP del server di destinazione della risorsa, partono una serie di *request* e *response* fra il client e server per ottenere i dati richiesti.

---

[4] HTTP: Hyper Text Protocol, il protocollo ISO a livello applicativo utilizzato dall'Internet per la visualizzazione delle pagine web

[5] URL: Uniform Resource Locator, serve per localizzare la risorsa web nell'Internet

[6] DNS: Domain Name System, sistema di trasformazione dei IP numerici in parole comuni riconoscibili in linguaggio comune e viceversa.

1.3 Motore di Ricerca

Il motore di ricerca è lo strumento fondamentale su cui si basano i web-browser per ricercare i dati richiesti nella rete internet. Il motore di ricerca è un grande sistema informatico che esplora metodicamente ed automaticamente tutte le parti accessibili di internet e ne cataloga ed archivia il contenuto. Questo catalogo e questo archivio vengono aggiornati costantemente e sono consultabili dagli utenti comuni visitando il sito del motore di ricerca.

Ogni motore di ricerca è composto da due elementi fondamentali: il *"crawler" o "spider"* che è il software esploratore e gli *algoritmi di ricerca,* che decidono i criteri d'importanza con i quali i dati vengono presentati all'utente.

Crawler è un software che analizza i contenuti della rete per conto di motore di ricerca. È un tipo di bot (programma o script che automatizza delle operazioni). Nell'ambito di WEB, il Crawler si basa su una lista di URL da visitare fornita dal motore di ricerca. Durante l'analisi di un URL, identifica tutti i collegamenti ipertestuali presenti nel documento e li aggiunge alla lista di URL da visitare.

L'algoritmo, invece, non ha un nome (tranne che nel caso di Google che lo chiama Page Rank, perché denomina il rango delle pagine web). Le tipologie degli algoritmi di ricerca sono varie: *ricerca sequenziale, ricerca hash, ricerca ricorsiva, Ricerca ricorsiva su alberi* e tante altre, però il modo in cui questi algoritmi vengono utilizzati da un motore di ricerca è tale da garantirne l'efficienza del processo completo della ricerca.

Le fasi di ricerca di un motore di ricerca si categorizzano in:

- Analisi del campo d'azione (tramite gli appositi Crawler)
- Catalogazione dei risultati ottenuti
- Risposta alle esigenze dell'utente

I motori di ricerca più diffusi sono Google, Yahoo!, Microsoft Bing, Baidu, Ask.com. I risultati presentati dai motori ricerca non sono altro che una serie di link che portano alla risorsa ricercata dall'utente.

Avendo parlato di come funziona una ricerca dal motore di ricerca, si può analizzare il caso specifico della ricerca in un sito web. Il sito web pur essendo una piccola parte nel mondo dell'internet, può essere considerato un mondo intero a sé stante. I siti web possono essere molto complessi strutturalmente, non è detto che l'informazione ricercata dall'utente ci sia sempre nella stessa pagina poiché essi contengono al loro interno i link verso altre pagine dello stesso sito o verso le pagine anche di altri siti esterni.

Per comprendere l'enormità dei dati e dei link contenuti nel sito web, si riporta nella figura 1 il Sitemap del sito web Multitraccia creato con il tool di informatica reperibile al sito web www.xml-sitemaps.com. Si può notare che pur essendo un'azienda non molto grande, contiene un'enorme quantità di link che portano all'utente verso altre pagine dello stesso sito.

Fare una ricerca per il contenuto d'interesse in questi tipi di siti web non è immediato ed è molto complesso non sapendo la localizzazione della pagina web in cui si risiede la risorsa ricercata.


```
▼<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.sitemaps.org/schemas/sitemap/0.9
 http://www.sitemaps.org/schemas/sitemap/0.9/sitemap.xsd">
 ▼<!--
     created with Free Online Sitemap Generator www.xml-sitemaps.com
  -->
 ▼<url>
    <loc>http://www.multitraccia.it/</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>1.00</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/index.lasso?lk=3</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/index.lasso</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/contatti.lasso?lk=2</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/index.lasso?lk=2</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/servizi.lasso?lk=2&sv=1</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/servizi.lasso?lk=2&sv=2</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
    <priority>0.80</priority>
  </url>
 ▼<url>
    <loc>http://www.multitraccia.it/servizi.lasso?lk=2&sv=3</loc>
    <lastmod>2019-03-02T17:38:08+00:00</lastmod>
```


Figura 1: Sitemap Multitraccia

Visto la mole dell'informazione contenuta all'interno dei siti web, viene predisposto una funzione di ricerca anche all'interno del sito web. Questo motore di ricerca si limita a fare la ricerca solamente nei contenuti del sito web in questione. In questo modo il Crawler effettua l'analisi del campo d'azione solo su quel determinato sito web, vengono catalogate le informazioni solo del sito web in questione e il risultato finale presentato all'utente si limita al sito web che sta visitando in quel momento.

I motori di ricerca utilizzati nei siti possono essere suddivisi in due macro-categorie:

- Motori di Ricerca già disponibili
- Motori di Ricerca Personalizzati

1.4 Motori di Ricerca già disponibili

1.4.1 Ricerca Personalizza di Google

Questo motore di ricerca si basa sulla tecnologia di ricerca di base di Google. La differenza fondamentale tra il motore di ricerca Google e motore di ricerca Google personalizzato è che quest'ultima mette in evidenza solamente i risultati riguardanti al sito web su cui è impostato.

I passi da seguire per includere la ricerca personalizzata nel proprio sito web sono molto semplici: al momento della configurazione viene chiesto di scegliere i siti web da includere nel motore di ricerca. Questa funzionalità consente allo sviluppatore di inserire una serie di URL appartenenti al dominio del sito da prendere in considerazione e di limitarne la funzione di ricerca di Google solo su quel dominio. Si possono anche escludere determinate URL dal motore di ricerca, in modo tale che Google non vada a interrogare le pagine web escluse dalla funzione di ricerca.

Durante la fase di creazione si possono fare tutte le scelte di *look&feel* del motore di ricerca che verrà poi incorporata nel sito web ad esempio la scelta del layout della casella di testo oppure attivazione/disattivazione della funzione auto-completamento ecc.

Una volta completata la fase di creazione, si può generare lo *snippet* del codice HTML da copiare e incollare nel punto del codice sorgente HTML del sito in cui si vuole che compaia il motore di ricerca

Affinché i contenuti del sito vengano inclusi nei risultati del motore di ricerca personalizzato, devono essere presenti nell'indice Google. L'indice di Google elenca tutte le pagine web conosciute dal Google, e quando Google visita un nuovo sito, rileva nuove pagine e aggiorna il proprio indice. Per far includere le pagine web o l'intero sito web nell'indice di Google, occorre mandare la richiesta di indicizzazione tramite il Google Search Console. Google visita un sito secondo l'importanza che ha per il Google. Per calcolare quanto sia importante un sito, Google conta quanti altri siti hanno un collegamento a quello; maggiore è tale numero, maggiore sarà l'importanza di quel sito per Google. Se nessun sito ha il collegamento verso il sito web da indicizzare, la probabilità che il Google prenda in considerazione le pagine web di quel sito diminuisce drasticamente. Per questo motivo la scansione e l'indicizzazione del sito web potrebbero non essere eseguite immediatamente.

1.4.2 Ricerca Personalizzata di Microsoft: Bing

La ricerca personalizzata di Bing funziona con le stesse modalità della ricerca personalizzata di Google con unica differenza che la versione di Bing è ad-free mentre la versione gratuita di Google Custom Search ha il requisito di avere un account Google AdSense con il quale si inseriscono gli annunci pubblicitari nelle pagine dei risultati forniti dal motore di ricerca.

Uno dei motivi per cui è stato analizzato il motore di ricerca Google nella sua versione personalizzata è perché, in termini di quota del mercato, Google sovrasta di gran lunga tutti gli altri motori di ricerca (figura 2) ed offre un maggior numero di opzioni di ricerca selettiva.

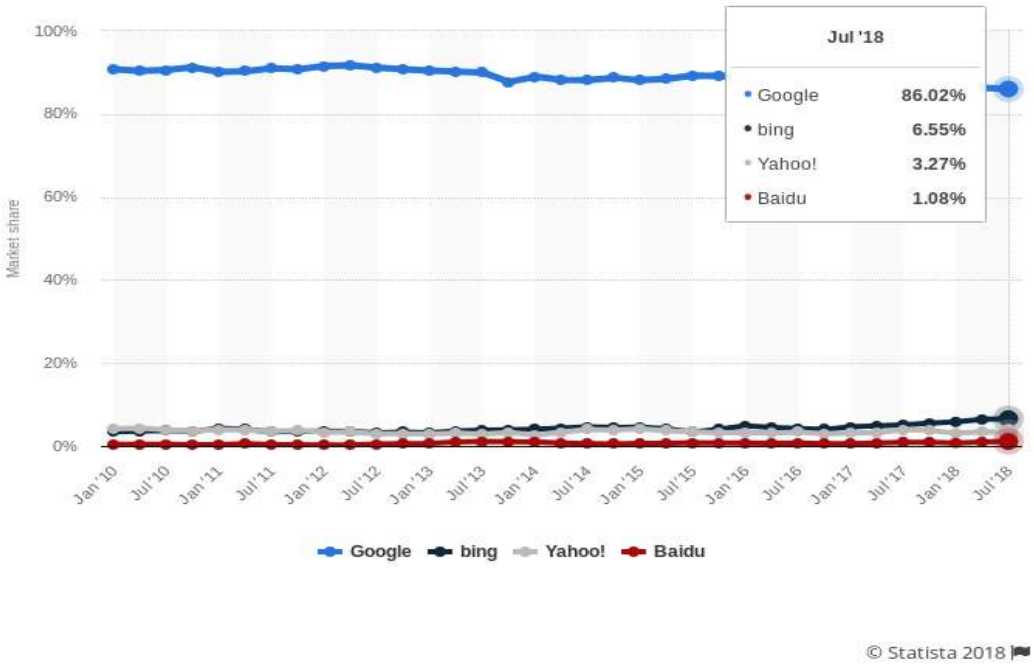


Figura 2: Statistica 2018 Quota Mercato Google

1.5 Motori di Ricerca Sviluppati ad Hoc

Il motore di ricerca personalizzato analizzato nel paragrafo precedente è la soluzione più facile da intraprendere, sia dal punto di vista economico che dal punto di vista dello sviluppo, però non è l'unica. Quando tutto il sito web dinamico di grande dimensioni viene sviluppato scegliendo un linguaggio di programmazione tra quelli disponibili, è consigliato avere anche lo script di ricerca nello stesso linguaggio, per mantenere la regolarità nella progettazione.

Integrazione del motore di ricerca può essere implementata attraverso degli script, che possono essere scritti in vari linguaggi (PERL, CGI, PHP, LASSO ...). Per poter integrare uno script nel sito, è necessario che questo risieda su un server che supporti l'esecuzione di tale script. I servizi di hosting professionale, che offrono il supporto a tecnologie dinamiche come PERL, PHP, SQL, LASSO, infatti, prevedono una quota mensile o annuale non certo trascurabile. Per questo motivo molti blog o i siti di dimensione piccole optano la prima soluzione ovvero quella di usufruire una tecnologia già implementata che è economicamente più vantaggiosa.

I siti web commerciali per cui esiste già la licenza di hosting sul server, possono usufruire dello script implementato con i linguaggi di programmazione server-side per personalizzare la funzione di ricerca a seconda delle esigenze del contesto. Essendo il dominio di ricerca molto ristretto, lo sviluppatore ha la maggiore flessibilità nella manipolazione dei dati, nell'ottimizzazione della funzione di ricerca, nella creazione di un motore di ricerca che risponde limitatamente alle esigenze del settore applicativo.

Se da un canto Scripting della funzione di ricerca da maggiore libertà agli sviluppatori, dall'altro canto porta anche degli aspetti negativi perché può nascondere una serie di errori facili da commettere, per esempio se uno script riceve un sovraccarico dei dati sempre maggiore dal sito allora l'host limita i tempi d'esecuzione dello script. Gli script programmati possono essere anche rischiosi per la sicurezza dei dati. Per non affaticare ulteriormente l'Host della propria pagina con una funzione di ricerca e per garantire un funzionamento stabile, i gestori di siti web ricorrono pertanto in modo crescente a fornitori esterni.

1.6 Tipi di Ricerca

Nelle scienze informatiche, l'algoritmica è la parte fondamentale della programmazione e soprattutto è la base della nozione teorica della calcolabilità di un problema. Gli algoritmi sono i passi elementari di ogni programma o di un'applicazione del computer. La nozione di ricerca in qualsiasi insieme di elementi non è nuova per la scienza. Gli algoritmi di ricerca permettono di trovare un elemento avente determinate caratteristiche all'interno di un insieme di dati. Gli elementi dell'insieme sono caratterizzati da una chiave univoca o multipla che consente agli algoritmi di ricerca di individuare l'elemento all'interno dell'universo degli insiemi. Gli algoritmi di ricerca si distinguono in base alla tecnica di ricerca utilizzata (ricerca ricorsiva, ricerca sequenziale, ecc.…)

Si possono elencare numerose tipologie degli algoritmi di ricerca:

- Algoritmi di Ricerca Sequenziale
- Algoritmi di Ricerca Hash
- Algoritmi di Ricerca Ricorsiva
- Algoritmi di Ricerca su alberi
- Algoritmi di String Matching
- Ricerca Full-Text

Nel contesto della funzione di ricerca dei siti web, si può analizzare Ricerca Full-Text per siti web dinamici che relazionano con i DBMS. La ricerca Full-Text è una delle tecniche dell'Information Retrieval (IR). È l'insieme delle tecniche e tecnologie informatiche che studia l'interrogazione dei database, l'organizzazione dei dati e il reperimento delle informazioni (ricerca dei dati) in un dispositivo di memorizzazione.

Lo scopo dell'Information Retrieval consiste nella soddisfazione dell'esigenza informativa dell'utente, garantendo a quest'ultimo un elenco di risorse attinenti e rilevanti (risultato), sulla base della richiesta (interrogazione) da lui effettuata. Un algoritmo IR è l'offerta di informazioni a una specifica domanda dell'utente. Lo schema del processo di ricerca delle informazioni è mostrato nella figura 3:

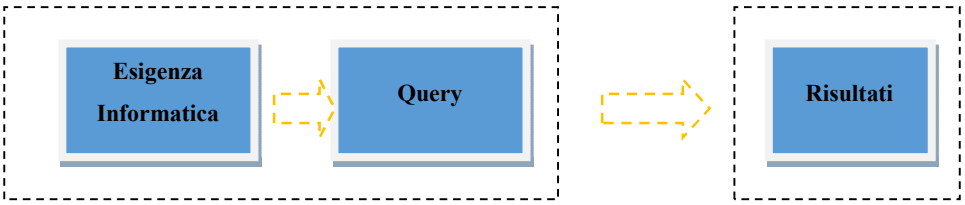


Figura 3: Schema processo di ricerca

L'esigenza informativa nasce da una situazione di deficit di informazioni dell'utente su un determinato argomento, questo stato è detto ASK (Anomalous State of Knowledge), che lo spinge ad effettuare una richiesta di informazioni (domanda di informazioni). A partire dall'esigenza informativa dell'utente, l'algoritmo ricerca le risorse informative più attinenti (resource finding) e le ordina per rilevanza, fornendo un elenco dei risultati (offerta di informazioni). Per ogni risorsa nell'elenco è indicato il relativo indirizzo di reperimento della stessa. Ad esempio, in un algoritmo di ricerca sul web ogni risorsa informativa è associata a un indirizzo URL (Uniform Resource Locator) che la identifica in modo univoco sulla rete. Le risorse informative nella lista dei risultati devono essere attinenti e rilevanti, rispetto all'esigenza informativa dell'utente.

- **Attinenza (aboutness):** La risorsa deve essere attinente all'argomento richiesto dall'utente. Un algoritmo di ricerca IR è efficace e utile soltanto se fornisce risultati attinenti.

- **Rilevanza:** La risorsa deve essere rilevante, ossia deve colmare il bisogno informativo dell'utente. La rilevanza è l'utilità, reale o presunta, che l'utente trae dalla risorsa nell'elenco dei risultati della ricerca.

L'attinenza e la rilevanza sono le qualità fondamentali dei risultati di ricerca. Una risorsa attinente ma poco rilevante è poco utile all'utente finale. D'altra parte, una risorsa rilevante su un argomento ma poco attinente all'esigenza informativa dell'utente è altrettanto inutile. Nella figura 4 si riporta un grafico che ne evidenzia la collocazione dei risultati ottenuti misurando l'attinenza e la rilevanza dei risultati di ricerca

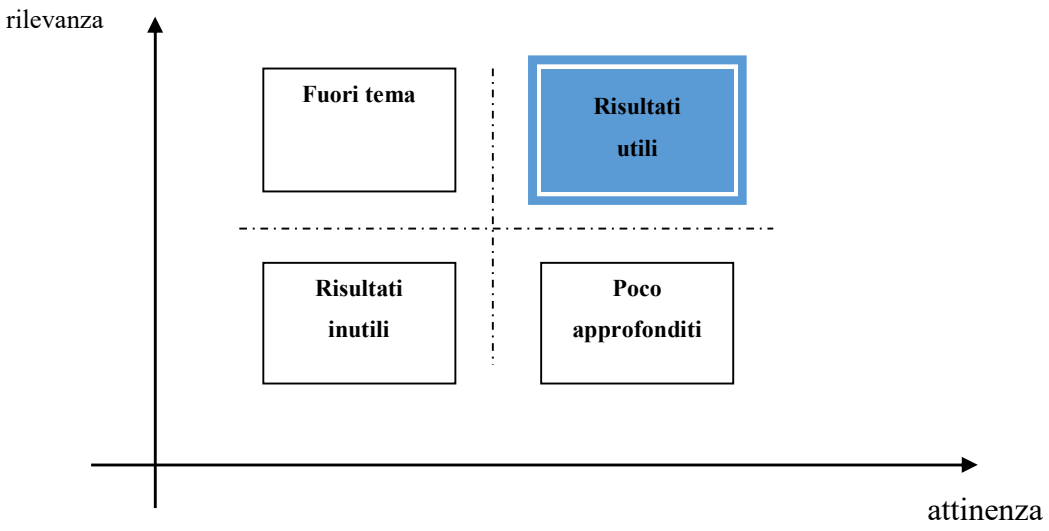


Figura 4: relazione rilevanza e attinenza dei risultati di ricerca

L'obiettivo di un sistema IR è offrire un elenco di risultati utili (rilevanti e attinenti). Tuttavia, non esiste un criterio oggettivo per misurare l'efficacia dell'utente poiché la soddisfazione dell'utente è un elemento soggettivo del fenomeno. È possibile misurare il grado di attinenza di una risorsa, in relazione alla richiesta, ma non è sempre possibile valutare la rilevanza poiché quest'ultima è determinata dal contesto in cui si trova l'utente e da una serie di fattori soggettivi. La stessa interruzione della ricerca non è sempre un indicatore della qualità informativa. Generalmente, l'utente conclude l'attività di ricerca quando trova una risorsa sufficientemente utile oppure quando il processo diventa troppo lungo e si convince di non poterla più trovare.

1.6.1 Ricerca Full-Text

La ricerca Full-Text è la caratteristica fondamentale di quasi tutti i database relazionali. La caratteristica speciale di queste ricerche è che non si limitano a rintracciare le occorrenze di una parola o di una frase al pari di quello che potrebbe fare l'operatore *LIKE*, ma individuano le righe in cui il campo o i campi sottoposti alla ricerca mostrano un'attinenza particolare con il pattern indicato.

Di seguito viene illustrata la ricerca full-text offerta dal database MY-SQL:

Le ricerche di tipo Full-Text possono essere eseguite su uno o più campi ma richiedono la presenza di un *Full-Text Index* che li coinvolga tutti.

La definizione seguente crea una tabella con un indice *Full-Text* applicato a un campo solo:

```
CREATE TABLE Blog
(
        id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
        title VARCHAR (50),
        body TEXT,
        FULLTEXT (body)
)
```

Come si vede, è sufficiente indicare la parola chiave FULLTEXT ed elencare, tra parentesi tonde, i campi coinvolti.

È bene comunque ricordare che tali indici possono essere creati solo su campi di testo (VARCHAR, TEXT, CHAR) e con gli Storage Engine MyISAM e InnoDB. In particolare, su quest'ultimo motore sono permessi solo a partire dalla versione 5.6 del DBMS.

La **ricerca** potrà essere effettuata usando il costrutto MATCH ... AGAINST:

```
SELECT * FROM Blog WHERE MATCH (body) AGAINST ('database');
```

L'esempio di query sopra riportato effettuerà una ricerca sul campo body della tabella Blog usando il termine database come pattern.

Nella clausola MATCH possono essere inclusi più campi, ma è fondamentale che tutti facciano parte dell'indice FULLTEXT creato. È altrettanto essenziale che, se l'indice include più campi, tutti vengano nominati nella clausola MATCH

Capitolo 2: Tecnologie Utilizzate

In questo capitolo vengono discusse a livello teorico le tecnologie utilizzate nell'implementazione della funzione di ricerca. In particolare vengono analizzate le caratteristiche dei linguaggi di programmazione server-side, le caratteristiche del web server e il loro ruolo all'interno dei siti web, e le funzionalità del DBMS.

2.1 DBMS

Nei paragrafi precedenti è stato discusso l'importanza del motore di ricerca, il modello Client-Server, un esempio di ricerca utilizzata nel caso in cui i dati da cercare sono memorizzati in un database.

Il world wide web è forse una delle grandi fonti di informazioni a cui ogni uno di noi si attinge. Per i siti web di piccole dimensioni dove la struttura gerarchica del sito non risulta molto complessa, possono contenere le risorse statiche perché la gestione di esse non risulta molto difficile. Per i siti web che sono strutturalmente molto complessi (basta pensare a sito web e-commerce di cui i contenuti cambiano ad ogni operazione di compra-vendita) e richiedono una manutenzione flessibile e dinamica nel tempo, è necessario che abbiano un collegamento con i dispositivi di memorizzazione dei dati organizzati come gli strumenti database.

Per questo motivo, spesso queste pagine non sono documenti statici, ma vengono creati dinamicamente quando vengono richiesti e le informazioni che contengono vengono estratte da un database. Se il database è un database relazionale (vedremo in seguito cosa significa), il linguaggio utilizzato per recuperare le informazioni che ci vengono mostrate può variare a seconda del tipo di database utilizzato.

Le pagine del sito web analizzato nel nostro caso sono memorizzate nel database Filemaker. Il linguaggio di Scripting del Filemaker è proprio FM Script con le regole proprie di Filemaker molto simili a quelle di SQL.

Prima di occuparsi del linguaggio di Scripting FM/SQL e Lasso, occorre capire cosa è un "Database", tradotto in italiano "Base di Dati" e cosa è una DBMS (Database Management System).

Un DBMS è sostanzialmente uno strato software che si frappone fra l'utente e i dati veri e propri. Grazie a questo strato intermedio l'utente e le applicazioni non accedono ai dati così come sono memorizzati effettivamente, cioè alla loro rappresentazione fisica, ma ne vedono solamente una rappresentazione logica. Ciò permette un elevato grado di indipendenza fra le applicazioni e la memorizzazione fisica dei dati. L'amministratore del database, se ne sente la necessità, può decidere di memorizzare i dati in maniera differente o anche di cambiare il DBMS senza che le applicazioni, e quindi gli utenti, ne risentano. La cosa importante è che non venga cambiata la rappresentazione logica di quei dati, che è la sola cosa che i loro utilizzatori conoscono. Questa rappresentazione logica viene chiamata 'Schema del database ed è la forma di rappresentazione dei dati più a basso livello a cui un utente del database può accedere.

Ad esempio, in figura 5 è rappresentata una situazione in cui l'amministratore del database ha deciso che per motivi di efficienza era necessario cambiare il disco su cui erano memorizzati alcuni dati, partizionandoli inoltre su più dischi per permettere accessi paralleli a sottoinsiemi di dati indipendenti. Dal punto di vista dell'utente non è cambiato assolutamente nulla e probabilmente egli non è nemmeno a conoscenza dell'avvenuto cambiamento.

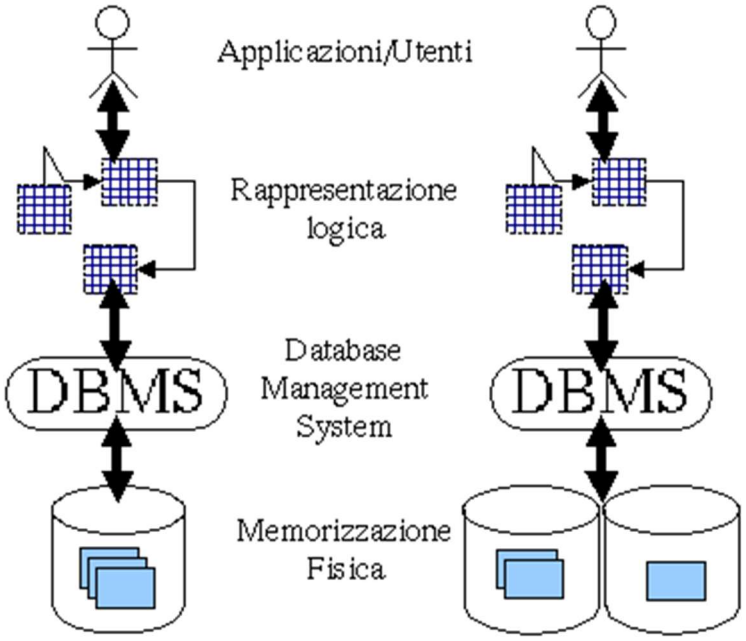


Figura 5: Cambiamento schema logico database

2.1.1 Tipi di Database

Nel corso degli anni sono stati adottati numerosi modelli per i dati, a fronte dei quali esistono quindi vari tipi di database. I tipi più comuni sono:

- **Database gerarchici**: i dati vengono organizzati in insiemi legati fra loro da relazioni di "possesso", in cui un insieme di dati può possedere altri insiemi di dati, ma un insieme può appartenere solo ad un altro insieme. La struttura risultante è un albero di insiemi di dati.

- **Database reticolari**: il modello reticolare è molto simile a quello gerarchico, e infatti nasce come estensione di quest'ultimo. Anche in questo modello, insiemi di dati sono legati da relazioni di possesso, ma ogni insieme di dati può appartenere a uno o più insiemi. La struttura risultante è una rete di insiemi di dati.

- **Database relazionali**: i database appartenenti a questa categoria si basano sul modello relazionale la cui struttura principale è la relazione, cioè una tabella bidimensionale composta da righe e colonne. Ciascuna riga, che in terminologia relazionale viene chiamata *Tupla*, rappresenta un'entità che noi vogliamo memorizzare nel database. Le caratteristiche di ciascuna entità sono definite invece dalle colonne delle relazioni, che vengono chiamate attributi. Le entità con caratteristiche comuni, cioè descritti dallo stesso insieme di attributi, faranno parte della stessa relazione.

- **Database ad oggetti** (object-oriented o ODBMS): lo schema di un database ad oggetti è rappresentato da un insieme di classi, che definiscono le caratteristiche ed il comportamento degli oggetti che popoleranno il database. La principale differenza con i modelli esaminati finora è la non passività dei dati. Infatti, con un database tradizionale

(intendendo con questo termine qualunque database non ad oggetti) le operazioni che devono essere effettuate sui dati vengono demandate alle applicazioni che li utilizzano. Con un database object-oriented, al contrario, gli oggetti memorizzati nel database contengono sia i dati che le operazioni possibili su tali dati. In un certo senso potremmo pensare agli oggetti come a dati a cui è stata fatta un'iniezione d'intelligenza, che gli permette di sapere come comportarsi, senza doversi appoggiare ad applicazioni esterne.

I primi due tipi di database, quelli gerarchici e reticolari, quasi appartengono ormai alla storia dell'informatica. La maggior parte dei database attualmente utilizzati appartiene alla categoria dei database relazionali. I motivi di questo successo (anche commerciale) vanno ricercati nella rigorosità matematica e nella potenzialità espressiva del modello relazionale su cui si basano, nella sua semplicità di utilizzo e, ultima ma non meno importante, nella disponibilità di un linguaggio d'interrogazione standard, SQL, che, almeno potenzialmente, permette di sviluppare applicazioni indipendenti dal particolare DBMS relazionale utilizzato.

I database ad oggetti sono la nuova frontiera nella ricerca sui database, infatti le loro caratteristiche di estendibilità, derivanti dalla possibilità di definire nuovi tipi di dati e comportamenti, li rendono particolarmente appetibili per tutte quelle applicazioni che richiedono dati complessi, come ad esempio immagini, suoni o coordinate. Purtroppo la mancanza di un modello per gli oggetti universalmente accettato e la non disponibilità di un linguaggio d'interrogazione standard fanno sì che ogni produttore implementi la propria visione specifica, di solito assolutamente incompatibile con tutte le altre. Di recente sono apparsi sul mercato alcuni database, definiti object-relational, che cercano di introdurre nel modello relazionale le caratteristiche di estendibilità proprie dei database object-oriented.

2.1.2 Cosa fa un DBMS

Indipendentemente dal tipo di database, le funzionalità principali che ci si deve aspettare da un DBMS sono quelle di:

- consentire l'accesso ai dati attraverso uno schema concettuale, invece che attraverso uno schema fisico;
- permettere la condivisione e l'integrazione dei dati fra applicazioni differenti;
- controllare l'accesso concorrente ai dati;
- assicurare la sicurezza e l'integrità dei dati.

Grazie a queste caratteristiche le applicazioni che vengono sviluppate possono contare su una sorgente dati sicura, affidabile e generalmente scalabile. Tali proprietà sono auspicabili per applicazioni che usano la rete Internet come infrastruttura e che hanno quindi evidenti problemi di sicurezza e scalabilità.

2.2 Web Server

Nel primo capitolo di questo documento è stato analizzato il modello Client-Server utilizzato dall'applicazione web per soddisfare le query dell'utente. Ripetendo brevemente quanto detto in precedenza, nel Web, la comunicazione (intesa come scambio di file) tra terminali diversi, avviene attraverso dei sistemi di trasmissione denominati protocolli, tra i quali il più conosciuto è il protocollo HTTP (Hyper-Text Transfer Protocol). Il funzionamento di questo protocollo prevede che l'utente richieda l'accesso a determinate risorse (input), attraverso la digitazione di URL o cliccando su link. Queste risorse gli vengono poi inviate (output), quando disponibili, generalmente sotto forma di pagine HTML o di immagini.

Con questa procedura, l'utente non fa altro che comunicare al browser, installato sul suo computer (client), il percorso che deve intraprendere per reperire le informazioni cercate, ospitate su di un computer differente (server). Con il termine specifico di ***Web Server***, quindi, parliamo di un computer che ha la funzione di ospitare e fornire siti internet formati sia da pagine statiche (contenenti solo codice HTML) che dinamiche (contenenti codice come il PHP e ASP).

La distinzione tra questi due tipi di pagine è molto importante per capire qual è la specifica funzione di un Web Server.

Una semplice pagina HTML (statica) non richiede particolari interventi da parte di un Web Server, in quanto il codice viene già interpretato dal browser installato sul computer client; il discorso invece cambia in presenza di codice PHP o ASP (pagine dinamiche) o LASSO (nel nostro caso) utilizzato per produrre dei comportamenti e per generare dinamicamente il codice HTML. È in questo secondo caso che l'intervento e la mediazione di un Web Server è necessaria. Quando la risposta del server web è una pagina web dinamica, il server web deve eseguire il codice di programmazione (in PHP o Java) prima di inviare la risposta. Il browser interpreta la risposta, che spesso comporta altre richieste al server, per richiedere ad esempio immagini incluse o file CSS.

Per la trasmissione dei dati viene utilizzato il **protocollo di trasferimento HTTP** (o la sua variante crittografata HTTPS), che si basa sui **protocolli di rete** IP e **TCP** (più raramente UDP). Un server web può consegnare contemporaneamente i contenuti di un sito a tanti diversi computer e quindi ai loro browser. La velocità di elaborazione delle richieste degli utenti dipende dall'hardware e dal carico dell'Host, ma anche la complessità dei contenuti web ricopre un ruolo importante, perché ad esempio i contenuti dinamici richiedono più risorse rispetto a quelli statici.

Giunti a questo punto, si può quindi definire un Web Server come un software installato all'interno di un computer (server), con lo scopo di elaborare pagine web e di generare contenuti in modo dinamico.

Anche se la trasmissione di contenuti è il loro compito principale, i server web hanno molte altre funzionalità:

Sicurezza	Comunicazione protetta tra il server web e il client tramite protocollo crittografato HTTPS
Autenticazione degli utenti	Autenticazione HTTP per aree specifiche di un'applicazione web
Re-indirizzamento	Inoltro di un documento tramite Re-write Engine
Caching	Memorizzazione temporanea di documenti dinamici per rispondere più velocemente alle richieste e per evitare un sovraccarico del server web
Attribuzione di cookie	Invio ed elaborazione di cookie

Insieme a un software per server web, un host può contenere anche altri programmi: per esempio un server FTP per l'upload di file o un server database per contenuti dinamici (anche se un provider di regola metterebbe a disposizione diversi hardware). In generale è possibile utilizzare diversi tipi di server per fini completamente diversi: insieme a questi vengono processati anche mail server, game server o proxy server.

I Web server al momento più diffusi sono: **Apache HTTP Server** (*sviluppato da* Apache Software Foundation) e **IIS** (*sviluppato da Microsoft*). Oltre a questi due web server ce ne sono altri per esempio: Apache Tomcat, nginx, LiteSpeed Web Server ecc.

2.2.1 IIS: Internet Information Services

Microsoft Internet Information Services (IIS) è un pacchetto software in cui rientra anche un server web. Inizialmente era distribuito come un'option pack per il SO Windows NT, che poi venne integrato in Windows 2000 e Windows Server 2003.

In passato IIS era un componente di sistema per Windows Server: questo programma in alcuni sistemi operativi Windows può essere installato in un secondo momento. Non stupisce tuttavia che Microsoft IIS possa essere utilizzato solo su server Windows.

2.2.2 Apache HTTP Server

Il server Apache HTTP (o semplicemente server web Apache) è open source ed è presente in tante versioni diverse. Il programma modulare funziona su quasi ogni sistema operativo ed è il server web più utilizzato. È adatto a contenuti dinamici: si possono integrare senza problemi linguaggi di Scripting come PHP, e tramite interfaccia CGI Apache può eseguire altri programmi.

La maggioranza di Apache Http Server viene installata sulle distribuzioni, avendo le caratteristiche simili ai programmi eseguibili Unix-like, però, le versioni correnti forniscono i servizi anche sui sistemi Windows. Apache aveva un ruolo fondamentale nella crescita iniziale di WWW. In 2009, Apache era il primo web server a servire più di 100 milioni di siti web[7]. L'implementazione e la manutenzione del server viene effettuata da un gruppo di sviluppatori volontari, che utilizzano l'internet come mezzo di comunicazione per condividere le idee per la realizzazione del server.

[7] february_2009_web_server_survey.html

L'installazione di questo programma, benché piuttosto complicata, è però facile da eseguire, utilizzando software come XAMPP o LAMP: questi rendono l'installazione ancora più semplice. Oltre ad Apache contengono inoltre altri software per database e diversi linguaggi di Scripting. Recentemente il server Apache HTTP ha però perso in termini di velocità rispetto ad altri software per server web: molti altri prodotti concorrenti si contraddistinguono per tempi di risposta più brevi nella trasmissione dei dati.

2.3 Server Side Implementation

Nei paragrafi precedenti, abbiamo analizzato il ruolo di database per la memorizzazione dei dati dinamici delle pagine web, il contributo del web server per fornire l'output all'utente in formato html estraendo i dati dai database oppure il contenuto statico quando i dati non vengono estrapolati dal database, adesso rimane la parte di programmazione che per le pagine web dinamiche risulta la base nella creazione del codice html finale mandato al web browser.

La tecnologia lato server è usata nello sviluppo di siti con elementi dinamici e nelle applicazioni web; si basa sull'uso di script che *vengono eseguiti dal web server con l'aiuto dei linguaggi di Scripting* più adatti, quando un client richiede i contenuti corrispondenti. Spesso il compito degli script è quello di raccogliere i giusti dati da un database ed inserirli nel sito web. L'utente vi accede tramite pagine HTML, ma il codice sorgente degli script rimane completamente nascosto e non è quindi visionabile.

Nei primi anni della nascita di web, gli script server side erano implementati con la combinazione dei programmi scritti con il linguaggio C, gli script Perl, e gli Shell Script utilizzando CGI[8](Common Gateway Interface). Questi script poi venivano eseguiti dal sistema operativo e restituiti al web server una volta eseguiti.
I web server moderni, invece, sono in grado di eseguire i linguaggi "online Scripting" tipo ASP, PERL, PHP e RUBY dal web server stesso o attraverso i moduli di estensione all'interno di web server.

L'uso di questi script lato server presuppone che il client continui a inviare altre richieste al web server per far arrivare agli utenti le nuove informazioni modificate. Ciò significa da una parte un sovraccarico delle capacità del server, cosa che si ripercuote sui tempi di risposta del web server, e dall'altra prevede una **connessione esistente** al server, indispensabile per far visualizzare l'offerta web.
Esistono numerosi linguaggi di Scripting Server-Side:

- ASP
- ASP.NET
- GO
- Google Apps Script
- Java
- Javascript
- Lasso
- Perl
- PHP
- Python
- Ruby

E tanti altri...

Nei paragrafi successivi si mettono in evidenza le caratteristiche del linguaggio di Scripting Lasso (poiché è stato il linguaggio di studio).

[8] CGI: offre il protocollo standard per il web server per eseguire i programmi come se fossero delle applicazioni console in esecuzione sul server che serve le pagine web dinamiche

2.3.1 Lasso

Il linguaggio di studio per il nostro caso è Lasso. Lasso non è solo un linguaggio di Scripting, ma un linguaggio di programmazione dinamico, orientato agli oggetti che fornisce un supporto molto ricco al livello astratto di database. Lasso-Runtime è la parte fondamentale con cui il linguaggio si interagisce per avere l'accesso al livello più basso del sistema operativo in maniera ben organizzata.

Il linguaggio di Scripting LASSO era stato creato nel 1995 da Vince Befanti per pubblicare un database Filemaker pro sul web. Il nome Lasso era in origine il nome in codice per il progetto, scelto perché era stato costruito su un canale chiamato Frontier che aveva un motivo da cowboy. La facilità d'uso, la semplicità e la velocità di BlueWorld e Lasso 1.0 gli consentono di superare rapidamente la concorrenza come applicazione preferita dagli sviluppatori basati su Mac. Nel andare avanti col tempo più parametri di sicurezza e più funzionalità vennero aggiunte (tra cui manipolazione delle immagini, integrazione completa di AJAX, connettori SQL). Dalla prima versione di Lasso lanciata in 1996, siamo arrivati alla versione 9 di Lasso lanciata in 2010 (figura 6 riporta la storia delle versioni).

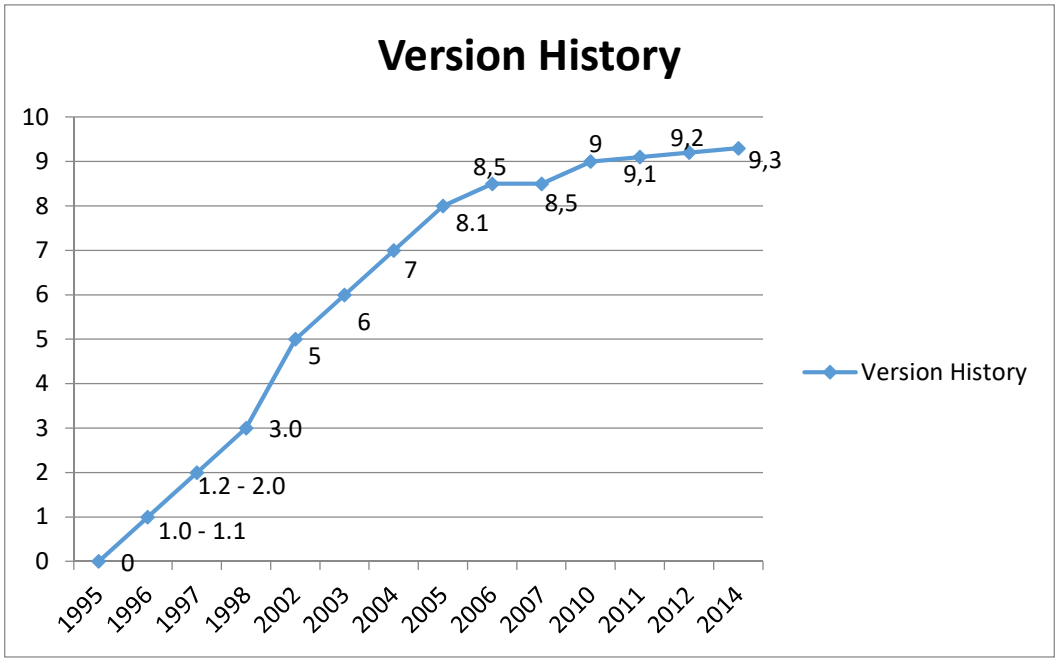


Figura 6: Storia versioni Lasso

Lo scopo del linguaggio è quello di consentire agli sviluppatori web di realizzare in modo veloce pagine dinamiche. Per le pagine dinamiche come già discuso in precedenza, si intendono pagine il cui contenuto viene, almeno in parte, generato nel momento in cui le stesse vengono richieste al web server. Un esempio classico di pagina dinamica è fornito dai ben noti motori di ricerca: i risultati che ci vengono restituiti a seguito di un'interrogazione non sono pagine web statiche, bensì documenti generati "su misura", sulla base della nostra richiesta. LASSO è un vero e proprio linguaggio di programmazione; è importante quindi rendersi conto della differenza con HTML, che non è un linguaggio di programmazione, ma un linguaggio di descrizione e formattazione del testo. Dalla documentazione ufficiale il LASSO viene definito come un «linguaggio script dal lato server immerso nel codice HTML». In seguito vedremo nel dettaglio questa caratteristica di Lasso chiamata "HTML embedded". Il codice LASSO inserito tra i marcatori HTML viene interpretato dall'elaboratore server e non dal navigatore del client. Il server elabora tramite l'interprete Lasso gli script i quali generano le pagine HTML che vengono fornite al client tramite il protocollo HTTP. Dunque il LASSO permette di rendere dinamiche le pagine HTML dal lato server. La figura seguente mostra i passaggi eseguiti dal server per rispondere alla richiesta del client.

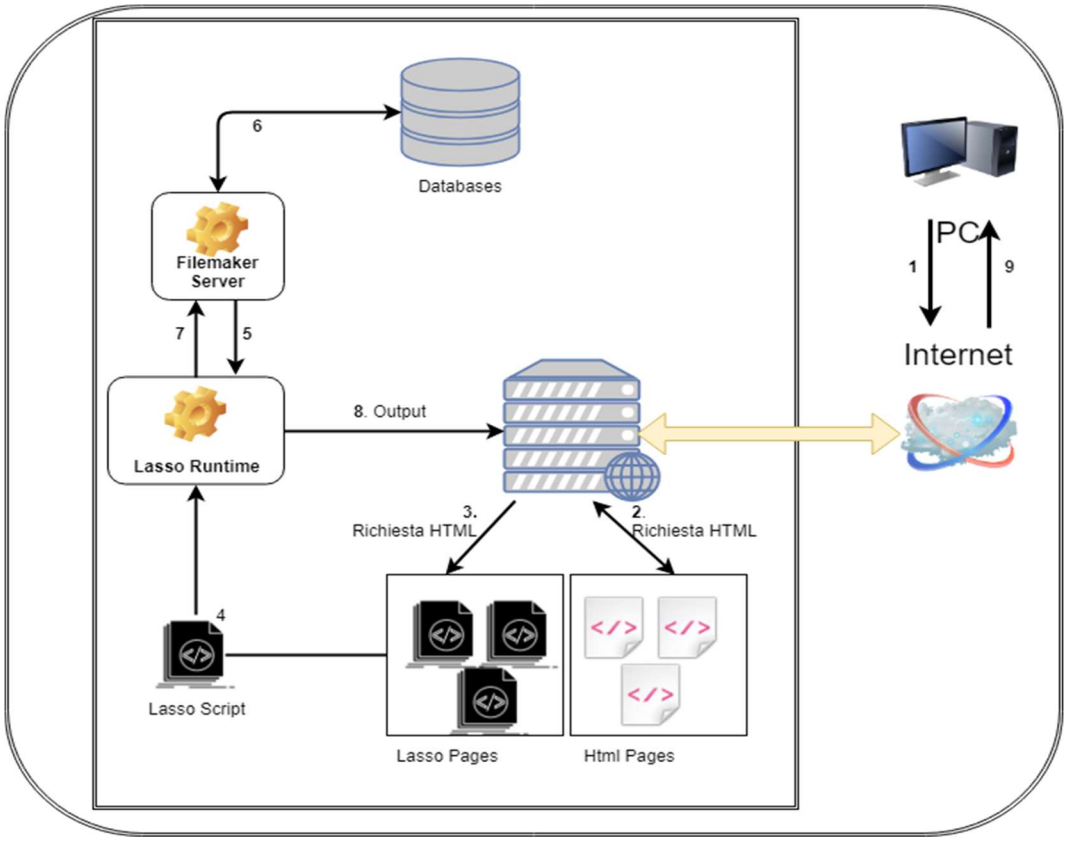


Figura 7: Flusso dati di un web server con Lasso

All'arrivo della richiesta (1) il server esegue un controllo sull'URL e decide se prelevare una pagina HTML statica (2) oppure se eseguire uno script per la generazione di una pagina dinamica (3). In questa ultima eventualità il codice viene passato al Lasso Runtime che lo esegue, con eventuali richieste al server Database (5, 6, 7) ed ai file system, e genera in output una pagina statica (8). La pagina finale (prelevata dal file system o generata dallo script) viene restituita in risposta al client (9).

2.3.1.1 Le Caratteristiche del Linguaggio

La definizione "ufficiale" del linguaggio Lasso, tratta dal manuale in lingua inglese su http://lassoguide.com/getstart/beginners/what-is-lasso.html, chiarisce le caratteristiche peculiari di questo linguaggio, e precisamente:

- **Lasso è un linguaggio di Scripting**

  I programmi scritti in linguaggio Lasso, denominati brevemente script, vengono eseguiti tramite un apposito software, “Lasso Runtime” che è una specie di interprete per gli script scritti nel linguaggio Lasso. Quest’ultimo si occupa di leggere il codice Lasso e, interpretandone le istruzioni, esegue le operazioni corrispondenti (ad esempio la lettura di un file o un calcolo aritmetico). Lasso è un linguaggio dinamicamente tipizzato ed orientato agli oggetti. I tipi e i metodi possono essere aggiunti nel codice nella fase di esecuzione. I tipi di oggetto e i metodi possono essere compilati in anticipo utilizzando il compilatore Lasso (**lassoc**). Il risultato di questa compilazione è una libreria dinamica che può essere caricata automaticamente su richiesta quando viene richiesto il tipo o il metodo. Ciò consente di mantenere i dati inutilizzati fuori dalla memoria, migliora i tempi di avvio e consente di creare e distribuire un'applicazione in modo modulare. Dunque il Lasso è quello che tecnicamente si definisce un linguaggio interpretato ed in questo esso si differenzia da altri linguaggi di programmazione, come ad esempio C++ e Java, il cui codice sorgente, per poter essere eseguito, deve prima essere compilato (tradotto cioè in codice macchina).

- **È un linguaggio "HTML-EMBEDDED"**

  Questa caratteristica si riferisce al fatto che il codice Lasso è immerso nell'HTML; gli script sono inseriti, in altre parole, nelle pagine HTML in cui devono produrre i loro effetti. Il web server riconosce le pagine Lasso, distinguendole da quelle "statiche", sulla base dell'estensione, che non sarà la solita .htm o .html ma piuttosto. lasso; quando il server riconosce una estensione associata a Lasso passa il testimone all'interprete, lasciando che sia quest'ultimo ad occuparsene.

- **Opera Server-Side**

  Il Lasso opera lato server. Ciò vuol dire che tutta l'elaborazione di uno script avviene sul server, prima che questi spedisca la pagina al browser (il "client"). Di conseguenza, chi accede a una pagina Lasso non ha la possibilità di leggere le istruzioni in essa contenute: essendo state processate, ciò che il client vedrà sarà il risultato dell'elaborazione; per riassumere: il client vedrà cosa fa lo script ma non come lo fa. L'esempio sopra citato del motore di ricerca è immediato e familiare, tuttavia la necessità di generare automaticamente il contenuto di pagine web si presenta in molte altre situazioni. Si pensi a un qualsiasi sito internet che fornisce in tempo reale (o quasi reale) gli indicatori dell'andamento del mercato azionario, o i risultati della giornata sportiva, o la situazione del traffico automobilistico, etc.; se non si vuole andare troppo lontano si pensi al caso più concreto dell'azienda che ha bisogno di estrapolare i dati contenuti nel loro database in base ai requisiti dell'utente che vuole fare una ricerca nel loro sito web. Nel caso specifico di questa tesi la necessità di generare in output i risultati della ricerca è legata alla possibilità di interrogare il database contenente i dati delle pagine web (ogni testo, immagine, video ecc. del sito web è contenuto nel database creato con la tecnologia Filemaker). In questa situazione occorre quindi avere a disposizione uno strumento che ci consenta appunto la comunicazione con un database rendendo il più possibile agevole la manutenzione dello stesso. Il linguaggio Lasso rappresenta la soluzione ideale ad una vastissima gamma di esigenze in quanto era il linguaggio originariamente creato proprio per pubblicare i risultati del database Filemaker pro sul sito web nel lontano 1995.

2.3.1.2 Lasso Runtime

"Il sistema runtime Lasso gestisce la compilazione dinamica del codice lingua Lasso fino al codice macchina nativo con schemi di caching integrati e ottimizzazione automatica dei metodi e dei tipi di oggetti più frequentemente utilizzati. Supporta inoltre una gamma flessibile di opzioni per la compilazione in anticipo del codice in librerie caricate dinamicamente e eseguibili stand-alone per piattaforme OSX e Linux."

Lasso Runtime utilizza il sistema di multi-threading per garantire un'alta qualità nella gestione delle richieste di compilazione. Il modello di threading di Lasso è integrato con il suo sottosistema di I/O trasparente basato sugli eventi per fare un uso efficiente dell'hardware con più CPU e assicurare che le richieste di rete vengano prontamente servite sotto carico pesante.

2.4 Altre Tecnologie Acquisite

Solo grazie al supporto di linguaggi dinamici (LASSO, PHP, ASP e altri) è possibile creare pagine web interattive. Ma per la realizzazione di questa applicazione è stato necessario acquisire la conoscenza di altre tecnologie, quali: HTML, Scripting Database, Connettore Ajax ecc.

2.4.1 HTML

Con il diffondersi del World Wide Web (cui spesso ci si riferisce usando la nota sigla WWW), una delle più diffuse applicazioni disponibili sulla rete Internet, si è sviluppato anche uno speciale linguaggio denominato Hypertext Markup Language (HTML) utilizzato per descrivere il contenuto di una pagina Web, ma non per costruirla. Poiché HTML rientra nella categoria dei linguaggi a marcatori, esso è costituito da un insieme di istruzioni (dette marcatori, o tag), che servono a

descrivere la struttura del documento. Ciò che si richiede quindi per scrivere un documento in HTML è la conoscenza della sintassi con cui i marcatori vengono introdotti all'interno del documento. Oggi esistono comunque dei programmi che permettono di creare ed aggiornare pagine web anche senza scrivere direttamente in linguaggio HTML. Alcuni di questi si usano come semplici programmi di videoscrittura e salvano i documenti direttamente in formato HTML. Fra gli editor HTML troviamo ad esempio Netscape Composer, incluso nel pacchetto di Netscape Communicator, Microsoft FrontPage e Macromedia Dreamweaver; anche con le ultime versioni di Microsoft Word è possibile creare una nuova pagina Web a partire da un documento di Word ed usare numerosi modelli che rappresentano i tipi più comuni di pagine Web. Se quindi viene data la possibilità di creare in via immediata documenti più o meno complessi da pubblicare nel Web anche all'utente non conoscitore della sintassi del linguaggio HTML, d'altro lato è comunque importante conoscere il linguaggio formale o perlomeno i suoi rudimenti, in quanto spesso gli editor consentono di visualizzare direttamente il codice HTML che sta dietro alla costruzione delle pagine Web e può quindi essere molto utile saper modificare direttamente il codice sorgente. Due sono le caratteristiche fondamentali del linguaggio HTML, riflesse anche dallo stesso nome:

- HTML è un linguaggio ipertestuale che permette di stabilire collegamenti (link) all'interno di uno stesso documento oppure fra più documenti nello stesso sito Web o in siti diversi, utilizzando le URL (Uniform Resource Locators);
- HTML è un linguaggio a marcatori che incorpora, oltre all'informazione che si vuole comunicare attraverso la pagina web, particolari sequenze di caratteri (marcatori, tag) che descrivono la struttura e il comportamento del documento. In un documento HTML troviamo quindi sia il testo (contenuto del documento) sia i marcatori (tag) che forniscono le istruzioni per la sua struttura. La stesura di un documento HTML richiede quindi la specifica di alcuni parametri per la formattazione del testo.

Il documento minimo in HTML è composto da:

```
<html>
        <head>
                <title>Titolo</title>
        </head>
        <body>
                Contenuto
        </body>
</html>
```

I tag sono racchiusi tra parentesi angolari < >, controllano il modo in cui il documento viene strutturato e visualizzato da un browser. In particolare:

- il tag <html> posto all'inizio del documento e il tag </html> posto alla fine indicano al browser che l'intero documento è scritto in HTML;
- il titolo del documento viene indicato dal tag <title> che marca l'inizio titolo e dal tag </title> che marca la fine del titolo;
- i tag <head> e </head> definiscono l'intestazione del documento e servono come contenitore per altri tag di intestazione
- i tag <body> e </body> delimitano l'inizio e la fine del contenuto della pagina Web.

Fra questi tag è compreso tutto il testo che compare nella pagina Web, inclusi i vari collegamenti.

È possibile riassumere alcune caratteristiche dei tag:

- i nomi dei tag sono generalmente un'abbreviazione della funzione stessa svolta dai tag e possono essere scritti indifferentemente con lettere maiuscole o minuscole;
- Affinché i tag non siano ignorati, occorre che i caratteri che compongono il nome siano contigui e quindi occorre non inserire spazi nei tag o fra le parentesi angolari e il nome dei tag;

- Spesso esiste un tag di apertura ed un tag di chiusura aventi lo stesso nome. Nel tag di chiusura il nome del marcatore è preceduto da "/" (esempio </html> indica il fine della pagina html)

Per visualizzare una pagina Web scritta in HTML si utilizza un browser, cioè un programma che interpreta e visualizza le pagine Web. Quando il browser visualizza il documento i tag non appaiono; invece il testo scritto fra il tag di apertura ed il tag di chiusura seguirà le istruzioni del tag nella visualizzazione del documento con il browser.

2.4.2 SQL

La sigla SQL, acronimo di Structured Query Language ovvero linguaggio di richiesta strutturato, è ormai diventata sinonimo di linguaggio standard per la definizione, l'interrogazione e l'aggiornamento di database relazionali. Nasce nel 1974 nei laboratori dell'IBM. Nasce come strumento per lavorare con database che seguano il modello relazionale. A quel tempo però si chiamava Sequel. Nel 1975 viene sviluppato un prototipo chiamato Sequel-XRM; con esso si eseguirono sperimentazioni che portarono, nel 1977, a una nuova versione del linguaggio, che inizialmente avrebbe dovuto chiamarsi Sequel/2 ma che poi divenne, per motivi legali, SQL. Su di esso si sviluppò il prototipo System R, che venne utilizzato da IBM per usi interni e per alcuni suoi clienti. Ma, dato il suo successo, anche altre società iniziarono subito a sviluppare prodotti basati su SQL. Nel 1981 IBM iniziò a vendere alcuni prodotti relazionali e nel 1983 rilasciò DB2, il suo DBMS relazionale diffuso ancor oggi. SQL divenne subito lo standard industriale per i software che utilizzano il modello relazionale. L"ANSI lo adottò come standard fin dal 1986, senza apportare modifiche sostanziali alla versione inizialmente sviluppata da IBM. Nel 1987 la ISO fece lo stesso. Questa prima versione standard è denominata SQL/86. Negli anni successivi si 49 realizzarono altre versioni, che furono SQL/89, SQL/92 e SQL/2003. Tale processo di standardizzazione mirava alla creazione di un linguaggio che funzionasse su tutti i DBMS (Data Base Management Systems) relazionali, ma questo obiettivo non fu

raggiunto. Infatti, i vari produttori implementarono il linguaggio con numerose variazioni e, in pratica, adottarono gli standard ad un livello non superiore al minimo, definito dal Ansi come Entry Level. Essendo un linguaggio dichiarativo, SQL non richiede la stesura di sequenze di operazioni (come ad es. i linguaggi imperativi), ma piuttosto di specificare le proprietà logiche delle informazioni ricercate. Esso si divide in tre sottoinsiemi: - Data Definition Language (DDL), linguaggio di definizione dei dati, viene utilizzato per definire lo schema della base di dati. Le definizioni dello schema vengono memorizzate in una base di dati speciale, detta Dizionario dei dati (Data Dictionary). Il DDL opera sia a livello esterno, sia a livello logico, sia a livello interno; in particolare, le istruzioni che definiscono il livello esterno prendono talvolta il nome di view definition (definizione delle viste, cioè degli schemi dell'utente), mentre le definizioni che definiscono il livello interno prendono il nome di DCL (device control language, o linguaggio per il controllo dei dispositivi).

- **Data Manipulation Language (DML)**, viene utilizzato per manipolare la base di dati, cioè per svolgere le seguenti funzioni: Estrarre informazioni dalla base di dati, cioè formulare un'interrogazione comunque complessa sulla base di dati ed ottenere la riposta all'interrogazione; modificare il contenuto della base di dati, cioè svolgere operazioni di inserimento di nuovi dati, cancellazioni di dati preesistenti, e modifica di specifici valori di dati preesistenti. I comandi utilizzati sono: SELECT per la ricerca, INSERT per l'inserimento, UPDATE per l'aggiornamento e DELETE per la cancellazione.

- **Data Control Language (DCL)**, linguaggio utilizzato per fornire o revocare agli utenti i permessi necessari per poter utilizzare i comandi DML e DDL, oltre agli stessi comandi DCL (che servono per poter a sua volta modificare i permessi su alcuni oggetti). Una caratteristica importante del linguaggio è il suo grado di

proceduralità: un linguaggio è procedurale se il programmatore deve specificare in modo esatto quale metodo di accesso ai dati dovrà essere utilizzato per ritrovare l'informazione; un linguaggio è non procedurale se l'utente deve solo indicare quale informazione dovrà essere ritrovata, e viceversa omettere di indicare il modo in cui l'informazione verrà ritrovata. In tal caso, il sistema determina in modo automatico il cammino di accesso ai dati.

2.4.3 JQuery / Ajax

JQuery è una libreria Javascript, pubblicato per la prima volta nel 2006 ed è tuttora in evoluzione. Essendo una sottocategoria del linguaggio Javascript che è un linguaggio di Scripting orientato agli oggetti, eredita tutte le sue caratteristiche. Era nato per semplificare la selezione, la manipolazione e la gestione degli eventi e animazione di elementi DOM in pagine HTML. L'approccio di tipo modulare di JQuery consente la creazione semplificata di applicazioni web e versatili contenuti dinamici.

AJAX (acronimo di **Asyncrounous**[9] **Javascript and XML**), è il linguaggio che viene utilizzato principalmente nei Javascript/JQuery. Esso permette la piena comunicazione tra cliente (nel nostro caso il web browser) e ed il server (nel nostro caso il lasso server che a sua volta si appoggia si IIS Server). Lo sviluppo delle applicazioni HTML con le integrazioni di AJAX si basa su uno scambio di dati in background fra web-browser e server, consentendo l'aggiornamento dinamico di una pagina web senza che l'utente lo richieda esplicitamente, ed è questo il motivo per cui è stato deciso di implementare la richiesta di connessione

[9] Differenza fra richiesta sincrona e asincrona sta nel fatto nel primo caso l'esecuzione del codice viene interrotta bloccando il browser temporaneamente fino al completamento della richiesta mentre nel secondo caso non occorre attendere che la richiesta sia ultimata per effettuare le altre operazioni.

tra browser e server attraverso JQuery/AJAX. Con AJAX, in pratica, gli sviluppatori web sono in grado di far corrispondere a un click o in generale ad un evento generato dall'utente, non più solo il mero "cambio pagina", ma effettuare una chiamata al server, ottenere da questi nuovi dati e utilizzarli per modificare dinamicamente i contenuti dell'attuale pagina. Il tutto in modo trasparente per l'utente, che crede di trovarsi sempre sulla stessa pagina e ne vede però "magicamente" cambiare i contenuti. In teoria, grazie ad AJAX, un sito web potrebbe essere formato da una sola pagina, il cui contenuto viene di volta in volta variato in conseguenza dei click dell'utente che però non vedrà mai il messaggio: "attendere, caricamento pagina in corso..." e avrà invece la sensazione di restare sempre e solo su un'unica pagina.

Funzionamento

La pagina web, in conseguenza del click, invoca una funzione Javascript, la quale, utilizzando un oggetto del browser, chiamato "xmlhttprequest" (XHR), apre una comunicazione con una procedura sul server, che nostro caso è LASSO. Il server restituisce i dati, in genere rappresentati in XML (ma nel nostro caso cono JSON/XHR) e invoca un'altra funzione Javascript che li elabora e li usa per modificare i contenuti della pagina.

Parte II

Parte della Realizzazione

Capitolo 3: Progettazione

Questo capitolo riguarda alla parte della realizzazione dello script di ricerca, in particolare in questo capitolo si affronta la parte di progettazione, mentre nel prossimo capitolo si affronterà l'implementazione di essa.

3.1 Breve Introduzione

Il progetto formativo riguardante questa tesi di laurea consiste nella creazione di una piccola funzione di ricerca per il sito web dell'azienda in cui è stato svolto lo stage formativo: Multitraccia Sc. È una piccola software house nella città di Reggio Emilia, che si occupa di creare il software per i suoi clienti, offre i servizi di tipo stampa, mette a disposizione la memoria nel cloud per i suoi clienti ecc. Il sito web dell'azienda è un sito web "data-driven" le cui pagine web sono dinamiche e contenute in un database. Il database utilizzato da loro è della tecnologia Filemaker. La scelta di utilizzare il database Filemaker è stata soprattutto per la sua integrabilità all'interno delle pagine web che sono state scritte con il linguaggio Lasso.

Ogni parte del sito web che sia un'immagine, o un video, o la descrizione del prodotto/servizio, o i link verso i siti interni, oppure il footer dei contatti dell'azienda, è tutto contenuto nel database. Per mantenere la caratteristica dinamica del sito web, le pagine web sono scritte nel linguaggio *Lasso*.

La scelta del linguaggio e la scelta dell'utilizzo della suite lasso (lasso server, lasso instance manager ecc.) per la creazione del sito è stato soprattutto esemplificativo per i suoi clienti, in modo da dare un esempio di prodotto creato puramente in Lasso sfruttando tutte le funzionalità di connessione con il database Filemaker pro che offre la suite Lasso.

Prima della scrittura dello script per la funzione di ricerca è stato necessario replicare "working tools" utilizzati dall'azienda sul proprio PC in modo tale da rendere funzionante la ricerca (e per fare le prove *in live*) ovvero è stato configurato Filemaker Server, Lasso Server, Lasso Intstance Manager in modo tale che questi strumenti si *"colleghino"* con il nostro web server: IIS web server, le cui caratteristiche sono state discusse nel capitolo precedente. Sia la suite Filemaker sia la suite Lasso sono gli strumenti a pagamento, installati nel PC dopo aver acquisito la licenza del software.

3.2 Configurazioni Utili

Installazione del Lasso Server

Per quanto riguarda Windows, Lasso Server è compatibile solamente con Windows Server 2012, Windows Server 2008 R2, Windows 8, and Windows 7 e i web server compatibili sono IIS7, IIS8 e Apache2.2.

Dopo aver installato i prerequisiti del server, si installa Lasso Server dal sito web di lassosoft. Una volta installato Lasso Server occorre completare l'installazione completando il form disponibile sulla pagina: http://your-server-domain.name:8090/lasso9/lux.

Configurazione Lasso Instance Manager

Dopo aver configurato il lasso server e completato il form di inizializzazione del server, occorre configurare il Lasso Instance Manager. Il Lasso Instance manager è un tool che accompagna il lasso server per una corretta gestione di una o più istanze del lasso server, che sono i processi isolati di Lasso Server ognuno in esecuzione nel loro ambiente. La creazione delle istanze è di fondamentale importanza nell'utilizzo nei progetti realizzati con i tool di Lasso.

Figura 8: Collegamento Web Server e Lasso Server

Lasso Instance Server svolge una funzione di intermediario tra Web-Server e le istanze di Lasso-Server come si vede dalla figura 8. Il lasso instance manager accoglie le richieste che arrivano da file con l'estensione ".lasso" e tutte le richieste che hanno il pattern URL "/lasso9/".

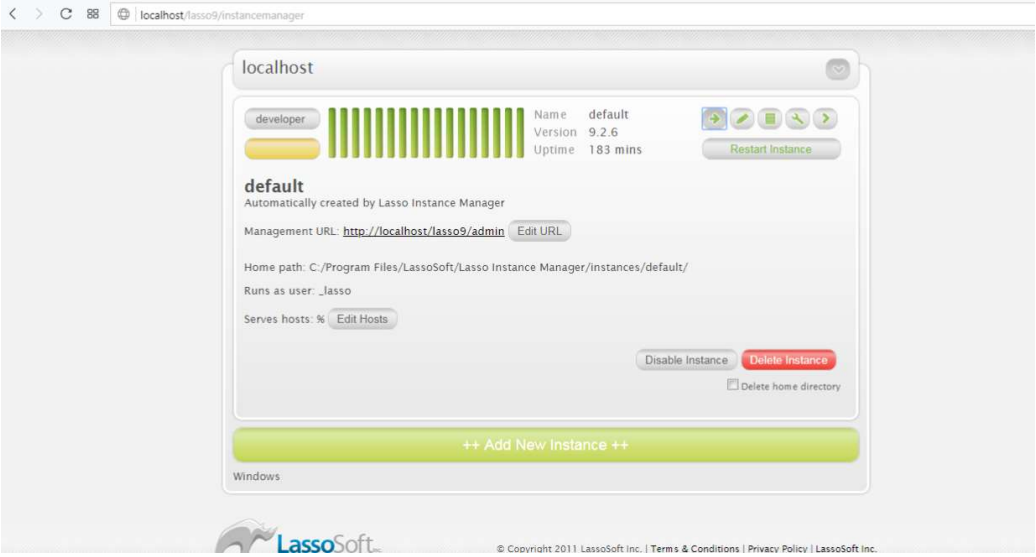


Figura 9: Istanza del progetto su instance manager

La figura 9 rappresenta l'istanza del progetto creato su lasso instance manager che fa da tramite tra risorse del lasso server e web server IIS 7.

Connessione Filemaker-Server e Lasso-Server

Per utilizzare il database del sito web che viene ospitato dal Filemaker Server, si crea una connessione tra il server che occupa database e il lasso server che deve interrogare il database e fornire i risultati in base alle esigenze dell'utente. Per fare l'utente admin o un altro utente registrato precedentemente nel server del lasso, deve configurare la connessione sulla porta 8080 utilizzando le credenziali d'acceso del database (username e password). Il lasso server consente di visualizzare in-line il contenuto nel database e eseguire delle query del contenuto. La figura 10 rappresenta la visualizzazione di una pagina del database con la funzionalità db-browser del Lasso Server. Questa funzionalità consente all'utente di scrivere un in-line script in linguaggio da testare direttamente sul contenuto del database.

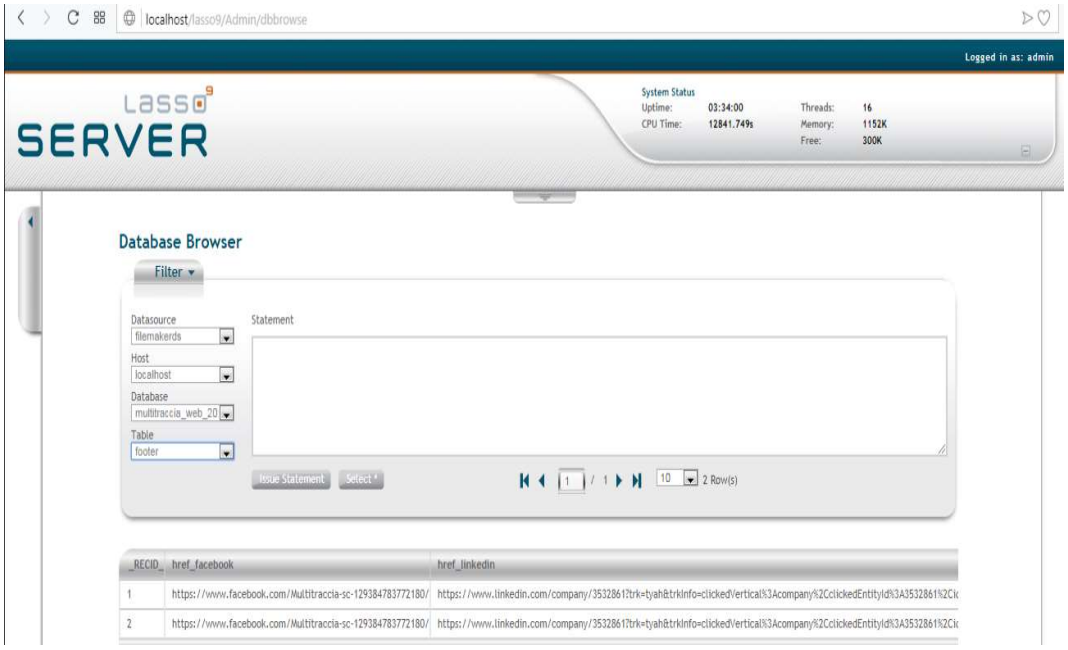


Figura 10: DB Browser interno al Lasso Server

3.3 Specifica dell'azienda

La specifica dell'azienda era quella di fare la ricerca in tutto il database (e quindi in tutte le pagine) ma di fornire il risultato nell'Home-page del sito, in modo che l'utente non lasci interamente il sito e abbia la visione completa anche di altri servizi offerti dall'azienda.

Per questo motivo quando l'utente fa una query di ricerca ad esempio per un progetto, il cursore di ricerca non va a sottolineare la *"parola"* ma visualizza le sezioni in cui possibilmente la stringa della query è presente. L'Home-Page del sito è suddiviso in varie sottosezioni, ogni uno identificabile con una id. È stato sfruttato questo ID per creare le ancóra interni alla pagina.

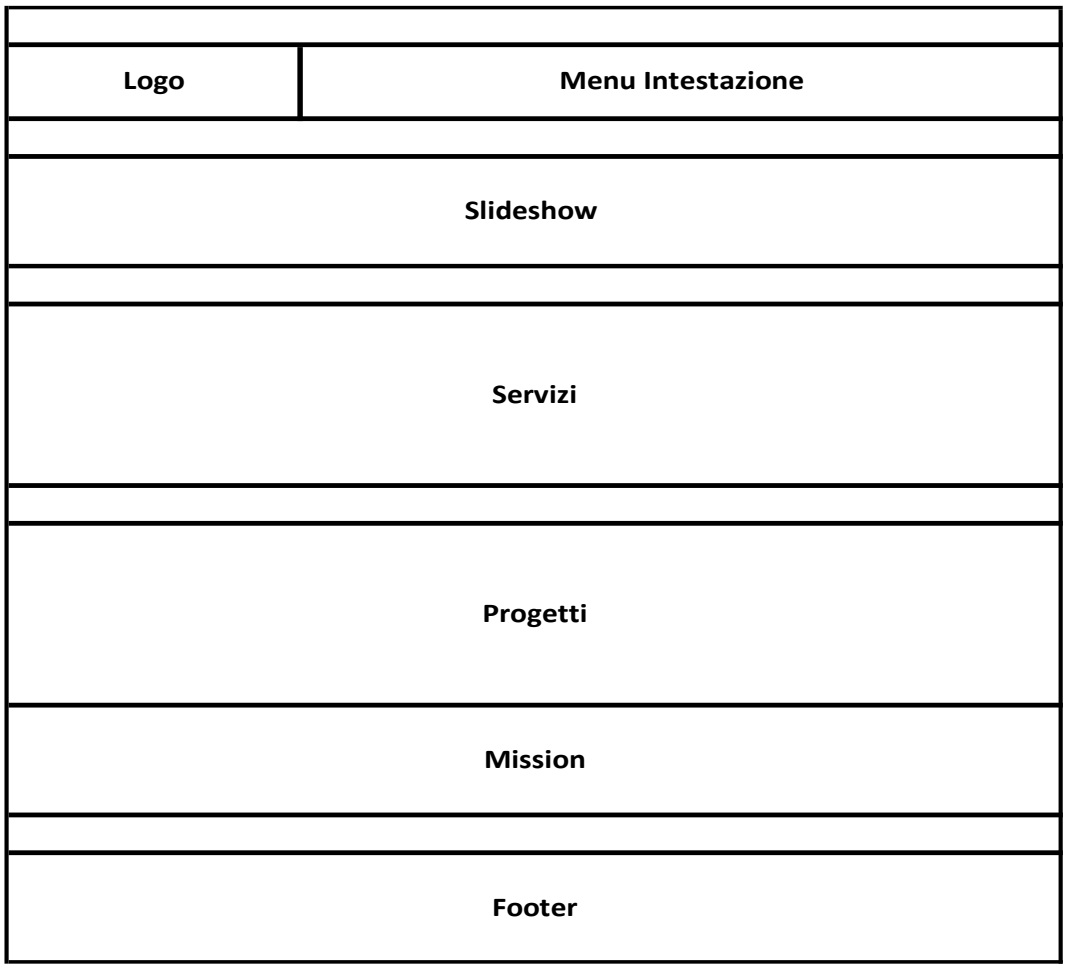


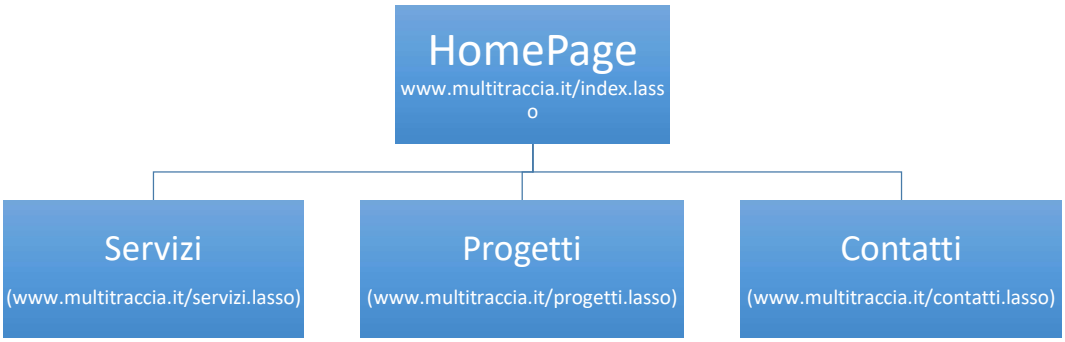
Figura 11: Le sezioni principali dell'Home Page

La figura 11 mostra la suddivisione delle sezioni del sito web di Multitraccia.

Nella nostra implementazione dello script di ricerca, oltre a spostare nella sezione dell'Home Page, abbiamo visualizzato anche i link delle pagine esterne nelle quali sono contenute le parole ricercate.

3.4 Progettazione

Le pagine del sito web in questione sono strutturate nel seguente modo:



Le pagine che vengono chiamate dall'Home Page (servizi.lasso, progetti.lasso) sono *"renderizzate"* all'utente in base all'indice assegnato al contenuto nel database.

Ad esempio cliccando sul link "MULTIcolore" si apre la pagina localizzata all'URL http://www.multitraccia.it/servizi.lasso?lk=2&sv=1 dove lk=2 è il variabile di lingua (lk=1 : inglese, lk=2 : italiano) e sv=1 è l'indice del servizio. Viene utilizzato il metodo *GET* nel passaggio degli argomenti per il corretto indirizzamento dell'URL.

La funzione ricerca parte dalla casella di ricerca creata con il form html, in cui l'utente può inserire una stringa o un insieme di strighe da cercare nel sito. Attraverso l'identificatore della casella della form, viene agganciata una funzione di JQuery/Ajax. La funzione di JQuery/Ajax serve per connettersi al database attraverso i parametri che sono stati definiti nel file db_conn.lasso. Questo file viene messo in tutte quelle parti delle pagine web dove si ha bisogno di interagire con il database dell'azienda.

Una volta creata la connessione, si può interrogare il database in tutte le tabelle e in tutti i campi in cui si rende necessario effettuare la ricerca in base alla specifica dell'azienda. I risultati vengono forniti al web server sotto forma di una variabile JSON serializzata.

I risultati vengono visualizzati in un pop-up inline alla pagina html (che contiene i link ancóra per evidenziare le sezioni in cui è contenuto la stringa cercata) e in una casella di testo contenente l'elenco dei siti web esterni all'Home Page del sito

Capitolo 4: Implementazione

4.1 DB Connection

Una delle caratteristiche del Lasso è la capacità di gestire direttamente i principali serventi di base di dati tramite connessioni dirette, senza l'utilizzo di strati intermedi quali, ODBC (anche se gestiti perfettamente) o JDBC; una volta installato l'interprete in modo corretto, si è in grado di connettersi a qualunque base di dati, nel nostro caso Filemaker. Lasso è in grado di gestire Filemaker mediante il linguaggio di interrogazione strutturato di Filemaker, che tramite una sintassi standard descritta nella manuale del database permette di effettuare tutte le operazioni necessarie sui dati (che saranno trattate più avanti), Le operazioni che consente di fare uno script Filemaker sono molto più avanzate rispetto al linguaggio di interrogazione SQL. Una volta fatta la connessione con la base di dati e attivate le politiche di accesso, ciò di cui si necessita per operare sui dati è:

- L'indirizzo IP del server che ospita il DBMS Filemaker;
- Nome della base di dati.

Ogni base di dati viene identificata attraverso un nome;

- **Username**: Il nome dell'utente abilitato alla manipolazione della base di dati;
- **Password**: La parola d'accesso alla base di dati. Appare chiaro che questi dati sono importanti e vanno tenuti segreti. Tutti i parametri necessari alla connessione sono stati concentrati in un unico file, tale file è incluso all'inizio di ogni pagina dalla quale si voglia effettuare una connessione alla base di dati. Il file che contiene i parametri alla connessione si chiama db_conn.lasso che contiene le seguenti righe:

```
<?lasso
        local(is_started=false);
        inline(
                -host=(:
                -datasource='filemakerds',
                -name='localhost',
                -port='80',
                -username='*****',
                -password='*****'
                ),
            )=> {^
                #is_started=true
                ^}
        if (!#is_started)
                '<br />'+'DB not Connected'+'<br />'+'<br />'
        /if
?>
```

A questo punto abbiamo tutti i dati necessari per la connessione al database. Questa la possiamo realizzare attraverso la funzione inline;

Il metodo *inline* viene utilizzato per specificare un'azione del database e per presentare i risultati di tale azione all'interno di una pagina Lasso. In questo caso per connettersi al database, prende in ingresso il parametro *host* che a sua volta prende in ingresso altri parametri di connessione al database che sono stati indicati nell'esempio del codice sopra citato (*datasource* = tecnologia con cui il database è stato realizzato che può essere trovato nella sezione "Datasources" di Lasso Server Admin, *name*=fonte dei dati, *port*=opzionale che indica la porta dell'origine dei dati, username e password sono le credenziali d'accesso per il database).
E' stata definita la variabile *#is_started* per verificare l'avvenuta connessione al database. Se la connessione va a buon fine ci restituisce un identificatore alla connessione che noi memorizziamo in *#is_started*. Questa variabile la utilizzeremo ogni volta che vorremmo fare un'operazione sul database. Se la connessione non dovesse andare a buon fine (per esempio se uno dei parametri fosse sbagliato) la variabile conterrà il valore booleano FALSE. Quindi abbiamo aggiunto una verifica con un if per visualizzare l'errore "DB not connected", grazie a queste righe se la connessione dovesse fallire, otterremmo il messaggio di errore.

4.2 Estrazione Dati dal Database

Per estrapolare i dati dal database, innanzitutto viene incluso il file "db_conn.lasso" per creare la connessione con il database contenente le pagine del sito, dopodiché viene salvata la stringa di parole inviata con una richiesta *POST* in una variabile:

```
<?lasso
        content_type('text/javascript')
        Include('db_conn.lasso')
        local(ricerca = action_param('search'))
?>
```

La variabile *ricerca* viene utilizzata per fare la query in tutte le tabelle del database.

La variabile della lingua è stata impostata manualmente sul valore 2 (*local(lingua = '2')*), in quanto la verifica è stata fatta per il sito web in Italiano. Per la versione Inglese del sito, la variabile della lingua deve essere impostata sul 1. Questo può essere automatizzato con la seguente funzione:

```
local(lingua = web_request->param('lk', void));*/
        /** controllo parametro lingua */
        if  (#lingua === void || #lingua->Size() == 0 )
                #lingua = 2;
        if
```

Per effettuare la ricerca sono state inizializzate alcune variabili di tipo *"map"* (tante quante sono le tabelle in cui è stata effettuata la ricerca) per memorizzare i risultati. Oltre a queste vengono memorizzate anche le informazioni relative al totale dei risultati trovati.

Il linguaggio Lasso integrato con il FM Scripting offre la funzione *Inline*, che effettua la ricerca di tipo "Full-Text" di cui abbiamo discusso nel capitolo 1. Si fornisce un esempio del codice di ricerca in una tabella del database con la relativa copia dei dati nella variabile che viene passata alla request di Ajax:

```
inline(
                -search,
                -database='multitraccia_web_2015',
                -table='progetti',
                        -cn, 'main_titolo'=#ricerca, -or,
                        -cn, 'nome_txt'=#ricerca, -or,
                        -cn, 'meta_tag'=#ricerca, -or,
                        -cn, 'progetto_titolo'=#ricerca, -or,
                        -cn, 'progetto_sottotitolo'=#ricerca, -or,
                        -cn, 'progetto_testo'=#ricerca, -or,
                        -cn, 'btc_testo'=#ricerca, -or,
```

```
                    -cn, 'quote_txt'=#ricerca, -or,
                    -cn, 'quote_nome'=#ricerca) => {^
                        local(trovato_progetti) =(found_count>0)
                        local(progetti = records_map(-keyField="z_id",
                                    - returnField='z_lingua',
                                    -returnField='main_titolo',
                                    -returnField='nome_txt',
                                    -returnField='meta_tag',
                                    -returnField='progetto_titolo',
                                  -returnField='progetto_sottotitolo',
                                    -returnField='progetto_testo',
                                    -returnField='btc_testo',
                                    -returnField='quote_txt',
                                    -returnField='quote_nome',
                                    -type="array"));^}
Iterate(#progetti,local(j))
                    #j = #j->Second
            If (#j->find('z_lingua') == #lingua)
                    #conta_progetti = loop_count
                        #progetti_results = #j->asCopyDeep
            /If
/Iterate
```

La funzione Inline viene utilizzata per tanti scopi, per questo motivo, il primo parametro della funzione viene settata su *"search",* per attivare la funzione ricerca dell'Inline. Dopodiché viene settato il parametro del database che deve essere coerente con il database che è stato connesso. Una volta settato questi parametri si può procedere con la ricerca nei vari campi della tabella, che nel nostro esempio è *'progetti'*. L'opzione **–cn** ci serve in modo che tutti i campi che vogliamo, vengano attivati nella ricerca.

La variabile *trova_progetti* contiene il valore di tipo Booleano: True nel caso vengano trovati i risultati nella tabella. La variabile *progetti* contiene un array degli oggetti trovati nella tabella.

Il nostro obiettivo è quello passare i risultati con il formato JSON per cui, è stato necessario copiare i dati dal array in una map che salva i dati con la relazione {Chiave, Valore}, quindi la copia dei dati da *progetti* a *progetti_results*.

Tutti i risultati di tutte le tabelle vengono salvate in una variabile di tipo map (*#json*) attraverso la funzione insert.

```
#json->insert('servizi_results'=#servizi_results)
#json->insert('progetti_results'=#progetti_results)
...
```

Per passare i dati al server e per rispondere alla richiesta di Ajax la variabile #json viene serializzata con la funzione *json_serialize*:

```
local(xout = json_serialize(#json))
```

Il risultato *xout è* la variabile di risposta alla richiesta di Ajax con tutti i dati della ricerca.

4.3 Grafica

Form di Ricerca

La figura 12 rappresenta la casella di ricerca.

Figura 12: casella di ricerca

```
<form class="form-box">
        <input id="searchquery" type="text" placeholder="cerca.."
                maxlength="20"autocomplete="off">
        <input id="submitbt" type="submit" value=">>">
</form>
```

Il primo input riguarda alla casella di ricerca mentre il secondo input riguarda al bottone che fa partire una serie di azioni quando viene cliccato. È stato agganciato l'id ***submitbt*** al bottone che viene agganciato allo script di JQuery per azionare la funzione di Ajax.

4.3.1 Visualizzazione Risultati

Implementando la connessione attraverso JQuery e Ajax, è stato possibile ottenere un pop-up inline nella pagina principale per far visualizzare i risultati come si vede dalla figura 13. Se si implementa la connessione con il normale Javascript si perdono tutte le animazioni, in quanto ogni volta che viene cliccato il bottone per cercare una parola, la pagina web subisce un refresh involontario.

Figura 13: Risultato

```
(document).ready(function() {
    var show;
    $('#submitbt').click(function() {
        var query = $("#searchquery").val();
        var thisObj = new Object();
        thisObj.search = $("#searchquery").val();
        if (query == null || query == "") {
            alert("No valid input");
            return false;
        } else {
            $.ajax({
                url: "db.lasso",
                data: thisObj,
                type: 'post',
                cache: false,
                dataType: 'json',
                success: function(xhr) {
                    if (xhr.total == 0) {
                        show = false;
                    } else {
                        show = true;
                        $('#internal-container-0').hide();
                        $('#internal-container-1').hide();
                        $('#internal-container-2').hide();
                        $('#internal-container-3').hide();
                        $('#internal-container-4').hide();
                        $('#internal-container-5').hide();
                        $('#out-total').html("");
                        $('#res-0').html("");
                        $('#res-1').html("");
                        $('#res-2').html("");
                        $('#res-3').html("");
                        $('#res-4').html("");
                    }
                    var s_term = '"' + xhr.ricerca + '"' + ' nella sezione: ';
```

```
            $('.intestazione').html(s_term);
            $('#out-total').html(xhr.output);
            $('#container-box').removeClass('animated fadeOutUp');
            $('#container-box').show().addClass('animated fadeInDown');
            update_results(xhr);
            if (show == true){
               $('#container-box-2').removeClass('animated
                        slideOutRight');
               $('#container-box-2').show().addClass('animated
                     slideInRight');
            }
            else{
               $('#container-box-2').hide();
            } } });
        return false
    }   });}
```

Attraverso il commando *$('#submitbt').click(**function()**)*, vengono agganciati gli eventi del click del mouse o dell'invio della tastiera azionati sul bottone che viene identificato con l'identificatore *#submitbt* alla funzione di JQuery.
Il valore che viene immesso nella casella di testo viene recuperato attraverso il commando *$("#searchquery").val()* e viene salvato nella variabile di tipo oggetto *thisObj.search* . Questo valore serve alla funzione Ajax per passare il valore al server del database che viene interrogato.

La funzione di Ajax prende in ingresso alcuni parametri che nel nostro caso sono:

url: URL della risorsa alla quale viene inviata la richiesta;

type: il tipo di richiesta HTTP da effettuare. Accetta GET o POST (il valore di default è: 'GET');

data: che può essere un oggetto del tipo {chiave: valore, chiave2: valore}, oppure una stringa del tipo "chiave=valore&chiave2=valore2" contenente dei dati da inviare al server, per questo motivo è stato definito una variabile ti tipo object per salvare il contenuto della casella di test al suo interno;

cache: se impostato su false forza il browser a ricaricare ogni volta i dati del server anche se non sono cambiati (il valore di default è: True);

dataType: Tipo di dato che si riceve di ritorno; JQuery tenta di capirlo da solo, ma è sempre meglio specificarlo. I tipi possibili sono: "xml", "html", "script", "json", "jsonp" e "text";

success (data, textStatus, XMLHTTPRequest): funzione che verrà eseguita al successo della chiamata. I tre parametri sono, rispettivamente, l'oggetto della richiesta, lo stato e la descrizione testuale dell'errore rilevato. Tutto quello che passato nel formato JSON alla chiamata di Ajax è accessibile nel nostro caso tramite l'oggetto XHR. È un oggetto che contiene i dati di tipo {chiave, valore}. Per esempio il totale che viene passato dal server alla chiamata di Ajax viene acceduto tramite l'identificatore xhr.total.;

Per visualizzare i risultati delle pagine esterne all'Home-Page, sono stati sfruttati gli identificatori (della lingua, del servizio/progetto, della categoria progetto). Il formato dell'indirizzo di pagine esterne della Multitraccia seguono il formato: *HomePage/servizi.lasso?lk=*&sv=** oppure *HomePage/progetti.lasso?lk=*&pg=*&mk=** .

- Lk: identificatore della lingua
- Sv: identificatore del servizio
- Pg: categoria dei progetti (WEB, COLOR, LINEA, ecc...)
- Mk: identificatore del progetto

Per formare l'indirizzo completo si concatenano gli identificatori ritrovati con la ricerca e la stringa URL base del sito web. L'esempio sotto riportato visualizza la formazione dell'indirizzo con la stringa dell'URL del sito web, la categoria del progetto e identificatore del progetto:

```
id_p = xhr.progetti_results.z_id;
id_cat = xhr.progetti_results.z_cat;
indirizzo_p = "www.multitraccia.it/progetti.lasso?lk=2&pg=" + id_cat + id_p;
```

La figura 14 riporta la visualizzazione dei risultati quando l'utente cerca la parola "multitraccia":

Figura 14: Risultati Pagine esterne

Conclusioni

In questa tesi è stata sviluppata una funziona di ricerca con un linguaggio di programmazione che non è stato studiato nel corso della laurea (Lasso). Sono state studiate le tecniche di implementazione, il modo in cui il server utilizzato da questo linguaggio si relaziona con il web server, nel nostro caso IIS Server. Oltre questa parte principale, sono stati discussi i vantaggi di implementare uno script ricerca con un linguaggio server-side.

Questo script di ricerca mette in relazione varie tecnologie web server-side come per esempio Ajax, FM Scripting e Lasso e dà la possibilità all'azienda di modificarlo come meglio ritiene anche per gli sviluppi futuri.

Bibliografia e Sitografia

[1] Wikipedia, «Motore di Ricerca,» Wikipedia, [Online]. Available: https://it.wikipedia.org/wiki/Motore_di_ricerca. [Consultato il giorno 10 03 2019].

[2] Wikipedia, «Server Side Scripting,» Wikipedia, .[Online]. Available: https://en.wikipedia.org/wiki/Server-side_scripting. [Consultato il giorno 15 03 2019].

[3] Wikipedia, «Sistema Client Server,» Wikipedia, .[Online]. Available: https://it.wikipedia.org/wiki/Sistema_client/server. [Consultato il giorno 15 03 2019].

[4] http://lassoguide.com/index.html

[5] https://fmhelp.filemaker.com/docs/14/it/fmp14_users_guide.pdf

[6] J.F. Kurose, K.W. Ross, "Reti di Calcolatori e Internet”