

ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA

SCUOLA DI SCIENZE

Corso di Laurea Magistrale in Informatica

Integrazione di strumenti di elaborazione
dei contenuti in un'applicazione
di filologia digitale

Relatore:
Chiar.mo Prof.
Fabio Vitali

Presentata da:
Chiara Minervino

Correlatore:
Chiar.ma Prof.
Francesca Tomasi

Sessione II
Anno Accademico 2017-2018

Alla mia famiglia.

Indice

Elenco delle figure	3
1 Introduzione	6
2 La filologia e le edizioni digitali	14
2.1 Il digitale come nuovo mezzo per il "vecchio"	16
2.2 Lo standard TEI	19
2.3 Edizioni digitali: lo stato dell'arte	20
3 Analisi del contesto	24
3.1 L'idea	24
3.2 Usabilità	25
3.3 Che cosa si intende per applicazione web	26
3.3.1 Storia delle applicazioni web	28
4 Il contesto di Voyeditor	30
4.1 Introduzione a Kwic Kwoc Kwac	30
4.1.1 Funzionalità dell'applicazione web	32
4.2 Introduzione a Voyant	33
4.3 Web Server Apache Tomcat	37
4.4 Lato server	39
4.5 Lato client	41
5 Voyeditor	43
5.1 Struttura del progetto KwicKwocKwac	43

5.2	Possibili modi per integrare Voyant in un ambiente web	44
5.3	Sviluppo delle funzionalità richieste	47
5.3.1	Come rendere i file dinamici	48
5.4	Wireframes	50
5.5	index.html	52
5.6	script.js	53
5.7	backend.js	55
5.8	Home page	57
5.9	Applicazione web finale	59
5.10	Modifiche apportate su Voyant ai fini dell'integrazione	60
5.11	Tool aggiunti	63
6	Funzionalità aggiunte attraverso Voyeditor	75
6.1	TextualArch	76
6.2	Cirrus	77
6.3	Parole nel documento	79
7	Conclusioni	81
	Risorse web	83
	Bibliografia	85

Elenco delle figure

2.1	esempio TEI	19
2.2	Vespasiano da Bisticci, Lettere	21
2.3	Philoeditor	22
2.4	phantoms home page	23
3.1	Sistema client-server	27
3.2	Funzionamento di una struttura client-server	29
4.1	significato Kwic Kwoc Kwac	31
4.2	Home attuale di KwickwocKwac	32
4.3	Home page Voyant	34
4.4	Voyant tool	35
4.5	Esempi Voyant tool	36
4.6	Home Tomcat	37
4.7	Home Tomcat	38
4.8	Lato server e lato client	39
4.9	Esempio di codice in Node.js	40
4.10	Esempio di documento html	42
4.11	Esempio codice javascript	42
5.1	Struttura progetto KwickwocKwac	44
5.2	Tools Voyant	45
5.3	Spiegazione Tools Voyant	46
5.4	come esportare un tool	47

5.5	come esportare un tool	47
5.6	prima ipotesi di implementazione	48
5.7	dove si trovano i file nel menu	49
5.8	configurazione da cambiare in Tomcat	50
5.9	dove posizionare i file	50
5.10	primo wireframe	51
5.11	secondo wireframe	52
5.12	codice per la creazione della home	53
5.13	voyant container	53
5.14	home	54
5.15	bottone home	54
5.16	voyant	55
5.17	layout	55
5.18	modifica backend	56
5.19	modifica backend	56
5.20	Home Page iniziale Kwic Kwoc Kwac	57
5.21	tasto Home	58
5.22	Home modificata	58
5.23	Home modificata	59
5.24	Home modificata	59
5.25	Home modificata	60
5.26	voyant.js	62
5.27	bubblelines	63
5.28	bubble	63
5.29	cirrus	64
5.30	link	64
5.31	Corpus Collocates	65
5.32	voyant.js	65
5.33	document terms	66
5.34	corpus terms	66
5.35	documents	67

5.36	nodi	67
5.37	loom	68
5.38	mandala	68
5.39	microsearch	69
5.40	phrases	69
5.41	scatter plot	70
5.42	stream graph	70
5.43	summary	71
5.44	termsberry	71
5.45	termsradio	72
5.46	textual arch	72
5.47	topics	73
5.48	trends	73
5.49	word tree	74
6.1	personaggio principale	77
6.2	personaggio secondario	77
6.3	nuvola di default	78
6.4	aggiornamento della stop list	78
6.5	nuvola modificata	79
6.6	individuazione parole rare	80

Capitolo 1

Introduzione

Questo progetto si pone come obiettivo la realizzazione di un'applicazione web per agevolare l'attività dei filologi e la ricerca fornendo strumenti efficienti per la modifica e l'analisi dei testi.

Nonostante le retrosie di alcuni filologi verso la digitalizzazione dei testi, è intento di questo lavoro offrire uno strumento innovativo per lo studio veloce e preciso dei testi letterari antichi. Una sfida che, nel corso della storia l'uomo si è trovato ad affrontare, per ciò che concerne la digitalizzazione di vari aspetti della vita: uno fra questi è la trasmissione della conoscenza e della cultura.

Il filologo oggi ha la possibilità di pubblicare su supporto digitale i risultati degli studi compiuti su un particolare testo, in quella che è comunemente chiamata edizione elettronica, uno strumento multimediale e ipertestuale per la ricerca accademica. Il testo in formato digitale, grazie alle varie possibilità di ricerca lessicale, permette di rintracciare legami difficili da cogliere, richiamare concetti che ritornano, evidenziare i richiami interni al testo con una precisione che la lettura del testo cartaceo non consente.

L'approccio verso il mondo tecnologico non è stato facile, i classicisti sono stati restii verso l'avvicinamento ad un nuovo uso del computer e non sempre propensi a cambiare e a tendere verso il mondo digitale, perché attenti a rispettare la sacralità del testo cartaceo.

La digitalizzazione porta con sé molti vantaggi, come la facilità con cui si può

condividere un file su cui si sta lavorando, ma anche molti svantaggi perchè per molti il web in generale è visto come una fonte non propriamente attendibile.

Essa risulta, però, fondamentale nell'ambito di quel processo in atto nella filologia che ha portato a cercare di oltrepassare la filologia tradizionale , mettendo in questione la figura dell'autore, riflessione iniziata alla fine degli anni sessanta con gli interventi di Barthes¹ e di Foucault e proseguita negli Stati Uniti dalla critica decostruzionista. E' stato così messo in discussione il criterio testamentario dell'ultima volontà dell'autore e all'idea di cercare il testimone migliore si è sostituita quella di dare pari dignità ai codici .

Ed è proprio il processo di moltiplicazione testuale che ne deriva ad essere agevolato e amplificato dalla possibilità di ricorrere ad una piattaforma digitale, assicurando maggiore flessibilità , possibilità di visualizzazioni simultanee e di rappresentazioni diverse.

Oggi si lavora molto su progetti che implementino la creazione di archivi elettronici in modo da poter dare dei vantaggi che riguardino sia la possibilità di poter lavorare su di un testo, sia la possibilità di garantire l'integrità delle opere. Si ottiene così il contenimento in spazi molto ridotti e a costi contenuti e la consultazione e la conservazione di grandi quantità di dati.

Il filologo non solo quindi ha bisogno di testi digitali da poter consultare, ma ha anche la necessità di lavorarci sopra, di modificarli e di dividerli. Nei primi anni della editoria elettronica non si è però badato tanto ad un problema fondamentale, la precoce obsolescenza dei prodotti di informatica umanistica, dovuta al rapido invecchiamento di apparecchiature e software, per cui molte banche dati presto sono diventate inutilizzabili.

A questo problema si può ovviare scegliendo un linguaggio di marcatura codificato a livello internazionale , durevole nel tempo e facilmente aggiornabile; uno standard di codifica al quale fare riferimento per la realizzazione di *open archive* realmente sostenibili e disponibili a lungo termine.

Per questa ragione per il filologo è necessario avere a disposizione dei tools informatici che permettano di aiutarlo, rendendo il lavoro e l'analisi dell'elaborato più

¹R. Barthes, *La mort de l'auteur* (1968)

efficiente, così da creare un'edizione digitale.

Bisogna però tener presente che gli strumenti non sono facili da utilizzare per un utente tipo, sia dal punto di vista dell'editore, le cui conoscenze informatiche spesso non vanno oltre la codifica del testo (XML, TEI, LATEX, etc.)², sia dal punto di vista dell'utente finale, che spesso si trova davanti a interfacce confusionarie e disorientanti, finendo per sfruttare solo una piccola parte delle informazioni e degli strumenti offerti.

Bisogna, quindi, fare in modo di creare un ambiente web che non solo permetta l'ausilio di tools informatici che aiutino l'analisi dei testi, ma che questi siano anche intuitivi e facili da utilizzare.

Con l'avvento dell'era digitale c'è stata un'ampia diffusione di progetti di digitalizzazione di documenti o di creazione di archivi e portali web per la fruizione degli stessi. Il filologo, dunque, ha la possibilità di svolgere studi e ricerche difficilmente realizzabili in maniera tradizionale ed è in grado di esplorare velocemente archivi, consultare documenti e stare in stretta collaborazione con altri ricercatori utilizzando nuovi strumenti di comunicazione come email e chat.

L'edizione digitale di un testo può essere arricchita grazie all'utilizzo di schemi di codifica testuale (HTML, SGML, XML, ecc.) che associano un insieme di caratteristiche o elementi che costituiscono il testo ad un insieme di simboli; si crea così un collegamento fra le strutture e le relazioni che esistono negli schemi e le relazioni sintattiche che esistono nei testi, il tutto regolato da una grammatica che ne governa l'uso.

In ambito internazionale si è anche creato un consorzio di istituzioni, la TEI (*Text Encoding Initiative*), che ha sviluppato un linguaggio di markup divenuto uno standard mondiale per la rappresentazione dei testi in formato digitale.

Il vantaggio dunque della digitalizzazione è quello di permettere facilmente la diffusione e la condivisione di un testo, rendendolo uno strumento ideale per una

²la proposta di un sistema di codifica internazionale TEI *Text Encoding Initiative* fondato su un metalinguaggio XML, che distingue con una certa accuratezza il momento della struttura da quello della rappresentazione, può consentire un rapido aggiornamento dei testi marcati in TEI in caso di cambiamento del metalinguaggio di riferimento. Questo sistema può essere applicato alla creazione di banche dati soprattutto per quei progetti filologici che possano sfruttare le possibilità di rappresentazione digitale multipla di un testo

comunità di ricercatori che abbia soprattutto necessità di condivisione. Con questa sensibilità, da diversi anni ci sono state delle collaborazioni fra il dipartimento di Informatica e quello di Filologia Classica ed Italianistica di questo Ateneo che hanno prodotto dei risultati interessanti che permettono di offrire grossi vantaggi. In conclusione, si cercherà di spiegare in che modo *Voyeditor*, la nuova applicazione creata e oggetto della presente tesi, sia riuscito ad integrare le funzionalità offerte da *KwicKwocKwac* e da *Voyant*, unendole, per migliorare e facilitare l'analisi in modo da soddisfare le aspettative di un filologo che ha necessità di uno strumento efficiente ed efficace.

La presente dissertazione continuerà come espresso di seguito in sintesi:

- il secondo capitolo si occuperà di definire una premessa sul rapporto tra la filologia e le edizioni digitali e cercherà di dare una panoramica generale sulla filologia digitalizzata e sulle forme in cui essa si è evoluta nel corso dell'intero decennio, tracciando delle linee guida per ripercorrere la storia e le tappe di questo sviluppo. Se il sistema di analisi di Lachmann è stato il modello principe del passato per rappresentare graficamente le relazioni tra i testi, una prima novità in campo filologico è avvenuta negli anni '60, '70 quando, per la collazione dei testi letterari, sono stati utilizzati dei calcolatori con lo scopo di automatizzare il processo lachmanniano, velocizzando il lavoro attraverso un algoritmo. L'elaborazione di alcuni software per la produzione di edizioni critiche è andata avanti sia attraverso la realizzazione di nuovi strumenti, che non prevedessero come unico output la stampa, sia con la pubblicazione elettronica e l'utilizzo di codifiche diverse (HTML, SGML, XML, ecc). A partire dagli anni novanta la filologia e le modalità di studio sono state rivoluzionate dal superamento dell'idea tradizionale di testo ultimo, per aprirsi al mondo dell'informatica e giungere alla produzione di qualcosa di più della semplice versione a stampa. Per la moltiplicazione digitale, la visualizzazione e il confronto dei testi è stato, perciò, possibile sfruttare al meglio le potenzialità offerte dallo sviluppo in un ambiente digitale non di un testo ultimo, ma di un ipotesto, un modello aperto all'influsso e alle collaborazioni provenienti dal web. La creazione di nuovi programmi e di

nuove applicazioni hanno, infine, cercato di rendere testi e immagini sempre più dinamici, manipolabili, revisionabili in rete, dando una nuova valenza alla figura dell'editore e al problema dell'autorizzazione a entrare a far parte di questo processo collaborativo di redazione di un documento. Da qui una breve definizione di TEI, le cui linee guida, per la codifica e l'interscambio di un testo elettronico, definiscono e documentano un linguaggio di markup per rappresentare le caratteristiche strutturali e concettuali dei testi digitali e, per concludere, alcuni brevi esempi di edizioni digitali:

- il primo esempio preso in considerazione è stato "Vespasiano da Bisticci, Lettere" [2], un progetto di ricerca del CRR-MM (Centro di Risorse per la Ricerca MultiMediale dell'Ateneo di Bologna) che prevede la realizzazione di un'edizione digitale ipertestuale delle lettere risalenti al XV secolo inviate e ricevute dal copista Vespasiano da Bisticci;
 - un esempio invece di editor testuale è *Philoeditor 3.0* che grazie alla creazione di edizioni *genetiche analitiche* permette lo studio di testi letterari su due livelli distinti, quello filologico e quello interpretativo;
 - un terzo esempio è *Phantoms*, un progetto nato per sviluppare un sistema web integrato con la visualizzazione e l'accesso ai servizi per la gestione dei testi.
- il terzo capitolo porrà le basi spiegando quale sia l'idea di partenza e si occuperà di fornire una piccola sintesi di ciò che è un'applicazione web. All'interno di un contesto accademico, la nuova applicazione Voyeditor intende proprio realizzare e implementare quella dinamicità, possibilità di revisione e manipolazione di cui si è parlato, offrendo funzionalità aggiuntive all'interno di un ambiente web preesistente. Indirizzandosi non solo al filologo e allo studioso, ma anche agli studenti, la nuova applicazione ha dovuto necessariamente contenere informazioni e servizi facilmente utilizzabili. L'usabilità è stata definita, pertanto, in corso di progettazione con il cercare di utilizzare sempre un linguaggio chiaro e assicurare una semplicità nella consultazione in tutti i nuovi contesti creati. Hanno concluso il capitolo una prima disamina su

cosa sia un'applicazione web, un'applicazione cioè accessibile via web per mezzo di un network, e una breve storia di come si siano diffuse e siano state implementate le prime applicazioni, in parallelo con il veloce sviluppo delle capacità operative delle macchine e dei linguaggi, fino alla versione HTML5. Un'affermazione delle applicazioni web che si è realizzata grazie alla facilità di accesso, perchè tali applicazioni sono indipendenti dall'hardware e dal sistema operativo utilizzati, al modo automatico con cui si verificano la distribuzione e l'aggiornamento delle stesse, alla riduzione dei costi di connettività e di gestione e alla possibilità che le nuove applicazioni hanno di crescere e svilupparsi in seguito alle esigenze dell'utente.

- il quarto capitolo si occuperà di spiegare in breve gli strumenti che sono stati necessari per la creazione del nuovo ambiente web, *Voyeditor*. Si è partiti dalla definizione dell'ambiente web all'interno del quale è nata la nuova applicazione, cioè *Kwic Kwoc Kwac*, di cui è stato spiegato l'origine del nome definendo gli indici che lo compongono; sono state elencate e analizzate le sue funzioni attraverso immagini e descrizioni specifiche, per poi passare a delineare le linee di sviluppo e le modalità operative di un altro ambiente web, *Voyant*, con esempi di alcuni tools presenti al suo interno: *Cirrus* e *Mandala*. Il capitolo si è poi concluso con un'introduzione al server web *Tomcat*, che è stato in sintesi descritto, e con un breve elenco delle tecnologie utilizzate.
- il quinto capitolo spiegherà i vari passi dell'implementazione e parlerà nello specifico dei tools che sono stati inseriti all'interno dell'applicazione web. Una volta mostrati i file che sono stati modificati, è stata indicata passo dopo passo graficamente la procedura da seguire per esportare un tool di *Voyant* attraverso un tag `<iframe>`. La nuova modalità operativa creata, però, non soddisfaceva pienamente le richieste in modo efficiente perchè non permetteva di poter creare un meccanismo generico. La mia attenzione si è concentrata dunque sulla risoluzione del problema, che ha trovato soluzione nell'aiuto del server web *Tomcat*, all'interno del quale sono stati inseriti i file e il progetto *Voyant*; sono stati allora mostrati nello specifico i vari passi

seguiti. Prima di procedere alla vera e propria modifica definitiva nel codice è stato necessario studiare la miglior soluzione possibile per disporre l'elemento, cercando sempre di essere fedele all'obiettivo prefissato, ma rispettando la sensibilità e le esigenze dell'utente. Il capitolo prende a questo scopo in esame i vari frammenti di codice che sono stati aggiunti, spiegando nel dettaglio la funzionalità aggiunta. *Voyeditor*, in conclusione, a seguito dell'implementazione dell'elemento esportato da *Voyant*, ha dovuto mostrare all'utente un' originale home page in modo da rendere la navigazione all'interno dell'ambiente più fluida.

- il sesto capitolo, infine, mostrerà i risultati ottenuti grazie alle funzionalità che sono state aggiunte.
 - un primo esempio operativo mostra i vantaggi ottenuti grazie all'utilizzo del tool *Textual Arch* che ha permesso di evidenziare all'interno di ellissi la diversa valenza e centralità di un personaggio principale come Cappuccetto Rosso, rispetto a quello secondario del lupo;
 - il secondo esempio mette, invece, in risalto i miglioramenti introdotti da *Cirrus*, che ha consentito di aggiungere la parola *article* all'interno della *stop list*, in quanto elemento non particolarmente significativo in un testo normativo, e dare maggiore centralità ai termini più significativi del testo in esame;
 - l'ultimo esempio ha, infine, proposto una diversa analisi possibile grazie a *Parole nel documento*, centrata sulla qualità e non sulla quantità dell'interpretazione delle frequenze che ha consentito di individuare all'interno di un testo lessemi rari e hapax.
- le conclusioni hanno evidenziato come le fasi di percorso e le modifiche apportate abbiano reso disponibile il nuovo ambiente web,

<http://chiara.minervino.tw.cs.unibo.it/Voyeditor/>

concretizzando quanto ci si era prefissati in premessa, cioè offrire in ambito accademico una piattaforma arricchita da nuove funzionalità. Questo lavoro

suggerisce ulteriori idee di approfondimento, rimanendo aperto ad innovazioni e aggiunte che potranno essere inserite, ponendosi come punto di partenza e offrendo spunti a progetti di studio ed implementazioni future.

Capitolo 2

La filologia e le edizioni digitali

A partire dagli anni '90 si è iniziata ad insediare nel mondo accademico editoriale la digitalizzazione. Questo è ancora oggi un argomento molto discusso che porta con sé pro e contro in quanto l'adozione di un computer in questo settore porta con sé tanti cambiamenti, ma anche tante ritrosie e riserve.

computers are proving to be far more than just electronic research assistants able to offer relief from 'idiotic' work; in fact they are leading us to question the hermeneutics and heuristics of textual scholarship.

[9]

La filologia digitale, cioè l'utilizzo di strumenti e tecniche informatiche per l'elaborazione di edizioni critiche, si è costituita come disciplina nell'ultimo decennio per rispondere all'esigenza di dipartimenti umanistici di molte università che, dotate di strutture adibite all'informatizzazione dei processi didattici e di ricerca, intendono realizzare ciò che oggi è reso disponibile dall'informatica.

Nel momento in cui un testo passa dal materiale all'immateriale, cioè da prodotto cartaceo a prodotto il cui peso è misurato in bit, avviene la codifica, che è di per sé un processo interpretativo sia quando ci si occupa dell'aspetto materiale del segno sia quando si analizza la struttura del testo; essa si basa su sistemi di marcatura, markup¹, che descrivono l'aspetto dei singoli elementi testuali utilizzati da TEI.

¹i linguaggi di markup più utilizzati sono SGML, Standard Generalized Markup Language e XML Extensible Markup Language

Indipendentemente dalla finalità di ricostruire l'originale da cui deriva, l'edizione critica rivaluta quelli che la critica classica di tipo lachmaniano ² definiva "errori" come "innovazioni" di cui esplorare il contesto storico, le motivazioni culturali e le ragioni testuali. Si creano così i presupposti per un'edizione critica intesa come raccolta di materiali diversi, come percorso di un testo di cui sia scandita la storia della sua trasformazione e attualizzazione. ³

Il grande vantaggio che ha reso la diffusione di questo fenomeno così importante è principalmente la disponibilità che deriva: ogni utente in qualsiasi momento può accedere alle fonti che vengono pubblicate e quindi sfruttare al meglio i dati elettronici in diversi modi. Ma ha di fatto determinato la crisi autoriale: nel momento in cui un documento assume una forma digitale può essere copiato, modificato, riutilizzato da altri.

Il ruolo dell'editor rimane, invece, indefinito davanti ad una sfilza di documenti che hanno bisogno di essere giudicati, valutati ed interpretati. Shillingsburg nel suo ultimo contributo afferma che non siamo ancora in grado di sfruttare tutte le potenzialità che la digitalizzazione può offrire e aggiunge che le edizioni digitali sono migliori ma non ancora abbastanza.

'Yes they are better. No, they are not good enough'

La questione di fondo è quella di non cercare di imitare la versione stampata nei suoi minimi dettagli, ma di fornire qualcosa in più. Il problema principale con gli artefatti digitali è la loro stabilità e longevità. Il digitale è fragile ed effimero, c'è il rischio che si possa perdere il duro lavoro dedicato alla creazione di una risorsa digitale.

Per questo motivo spesso molti studiosi non inseriscono all'interno della loro ricerca

²il metodo di Lachmann è lo strumento indispensabile ai fini della pubblicazione dell'edizione critica; fu teorizzato dal filologo tedesco Karl Lachmann che, nel 1850 con la sua celebre edizione del *De rerum natura* di Lucrezio, definisce delle procedure per quanto possibile oggettive e meccaniche, divise in fasi, che portano alla ricostruzione dello *stemma codicum* attraverso l'eliminazione di codici ritenuti copie sulla base della individuazione di "errori" A.Stussi, *Breve avviamento alla filologia italiana*, Bologna 2002, Il Mulino p.73-83

³la prospettiva semiologica di Segre e Avalle si basa sulla nuova modalità di analizzare un percorso di un testo non solo nelle sue fasi di formazione, creazione e scrittura, ma anche di diffusione, riscrittura, copia, nei vari "diasistemi", sistemi cioè intermedi tra modello e copia

le risorse digitali anche se queste dovessero risultare particolarmente significative. La filologia ha, insomma, cercato nuove strade, sia rivalutando il ruolo di altre figure che partecipano alla tradizione del testo sia superando l'idea di testo definitivo, un testo ultimo che superi i precedenti stadi testuali. E' proprio nell'ambiente digitale, con la moltiplicazione testuale, la visualizzazione e il confronto dei testi, che si concretizza il superamento dell'idea del testo definitivo: il testo del filologo non è più un testo univoco, ma un ipotesto, in quanto è necessario seguire il percorso di un testo non solo nelle sue fasi di formazione e diffusione, ma anche di riscrittura e riadattamento: un compito che il supporto digitale consente di realizzare nello spazio immenso del web e grazie alla capacità di conservare una quantità enorme di dati.

L'operazione di codifica si trasforma nell'operazione di allestire un modello, un ipotesto appunto, con la consapevolezza dell'insufficienza di un modello limitato alla sola dimensione linguistico-tipografica, ma aperto all'interattività e alle forme di collaborazione offerte dalla comunicazione digitale.

2.1 Il digitale come nuovo mezzo per il "vecchio"

Per poter contribuire alla realizzazione di una piattaforma che si occupi di digitalizzare delle opere, si deve prima prendere in analisi il ruolo dell'editore. Il ruolo dell'editore è quello di gestire le varie opere e di dare veridicità ad esse. Secondo la teoria di McGann⁴ viene sottolineato il fatto che il testo non sia solo un prodotto di ciò che l'autore voleva esprimere e il frutto delle sue idee, ma il risultato di una per così dire revisione da parte di molti agenti, tra cui per esempio l'editore, che hanno in qualche modo contribuito e a molti processi materiali come la pubblicazione.

"The signifying process of the work become increasingly collaborative and socialized"

⁴Jerome McGann, Siemens et al., 2010

⁵ Quindi, nasce questo concetto di testo come processo collaborativo perchè appunto l'editore non si può occupare di ogni aspetto in prima persona. Ogni attestazione o interpretazione acquista di per sè valore come testimone di un'intenzionalità non meno valida di quella dell'autore.

Ma non si tratta solo della redazione di un documento, la parte ritenuta più importante dagli storici è la parte critica del testo. Viene riconosciuto che gli ipertesti ⁶ riescano ad esprimere al meglio sia i testing documentari sia l'editing critico e, quindi, a supportare entrambe le attività. Sulla base di queste teorie, il mondo accademico si chiede se il digitale sia semplicemente un nuovo mezzo per metodi "vecchi" o se sia una metodologia completamente innovativa. E' importante prendere in considerazione il lavoro editoriale, di cui abbiamo parlato prima, e l'importanza che esso ha.

La prima cosa da capire è cosa si intende esattamente per digital editing, se un testo per essere definito tale deve semplicemente essere pubblicato su una piattaforma o se entrano in scena altri aspetti. La vera e propria distinzione tra pensiero classico e pensiero digitale potrebbe risiedere nel tipo di flusso di lavoro adottato, nel tipo di output prodotto e nelle idee degli editori.

L'editing come processo assistito dal computer mirava inizialmente a fornire un aiuto e a semplificare il lavoro editoriale tradizionale, così nasce l'idea che gli editori possano usare il computer per accelerare il loro flusso di lavoro, senza però cambiare la natura di esso.

Il testo digitalizzato è sicuramente un concetto diverso rispetto al testo tradizionale cartaceo. Si deve sempre tenere a mente il fatto che il testo digitalizzato non può essere stampato mantenendo tutte le sue funzionalità. L'edizione digitale è guidata da un paradigma diverso che non è limitato allo spazio della "pagina". La nascita del web ha dato una nuova visione della sfera digitale, ponendo l'accento sugli utenti e su come interagiscano tra di loro, creando così le basi per l'edizione collaborativa e sociale.

L'edizione digitale segna quindi una svolta, perchè non si tratta solo di rendere un testo digitalizzato, ma di permettere alla comunità di contribuire attraverso

⁵ibidem nota 4

⁶ipertesti nella terminologia del 1995 indicavano le edizioni digitali

annotazioni, traduzioni, commenti, di inserire nuovi testi o di modificare quelli esistenti.

Ci sono state molte critiche riguardo al social editing perchè molti studiosi ritenevano che ci dovessero essere delle regole e dei modi per poter decidere chi avesse le capacità di poter partecipare e, quindi, il potere di modificare il testo. Sono stati così sperimentati diversi modelli che mettono in pratica questo nuovo approccio, permettendo ad un individuo di modificare il testo solo nel momento il cui le sue credenziali e competenze siano state accettate.

Il crowdsourcing è definito come l'uso di strumenti di social media per consentire agli utenti di contribuire a uno specifico progetto o piattaforma con nuovi contenuti. Uno degli esempi più famosi è Wikipedia, che permette a chiunque di poter contribuire alla diffusione di informazioni. In generale è stato impiegato con grande successo in attività prettamente scientifiche, mentre in ambito letterario presenta un ostacolo all'affermarsi dovuto alla difficoltà di verificare che le modifiche siano state effettuate in base a competenze.

Il crowdsourcing mira a fornire un'apertura verso una comunità, ma allo stesso tempo cerca di mantenere un controllo su ciò che viene pubblicato, mantenendo l'importanza del ruolo che ha l'editore. Affinchè questo approccio funzioni, le attività richieste agli utenti devono essere intuitive, facili da capire, gestibili in una piccola porzione di tempo e dare una ricompensa immediata. In questo modo, non solo le persone si impegnano in un progetto, ma forniscono ad esso un contributo. Le edizioni accademiche digitali si basano sul markup, espresso per la maggior parte in XML; nel momento in cui si adottano queste tecnologie, gli editori aggiungono tag al loro testo per effettuare delle annotazioni che vengono interpretate da uno script e trasformate in un formato convenzionale per la visualizzazione, generalmente TEI. Un testo così descritto sarà diviso nelle seguenti parti:

- un file sorgente che conterrà il testo trascritto, generalmente in formato XML che al momento rappresenta il modo migliore per trasmettere la codifica del testo;
- un insieme di script che provvederà a rendere visibile il testo;

- una serie di file di stile, una combinazione tra CSS, Javascript e HTML , che lavorino sull'output.

Dovremmo distinguere però tra edizioni digitali ed edizioni digitalizzate in quanto un'edizione può essere veramente definita digitale se, stampandola, non si ha una perdita di funzionalità; ma la versione stampata non sarà mai in grado di contenere le diverse sfaccettature offerte dalla versione online.

Generalmente le edizioni digitalizzate sono in grado di fornire delle funzionalità extra che vanno al di là della pura e semplice trascrizione.

2.2 Lo standard TEI

```

1 <TEI xmlns="http://www.tei-c.org/ns/1.0">
2 <teiHeader>
3 <fileDesc>
4 <titleStmt>
5 <title>Death row inmate's fight for his life shines light on use of jailhouse informants </title>
6 </titleStmt>
7 <publicationStmt>
8 <p>Converted into TEI by "Kwiç Kwoc Kwac 1.0" on 29/11/2020, 18:39:56 from the original source at "https://abcnews.go.com/US/death-row-
inmates-fight-life-shines-light-jailhouse/story?id=68357003". </p>
9 </publicationStmt>
10 <sourceDesc>
11 <listPerson>
12 <person xml:id="ChristinaCarrega">
13 <persName>Christina Carrega</persName>
14 </person>
15 </listPerson>
16 </sourceDesc>
17 </fileDesc>
18 </teiHeader>
19 <text>
20 <body>
21 <div xml:id="work">
22 <head>Death row inmate's fight for his life shines light on use of jailhouse informants </head>
23
24 <head>Some states seek to change how prosecutors evaluate jailhouse informants.</head>
25 </body>
26 </text>
27 </TEI>

```

Figura 2.1: esempio TEI

Lo schema di codifica della TEI (*Text Encoding Initiative*) può essere adottato da tutti coloro, in particolare umanisti, che intendano produrre e diffondere testi in formato elettronico. La finalità scientifica e di ricerca dello schema consente di rappresentare la struttura astratta di varie tipologie testuali e le connesse

caratteristiche testuali rilevanti. Inizialmente basato sul linguaggio SGML , *Standard Generalized Markup Language*, ha poi utilizzato il più recente e diffuso XML, *Extensible Markup Language*. Obiettivo della codifica TEI è:

- avere delle linee guida semplici, chiare e concrete;
- essere di semplice utilizzazione per i ricercatori , senza il ricorso a software specializzati;
- permettere una definizione rigorosa e un'efficiente elaborazione dei testi;
- consentire estensioni definite dall'utente;
- essere conformi agli standard esistenti o in procinto di essere adottati.

2.3 Edizioni digitali: lo stato dell'arte

Riprendendo quanto anticipato nell'introduzione, verranno ora prodotti alcuni esempi su come creare un ambiente web che non solo permetta l'ausilio di tools informatici che aiutino l'analisi dei testi, ma che questi siano anche intuitivi e facili da utilizzare. Il rischio da evitare è quello di costruire una letteratura per filologi, per soli esperti; scopo dei progetti analizzati è quello invece di proporre edizioni digitali fruibili ai più , siano essi studenti o un lettore medio, non specialista.

Vespasiano

Nell'ambito delle sfide poste dalle trasformazioni tecnologiche, dall'introduzione delle stesse in campo umanistico e dalla conseguente apertura verso nuove prospettive e nuove pratiche di ricerca, di didattica e di condivisione del sapere, è stato avviato un progetto per rendere possibile e più agevole la fruizione di documenti che, messi in rete, possano essere letti e analizzati attraverso percorsi e strumenti di ricerca nuovi.

A partire da una riflessione profonda sulla natura dello strumento digitale e della sua applicabilità ad ambiti che rispondano alle esigenze delle discipline umanistiche,

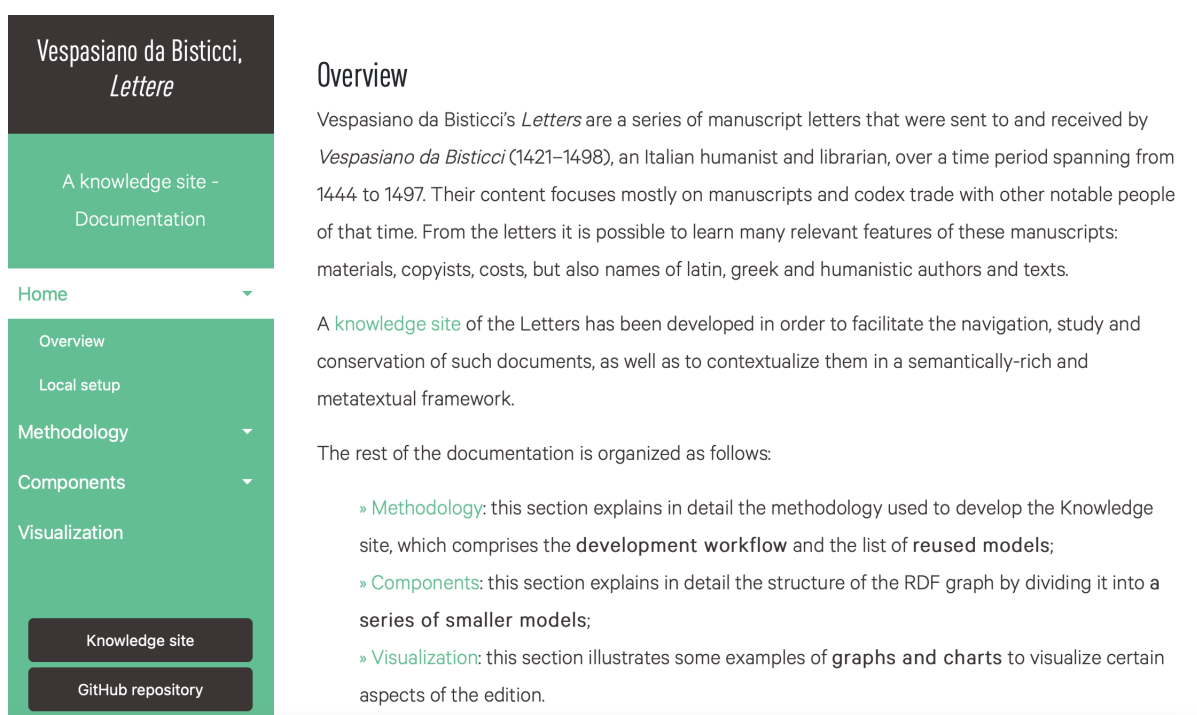


Figura 2.2: Vespasiano da Bisticci, Lettere

il progetto di ricerca *Vespasiano da Bisticci Lettere* si inserisce nell'ambito dell'attività del CRR-MM (Centro di Risorse per la Ricerca MultiMediale dell'Ateneo di Bologna) e propone un'edizione digitale delle Lettere di Vespasiano da Bisticci, risalenti al XV secolo, inviate e ricevute dal copista, alle quali è possibile accedere attraverso alcune faccette tematiche che permettono di scegliere l'ambito in cui calarsi: luogo, data, corrispondente, segnatura.

Il tutto corredato da:

- strumenti filologici, come indici tematici, tavole sinottiche, note, descrizione delle fonti;
- collegamenti contestuali per l'approfondimento e la critica alla raccolta, ai corrispondenti, ai manoscritti, ai copisti e alle biblioteche.

L'edizione risponde pienamente all'esigenza moderna di essere aperto all'interattività e alle forme di collaborazione offerte dalla comunicazione digitale e si presenta

come un archivio in costante arricchimento ed evoluzione. Ogni lettera è marcata in formato XML/TEI, secondo un modello di markup ad hoc, i cui elementi significativi sono indicizzati.

Philoeditor

Un editor testuale è un software o un'applicazione o un programma per comporre o modificare testi, per scrivere in codice HTML con funzionalità aggiuntive quali ad esempio digitare un tag per avviare una funzione o distinguere tag con colori diversi etc.

Nel campo degli editor testuali di ambito filologico emerge il progetto *PhiloEditor 3.0* che, a supporto del lavoro del filologo, crea nuovi tipi di edizioni definite *genetiche analitiche* le quali propongono un modo nuovo di studiare i testi: permettono di mostrare e tracciare le modifiche di un testo in due versioni successive.

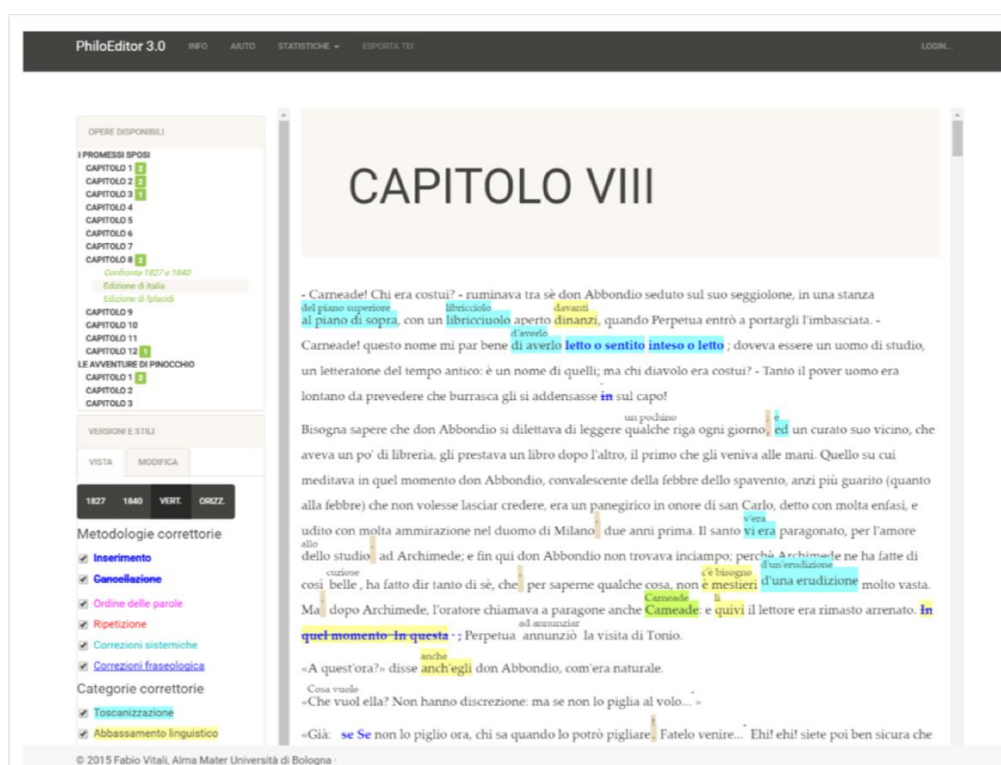


Figura 2.3: Philoeditor

Phantoms

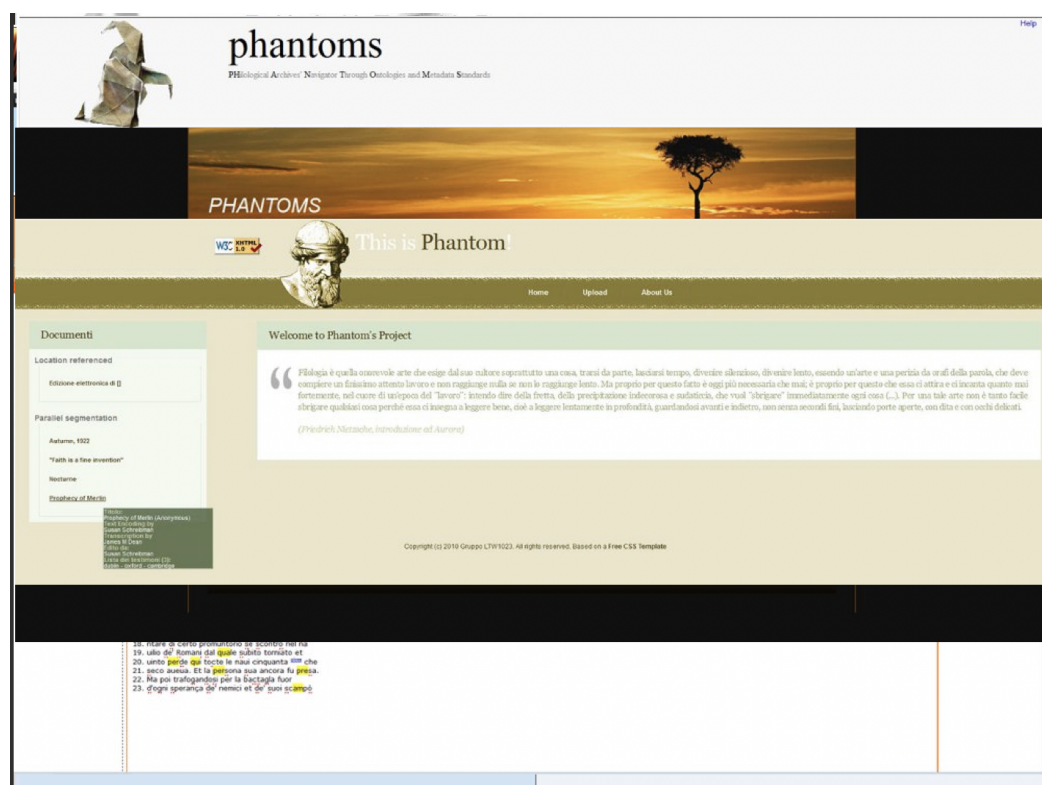


Figura 2.4: phantoms home page

Sono ormai numerosi i testi in formato XML/TEI della tradizione letteraria e culturale italiana che è possibile trovare in rete su siti Web . Il Progetto *Phantoms* nasce dalla necessità di sviluppare un sistema web integrato per la visualizzazione e l'accesso ai servizi per la gestione di testi letterari in formato XML/TEI. A partire da un catalogo, fornito da un provider SP, elemento fondamentale del protocollo, si può scegliere il servizio da richiamare.

Il catalogo è un documento XML diviso in due parti: un set di metadati comuni a tutti i servizi e una sequenza di descrittori di servizio. In seguito ad una richiesta, il provider SP dà una risposta http.

Capitolo 3

Analisi del contesto

La creazione di un nuovo strumento digitale *Voyeditor* deve avere delle caratteristiche fondamentali che si adattino alla tipologia di utente di riferimento.

Lo scopo è quello di implementare un'applicazione web partendo da una già esistente, chiamata *KwicKwocKwac*, fornendo dei servizi supplementari dati da *Voyant*, che siano utili, ma che allo stesso tempo siano anche facili da usare. L'aspetto principale da prendere in considerazione, quando si crea un ambiente utilizzato da un utente tipo, è considerare che qualsiasi passaggio deve essere intuitivo, l'utente deve trovarsi davanti alla pagina iniziale e deve trovare facilmente quello che cerca. Sulla base di ciò che viene descritto come social editing, si vuole permettere ad una comunità accademica di poter lavorare su un determinato testo in modo che tutti possano facilmente modificarlo, ma anche condividerlo.

3.1 L'idea

L'ambiente web *Voyeditor* che andrò a sviluppare è stato creato per essere utilizzato in un contesto accademico. Le principali finalità sono:

- lavorare su un testo modificandone facilmente le singole istanze;
- avere la possibilità di ricevere in output un documento in formato standard facilmente condivisibile;

- avere a disposizione dei tools che aiutino la semplice analisi e forniscano dati statistici relativi al testo preso in considerazione.

L'obiettivo prefissato è quello di permettere al filologo che utilizza l'applicazione web *Voyeditor* di avere delle funzionalità aggiuntive oltre quelle già date da *Kwic-KwocKwac*, implementando ed inserendo i tools di Voyant in modo da agevolare il lavoro sui testi e la relativa analisi.

L'implementazione deve non solo permettere di migliorare notevolmente l'elaborazione dell'analisi del testo, aggiungendo tools che siano sia utili che creativi, ma deve anche essere facile da usare.

3.2 Usabilità

Per usabilità si intende la proprietà di un oggetto di essere facilmente captato, utilizzato e gradito. In questo caso un prodotto software si definisce usabile se è adeguato ai bisogni e alle aspettative dell'utente, con un funzionamento che sia intuitivo, facilmente apprendibile ed esteticamente piacevole. Lo standard ISO 9241¹ definisce l'usabilità come:

"La misura in cui un prodotto può essere usato da specifici utenti per raggiungere determinati obiettivi con efficacia, efficienza e soddisfazione in uno specifico contesto d'uso."

Gli aspetti rilevanti dell'usabilità sono:

- *disponibilità e accessibilità*: il sito deve funzionare, altrimenti non è utilizzabile; è importante assicurarsi che non vi siano messaggi di errore, controllare che i link di riferimento siano funzionanti e assicurarsi che il sito sia visibile in modo ottimale su qualsiasi schermo di qualsivoglia dimensione;
- *chiarezza*: l'utente non deve essere distratto da informazioni che non siano necessarie, altrimenti potrebbe impiegare troppo tempo a trovare quello che

¹«Usability: extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.» da ISO 9241-11:1998(en)

sta cercando. Si mira alla ricerca del soddisfacimento dell'utente che vuole eseguire il suo goal il più velocemente possibile;

- *apprendibilità*: indica la capacità dell'utente di operare a livelli definiti di competenza dopo un determinato periodo di navigazione e, quindi, la facilità con la quale comprende ed impara ad utilizzare il sito. Progettare interfacce intuitive è molto importante per non mettere in difficoltà l'utente che, in caso contrario, si vedrebbe costretto a consultare le guide e a perdere tempo. La maggior parte degli utenti ha familiarità con la tecnologia e questo rende più facile fornire un'interfaccia intuitiva che sia facile da imparare;
- *credibilità*: fare in modo che l'utente non dubiti delle informazioni che trova all'interno del sito web, in questa fase si cercano di evitare gli errori;
- *rilevanza* il sito web deve essere user-friendly.

3.3 Che cosa si intende per applicazione web

Un' applicazione web è una applicazione client/server per un ambiente accessibile via web che utilizza le tecnologie internet e indica tutte le applicazioni distribuite web-based, cioè quelle applicazioni che non risiedono direttamente sulle macchine che le usano, ma su server remoti che potrebbero essere dall'altra parte del pianeta. Lo scopo dell'applicazione web è quello di fornire servizi all'utente che siano efficienti, ma anche facili da usare.

Pertanto, saper programmare per il web significa conoscere i diversi meccanismi e strumenti per conservare o passare i dati, detti parametri, tra le diverse pagine dell'applicazione web. In pratica, un'applicazione web è un programma che non necessita di essere installato nel computer, in quanto esso si rende disponibile su un server in rete e può essere fatto funzionare attraverso un normale Web browser (es. Google Chrome, Mozilla Firefox, Opera, ecc.) in posizione di client.

Il client, dopo aver instaurato una connessione con il server, invia la richiesta per un servizio; il server dopo aver elaborato i dati necessari rende disponibile al client il servizio richiesto. A differenza dei siti web statici (HTML), l'applicazione web

viene realizzata con una o più tecnologie (PHP, Ajax, Servlet, Database ecc.) che permettono la creazione di un sito dinamico, cioè di un sito nel quale il contenuto delle pagine varia durante l'interazione.

L'utilità di un sistema in remoto sta nel fatto che bisognerebbe preoccuparsi solo di usare il software, non della sua manutenzione, nè del suo backup. Oltre a questo, un'applicazione web può essere usata da qualunque postazione nel mondo: di fatto, in ogni angolo del pianeta, con una macchina con un browser installato, sarà come essere nel proprio ufficio. Le applicazioni web si pongono come valida alternativa

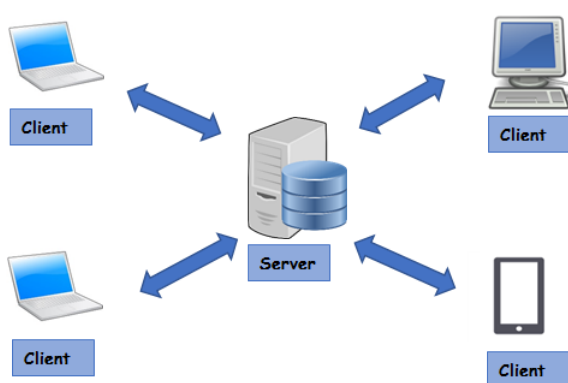


Figura 3.1: Sistema client-server

alle tradizionali applicazioni Client/Server per vari motivi:

- facilità di distribuzione e aggiornamento: un'applicazione web risiede interamente sul server, per cui la sua pubblicazione coincide con la distribuzione e l'aggiornamento ed è automaticamente disponibile a tutti gli utenti;
- accesso multiplo e facile: l'accesso all'applicazione è indipendente dall'hardware e dal sistema operativo utilizzato dagli utenti;
- senza costi: l'uso di Internet come infrastruttura per un'applicazione web riduce notevolmente sia i costi di connettività che i costi di gestione dei client.

Mentre una pagina Web si limita a visualizzare un contenuto statico e permette all'utente di spostarsi al suo interno, un'applicazione web offre agli utenti un'esperienza più interattiva.

3.3.1 Storia delle applicazioni web

Quando cominciarono a diffondersi le prime applicazioni client-server, a partire dalla metà degli anni Ottanta, c'erano ancora molte limitazioni che riguardavano il fatto che non si potesse ottenere l'interoperabilità che si ha oggi e che rappresenta uno dei maggiori vantaggi. Ogni macchina doveva avere la parte client installata che andava periodicamente aggiornata e sorgevano quindi dei problemi tra i differenti tipi di sistemi per le differenti macchine.

I continui aggiornamenti necessari al funzionamento alzavano notevolmente i costi per il produttore e rendevano l'applicazione non versatile, quindi, si limitava l'applicazione all'uso esclusivo su un solo sistema operativo.

Nelle prime applicazioni web il client navigava attraverso una sequenza di pagine, ricevute sotto forma di documenti Web statici. Ed era proprio questa sequenza di pagine visitate a fornire una forma di esperienza interattiva. Ogni volta che la pagina web richiesta veniva modificata, il server veniva chiamato in causa per effettuare il refresh della pagina stessa.

A partire dagli anni 90 con la nascita del Web, non si è stati più costretti ad eseguire gli aggiornamenti grazie alla dinamicità offerta dalle pagine web e sono abbattuti i costi per il produttore. Si dà il via così ad una standardizzazione fornita da HTML e Javascript . Questi anni hanno segnato quindi la nascita di un client universale per tutte le applicazioni web perchè attraverso il Web vengono ricevute le pagine dal server, interpretate e visualizzate sullo schermo. Nel 1996 Macromedia introduce Flash, un plug-in che poteva essere incluso nel browser per la realizzazione di animazioni sulle pagine web e per l'arricchimento dell'interfaccia utente, evitando di comunicare con il server. Anche in questo caso si utilizzava un linguaggio di scripting client-side. Nel 2005 ci fu un'altra svolta fondamentale data dall'introduzione di Ajax, un insieme di tecniche per lo sviluppo web, che permetteva di creare delle applicazione web asincrone e con delle parti client via via più interattive. Questo era possibile utilizzando degli script client-side che contattavano il server per effettuare il download di dati, senza dover ricaricare l'intera web page. Nel 2011 è stato ultimato HTML5, tramite il quale si possono realizzare interfacce utente più ricche di animazioni ,senza il

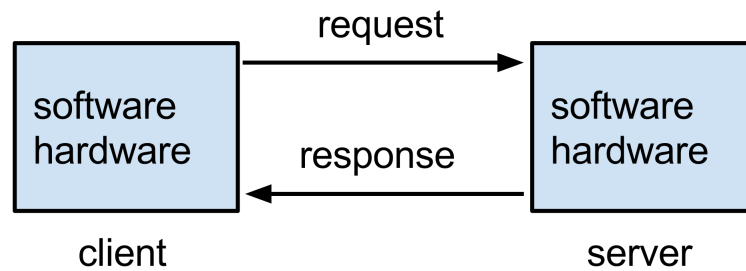


Figura 3.2: Funzionamento di una struttura client-server

bisogno di utilizzare plug-in client-side (come Flash). HTML5 introduce nuovi elementi che includono e gestiscono in modo nativo i contenuti multimediali e grafici all'interno delle web page. Nuovi elementi vengono definiti anche per strutturare in modo migliore i documenti. HTML5 porta con sé anche delle API tramite le quali si possono sviluppare web application molto complesse. Il processo client è tipicamente dedicato ad interagire con l'utente finale; esso svolge un ruolo attivo, in quanto genera autonomamente richieste di servizi, mentre il processo server è reattivo perché svolge una computazione solo a seguito di una richiesta da parte di un qualunque client ed è sempre in esecuzione per essere pronto a rispondere alle richieste.

Capitolo 4

Il contesto di Voyeditor

Come è stato anticipato nel capitolo precedente, il mio lavoro è partito da una versione dell'applicazione web preesistente chiamata *Kwic Kwoc Kwac* a cui sono stati aggiunti i tools dal sito web Voyant.¹

In questa sezione verrà fatta una piccola presentazione dell'applicazione web e di *Voyant*, spiegando quello che forniscono separatamente, prima di mostrare quali risultati ho ottenuto unendo i due strumenti.

4.1 Introduzione a Kwic Kwoc Kwac

KwicKwocKwac rappresenta l'ambiente web da cui parte il mio lavoro, esso presenta una serie di funzioni che verranno brevemente elencate ed analizzate in modo da considerare nella sua complessità i risultati ottenuti.

Come indicato nella documentazione:

KwicKwocKwac è un ambiente Web con l'obiettivo di fornire ai ricercatori uno strumento semplice e intuitivo per arricchire il testo di documenti in formato digitale attraverso funzionalità di marcatura semi-automatica e metadatazione.

¹<https://voyant-tools.org>

Che cosa significa KwicKwocKwac?

Kwic, *Kwoc* e *Kwac* sono degli indici, conati negli anni Sessanta, con lo scopo di aiutare nella ricerca delle parole chiave e ad avere un elenco in ordine alfabetico, rispettando il contesto al quale appartengono.

Questi indici non solo venivano utilizzati per il recupero delle informazioni, ma anche per l'analisi del contenuto. Era un utile metodo di indicizzazione per i manuali tecnici prima che la ricerca computerizzata del testo completo diventasse comune.

Nella figura 4.1 viene mostrato un esempio di come dovrebbe apparire il testo successivamente all'applicazione dell'indice.

Title	Catalognummer	Keyword	Title	Catalognummer	Title	Catalognummer
Office 2003 Timesaving Techniques for Dummies	180344	Excel	Beginning Excel: What If Data Analysis Tools	8467	Dummies, Office 2003 Timesaving Techniques for	180344
Excel for Scientists and Engineers	2704	Excel	Creating Spreadsheets and Charts in Microsoft Office Excel 2007	203	Engineers, Excel for Scientists and	2704
How To Do Everything With Microsoft Office Excel 2003	904	Excel	Data Analysis Using SQL and Excel	12377	Everything With Microsoft Office Excel 2003, How To Do	904
How To Do Everything With Microsoft Office PowerPoint 2003	900	Excel	Excel 2003 Formulas	708	Everything With Microsoft Office PowerPoint 2003, How To Do	900
Data Analysis Using SQL and Excel	12377	Excel	Excel 2003 VBA Programmer's Reference	3089	Excel 2003 Formulas	708
Automated Data Analysis Using Excel	3434	Excel	Excel for Scientists and Engineers	2704	Excel 2003 Inside Out, Microsoft Office	18765
How To Do Everything With Microsoft Office Excel 2003	904	Excel	How To Do Everything With Microsoft Office Excel 2003	904	Excel 2003 Programming with VBA, Mastering	7655
Accessing and Analyzing Data with Excel 2003	5	Excel	Mastering Excel 2003 Programming with VBA	7655	Excel 2007, Creating Spreadsheets and Charts in Microsoft Office	203
Excel 2003 Formulas	708	Excel	Microsoft Office Excel 2003 Inside Out	18765	Excel for Scientists and Engineers	2704
Microsoft Office Excel 2003 Inside Out	18765	Fixing	Fixing PowerPoint Annoyances	5575	Excel, Automated Data Analysis Using	3434
Mastering Excel 2003 Programming with VBA	7655	Formulas	Excel 2003 Formulas	708	Excel, Data Analysis Using SQL and	12377
Excel 2003 VBA Programmer's Reference	3089	Guide	Microsoft Visio 2003 User's Guide	32081	Excel: What If Data Analysis Tools, Beginning	8467
Creating Spreadsheets and Charts in Microsoft Office Excel 2007	203	How	How To Do Everything With Microsoft Office Excel 2003	904	Fixing PowerPoint Annoyances	5575
Excel for Scientists and Engineers	2704	How	How To Do Everything With Microsoft Office PowerPoint 2003	900	Formulas, Excel 2003	708
Beginning Excel: What If Data Analysis Tools	8467	Inside	Microsoft Office Excel 2003 Inside Out	18765		
Fixing PowerPoint Annoyances	5575	Mastering	Mastering Excel 2003 Programming with VBA	7655		
Excel 2003 Formulas	708	Microsoft	Creating Spreadsheets and Charts in Microsoft Office Excel 2007	203		
Microsoft Visio 2003 User's Guide	32081	Microsoft	How To Do Everything With Microsoft Office Excel 2003	904		
How To Do Everything With Microsoft Office Excel 2003	904	Microsoft	How To Do Everything With Microsoft Office PowerPoint 2003	900		
How To Do Everything With Microsoft Office PowerPoint	900	Microsoft	Microsoft Office Excel 2003 Inside Out	18765		
Microsoft Office Excel 2003 Inside Out	18765	Microsoft	Microsoft Visio 2003 User's Guide	32081		
Mastering Excel 2003 Programming with VBA	7655	Office	Creating Spreadsheets and Charts in Microsoft Office Excel 2007	203		
How To Do Everything With Microsoft Office Excel 2003	904	Office	How To Do Everything With Microsoft Office Excel 2003	904		
		Office	How To Do Everything With Microsoft Office PowerPoint 2003	900		
		Office	Microsoft Office Excel 2003 Inside Out	18765		

Figura 4.1: significato Kwic Kwoc Kwac

- **Kwic** sta per *parola chiave nel contesto*, è formato ordinando e allineando le parole all'interno del titolo di un articolo per consentire a ciascuna parola nei titoli di essere ricercabile alfabeticamente nell'indice;
- **Kwac** sta per *parola chiave accanto al contesto*, è una modifica di KWIC, fornisce ulteriori parole chiave, tratte dall'abstract o dal testo originale del documento, che vengono inserite nel titolo per fornire ulteriori voci di indice;
- **Kwoc** sta per *parola chiave fuori contesto*, visualizza anche il termine di accesso a sinistra, ma le coppie di parole non vengono conservate nella sequenza alfanumerica di parole chiave.

4.1.1 Funzionalità dell'applicazione web

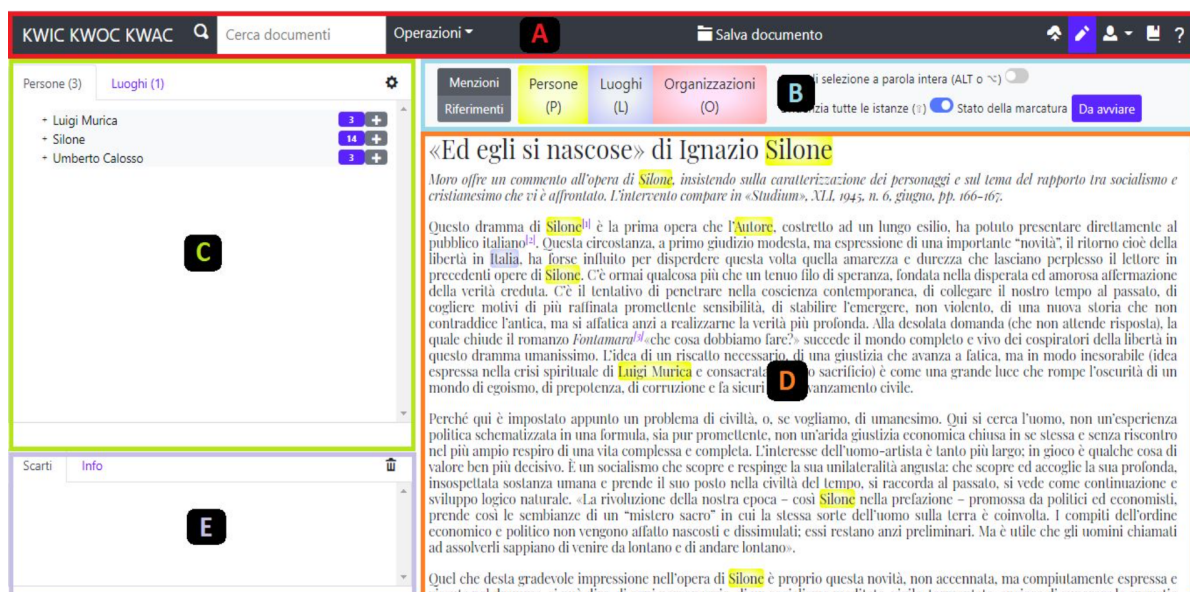


Figura 4.2: Home attuale di KwicKwocKwac

L'immagine mostra come si presenta l'ambiente web, esso è diviso in varie sezioni che hanno diverse funzionalità, le più importanti secondo la documentazione sono:

- la possibilità di marcare gli elementi presenti nel testo, scegliendo un ambiente generico come persone o luoghi;
- il riconoscimento automatico delle tipologie di note presenti nel testo (note del ricercatore e note di Moro);
- la disambiguazione dei riferimenti generici alle entità a cui il documento si riferisce (es. termini come “Presidente del Consiglio”, “Papa”, etc.);
- l'inserimento di metadati bibliografici relativi al documento (es. tematica del documento, tipologia del documento, etc.);
- il download del documento marcato in più formati (es. HTML e TEI/XML).

L'interfaccia principale mostra le varie sezioni della pagina web come la barra strumenti, in cui è possibile avere:

- la ricerca di un documento;
- la possibilità di attivare la modalità editing;
- la gestione del login;
- l'upload di un file.

Sul lato sinistro, invece, si trova un pannello che gestisce le entità e un pannello che gestisce gli scarti e il cestino.

Nella sezione principale è possibile vedere l'articolo che si può facilmente scegliere tra quelli predefiniti che sono già presenti all'interno dell'ambiente. Nella sezione operazioni, invece, l'utente ha la possibilità di scaricare il documento in due possibili formati permettendo in questo modo di avere una standardizzazione dei formati:

- HTML
- TEI/XML

4.2 Introduzione a Voyant

Voyant Tools è un ambiente di analisi, lettura e visualizzazione del testo basato sul web.

Venne sviluppato da un team di studiosi di discipline umanistiche digitali guidati da Stéfán Sinclair e Geoffrey Rockwell, ideato a scopo di ricerca. *Voyant Tools* è progettato per una gamma molto ampia di applicazioni e utenti, dagli studenti ai ricercatori, dai giornalisti agli analisti di mercato, è open source, quindi, facilmente reperibile poichè, appunto, lo scopo è proprio che venga implementato all'interno di altri progetti per fornire delle funzionalità aggiuntive.

Dagli anni '90, Humanities Computing e Digital Humanities si sono fortemente sviluppate e strutturate in Canada . Negli anni 2000, Stéfán Sinclair ha sviluppato



Aggiungi Testi 🔍 🌙 ?

Inserisci uno o più URL separatamente o incolla un testo completo.

📄 Apri
📄 Carica
✓ Rivela

Voyant Tools è un ambiente web per la lettura e l'analisi di testi digitali
 Traduzione italiana a cura di Fabio Ciotti (AIUCD team: Alessandra Baldelli, Cristiana Bettella, Eleonora Durban, Federico Caria, Federico Meschini, Greta Franzini, Giorgio Guzzetta, Roberto Rosselli del Turco, Elena Spadini, Tiziana Mancinelli)

Figura 4.3: Home page Voyant

il software HyperPo come parte della sua ricerca di dottorato su Oulipo. A partire dal 2008 circa, la piattaforma è stata aperta con il nome di Voyeur, poi Voyant. In particolare, utilizza le tecnologie Flash e Java. Nel 2015 è apparso Voyant 2.0, che includeva una riscrittura in HTML 5, filtri di query migliorati e strumenti introdotti basati sul calcolo di prossimità e sequenze di elementi o diagrammi.

Voyant Tools si basa essenzialmente sul principio di apertura. Il suo codice è pubblicato in open source e sono ben accetti contributi di ogni tipo.

L'idea è proprio quella che il loro progetto venga utilizzato e implementato da studenti e ricercatori in modo da poter aiutare attraverso un arricchimento di funzionalità. La possibilità di inserire uno strumento o delle viste dinamiche in una pagina web va già verso una direzione innovativa.

Voyant Tools è senza dubbio il tipo di progetto che ci porta verso una cultura digitale e statistica più condivisa. Le caratteristiche di *Voyant* sono:

- studiare testi che si trovano sul web o testi che sono stati modificati e che si trovano sul proprio computer;

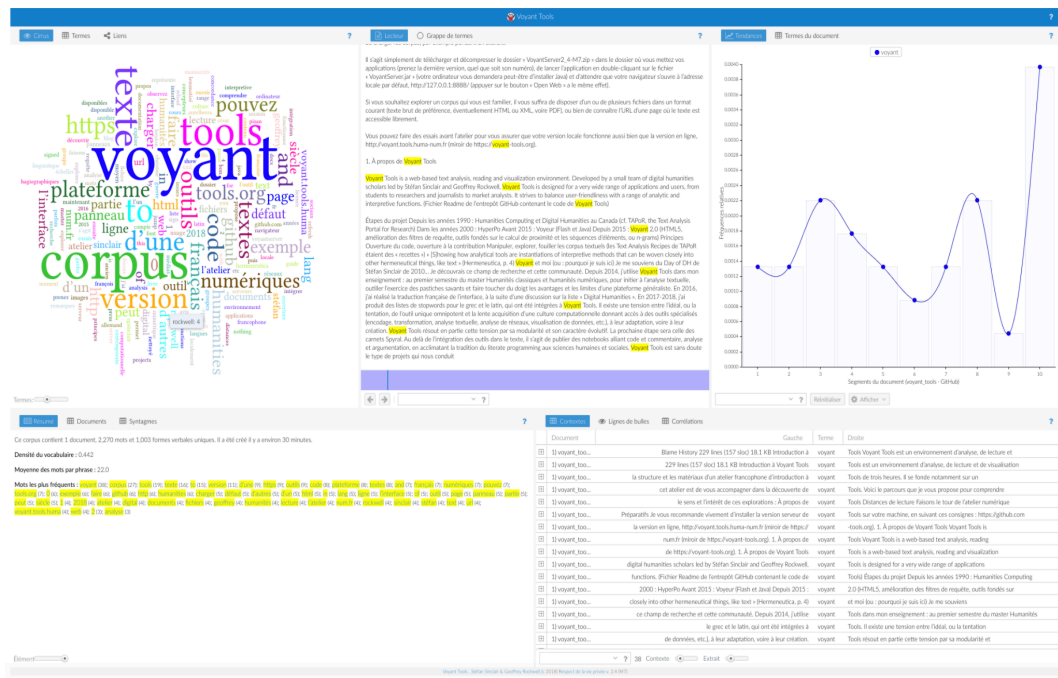


Figura 4.4: Voyant tool

- aggiungere funzionalità alle raccolte online, riviste, blog o siti web in modo che altri possano vedere i testi con strumenti analitici;
- permette di caricare sia un testo completo sia il link a URL esterno, ad esempio una rivista online contenente articoli;
- aggiungere prove interattive al proprio testo finalizzato alla pubblicazione online.

Nella figura 4.3 viene mostrato come appare l'interfaccia di Voyant dopo aver caricato un file e mostra la possibilità di scegliere il tool più appropriato.

Lo strumento fornisce molteplici funzionalità attraverso un'interfaccia intuitiva, supportata da una dettagliata documentazione. Offre la possibilità di appropiare testi (di qualsiasi tipologia e/o lingua) dando, attraverso la grande varietà di tool, una visione più globale del proprio lavoro, prendendo in esame diverse sfaccettature.

Un paio di esempio sono:

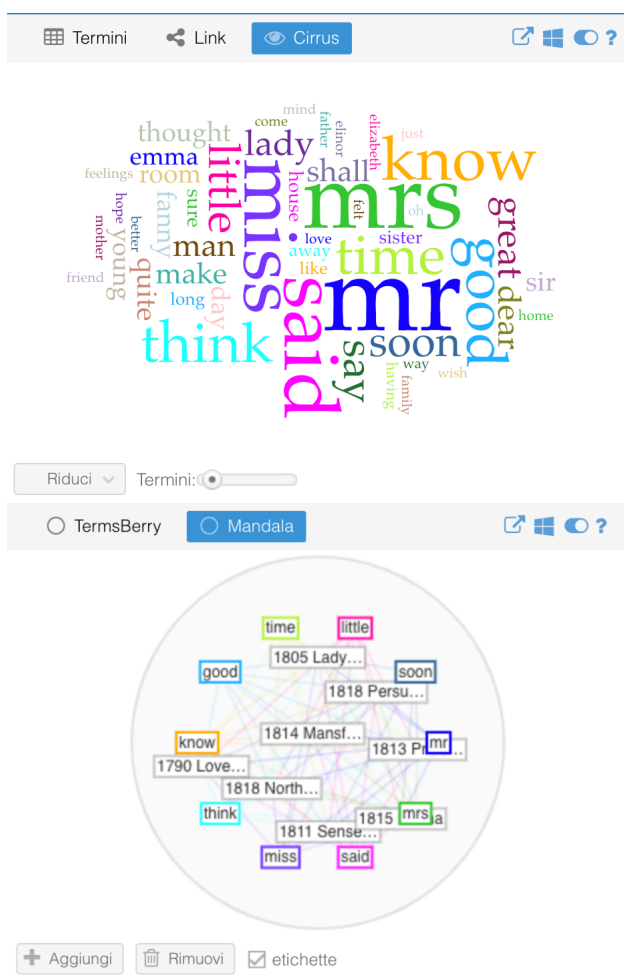


Figura 4.5: Esempi Voyant tool

- *Cirrus*: crea una word cloud, cioè una rappresentazione grafica in cui le parole più frequenti in un corpus o documento sono al centro e più grandi;
- *Mandala*: è un tool di visualizzazione che permette, a partire da un termine chiave o magnete (scelto dall'utente) di attrarre a sé tutti i documenti che contengono quel termine in base alla sua frequenza relativa.

4.3 Web Server Apache Tomcat

Tomcat è un Servlet Container ed un JSP Engine. Un motore quindi in grado di eseguire lato server applicazioni Web basate sulla tecnologia J2EE e costituite da componenti Servlet e da pagine JSP


Apache Tomcat (o semplicemente Tomcat) è un server web (nella forma di contenitore servlet) open source sviluppato dalla Apache Software Foundation. Implementa le specifiche JavaServer Pages (JSP) e servlet, fornendo, quindi, una piattaforma software per l'esecuzione di applicazioni web sviluppate in linguaggio Java. La sua distribuzione standard include anche le funzionalità di web server tradizionale, che corrispondono al prodotto Apache. Attualmente Tomcat è oggetto di un progetto indipendente distribuito sotto la Licenza Apache, ed è scritto interamente in Java; può, quindi, essere eseguito su qualsiasi architettura su cui sia installata una JVM. E' costituito da 3 componenti principali:

The screenshot shows the Apache Tomcat website. At the top left is the Tomcat logo (a yellow cat) and the text 'Apache Tomcat®'. To the right is the Apache Software Foundation logo. Below the logo is a search bar with 'Search...' and a 'GO' button. A banner for 'APACHECON @home' is visible. The main content area has a header 'Apache Tomcat' and a search bar. Below this is a paragraph of text: 'The Apache Tomcat® software is an open source implementation of the Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket technologies. The Java Servlet, JavaServer Pages, Java Expression Language and Java WebSocket specifications are developed under the [Java Community Process](#).' This is followed by another paragraph: 'The Apache Tomcat software is developed in an open and participatory environment and released under the [Apache License version 2](#). The Apache Tomcat project is intended to be a collaboration of the best-of-breed developers from around the world. We invite you to participate in this open development project. To learn more about getting involved, [click here](#).' A third paragraph states: 'Apache Tomcat software powers numerous large-scale, mission-critical web applications across a diverse range of industries and organizations. Some of these users and their stories are listed on the [PoweredBy](#) wiki page.' A fourth paragraph says: 'Apache Tomcat, Tomcat, Apache, the Apache feather, and the Apache Tomcat project logo are trademarks of the Apache Software Foundation.' Below this is a section header 'Tomcat 7.0.107 Released' with the date '2020-11-23'. The text below reads: 'The Apache Tomcat Project is proud to announce the release of version 7.0.107 of Apache Tomcat. This release contains a number of bug fixes and improvements compared to version 7.0.106.' A bullet point follows: '• Ensure that none of the methods on a ServletContext instance always fail when running under a SecurityManager. Pull request provided by Kyle Stiemann.' Below this is a paragraph: 'Full details of these changes, and all the other changes, are available in the [Tomcat 7 changelog](#).' A 'Note' section states: 'Note: End of life date for Apache Tomcat 7.0.x is announced. [Read more...](#)' and a 'Download' link is provided. At the bottom is another section header 'Tomcat 9.0.40 Released' with the date '2020-11-17'. The text below reads: 'The Apache Tomcat Project is proud to announce the release of version 9.0.40 of Apache Tomcat. The notable changes compared to 9.0.39'.


Figura 4.6: Home Tomcat

- Catalina Catalina è il contenitore di servlet Java di Tomcat. Catalina implementa le specifiche di Sun Microsystems per le servlets Java e le "JavaServer Pages (JSP, Pagine JavaServer)". In Tomcat un elemento del Reame rappresenta un database di usernames, passwords e ruoli (analoghi dei gruppi di UNIX) assegnati a quegli utenti. Differenti implementazioni del Reame permettono a Catalina di essere integrato in ambienti dove tali informazioni di autenticazione sono già state create e supportate, e poi gli permettono di utilizzare tali informazioni per implementare una cosiddetta "Container Managed Security" come descritto nelle Specifiche delle Servlet.

Home Documentation Configuration Examples Wiki Mailing Lists Find Help

Apache Tomcat/9.0.38 

If you're seeing this, you've successfully installed Tomcat. Congratulations!

 **Recommended Reading:**
[Security Considerations How-To](#)
[Manager Application How-To](#)
[Clustering/Session Replication How-To](#)

Server Status
 Manager App
 Host Manager

Developer Quick Start

[Tomcat Setup](#) [Realms & AAA](#) [Examples](#) [Servlet Specifications](#)
[First Web Application](#) [JDBC DataSources](#) [Tomcat Versions](#)

Managing Tomcat
 For security, access to the `manager webapp` is restricted. Users are defined in:
`$CATALINA_HOME/conf/tomcat-users.xml`
 In Tomcat 9.0 access to the manager application is split between different users.
[Read more...](#)

[Release Notes](#)
[Changelog](#)
[Migration Guide](#)
[Security Notices](#)

Documentation
[Tomcat 9.0 Documentation](#)
[Tomcat 9.0 Configuration](#)
[Tomcat Wiki](#)
 Find additional important configuration information in:
`$CATALINA_HOME/RUNNING.txt`
 Developers may be interested in:
[Tomcat 9.0 Bug Database](#)
[Tomcat 9.0 JavaDocs](#)
[Tomcat 9.0 Git Repository at GitHub](#)

Getting Help
FAQ and Mailing Lists
 The following mailing lists are available:

[tomcat-announce](#)
 Important announcements, releases, security vulnerability notifications. (Low volume).

[tomcat-users](#)
 User support and discussion

[taglibs-user](#)
 User support and discussion for [Apache Taglibs](#)

[tomcat-dev](#)
 Development mailing list, including commit messages

Figura 4.7: Home Tomcat

- Coyote Coyote è il componente "connettore HTTP" di Tomcat. Supporta il protocollo HTTP 1.1 per il web server o per il contenitore di applicazioni. Coyote ascolta le connessioni in entrata su una specifica porta TCP sul server

e inoltra la richiesta al Tomcat Engine per processare la richiesta e mandare indietro una risposta al client richiedente.

- Jasper Jasper è il motore JSP di Tomcat. Tomcat 5.x utilizza in realtà Jasper 2, che è un'implementazione delle specifiche 2.0 delle Pagine JavaServer (JSP) di Sun Microsystems. Jasper analizza i file JSP per compilarli in codice Java come servlets (che verranno poi gestite da Catalina). Al momento di essere lanciato, Jasper cerca eventuali cambiamenti avvenuti ai file JSP e, se necessario, li ricompila.

Dopo aver eseguito l'installazione di Tomcat si dovrà mandare in esecuzione il terminale nella cartella bin e si avrà certezza di averlo installato correttamente quando apparirà la seguente schermata al link <http://localhost:8080>

4.4 Lato server

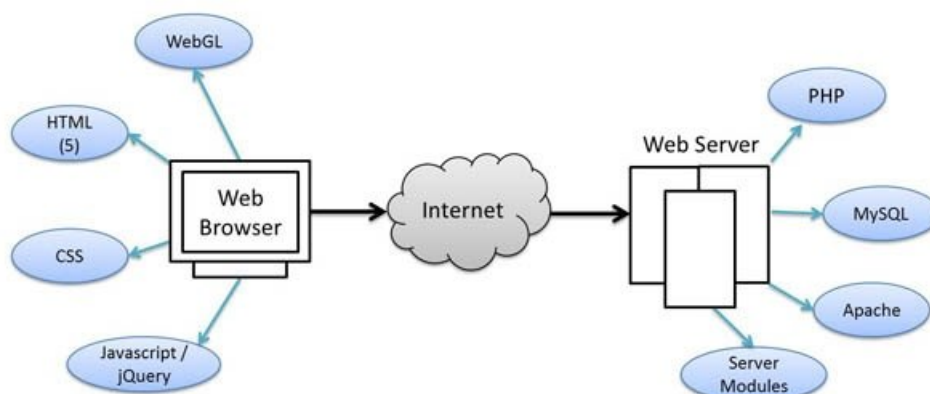


Figura 4.8: Lato server e lato client

In questa sezione sono descritte in dettaglio le tecnologie utilizzate per la realizzazione delle modifiche che ho apportato all'applicazione. Lo studio di esse ha occupato una parte rilevante del mio lavoro in quanto sono tecnologie di nuova generazione. Viene proposta di seguito l'analisi dei linguaggi di programmazione e di tutti gli strumenti che con essi vanno ad integrarsi per la realizzazione del lavoro descritto.

Una tecnologia lato server raggruppa quei linguaggi che vengono elaborati dal server, il quale mette a disposizione un insieme di servizi utili per il reperimento di informazioni o funzionalità non disponibili nella macchina dell'utente. E' usata nello sviluppo di siti con elementi dinamici e nelle applicazioni web, si basa sull'uso di script che vengono eseguiti dal web server attraverso linguaggi di scripting. Ad ogni richiesta effettuata dal lato client verrà poi spedito il risultato prodotto dal server.

Node.js

Node è un framework che viene utilizzato per scrivere applicazioni in Javascript lato server, costruita sul motore Javascript V8 di Google Chrome. Si tratta di un framework che permettere di porre una soluzione lato server basata su un modello di I/O asincrono che opera sugli eventi, quindi, richiede di ricevere notifiche ogni qual volta si verifichi un evento e rimane in sleep fino a quando non arriva la notifica.

```
1  const router = require('express').Router();
2  const fs = require('fs');
3  const fgc = require('file-get-contents');
4  const mkdir = require('make-dir');
5  const mammoth = require('mammoth');
6  var HTMLParser = require('node-html-parser');
7  var request = require('sync-request');
8
9  const dir = 'public/files';
10 const src = /src=([\|\'])(?!http)(?!#footnote)/g;
11 const href = /href=([\|\'])(?!http)(?!#footnote)/g;
12 const out = [];
13
14 let listing = new Promise((resolve, reject) => {
15   fs.readdir(dir, (err, files) => {
16     if (err) return console.log('ERROR', err);
17     else {
18       files.forEach(file => {
19         if (file.substring(0, 1) !== '.') {
20           let obj = {
21             url: file,
22             label: file
23           };
24           out.push(obj);
25         }
26       });
27     }
28     resolve('success');
```

Figura 4.9: Esempio di codice in Node.js

4.5 Lato client

La tecnologia lato client raggruppa quei linguaggi in cui l'esecuzione e l'interpretazione delle istruzioni avvengono in locale, sul computer che effettua la richiesta al server. A differenza della variante lato server, gli script non vengono però programmati dal server, bensì elaborati ed eseguiti dal client richiedente. Ne deriva che il risultato visibile sul terminale del client dipende fortemente dalla presenza di un software adatto, che dovrà interpretare le istruzioni che gli sono state inviate. Node.js è un progetto di sviluppo open source, caratterizzato dalla sua architettura event-driven, quindi, mentre sta elaborando una procedura ne può eseguire un'altra anche se la prima non è ancora stata portata a termine.

HTML

HTML *Hypertext Markup Language*, linguaggi di "marcatura" di *Iper testi* è il linguaggio per creare pagine ipertestuali (pagine web).

HTML è un linguaggio che serve a creare e dare uno stile alle pagine internet, che data la complessità di elementi che contiene al suo interno (suoni, immagini, filmati ecc.) , viene definita ipertesto. I documenti HTML sono spesso chiamati "Pagine Web", in pratica un file in formato HTML è un file che, oltre a contenere il testo che verrà visualizzato dal browser, contiene anche dei comandi (racchiusi sempre tra i simboli "<" e ">" chiamati "tag") che associano al testo un particolare attributo, i tag vengono riconosciuti ed interpretati dai browser web. Questo è un esempio di codice html. Gli elementi del linguaggio vengono detti mark up tag o semplicemente tag: essi di solito sono utilizzati a coppie, e possono contenere uno o più attributi. Una pagina in codice HTML può essere redatta con un semplice editor di testi e salvata con estensione .html o .htm. Quando il browser (Mozilla, Firefox, Google Chrome ecc.) carica un file HTML, legge e interpreta i tag in esso contenuti e presenta il risultato di tale elaborazione sullo schermo.

```
1 <html>
2 <html>
3 <head>
4 <title>
5 TITOLO
6 </title>
7 </head>
8 <body>
9 <h1>TITOLO PRINCIPALE</h1>
10 <p>IL <strong>PRIMO</strong> PARAGRAFO</p>
11 <h3>SOTTOTITOLO</h3>
12 <p>ALTRI PUNTI</p>
13 </body>
14 </html>
15
```

Figura 4.10: Esempio di documento html

Javascript

Javascript è un linguaggio di programmazione, conosciuto come linguaggio di scripting client-side per pagine web . JavaScript viene eseguito direttamente lato client della pagina web e può essere utilizzato per la formazione di un design e stabilire il comportamento delle pagine web quando viene scatenato un particolare evento da parte dell'utente. JavaScript è semplice da apprendere e nello stesso tempo rappresenta un linguaggio che permette un controllo quasi totale sulla pagina web. La sua sintassi deriva dai linguaggi Java e C, quindi, molte strutture da questi linguaggi ricorrono anche in Javascript. Per poter scrivere codice Javascript è sufficiente un editor di testo da salvare con estensione ".js" per poter essere richiamato nelle pagine HTML, oppure si può inserire codice Javascript direttamente all'interno dei file HTML utilizzando gli opportuni tag:

```
1 <script>
2
3 // codice javascript
4
5 </script>
6
7
```

Figura 4.11: Esempio codice javascript

Capitolo 5

Voyeditor

In questo capitolo vengono introdotte nel dettaglio le strutture dati e le funzioni utilizzate per la realizzazione dell'applicazione. Per ogni componente vengono, inoltre, descritte e giustificate nel dettaglio le scelte implementative effettuate e quanto esse si siano rivelate adatte al raggiungimento degli obiettivi preposti.

5.1 Struttura del progetto KwickKwocKwac

Il progetto si presenta organizzato in diverse cartelle, alcune delle quali non saranno modificate perchè non inerenti ai cambiamenti richiesti. Nella figura 5.1 viene mostrato come si presenta graficamente il progetto, verranno però modificati i seguenti file:

- index.html
- script.js
- backend.js

Nei paragrafi successivi verranno mostrati e spiegati nel dettaglio i frammenti di codice che sono stati modificati e si potranno vedere infine le modifiche effettuate e l'aspetto finale dell'applicazione web.

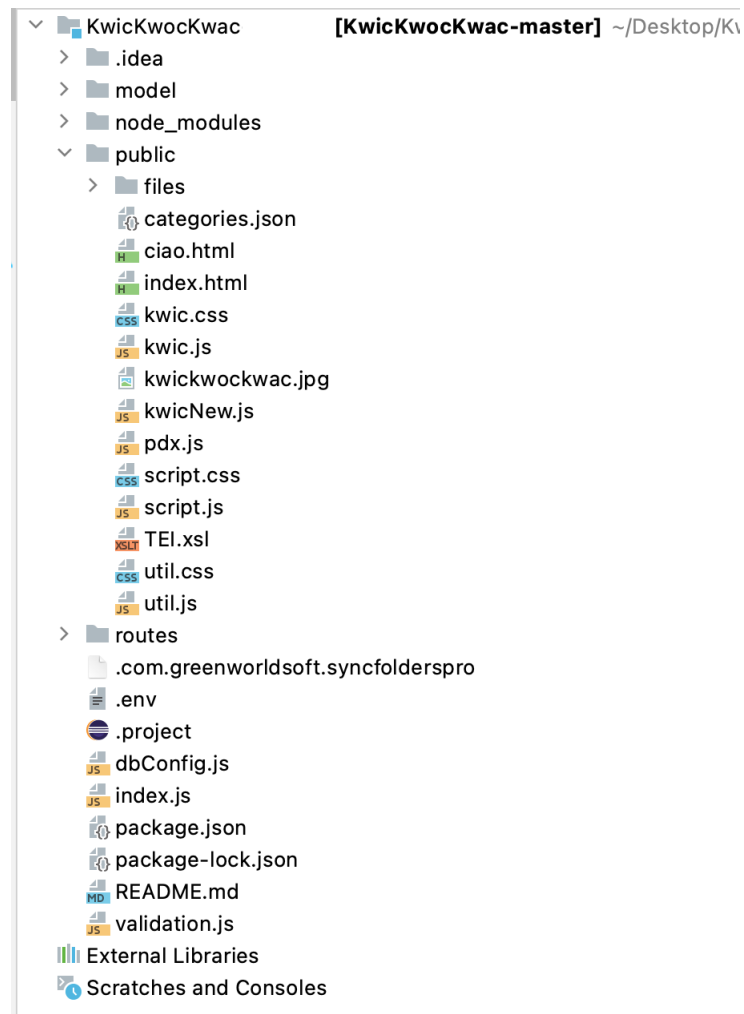


Figura 5.1: Struttura progetto KwicKwockWac

5.2 Possibili modi per integrare Voyant in un ambiente web

Partendo dalla home page di Voyant ci si trova davanti a due scelte:

- caricare un file già esistente;
- effettuare l'upload di un file presente sulla nostra macchina.

Successivamente apparirà una schermata in cui sarà possibile vedere tutti i tools che vengono messi a disposizione.

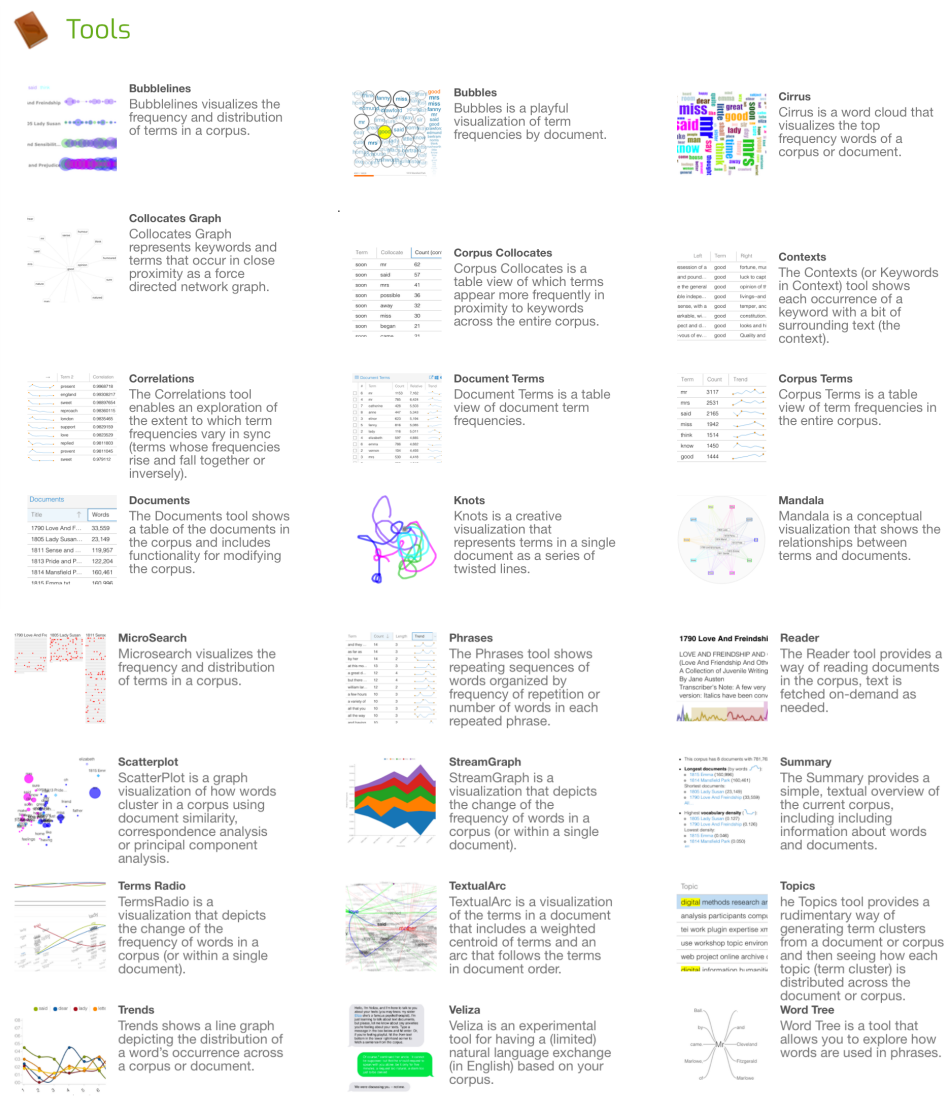


Figura 5.2: Tools Voyant

Tra questi verrà fatta una cernita mantenendo quelli che sono ritenuti più efficienti per il nostro scopo e che abbiano più efficacia all'interno dell'ambiente web. Come si vede nella figura 5.3 ci si troverà in una schermata in cui vengono visualizzati tutti i tools presenti che analizzano diversi aspetti del testo, che sia la frequenza di una parola o la sua localizzazione all'interno del brano. Si possono effettuare diverse operazioni, l'utente può a suo piacimento scegliere quale grafico visualizzare semplicemente accedendo alla barra di menu e selezionando il

The image displays two screenshots of the Voyant Tools web application. The top screenshot shows the main interface with a word cloud on the left, a central text area displaying a document snippet, and a line graph on the right. The bottom screenshot shows the 'Esporta' (Export) dialog box, which offers three options for exporting the current view: as a URL, as an HTML fragment, or as a bibliographic reference. The dialog also includes an 'Esporta la Vista' (Export View) button and an 'Elimina' (Delete) button.

Figura 5.3: Spiegazione Tools Voyant

tool che preferisce.

E' presente un ulteriore tasto che permette di esportare il tool attraverso un frammento di codice html in modo da permettere una facile implementazione, o, semplicemente è possibile esportare un file *png* che permetta di scaricare sul proprio dispositivo l'immagine del tool che si preferisce.

Cliccando sull'icona evidenziata si accederà ad una sezione attraverso la quale si potrà scegliere di esportare un tag *<iframe>* che permetterà la visione del grafico all'interno della nostra pagina web.

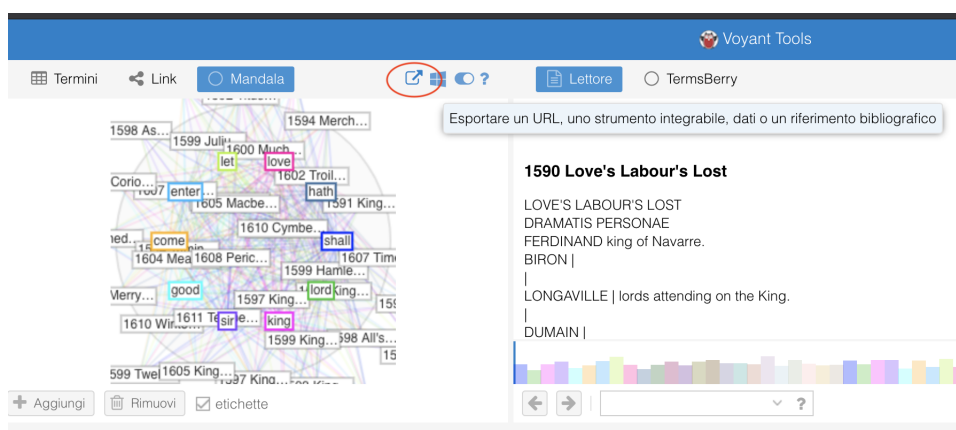


Figura 5.4: come esportare un tool

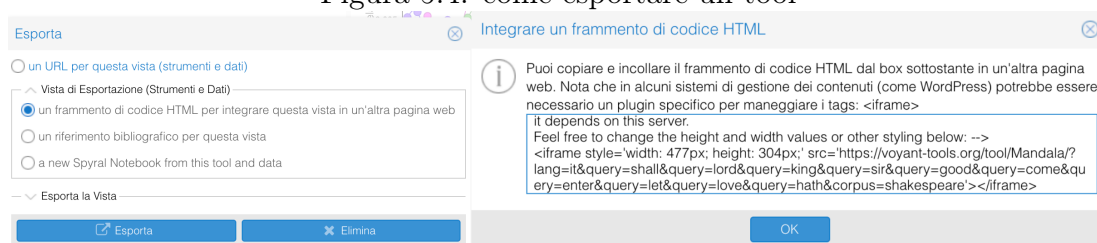


Figura 5.5: come esportare un tool

5.3 Sviluppo delle funzionalità richieste

La prima ipotesi è stata quella di provare semplicemente ad inserire il tag *iframe* all'interno della pagina web. Il problema riscontrato è che in questo modo si sarebbe dovuto andare a gestire manualmente la creazione dei grafici con i nostri file. Per poter inserire all'interno del tag *iframe* il link del file analizzato da Voyant ho avuto bisogno di un link data-src, l'unico modo che avevo per poter esportare il link necessario era quello di inserire manualmente il file che volevo analizzare, inserirlo nel sito Voyant ed infine esportare il link data-src ed inserirlo nella pagina html.

Come si evince questo metodo avrebbe permesso di esportare facilmente il grafico, ma non in modo efficiente in quanto ci sarebbe dovuto essere un lavoro manuale non produttivo ai fini dell'ambiente web.

Questa soluzione sarebbe stata efficiente solo per un uso limitato dei file presenti

```

<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>prova</title>

  <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>
  <script type="text/javascript" src="jQuery.js"></script>
  <script type="text/javascript"></script>

  <<
  <script>
    $(document).ready(function() {
      h1 = document.getElementsByTagName("h1")[0].getAttribute("data-src");
      //console.log(encodeURIComponent(h1));

      var ifrm = document.getElementById('ifrm');
      ifrm.src = 'http://voyant-tools.org/tool/Trends/?input='+encodeURIComponent(h1);
    });
  </script>
</head>

<body>
<h1 data-src="https://abcnews.go.com/US/death-row-inmates-fight-life-shines-light-jailhouse/story?id=68357003">Death row inmate's fight
  for his life shines light on use of jailhouse informants </h1>

  Click on this to see a dialogue box.

  prova
  <p>
  <iframe id="ifrm" style='...' src=''></iframe>
  </p>
</body>
</html>

```

Figura 5.6: prima ipotesi di implementazione

nella lista predefinita dell'applicazione web. Pensando, invece, ad un ipotetico upload di file si sarebbero dovuti manualmente eseguire i passaggi elencati prima per creare il link data-src così da esportare i relativi grafici. Il problema generale che sorgeva era che i file presenti nella lista, essendo dei file statici, non sono presenti sul web. Questo non permetteva di poter creare un meccanismo generico che fosse in grado di generare i tools automaticamente nel momento in cui avveniva il cambio di file.

5.3.1 Come rendere i file dinamici

All'interno dell'applicazione web, invece, sono stati caricati dei file predefiniti in una cartella, mettendo quindi a disposizione una serie di file index.html che però risultano essere dei file statici, non hanno un percorso sul Web e questo potrebbe risultare essere limitante per il lavoro e per il raggiungimento dell'obiettivo.

Si doveva trovare un modo per pubblicare i file sul web in modo che si possano richiamare facilmente.

Ho pensato ad una piattaforma su cui poter pubblicare i file che fosse gratuita o una

piattaforma che avesse disponibile una versione per studenti. Ho provato a caricare i file in una directory su Github e ad eseguire una GET . Il problema riscontrato

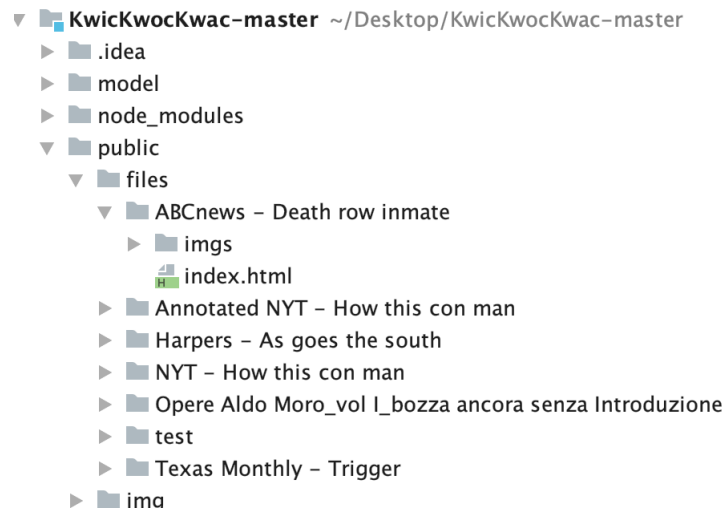


Figura 5.7: dove si trovano i file nel menu

con questo approccio è che viene preso in input il file html e nel momento in cui si generava il grafico tool vengono analizzate non solo le parole che componevano il testo, ma anche le parole che formano i tag, quindi, il risultato atteso non era soddisfacente e bisognava cercare un altro modo per arrivare ad un risultato ottimale.

Per questo motivo ho ritenuto necessario richiedere l'aiuto di Tomcat, di cui abbiamo parlato precedentemente, un server web che ha permesso di aiutare a gestire la parte statica del progetto.

Per quanto riguarda la configurazione di Tomcat è necessario apportare una modifica nel file web.xml situato nella cartella *conf* per far sì che si possa recuperare la lista dei documenti direttamente dal browser. Una volta modificata la configurazione si dovranno inserire all'interno della cartella *webapp* il file .war di Voyant e la cartella files contenente i nuovi file modificati.

```
<servlet>
  <servlet-name>default</servlet-name>
  <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>0</param-value>
  </init-param>
  <init-param>
    <param-name>listings</param-name>
    <param-value>>false</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>

<servlet>
  <servlet-name>default</servlet-name>
  <servlet-class>org.apache.catalina.servlets.DefaultServlet</servlet-class>
  <init-param>
    <param-name>debug</param-name>
    <param-value>0</param-value>
  </init-param>
  <init-param>
    <param-name>listings</param-name>
    <param-value>>true</param-value>
  </init-param>
  <load-on-startup>1</load-on-startup>
</servlet>
```

Figura 5.8: configurazione da cambiare in Tomcat

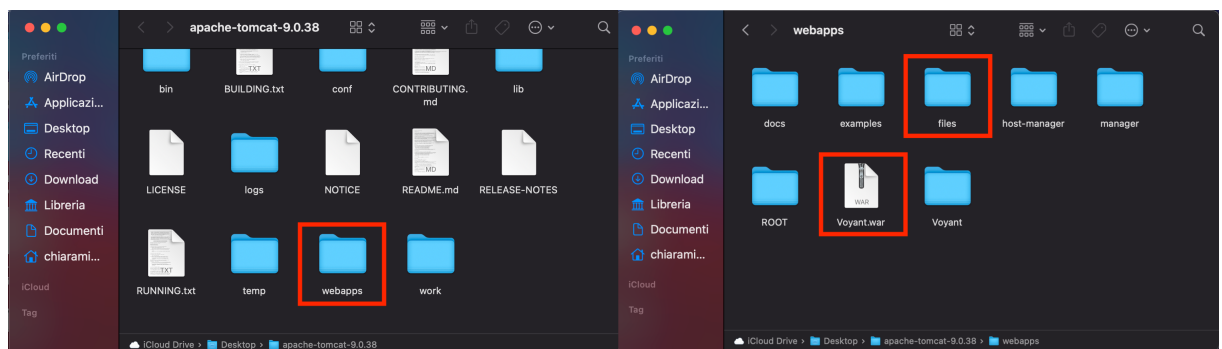


Figura 5.9: dove posizionare i file

5.4 Wireframes

Un wireframe viene realizzato per poter immaginare al meglio la posizione degli elementi che devono essere inseriti all'interno della pagina web. Lo scopo è quello di studiare la miglior soluzione possibile per disporre gli elementi. Per

creare i wireframe è stato utilizzato un programma di nome Balsamiq.¹ Sono

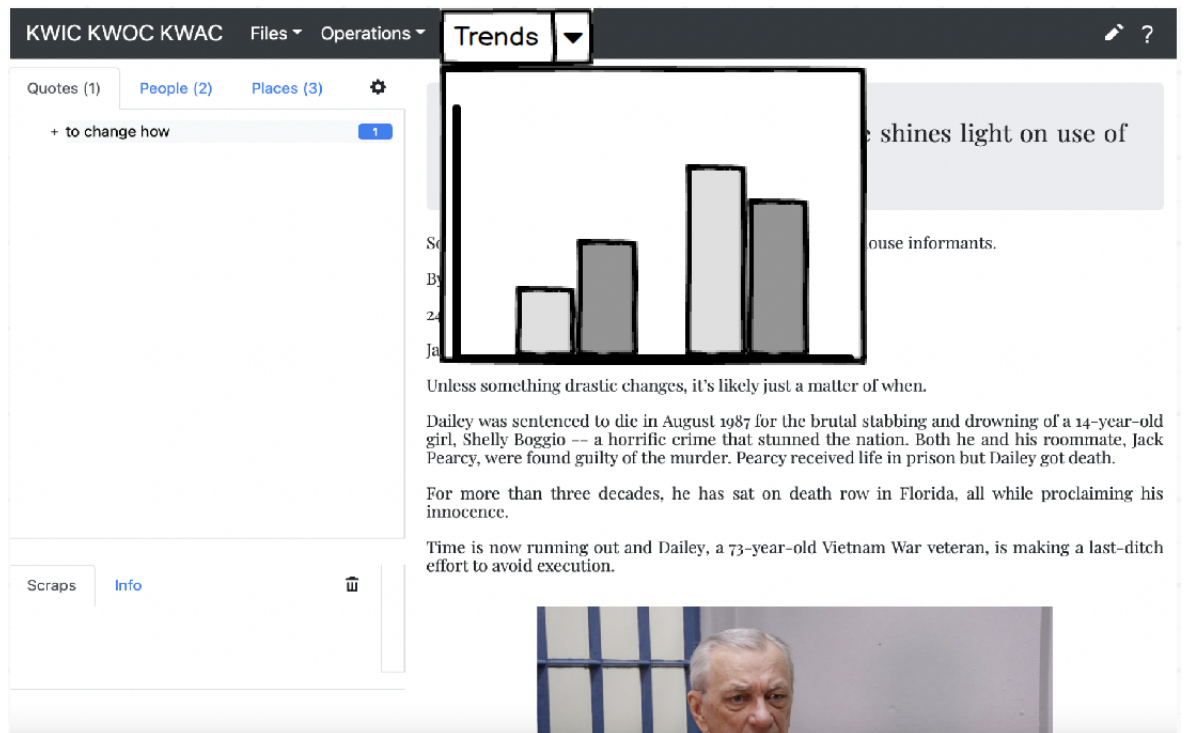


Figura 5.10: primo wireframe

state fatte due ipotesi per l'inserimento dell'elemento, come viene rappresentato figura 5.10: la prima ipotesi è stata quella di inserire una ulteriore combobox nel menu che permettesse di visualizzare l'elemento selezionandolo da un menu a tendina; la seconda ipotesi, invece, rappresentata nella 5.11 è stata ritenuta migliore perchè più visibile. Nel momento in cui l'utente accede, a prima vista sulla sinistra nota subito l'elemento e quindi è portato ad utilizzarlo, anche se non aveva programmato di farlo. Un altro vantaggio è il fatto che mentre si lavora sul testo contenuto nel pannello principale si vede in contemporanea l'elemento che fornisce delle informazioni utili aggiuntive.

¹Balsamiq <https://balsamiq.com/wireframes/>

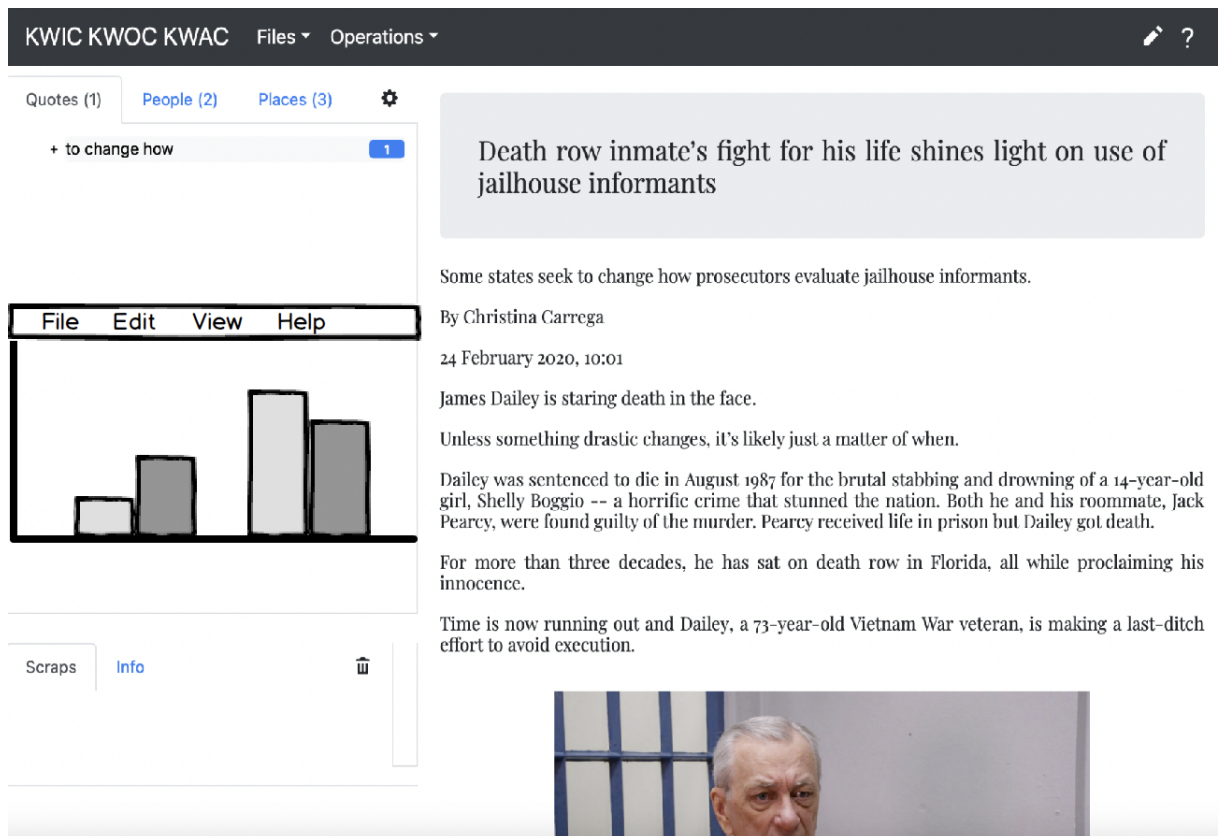


Figura 5.11: secondo wireframe

5.5 *index.html*

All'interno del file html (5.12) è stata per prima cosa creata la sezione della home in cui si va a inizializzare un *home-cointainer* che vada a definire lo spazio dedicato all'introduzione, in modo che l'utente abbia sempre a disposizione una pagina principale. Nella figura 5.13, invece, è stato il *voyant-container*, ponendolo nella sezione a sinistra, creando un altro pannello tra quello che mostra le entità e quello che mostra gli scarti.

Si è inoltre creato un modale per permettere di visualizzare una finestra a scomparsa più ampia e che mostrasse i possibili tools e desse una panoramica generale dell'analisi specifica. Nel paragrafo successivo si vedranno nel dettaglio le modifiche che sono state apportate dal lato grafico.

```

85     </div>
86   </nav>
87   <div id="home-container">
88     <center>
89       <h1>Voyeditor</h1>
90     </center>
91     <div style="margin:80px;">
92
93     <p>
94
95     <b>Voyeditor</b> è un ambiente web che presenta una soluzione efficiente per fornire delle funzionalità aggiuntive all'applicazione web già esistente.
96     L'utente, prettamente uno studente o un filologo, può lavorare su di un testo e avere a disposizione tutte le funzionalità che offre Voyant.
97
98     Vengono implementate diverse funzionalità per aiutare l'attività dei filologi e
99     la ricerca fornendo strumenti efficienti per la modifica e l'analisi dei testi.
100
101     <li>scegli il testo da analizzare selezionando su <b>Files</b></li>
102     <li>seleziona il tool che preferisci per aiutarti nell'analisi</li>
103   </ul>
104 </p>
105 <center>
106   
108 </center>

```

Figura 5.12: codice per la creazione della home

```

409
410 <div class="modal fade" id="voyant-modal" tabindex="-1" role="dialog" aria-labelledby="voyant-modal-label"
411   aria-hidden="true">
412   <div class="modal-dialog modal-xl" role="document">
413     <div class="modal-content">
414       <div class="modal-header">
415         <h5 class="modal-title" id="voyant-modal-label">Voyeditor Analysis</h5>
416         <button type="button" class="close" data-dismiss="modal" aria-label="Close">
417           <span aria-hidden="true">&times;</span>
418         </button>
419       </div>
420       <div class="modal-body">
421         <div id="voyant-frame-container"></div>
422       </div>
423     </div>
424   </div>

```

Figura 5.13: voyant container

5.6 script.js

Di seguito vengono riportati i frammenti di codice che sono stati modificati per poter implementare l'elemento Voyant da aggiungere all'interno dell'ambiente web *Voyeditor*.

E' stato aggiunto un container a scomparsa, opportunamente strutturato ed inserito nell' index esistente, in modo tale da mettere a disposizione dell'utente uno strumento molto importante quale Voyant, senza però togliere spazio o visibilità al documento.

Sono state effettuate delle modifiche per quanto riguarda la creazione della ho-

mepage(5.14,5.15) settando la scritta *VOYEDITOR* in modo che diventasse un bottone così da permettere all'utente di poter tornare in ogni momento alla pagina principale.

Nella figura 5.17 sono state modificate le dimensioni del layout per adattare al meglio tutti i pannelli presenti.

```
37
38 var expandableSelector = '.treeExpand' // selector for expandable items in the tree in the left pane
39 var draggableSelector = '.draggable' // selector for elements that can be dragged in the left pane
40 var droppableSelector = '.dropPoint' // selector for elements that can receive draggable elements in the left pane
41
42 var homeOpened = false;
43
44 $(document).ready(main);
45
```

Figura 5.14: home

```
87 fetch('/api/list').then((res) => res.json()).then((elements) => docList(elements)).catch( () => alert(''))
88 fetch('/categories.json').then((res) => res.json()).then((json) => categoriesList(json)).catch( () => alert(''))
89
90 //bottone di home
91 $(".navbar-brand").click(function() {
92     homeOpened = true;
93     $('#main').css('visibility', 'hidden');
94     $('#home-container').css('visibility', 'visible');
95     $('#home-container').css('height', 'unset');
96 });
97
98 // setup event callbacks
99 basicCallbacks()
100 editCallbacks(editMode)
101 editSetup(editMode)
102 }
```

Figura 5.15: bottone home

```

465     currentFilename = file;
466     editMode = false;
467     $('#file').html(content)
468     $('#file').animate({ scrollTop: 0 }, 400);
469     $('#commandList').removeClass('d-none');
470     let voyantUrl = 'http://chiara.minervino.tw.cs.unibo.it/Voyant';
471     let voyantCirrusUrl = '/tool/Trends';
472     let voyantParams = '?categories=&input=' + currentFilename.split('index.html')[0] + 'corpus.txt';
473     //appendo voyantPane dopo topPane
474     $('#voyantPane').remove();
475     $('#topPane').after("<div class='row p-1 mb-4' id='voyantPane'><div class='col-12 p-0 mb-2'><span class='ml-3'></span><a data-toggle='modal'
476     |> voyantUrl + voyantCirrusUrl + voyantParams + \" style='width: 100%; height: 300px;'></iframe></div></div>");
477     // appendo nel modale il frame di analisi
478     $('#voyant-frame').remove();
479     $('#voyant-frame-container').append("<iframe src='\" + voyantUrl + voyantParams + \"\" style='width: 100%; height: 75vh;'></iframe>");
480     setupKWIC(documentLocation, false);
481

```

Figura 5.16: voyant

```

175 //-----qui vengono cambiate le dimensioni del layout-----
176     function layoutSetup() {
177         setLayout('width', 6);
178         setLayout('height', 70);
179         setTimeout(() => {
180             $('#pref-' + currentStyle).addClass('active')
181             $('#pref-' + currentSort).addClass('active')
182         }, 200)
183     }
184

```

Figura 5.17: layout

5.7 backend.js

All'interno del file *backend* in primo luogo sono state aggiunte due librerie, *HTMLparse*, una comune libreria di parsing e poi è stata aggiunta una request ausiliaria per permettere la visione dei file presi da Tomcat. Ho inserito un ciclo che prende in input i file da Tomcat, ma bisogna interrompere l'avanzamento del ciclo fino a quando i dati non sono pronti, altrimenti il ciclo stesso avanzerebbe al file successivo.

E' necessario fare una chiamata get per recuperare i file, per ogni tag è stato preso in analisi il link a cui tale tag punta; se il link non contiene *DS_store* viene aggiunto alla lista *filelist* la chiamata al singolo file, ogni elemento della lista avrà una label ovvero il titolo del documento e l'url che punta all'*index.html* del file.

```
1  const router = require('express').Router();
2  const fs = require('fs');
3  const fgc = require('file-get-contents');
4  const mkdir = require('make-dir');
5  const mammoth = require('mammoth');
6  var HTMLParser = require('node-html-parser');
7  var request = require('sync-request');
8
```

Figura 5.18: modifica backend

```
33
34  try {
35    var fileList = [];
36    var filesRequest = request('GET', 'http://chiara.minervino.tw.cs.unibo.it/files/');
37    var parsed = HTMLParser.parse(filesRequest.getBody());
38    aTags = parsed.querySelectorAll('a');
39    aTags.forEach(tag => {
40      let href = tag.getAttribute('href');
41      console.log(href);
42      if(!href.includes('.DS_Store')){
43        let titleUrl = 'http://chiara.minervino.tw.cs.unibo.it' + href + 'title.txt';
44        var titleRequest = request('GET', titleUrl);
45        fileList.push({
46          label: titleRequest.getBody().toString(),
47          url: 'http://chiara.minervino.tw.cs.unibo.it' + href + 'index.html'
48        });
49      }
50    });
51    res.send(fileList);
52  } catch (err) {
53    res.status(400).send(err);
54  }
55
56
57 });
58
59 router.get('/load', async (req, res) => {
60   try {
61     res.set('Content-Type: text/html');
62     res.send(request('GET', req.query.file).getBody());
```

Figura 5.19: modifica backend

5.8 Home page

La home page per un utente che accede per la prima volta si presentava in questo modo. Si evince subito il fatto che la pagina sia un po' scarna e che manchi

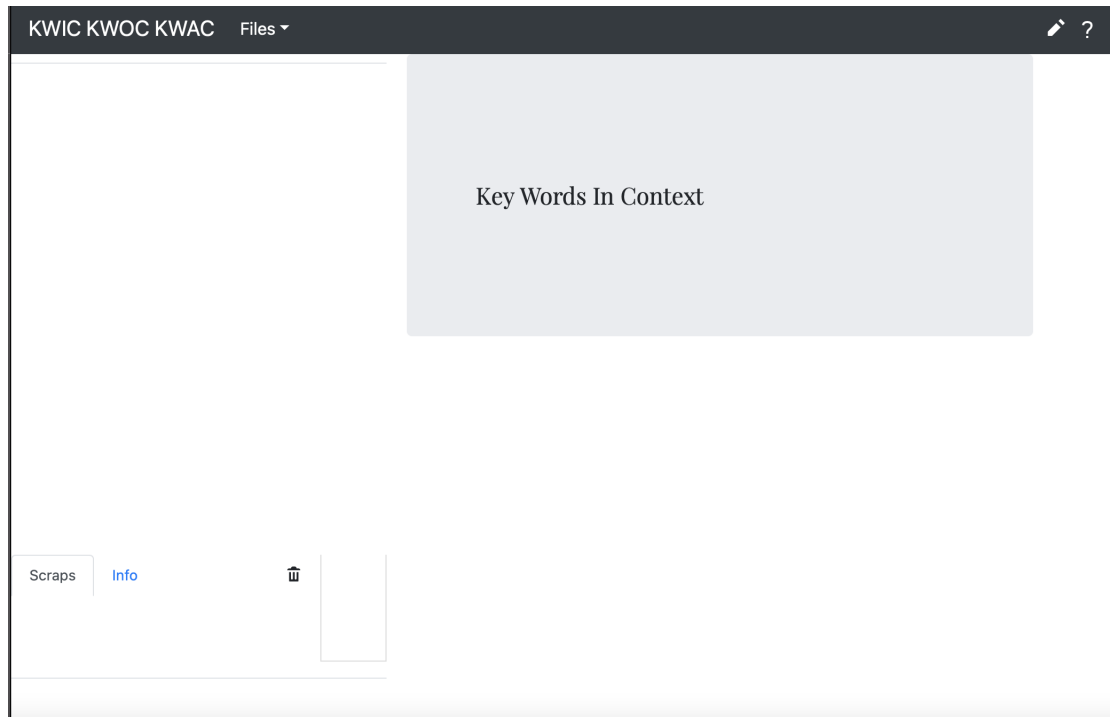


Figura 5.20: Home Page iniziale Kwic Kwoc Kwac

effettivamente una home page che permetta all'utente di poter avere una vera e propria pagina di riferimento.

Per questo motivo ho deciso di inserire una breve documentazione introduttiva e la possibilità di poter cliccare sul simbolo indicato nella figura ?? per poter in ogni momento della navigazione tornare alla home. Una volta modificato il codice index.html, la home ha presentato un nuovo aspetto, che permetterà una navigazione più fluida all'interno dell'ambiente web in modo da risultare anche essere d'aiuto all'utente.



Figura 5.21: tasto Home

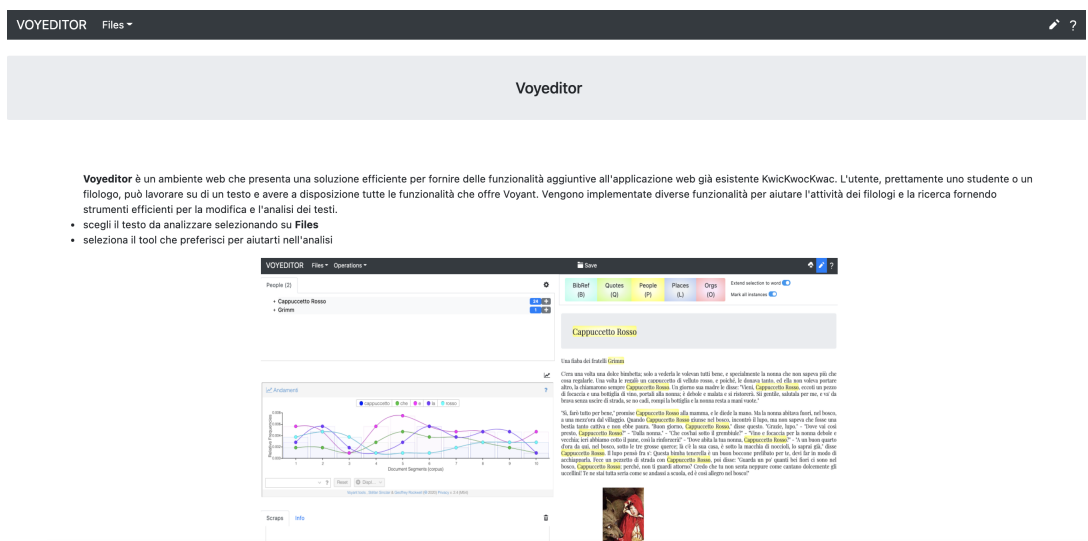


Figura 5.22: Home modificata

5.9 Applicazione web finale

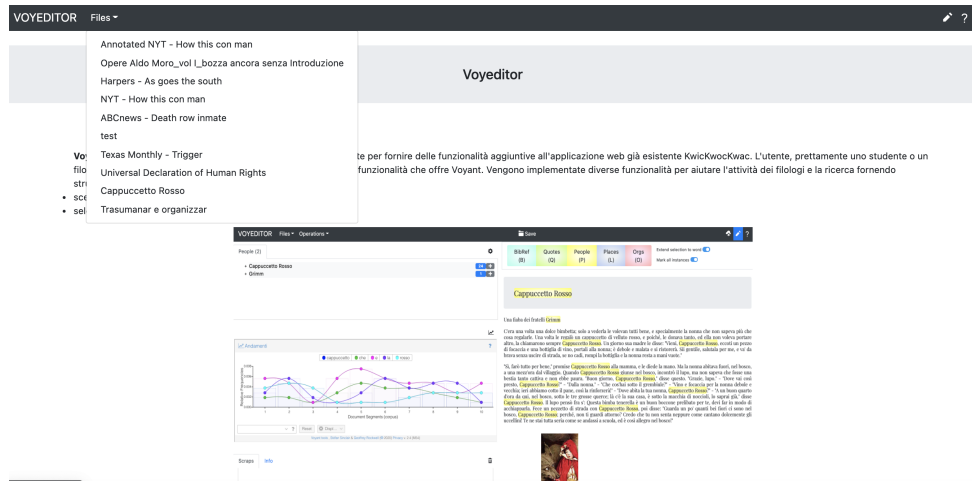


Figura 5.23: Home modificata

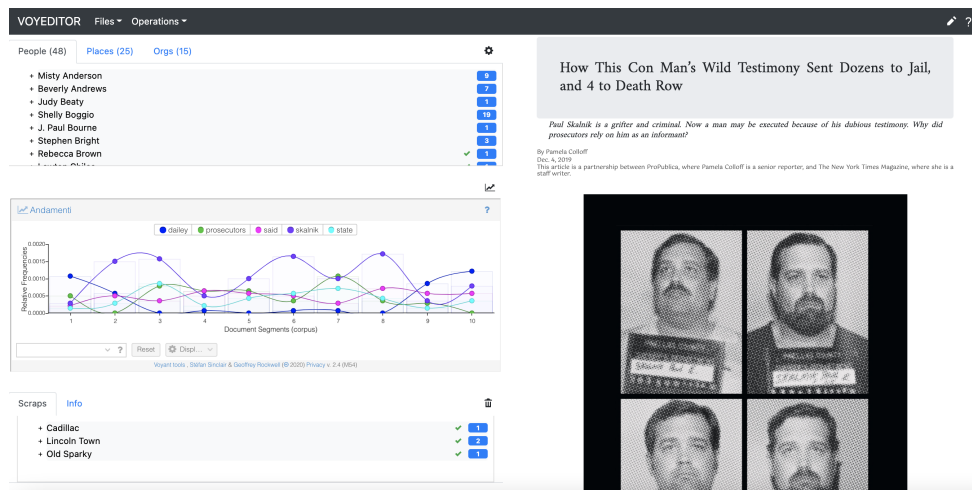


Figura 5.24: Home modificata

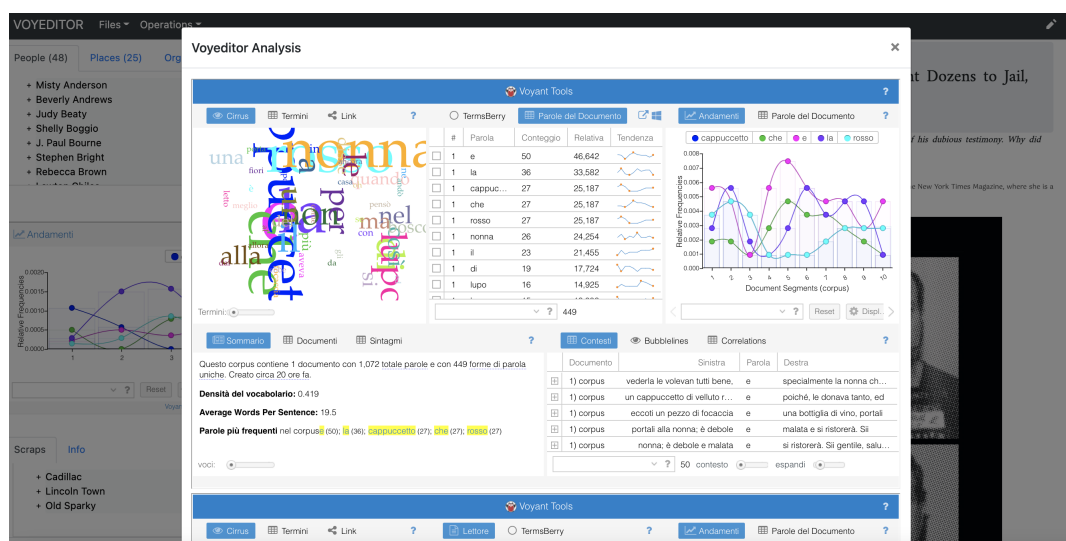


Figura 5.25: Home modificata

5.10 Modifiche apportate su Voyant ai fini dell'integrazione

Si è deciso di non riportare esattamente l'implementazione di Voyant, ma di modificarla in modo da adattarla alle esigenze dei filologi e in modo da garantire dei tools perfettamente efficienti nei loro minimi particolari.

Per questo motivo ho modificato il progetto di Voyant, disponibile su github [8], in modo da avere un menu di tools ad hoc per l'ambiente web. Il menu si può modificare all'interno del file *voyant.js* eliminando dalla stringa *moreTool* il relativo nome del tool da eliminare.

Successivamente si è andato a comprimere il file attraverso un processo chiamato minificazione. La minificazione è un processo per ridurre la dimensione di un codice sorgente rimuovendo da esso elementi inutili al compilatore (tipo le tabulazioni per indentare il codice, commenti, e altri elementi a seconda del linguaggio considerato) ma magari utili al programmatore durante la fase di produzione del codice. Questo processo ha ridotto quindi le dimensioni del file sorgente, cosa che in ambito Web è piuttosto utile, poichè meno è pesante un file da inviare al browser, meno tempo

la pagina ci metterà a caricarsi. La minificazione di solito si esegue con script Javascript o fogli CSS, raramente con HTML, poichè questi file vengono inviati direttamente al browser. Si è dovuto infine creare un nuovo file `Voyant.war` da sostituire al precedente nella cartella `webapp` di Tomcat. I tools che sono stati eliminati sono:

- Veliza
- Reader
- Dreamscape
- Correlation

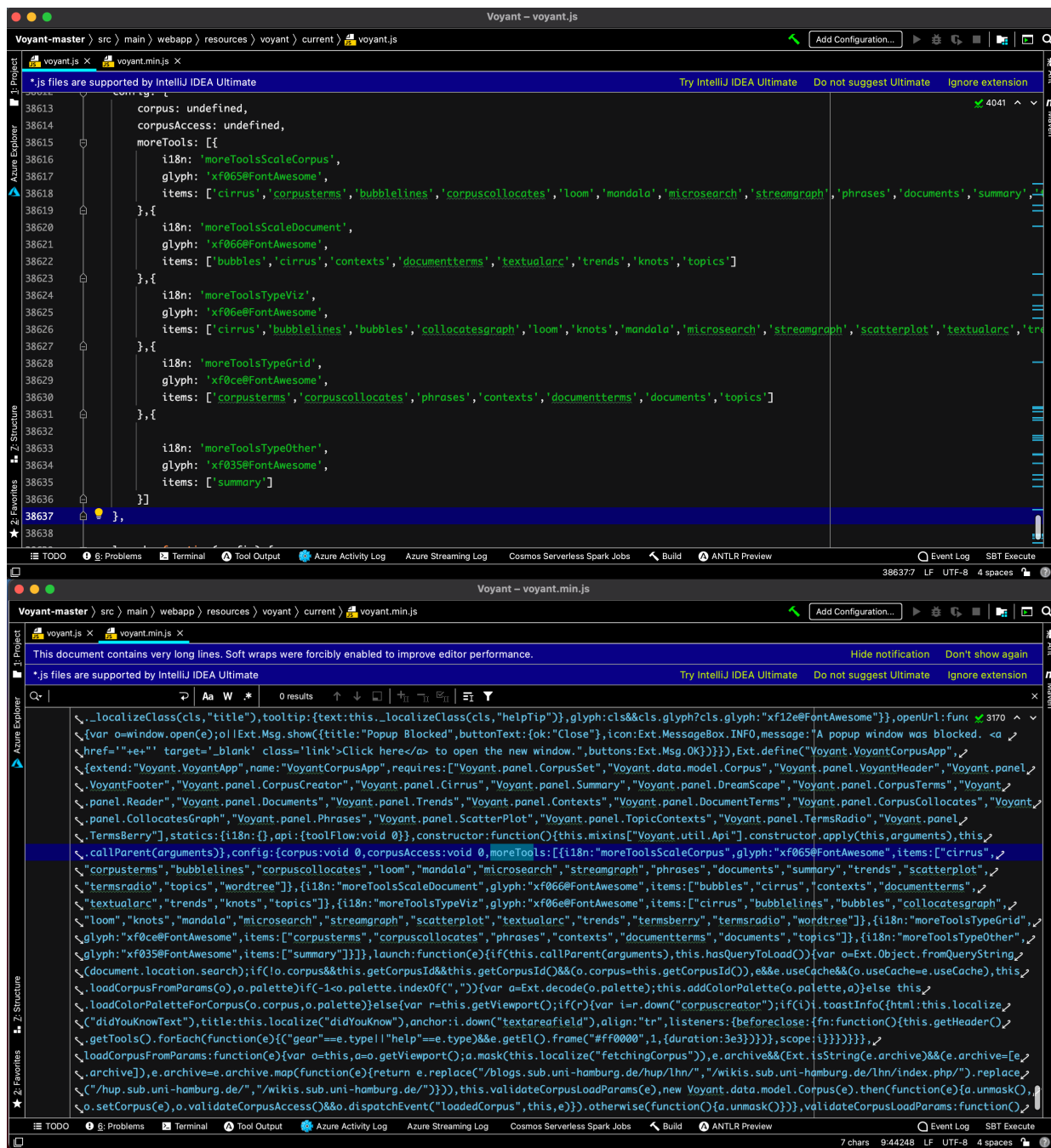


Figura 5.26: voyant.js

5.11 Tool aggiunti

Bubblelines

Visualizza la frequenza e la distribuzione di un termine all'interno di un corpus.

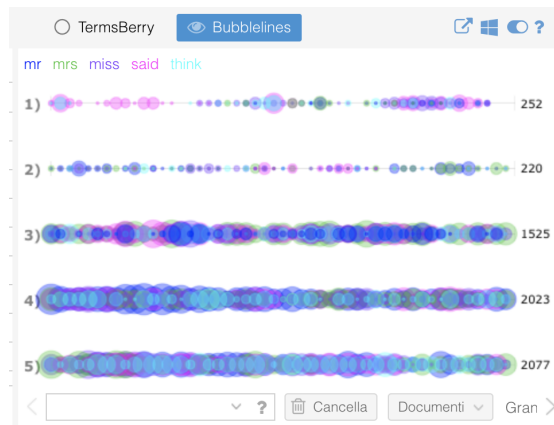


Figura 5.27: bubblelines

Bubbles

Una visualizzazione giocosa delle frequenze di un termine all'interno del documento

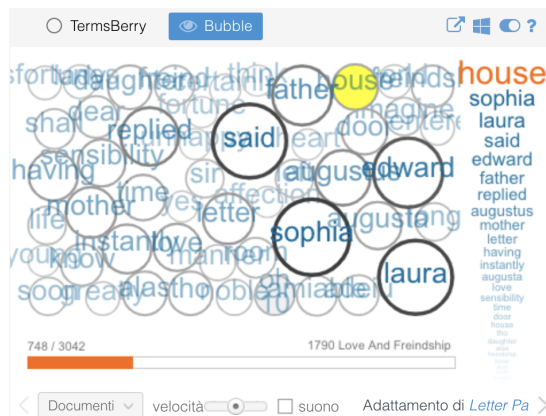


Figura 5.28: bubble

Cirrus

Una word cloud che visualizza le frequenze più alte delle parole di un corpus



Figura 5.29: cirrus

Collocates Graph

Rappresenta parole chiave e termini che si verificano in stretta prossimità come un grafo diretto di rete

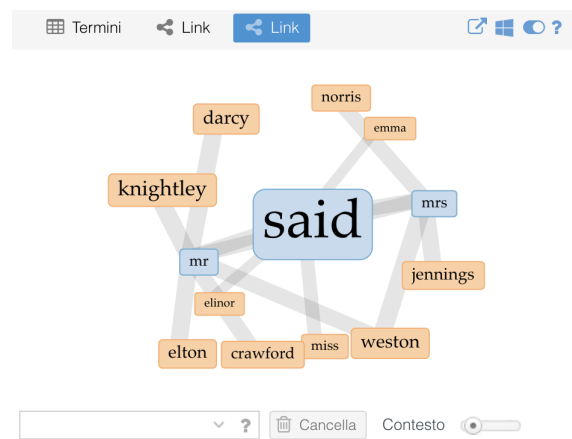
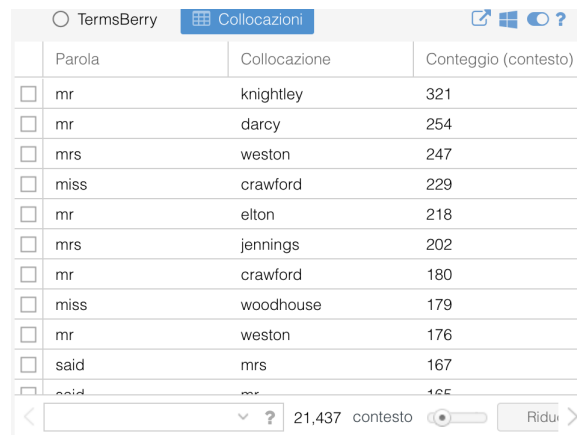


Figura 5.30: link

Corpus Collocates

E' una tabella di visualizzazione in cui i termini appaiono più frequentemente in prossimità di parole chiave nell'intero corpus.



Parola	Collocazione	Conteggio (contesto)
<input type="checkbox"/> mr	knightley	321
<input type="checkbox"/> mr	darcy	254
<input type="checkbox"/> mrs	weston	247
<input type="checkbox"/> miss	crawford	229
<input type="checkbox"/> mr	elton	218
<input type="checkbox"/> mrs	jennings	202
<input type="checkbox"/> mr	crawford	180
<input type="checkbox"/> miss	woodhouse	179
<input type="checkbox"/> mr	weston	176
<input type="checkbox"/> said	mrs	167
<input type="checkbox"/> said	mr	165

21,437 contesto

Figura 5.31: Corpus Collocates

Context

Mostra ogni occorrenza di una parola chiave con un po' di testo circostante. Può essere utile per studiare più da vicino come i termini vengono utilizzati in contesti diversi.



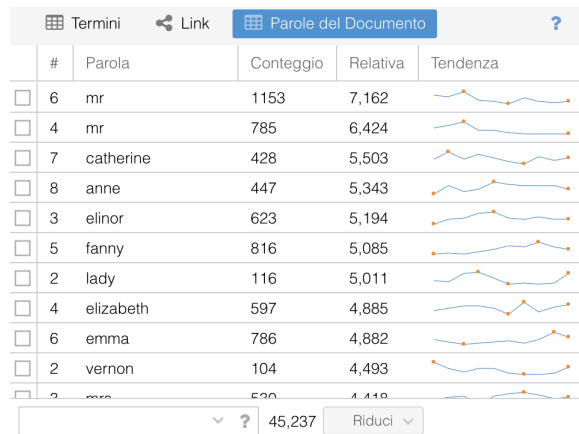
Documento	Sinistra	Parola	Destra
1) 1790 L...	genteel family since ...	mr	and Mrs Marlowe ar...
1) 1790 L...	A brother of Mrs Mar...	mr	Cleveland is with the...
1) 1790 L...	thirded by the entre...	mr	. Fitzgerald who is c...
1) 1790 L...	a Concert or a Ball.	mr	Marlowe is so desiro...
1) 1790 L...	Kickabout's; we wer...	mr	Fitzgerald who is a v...
1) 1790 L...	of my Charlotte at Br...	mr	and Mrs M. were the
1) 1790 L...	handsome?) The ele...	mr	Cleveland, his polis...
1) 1790 L...	conspicuous in the ...	mr	Cleveland. The appr...
1) 1790 L...	We have since often...	mr	and Mrs Marlowe bu...
1) 1790 L...	and whose only one...	mr	Whitaker, Mrs Lefroy...
1) 1790 L...	particulars of his life	mr	Shedden's play of th...

3,117 contesto

Figura 5.32: voyant.js

Document Terms

E' una tabella di visualizzazione delle frequenze dei termini per ogni documento.



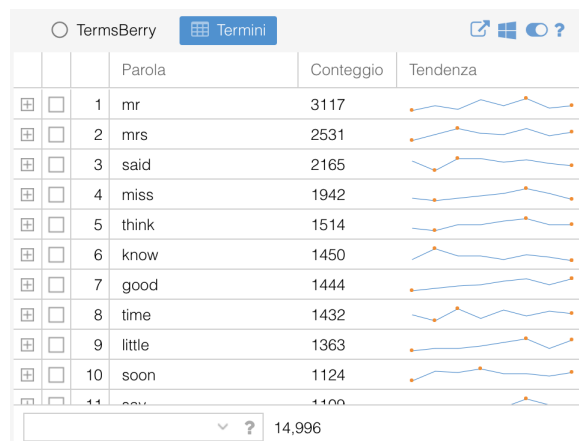
#	Parola	Conteggio	Relativa	Tendenza
6	mr	1153	7,162	
4	mr	785	6,424	
7	catherine	428	5,503	
8	anne	447	5,343	
3	elino	623	5,194	
5	fanny	816	5,085	
2	lady	116	5,011	
4	elizabeth	597	4,885	
6	emma	786	4,882	
2	vernon	104	4,493	
2	...	500	4,418	

45,237 Riduci

Figura 5.33: document terms

Corpus Terms

E' una tabella di visualizzazione delle frequenze dei termini nell'intero corpus.



	Parola	Conteggio	Tendenza
1	mr	3117	
2	mrs	2531	
3	said	2165	
4	miss	1942	
5	think	1514	
6	know	1450	
7	good	1444	
8	time	1432	
9	little	1363	
10	soon	1124	
11	...	1100	

14,996

Figura 5.34: corpus terms

Documents

Mostra una tabella dei documenti nel corpus e include funzionalità per modificare il corpus.

Termini Link Documenti					
	Titolo	Parole	Tipi	Ratio	Words/Sentence
1	1790 Love And Freindship	33,...	4...	13%	25.8
2	1805 Lady Susan	23,...	2...	13%	25.2
3	1811 Sense and Sensibility	119,...	6...	5%	23.9
4	1813 Pride and Prejudice	122,...	6...	5%	20.7
5	1814 Mansfield Park	160,...	8...	5%	23.6
6	1815 Emma	160,...	7...	5%	19.2
7	1818 Northanger Abbey	77,...	6...	8%	22.2
8	1818 Persuasion	83,...	5...	7%	23.3

0 Modifica Download

Figura 5.35: documents

Knots

E' una visualizzazione creativa che rappresenta i termini in un unico documento come una serie di linee contorte. Ogni ricorrenza di un termine è rappresentata da una piega nella linea, quindi più una linea è attorcigliata, più un termine si ripete e gli allungamenti dritti non rappresentano alcun evento.



Figura 5.36: nodi

Loom

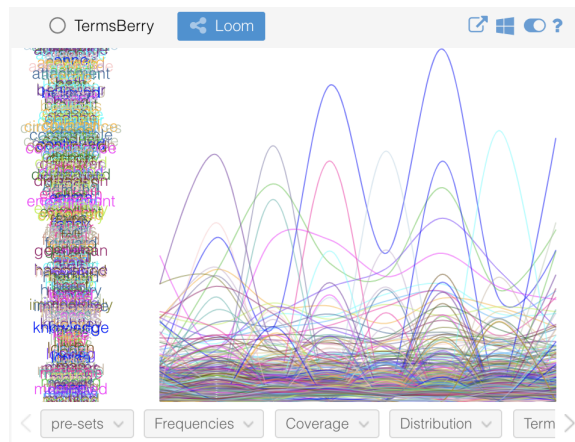


Figura 5.37: loom

Mandala

E' una visualizzazione concettuale che mostra le relazioni tra termini e documenti. Ogni termine di ricerca (o magnete) attira i documenti verso di esso in base alla frequenza relativa del termine nel corpus.

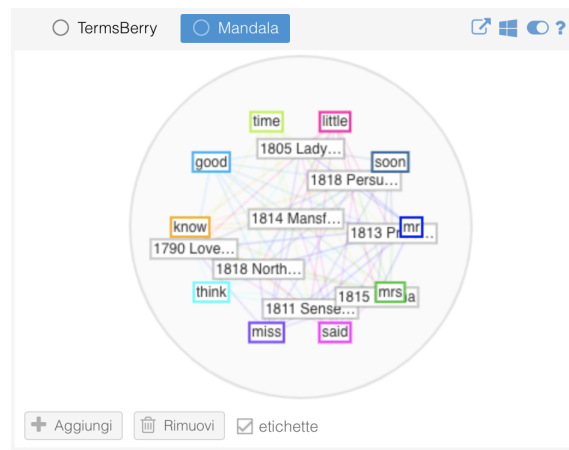


Figura 5.38: mandala

Microsearch

Visualizza la frequenza e la distribuzione dei termini in un corpus.

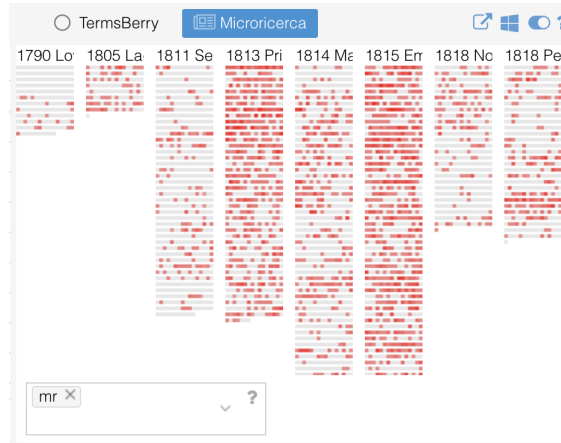


Figura 5.39: microsearch

Phrases

Mostra sequenze ripetute di parole organizzate per frequenza di ripetizione o numero di parole in ogni frase ripetuta.

Termini	Conteggio	Lunghezza	Tendenza
<input type="checkbox"/> what reverse we have ...	2	28	
<input type="checkbox"/> my first displays the w...	2	17	
<input type="checkbox"/> you and miss smith an...	2	16	
<input type="checkbox"/> another view of man m...	2	15	
<input type="checkbox"/> the loss of mary i must...	2	14	
<input type="checkbox"/> for he would carve the...	2	13	
<input type="checkbox"/> the borders of an exte...	2	13	
<input type="checkbox"/> i did not come to bath ...	2	11	
<input type="checkbox"/> mrs vernon to lady de ...	2	11	
<input type="checkbox"/> the shape of the eye a...	2	11	
<input type="checkbox"/> epped that the best th...	2	10	

< 27,875 | Lunghezza >

Figura 5.40: phrases

ScatterPlot

E' una visualizzazione grafica del modo in cui le parole si raggruppano in una somiglianza di un documento corporeo, analisi della corrispondenza o analisi delle componenti principali.

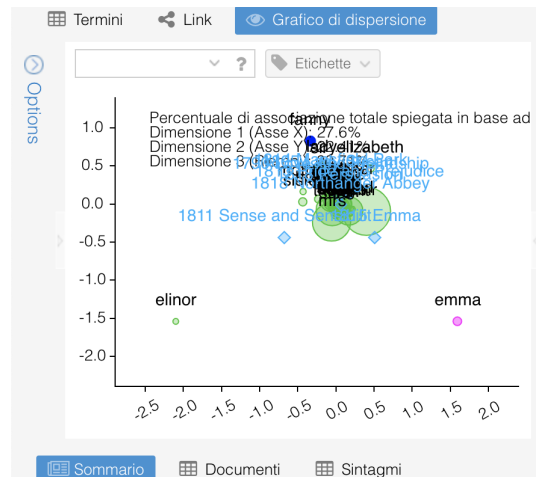


Figura 5.41: scatter plot

StreamGraph

E' una visualizzazione che rappresenta il cambiamento della frequenza delle parole in un corpus (o all'interno di un singolo documento).

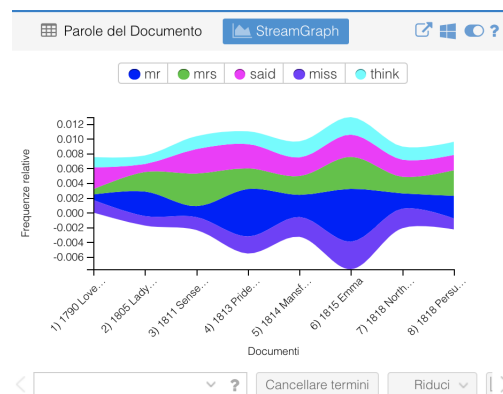


Figura 5.42: stream graph

Summary

Fornisce una semplice panoramica testuale del corpus corrente, incluso (se applicabile per più documenti) numero di parole, numero di parole uniche, documenti più lunghi e più brevi, densità di vocabolario più alta e più bassa, numero medio di parole per frase, più frequente parole, notevoli picchi di frequenza e parole distintive.



Figura 5.43: summary

TermsBerry

Fornisce un modo per esplorare i termini ad alta frequenza e i loro collocati (parole che si trovano in prossimità).

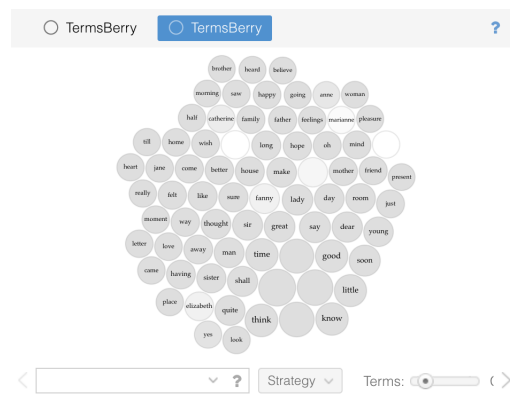


Figura 5.44: termsberry

TermsRadio

E' una visualizzazione che rappresenta il cambiamento della frequenza delle parole in un corpus (o all'interno di un singolo documento).

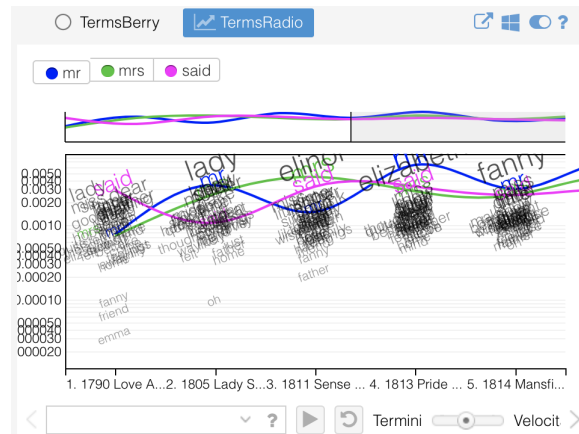


Figura 5.45: termsradio

TextualArc

E' una visualizzazione dei termini in un documento che include un centroide ponderato dei termini e un arco che segue i termini nell'ordine del documento.

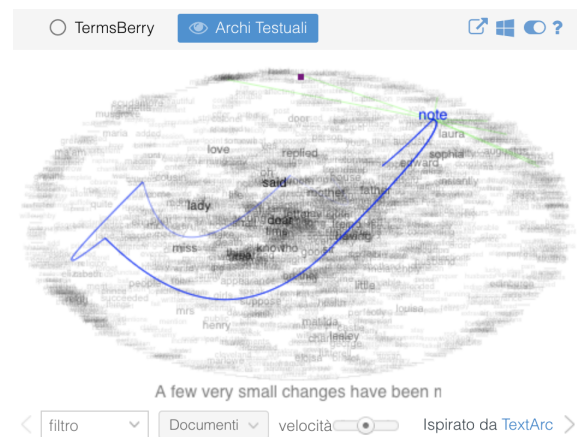


Figura 5.46: textual arch

Topics

Fornisce un modo rudimentale per generare cluster di termini da un documento o corpus e quindi vedere come ogni argomento (cluster di termini) è distribuito nel documento o corpus.



Figura 5.47: topics

Trends

Mostra un grafico a linee che rappresenta la distribuzione dell'occorrenza di una parola in un corpus o in un documento.

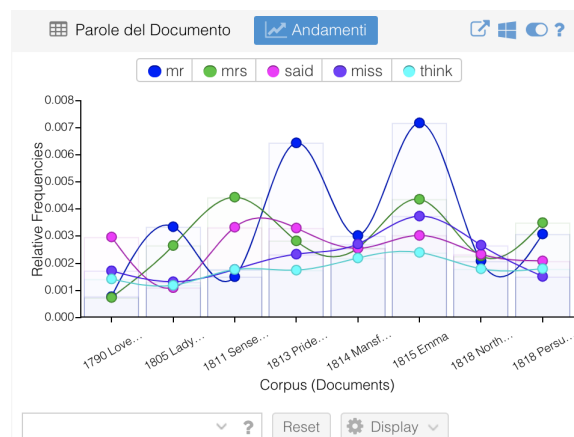


Figura 5.48: trends

WordTree

Consente di esplorare come le parole chiave vengono utilizzate nelle diverse frasi del corpus.

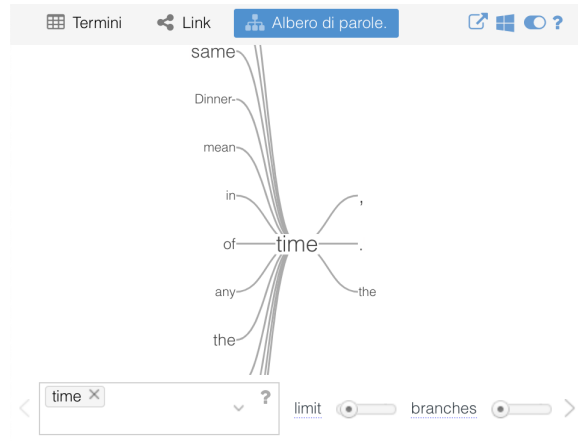


Figura 5.49: word tree

Capitolo 6

Funzionalità aggiunte attraverso Voyeditor

Grazie al digitale oggi è possibile pubblicare tutti i documenti relativi ad una determinata opera, senza ricorrere ad una presentazione selettiva dei materiali, prospettiva ben più diversa da quella tradizionale che, predilige la ricostruzione di un originale presunto sia al documento sia all'evidenza storica. Tale nuovo approccio digitale favorisce lo sviluppo di progetti sotto forma di archivio ed edizione, anche per trascrizioni di singoli testimoni o di casi tradizionali molto complessi. Si chiarificherà adesso l'utilità concreta, attraverso esempi pratici, di creazioni di indici di frequenza, concordanze e della evidente implementazione e facilitazione che *Voyeditor* consente a questo tipo di studi.

In primo luogo le informazioni quantitative che si possono ritrovare in un testo ci offrono due possibilità:

- trovare una "parola" in contesti testuali molto ampi;
- calcolare il numero di occorrenze della parola.

La grand parte delle applicazioni oggi si basa sulla semplice individuazione dell'indice di frequenza, invece *Voyeditor* intende avviare una automatizzazione della procedura per "parole"; in futuro sarà possibile aprirsi ad un ulteriore sviluppo di questo ambiente web procedendo per concetti ed ambiti semantici, anche se meno

facilmente automatizzabili.

In questo studio, per *parola* si intende una qualsiasi sequenza di lettere nella catena parlata separata da spazi, per *occorrenza* le apparizioni di una parola in un testo, per *forma* la varietà lessicale, per *lemma* un'unità grafica che costituisce l'intestazione e per *cooccorrenza* una combinazione di due o più parole che tendono a presentarsi insieme.

La filologia tradizionale è stata sempre orientata al riconoscimento della paternità autoriale di testi anonimi, in questo studio invece si propone un percorso ben strutturato che costa di una prima fase di un'analisi stilistica vera e propria:

- la distribuzione della frequenze di parole;
- la misura della varietà lessicale;
- l'analisi di parole chiave;
- le concordanze;
- le cooccorenze.

Alla presentazione di concetti e metodologie seguirà una proposta di casi applicativi concreti mirati a fornire degli esempi che spieghino in che modo l'implementazione di Voyant tool abbia arricchito l'ambiente web preesistente indirizzando pragmaticamente il lettore nel panorama vasto e mutevole dell'analisi testuale digitale.

Le prossime sezioni forniranno degli esempi operativi di analisi del testo utilizzando alcuni dei tools ritenuti più efficaci.

6.1 TextualArch

In questo esempio, vengono individuate le parole più ricorrenti che sono disposte lungo una forma ellittica ed evidenziate con colori più o meno intensi a seconda della frequenza. Inoltre, le singole parole sono però collegate in successione con linee curve, non con tutte le parole presenti, ma solo con alcune.

Selezionando una parola di nostro interesse possiamo visualizzare sia i legami

indicati da linee rosse che dalla parole si dirigono in vari punti dell'ellissi, sia possiamo seguire la sequenza del racconto attraverso le linee curve azzurre.

Nell'esempio riportato in figura 6.1 nella fiaba di Cappuccetto Rosso i personaggi appaiono più o meno decentrati a seconda dell'importanza e della rilevanza che hanno all'interno del racconto: Cappuccetto Rosso non solo si trova al centro, ma presenta un numero maggiore di legami con le altre parole rispetto, ad esempio, al personaggio del lupo che, come si evince nella figura 6.2, è più decentrato e presenta un minor numero di linee che lo legano agli altri termini del testo.

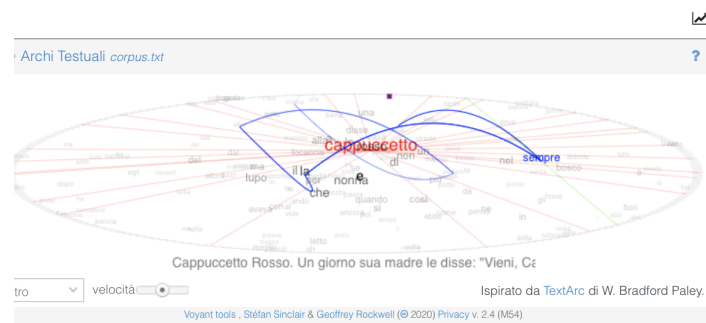


Figura 6.1: personaggio principale

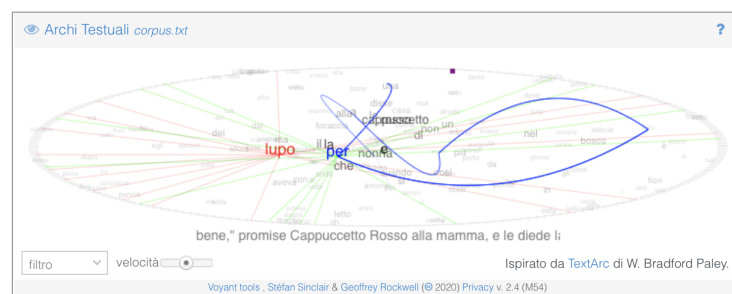


Figura 6.2: personaggio secondario

6.2 Cirrus

In Cirrus le parole con un numero maggiore di occorrenze sono rappresentate proporzionalmente in una dimensione più grande, in una nuvola che, affiancata al

testo, già ci permette di individuare una serie di parole ricorrenti che, se pur in modo casuale, già ci forniscono un'idea della tematica centrale del testo. Nella figura 6.3 appaiono parole chiave fondamentali come right, equal, freedom, social, protection, law.



Figura 6.3: nuvola di default

Questa prima visualizzazione, sicuramente abbastanza efficiente, potrebbe però essere inficiata dalla presenza di parole appartenenti alla stop list, contenente stop words parole vuote come articoli e congiunzioni, non significative per l'analisi da effettuare. Questa opzione può essere facilmente modificata, aggiungendo alla lista la parola che si vuole eliminare dall'analisi come viene mostrato in figura 6.4. La nuvola appare adesso così modificata restringendo il campo visivo alle parole

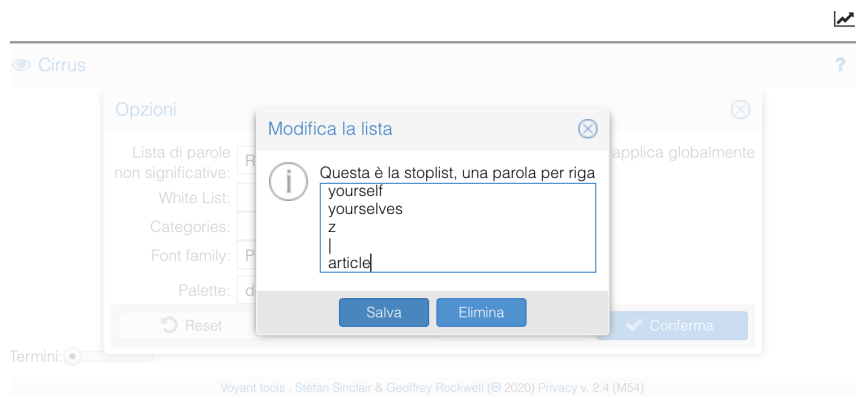


Figura 6.4: aggiornamento della stop list

pregnanti di contenuto: la sola visione della nuvola consente al filologo di avere un'idea immediata del cuore della trattazione con la messa in evidenza delle parole chiave.

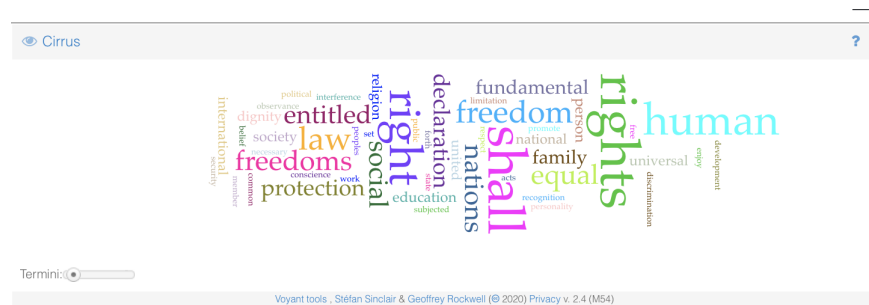


Figura 6.5: nuvola modificata

6.3 Parole nel documento

Questo tool individua la frequenza assoluta e la frequenza relativa delle parole affiancata da un grafico con una linea di tendenza che riporta l'incidenza e la posizione della parola all'interno del testo.

Confrontando questi dati notiamo che il valore della parola aumenta proporzionalmente al numero di volte che compare nel documento, ma tale funzione può essere utilizzata al contrario, cioè per identificare e filtrare le parole rare o *hapax* come parole rilevanti e significative. Ad esempio il titolo della raccolta di poesie di Pier Paolo Pasolini *Trasumanar e organizzar (1971)* riprende il celeberrimo verbo, *trasumanar hapax* nella *Commedia* (Par I, 70).

Oltre all'analisi quantitativa si propone analizzare la qualità e misurare la stabilizzazione e la variazione di un segmento originale e innovativo del vocabolario, in questo caso dantesco; quindi è stato isolato un lessema da classificare come *hapax*, rilevante per lo studio delle lingue scomparse e in filologia e fondamentale per stabilire ad esempio l'attribuzione di un'opera ad un autore o mettere in discussione attribuzioni precedenti. Attraverso questo tool, come mostrato nella figura 6.6, si nota la dichiarata citazione di un *hapax* dantesco da parte di Pasolini per sotto-

	#	Parola	Conteggio	Relativa	Tendenza
<input type="checkbox"/>	1	tesserti	1	1,629	
<input type="checkbox"/>	1	tolemaico	1	1,629	
<input type="checkbox"/>	1	tondo	1	1,629	
<input type="checkbox"/>	1	trasumanar	1	1,629	
<input type="checkbox"/>	1	tre	1	1,629	
<input type="checkbox"/>	1	tutta	1	1,629	
<input type="checkbox"/>	1	tutti	1	1,629	

▼ ? 298

Voyant tools - Stefan Sinclair & Geoffrey Rockwell (© 2020) Privacy v. 2.4 (M54)

Figura 6.6: individuazione parole rare

lineare meglio il concetto (*trasumar ovvero ascesa spirituale e l'organizzazione*)

1

¹Pasolini intitola la sua raccolta in versi *Trasumanar e organizzar, 1971* usando un hapax dantesco e il suo esempio viene successivamente ripreso in iniziative pubbliche, manifestazioni, ecc.

Capitolo 7

Conclusioni

Questa tesi presenta una soluzione efficiente per fornire delle funzionalità aggiuntive all'ambiente web già esistente KwickwocKwac. I risultati ottenuti rispettano i vincoli di progetto discussi in fase di analisi.

L'applicazione web permette all'utente, prettamente uno studente o un filologo, di poter lavorare su di un testo e di poter avere a disposizione tutte le funzionalità che offre Voyant, in una nuovo ambiente web chiamato *Voyeditor*.

Per comprendere al meglio la finalità del progetto è stato necessario iniziare la trattazione con una panoramica generale sul mondo della filologia e dell'edizione digitale. Seppur in sintesi, per la chiara scelta di voler privilegiare l'aspetto più concreto dell'esposizione puntuale e approfondita delle funzionalità ideate che questo studio esige, si è ripercorso lo sviluppo e la storia di un incontro tra filologia e informatica che può agevolare l'analisi dei testi letterari antichi, arricchendone e rendendo più veloce e preciso lo studio. Oltre ai cenni storici, sono stati anche proposti alcuni brevi esempi di ambienti web arricchiti da funzionalità aggiuntive a titolo esemplificativo.

Attraverso l'utilizzo del server web Tomcat, si è cercato un modo di adattare l'implementazione realizzata, affinché potesse essere utilizzata al meglio in futuro e fosse facilmente adattabile alla nuova versione.

Successivamente sono stati introdotti gli strumenti e le tecnologie utilizzate per la realizzazione del progetto, spiegando cos'è un'applicazione web ed elencando le

funzionalità presenti nell'ambiente già esistente fornendo una piccola introduzione sulle funzionalità da aggiungere.

La trattazione è stata, infine, condotta con taglio strettamente applicativo per offrire esempi di utilizzo delle metodologie e dei tools integrati e favorire curiosità e sperimentazione nel settore filologico.

E' stata infatti creata e resa disponibile una demo caricata sul server dell'università a cui si può facilmente accedere attraverso il link:

<http://chiara.minervino.tw.cs.unibo.it/Voyeditor/#>

Alcuni esempi di utilizzo di questa demo hanno voluto fornire una prima dimostrazione pratica sull'uso delle procedure di nuovi tools, anche se in realtà le potenzialità operative sono molto ampie grazie alla possibilità di scegliere tra molti altri tools. Ad esempio, nel caso di *Cirrus*, è stata presentata una modalità di visualizzazione di parole per arrivare in modo più efficiente alla visione immediata delle tematiche centrali.

Per quanto riguarda i possibili sviluppi futuri si dovrà aggiungere l'implementazione dei tools *Voyeditor* alla nuova versione di KwicKwockKwac che permetterà di avere delle ulteriori funzionalità che riguardano il login, la creazione di metadati e l'upload di un file in qualsivoglia formato.

Un traguardo importante da raggiungere in ambito filologico sarebbe la possibilità di poter operare non solo su un singolo testo ma su un corpus o su testi comparabili di autori diversi.

Risorse web

- a <https://dharc-org.github.io/vespasiano-da-bisticci-letters-de/documentation/index.html#overview>
- b TEI, <https://tei-c.org>
- c Tomcat <http://tomcat.apache.org>
- d Voyant <https://voyant-tools.org>
- e http://linuxdidattica.org/docs/altre_scuole/planck/tecnologie-web/tecnologie-web15.html
- f Voyant guide <https://voyant-tools.org/docs/#!/guide/trends>
- g Voyant Github <https://github.com/sgsinclair/VoyantServer/releases>
- h https://en.wikipedia.org/wiki/Voyant_Tools
- i https://en.wikipedia.org/wiki/Voyant_Tools
- j https://fem-modena.github.io/linguistica_strumenti-per-la-didattica/
- k <https://linda.education/strumenti/>
- l http://arkiv.iva.ku.dk/kolifeboat/SPECIFIC%20SYSTEMS/kwic_kwac_kwoc.htm

m https://en.wikipedia.org/wiki/Key_Word_in_Context

n <http://inmyownterms.com/kwic-kwac-kwoc-not-knock-knock-joke/>

o https://it.wikipedia.org/wiki/Apache_Tomcat

p <https://www.html.it/pag/16729/tomcat-e-apache/>

q <https://www.html.it/articoli/tomcat-lapplicazione-servita/>

r <https://it.nex-software.com/che-cos39e-tomcatexe>

s <https://it.wikipedia.org/wiki/XML>

t <https://www.webdomus.net/5-principi-fondamentali-usabilita-un-sito-web/>

u <http://www.mondomatica.it/usabilita.htm>

v <https://www.ionos.it/digitalguide/siti-web/>

[programmazione-del-sito-web/script-lato-server-e-lato-client-differenze](https://www.ionos.it/digitalguide/siti-web/programmazione-del-sito-web/script-lato-server-e-lato-client-differenze)

Bibliografia

- [1] "KwicKwocKwac 1.0, Guida all'ambiente di marcatura", 2020, Università di Bologna, <http://aldomorodigitale.unibo.it/documentazione>
- [2] Elena Pierazzo, *Digital Scholarly Editing: Theories, Models and Methods*, 2015, Université de Grenoble 'Stendhal', France
- [3] Matthew James Driscoll and Elena Pierazzo, *Digital Scholarly Editing Theories and Practices*, 2016, Open Book Publishers (Cambridge, UK)
- [4] Claudia Bonsi, Angelo Di Iorio, Paola Italia, Francesca Tomasi, Fabio Vitali, Ersilia Russo, "PhiloEditor: simplified HTML markup for interpretative pathways over literary collections", Proceedings of ACM SAC Conference, Lymassol, Cyprus, April 8-12, 2019
- [5] Lou Burnard, C. M. Sperberg-McQueen, *Manuale di codifica XML/TEI*, Documento N. TEI U5, Giugno 1995, Traduzione italiana di: Fabio Ciotti, Guendalina Demontis, Giuseppe Gigliozzi, Massimo Guerrieri, Andrea Loreti
Revisione e cura traduzione italiana di: Fabio Ciotti Gennaio 1998 Revisione versione XML a cura di: Francesca Tomasi
- [6] Francesca Tomasi, Fabio Vitali, "An integrated architecture for scholarly digital editing", 2012, Proceedings of "Culture & Technology" - 3rd European Summer School in Digital Humanities, Universität Leipzig (Germany), 23.-31.07.2012
- [7] A.Stussi, *Breve avviamento alla filologia italiana*, Bologna 2002, Il Mulino

- [8] Gioia Donati, *Philoeditor 3.0: Un web editor per la ricerca filologica*, Università degli studi di Bologna, 2016
- [9] Chiara Di Pietro, *EVT per le edizioni critiche digitali: progettazione e sviluppo di una nuova GUI basata sullo schema progettuale MVC*, Università degli studi di Pisa, 2015
- [10] Maria Luce Bottazzo, *Virtual research environment: una proposta di mashup application per digital scholarly editions*, Università di Bologna, 2017