UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

VERCELLI BOOK DIGITALE:
Visualisation Software for a Web-based Digital Edition

Candidato: *Francesca Capochiani*

Relatore: *Dott. Roberto Rosselli Del Turco*

Correlatore: *Dott.ssa Maria Simi*

Anno Accademico 2010-2011

An interface is humane if it is responsive to human needs
and is considerate of human frailties.
(Jef Raskin - The humane interfaces)

ABSTRACT

La versione 2.0 del software di visualizzazione per l'edizione digitale del *Vercelli Book*, antico manoscritto medievale conservato con segnatura CXVII presso l'archivio della Biblioteca Capitolare di Vercelli, è stata realizzata nell'ambito del progetto Vercelli Book Digitale per mettere a disposizione il manoscritto e i suoi contenuti a tutti gli studiosi e appassionati interessati alla sua consultazione.

Il software messo a punto permette di accedere ai contenuti testuali e alle riproduzioni fotografiche in facsimile del manoscritto originale. L'applicazione fornisce inoltre una serie di funzionalità che arricchiscono questa edizione: sono a disposizione degli utenti strumenti per la manipolazione delle immagini e per la ricerca testuale.

Progettare interfacce utente che siano facili ed efficaci da utilizzare è un processo complesso che richiede la conoscenza di principi di progettazione grafica, linee guida e un approccio strutturato per il raggiungimento di usabilità. Lo scopo di questa tesi è pertanto quello di trovare un approccio efficace alla progettazione grafica di un software di visualizzazione EVT. I contenuti, la modalità d'approccio, la definizione della tecnica, i requisiti funzionali sono stati attentamente valutati nella costruzione di un'interfaccia utente che esprimesse bene l'interazione uomo-macchina.

In molti casi, le persone che partecipano alla creazione di una GUI non seguono gli standard e le decisioni sono spesso basate su ciò che è meglio per loro. Di conseguenza, nella maggior parte dei casi, i programmi si rivelano inconsistenti e difficilmente utilizzabili. Dopo aver realizzato alcuni prototipi al fine di dimostrare un sistema di sviluppo di varia natura, il risultato è stato un prodotto Web-based.

Keywords: EVT, IMT, XHTML, CSS, GUI, XML, JQuery, Graphic Design, JavaScript, Human-Computer Interaction.

INDICE DEI CONTENUTI

CAPITOLO IV- IMPLEMENTAZIONE DEL PROTOTIPO

CAPITOLO V – CONCLUSIONI

INTRODUZIONE

Le edizioni digitali dei manoscritti sono una risorsa preziosa per gli studiosi e il grande pubblico. La loro disponibilità on-line offre la massima flessibilità e la possibilità di raggiungere il grande pubblico.

Questo progetto vuole mostrare come sia possibile offrire testi scientifici di alta qualità sul Web. Lo scopo della tesi è quello di creare uno strumento consultabile tramite i *browser web* per la creazione e l'editing di immagini basate attraverso l'edizione elettronica e gli archivi digitali di testi umanistici. Chiameremo questo software EVT (Edition Visualization Technology). Esso consentirà agli utenti di sfogliare, esplorare e studiare l'edizione digitale di uno dei più antichi manoscritti medievali: il *Codex Vercellensis*.

Il software di visualizzazione del *Vercelli Book* dovrebbe essere estremamente facile da usare e intuitivo anche per utenti con competenze informatiche di medio o basso livello.

La ricerca prevede l'introduzione di questa tesi con un accenno alle edizioni elettroniche di testi antichi, fornendo una sintesi delle informazioni di base. Dopo aver analizzato e studiato le soluzioni e le proposte esistenti è stato realizzato uno schema che riassume quali sono le caratteristiche del progetto (Capitoli I e II).

Una volta trattata questa parte di studio, è stata posta l'attenzione sull'implementazione del software spiegando in dettaglio quali sono state le scelte di programmazione, quale il software utilizzato e come possiamo migliorare EVT sul Web 2.0. (Capitolo III). Di seguito vengono mostrati i risultati ottenuti al termine della realizzazione e la creazione di prototipi EVT (Capitolo IV).

L'ultima parte della tesi, infine, suggerisce possibili direzioni di ricerca futura. (Capitolo V).

L'implementazione proposta in questa tesi è sufficientemente flessibile per essere utilizzata in altri progetti, offrendo un miglioramento rispetto alle produzioni esistenti. Speriamo che si tratti di uno strumento sufficientemente potente da offrire soluzioni reali nel settore delle edizioni digitali.

CAPITOLO I

LA CODIFICA DEI TESTI DIGITALI

1.1 La visualizzazione di edizioni digitali

C'è stato un tempo in cui l'accesso al materiale digitale era difficile e disponibile solo ad una ristretta cerchia di addetti ai lavori. Questi tempi sono superati e oggi l'edizione digitale è in grado di fornire informazioni non più unicamente ai ricercatori, ma a qualunque tipo di utenza grazie alle potenzialità del Web 2.0.

Lo scopo del progetto è quello di creare un set completo di strumenti flessibile e altamente personalizzabile sviluppato per consentire agli utenti di visualizzare, leggere e confrontare le edizioni in un ambiente elettronico. Chiameremo questo strumento EVT: Edition Visualization Technology.

EVT favorisce una visualizzazione finalizzata ad una migliore comprensione dei dati scientifici e cognizione umana. Consente ai ricercatori di visualizzare e studiare in un ambiente elettronico immagini di manoscritti strettamente connessi con i documenti dell'edizione: trascrizione, glossario etc.

Le unità di base di un EVT sono le immagini e testi, quest'ultimi direttamente correlati alla trascrizione e al tipo di edizione. EVT fornisce quindi un'analisi che coinvolge sia il livello testuale che quello iconografico.

Nell'ambito della consultazione di un'edizione elettronica trovano particolare interesse e intervento gli studiosi *umanisti*, principali fruitori di immagini e testi, e gli *informatici,* che troveranno interesse, per esempio, nei meccanismi di rappresentazione nell'ambito dell' *Information Retrieval* e della progettazione dell'interfaccia utente.

Tradizionalmente, i progetti realizzati fino a pochi anni fa presentavano note di testo senza l'accesso diretto con l'oggetto fisico con cui il testo era correlato. Inoltre, questi progetti difficilmente presentavano marcatori che associassero il testo con la pagina del manoscritto; non fornivano quindi una diretta sincronizzazione testo/immagine.
Ciò che ci si aspetta oggi da un visualizzatore per le edizioni digitali di alta qualità è la possibilità di fornire interventi editoriali, immagini ad alta risoluzione, note di testo, note esplicative, materiali bibliografici etc.
Pertanto, il concetto di software per la consultazione di edizioni digitali si sta evolvendo da semplice funzionalità di visualizzazione dell'immagine a fianco del testo a dispositivo dinamico in grado di interagire tramite immagini, testo e *markup*.

1.1.1 La codifica del testo

La codifica di un testo è un'attività interpretativa che comporta scelte e valutazioni da parte del codificatore. Lo scopo di una codifica digitale del testo è rendere il testo comprensibile a un calcolatore. Si tratta di scelte e valutazioni da parte dell'*encoder*, che non può essere considerato solo un trascrittore del testo, ma può essere visto come un traduttore[1] che trasforma il testo in una forma analizzabile attraverso strumenti informatici permettendo nuove forme di fruizione del testo.
Un documento codificato si pone come obiettivo ultimo quello di diventare una risorsa accessibile e disponibile per la condivisione. Per fare questo, bisogna utilizzare sistemi di rappresentazione e memorizzazione tecnica che seguano degli standard ben definiti. La proliferazione di strumenti diversi, ognuno indipendente e spesso incompatibile con gli altri, ha portato la comunità scientifica a realizzare progetti con l'obiettivo di sviluppare schemi generali per la codifica dei testi in ambito umanistico. Uno di questi progetti è la TEI: Text Encoding Initiative.

1.1.2 TEI: Text Encoding Initiative

La Text Encoding Initiative[2] (TEI) produce uno standard per la codifica di dati testuali in formato XML e mette a disposizione una ricca documentazione relativa

[1] Elena Pierazzo, *La codifica dei testi*, Carocci, Roma, 2005.
[2] Sito web del consorzio TEI, *Text Encoding Intiative*: http://www.tei-c.org/index.xml.

allo schema di codifica (*Guidelines for Electronic Text Encoding and Interchange*[3]). La TEI costituisce un progetto di ricerca internazionale, nato nel 1987 e sostenuto da tre importanti associazioni internazionali nel campo dell'Informatica Umanistica: l' ALLC (Association for Literary and Linguistic Computing[4]), l'ACH (Association for Computing and Humanities[5]) e l'ACL (Association for Computational Linguistics[6]). Dal 2000 la TEI ha formato un consorzio i cui uffici esecutivi risiedono a Bergen, in Norvegia. L'obiettivo principale della TEI è stabilire norme per la rappresentazione di materiali testuali in formato digitale.
Inoltre la TEI offre:

- l'interscambio e l'integrazione dei dati accademici,
- il supporto per tutti i testi, in tutte le lingue,
- le linee guida per l'oggetto da codificare,
- assistenza per lo specialista sulle modalità di codifica del testo.

Il lavoro del consorzio è stato portato avanti da diverse commissioni; sono stati pubblicati innumerevoli documenti tecnici e teorici: le prime specifiche provvisorie sono state pubblicate nel 1991 con il titolo *Guidelines for Electronic Text Encoding and Interchange, TEI P1* (dove la P sta per *Proposal*). Si tratta di linee guida:

- semplici, chiare e concrete;
- di facile consultazione per i ricercatori;
- che permettono una definizione rigorosa e un'efficiente elaborazione dei testi;
- conformi agli schemi di codifica esistenti.

Le informazioni vengono continuamente aggiornate e vengono rilasciate nuove versioni delle linee guida, ultima in ordine di tempo la TEI P5, rilasciata

[3] L'intera doumentazione delle *Guidelines* è consultabile a questo indirizzo: http://www.tei-c.org/Guidelines/.
[4] Sito web della ALLC, Association for Literary and Linguistic Computing: http://www.allc.org/.
[5] Sito web della ACH, Association for Computing and Humanities: http://www.ach.org/.
[6] Sito web della ACL, Association for Computational Linguistic: http://www.aclweb.org/.

ufficialmente nel Novembre del 2007. La versione della TEI *Guidelines* utilizzata per il progetto del *Vercelli Book Digitale* è appunto la P5; essa presenta molte importanti novità quali: descrizioni dei manoscritti, supporto dei namespace XML, nuovi meccanismi di collegamento e altri miglioramenti tecnici[7]. Nell'ultima versione è stato anche inserito un nuovo modulo, *Manuscript Description*[8], per descrivere in maniera dettagliata un testo manoscritto. Questo modulo, tramite l'elemento principale <msDesc>, consente di descrivere il manufatto non solo per quanto riguarda il contenuto, ma anche per fornire notizie riguardanti dati generali come il luogo attuale di conservazione.

Le *Guidelines* forniscono quindi uno strumento per rendere esplicite determinate caratteristiche di un testo in modo tale da facilitarne l'elaborazione mediante programmi informatici basati su diverse piattaforme.

Esistono diversi approcci per la visualizzazione di un file XML TEI. I principali sono: Anastasia, Pixelise e l'uso di fogli di stile XSLT per produrre documenti XHTML (eventualmente accompagnati da fogli di stile CSS).

- *Anastasia* – è un'applicazione Web scritta in C il cui codice sorgente è altamente integrato nel server Apache. I sistemi commerciali XML all'epoca della realizzazione di Anastasia erano veramente costosi e molto lenti, per questo Anastasia era pionieristica. Ha permesso la realizzazione di molte edizioni elettroniche, favorendo il successo, per esempio, dell'edizione del digitale del *Canterbury Tales*[9].
- *Pixelise*[10] – è un sistema di pubblicazione dei documenti XML. Realizzato nel 2008 da Andrew West, fornisce un accesso molto potente a XQuery (da DB XML) attraverso oggetti Python. Le query sono eseguite in piccole funzioni Python, ma la presentazione è separata (sul modello di Django). Con *Pixelise* si può eseguire qualsiasi tipo di XPath o XQuery. Per il lavoro sul lato server XML, l'approccio object oriented per XML non può essere battuto

7 Per approfondimenti si consulti la documentazione ufficiale: http://www.tei-c.org/Guidelines/P5/
8 Documentazione del modulo: http://www.tei-c.org/release/doc/tei-p5-doc/en/html/MS.html.
9 Il progetto Canterbury Tales è disponibile a questo indirizzo: http://www.canterburytalespro-ject.org/.
10 Sito ufficiale di *Pixelise*: https://launchpad.net/pixelise/.

in termini di flessibilità e facilità d'uso. Pixelise potrebbe avere un impatto davvero grande in situazioni come la gestione aziendale o goverantiva, dove si dispone di una notevole mole di XML da elaborare e pubblicare.

- XSLT[11] - *Extensible Stylesheet Language* permette di trasformare documenti XML in altri formati quali XML, come XHTML, XSLFO.
- XHTML – è il linguaggio di scripting tradizionale per la gestione ed elaborazione del testo.

Realizzare un' edizione digitale di un manoscritto è un esempio di "uso intelligente" della TEI P5. Infatti uno degli obiettivi del progetto è quello di indagare le tecnologie che facilitano la gestione, il reperimento e l'annotazione di set di immagini ad alte risoluzioni, soprattutto se il corpus è composto da molte immagini.

1.2 I contenuti del manoscritto

Creare il software di visualizzazione del *Codex Vercellensis* è il compito di questa tesi. Il *Codex Vercellensis* (o *Vercelli Book*) è un'antica collezione di omelie e componimenti poetici in inglese antico che risale alla fine del X secolo, di argomento religioso, relativo a componimenti poetici in versi e in prosa.

Il prototipo costruito fornisce la raccolta delle immagini e dei testi del manoscritto attraverso lo sviluppo di un ambiente di visualizzazione il più estensibile possibile. Un ambiente che fornirà uno strumento generico di visualizzazione per i ricercatori i quali saranno in grado di gestire collezioni di immagini arbitrarie. L'efficacia della codifica delle immagini in un formato aperto (quali ad esempio, TIFF e JPG) è stata valutata e attuata nell'ambito del progetto ponendo l'attenzione sui problemi associati al recupero delle immagini e alla flessibilità in termini di set di dati.

Il manoscritto è conservato a Vercelli, presso la Biblioteca Capitolare[12] della Cattedrale dedicata a S. Eusebio, con segnatura CXVII e si compone di 136 fogli di pergamena sottile.

[11] XSLT nelle specifiche del W3C: http://www.w3schools.com/xsl/.
[12] Bibiloteca e Archivio Capitolare di Vercelli: http://www.tesorodelduomovc.it/.

I fogli misurano 31x20 cm circa e ciascuno contiene da 23 a 32 linee di testo. Il *Vercelli Book* contiene 23 omelie in prosa e 6 composizioni poetiche. L'importanza di questo codice è dovuta al fatto che è considerato uno dei più antichi testi in lingua inglese, insieme al *Exeter Book*, al manoscritto *Cotton Vitellius* e al manoscritto *Junius* (tutti i codici che risalgono alla fine del X secolo). Questi quattro codici costituiscono circa il 90% di tutta la produzione poetica anglosassone.

Questo manoscritto è di particolare interesse per la storia della letteratura religiosa del periodo, così come per gli studiosi di prosa anglosassone.

La presenza del manoscritto è stata confermata a Vercelli a partire dall'inizio del XII secolo. Ma il percorso che, ad un certo punto nel XI secolo, ha portato questo codice redatto nel sud dell'Inghilterra fino alla città piemontese come la sua sede definitiva è ancora poco noto: secondo l'ipotesi prevalente si tratterebbe del dono di un pellegrino, grato per l'ospitalità ricevuta a Vercelli durante un lungo viaggio a Roma.

Il dono del manoscritto al religioso testimonia l' importanza che Vercelli ha avuto per lungo tempo nel Medioevo come un punto di scambio e di sosta lungo il percorso pellegrinaggio a Roma.

1.3 Costruire una "buona" edizione digitale

Costruire l'edizione digitale di un manoscritto è lavoro che richiede una buona dose di meticolosità e pazienza. Il progetto prevede risorse materiali, umane, computazionali e finanziarie. Per questo motivo questa tipologia di progetti è sempre il risultato di un team di lavoro di studiosi. L'uso della parola "buona"[13] riferito alla costruzione dell'edizione digitale richiede alcune precisazioni. Lo sviluppo delle collezioni digitali attraversa oggi una fase di pieno sviluppo e maturazione: non è più sufficiente creare delle edizioni che siano solamente consultabili.

Una "buona" edizione richiede livelli di usabilità e accessibilità; in ultimo, ma non meno importante, deve essere in grado di interagire con l'utente.

[13] Per un'ulteriore documentazione si consulti l'articolo *Guides to Good Practice website* : http://www.ahds.ac.uk/creating/guides/index.htm.

Per fare ciò occorre innanzitutto bisogno di suddividere il progetto in sotto-processi, ognuno con il suo obiettivo. Questi piani determineranno la creazione del prototipo finale.
Ci sono pertanto quattro piani di intervento da individuare:

1. trascrizione del testo,
2. scansione digitale,
3. restauro del manoscritto,
4. progettazione del software di visualizzazione.

In questo contesto le prime tre parti del lavoro non riguardano lo sviluppo di questa tesi dal momento che sono stati già precedentemente elaborati; l'attenzione andrà quindi esclusivamente sulla progettazione del software di visualizzazione. Ma prima, vediamo nel dettaglio quali sono stati i primi tre passaggi e in cosa consistono:

1. *La trascrizione del testo*. La trascrizione del *Vercelli Book* è stata realizzata utilizzando XML. Si tratta di un linguaggio standard di markup, ben documentato e ampiamente utilizzato per la produzione di documenti facilmente modificabili e convertibili in diversi formati. L'adozione di XML ha portato alla scelta di uno schema di codifica: la scelta è ricaduta sullo schema di codifica basato sulla TEI P5.
2. *Scansione digitale.* L'acquisizione digitale delle immagini del manoscritto richiede standard aperti e ben documentati. I formati di memorizzazione che sono stati scelti sono TIFF e JPEG, corredati dai relativi metadati in XML.
3. *Restauro del manoscritto.* Per quanto riguarda il restauro digitale[14] del manoscritto, i risultati ottenuti con tecniche tradizionali (fibre ottiche e UV) non erano apprezzabili.
4. *Progettazione del software di visualizzazione.* Il mio progetto riguarda l'implementazione di un software per la consultazione dell'edizione digitale. È

[14] Leoni C., (2011), "Tecniche di restauro virtuale di manoscritti medievali e codifica dei risultati per una edizione digitale".

stato applicato al caso del *Vercelli Book*, ma è inteso come un prototipo generale applicabile in futuro ad altri tipi di manoscritti e risorse simili.

1.4 Pianificazione del progetto

Nel processo di costruzione di un'edizione digitale, la realizzazione del software è solo una parte di tutto il lavoro. È quindi necessario suddividere il planning in sotto-processi di realizzazione del software finale. Si possono così distinguere le fasi di realizzazione del progetto in tre fasi:

1. *Progettazione della GUI* (Capitolo II). Il progetto è iniziato con un lungo periodo di studio per la progettazione e sviluppo dell'interfaccia utente più adatta per l'applicazione. Questa fase ha visto discussioni e scambi con le persone coinvolte nel progetto e l'analisi delle attuali applicazioni simili.
2. *Implementazione dell'architettura del software* (Capitolo III). Una volta scelta l'interfaccia grafica si procede alla realizzazione effettiva del prototipo. Applicando il principio del "modello a cascata", è stato definito il tipo di linguaggio di programmazione da usare e i requisiti base per procedere alla definizione del più adeguato modello di GUI.
3. *Creazione del prototipo finale* (Capitolo IV). Questo studio ha portato alla creazione di diversi mock-up della GUI e alla scelta del modello finale. Il lavoro si è concentrato sull'attuazione pratica dell'interfaccia e la progettazione di alcune delle caratteristiche e funzionalità che integrassero l'EVT.

1.5 Linee guida del progetto

L'approccio utilizzato per migliorare l'interfaccia grafica si basa sui risultati ottenuti nel campo dell' HCI[15] (Human-Computer Interaction).

Questi studi tengono conto di aspetti di informatica, progettazione grafica e psicologia finalizzati allo studio del comportamento degli utenti.

[15] La principale risorsa sul concetto di *Human-Computer Interaction* è reperibile sul Web a questo indirizzo: http://hcibib.org/.

L'analisi delle edizioni digitali ci permette di tracciare un primo passo verso la realizzazione di una buona interfaccia. Per implementare la GUI sarà quindi necessario:

1. capire quali sono le tipologie di potenziali utenti che utilizzeranno il software;
2. decidere la modalità di presentazione del contenuto;
3. decidere come muoversi nell'ambito della navigazione;
4. pensare a come rendere disponibili tutte le funzioni e gli strumenti che verranno implementati.

Per la realizzazione dobbiamo attenerci alle norme e agli standard che riguardano l'usabilità. Inoltre abbiamo bisogno di:

1. un sistema di navigazione chiaro ed efficace;
2. strumenti per la gestione delle immagini;
3. la possibilità di visualizzare testo e immagini simultaneamente;
4. strumenti per l'analisi e la ricerca testuale;
5. usare una gerarchia visiva per garantire che le cose più importanti siano facilmente reperibili;
6. utilizzare etichette di navigazione che evitino un comportamento imprevisto per l'utente;
7. identificare regioni cliccabili.

Tutto questo è stato realizzato usando le tecnologie messe a disposizione oggi dal Web 2.0. Possiamo ora presentare una serie di criteri di "buona pratica" per la creazione delle interfacce utente. Questi criteri non devono essere confusi con i requisiti specifici presentati nel Capitolo III.

- *Utilizzo delle metafore*. È necessario ricorrere all' uso di metafore per trasmettere concetti e caratteristiche comprensibili all'utente. Le metafore

aiutano gli utenti a cogliere immediatamente anche i dettagli più minuti del modello concettuale.

- *Coerenza.* Permette agli utenti di trasferire conoscenze e competenze già acquisite. Un esempio di questo è l'uso dei colori standard di collegamento. Il blu, ad esempio, indica che l'utente non ha ancora accesso a un link, mentre il rosso o il viola indicano i collegamenti alle pagine già viste. I principi di coerenza prevedono che azioni simili siano dichiarate ed eseguite nello stesso modo e nelle stesse circostanze.
- *Stabilità.* L'interfaccia deve essere comprensibile, prevedibile e familiare. Non bisogna assegnare un nuovo comportamento a funzioni esistenti: questo incrementerebbe soltanto il livello di confusione per l'utente.
- *Uniformità.* Integrità significa che l'informazione deve essere ben organizzata, non solo sotto il profilo contenutistico, ma anche sotto il profilo estetico.
- *Progettazione grafica.* La grafica deve tralasciare dettagli insignificanti. Il design deve essere attraente e ben visualizzabile su schermi a risoluzione diversa. La grafica deve completare l'argomento, non sminuirlo. Le immagini e tutte le opzioni grafiche devono essere rese ottimali dove possibile.
- *Controllo.* L'utente deve avere il controllo dell'ambiente di lavoro. Non deve imparare funzioni complicate. I progettisti devono fornire degli indicatori, dei comandi per la gestione degli errori e consentire l'inversione di azioni.

Questi obiettivi individuali possono essere incapsulati in un quadro gerarchico. È possibile trasporre nel campo della progettazione la *Gerarchia dei Bisogni* di Maslow[16].

L'idea della *Gerarchia dei Bisogni*, adattata al design si basa sul presupposto che un progetto deve soddisfare i bisogni di base prima di poter soddisfare le esigenze di livello superiore. La *Gerarchia* procede dal basso verso l'alto e vede i seguenti

[16] Si tratta di una teoria psicologica proposta da Abraham Maslow nel 1943 nell'articolo "A Theory of Human Motivation".

parametri: funzionalità, affidabilità, usabilità, competenza e creatività, come nell'immagine che segue (Figura 1).

Analizziamo quindi i livelli della gerarchia: alla base della piramide troviamo il parametro funzionalità (*functionality)*. Il presupposto base è che, in prima istanza, il progetto funzioni e rispetti gli obiettivi prestabiliti.

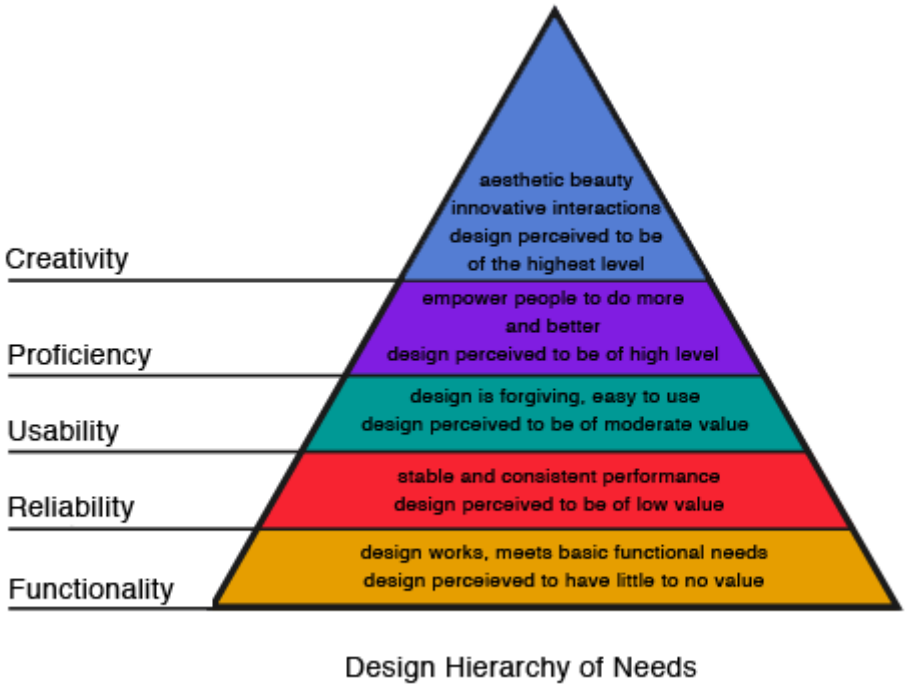


Figura 1 – Gerarchia dei bisogni di Maslow (adattata al campo del design)

In questo caso, il software di visualizzazione del *Vercelli Book* deve essere in grado di visualizzare, cercare e leggere le immagini ed i testi del manoscritto. Se non è possibile eseguire queste funzioni, allora la progettazione non è andata a buon fine.

Il secondo gradino della gerarchia vede il parametro affidabilità (*reliability);* l'affidabilità permette che le prestazioni risultino stabili e costanti. Il *Vercelli Book* deve funzionare ogni volta che il software viene lanciato.

Nel cuore della piramide troviamo uno dei parametri essenziali: l'usabilità. Il progetto sarà utilizzabile se ha una navigazione e organizzazione facile da capire e da usare. La competenza *(proficiency)* è un parametro auspicabile, ma non va considerato un imperativo categorico anche se significherebbe migliorare

notevolmente il progetto in questione. Va quindi considerata come una funzione di alto livello.

Integrare nel modo più efficace funzioni non disponibili in precedenza è una vera sfida per il programmatore. Oggi la progettazione deve inevitabilmente pensare all'interazione con gli utenti e sviluppare questo aspetto in modo innovativo. Oltre all'interazione, un progetto creativo dotrebbe ricorrere anche a un'architettura grafica interessante.

CAPITOLO II

PROGETTARE LA GUI

2.1 GUI per le edizioni digitali

Una volta ottenuto il materiale in forma digitale[17], una volta stabilite le misure da adottare per costruire un software per le edizioni digitali e tracciate le prime linee guida del suo sviluppo, il passo successivo è l'implementazione del modello GUI per una buona riuscita del progetto.

Il modello che andremo a creare deve essere preceduto da due definizioni di base:

- Definizione del target di riferimento (end users).
- Distribuzione finale.

Prima di iniziare con la progettazione è necessario definire il target a cui è destinata l'edizione digitale e come questo pubblico possa essere raggiunto.

2.1.1 Definizione del target di riferimento

La mancata corrispondenza tra il modello utente e il modello di implementazione fa sorgere inevitabilmente problemi di usabilità. L'edizione digitale del *Vercelli Book* è rivolta a tre fasce di pubblico: in primo luogo agli studiosi del settore, in secondo luogo agli studenti e in ultima istanza al pubblico interessato. È molto difficile riuscire ad armonizzare queste tre fasce di utenza: collezioni create per un pubblico di livello accademico, per esempio, possono avere un basso livello di contestualizzazione e interfacce di ricerca più sofisticate rispetto a quelle sviluppate per un pubblico con una formazione da scuola superiore.

Abbiamo deciso di creare un prodotto che fosse semplice da utilizzare, sia per quanto riguarda la presentazione dei contenuti, sia per quanto riguarda la navigazione stessa. Le collezioni presenti sul Web trovano inevitabilmente un pubblico più ampio rispetto all'idea con cui il progetto è stato inizialmente concepito, pertanto la

[17] Si veda il Capitolo I.

pianificazione del progetto dovrebbe tener conto della reale possibilità di una divulgazione al pubblico.

2.1.2 Distribuzione: il vantaggio di un applicazione Web-based

L'inizio di questo percorso di tesi ha visto come prima ipotesi l'intenzione di creare un software da divulgare tramite DVD o CD-ROM per la consultazione del manoscritto digitale, ipotesi successivamente scartata a favore di un'edizione per il Web 2.0.

Decidere quale dei due percorsi sarebbe stato il migliore da seguire è stato un punto di cruciale importanza: da questa scelta dipendeva la creazione dell'interfaccia finale.

L'idea di una applicazione Web-based è stata la scelta vincente: permette infatti di sfruttare appieno le potenzialità offerte dal browser.

Stabilita l'idea di creare prodotto accessibile a browser Web e dopo un'attenta lettura e consultazione della tesi realizzata da Francesca Fiorentini[18'], artefice della versione 1.0 del software di consultazione del *Vercelli Book*, si è quindi pensato di continuare verso la strada di implementazione di un software arricchendo quello precedente di nuove funzioni.

Si sono tuttavia riscontrate alcune perplessità sul proseguimento del lavoro precedente; il sofware da lei implementato con Ext-JS (framework di cui parleremo dettagliato nel Capitolo III) sebbene sia uno dei prodotti leader nel settore dei framework JavaScript, consta di *toolkit* lontani dalle esigenze di implementazione del *Vercelli Book*. In seconda istanza, le perplessità riguardavano l'implementazione e la gestione del codice.

L'obiettivo successivo è stato la definizione del linguaggio di programmazione da usare. Dal momento che lo scopo finale era quello di creare un software flessibile che potesse soddisfare tutti i requisiti di un'edizione digitale, un'ipotesi che è stata presa in considerazione prima di ricorrere a una programmazione in JavaScript è stata quella di ricorrere all'uso del PHP.

---

[18] EVT: Edition Visualization Technology; Progettazione e sviluppo un software per la consultazione di edizioni digitali"
http://etd.adm.unipi.it/theses/available/etd-09132008-133733/unrestricted/Tesi_Fiorentini.pdf

Questa soluzione, tuttavia, non era vicina all'utilizzo del prodotto finale, o meglio, si tratterebbe di uno step di implementazione successivo. Il punto di forza di PHP risiede nella sua diretta corrispondenza con un database. Sicuramente, avere a disposizione un database che raccoglie tutte le immagini e testi del manoscritto si rivelerebbe una fonte sicura di ausilio per il programmatore e la gestione dei dati.
Il linguaggio PHP, sebbene sia un linguaggio eccellente per gestione dei file, rivelatosi anche un buon linguaggio per la gestione delle immagini come nel progetto *Imap Viewer*[19] e nel campo dell'archiviazione dei manoscritti come dimostrato dal progetto *Gallica*[20] è un prodotto attualmente lontano dalle esigenze del *Vercelli Book.*

2.1.3 Una GUI che guarda al NUI

Gli sviluppatori di Web User Interface (WUID) dovrebbero seguire i sei principi[21] per la progettazione di RIA al fine di progettare un sistema facile da imparare per l'utente. La dimestichezza nell'uso di queste tecnologie ha portato a coniare i concetti di NUI (Natural User Interface) e TUI (Tangible User Inteface).
Questo tipo di interfacce, come suggeriscono i nomi, ci permettono di interagire con il software nel modo più naturale possibile. Il concetto di NUI si basa sul fatto che un utente può rapidamente imparare l'uso del software attraverso un sistema che dà la sensazione di agire immediatamente e continuamente con successo, quindi l'utente passa rapidamente dal primo utilizzo come "principiante" ad un stato di "esperto". Questo concetto è ora supportato da dispositivi informatici basati su *touch screen* e sensori di movimento, prossimità, etc.; questi dispositivi consentono di fare movimenti naturali o gesti che ci permettono di controllare e manipolare i dati in un software.
EVT è uno strumento che si pone al confine tra queste due strutture: GUI e NUI/TUI. Anche se (al momento) non è abilitato per riconoscere i sensori di movimento,

[19] The Image Markup and Presentation tool (ImaP)
http://lettuce.tapor.uvic.ca/~taprimap/imapdemo/demo/.
[20] Pagina ufficiale del sito Gallica: http://gallica.bnf.fr/.
[21] I sei principi per la progettazione delle RIA sono state proposti da Bill Scott e Theresa Neil in *Designing Web Interfaces* (2009). Essi sono: “make it directive, keep it lightweight, stay on the page, provide invitations, use transitions, react immediately”.

caratteristica tipica di interfacce NUI, tuttavia, vuole proporsi come una nuova edizione innovativa: è semplice, coerente e (speriamo) piacevole da usare.
Consideriamo adesso questa situazione: abbiamo una serie di prodotti da scegliere e da aggiungere ad una lista inizialmente vuota. Cosa avremmo fatto in una situazione reale? Avremmo preso i prodotti fisici e li avremmo spostati sulla nostra lista. Perché quindi non portare l'idea anche in interfacce virtuali? I comandi *drag-n-drop* sono un buon esempio di interazione naturale, simile alle nostre azioni nel mondo reale.
Probabilmente in futuro la maggior parte delle innovazioni sull'interfaccia utente verrà dal Web. Ecco perché un WUID deve seguire il rapido cambiamento di questi strumenti.

2.2 Confronto con altre edizioni digitali

Per il progetto mi sono basata sull'osservazione di software esistenti ed edizioni digitali già pubblicate, in particolare, dal momento che il campo della letteratura del settore è pieno di esempi, mi sono concentrata sulle edizioni digitali dei manoscritti.
Questa analisi mi ha portato ad individuare come tipologia di interfaccia più comune quella che utilizza strutture come frame e tabelle, mentre la visualizzazione dei contenuti generalmente si articola su vari livelli tramite finestre pop-up che occupano determinate aree dello spazio di lavoro.
Di seguito vengono proposti diciannove esempi di edizioni digitali significative per lo studio dell'interfaccia utente.

1. *Heliand Project*[22]. Il progetto Heliand è un caso interessante per la gestione di finestre multiple.

[22]HomePage del progetto Heliand: http://venus.unive.it/mbuzzoni/heliand.html.

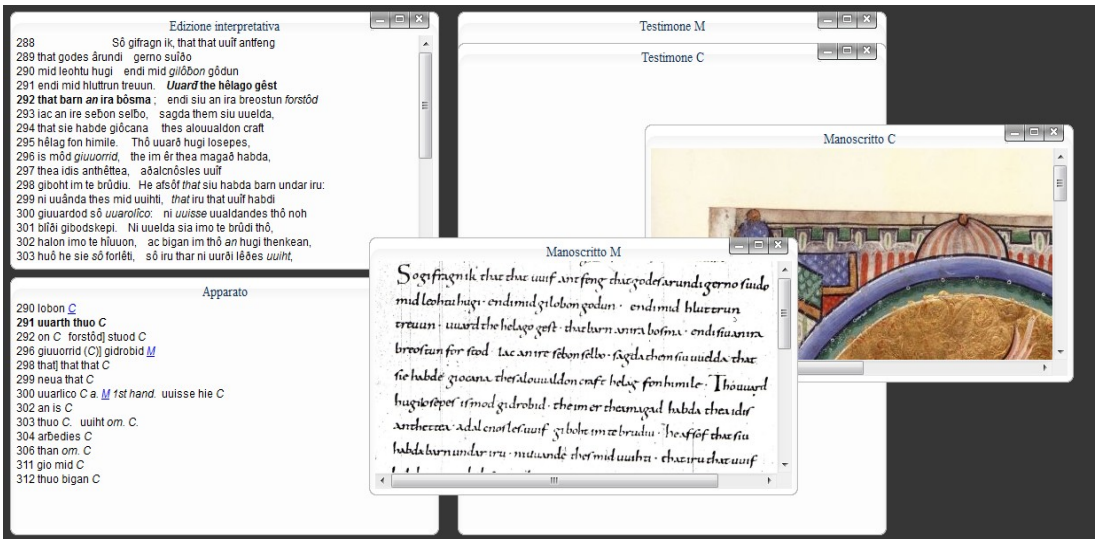


Figura 2 – Il Progetto Heliand

Una celebre dichiarazione di Bruce Tognazzini, consulente dell'usabilità e collaboratore di Donald Norman e Jakob Nielsen nella *Nielsen Norman Group*, ben identifica i pericoli derivati da questo approccio:

> "Give users some breathing room. Users learn quickly and gain a fast sense of mastery when they are placed "in charge." Paradoxically, however, people do not feel free in the absence of all boundaries. A little child will cry equally when held too tight or left to wander in a large and empty warehouse. Adults, too, feel most comfortable in an environment that is neither confining nor infinite, an environment explorable, but not hazardous.[23]"

La prima impressione che si può avere dalla consultazione dell'*Heliand* è quella di una sensazione di smarrimento. Gli utenti di oggi vogliono interagire con i programmi e gli strumenti nel modo più flessibile e libero possibile. Paradossalmente però, le persone non si sentono libere in assenza di tutti i confini. Limitare la capacità dell'utente è sicuramente una scelta restrittiva, ma anche lasciare una libertà totale è

[23]Tognazzini, B. *First Principles of Interaction Design,* (1980) Si veda il sito: http://www.asktog.com/basics/firstPrinciples.html.
La versione italiana del testo è disponibile in formato elettronico a questa URL: http://www.10people.net/tutorial/interaction_design-ask_tog/pricipi_di_interaction_design.html

sbagliato: ciò genera un sentimento di disorientamento. Oltre al dinamismo eccessivo delle finestre, la perplessità che può destare nell'utente il progetto *Heliand* risiede nella sovrapposizione delle finestre stesse. Dal momento che dispone di molte finestre, e tante funzioni, l'utente non ha più la conoscenza della quantità di finestre effettivamente aperte e che cosa può effettivamente fare con questo strumento. Questo approccio è una vera "arma a doppio taglio" per l'utente.
Anche se rimango sostanzialmente incline all'uso finestre multiple dinamiche, questo strumento deve essere utilizzato con saggezza, altrimenti si può innescare l'effetto opposto. Possiamo dunque aggiungere altre considerazioni al caso *Heliand*:

- è molto facile nascondere le finestre le une sotto le altre (soprattutto dopo averne aperte un certo numero);
- se le finestre iniziali sono piccole, la tentazione immediata è quella di ingrandirle: basta ingrandirne una al 100% dello spazio disponibile per nascondere tutte le altre;
- una finestra potrebbe essere nascosta spostandola quasi tutta al di fuori dell'area visibile nel navigatore;
- la gestione manuale può risultare onerosa per l'utente (ad esempio: se voglio confrontare due testi usando tutto lo spazio disponibile devo spostare le due finestre in modo da affiancarle occupando ciascuna metà dello spazio disponibile, poi devo estenderle verticalmente e "aggiustarle");
- è difficile effettuare una politica sensata di collocamento delle finestre (es. le immagini tendenzialmente a sinistra, i testi a destra) se gli strumenti di navigazione non aiutano in questo senso (ad esempio aprendo la finestra nel mezzo dello schermo); guardando alle varie edizioni che mi sono procurata, nella grande maggioranza le immagini stanno a sinistra e i testi a destra: ovviamente non dev'essere una distinzione così rigida, perché altrimenti non sarebbe possibile confrontare due immagini, ad esempio; diciamo che il "processo standard" per l'apertura delle finestre vede le immagini collocate a sinistra.

Ciò che è necessario in un ambiente di questo tipo è una serie di funzioni relative alla gestione delle finestre, l'utente deve avere la possibilità di:

- elencare le finestre aperte e accedere a una di esse eventualmente non visibile, portandola in primo piano;
- riordinare automaticamente le finestre (a cascata, per area dello schermo, etc.);
- affiancare ed espandere automaticamente due finestre: questo potrebbe avvenire selezionando la finestra A, espandendola in modo che occupi tutto lo spazio verticale disponibile e metà di quello orizzontale;
- iconificare e ripristinare le finestre in maniera efficace (deve essere visibile un'etichetta per richiamare rapidamente la finestra);
- espandere verticalmente una finestra.

2. *E-codices*[24]. L'obiettivo di questo progetto è quello di rendere accessibili i manoscritti medievali conservati in Svizzera con la creazione di una biblioteca virtuale.

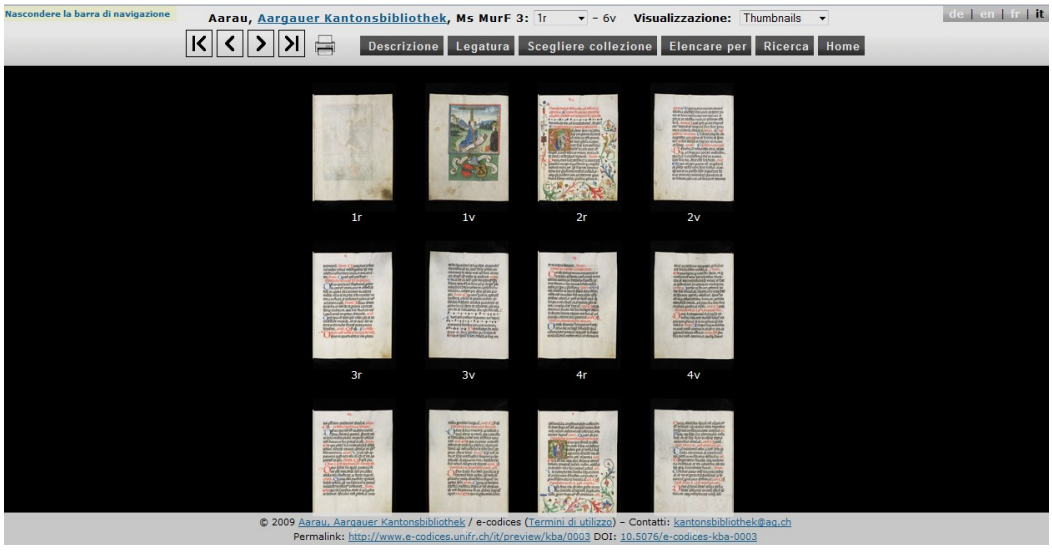


Figura 3 – E-Codices

[24] Sito ufficiale del *Virtual Manuscript Library of Switzerland :* http://www.e-codices.unifr.ch/it.

Lo strumento di navigazione è reso accessibile all'interno del programma come una scheda nella casella di testo. Si tratta di una struttura innovativa rispetto a molte altre edizioni reperibili sul Web. Inoltre, il passaggio del cursore del mouse sopra una miniatura visualizza una versione più grande della stessa, in modo analogo offerto da molte gallerie di foto reperibili sul Web.

3. *Codex Sinaiticus*[25]. Questo progetto consiste in una collaborazione internazionale che ha come obiettivo quello di riunire l'intero manoscritto in forma digitale e virtuale e a renderlo accessibile a un pubblico globale per la prima volta. Basandosi sulle competenze dei principali studiosi, conservatori e curatori, il progetto offre a tutti la possibilità di accedere direttamente al il famoso manoscritto.
Il *Codex Sinaiticus* è stato il mio primo approccio alle edizioni digitali. Quando ho iniziato a lavorare a questa tesi, sono rimasta affascinata da questo progetto. Eppure oggi, a mesi di distanza, noto elementi di visualizzazione discordanti per l'approccio ad un'edizione digitale. In primo luogo, la disposizione dei vari elementi non è chiara e ordinata. Sarebbe utile essere in grado di ridurre, iconificare e ripristinare le finestre, in modo da non ostacolare ulteriormente la vista dell'immagine, con la possibilità di recuperarle in un secondo momento, magari quando si stanno esaminando i dettagli di un'altra parte del manoscritto. Le finestre devono essere iconificate, raccolte e rese accessibili nella parte inferiore dello schermo (area designata a questo scopo).
Il *Codex* supporta anche contenuti multilingue, tra cui alfabeti speciali (greco, ebraico, cirillico) codificati tramite Unicode.

[25] Pagina di accesso all'edizione digitale del *Codex Sinaiticus:* http://codexsinaiticus.org/en/.

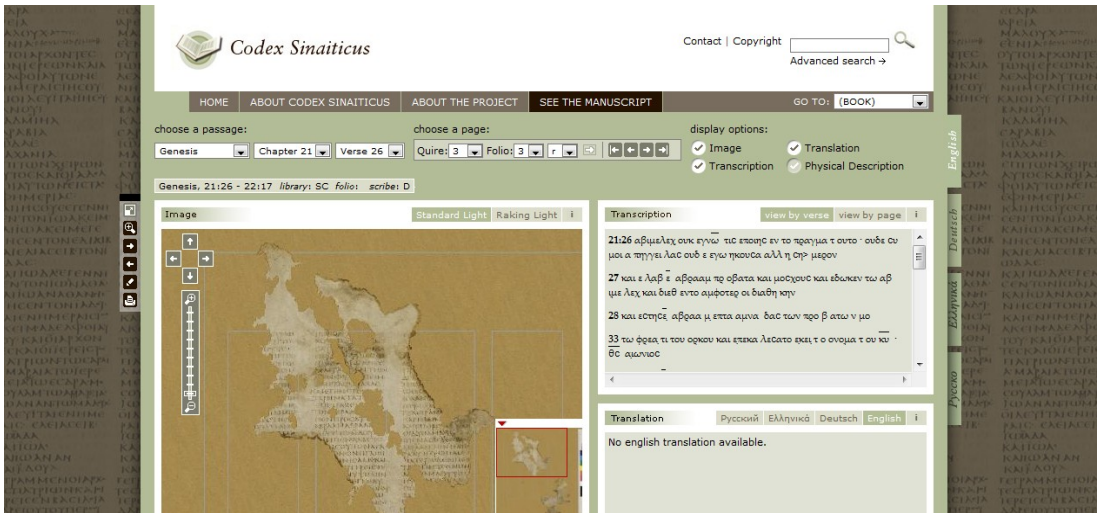


Figura 4 – The Codex Sinaiticus

4. *Electronic Beowulf*[26]. La prima versione dell' *Electronic Beowulf* risale al 1999 e fornisce immagini ad alta risoluzione del manoscritto. Le immagini sono state scattate ed elaborate sotto luce ultravioletta e in fibra ottica. L' *Electronic Beowulf* costituisce il prototipo del concetto di image-based nel campo delle edizioni digitali. Per quanto riguarda il testo, l'eBeo fornisce una trascrizione del testo poetico e strutture per la ricerca attraverso la codifica SGML.

La finestra principale dove si concentra l'attività dell'utente viene aperta partendo da valori predefiniti (i cosidetti *sane defaults*), ovvero una composizione di elementi della GUI disposti in maniera tale da non confondere l'utente. Essi permettono di visualizzare l'edizione in base agli interessi del singolo utente.

[26]Per l'accesso al progetto *Electronic Beowulf* si veda: http://ebeowulf.uky.edu/gettingstarted.

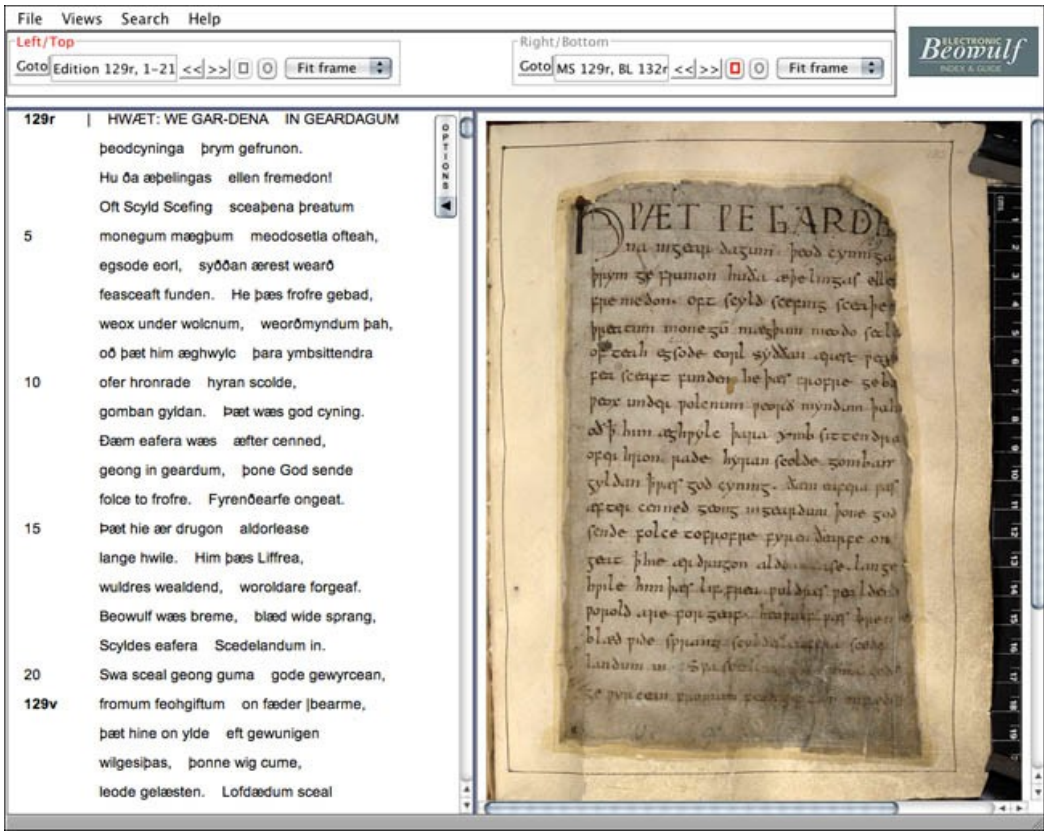


Figura 5 – eBeo

5. *London Lives*[27]. Questo progetto mette a disposizione, in una forma completamente digitalizzata e consultabile, una vasta gamma di fonti primarie della Londra del XVIII secolo. Questa risorsa include oltre 240.000 manoscritti e pagine che provengono da otto archivi di Londra. *London Lives* è uno strumento di ricerca interessante: difatti è possibile impostare una ricerca secondo le parole chiave, le persone e altri tipi di riferimento.

6. *The Shakespeare Quartos Archive*[28]. È una raccolta digitale delle commedie di William Shakespeare. Le funzioni includono la possibilità di sovrapporre immagini e testo, confrontare le immagini fianco a fianco, effettuare ricerche *full-text* o basate sul *mark-up,* e creare un set di annotazioni da rendere pubblico o lasciare privato.

[27] Progetto *London Lives*: http://www.londonlives.org/index.jsp.
[28] Collezione *The Quartos Archives:* http://www.quartos.org/index.html.

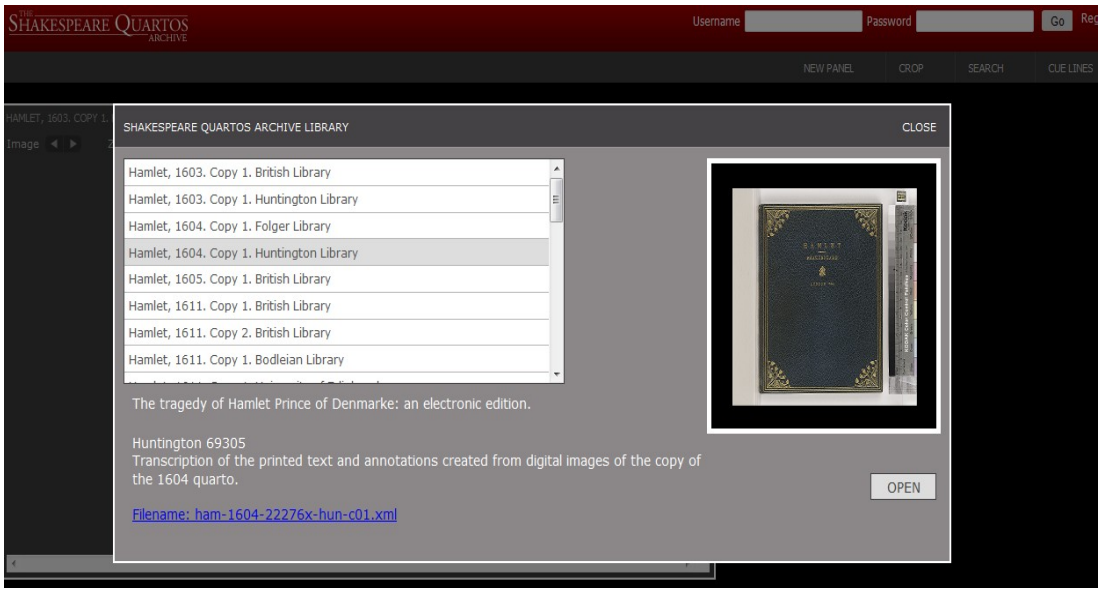


Figura 6 – The Shakespeare Quartos Archive

7. *Le Roman de la Rose*[29]. L'Università di Chicago ha realizzato la digitalizzazione di due popolari testi medievali: *Le Roman de la Rose* e *Le Jeu des échecs moralisé;* il primo è il romanzo cortese per eccellenza, il secondo un trattato sulla società medievale. Entrambi sono stati composti in Francia intorno al XII secolo. Accostando due testi diversi all'interno di un'unico progetto e sottoponendoli a codifica digitale, l'Università di Chicago[30] ha permesso lo studio simultaneo dei manoscritti divulgando la conoscenza della loro origine comune e la storia della loro produzione. Questo sito Web fornisce informazioni di base sui testi e i manoscritti di Chicago, fornendo l'accesso ai manoscritti completi.

In questo progetto, la barra degli strumenti di navigazione occupa l'intera parte superiore della finestra. Contiene i pulsanti di menù che permettono di accedere alle funzionalità del software e per navigare velocemente tra le varie parti di questa edizione e l'area centrale contiene un frame di navigazione per immagini. La visualizzazione è affidata a un plugin implementato con Flash.

[29] *Le roman de la Rose* in versione digitale: http://roseandchess.lib.uchicago.edu/index.html.
[30] Sito ufficiale dell'Università di Chicago: http://www.uchicago.edu/index.shtml

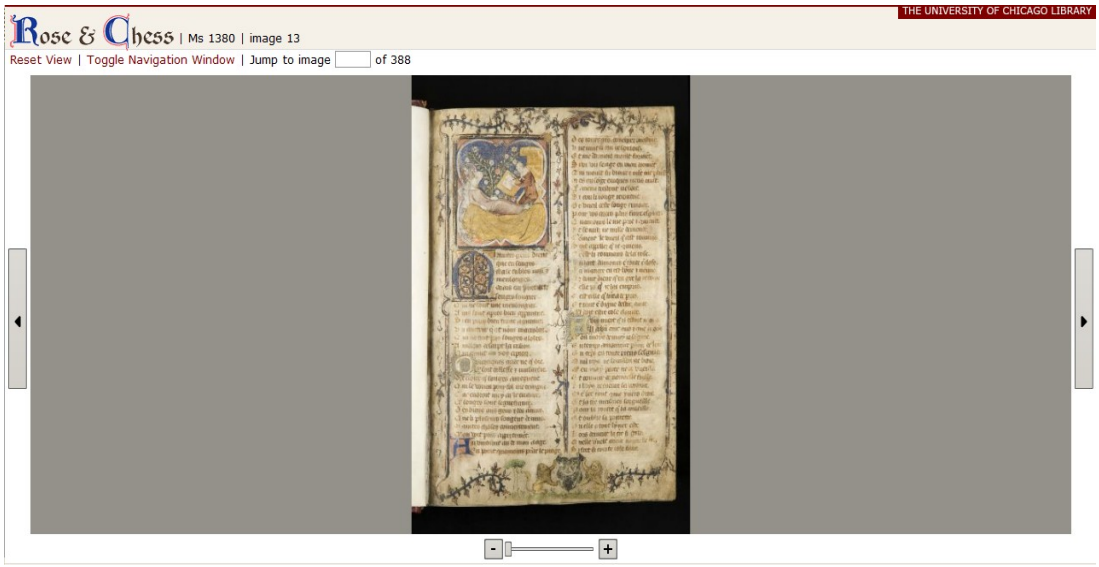


Figura 7 – Le Roman de la Rose

8. *Regnum Francorum Online*[31]. Questo è un sito contenente le mappe interattive del regno dei Franchi. Queste mappe tracciano le attività dei re merovingi e carolingi, come le donazioni della nobiltà e, lo sviluppo della proprietà di monasteri o altre istituzioni. I luoghi sulla mappa sono cliccabili e collegati a citazioni e riferimenti letterari: è sufficiente cliccare su una posizione e scoprire quali zone (riferite a una città in particolare) è possibile consultare. Vi è una panoramica delle mappe interattive nella sezione *Galleria*, intesa come punto di partenza se si è nuovi alla consultazione del sito. Questo progetto presenta:

- lo sviluppo di un sistema online di informazione geografica (GIS) per visualizzare e analizzare le informazioni relative agli eventi storici;
- l'implementazione di un'interfaccia che consente all'utente di visualizzare le mappe del regno franco: per esempio, la divisione del regno franco nel 768 d.C tra Carlo Magno e Carlomanno, figli di re Pipino il Breve;
- un database di città, istituzioni, persone e nomi personali con riferimento a dati spazio-temporali.

[31]Il progetto "Regnum Francorum Online" è accessibile a questa URL: http://www.francia.ahlfeldt.se/.

Le singole pagine sono realizzate con tecnica AJAX. È un'edizione interessante per sfogliare i contenuti in maniera dinamica. Uno dei punti di forza del progetto risiede proprio nell'interoperabilità attraverso:

- l'inclusione di pagine esterne realizzate in HTML, servlet, API, come ad esempio Google Maps o Google Books;
- l'inclusione di dati esterni in XML (ad esempio la *Tabula Peutinger*[32]);
- il collegamento diretto alle risorse quali pagine, immagini e voci di database;
- il download dei dati e l'elaborazione del testo.

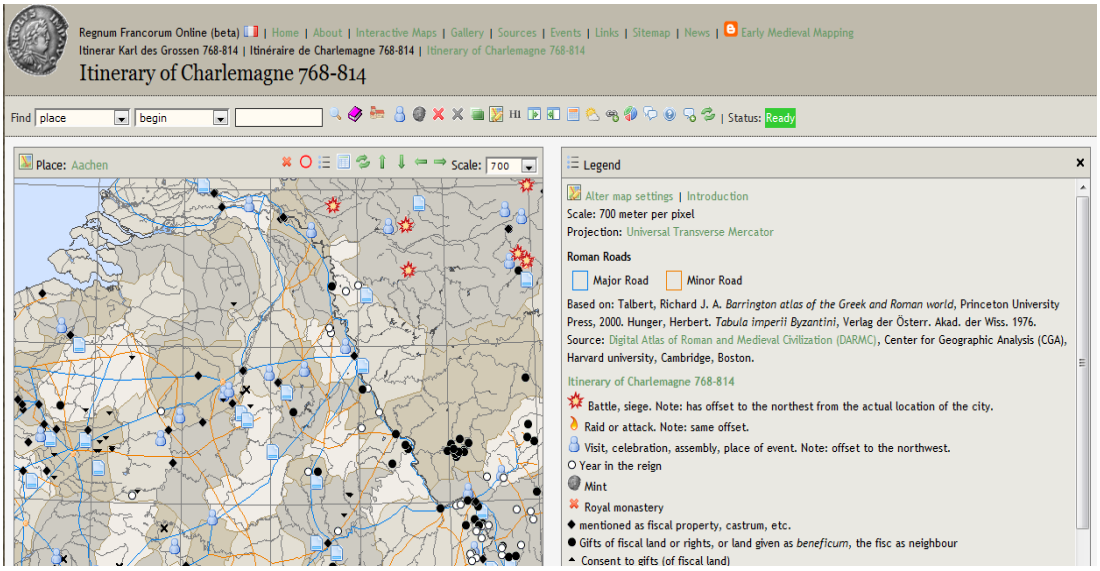


Figura 8 – Regnum Francorum Online (The GIS-application)

9. *The TEIViewer Project*[33]. Il Visualizzatore TEI è un progetto Web-based che fa uso di fogli di stile XSLT e della libreria jQuery. Il progetto non è stato ancora completato.

[32]La Tabula Peutingeriana è una copia medioevale di un'antica mappa romana, fonte inestimabile importanza per lo studio della topografia antica. Per un'ulteriore documentazione si consulti: http://www.unc.edu/awmc/index.html.
[33]*The TEIViewer Project* è attualmente reperibile a questa URL: http://teiviewer.org/.

10. *Evellum*[34] è uno dei produttori leader di facsimili digitali, edizioni e strumenti pedagogici. La società è stata fondata quasi quindici anni fa ed ha sede presso l'Università di Melbourne. *Evellum* si basa su un software proprietario. Al momento della stesura della tesi[35] le edizioni dei manoscritti anglosassoni pubblicati da *Evellum* sono:

- *The Exeter Book* – edizione digitale dell' *Exeter Dean & Chapter MS 3501.*
- *The Life of Saint Wilfrid* – Vita sancti Wilfridi Edmero auctore.
- *MS Junius 11*[36] – l'interfaccia grafica è presentata nella Figura 9.

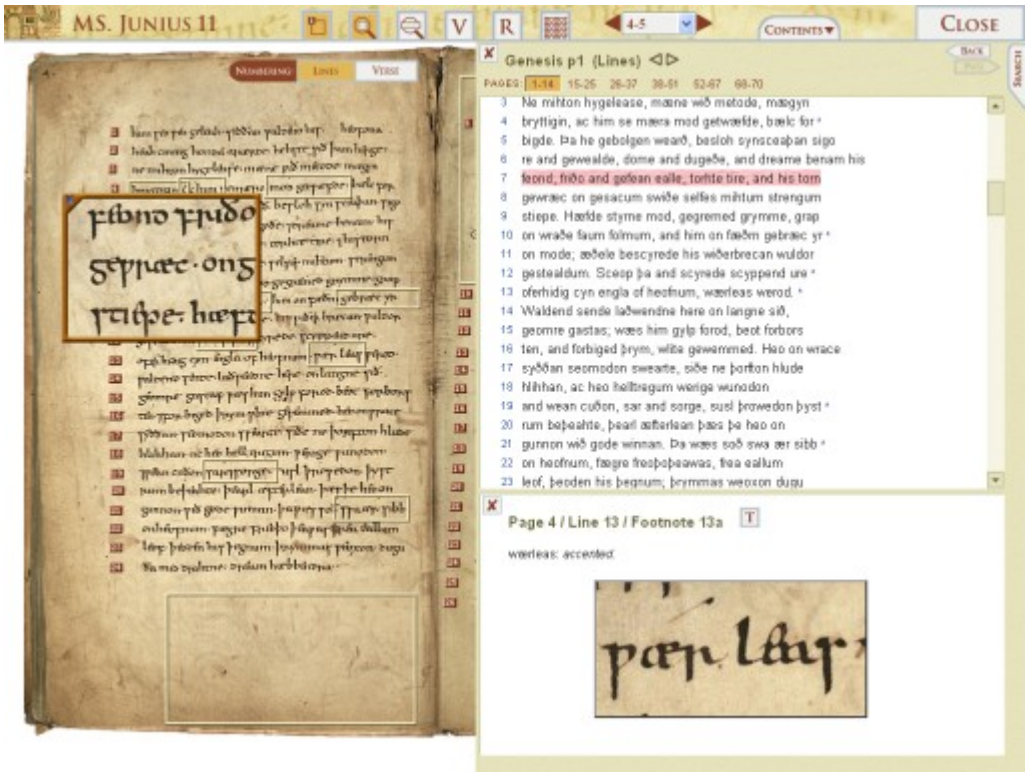


Figura 9 – Evellum (Junius MS)

[34] HomePage del progetto *Evellum*: http://www.evellum.com/,
[35] Agosto 2011.
[36] I l poema è attualmente conservato nella Libreria Vaticana (Cod. Pal.Lat 1447).

11. *Map of London*[37] è un progetto di Janelle Jenstad[38] sostenuto dal *Social Sciences and Humanities Research Council*[39] dell'Università di Londra.

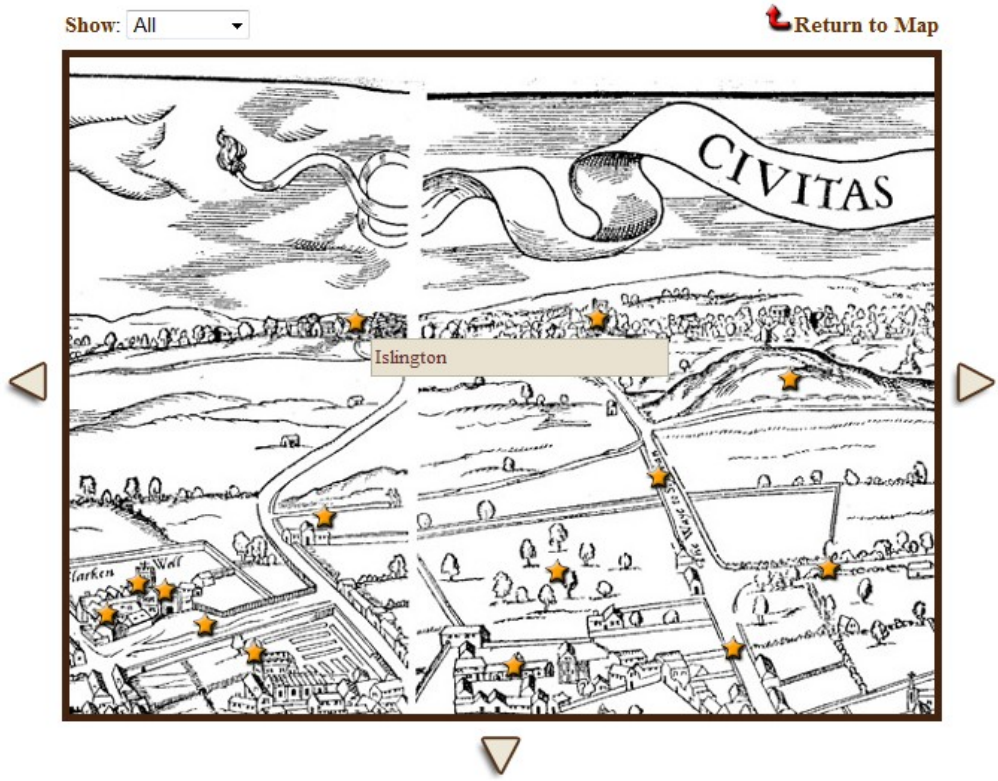


Figura 10 – Map of London

The Map of Early Modern London effettua il markup di strade, luoghi e confini significativi della Londra del tardo Cinquecento e degli inizi del XVII secolo. Qui, la funzione di zoom in/out per le immagini del manoscritto è stata implementata con due lenti separate, una per un'area quadrata e l'altra per tutta l'area dell'immagine: la prima viene utilizzata "in loco", la seconda presenta un ingrandimento in un layout separato. Il progetto fa uso del database eXist[40] XML per immagazzinare i documenti e codificarli nel formato TEI P5. Il sito funziona grazie a un software PHP v.5 che gira su

[37] *The Map of Early Modern London*: http://mapoflondon.uvic.ca/.
[38] Pagina di Janelle Jenstad: http://mapoflondon.uvic.ca/jenstad.php.
[39] *Social Sciences and Humanities Research Council:* http://www.sshrc-crsh.gc.ca/.
[40] eXist-db Open Source Native XML Database: http://exist.sourceforge.net/.

Apache e si collega al database eXist tramite XML-RPC (XML Remote Procedure Call).

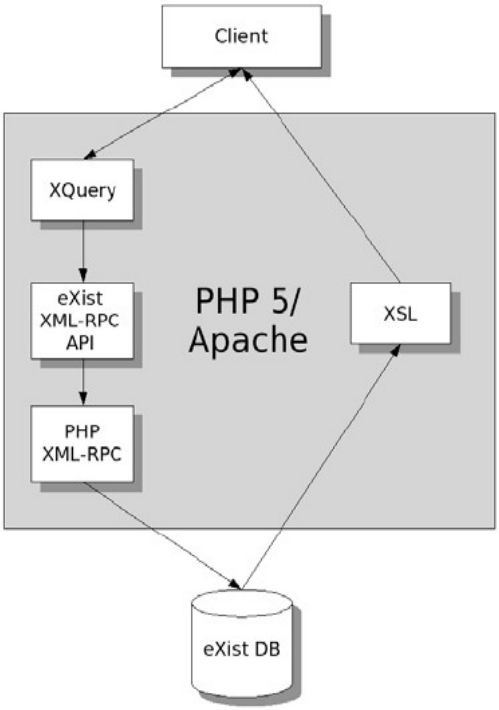


Figura 11 – Dataflow Diagram (Map of Early Modern London)

Il DOM[41] (Document Object Model), standard ufficiale del W3C[42] per la rappresentazione di documenti strutturati, è manipolato con JavaScript per creare nuovi tag che corrispondono al centro del rettangolo definito dal tag `<zone>`. I fogli di stile CSS definiscono gli stili che danno al tag un'immagine di sfondo per completare l'effetto. Il sito funziona meglio con Mozilla Firefox mentre Internet Explorer richiede l'inserimento di un plugin JavaScript per creare gli effetti di trasparenza. Il sito è conforme agli standard XHTML 1.1 e CSS 2.0. Lo schema di codifica è la TEI P5.

12. *Digital MappaeMundi*[43] (DM) è una risorsa che cambia radicalmente il modo in cui le mappe geografiche medievali e i testi possono essere modificati e studiati. Una

[41] Documentazione DOM: http://www.w3.org/DOM/
[42] Documentazione W3C: http://www.w3.org/
[43] Il progetto *Digital MappaeMundi*: http://bob.drew.edu/mappaemundi/.

prima fase del progetto risale al biennio 2006-2008. Oggi il lavoro del team[44] ha apportato delle migliorie a questa applicazione provvedendo all'implementazione di una nuova interfaccia grafica. Il progetto ha un obiettivo a lungo termine: fornire un'applicazione sia per tecnici e studiosi sia per i "non addetti ai lavori". Entrambi devono poter utilizzare e modificare i contenuti digitalizzati. È molto vicino all'idea del prototipo finale del visualizzatore del *Vercelli Book* soprattutto per la struttura duale nel rapporto immagine / testo. Il DM, esattamente come l'EVT del *Vercelli Book* prevede: una barra di navigazione, una toolbar, l'implementazione del contenuto all'interno dei tab.

Una delle cose che contraddistingue questa edizione digitale è l'utilizzo di un editor di testo. Affronteremo il problema dell'inserimento di un editor di testo WYSIWYG[45] trattando lo sviluppo dell'implementazione del software. Per adesso accenniamo al fatto che il prototipo finale del *Vercelli Book* farà uso di questo sistema di gestione del testo.

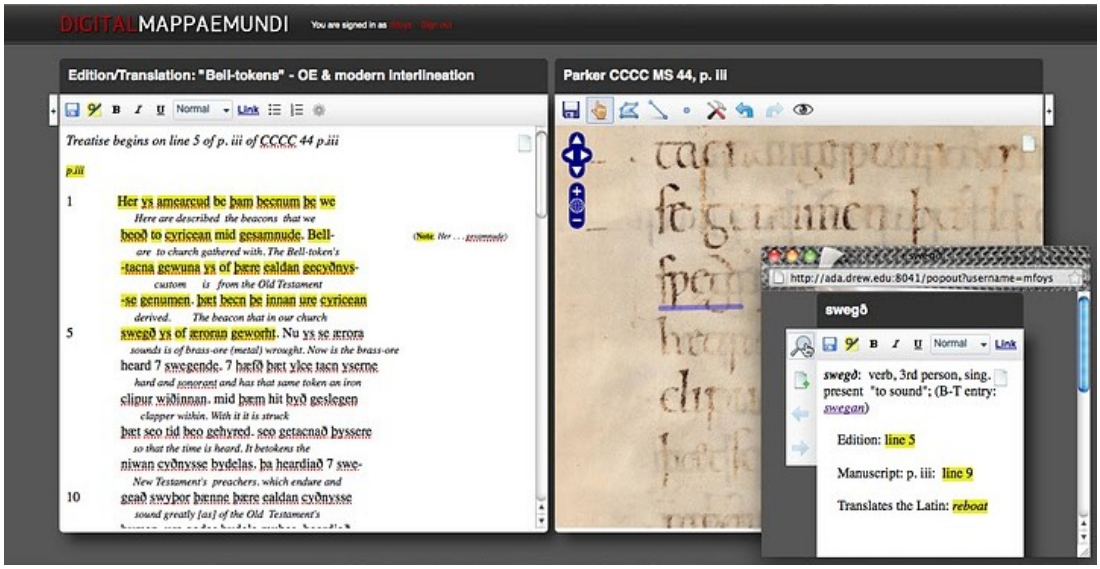


Figura 12 – Digital MappaeMundi

[44] Martin K. Foys, Associate Professor, English and Communication Arts (Hood College) and Visiting Associate Professor of English, 2008-2010 (Drew University) Asa Simon Mittman, Assistant Professor of Art History (California State University, Chico) Shannon Bradshaw, Associate Professor of Computer Science (Drew University).

[45] Si veda il Capitolo IV.

13. *Virtual Manuscript Room.* Il *Virtual Manuscript Room* è un progetto realizzato dall' Università di Birmingham[46] che presenta un'interessante caratteristica supplementare rispetto alle edizioni precedentemente analizzate: l'implementazione del *thumbnail navigator*. Esso occupa una fascia orizzontale nella parte inferiore dello schermo, collocata nel footer: visualizza un singolo set di miniature, ma si può continuare a navigare e visualizzare l'immagine selezionata al centro della finestra. È possibile accedere alle miniature delle immagini attraverso la barra delle finestre e i pulsanti di navigazione, inoltre tutti i documenti del *Virtual Manuscript Room* possono essere consultati dall'utente tramite un server OPAC.

Il punto debole del *VMR* risiede nella parte grafica. Com'è possibile vedere dalla Figura 13, l'accesso alla versione digitale del manoscritto presenta una grafica decisamente minimalista.

Figura 13 – VMR - Virtual Manuscript Room

14. *Text-Image Linking Environment* (TILE)[47] Il progetto TILE è uno strumento di gestione delle edizioni elettroniche basate su immagini e archivi digitali di testi umanistici. TILE 1.0 supporta i seguenti strumenti e funzionalità:

[46]Il progetto *Virtual Manuscript Room* è reperibile a questo indirizzo: http://www.birmingham.ac.uk/research/activity/itsee/projects/index.aspx.
[47]Link di riferimento al *Text-Image Linking Environment :* http://mith.umd.edu/tile/.

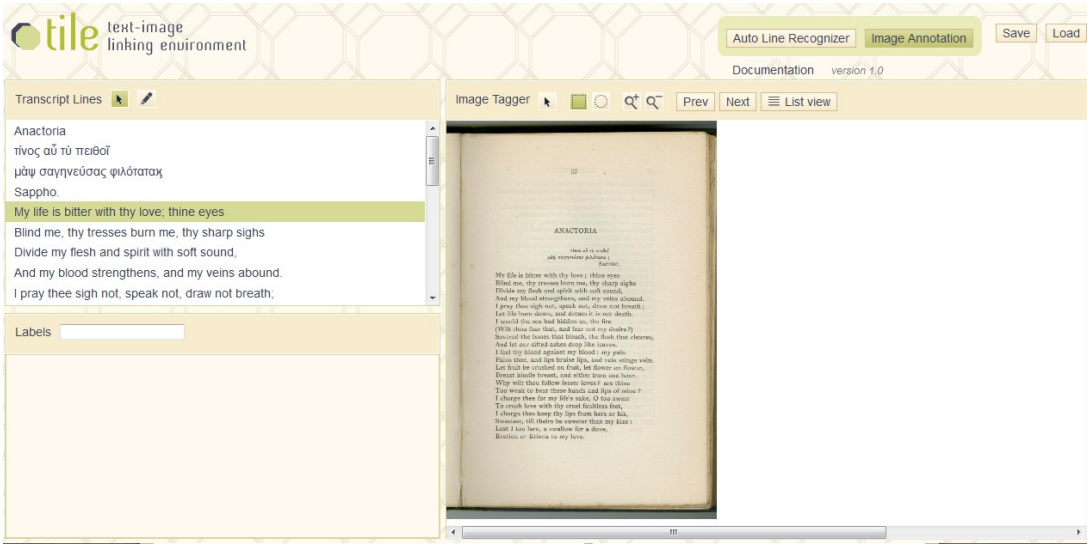


Figura 14 – (TILE) Text-Image Linking Environment

- *Image markup tool* – Annota le regioni di un'immagine attraverso rettangoli, poligoni ed ellissi, applica le etichette alle selezioni, e crea i collegamenti tra le aree di un'immagine e le righe della trascrizione.
- *Importing and exporting tools* – In TILE è incorporato uno script per l'importazione da diversi formati XML e l'esportazione dei dati nel formato TEI o JSON. Consente inoltre di salvare i file per l'output nei formati XML, HTML o testuale. Ulteriori strumenti di import / export possono essere sviluppati come plugin.
- *Semi-automated line recognizer* – Implementato in JavaScript, il "riconoscimento di linea semi-automatico" annota le immagini individuando le singole righe e selezionando le aree dell'immagine corrispondente.
- *Plugin architecture* – Estende le funzionalità di base di TILE con la creazione di un plugin che può manipolare l'interfaccia TILE, filtrare ed elaborare i dati, e connettersi ad altri strumenti.

TILE gestisce le visualizzazioni multiple della stessa immagine. Questa tecnica è particolarmente potente quando l'informazione è sufficientemente complessa da

richiedere diversi tipi di visualizzazioni per diversi aspetti. In questa sede non ci addentreremo ulteriormente sui meccanismi di interazione di TILE. Poniamo ora l'accento sul tipo di interfaccia grafica realizzata.

Com'è possibile constatare dalla Figura 14, la GUI di TILE suddivide lo spazio in due zone, spartite nella zona sinistra dedicata al testo e quella destra dedicata alla visualizzazione dell'immagine.

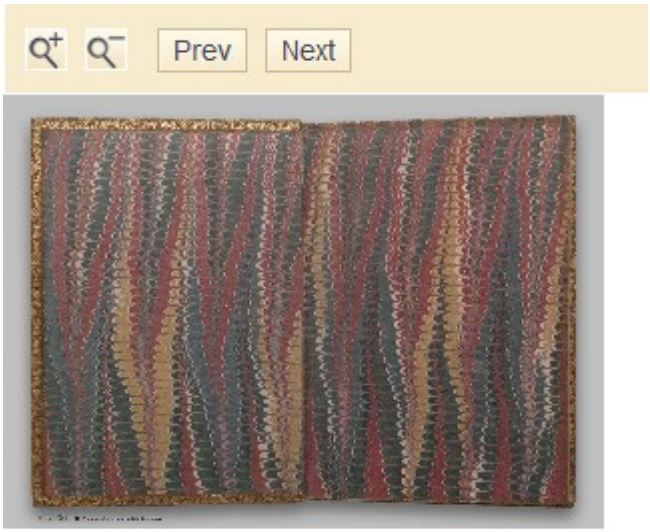


Figura 15– (TILE) Page Turner

TILE fa uso di uno script di zoom chiamato Page Turner[48] (Figura 15). Sebbene si tratti di uno script scaricabile gratuitamente e leggero in termini di Kb, presenta delle lacune: innanzitutto lo spazio dello zoom viene ingrandito facendo un uso sproporzionato dello spazio non favorendo l'inserimento dello script in un tab. In secondo luogo non presenta un'immagine in miniatura che possa essere di ausilio all'utente per verificare lo spazio di navigazione dell'immagine. Infine, non è possibile spostarsi nell'area dell'immagine selezionata una volta effettuato l'ingrandimento sull'immagine.

[48]PageTurner in azione: http://mith.umd.edu/tile/PageTurner/.

15. *Internet Archive BookReader*[49]. Il *bookreader* è una nuova forma di lettura dei documenti in formato digitale. Questo concetto si traduce in uno script (di solito realizzato in JavaScript) chiamato "lettore di pagina" o "pageflip". Oggi possiamo trovare sul Web molti software dedicati alla lettura digitale di riviste, giornali e e documenti come Flipping Book[50], Megazine 3[51], MartView[52]. Queste applicazioni consentono di produrre dei visualizzatori dotati di notevoli effetti puntando sull'utilizzo del *pageflip*. Alcune di queste applicazioni supportano il touch-screen permettendo di creare ebook con effetto *page flip*. In altre parole, il *bookreader* permette la produzione di semplici libri multimediali e interattivi che possono essere sfogliati dagli utenti. Basta importare il formato in PDF, configurare l'aspetto e premere il pulsante “publica”. *Internet Archive BookReader* si inserisce in questo contesto, ma differisce sotto molti altri aspetti: rispetta gli standard e presenta un grado complesso di interazione con l'utente[53]. La versione digitale del *Vercelli Book* sarà integrata con un *pageflip.*

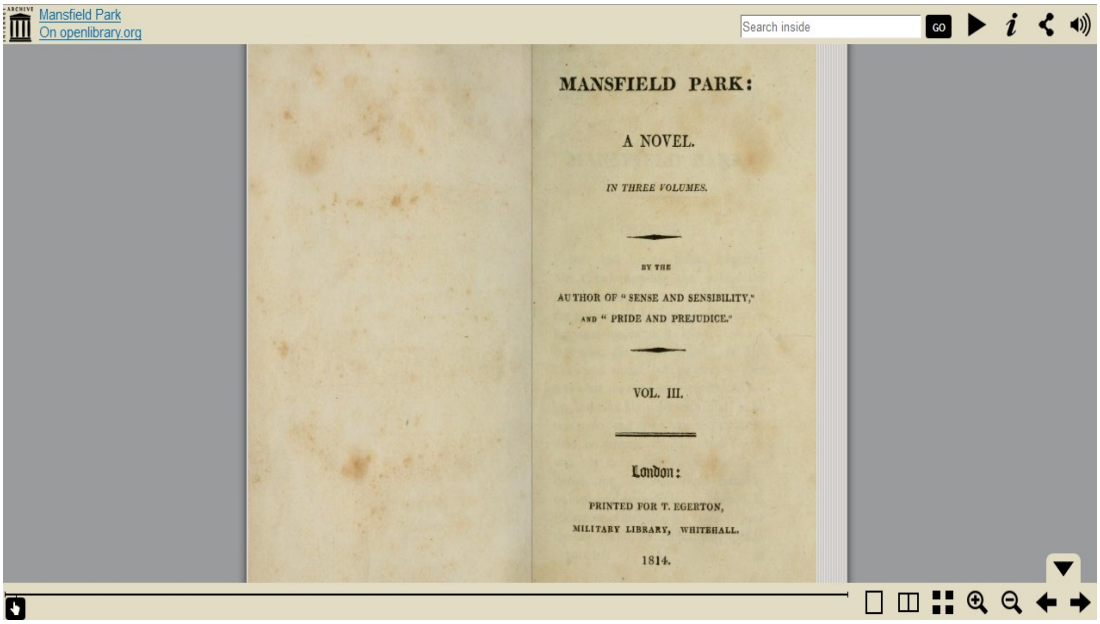


Figura 16 – Internet Archive BookReader

[49] Homepage del progetto *The Open Library*: http://openlibrary.org/dev/docs/bookreader.
[50] Page Flip: http://page-flip.com/.
[51] Megazine 3: http://www.megazine3.de/ .
[52] Mart View: http://www.martview.com/.
[53] BookReader User Interactions: http://openlibrary.org/dev/docs/bookreaderinteractions.

16. *ViHistory*[54] Il sito Web viHistory utilizza PHP e un database PostgreSQL per creare dinamicamente le pagine Web. PHP è linguaggio di scripting server che crea pagine Web su richiesta. PostgreSQL è un completo database relazionale ad oggetti rilasciato con una licenza BSD. Il sito è accessibile attraverso tutti i browser Web e si avvale delle tecnologie più recenti, come XHTML, XML e CSS.

I dati disponibili sul sito viHistory sono stati elaborati tramite SPSS, Paradox [per DOS], e Microsoft Access. Le informazioni in esso contenute sono memorizzate in un database PostgreSQL e in una collezione di documenti XML. Attualmente questo sito consta di circa 100.000 record. Allo scopo di rendere il sito accessibile a tutti i moderni browser sono stati sfruttati i linguaggi XHTML, XML, JavaScript e CSS. Il sito è compatibile con gli standard XHTML 1.1 e CSS Level 2: è stato testato con Firefox, Mozilla, Opera e Internet Explorer su Windows, e Firefox e Safari su Macintosh. Ci sono tuttavia alcune piccole differenze nella visualizzazione delle pagine con i vari browser.

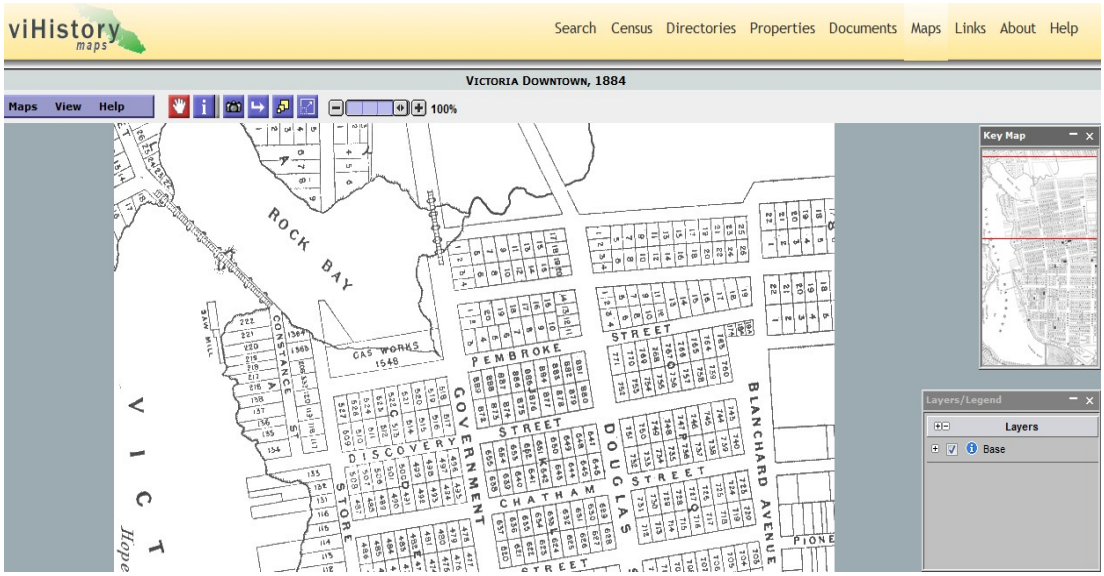


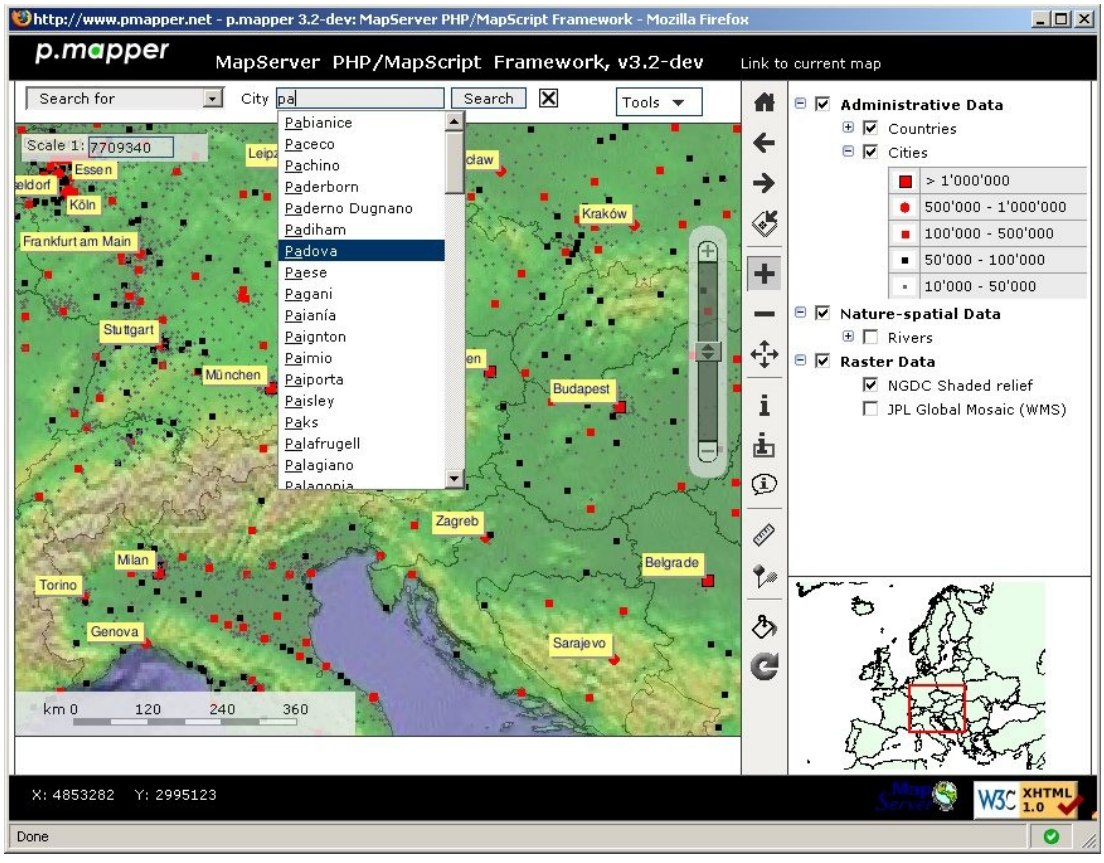
Figura 17 – ViHistory

[54]Mappa interattiva di *ViHistory*: http://vihistory.uvic.ca/content/maps/imapbasic.php?map=vicstreetcar1936.

17. *p.mapper*[55]. Nell'ambito di questa trattazione vale la pena ricordare anche il progetto *pmapper*, sebbene non si tratti di un lavoro nato in ambito accademico. *p.mapper* ha lo scopo di offrire un'ampia funzionalità e configurazioni multiple per facilitare l'installazione di una applicazione basata su MapServer PHP e MapScript.
Le funzioni incluse sono:

- DHTML (DOM) zoom / pan di interfaccia (senza utilizzo di frame);
- una configurazione molto flessibile di funzioni di query (identificazione, selezione e ricerca);
- un'interfaccia utente multilingue;
- la possibilità di aggiungere funzioni personalizzate tramite API.

Questo software delude in quanto a flessibilità. Non è possibile migliorare le funzioni come nel caso dello sviluppo di un'applicazione in jQuery ed è difficile capire come aggiungere funzioni. Altre applicazioni che hanno usato *p.mapper* hanno le stesse funzioni; il massimo livello di variazione si verifica nella leggera modifica del layout.



18. *Digital Library of the Caribbean*[56]. La dLOC fornisce agli utenti l'accesso a materiali storici e di ricerca tratti da archivi, biblioteche e collezioni private a scopo di ricerca. Il dLOC comprende collezioni che si estendono a giornali, archivi, documenti ufficiali, mappe storiche e contemporanee, storie orali e popolari, racconti di viaggio, letteratura e poesia, musica espressioni, e artefatti.

L'interfaccia utente si limita alle funzionalità base di un software per visualizzazione delle edizioni digitali: i pultanti successivo e precedente (*next* e *prev)* e un comando di selezione (*select)* che restituisce l'immagine della pagina desiderata del manoscritto. È assente qualunque tipo di zoom dell'immagine. Molto dettagliate sono invece le sezioni dedicate alla ricerca sul testo e alla gestione dei metadati.

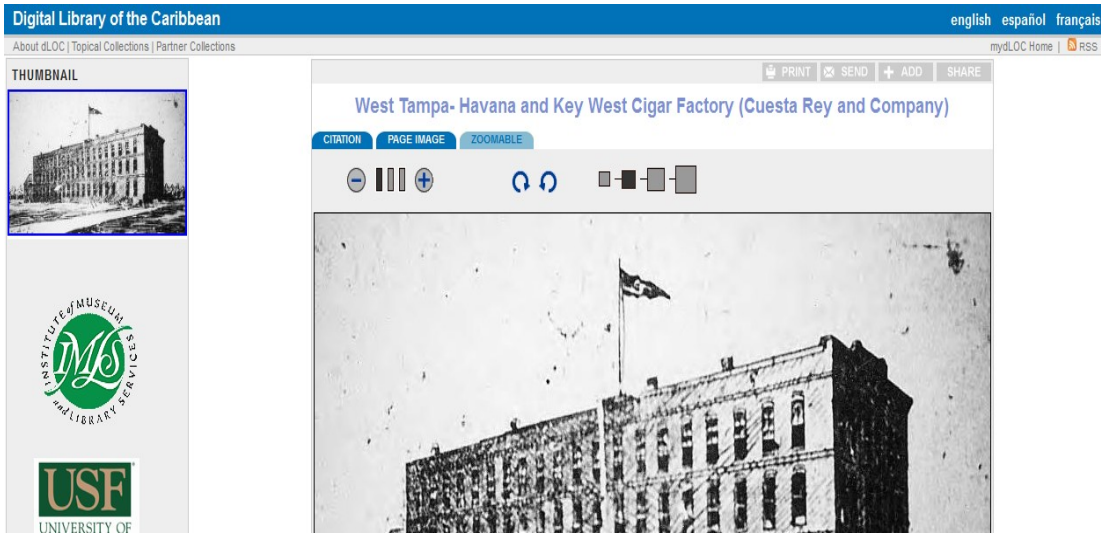


Figura 19 – Digital Library of the Caribbean

19. *Virtual Vellum*[57]. *Virtual Vellum* (VV) è un progetto e-Science finanziato da *EPSRC / JISC / Arts & Humanities e-Science Initiative*. È un programma con l'obiettivo di promuovere e dimostrare l'uso della tecnologia all'interno della ricerca umanistica. L'architettura del *Virtual Vellum* comprende le seguenti caratteristiche principali:

[56]*Digital Library of the Caribbean:* http://www.dloc.com/.

[57]Il progetto *Virtual Vellum*: http://www.shef.ac.uk/hri/projects/.../virtualvellum.htm.

- presenta *image panning* e zoom partendo da una scala dell' 1%;
- mette a disposizione una barra del righello orizzontale e verticale e una scala cromatica ridimensionabile in scala in base alle dimensioni del display;
- dispone di una finestra di anteprima che può essere attivata o disattivata a piacimento;
- presenta un'interfaccia completamente configurabile per la visualizzazione di documenti multipli.

Virtual Vellum è interamente scritto in JavaScript. La figura 20 presenta una finestra a sinistra che mostra il foglio per intero del MS Besançon 865. A destra il suo ingrandimento. Le immagini sono proprietà della "Bibliothèque Municipale di Besançon".

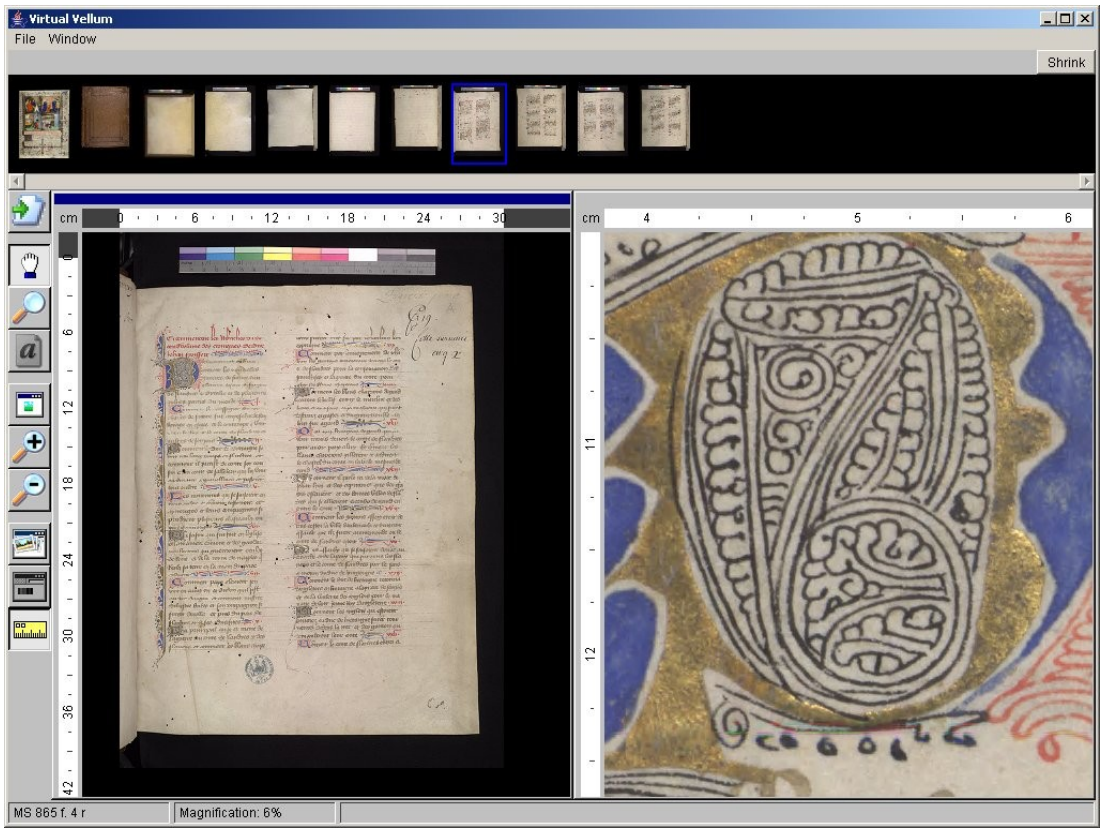


Figura 20 – Virtual Vellum

Ho consultato molti altri progetti[58]durante lo studio della GUI per la creazione del software di visualizzazione del *Vercelli Book*. È stata tuttavia effettuata una selezione e ho ritenuto opportuno presentare solo i progetti che presentano le caratteristiche più significative ai fini della progettazione dell'interfaccia del *Vercelli Book*.
La nascita delle prime edizioni di software digitali sul Web risale agli inizi degli anni '90: questo spiega l'alto numero di edizioni digitali disponibili sul Web. Solo poche di esse tuttavia, rispondono fedelmente agli standard sia per quanto riguarda il formato dei file (testi in TEI XML, XHTML, immagini in JPEG, TIFF), sia per quanto riguarda l'aderenza agli standard del web (AJAX), e l'uso di piattaforme Open Source diffuse e ben supportate (toolkit jQuery) per ridurre quanto più possibile problemi di manutenzione e conservazione delle edizioni digitali visualizzate con EVT.
Un'attenta analisi delle problematiche viste nell'analisi di queste edizioni ci permette di trovare i principali vantaggi di una edizione digitale:

- l'edizione digitale è disponibile in forma di ipertesto, in questo modo l'utente è in grado di personalizzare il proprio percorso di studio;
- l'edizione digitale diventa così uno strumento di lavoro e di studio che può raggiungere un livello di pubblico prima impensabile: non solo gli studiosi ma anche i semplici interessati.

Al contrario, le principali questioni globali sollevate da questi modelli riguardano diversi aspetti. Innanzitutto sussiste un problema generale come l'assenza dell'interfaccia a schermo intero. Un'edizione elettronica deve massimizzare lo spazio disponibile ed essere facilmente visualizzabile a schermo intero da un browser Web. Si riscontrano anche dei problemi attuali quali:

- la difficoltà di trovare dei layout che permettano una visualizzazione coordinata testo/immagine;

[58] Per un ulteriore documentazione si consulti il portale *Humanities Research Institute* a questa URL: http://www.shef.ac.uk/hri/projects/completepps.html.

- la mancanza di libertà da parte dell'utente di effettuare personalizzazioni sulla gestione del layout;
- una dipendenza nei confronti di hardware e software proprietari (come nel caso del progetto *Evellum*).

Sarebbe quindi necessario permettere all'utente:

- di gestire la dimensione del contenuto da visualizzare.
- di effettuare personalizzazioni sulla gestione del layout.

Tra gli strumenti da mettere a disposizione per l'utente è bene includere un "thumbnail navigator", al fine di rendere la navigazione ancora più intuitiva. Sarebbe anche preferibile utilizzare una piccola barra degli strumenti, non invadente, in modo da non eliminare la possibilità di visualizzazione del contenuto.
È quindi essenziale progettare delle interfacce "ad hoc" in quanto gli schemi tradizionali della progettazione grafica non sono in grado di soddisfare tutte le esigenze richieste per la realizzazione di un'interfaccia nel campo dell'EVT.
Vedremo in dettaglio il problema sul linguaggio di programmazione utilizzato per la costruzione del software nel Capitolo III. Per ora possiamo anticipare che jQuery è stato selezionato come linguaggio di scripting finale per la realizzazione dell'EVT del *Vercelli Book.*

2.3 Economia dello spazio: layout fluido e fisso

Accade spesso che le esigenze e le ambizioni del singolo progettista vadano a cozzare con le reali possibilità di realizzazione. In primo luogo, uno dei problemi con cui la nostra interfaccia grafica è andata a collidere è stata la modalità di gestione dello spazio in maniera efficace ed economica.
Le innovazioni tecnologiche nel campo dell'informatica determinano una continua evoluzione del mercato e la possibilità per i consumatori di disporre di computer e monitor di varie fasce di prezzo (netbook, pc, notebook, etc.) a costi inferiori rispetto

ad alcuni anni fa. Oggi i monitor 800x600 non sono più in commercio e la risoluzione minima presente sul mercato è 1024x768.
È stato quindi necessario porsi una domanda cruciale: conviene ancora basare il proprio progetto su un layout fisso?
Recenti statistiche[59] confermano l'utilizzo da parte dell'utente di risoluzioni elevate:

- le ricerche effettuate da w3Counter.com e w3Cschools.com rilevano che le risoluzioni 1280x800 o superiori sono utilizzati dal 50% degli utenti (dati aggiornati al mese di gennaio 2009); i dati mostrano che dal mese di gennaio 2007 al mese di gennaio 2008 c'è stato un incremento del 10% delle risoluzioni superiori a 1024x768;
- i risultati di una ricerca pubblicata da Net Applications[60] mostrano che, nel mese di dicembre 2008, 1024x768 sembra esserela risoluzione più diffusa, con il 37% degli utenti. Le risoluzioni 1280x960 e 1280x1024 sono utilizzate rispettivamente dal 20% e 13% degli utenti;
- alcuni importanti portali come Apple.com, Microsoft.com e YouTube.com hanno scelto un layout rigido ottimizzato per visualizzazioni 1024x768. Questa tendenza è confermata anche in Italia dove i principali quotidiani, La Repubblica.it ed il Corriere.it, hanno effettuato da più di un anno la stessa transizione.

I dati presentati, anche se provenienti da fonti differenti, confermano un indiscutibile cambiamento d'uso in merito alle risoluzioni utilizzate dagli utenti. In base ciò rispondiamo alla domanda che ci siamo posti all'inizio: nel costruire il layout di un progetto, scegliamo un layout fluido o fisso? In caso di layout fisso, a quale risoluzione facciamo riferimento?

Vantaggi nell'utilizzo di un layout fisso	*Svantaggi nell'utilizzo di un layout fisso*

[59] Si tratta di statistiche tratte dal w3Counter (http://www.w3counter.com/) e dal w3School (http://www.w3schools.com/).
[60] Homepage di Net Applications: http://www.netapplications.com/.

• è più facile da realizzare; • permette un maggior controllo del layout al variare delle risoluzioni e all'ingrandimento dei caratteri.	• a basse risoluzioni o a finestra ridotta è necessario utilizzare le barre di scorrimento orizzontale per raggiungere tutte le informazioni.

Tabella 1. Vantaggi e svantaggi di un layout fisso.

Vantaggi nell'utilizzo di un layout fluiso	*Svantaggi nell'utilizzo di un layout fluido*
• i contenuti si adattano alla risoluzione dell'utente, anche a basse risoluzioni o a finestra ridotta.	• ad alte risoluzioni i contenuti testuali potrebbero allungarsi eccessivamente in orizzontale e creare difficoltà di lettura; • a risoluzioni basse o a finestra ridotta potrebbero verificarsi sovrapposizioni di contenuti; • la gestione degli spazi e delle proporzioni grafiche è più complessa rispetto al layout rigido.

Tabella 2. Vantaggi e svantaggi di un layout fluido.

Alcuni sviluppatori preferiscono utilizzare delle soluzioni miste che prevedono, per esempio, una colonna fissa in pixel ed un corpo dei contenuti con larghezza massima e minima fissate. Il trend dei portali di larga diffusione sembra preferire un layout con una larghezza (width) non superiore ai 960 pixel, in modo da adattarsi alla risoluzione minima di 1024x768 px. Nel caso in cui ci si voglia orientare, invece, verso un layout fluido, si prevede l'inserimento di:

- una larghezza minima (min-width) di circa 700 pixel al contenitore del layout, per evitare la sovrappozione del contentuto a basse risoluzioni;

- una larghezza massima (max-width) di circa 1400 pixel per evitare che il testo del sito risulti eccessivamente esteso in orizzontale ad alte risoluzioni.

L'interfaccia utente del software di visualizzazione del *Vercelli Book* deve essere flessibile e in grado di regolare automaticamente diversi insiemi di contenuti. La chiave per realizzare un'interfaccia flessibile risiede nella creazione di un interfaccia basata su oggetti. Al centro dell'interfaccia basata su oggetti c'è una classe JavaScript chiamata BOX. Ogni scatola è un'istanza della classe BOX. Un BOX è un elemento dell'interfaccia utente che può contenere una caratteristica. Una caratteristica è tutto ciò che può andare bene all'interno di una scatola.
Ho cercato dei *tools* JavaScript predefiniti per realizzare il progetto dell'interfaccia utente, ma nessuno di loro si avvicinava all'idea del prodotto finito.
L'unica soluzione possibile è stata, una volta definitone l'aspetto genereale, quella di cominciare a implementare manualmente l'interfaccia GUI.

CAPITOLO III

L'ARCHITETTURA EVT

3.1 Il "Modello a Cascata"

Il *modello a cascata*[61] meglio conosciuto con il nome inglese di *waterfall model*, è una metodologia generica per il sistema di sviluppo di un sofware. È stata scelta come architettura di progettazione applicata al *Vercelli Book* perché copre l'intero ciclo di vita di un sistema.

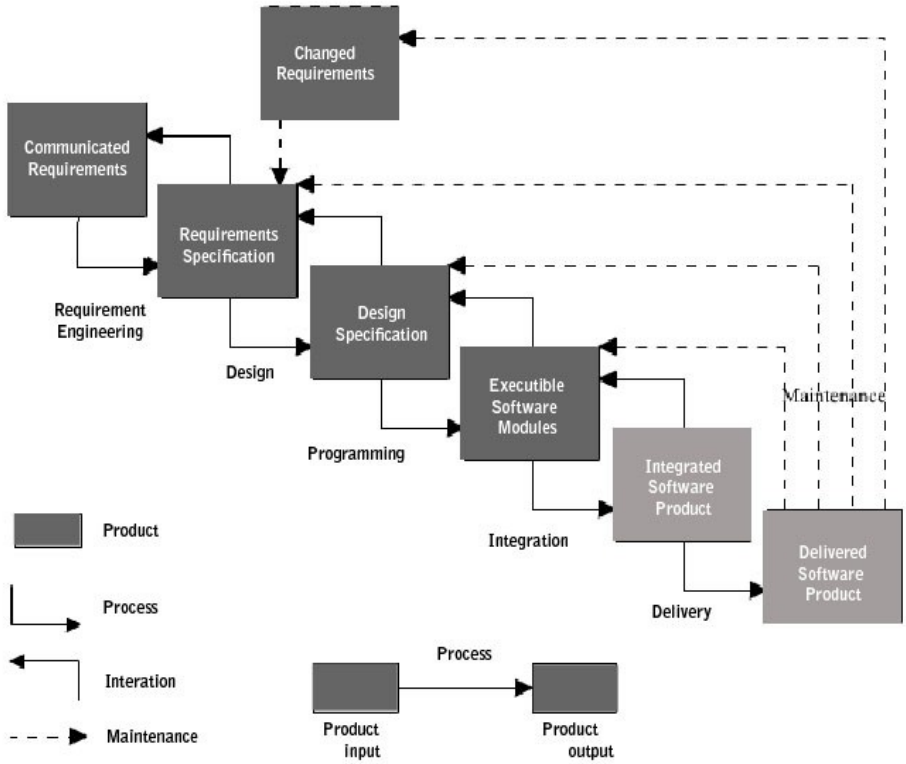


Figura 21 – Il modello a cascata (The Waterfall Model)

[61]Un'ulteriore documentazione su questo modello è disponibile a questo indirizzo: http://courses.cs.vt.edu/csonline/SE/Lessons/Waterfall/index.html.

Il modello a cascata è piuttosto semplice; in questo processo i passi sono sequenziali, ciò significa che si "nutrono" a vicenda con le informazioni. Di conseguenza, una fase non può iniziare prima che quella precedente sia finita. Tuttavia, l'esperienza mostra che il processo può essere molto ripetitivo e che le fasi possono sovrapporsi.
Lo studio del *modello a cascata* è importante perché è il modello più noto e perché l'analisi delle diverse fasi ci permette di descrivere le attività che fanno parte di tutti i modelli di processo consentendo una pianificazione dell'intero percorso di costruzione dell'EVT.
Il *modello a cascata* fornisce una sequenza di passaggi consecutivi. Ogni fase produce dei *deliverable*. I d*eliverable* sono documenti relativi al processo, alla documentazione del prodotto, al codice sorgente e vengono ulteriormente elaborati da fasi successive.
Il software EVT, seguendo l'architettura del modello a cascata, consta di una sequenza lineare di fasi di lavoro.
Il numero, il contenuto e la denominazione delle fasi ci lavoro può variare da un'organizzazione all'altra e da un progetto all'altro. Di solito il processo è diviso in cinque fasi come mostrato nella Figura 21:

1. Communicated requirements and requirements specification.
2. Design Specification.
3. Executable Software Modules.
4. (Integrated Software Product).
5. (Delivered Software Product).

Ogni fase ha un *deliverable* di input emerso dalla fase precedente e ogni fase produce uscite che sono l'ingresso della fase successiva. Così una prima fase inizia solo quando la precedente è stata completata. I passi sono fondamentali perché ogni passo corrisponde ad una fine. In questo modo sarà più semplice analizzare i processi dell'intero progetto.
Nel caso della produzione del software di visualizzazione del *Vercelli Book*, le ultime due tappe (3 e 4) non rientrano nell'ambito del progetto.

Adesso possiamo passare all'analisi dettagliata delle fasi che caratterizzano questa struttura:

1. *Communicated requirements and requirements specification*: in questa fase ciò che è importante è comprendere e descrivere in maniera più completa e accurata possibile ciò che il cliente vuole dal software (requisiti tecnici). La fonte principale dei requisiti sono i futuri utenti del software da sviluppare. Il metodo prevede il ricorso a interviste o presentazioni di prototipi esistenti. In questa fase dovranno essere definite:

- *le funzioni* - ciò che il software deve fare (funzionalità), le caratteristiche, come sarà utilizzato l'EVT etc.;
- *le caratteristiche generali* - affidabilità, usabilità, prestazioni, compatibilità, etc;
- *le tecnologie* - tecnologie da utilizzare per lo sviluppo, ad esempio, JavaScript, CMS, etc.

2. *Design Specification.* La progettazione del software deve produrre un sistema che soddisfi i requisiti degli utenti. L'architettura del sistema è la visione d'insieme di moduli software che eseguono le funzioni (attività/servizi) richieste dall'utente e comprende le relazioni tra i moduli e il loro uso. I moduli da realizzare sono visti come scatole nere, mattoni logici.

3. *Executable Software Modules*. La fase di codifica essere seguita da l'implementazione del software con la definizione del linguaggio di programmazione scelto.

4. (*Integrated Software Product).* In questa fase il software sviluppato è installato nell'ambiente di destinazione del cliente.

5. (*Delivered Software Product).* Il *modello a cascata* prevede, sin dalle prime fasi, una continua modifica del prodotto software. La manutenzione e gestione, prevedono la riparazione di difetti del prodotto consegnato al cliente e l'aggiornamento del codice, fornendo, a volte, una nuova versione.

Il *Waterfall Model* non è l'unico modello di architettura per il ciclo di vita di un software. Nel 1988, Barry Boehm ha proposto un modello più completo chiamato il *modello a spirale*[62]partendo dalle carenze del modello a cascata. Quindi, mentre il modello a cascata presenta una visione chiara del ciclo di vita del software, lo *Spiral Modle* è appropriato solo per alcune categorie di sviluppo del software. In particolare, il modello a cascata funziona bene quando i requisiti software sono ben noti e la natura dello sviluppo del software comporta accordi contrattuali. Il modello a cascata è una scelta naturale per lo sviluppo software basato su contratto dal momento che questo modello è guidato, cioè, molti dei prodotti come la specifica dei requisiti e la progettazione sono documentati. Questi documenti diventano la base per lo sviluppo del software.
Il Waterfall model è concettualmente meno articolato perchè permette di dividere il problema in fasi distinte che possono essere eseguite in modo indipendente. È anche più facile da controllare durante la costruzione del progetto.
In realtà, i progetti reali raramente seguono in modo rigoroso la sequenza proposta dal modello. Spesso infatti è necessario ripetere fasi o attività precedentemente svolte. E così è successo nella realizzazione del software di visualizzazione del *Vercelli Book*.

3.2 Fase I. Requisiti Tecnici

[62] Descrizion del *The Spiral Life Cycle Model*: http://www.freetutes.com/systemanalysis/sa2-spiral-model.html.

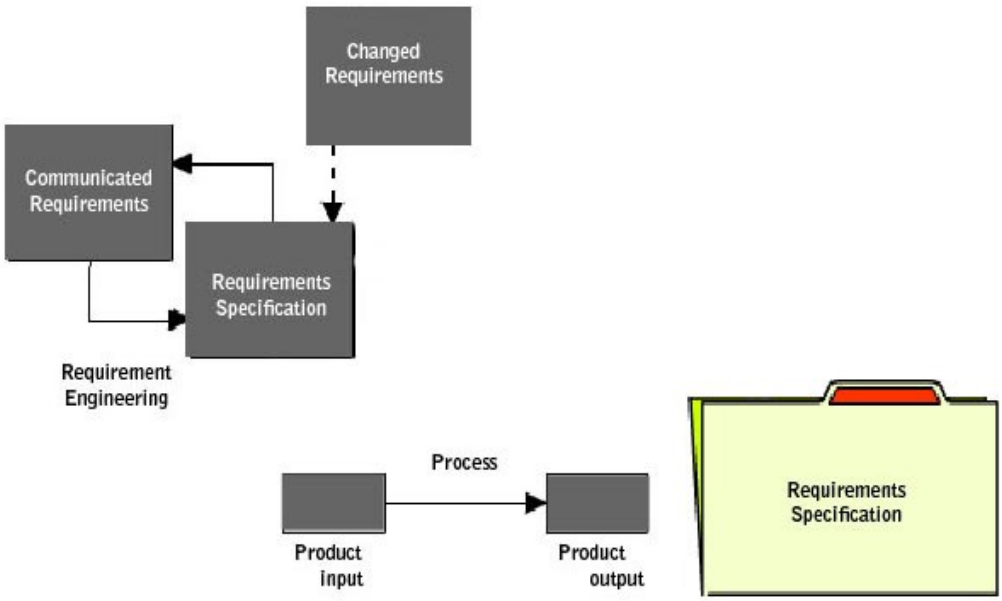


Figura 22 – Il modello a cascata (Fase 1)

Questi sono requisiti generali del programma per quanto riguarda il software, le caratteristiche, l'interfaccia utente grafica, il formato dei file da utilizzare:

1. deve trattarsi di un software multi piattaforma, in esecuzione con i sistemi operativi più diffusi (Windows, Unix / Linux, MacOS);
2. deve essere multilingua, consentire la localizzazione del programma (menu, pulsanti, etc.), l'inserimento e la gestione di contenuti multilingue;
3. deve essere sviluppato e distribuito come software Open Source, da condividere con la comunità accademica;
4. si deve evitare la richiesta di hardware o software speciali o particolarmente potenti, anche se la gestione di immagini ad alta risoluzione richiederà una velocità della CPU e una quantità di RAM ragionevole;
5. deve essere espandibile in modo da permettere alle versioni successive funzionalità in più e diverse;
6. l'interfaccia grafica utente deve essere il più intuitiva e meno vistosa possibile;

7. l'interfaccia grafica utente deve essere perfettamente integrata nel sistema operativo è in esecuzione, in conformità con il look and feel nativo;
8. il software deve essere in grado di gestire il formato di testo, HTML e XML. I file XML possono essere resi in HTML a condizione che questa operazione non influisca sulle performance del programma;
9. per quanto riguarda i documenti XML, il software deve essere abbastanza flessibile da consentire l'uso di qualsiasi tipo di DTD / schema, non necessariamente la TEI 1;
10. il programma deve essere in grado di integrare un motore di ricerca XML.

3.2.1 Funzionalità principali

Il set di base di funzionalità e caratteristiche che devono essere presenti nella versione 2.0 sono:

1. immagini e testo presenti in frame separati: solo immagini, solo testo, immagine e testo, immagine e immagine (rectoverso, hotspot, immagine del manoscritto con zoom), testo e testo (traduzione e vaire edizioni). Queste strutture devono essere facilmente ampliate, e trasferibili a icona;
2. la finestra che ospita il testo deve consentire la visualizzazione per diversi livelli di fruizione: l'utente deve poter scegliere tra un'edizione diplomatica o semi-diplomatica, o diverse modalità di visualizzazione per le caratteristiche testuali (abbreviazioni, ampliamenti, aggiunte, cancellazioni, le lacune, danni, ecc);
3. collegamento immagine/testo: cliccando su una parola del testo dell'edizione si passa alla corrispondente area dell'immagine del manoscritto;
4. zoom in / out per le immagini del manoscritto: questo può essere attuato in diversi modi. l'*Electronic Junius*[63] offre due lenti di ingrandimento separate, una per una superficie quadrata e l'altra per tutta la linea: quest'ultima appare ingrandita in un riquadro distinto;

[63] Electronic Junius Manuscript: http://omacl.org/Junius/.

5. l'uso di un righello e altri strumenti grafici per capire le dimensioni reali di un'immagine o di una parte dell'immagine;
6. un "navigatore in miniatura", ovvero un layout separato che contiene le immagini in miniatura di tutti i fogli del manoscritto in modo che l'utente possa scegliere direttamente l'immagine da aprire;
7. *hot-spot* che compongono i dettagli del manoscritto, attivabili dall'utente.
8. mostrare note, traduzioni e altri tipi di informazioni relative al testo e all'immagine;
9. perfetta integrazione del motore di ricerca XML, consentendo ricerche semplici e ricerche complesse sui testi che compongono l'edizione;
10. anche se uno degli obiettivi principali del progetto è quello di fornire un'interfaccia utente intuitiva, il software deve essere accompagnato da una adeguata documentazione elettronica.

Un elenco di funzionalità avanzate e di idee che potrebbero essere successivamente implementate potrebbero essere:

1. *collegamento avanzato testo-immagine*: fornendo una perfetta corrispondenza dovrebbero essere disponibili anche per le parole risultanti da una ricerca effettuata dall'utente; l'utente dovrebbe essere in grado di visualizzare sia la parola nel testo dell'edizione che nell'immagine manoscritto;
2. *strumenti di elaborazione delle immagini*: una serie di semplici filtri da applicare alle parti specifiche di un'immagine;
3. *stalvataggio delle impostazioni di filtro*: l'utente dovrebbe essere in grado di salvare il risultato del filtro applicato in modo da replicare esattamente lo stesso processo in un momento successivo;
4. *applicazione dei filtri ad aree definite*: i filtri dovrebbero essere applicati a tutta l'immagine, o ad una specifica zona selezionata dall'utente;
5. *segnalibri*: per accelerare la navigazione del testo e dell'immagine gli utenti dovrebbero essere in grado di piazzare dei segnalibri temporanei e selezionarli in modo da accedere rapidamente al punto indicato.

6. *note di testo (sticky notes):* gli utenti dovrebbero essere in grado di prendere appunti e annotare le loro osservazioni su parti specifiche della edizione (un dettaglio in un'immagine, una parola dell'edizione diplomatica, ecc). Lo sticky notes deve essere composto delle funzioni: mostra, nascondi, taglia e copia riferito al testo a cui stiamo lavorando.

3.3 Fase II. Framework JavaScript per le interfacce GUI

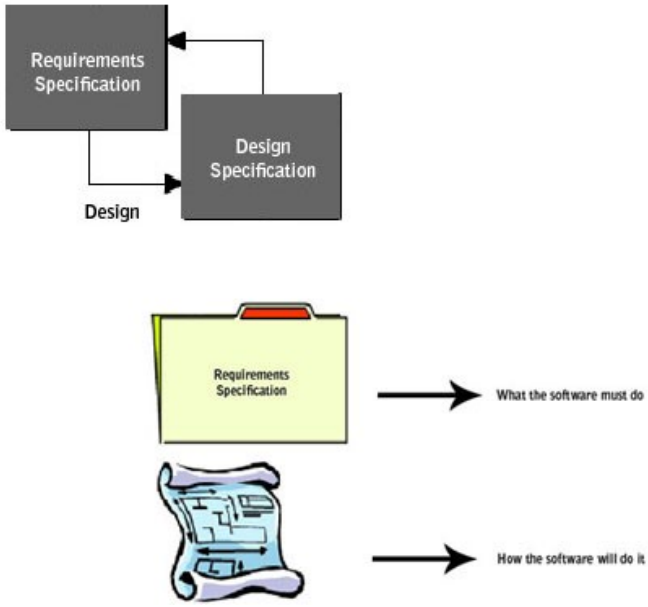


Figura 23 – Il modello a cascata (Fase 2)

JavaScript è senza dubbio uno dei linguaggi di programmazione più importanti nel campo dell'informatica moderna.

Naturalmente, non tutti la pensano allo stesso modo, anche perchè uno dei pesi maggiori per gli sviluppatori JavaScript è il montaggio. Tuttavia Javascript è l'unico linguaggio che può essere eseguito in tutti i browser moderni senza l'ausilio di plug-in, indipendentemente dalla piattaforma: JavaScript è la "lingua franca"[64] del Web clien-

[64] McAllister, N., *The great JavaScript debate: Improve it or kill it?* http://www.infoworld.com/d/application-development/the-great-javascript-debate-improve-it-or-kill-it-173674

t-side. JavaScript è stato scelto come linguaggio di scripting per l'implementazione del software di visualizzazione EVT tenendo conto:

1. dell'importanza degli standard rispetto a framework/linguaggi proprietari (come Flash). Infatti, per produrre un software Open Source è necessario non essere vincolati da licenze commerciali;

1. della futura scelta operata da Microsoft (Windows 8) e Apple (iOS), oltre che da migliaia di sviluppatori, di utilizzare HTML5 e JavaScript come piattaforma per il Web 3.0.

Al momento della stesura di questa tesi[65]numerosi framework JavaScript di ottima qualità sono disponibili sul Web. I più noti sono MooTools, Dojo, Ext, YUI, Prototype e jQuery. Tutte queste strutture hanno caratteristiche interessanti, ma API molto diverse. Come fare quindi la scelta giusta? Brian Reindel, nel suo articolo intitolato "Come scegliere un framework JavaScript" presenta alcuni consigli per aiutarci a fare la scelta più corretta quando abbiamo bisogno di creare un'applicazione e vogliamo utilizzare le funzionalità di un framework:

> "A JavaScript framework may not make you a better programmer, but it will make you more efficient. That alone should be reason enough to choose a JavaScript framework, or library if you prefer. Unless you decide to build your own, there are plenty of options available to developers. However, choosing the right framework can be tricky, and weeding through a mess of opinionated fanboys (myself included) is intimidating.[66]"

L'attenzione di Reindel si concentra molto sulla comunità che si è sviluppata attorno all struttura del framework stesso: la sua documentazione, l'estensibilità, ecc...
La scelta è molto difficile, e si dovrebbe verificare attentamente quale API è quella più vicina alle esigenze del software che andremo a costruire.

---

65 Settembre 2011.
66 Reindel, B. *How to choose a JavaScript framework* (2008).

Quasi tutte le applicazioni che sono presenti sul Web oggi si basano su una serie di controlli dell'interfaccia utente non solo per mezzo di strutture che semplificano notevolmente lo sviluppo di applicazioni, ma anche fornendo interfacce grafiche interattive, coerenti e affidabili.
Non tutte le librerie sono adatte per ciascun tipo di progetto. Esaminiamo meglio i framework JavaScript per le interfacce GUI, ognuno dei quale offre una soluzione diversa.

1. **IT Mill Toolkit[67].** IT Mill Toolkit è un framework Open Source. Fornisce widget e strumenti per lo sviluppo di Rich Internet Application (RIA). Consente la progettazione di applicazioni Web, senza preoccuparsi dell' incompatibilità del browser.

2. **LivePipe IU[68].** ITLivePipe UI è una suite di widget di alta qualità e controlli per applicazioni Web 2.0 costruita utilizzando il Framework Prototype JavaScript. Ogni controllo è ben collaudato, altamente estensibile, ampiamente documentato.

3. **Iwebkit iPhone / iPod touch framework[69].** Iwebkit è il kit utilizzato per creare siti Web di alta qualità e iPod touch in pochi minuti. Rispetto ai primi quattro mesi della sua esistenza il gruppo si è notevolmente evoluto fino a guadagnarsi la fama mondiale.

4. **Jitsu[70].** Jitsu contiene un insieme integrato di strumenti che consentono agli sviluppatori di creare e implementare interfacce utente sofisticate per le applicazioni Web. Questi includono un linguaggio di markup XML, un motore di associazione dati, il runtime JavaScript, il motore di animazione, la libreria multi-piattaforma, e il supporto per Ajax.

[67]IT Mill Toolkit Official Website: http://www.itmill.com/.
[68]LivePipe IUOfficial Website: http://livepipe.net/.
[69]Iwebkit iPhone Official Website: http://snippetspace.com/.
[70]Jitsu Official Website: http://jitsu.org/.

5. **MochaUI**[71]. Si tratta di una "Web applications user interface library" basata sul framework JavaScript Mootools. I suoi usi riguardano applicazioni Web, siti Web, widget alle finestre di dialogo. MochaUI è un' estensione creata per il popolare framework Javascript “Mootools” e ExplorerCanvas.

6. **Echo Web Framework**[72]**.** Echo è un framework Open Source per lo sviluppo di RIA. Dal punto di vista delle azioni dello sviluppatore, Echo si configura come un insieme di strumenti di interfaccia utente - come Eclipse Swing o SWT.

7. **The Yahoo! User Interface Library (YUI)**[73]**.** La Biblioteca YUI è un insieme di *utility* e controlli, scritta in JavaScript, per costruire applicazioni Web fortemente interattive usando tecniche come DOM scripting, DHTML e AJAX. YUI è disponibile sotto licenza BSD ed è gratuito per tutti gli usi.
   YUI, attualmente alla sua terza edizione, è un archivio costantemente aggiornato grazie al suo grande team di sviluppatori. *Yahoo! YUI Library* è il padre dei framework di interfaccia utente ed è molto popolare sul Web. YUI è ricco di funzioni, componenti, servizi, moduli e controlli. Per quanto riguarda la ricerca, l'interfaccia utente YUI è forse una delle migliori attualmente in commercio.

8. **Sigma UI**[74]**.** Scritto in JavaScript e PHP, Sigma Ajax UI è un “Web-based builder” per gli sviluppatori professionali. Fornisce numerosi *TreeBar*, *TreeGrid*, Layout e Menu. Consente agli sviluppatori di risparmiare tempo nella creazione del prototipo e di applicazioni Web.

[71] Mocha UI Official Website: http://mochaui.org/.
[72] Echo Official Website: http://echo.nextapp.com/site/.
[73] YUI Official Website: http://developer.yahoo.com/yui/2/.
[74] Sigma UI Official Website: http://sourceforge.net/projects/ajaxuibuilder/.

9. **WUI Web UI Framework[75].** WUI (Web User Interface) per la scrittura di interfacce utente Web sviluppato un unico linguaggio: Java. Scrive applicazioni Web con componenti, widget, e gli eventi sono implementati in JSP. Funziona su qualsiasi servlet container. Simile ad Apache *Struts[76],* solo decisamente meglio.

10. **Butterfly Web UI[77].** È un framework di componenti per Java Web oriented, come Wicket[78] o Tapestry[79]. Il vantaggio principale di questo framework è che si integra naturalmente con *container* implementati nel progetto, fornendo una gerarchia che consente di separare i componenti interni dalle applicazioni Web.

11. **UKI - Simple UI Kit for Complex Web Apps[80].**UKI è un semplice toolkit grafico scritto in JavaScript. Crea applicazioni Web costruite per desktop. È un software veloce e leggero, fornisce una ricca biblioteca che va dai cursori per la vendita agli SplitPane. Utilizza noti linguaggi come DOM, JS, selettori CSS, eventi e attributi. Richiede molto tempo di apprendimento.

12. **JxLib[81]** JxLib è un framework JavaScript interfaccia utente costruito su MooTools. Esso fornisce gli elementi di base necessari per le applicazioni, come i pulsanti, le schede, i menu, le finestre di dialogo e alberi, così come una serie di altre funzioni aggiuntive. JxLib incorpora anche un meccanismo per la selezione di un *look and feel* attraverso le *skin*, sulla base di una serie di immagini e CSS.

[75] WUI Web UI Framework Official Website: http://wui.sourceforge.net/.
[76]Apache *Struts* Official Website: http://struts.apache.org/.
[77]Butterfly Web UI Official Website: http://butterfly.jenkov.com/webui/docs/configuration.html.
[78]Official Website: http://wicket.apache.org/.
[79]Official Website: http://tapestry.apache.org/.
[80]UKI Official Website: http://ukijs.org/.
[81]JxLib Official Website: http://jxlib.org.

13. **Dijit - The Dojo Toolkit[82].** Dijit è un sistema di widget di implementato su Dojo. Dijit è un buon punto di partenza se non si ha ancora buona dimestichezza con Dojo. È possibile creare incredibili esperienze Web 2.0 anche con una bassa competenza in JavaScript. Tutto in Dijit è progettato per essere accessibile a livello globale - per accogliere utenti con lingue diverse così come quelli con abilità diverse. Traduzioni, testo bidirezionale così come la rappresentazione di numeri e di date vengono incapsulati all'interno del widget.

14. **Qutensil JavaScript Toolset[83].** Qutensil è ancora in sviluppo, ma sta mostrando alcune implementazioni molto promettenti. È stato costruito sulle librerie Prototype e Scriptaculous. Comprende sistema di messaggistica, selettore dei colori, cursori, sistema di tooltip, finestre trascinabili.

15. **XUI[84]** XUI è un framework JavaScript pensato per la creazione di applicazioni Web Mobile. Il motivo per cui il peso di XUI è così leggero è dovuto al fatto che è completamente privo di tutti i codici che non sono essenziali per lo sviluppo di applicazioni mobili. Vuole essere un framework di riferimento di prima classe per i dispositivi mobili come Fennec[85] e Opera.

16. **iUI: iPhone User Interface Framework[86]** iUI è una libreria dal peso leggero con un'interfaccia minimalista per l'utente che sviluppa applicazioni iPhone uniformi.

17. **Alloy UI[87]** AlloyUI è un framework ricco di funzionalità, costruito su YUI 3 e in qualche misura su YUI 2. Comprende una vasta gamma di componenti - oltre 60 in tutto - che spaziano dalle utility alle widget UI. I suoi controlli

[82] The Dojo Toolkit: http://dojotoolkit.org/reference-guide/dijit/index.html.
[83] Qutensil JavaScript Official Website: http://qutensil.com/.
[84] XUI Official Websitehttp://xuijs.com/.
[85] Si tratta di Firefox mobile, precedentemente conosciuto con il nome di *Fennec.*
[86] iUI: iPhone User Interface Framework: http://code.google.com/p/iui/.
[87] Alloy UI Official Website: http://alloy.liferay.com/.

includono: immagini, dialoghi, TreeView, Panel, completamento automatico, Button, Calendar, Google Toolbar.

18. **Script.aculo.us**[88] script.aculo.us è un popolare kit per l'interfaccia utente che estende il framework Prototype con l'aggiunta di effetti visivi, controlli dell'interfaccia utente e le *utility* attraverso il DOM.

19. **QooXdoo**[89]**.** QooXdoo è un framework completo e innovativo per la creazione di RIA. Ha una licenza Open Source e gradevoli elementi di interfaccia utente. Purtroppo QooXdoo ha uno sviluppo recente. Non ha una comunità di sostegno (*community*) e non fornisce alcune delle caratteristiche che sono requisiti fondamentali per lo sviluppo del software di visualizzazione per il *Vercelli Book*. QooXdoo comprende una piattaforma indipendente e un avanzato servizio client-server a livello di comunicazione.

3.3.1 I principali candidati: Motools, Prototype e Backbase.

Le librerie sopra presentate spumeggiano ai margini rispetto ai *Requisiti Tecnici*[90] per l'implementazione del software EVT. La catalogazione dei framework prosegue quindi con l'analisi di altre tre librerie che iniziano ad avvicinarsi al modello richiesto per la creazione dell'interfaccia del *Vercelli Book*. Si tratta dei ben più conosciuti Motools, Prototype e Backbase. Vediamoli nel dettaglio.

1. **Motools**[91]. Motools è un framework JavaScript super leggero, un object-oriented. La dimensione della biblioteca è 55,9 Kb. La tabella che segue **(**Tabella 3**)** elenca i vantaggi e gli svantaggi dell'uso di Motools per le proprie applicazioni Web.

Vantaggi d'uso del Framework Mootols	*Svantaggi d'uso del Framework Mootols*
1. Si tratta di un framework	1. *Riutilizzo.* Rispetto a jQuery è

[88]Script.aculo.us Official Website: http://script.aculo.us/.
[89]Qooxdoo Official Website: http://qooxdoo.org/.
[90]Si veda il paragrrafo 3.2 “Fase I. Requisiti Tecnici”.

[91]Mootools Official Website: http://demos.mootools.net/.

altamente personalizzabile. 2. Presenta componenti quali MooTools Core . 3. *Software OOP.* Presenta una programmazione orientata agli oggetti. Ciò lo rende un quadro altamente personalizzabile.	più facile da imparare e da gestire, ma presenta alcuni problemi per il riutilizzo del codice e la sua manutenzione. jQuery, per esempio, non ha questo problema di riutilizzo. 2. *Plugin*. Ci sono solo una dozzina di plugin ufficiali su mootols.net. Molti meno rispetto a jQuery. 3. *Documentazione*. Motools ha poca documentazione ma è sempre più sulla scena internazionale. Manca un sito che fornisca informazioni dettagliate sull'interazione tra utenti e comunità.

Tabella 3. Vantaggi e svantaggi di Mootools

2. Prototype UI[92] è una libreria JavaScript facile da usare, ricca di componenti di interfaccia utente basata su Prototype e Script.aculo.us. L'interfaccia di Prototype fornisce moduli come Carousel, i quali possono essere utilizzati come pacchetto o in modo assolutamente indipendente.

Vantaggi d'uso del Framework Prototype	*Svantaggi d'uso del Framework Prototype*
1. *Animazioni*. Prototype fornisce strumenti per lavorare con gli elementi DOM della pagina. 2. *Metodi*. Proprio come nel caso di MooTools, Prototype offre una	1. *Durata del progetto.* Il framework Prototype è uno dei primi ad essere stati implementati. 2. *Script.aculo.us* (la libreria alla

[92]Prototype Official Website: http://www.prototypejs.org/.

vasta gamma di metodi (settando Element.Methods, si vanno a coprire la gran parte delle esigenze). 3. *Struttura basata su classi.* Implementato su una struttura orientata su oggetti, si distingue rispetto ai framework concorrenti.	base di Prototype) non è in grado di competere con altri framework, in particolare con jQuery, framework che lo ha superato nel campo della creazione degli effetti.

Tabella 4. Vantaggi e svantaggi di Prototype UI

3. **Backbase**[93] Backbase AJAX è un framework per la gestione di siti Web dinamici. Tramite l'installazione del pacchetto, è possibile, attraverso pochissime righe di codice generare applicazioni Web evolute dall'aspetto innovativo e dalle funzionalità degne di nota.

Vantaggi d'uso del Framework BackBase	*Svantaggi d'uso del Framework BackBase*
1. *Codice*. Utilizza taglibs in modo da non dover utilizzare troppo codice scritto in puro Javascript. Di conseguenza si ottiene un codice molto pulito. 2. *Interfaccia*. Presenta molti elementi per gestire l'interfaccia utente. 3. *Licenza*. Si tratta di un software gratuito. 4. *Documentazione*. Backbase ha una buona documentazione ed	1. *Difficoltà nel debug degli errori.* 2. *Lentezza.* È uno dei framework più lenti a livello di caricamento. Decisamente più lento di ExtJS e jQuery. 3. *Un software per pochi.* È necessaria una buona conoscenza di XML. Gli sviluppatori devono anche conoscere il linguaggio di programmazione JS. 4. *Caratteristiche del prodotto*. Il lato client del quadro ha un' interfaccia

[93]BackBase Official Website: http://www.backbase.com/.

esempi.	utente molto particolare. 5. *Comunità*. Come per Qooxdoo, la comunità dei partecipanti al progetto è ristretta.

Tabella 5. Vantaggi e svantaggi di Backbase

Tutti questi framework sono stati attentamente provati e valutati ai fini della creazione della versione digitale del *Vercelli Book*. In realtà, come emerso dall'analisi appena effettuata, nessuna delle librerie suggerite sopra rispondeva alle richieste specifiche del progetto. Diversi mesi sono stati dedicati alla selezione dei migliori framework JavaScript per il progetto. Molto tempo è stato speso nella ricerca del framework più adatto all'uso del software di visualizzazione EVT. In ultimo, maggior attenzione è stata rivolta nei confronti di Ext-JS e jQuery, che ha portato come decisione finale all'utilizzo di quest'ultimo. Vediamo adesso le ragioni che hanno portato a questa scelta.

3.3.2 Ext-Js versus jQuery

Ext-JS[94] è una libreria JavaScript cross-browser dedicata alla creazione di RIA. La libreria ExtJS offre un ambiente completo di sviluppo per quasi tutti i componenti presenti nelle normali applicazioni desktop.

Possiamo contare su un grande numero di componenti GUI, quali griglie, menu, finestre di dialogo, pulsanti, finestre, alberi, pannelli e molto altro ancora[95]. La prima intenzione nello sviluppo del progetto è stata quella, appunto, di utilizzare Ext-Js. Le ragioni di questa scelta erano state le seguenti:

1. *L'esperimento precedente.* Ext-JS è stato utilizzato da Francesca Fiorentini nel software per l'edizione digitale del *Vercelli Book* v.1.0 e riutilizzato nella

[94] *Ext JS* Documentazione ufficiale: http://www.sencha.com/products/js/.
[95] *What Ext JavaScript Library is all about* (Sencha 2008) http://www.sencha.com/learn/legacy/Screencast:What_Ext_JavaScript_Library_is_all_about.

tesi di Alessandra Giammalva[96]. Si tratta quindi di uno strumento ben definito e ha avuto successo in questo tipo di lavoro.

2. *Un marchio leader*. Ext-JS si presenta come uno dei prodotti leader nel mercato, nonostante i rapidi cambiamenti nella licenza commerciale.
3. *Design*. Il design di Ext-JS è uno dei più apprezzati a livello internazionale. Ext-JS offre un design è accettabile, esteticamente gradevole e confortevole.
4. *Comunità*. La comunità[97] è attiva e partecipa alla costruzione e sviluppo di nuovi materiali per Ext-JS.
5. *Elementi*. Ext-JS consta di un'articolata collezione di elementi dell'interfaccia utente. I temi grafici sono intercambiabili e tutti ben implementati.

In aggiunta si tratta di un software:

- distribuito in base a una licenza Open Source LGPL,
- compatibile con altri framework, ad esempio YUI, Prototype, e jQuery.

Al contrario, sono stati trovati i seguenti difetti:

1. *Pesantezza*. Notevole è la quantità di codice JS da caricare. Il JavaScript non compresso pesa circa 300-400Kb. La maggior parte dei progetti Ext-JS fornisce un "carico" prima che l'interfaccia utente possa iniziare a lavorare.
2. *Complessità del codice*. Il codice presentato da Ext-JS è molto complesso e non intuitivo.
3. Scarsa documentazione sul codice. Sebbene la comunità sia attiva, pochi sono i tutorial e le *community* di sviluppatori, diversamente da jQuery.
4. *Segnalazione degli errori*. Ho lavorato per diversi mesi con Ext-JS e ho verificato che manca si una buona soluzione nel campo della segnalazione degli errori.Si spendono ore cercando di capire dove trovare il codice di errore.

[96]Giammalva, A. *Il Flos di Leonardo Pisano, Edizione elettronica a cura di Alessandra Giammalva"* http://iu.di.unipi.it/flos/.
[97] Per il forum di consultazione di ExtJs si veda: https://www.sencha.com/forum/.

5. *Il tasso di utilizzo.* Ext-JS certamente non è veloce come altri framework. Ha un gran numero di elementi per l'interfaccia utente, ma risente in termini di lentezza e poca flessibilità nelle operazioni di carico.

In aggiunta, il documento deve presentare le seguenti righe di codice (all'interno del tag `<head>`) che rappresentano le librerie di Ext-JS:

```
<head>
    <link rel="stylesheet" type="text/css"
href="../extjs/resources/css/ext-all.css">
    <script type="text/javascript"
src="../extjs/adapter/ext/ext-base.js"></script>
    <script type="text/javascript" src="../extjs/ext-all-
debug.js"></script>
    <script type="text/javascript"
src="applayout.js"></script>

</head>
```

La scelta è quindi ricaduta su uno dei framework più utilizzati del momento: *jQuery*[98]. jQuery è una libreria costruita con widget e interazioni che permettono di animare i diversi elementi attraverso interazioni front-end con relativa facilità. Il pacchetto di interfaccia utente è sostanzialmente una raccolta di funzioni che possono essere suddivisi in tre moduli principali:

1. i widget che contengono interfacce utente predefinite e personalizzabili,
2. gli effetti e le animazioni – ad esempio è possibile lavorare su un elemento della pagina (scuoterlo, espanderlo, e così via),
3. interagire col mouse gli elementi della pagina espansa (drag and drop).

In particolare jQuery si contraddistingue per i seguenti vantaggi:

[98]jQuery Documentazione ufficiale disponibile alla URL: http://jquery.com/.

1. *Leggerezza*. È molto leggero. JQuery "pesa" circa 20 Kb in totale.
2. *Plugin*. Gli elementi dell'interfaccia utente e i jQuery plugin forniti sono facilmente personalizzabili e modificabili.
3. *Documentazione*. È rintracciabile una vasta documentazione di qualità.
4. *Comunità*. Ha un grande forum di supporto (e molte risposte alle domande più frequenti). È una comunità molto grande che implementa molti plugin. È vero jQuery UI che manca di elementi complessi come la griglia ExtJS, ma non sono stati necessari per la costruzione del nostro progetto.
5. *Licenza*. La licenza jQuery è Open Source.
6. *Integrazione*. È l'ideale se si desidera aggiungere JS o AJAX a una interfaccia utente esistente.

In aggiunta alle caratteristiche sopra descritte, JQuery:

1. è compatibile con qualsiasi altro framework (a differenza di altri),
2. viene utilizzato da WordPress e Drupal,
3. è stato scelto da Microsoft per l'integrazione con Visual Studio[99].

Tuttavia anche JQuery presenta alcuni svantaggi nel suo utilizzo, quali:

1. *Mancanza di stabilità*. La maggior parte delle funzionalità è disponibile in plugin da parte dei sostenitori di comunità diverse, quindi non ci sono elementi stabili. Oltre al fatto che le versioni di jQuery cambiano tanto velocemente esattamente come le versioni di Wordpress.
2. *Plugin*. La maggior parte dei plugin disponibili richiedono l'integrazione dello sviluppatore. È difficile trovare dei “pacchetti completi”.
3. *Interfaccia*. Come descritto spora, manca di elementi complessi di UI come la griglia ExtJS.

[99] *Visual Studio* Product: http://msdn.microsoft.com/it-it/vstudio/aa718325.

Per capire intuitivamente le differenze fondamentali che esistono tra jQuery e gli altri framework, si consideri il seguente confronto tra codici[100] per la costruzione di una tabella.

DOM

```
var tables = document.getElementsByTagName("table");
for ( var t = 0; t < tables.length; t++ ) {
var rows = tables[t].getElementsByTagName("tr");
 for ( var i = 1; i < rows.length; i += 2 )
     if ( !/(^|\s)odd(\s|
$)/.test( rows[i].className ) )
      rows[i].className += " odd";
     }
```

Tabella 6. Implementazione di una tabella con DOM

PROTOTYPE 1.5

```
$$("table").each(function(table){
  Selector.findChildElements(table, ["tr"])
    .findAll(function(row,i){ return i % 2 == 1; })
    .invoke("addClassName", "odd");
});
```

Tabella 7. Implementazione di una tabella con Prototype

MOTOOLS

```
$$("table").each(function(table){
S("tr", table).each(function(row,i){
if ( i % 2 == 1 )
row.addClass( "odd" );
  });
});
```

Tabella 8. Implementazione di una tabella con Mootools

JQUERY

```
$("tr:nth-child(odd)").addClass("odd");
```

Tabella 9. Implementazione di una tabella con jQuery

[100] Argomento trattato nel forum di discussione di jQuery: http://blog.jquery.com/.

La differenza è notevole: con jQuery le righe di codice sono notevolmente ridotte.
Questo appriccio LISP[101]-like può ridurre significativamente il numero di righe di codice. Questo è l'esempio ideale di filosofia jQuery: "scrivere meno, fare di più[102]". Scrivere codice JavaScript diventa davvero molto divertente per gli sviluppatori che si rendono conto della facilità con cui si possono realizzare alcune funzioni.

In conclusione, questi due framework analizzati (Ext-Js e JQuery) sono prodotti di alta qualità e completi. Essi permettono di creare applicazioni di nuova generazione e semplificano notevolmente il lavoro degli sviluppatori.
Tuttavia, ciò che è emerso da questa breve analisi, sono differenze sostanziali tra i vari software: considerando le caratteristiche uniche offerte dai vari framework Javascript, come pure l'inevitabile mix di vantaggi e svantaggi di ciascuno di essi, la scelta finale è caduta su jQuery, ritenuto il più appropriato per il tipo di progetto da portare a termine.
Pertanto, il linguaggi di programmazione che sono stati utilizzati nella creazione del software di visualizzazione del *Vercelli Book* all'interno del framework jQuery sono stati:

- XHTML 1.1,
- CSS 2.0,
- JavaScript.

Per essere in grado di integrare in maniera efficace tutti questi linguaggi di programmazione è stato usato *jFiddle*[103], un editor Web Based. *jsFiddle* consente di inserire, modificare e selezionare XHTML e CSS oltre che alcuni dei più noti framework Ajax tra cui, ad esempio, jQuery, Mootools, Prototype, YUI, Dojo e Glow e puro codice JavaScript (Figura 24).

[101]LIPS significa "Processor Language List" è un linguaggio di programmazione creato da John McCarthy nel 1959 e ampiamente utilizzato nel campo dell'intelligenza artificiale.
[102] Il motto originale di jQuery è: " write less, do more".
[103]JsFiddle Official Website: http://jsfiddle.net/.

L'applicazione consente di condividere il design tramite link, inserire codice, allineare il codice mal implementato grazie alla funzionalità TidyUp; e, in ultimo, effettuare delle prove utilizzando Ajax. *jsFiddle* offre anche una ricca libreria, pronta per essere modificata in base alle proprie esigenze.

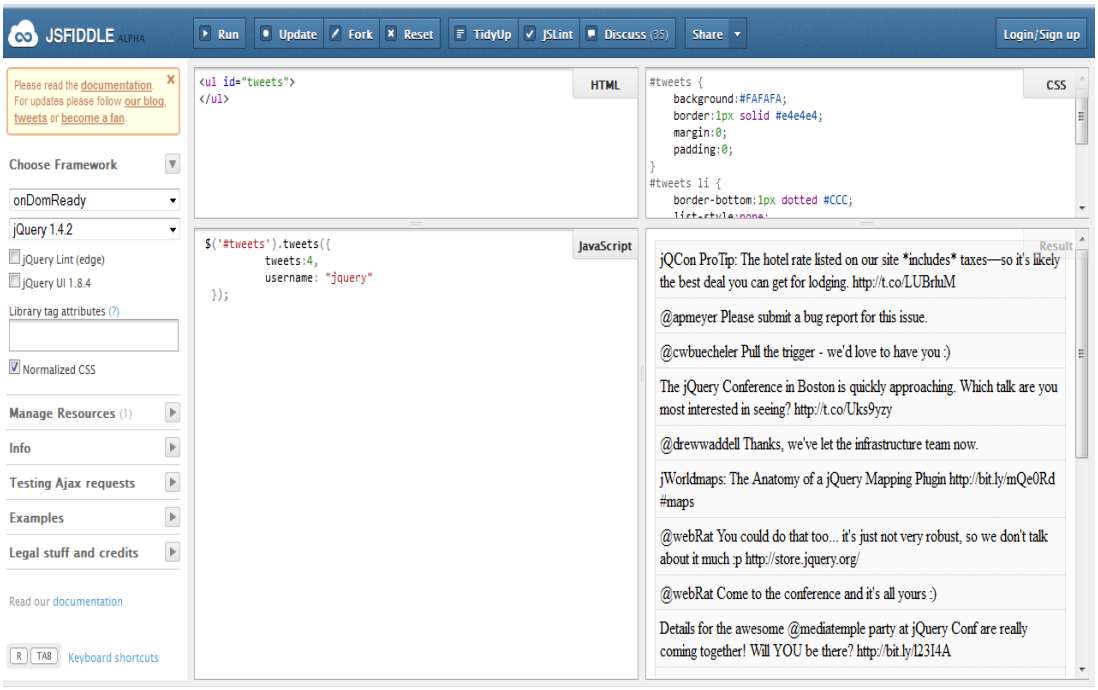


Figura 24 – jsfiddle.net

Come si può vedere aprendo la pagina del progetto, ci sono tre aree nella stessa schermata per inserire il testo in tempo reale e tutti gli elementi principali che compongono una pagina Web statica - il markup HTML, CSS e il codice JavaScript; infine, in basso a destra abbiamo una quarta area per testare il risultato dopo aver selezionato il pulsante e*segui*.

Con il comando *include* nel menu laterale a sinistra è anche possibile includere il codice all'interno dell'evento onLoad e le versioni delle librerie JavaScript più comuni (ad esempio, se voglio verificare la correttezza un codice scritto jQuery potrò stabilire la versione: JQuery 1.6.2, jQuery 1.5.2, jQuery 1.4.4 etc.).

Infine, *jsFiddle* offre la possibilità di simulare richieste asincrone, specificare il DTD, il tag title, e altro ancora, come riportato dalla documentazione [104].

[104]Per un'ulteriore documentazione di *jsFiddle* si consulti: http://doc.jsfiddle.net/.

3.4 Fase III. La scelta del modello più adatto

Le considerazioni emerse fino a questo momento costituiscono un terreno sufficiente per iniziare a costruire un primo modello di interfaccia GUI. Nel capitolo successivo (Capitolo IV) tratteremo la modalità di implementazione di questo modello seguendo un preciso iter e sfruttando le specifiche ufficiali.

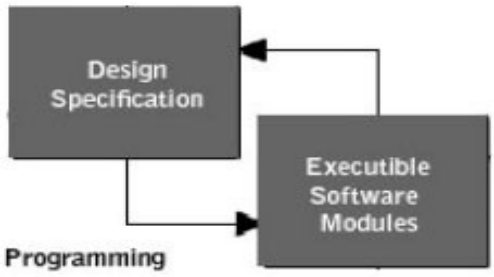


Figura 25 – Il Modello a Cascata (Fase 3)

Avendo attentamente valutato le edizioni precedentemente realizzate e disponibili sul Web possiamo delineare le caratteristiche principali del prototipo che andiamo a implementare. Innanzitutto la caratteristica principale sarà la presenza di numerose aree, personalizzabili tramite CSS. Saranno aree facilmente gestibili: ogni pannello potrà essere compresso e ridimensionabile e il layout potrà essere nidificato.
Avremo inoltre bisogno di:

- *un layout estensibile*: lo spazio sarà suddiviso fino a 5 regioni di layout, con la possibilità di nidificazione. Facile da usare, potente e semplice nella sintassi;
- *controllo dei CSS*: decine di classi generate automaticamente permetteranno di creare diversi stili per l'interfaccia utente;
- *gestione dei riferimenti*. Fornire pulsanti personalizzati di controllo: si integra con i pulsanti per uno stile personalizzato;

- *strumento nascondi/ripristina*: le finestre devono poter essere nascoste completamente in fase di avvio o in qualsiasi momento. Ogni pannello potrà essere chiuso e riaperto secondo la volontà dell'utente;
- *un layout ridimensionabile*: ogni pannello potrà essere ridimensionato nei limiti delle dimensioni automatiche permesse dal browser e definite dall'utente;
- *capacità di nidificazione*: creare *inner layout* per la gestione di layout complessi;
- *compatibilità con i widget UI*: per essere combinato con i widget dell'interfaccia utente.

Quindi, la divisione dello spazio della GUI può essere riassunta in questo modo:

- *Area Nord:* sezione orizzontale che occupa l'header della pagina. È l'area che definisce la modalità di visualizzazione. Essa gestisce i link per una visualizzazione immagine/testo, immagine/immagine e testo testo.
- *Area Centrale:* deve consentire la consultazione del manoscritto. I contenuti principali del manoscritto, ossia immagini e trascrizione del testo dovranno essere presentati attraverso due riquadri affiancati: questo è uno dei vantaggi principali messi a disposizione da un'edizione digitale: l'utente può avere accesso ai diversi contenuti del manoscritto attraverso uno strumento unico, permettendogli un confronto immediato e avendo la possibilità di interagire e intervenire sulla visualizzazione e l'analisi dei contenuti.
- *Aree laterali:* contengono schede che raccolgono le funzioni da applicare a immagini e testi.
- *Area Sud:* è la sezione orizzontale che occupa il footer della pagina. Contiene elementi a icona e la navigazione del manoscritto attraverso i *thumbnail*.

Lo scheletro del nostro progetto avrà quindi una struttura di questo tipo (Figura 26).

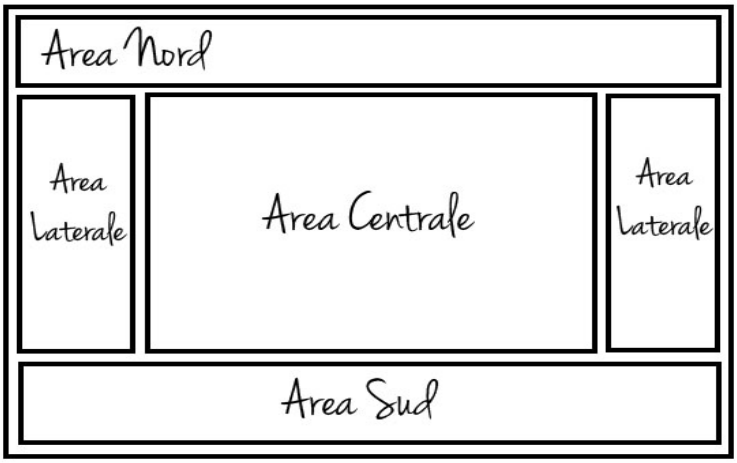


Figura 26 – Lo scheletro della GUI

In realtà, come vedremo nella realizzazione dei *mockup* e dal risultato finale, una delle due aree laterali (quella di destra) verrà abbandonata per lasciare maggiore spazio all'area centrale, che contiene il nucleo della visualizzazione.

3.4.1 Sincronizzazione immagine e testo (IMT)

Image Markup Tool[105] (o IMT) è un software che permette di annotare le immagini e memorizzare i dati in un file XML. Ci sono molti strumenti che permettono di ottenere gli stessi risultati: uno di questi è Inote[106] uno strumento di annotazione delle immagini scritto in Java. Tuttavia, diversamente da INote, IMT offre qualcosa di più:

- produce un file XML il cui formato è basato su TEI P5,
- ha un'interfaccia semplice ed intuitiva,
- è Open Source.

Per le edizioni digitali dei manoscritti è uno strumento decisamente potente: infatti spesso l'originale del manoscritto è danneggiato o comunque poco leggibile e una riproduzione di esso tramite immagine spesso non è sufficiente per la sua comprensione. Le annotazioni aggiunte all'immagine permettono così all'utente di accedere a

[105]IMT – Image Markup Tool: http://tapor.uvic.ca/~mholmes/image_markup/index.php.
[106]INote Official Website: http://www.iath.virginia.edu/inote/.

tutta una serie di apparati accessori, come note all'edizione, glosse o abbreviazioni, direttamente a partire dall'immagine, tramite *hot-spot* che visualizzano le informazioni richieste.

Il software di visualizzazione implememententato utilizza questo strumento per produrre immagini annotate del manoscritto: è possibile catalogare informazioni aggiuntive direttamente all'immagine e le annotazioni possono essere visualizzate dall'utente con un semplice click in qualsiasi punto dell' immagine[107]. Le annotazioni aggiunte all'immagine permettono all'utente di accedere a una gamma di funzionalità come ad esempio le note dell'edizione, le glosse tramite *hot-spot* che visualizzano le informazioni richieste. L'interfaccia grafica del software è molto intuitiva e facile da usare, consente di caricare le immagini originali, aggiungere annotazioni e salvare l'intero progetto provvedendo automaticamente alla generazione del codice necessario per visualizzare le aree annotate e utilizzarle automaticamente. L'interfaccia consente di aggiungere annotazioni tramite una finestra dedicata agli strumenti di markup. I file generati in automatico dal software comprendono:

- un file XML dell'immagine annotata,
- un file HTML,
- un file JavaScript,
- un foglio di stile XSL per la visualizzazione del risultato tramite Web-browser.

Il risultato che si ottiene è il seguente:

[107]Un esempio disponibile è "Le mariage sous L'Ancien Régime". Attualmente disponibile a questa URL: http://mariage.uvic.ca/.

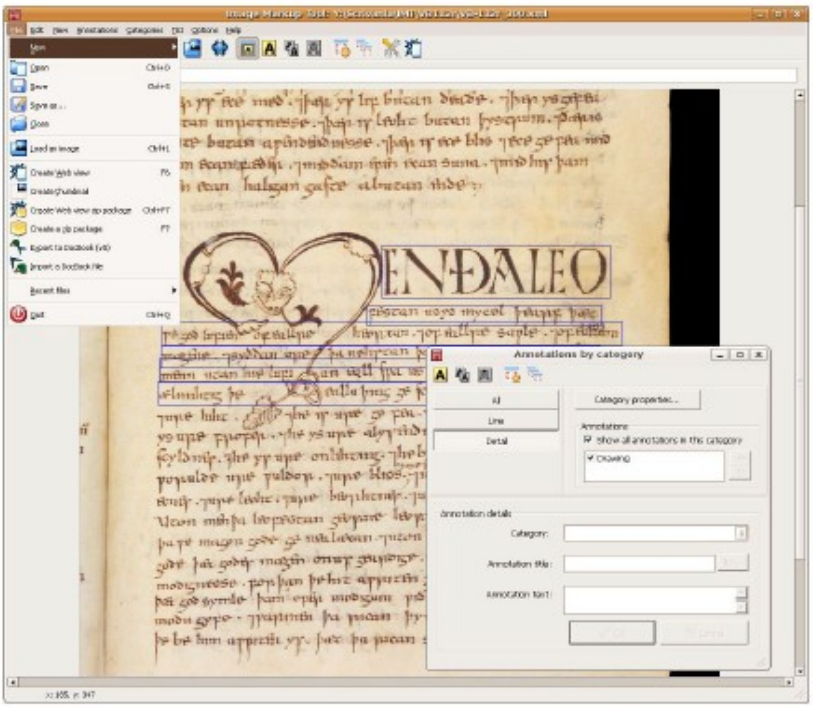


Figura 27 – Web View di un'immagine annotata con IMT

Come si può vedere dall'immagine la Web View comprende:

- *l'immagine annotata* – il testo delle annotazioni è visibile tramite piccole finestre poste accanto all'oggetto annotato o tramite *tooltips*. Le annotazioni sono state suddivise in categorie distinguibili grazie all'uso di colori diversi;
- *il menu delle annotazioni* – contiene l'insieme delle annotazioni e permette di scegliere, cliccando sul collegamento relativo, quale annotazione visualizzare. Viene generato in automatico dal programma.

CAPITOLO IV
IMPLEMENTAZIONE DEL PROTOTIPO

4.1 Prototipo finale: User Interface layout

Il prototipo finale si ispira direttamente al *border layout Ext-JS*[108] (Figura 28). A differenza di quest'ultimo, l'intera progettazione del prototipo non fa uso di un layout "pre-confezionato" sul quale è possibile aggiungere delle funzioni, ma è stato costruito su misura, esattamente come un lavoro di sartoria.

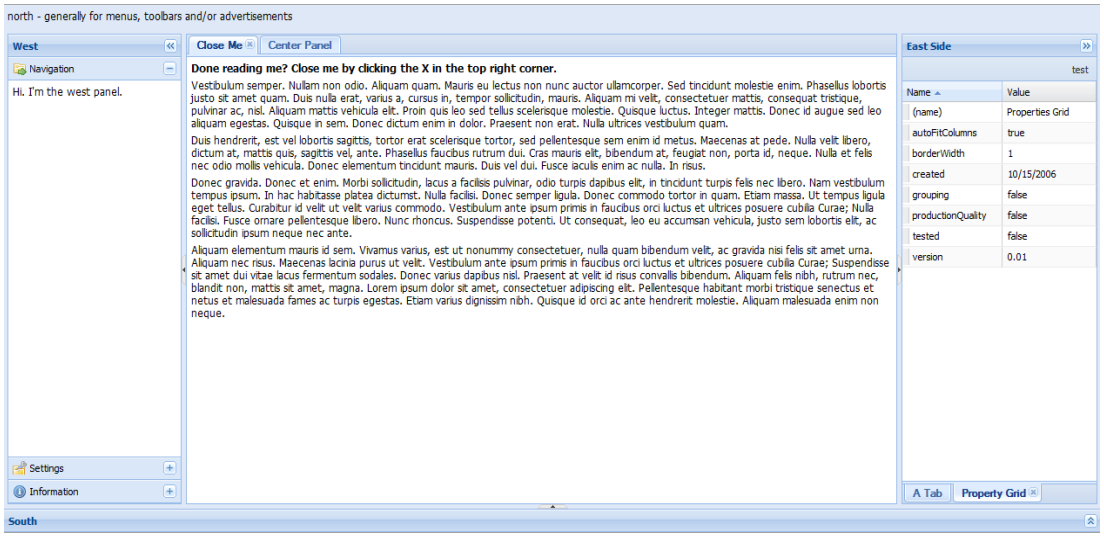


Figura 28 – Ext JS Border-layout

Nel layout del *Vercelli Book* tutti i riquadri sono circondati da zone ridimensionabili detti "riquadri di confine" (Nord, Sud, Est e Ovest). Questo è un esempio di aspetto minimalista con quattro border-layout, associato ad uno stile di base (applyDefaultStyles = true):

[108] L' *Ext-JS border layout* è disponibile a questa URL: http://extjs.com/deploy/dev/examples/layout/complex.html.

```
<html>
      <head>
      <script src="jquery.js"></script>
      <script src="jquery.layout.js"></script>
      <script>
          $(document).ready(function () {
              $('body').layout({ applyDefaultStyles: true
      });
          });
      </script>
      </head>
      <body>
      <div class="ui-layout-center">Center</div>
      <div class="ui-layout-north">North</div>
      <div class="ui-layout-south">South</div>
      <div class="ui-layout-east">East</div>
      <div class="ui-layout-west">West</div>
      </body>
</html>
```

Il risultato, con un foglio di stile di base è il seguente:

Figura 29 – Border-layout con jQuery (Base)

Questa base può essere personalizzata facendo uso di jQuery UI[109], Widget e una serie di Plugin[110]. In questo modo è possibile creare un'applicazione potente e sofisticata. A differenza di Ext-JS, la comunità jQuery mette a disposizione tanti plugin a seconda delle esigenze. Inoltre fornisce il servizio jQuery UI che consente di creare o personalizzare la base di una skin esistente per la modellazione dell'intefaccia. Questo servizio, chiamato ThemeRoller[111] è mostrato nella Figura 30. Nel caso del EVT è stata creata una skin su misura partendo da uno dei temi presenti nella galleria.

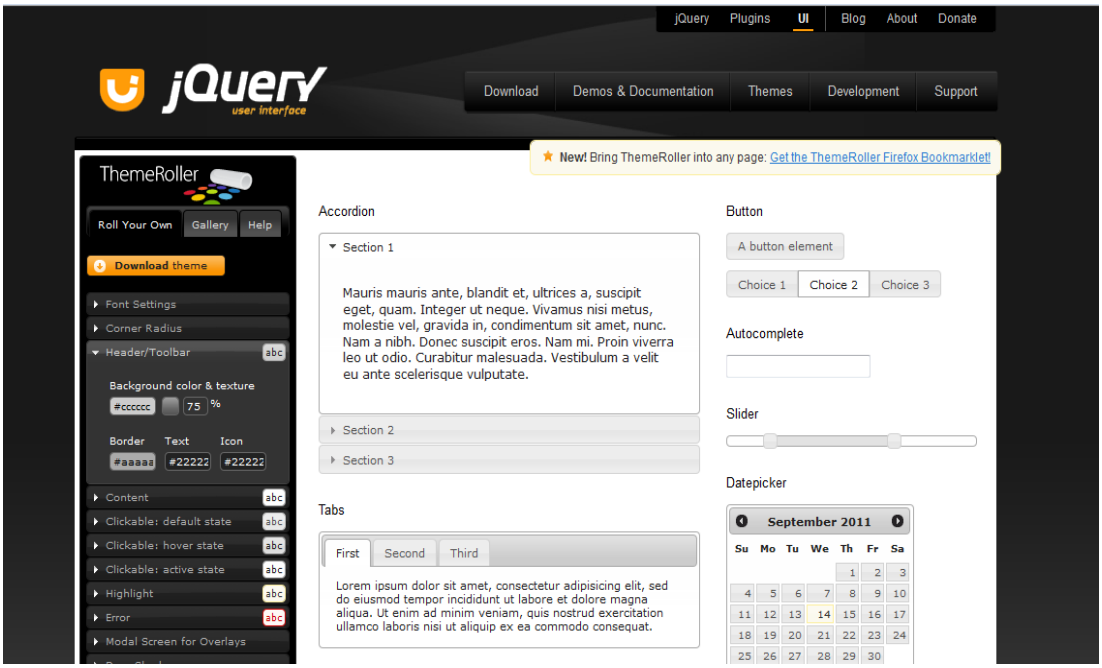


Figura 30 – jQuery UI (ThemeRoller)

4.1.1 Proprietà del Layout

Il layout del *Vercelli Book* è un layout estremamente complesso. Per capire la sua struttura introduciamo una serie di terminologie utilizzate nella progettazione.

[109] jQuery UI: http://jqueryui.com/.
[110] jQuery Plugins: http://plugins.jquery.com/.
[111] jQuery Themeroller: http://jqueryui.com/themeroller/.

Container. Il contenitore è l'elemento specificato durante la creazione di un layout. Più comunemente questo sarà l'elemento <body>, ma si potrebbe inserire un layout all'interno di qualsiasi elemento di blocco, compreso all'interno di un pannello di un altro layout (il cosidetto inner layout). Di seguito riportiamo un esempio che crea un layout che copre l'intera pagina <body>, e un secondo layout che si trova all'interno del riquadro centrale del primo layout:

```
$("body").layout();
$("body > .ui-layout-center").layout();
```

Quando un layout è stato creato, ha fino a cinque regioni: 'nord', 'sud', 'ovest', 'est' e 'center'. Queste regioni sono indicati come pannelli. La maggior parte delle opzioni si applica solo ai 'border-layout', il che significa tutti, tranne il pannello centrale.
Tutti i pannelli sono DIV esistenti nel codice HTML; i pannelli non sono generati automaticamente. Gli elementi del riquadro devono essere collegati direttamente con i 'figli' dell'elemento contenitore - cioè, non “nidificati” all'interno di altri elementi. Nel corso del progetto quando PANE è scritto in maiuscolo significa “uno dei pannelli”. Per esempio “ui-layout-PANE" significa "ui-layout-north", o "ui-layout-center", etc. La barra di ridimensionamento degli elementi ha il nome della classe collegata. Per esempio "ui-layout-resizer-west" è il resizer-bar per il pannello ovest: "ui-layout-pane-west".

Pane Spacing. Il pannello di spaziatura si riferisce allo spazio sul 'bordo interno' di un pannello. Per il riquadro Nord, questo è il bordo inferiore, per il riquadro ovest, è il bordo destro. Un riquadro può avere la spaziatura impostata a 0, il che significa che nessuno spazio esiste tra esso e le zone adiacenti. Se un riquadro non ha spazio, allora non può avere un resizer-bar o un *toggler*. La spaziatura può essere diversa quando il pannello è aperto rispetto a quando chiuso: ogni riquadro ha opzioni per “spacing_open” e “spacing_closed”.

Resizer. Un elemento DIV viene creato e riempie la 'distanza' tra i riquadri. Il DIV resizer è sempre l'intera larghezza della spaziatura e l'altezza del riquadro. I riquadri possono avere spaziatura diversa per gli stati aperti e chiusi. Il resizer-bar viene automaticamente ridimensionato e riposizionato ogni volta che viene aperto o chiuso un riquadro**.**

Slider. La resizer-bar agisce come uno slider bar quando il pannello è chiuso. Cliccando sul resizer o sulla barra di scorrimento ci sarà un riquadro aperto, nel senso che scivola sopra la parte superiore dei riquadri adiacenti senza ridimensionarle. Quando il mouse si sposta fuori dal riquadro, automaticamente viene richiusa. Questa funzione può essere personalizzata o disabilitata per ogni riquadro nelle opzioni di layout.

Toggler / Toggler-button. Questo DIV agisce come un pulsante toggler per aprire e chiudere un riquadro. È possibile effettuare l'aggiunta di colori, bordi e uno sfondo-immagine ad esso. Oppure è possibile inserire testo, immagini o HTML personalizzato all'interno del *toggler* utilizzando le opzioni *togglerContent*.
Il pulsante *toggler* riempie sempre l'intera larghezza della resizer-bar. La lunghezza del *toggler* è un'opzione, che può essere impostata al 50% oppure al 100%. Se è impostata al 100%, il toggler copre completamente il resizer-bar. Ogni pannello può avere diverse lunghezze *toggler*, e questi possono essere diversi a seconda che il pannello è aperto o chiuso, ad esempio:

```
,   togglerLength_open:         50
,   togglerLength_closed:       "100%"
,   north__togglerLength_open:  "100%"
,   south__togglerLength_open:  "100%"
```

4.1.2 Metodi, Opzioni e Classi

jQuery basa il suo linguaggio su una progettazione basata su eventi[112]. Essi richiedono l'inizializzazione di metodi, opzioni e classi. La maggior parte dei metodi implementati nell'EVT richiedono di specificare un pannello - di solito un border-pane: "nord", "sud", "est" o "ovest". I metodi utilizzati nel progetto sono stati:

- *open(pane).* Apre il riquadro specificato. Se il riquadro è già aperto, non succede nulla. Se il riquadro è attualmente nascosto (non chiuso!), il riquadro sarà impostato come *unhidden* e *open*.
- *close(pane).* Chiude il riquadro specificato. Se il riquadro è già chiuso, non succede nulla.
- *toggle(pane).* Alterna il riquadro specificato in aperto o chiuso impostando la funzione all'opposto del suo stato attuale.
- *sizePane(pane, size_in_pixels).* Determina la dimensione del pannello specificando i due valori pane e size_in_pixels.
- *hide(pane).* Nasconde il riquadro specificato. Quando un pannello è nascosto, non occupa spazio, come se non esistesse.
- *show(pane, openPane = true).* Normalmente si apre anche il pannello, ma se si passa 'falsÈ come secondo parametro, poi riquadro diventa *unhide*.
- *resizeContent(pane).* Ridimensiona il 'contenitore contenuto' all'interno del riquadro specificato. Il contenuto viene ridimensionato automaticamente quando il pannello viene ridimensionato o aperto, quindi questo metodo necessita di un richiamo manuale.
- *ResizeAll().* Ridimensiona l'intero layout per adattare il suo contenitore. Questo metodo viene eseguito automaticamente per tutti i layout quando la finestra del browser viene ridimensionata.

[112] Una guida dettagliata di jQuery è disponibile alla seguente URL: http://javascript.html.it/guide/leggi/168/guida-jquery/.

In jQuery ci sono opzioni per personalizzare quasi ogni aspetto di un layout, ma tutte le opzioni sono facoltative. Le opzioni di layout vengono “passate” durante la creazione di un layout. È possibile utilizzare le opzioni predefinite per creare un layout semplice e veloce, o creare un look completamente personalizzato per l'applicazione. Le opzioni di chiamata possono contribuire a integrare il layout. Per esempio, se i pannelli nord, sud ed est sono 'chiusi' quando il layout è stato creato - in modo che solo il riquadro ovest resti aperto – ciò significa che il layout è stato inizializzato come segue:

```
initClosed: true
west__initClosed: false
```

Questo fa sì che tutti i pannelli, tranne quello ovest, rimangano chiusi al momento della creazione. Il prototipo dell' EVT ha assegnato il valore “true” a tutti i pannelli.
Il formato attraverso cui è possibile specificare le opzioni è il comando *list*. Esso utilizza prefissi per identificare il nome del riquadro e due underscore, ad esempio:"north__fxName".
Opzioni specifiche senza un prefisso (ad esempio, fxName) sono considerate default predefinite. Il formato permette di specificare opzioni in qualsiasi ordine e secondo i parametri che vogliamo inizializzare. Di seguito riportiamo l'inizializzazione del campo <body>.

```
$("body").layout({
    fxName:  "slide"
,   fxSpeed:  "slow"
,   north__fxName:  "none"
,   south__fxName: "none"
,   spacing_closed: 14
,   north__spacing_closed: 8
,   south__spacing_closed: 8
,   north__togglerLength_closed:  "100%"
```

```
,  south__togglerLength_closed:  "100%"
,  initClosed:  true
}
```

Le opzioni possono essere impostate come default o essere impostate per riquadri specifici (ad esempio, "nord"). Riportiamo nella Tabella 10 tutte le opzioni utilizzate nel *Vercelli Book*.

Opzioni	Valori
Basic Layout/Pane Options	applyDefaultStyles, scrollToBookmarkOnLoad, showOverflowOnHover, closable, resizable slidable.
Element Selectors Strings	paneSelector, contentSelector, contentIgnoreSelector
Auto-generated Classnames	paneClass, resizerClass, togglerClass, buttonClass, Pane Size & Spacing.
Pane Size & Spacing	size (initial size) minSize, maxSize, spacing_open, spacing_closed.
Resizer-bar Formatting	resizerTip resizerCursor resizerDragOpacity sliderTip sliderCursor maskIframesOnResize slideTrigger_open slideTrigger_close.
Toggler-button Formatting	togglerTip_open, togglerTip_closed, togglerLength_open, togglerLength_closed, hideTogglerOnSlide, togglerAlign_open, togglerAlign_closed, togglerContent_open, togglerContent_closed.
Pane Animation Effects	fxName, fxName_open, fxName_close, fxSpeed fxSpeed_open, fxSpeed_close, fxSettings, fxSettings_open, fxSettings_close.
Layout State Initialization	initClosed, initHidden.
Layout Event Callbacks	onshow, onshow_start, onshow_end, onhide, onhide_start, onhide_end, onopen, onopen_start, onopen_end, onclose, onclose_start, onclose_end, onresize, onresize_start, onresize_end.
Element Selectors Strings	paneSelector, contentSelector, contentIgnoreSelector

Tabella 10. Elenco di opzioni jQuery integrate nel *Vercelli Book*.

Passiamo ora alla definizione delle classi. Le opzioni vengono integrate con il valore *class* nel nome (ad esempio togglerClass).
Per esempio, se si imposta questa opzione:

```
togglerClass: "myToggler″
```

Questo genera tutte queste classi per il bottone (supponiamo si tratti del bottone di sinistra):

```
class = "myToggler
        myToggler-west
        myToggler open
        myToggler-west-open"
```

Quando il riquadro ovest è chiuso, le ultime due classi diventano:

```
class="...
myToggler-closed
myToggler-west-closed"
```

Questo consente un controllo totale degli elementi usando i CSS: è possibile applicare stili globali usando "myToggler" o "myToggler-close", e stili per gli elementi specifici utilizzando "myToggler-west" o "myToggler-ovest-open".
Inoltre, quando si crea un layout, numerosi class-names CSS vengono aggiunti al pannello originale, così come gli elementi che generano il 'resizer-bar' e il toggler. Queste opzioni vengono utilizzate per impostare la base dei *class-name*:

```
paneClass:    "ui-layout-pane"
resizerClass: "ui-layout-resizer"
togglerClass: "ui-layout-toggler"
```

Ognuno dei 3 tipi di elementi - panes, resizers and togglers - ha più class-names generati sulla base di queste opzioni. Alcune classi cambiano per indicare se il pannello è aperto o chiuso. Questa serie di classi auto-generate rendono facile l'indicizzazione di tutti gli elementi come un gruppo, in particolare, lo stato (aperto / chiuso). Quindi i *resizer element* del pannello ovest saranno:

- `ui-layout-pane`
- `ui-layout-pane-[PANE-NAME]`
- `ui-layout-pane-[PANE-STATE]`
- `ui-layout-pane-[PANE-NAME]-[PANE-STATE]`

che associati al div si presentano in questo modo:

```
<div class="ui-layout-pane
            ui-layout-pane-west
            ui-layout-pane-open
            ui-layout-pane-west-open">
```

Quando invece il pannello è chiuso avremo:

```
<div class="...
            ui-layout-pane-closed
            ui-layout-pane-west-closed">
```

4.2 Un approccio per l'usabilità

Prima di procedere con la visualizzazione dei vari *mock-up* e l'implementazione della GUI finale è necessario soffermarci sull'importanza dell'usabilità nel progetto EVT (e non solo).

L'usabilità è un ulteriore passo verso la realizzazione di una buona interfaccia. Questo oncetto è strettamente legato al problema del target di riferimento[113]. L'usabilità tiene conto di quali sono i potenziali tipi di utenti che utilizzeranno il software e, di conseguenza, favorisce la presentazione dei contenuti (per esempio la navigazione),

[113] Si veda il Capitolo II.

la gestione di tutte le funzioni e gli strumenti che vogliamo associare all'interfaccia. Per la loro attuazione è tuttavia consigliabile seguire gli standard e le linee guida per l'usabilità. L'usabilità si applica a tutti aspetti di un sistema con cui un essere umano può interagire e descrive come le persone svolgono le attività con l'ausilio di sistemi computerizzati. Una GUI ben progettata è determinante per la facilità d'uso per l'utente. L'analogia dell'iceberg applicato all'usabilità[114] descrive come immagini, tecniche di interazione e il modello dell'utente possano contribuire all'interno di questo sistema (Figura 31). La metafora dell'iceberg è potente e d'ausilio alla spiegazione dei diversi aspetti della progettazione grafica.

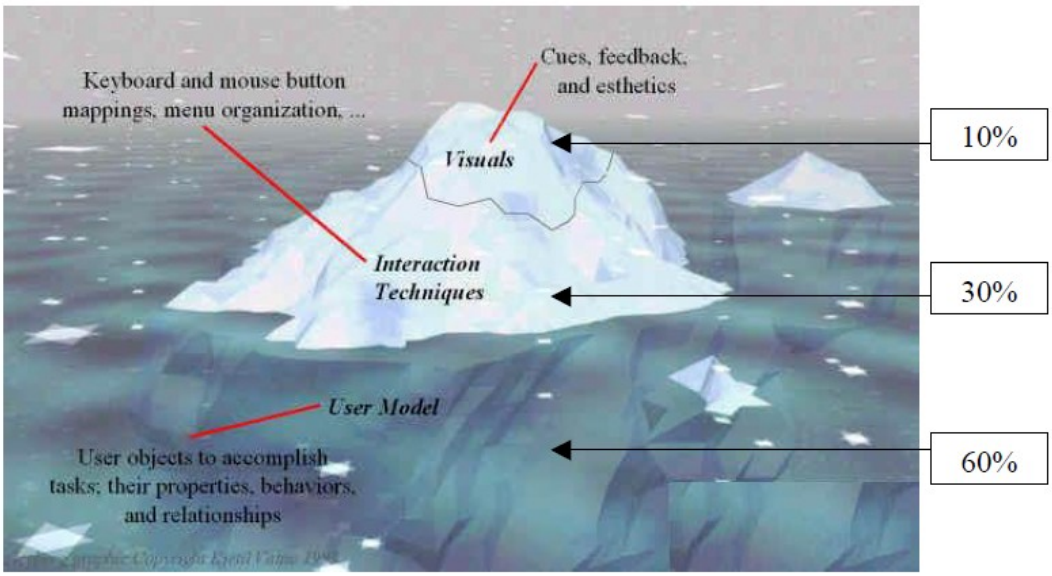


Figura 31 – The Iceberg Analogy of Usability

Questa metafora prevede diverse componenti:

- *Visuals.* Molti insegnamenti possono essere appresi dalla progettazione di cose quotidiane. L'estetica non è una questione soggettiva in quanto comprende l'uso del colore, stili di linea, tipografia ecc. Tuttavia, secondo questa scala di valori, l'aspetto è stimato a contribuire solo il 10% dell'usabilità complessiva.

[114] Berry, D. *The user experience - The iceberg analogy of usability,* 2006 (http://www.ibm.com/developerworks/library/w-berry/).

- *Interaction techniques.* Il 30% della fruibilità di un progetto deriva dalle tecniche di interazione. Concerne la preoccupazione sulle mappature, strutture di menu, le scorciatoie e la navigazione.
- *The user model.* Svolge il ruolo importante nel raggiungimento dell' usabilità, oltre il 60%. Questo concetto verifica le attività, le proprietà, i comportamenti e le relazioni dell'utente di fronte a un oggetto. Poiché il modello utente si trova sotto la superficie, la sua costruzione e il comportamento non sono evidenti e facile a vedersi: il fenomeno si verifica a causa di fattori sottostanti.

Se si vuole capire perché qualcosa non funziona come previsto o perché l'utente è insoddisfatto, si deve guardare alle regole formali. Questa parte è spesso il più difficile da affrontare, dal momento che il problema potrebbe essere di natura sociale e psicologica. Per raggiungere l'usabilità, l'interfaccia utente deve corrispondere al modello utente nell'utilizzo del programma. Se l'interfaccia grafica è coerente con il modello di utente, la GUI funzionerà come gli utenti si aspettano che accada.
Le diverse parti di una interfaccia utente sono altamente dipendenti le une dalle altre, dal momento che le proposte di design dovrebbero avere la loro origine nel modello utente.

4.3 Test di Usabilità

I test di usabilità sono in grado di fornire un prezioso feedback su come gli utenti percepiscono un sito Web o qualsiasi altro tipo di interfaccia utente.
Un test di usabilità è un metodo in cui il programmatore verifica la buona riuscita di una GUI. Lo sviluppatore dà ad uno o più utenti una serie di attività per completare l'utilizzo del sito. I test di usabilità offrono l'opportunità di osservare direttamente come gli utenti interagiscono con un'interfaccia.
Eseguire un test di usabilità è un processo iterativo, e rientra, sebbene in ultima istanza, nel processo di sviluppo di un software. L'approccio migliore è quello di eseguire una sessione con gli utenti (un test alla volta), identificare i problemi, migliorare l'interfaccia, e provare di nuovo. Così ho agito nello sviluppo della costruzione del soft-

ware di visualizzazione del *Vercelli Book* ed è anche uno di motivi per cui l'implementazione ha richiesto molto tempo.

Per essere più efficace, il test di usabilità deve far parte del ciclo di sviluppo. Idealmente, il collaudo ha inizio nelle prime fasi del modello a cascata. Identificare problemi di usabilità in una fase iniziale, consente al team del progetto di apportare modifiche prima di costruire l'interfaccia.

4.3.1 Mock Up v.1

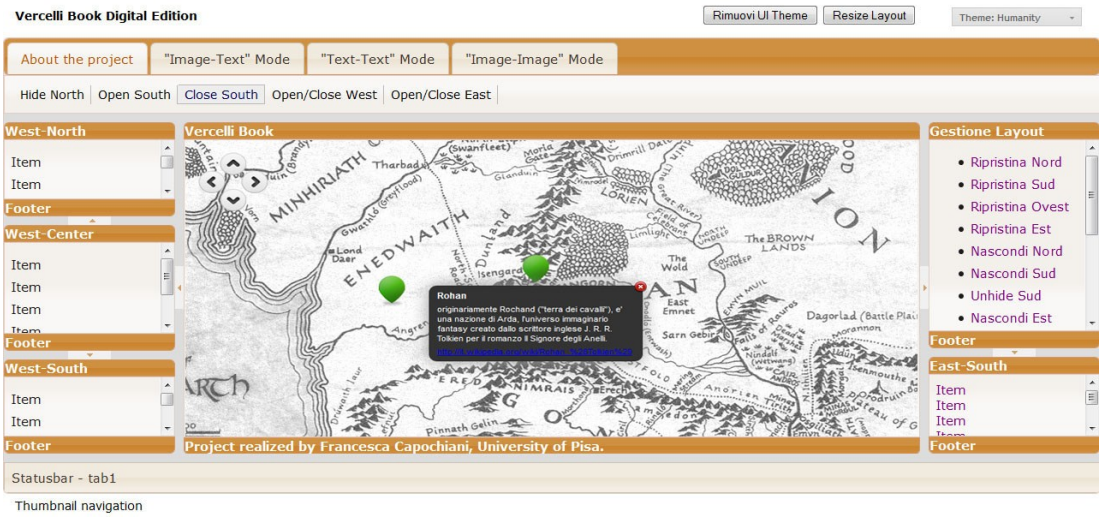


Figura 32 – Mock-up v.1

Il testing di usabilità è avvenuto su un campione di 10 persone, laureate e competenti nel settore dell'Informatica Umanistica. Abbiamo chiesto a ogni *task* di esprimere un valore da 1 a 5 come giudizio sulla GUI, rispondendo sui parametri di *learnability*, *speed of performance*[115] e *satisfaction*.

	Learnability	Speed of performance*	Satisfaction
Task 1	3	4	3
Task 2	3	4	2
Task 3	2	3	3
Task 4	3	4	2

[115]Questo parametro varia in base all'utilizzo del browser utilizzato dall'utente e al sistema operativo adottato.

Task 5	3	4	2
Task 6	2	3	3
Task 7	3	1	2
Task 8	3	4	3
Task 9	2	3	2
Task10	2	2	3

Tabella 11 – Risultato del testing effettuato sul MockUp v.1

Queste sono alcune delle osservazioni emerse:

- tutti gli utenti hanno preferito usare il mouse come strumento di navigazione e non la tastiera sebbene fossero state implementate funzioni per i "comandi brevi";
- tutti gli utenti hanno svolto i compiti senza grandi problemi;
- il prototipo non era adatto a tutte le risoluzioni dello schermo;
- il giudizio complessivo è stato basso sia sotto il profilo grafico sia sulla gestione dei comandi di navigazione.

Uno strumento alternativo e complementare al campione sono le *heat-map* per l'ottimizzazione delle GUI. Due tra gli strumenti più conosciuti sono *Feng-GUI*[116] e *Usaura*[117]. Si tratta di servizi web che creano automaticamente gli hot-spot al fine di ottimizzare e migliorare l'esperienza dell'utente. Una volta caricata l'immagine, è possibile condividere il test per svolgere l'indagine e individuare quali sono le zone più sensibili della GUI. Questi test, anche se piuttosto superficiali, permettono di scoprire quali sono le aree di maggiore interesse e di ottimizzare le posizioni degli oggetti della GUI. Ecco il risultato del test applicato al primo mock-up del *Vercelli Book* (Figura 33).

[116] Feng-gui Official Website: http://www.feng-gui.com/.
[117] Usaura Official Website: http://www.usaura.com/.

Figura 33 – Mock-up v.1 (Feng-GUI)

4.3.2 Mock Up v.2

Il risultato ottenuto da un primo re-design del progetto ha portato a un cambiamento sostanziale nella struttura. Ecco il secondo *mockup* realizzato con Pencil[118]

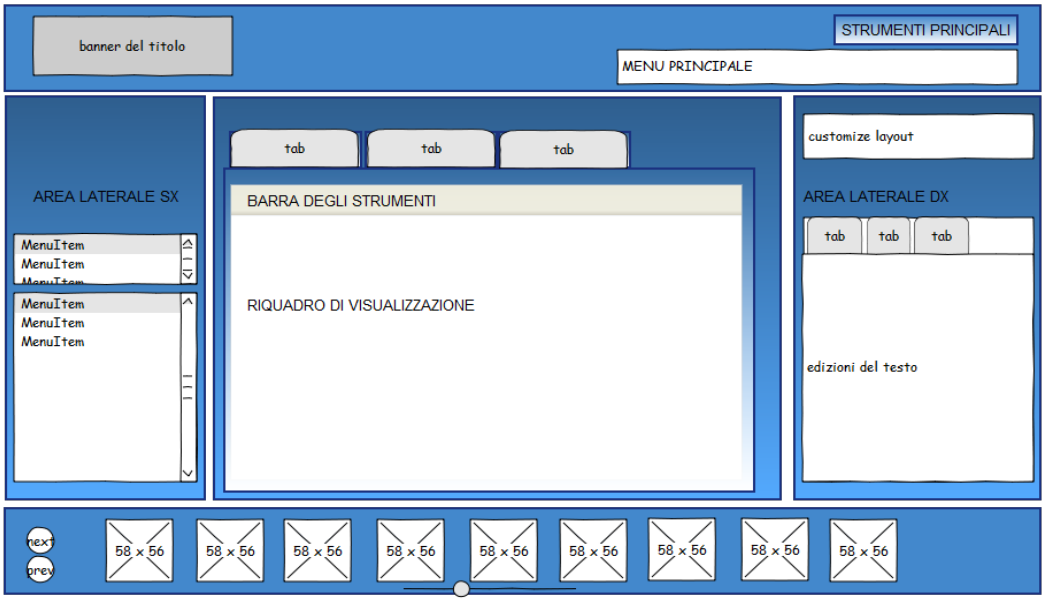


Figura 34 – Mock-up v.2

Presentiamo ora l'implementazione senza e con i fogli di stile (Figure 35 e 36) e analizziamo nuovamente il giudizio degli utenti.

[118] *The Pencil Project* Official Website: http://pencil.evolus.vn/.

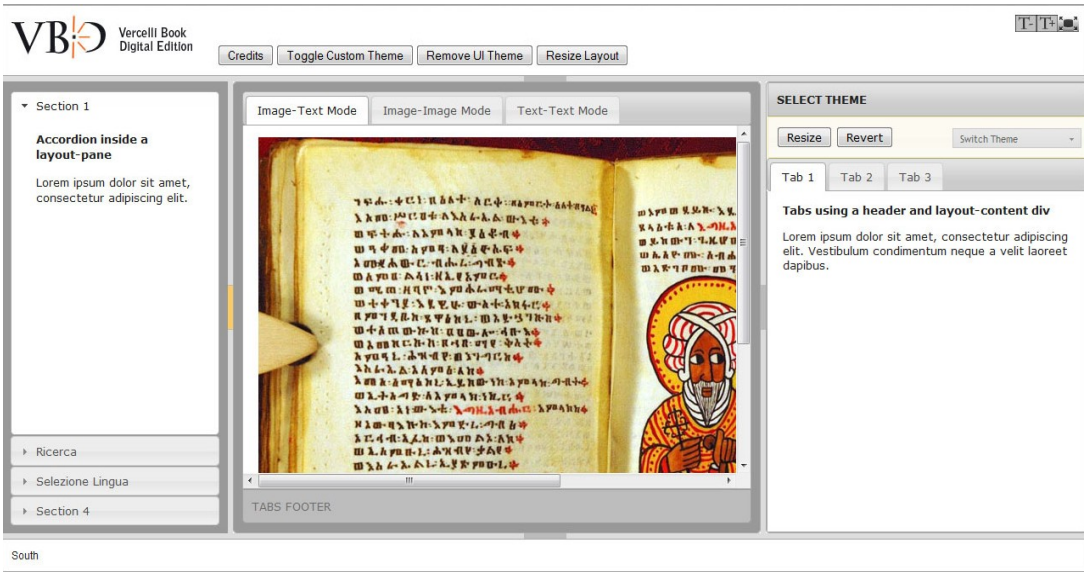


Figura 35 – Mock-up v.2 (senza fogli di stile)

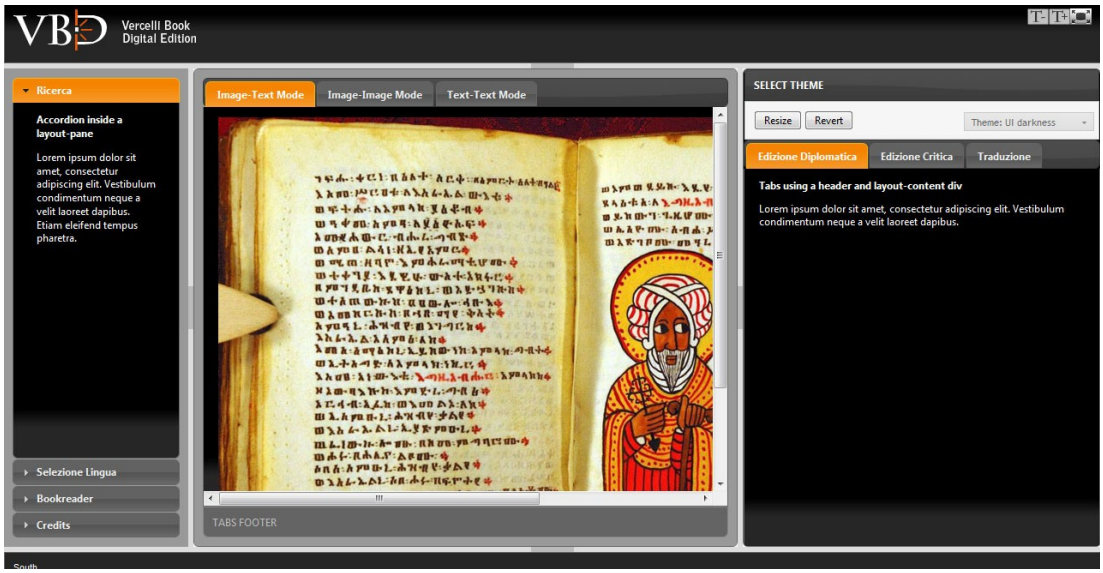


Figura 36 – Mock-up v.2 (con fogli di stile)

	Learnability	Speed of performance	Satisfaction
Task1	3	3	4
Task2	3	4	3
Task3	2	3	4
Task 4	3	4	2
Task 5	3	3	2
Task 6	4	2	3

Task 7	3	3	2
Task 8	2	4	3
Task 9	4	4	4
Task10	3	2	3

Tabella 12 . Risultato del testing effettuato sul MockUp v.2

Diversamente dal modello precedente, sono stati aggiunti:

- tabs per la navigazione testo/testo, immagine/testo, immagine/testo;
- *thumbnail navigation* nella zona del footer;
- un pulsante di personalizzazione grafica del layout (che si è rivelato deprecabile e superfluo).
- un unico menu principale.

L'effetto grafico è risultato più gradevole. Sopratutto grazie all'utilizzo dei fogli di stile. Una perplessità, che si è conservata anche nel MockUp v.3 e risolta solo nella versione finale, riguarda lo spazio della visualizzazione del testo e delle immagini.
Lo spazio per la visualizzazione è minimo nei MockUp v.2 e v.3. Bisogna fare un eccessivo ricorso al “resize” dei tab. E questo è deprecabile. Inoltre, l'area laterale destra si è rivelata superflua.

4.3.3 Mock Up v.3

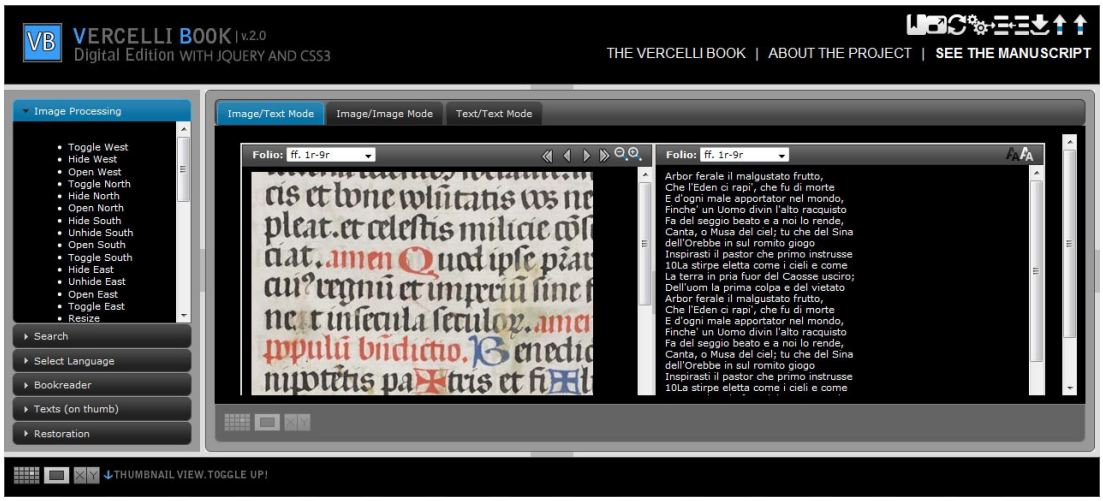


Figura 37 – Mock-up v.3 con Inner Layout

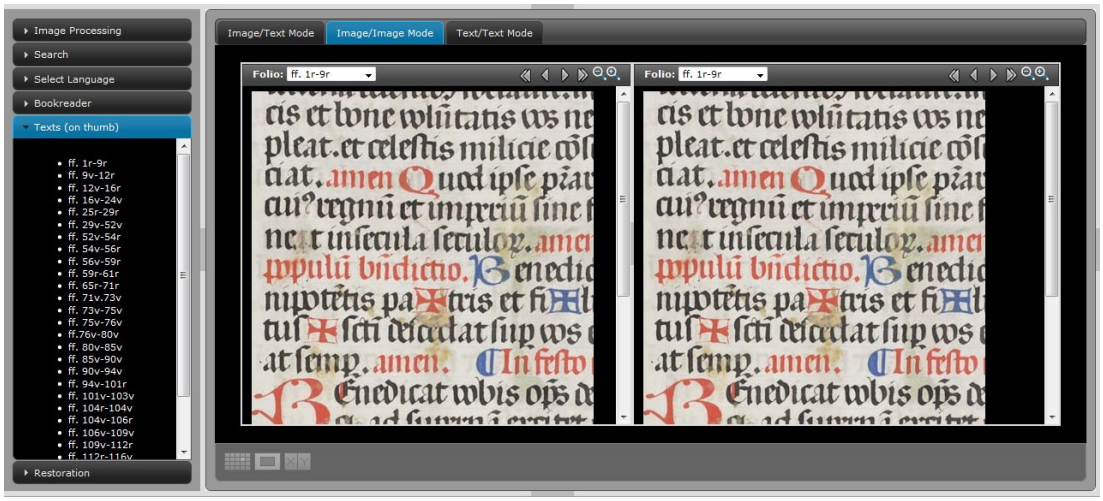


Figura 38 – Mock-up v.3 con Inner Layout (collapse)

Il MockUp v.3 parte dagli errori degli sviluppi precedenti e presenta ulteriori modifiche:

- assenza del tab laterale di destra;
- cambiamento della struttura grafica;
- eliminazione del pulsante di personalizzazione della skin ;
- espansione totale di tutte le aree del layout;

- creazione di un inner layout. Più precisamente si tratta di un layout dentro un container che viene ridimensionato automaticamente. Nell'esempio riportiamo la “chiamata” dell'innerLayout nel outerLayout.center (il pannello centrale).

```
outerLayout = $("body").layout({
center__onresize: "innerLayout.resizeAll"
});
```

Questo prototipo inizia ad avvicinarsi al modello finale ma, esattamente come nel MockUp v.2, il collasso delle sezioni non porta ad un piacevole risultato. Lo spazio non è ancora ben gestito. Il layout interno non è una soluzione adatta alle esigenze del software di visualizzazione del *Vercelli Book*.
Prima di concludere questa trattazione sui mockup riportamo, anche in questo caso, la tabella di valutazione degli utenti.

	Learnability	Speed of performance	Satisfaction
Task1	4	3	5
Task2	4	4	5
Task3	3	3	4
Task 4	3	4	3
Task 5	4	4	4
Task 6	2	3	3
Task 7	4	4	4
Task 8	3	4	3
Task 9	2	3	3
Task10	4	5	4

Tabella 13. Risultato del testing effettuato sul MockUp v.3

4.4 Implementazione della GUI

La proposta finale tiene conto di tutte le considerazioni e test precedentemente effettuati. Questa versione finale risolve il problema dell' *Inner layout* (MockUp v.2 e v.3). In essa è possibile collassare la barra delle miniature, la barra nord e quelle lat-

erali (destra e sinistra). Si può anche visualizzare a tutto schermo sia il progetto che una delle due finestre centrali.
Ho mantenuto separata la parte di programmazione dalla parte di codice HTML e, naturalmente, dalla parte di codice relativa alla visualizzazione grafica gestita tramite fogli di stile CSS. Andiamo ora ad analizzare nel dettaglio le sezioni.

4.4.1 Running the software: un tour con jQuery.

Per rendere la consultazione più veloce ed intuitiva ho cercato di strutturare il software nella maniera più semplice possibile. Tuttavia, visto che alcune funzionalità potrebbero risultare ambigue è stato realizzato un piccolo tour con jQuery in modo da accompagnare l'utente all'utilizzo di questa edizione. L'assistenza diretta all'utente viene anche suggerita tramite l'implementazione di *tooltip*.

4.4.2 Sezione Nord

Figura 39 – Implementazione del Prototipo: Sezione Nord

La sezione nord consta di due zone. Nella parte sinistra è collocato il logo del progetto. La grafica rispecchia i colori di jQuery. Ho insistito quindi sull'alternanza dei colori blu, grigio e nero. Nella parte destra sono disponibili tre pulsanti di navigazione che richiamano la visualizzazione del manoscritto secondo tre modalità di visualizzazione dell'immagine. Ciò significa che avremo un pulsante per la visualizzazione immagine/testo, uno per la visualizzazione immagine/immagine e uno per il confronto testo/testo.

4.4.3 Sezione Sud

Questo oggetto, collocato nella parte del footer, costituisce uno strumento di navigazione molto efficace, specialmente per chi già conosce i contenuti del manoscritto. L'area di anteprima è situata nella parte inferiore della finestra principale seguendo

l'implementazione del *Digital Library of the Caribbean*. Si discosta invece dalla collocazione sulla parete superiore presentata nel *Virtual Vellum*[119].

Lo strumento mostra tutte le miniature che riproducono i facsimile dell'originale: l'utente può quindi scegliere i contenuti da visualizzare nelle finestre centrali cliccando sulla miniatura. Tramite questa azione di default vengono visualizzati nella finestra di sinistra, l'immagine e, in quella di destra, la trascrizione del testo, relative alla miniatura selezionata.

Figura 40 – Implementazione del Prototipo: Sezione Sud

Quando ci sono più miniature da visualizzare, come nel caso del manoscritto del *Vercelli Book* che, ricordiamo, consta di 23 omelie e 6 testi poetici, non vi è spazio disponibile nella finestra di anteprima; la barra di scorrimento che scorre lungo la parte inferiore dell'area di anteprima viene attivata per navigare avanti e indietro (prev-next) attraverso la raccolta.

La sezione Sud presenta anche tre pulsanti: il primo apre un dialog che racchiude tutti i *thumbs* del manoscritto, il secondo permette la visualizzazione a schermo intero e il terzo consente una doppia visualizzazione (fronte-retro).

4.4.4 Sezione Centrale

La sezione centrale è in assoluto la zona di implementazione più complessa in quanto rappresenta la zona di visualizzazione delle finestre. Essa deve permettere la visualizzazione secondo le tre modalità presentate nella sezione nord.

[119] Capitolo II.

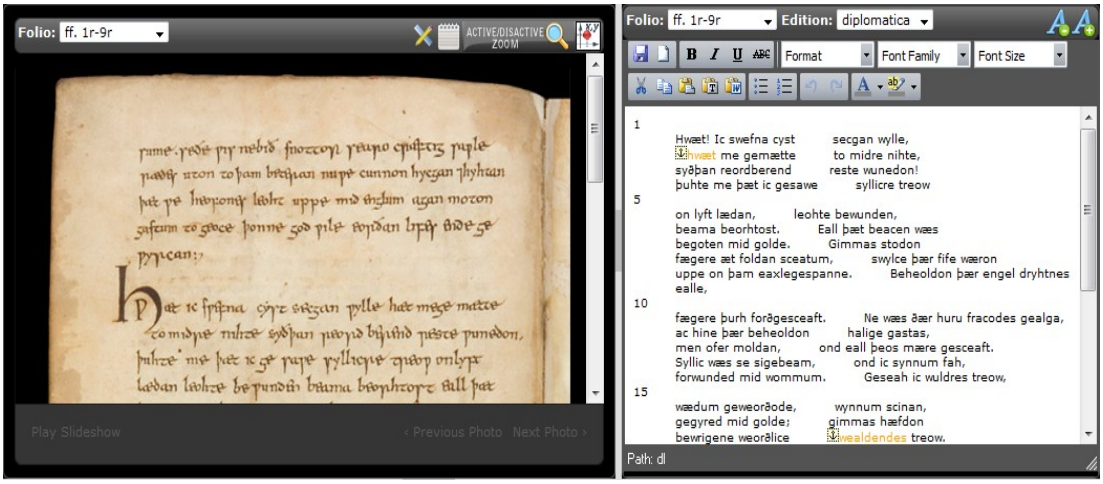


Figura 41 – Implementazione del Prototipo: Sezione Centrale

La barra degli strumenti dell'immagine consta di:

- *Una select text.* Il pulsante Select Text apre un menu a tendina, attraverso il quale si possono selezionare i diversi fogli del manoscritto. In particolare, nel menu Select Text, nella finestra di sinistra, sono presenti le pagine della trascrizione del testo suddivise per fogli.
- *Un righello.* Questa funzione viene richiamata nella pagina tramite il file rule-loader.js La barra del righello orizzontale e verticale si ridimensiona automaticamente. Il righello appare in cima l'immagine e sul lato sinistro dello schermo.
- *Un pulsante Skicky Notes.* Grazie all'introduzione degli Sticky Notes gli utenti possono annotare le loro osservazioni riguardo l'edizione. Le annotazioni costituiscono, come il righello, una delle funzionalità avanzate nella costruzione di una EVT.
- *Un pulsante coordinate.*
- *Una lente di ingrandimento per l'immagine.* La funzionalità 'lente di ingrandimento' serve a mettere in evidenza, ingrandedola appunto, la parte dell'immagine su cui viene attivata. Si accede a questo strumento cliccando sul pulsante relativo che si trova sulla barra degli strumenti delle finestre centrali. Al passaggio del mouse sopra l'immagine viene visualizzato un riquadro che visu-

alizza l'ingrandimento. La profondità e la grandezza della porzione di immagine da ingrandire possono essere modificate cliccando e trascinando il mouse: dall'alto in basso viene aumentata e diminuita la profondità dello zoom, mentre da sinistra verso destra è ingrandito o ridotto il riquadro dello zoom.

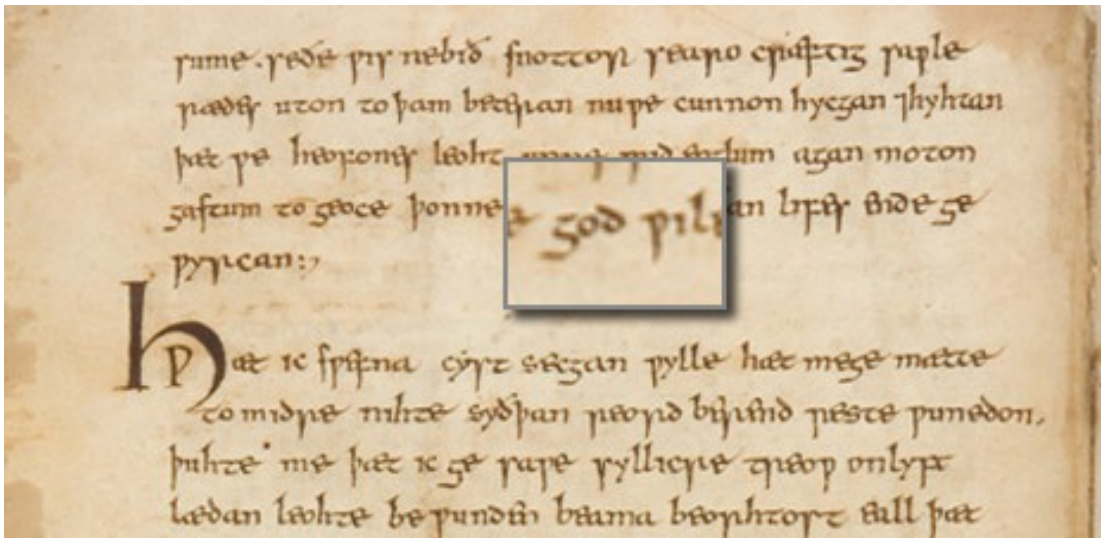


Figura 41b – Implementazione del Prototipo: Zoom

La barra degli strumenti del testo invece presenta:

- *Due select text.* Oltre al menu che seleziona i folii (r-v) vengono configurate le edizioni disponibili: versione diplomatica, traduzione e versione critica. È comunque è possibile espandere questa funzionalità.
- *Un editor WYSIWYG.* Nei progetti Web, e ancor di più nella realizzazione di un software EVT, quando vogliamo consentire all'utente di inserire testi e immagini, si rivelano molto utili i cosiddetti "editor JavaScript". Questi editor sostituiscono la classica casella di testo fornendo funzioni come la formattazione e allineamento del testo, inserimento di immagini, tabelle ed elenchi puntati. *TinyMCE*[120], uno degli editor JavaScript più diffusi, a cui si appoggiano anche applicazioni commerciali. È un prodotto gratuito, rilasciato con licenza LGPL. È compatibile con la quasi totalità dei browser esistenti ed

[120] L'editor TinyMCE è disponibile a questo indirizzo: http://www.tinymce.com/

è realizzato interamente in JavaScript. Può essere integrato in siti di qualunque tipo: HTML statico, PHP, ASP, ASP.NET, JSF.
Dopo aver scaricato l'archivio contentente la piattaforma e la sua documentazione, è stata estratta la cartella "tiny_mce" contenuta in "`tinymce/jscripts/`" e salvata nella root del progetto. Di seguito, è stata inserita la seguente stringa nell'instestazione `</head>`:

```
<script src="tiny_mce/tiny_mce.js" language="javascript"
type="text/javscript">
</script>
<script language="javascript" type="text/javascript">
tinyMCE.init({
 mode : "textareas"
});
</script>
```

Lo spaccato di codice inserito nella sezione di installazione va ad inizializzare l'editor WYSIWYG su tutti i campi `<textarea>` presenti nella pagina (mode :"textareas"). Altre opzioni per il parametro *mode* sono:

- *none* - non converte nulla; utile se si imposta un pulsante per l'attivazione dello script.
- *specific_textareas* - converte solo le textarea con un trigger impostato su *true.*
- *exact* - converte solo una determinata textarea o div.

Molto importante è la gestione delle toolbar con i pulsanti: è infatti questa la sezione necessaria per personalizzare tinyMCE. Esistono due differenti profili per la configurazione dei pulsanti: *simple* e *advanced*. La versione semplice non prevede modifiche ai pulsanti e integra di base i pulsanti più comuni e

importanti quali grassetto (**bold**), corsivo (*italic*) e sottolineato (underline) insieme agli elenchi puntati/numerati e la gestione dei links. La versione *advanced*, invece, permette una configurazione molto più ampia.

```
tinyMCE.init({
  …
  theme : "advanced",theme_advanced_buttons1 :
"inserttime,preview,zoom,separator,forecolor",
 theme_advanced_buttons2 : "bullist,numlist,se-
parator,outdent,indent,separator",
  theme_advanced_buttons3 : ""
});
```

La codifica dei caratteri è stata realizzata con Junicode[121] (abbreviazione di Junius-Unicode). La versione attuale è una beta. Junicode attualmente contiene 3.096 *characters* in stile regolare (corsivo, grassetto e grassetto corsivo sono meno completi).

- *Due pulsanti di ingrandimento del testo* A+,A- La leggibilità del testo è uno dei requisiti di base. Molte sono le edizioni consultate[122] che presentano questo problema. In generale, il tasto di icona associato a questa funzione, a volte è assente a altre volte è difficile da individuare.

4.4.4 Sezione Ovest

La sezione ovest è una sezione dinamica che varia a seconda della modalità di visualizzazione del manoscritto. Le pagine di confronto immagine/testo, immagine/immagine e testo/testo presentano tre menu laterali come riportato in Figura 42.

[121] Documentazione Junicode: http://junicode.sourceforge.net/.
[122] Per esempio il *Canterbury Tales Project.*

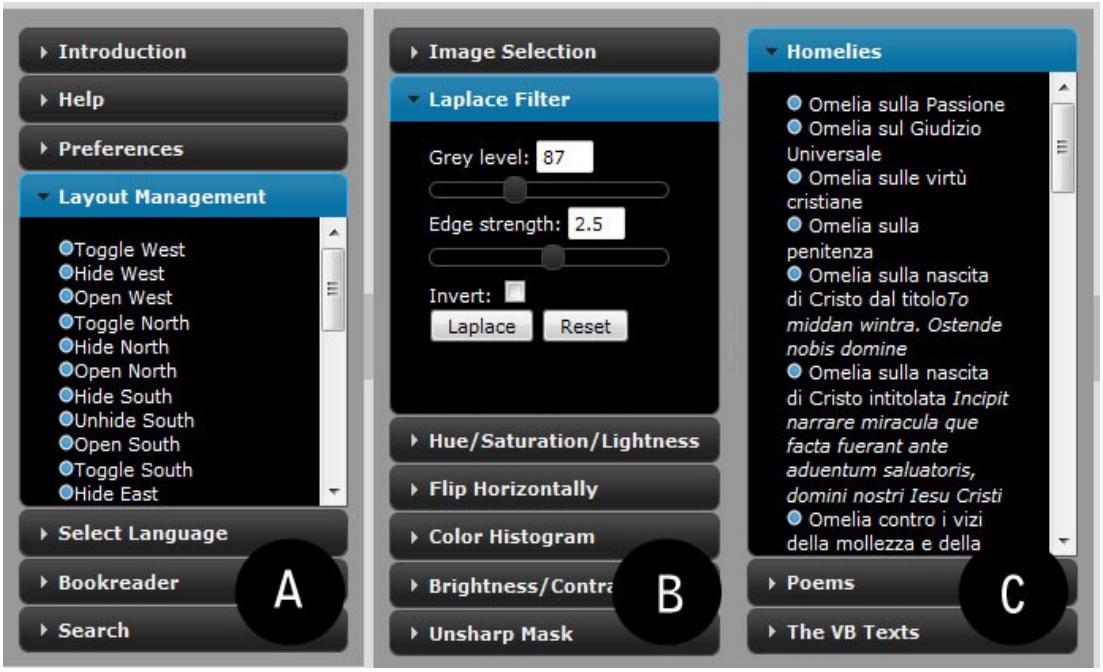


Figura 42 – Implementazione del Prototipo: Sezione Ovest

Il menù presenta un effetto *accordition* e si articola come in Figura 42.

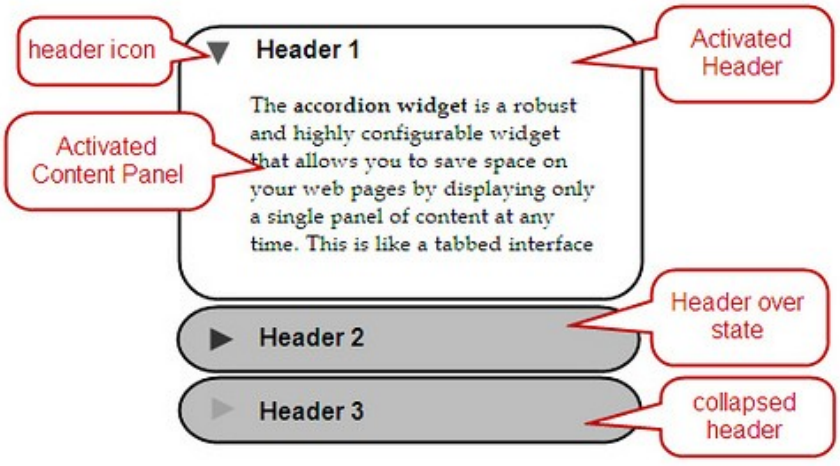


Figura 43 – Architettura del menu Accordition

Cliccando sul pulsante di attivazione si apre un elenco di contenuti speciali: si può accedere alle spiegazioni, a funzioni complete sul manoscritto, sia lavorando sull'edizione diplomatica che sulla traduzione o sulla versione critica.

Passiamo ora ad analizzare nel dettaglio le funzioni implementate nei tre menu.
Nella pagina di visualizzazione immagine/testo troviamo i seguenti pulsanti (Figura 41a):

- *Introduction.*. Il pulsante apre una finestra contenente i dati principali di questa edizione del *Vercelli Book*.
- *Help.* Il pulsante Help apre una finestra contenente le istruzioni per la consultazione del *Vercelli Book* e per il corretto utilizzo degli strumenti.
- *Preferences.* Questa zona è dedicata alle funzioni dello schermo e della skin. Si tratta delle funzioni *refresh page*, *full screen*, *resize theme* e *revert theme.*
- *Layout management.* Sono state qui implementate una serie di funzioni per la gestione del layout. In particolar modo le funzionalità *Toggle, Hide e Open* (adattate ai pannelli West, East, North, South) permettono di nascondere, ripristinare o chiudere la zona selezionata. In più abbiamo le funzioni *Resize North*, *Resize South* e dodici funzioni *Show Option*.
- *Select Language.* Abbiamo più volte ribadito durante questa trattazione che la divulgazione del software è uno degli obiettivi finali a cui si deve tendere. Ebbene, implementare il software in diverse lingue è un mezzo di trasmissione efficace ed efficente per questo scopo.
- *Bookreader*[123]. È uno strumento implementato per la visualizzazione delle immagini del manoscritto in un layout flipbook. Le opzioni includono una tastiera di navigazione (utilizzare le frecce), i numeri di pagina e le ombre (che si attivano al momento dell'animazione).
- *Search.* Gli utenti che consultano un EVT spesso hanno bisogno di funzionalità di ricerca avanzata. Soprattutto quando il testo sottostante è codificato in eXtensible Markup Language (XML). Inserire uno strumento di ricerca avanzata dovrebbe includere opzioni che consentono all'utente di approfittare della potenzialità di marcatura data dall'XML. L'attuale progettazione del *Vercelli Book* manca del lato server (come abbiamo detto[124] le ultime due fasi del

[123]Per una trattazione sul ruolo del BookReader si rimanda al Capitolo II.
[124] Nel Capitolo III.

modello a cascata non vengono trattate in questa sede). Non è stato ancora possibile implementare la parte di ricerca sul testo ma in futuro verrà utilizzato un Database XML che svolgerà questa funzione. Sono stati attualmente consultati sei database XML: Xaria, Philologic, XTF[125], eXist-DB[126], Base X e Berkeley. I candidati all'implementazione sono eXist-DB e XTF.

Introduction, *Help* e *Preferences* sono finestre di dialogo. Un dialogo è una finestra mobile che contiene una barra del titolo e una zona contenuto. La finestra di dialogo può essere spostata, ridimensionata e chiusa con l'icona di default "X". Se la lunghezza del contenuto supera l'altezza massima, una barra di scorrimento apparirà automaticamente.

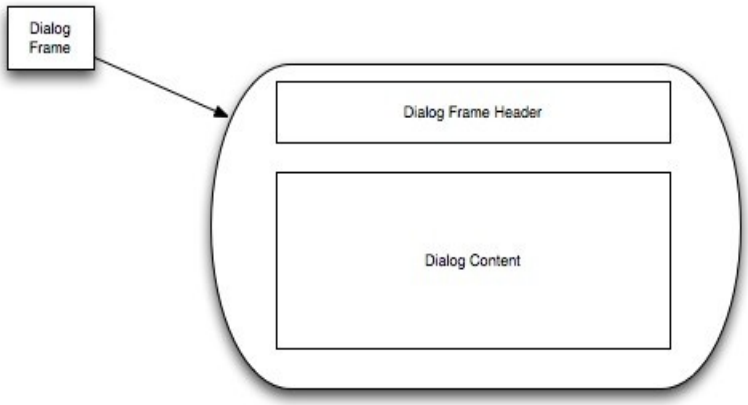


Figura 44 – Architettura di una finestra di dialogo

Per realizzare una chiamata di dialogo utilizziamo `$(foo).dialog()` Questa chiamata inizializza e apre la finestra in modalità default. Ecco la struttura di un esempio di chiamata:

Definiamo la funzione:

```
$("#dialog").dialog();
```

[125] Documentazione su XTF: http://www.cdlib.org/services/publishing/tools/xtf/.
[126] Documentazione su Exist-DB: http://exist.sourceforge.net/.

Riportiamo all'interno del tag <head> il seguente script:

```
...
  <script>
  $(document).ready(function() {
    $("#dialog").dialog();
  });
  </script>
...
```

E richiamiamo nel campo <body> come segue:

```
<body>
<div id="dialog" title="Dialog Title">I'm in a
dialog</div>
</body>
```

Passiamo adesso alla pagina di confronto tra immagini. In essa sono state implementate le seguenti funzionalità (Figura 41b):

- *Laplace Filter.* L' EVT del *Vercelli Book* svolge, a differenza delle altre edizioni, anche delle funzioni di filtro sull'immagine visualizzata. Uno dei filtri più interessanti applicati al software di visualizzazione del *Vercelli Book* è il rilevamento laplaciano. Questo filtro rileva i bordi nell'immagine utilizzando il metodo laplaciano che produce bordi sottili larghi un pixel creando l'effetto mostrato nella Figura 45.

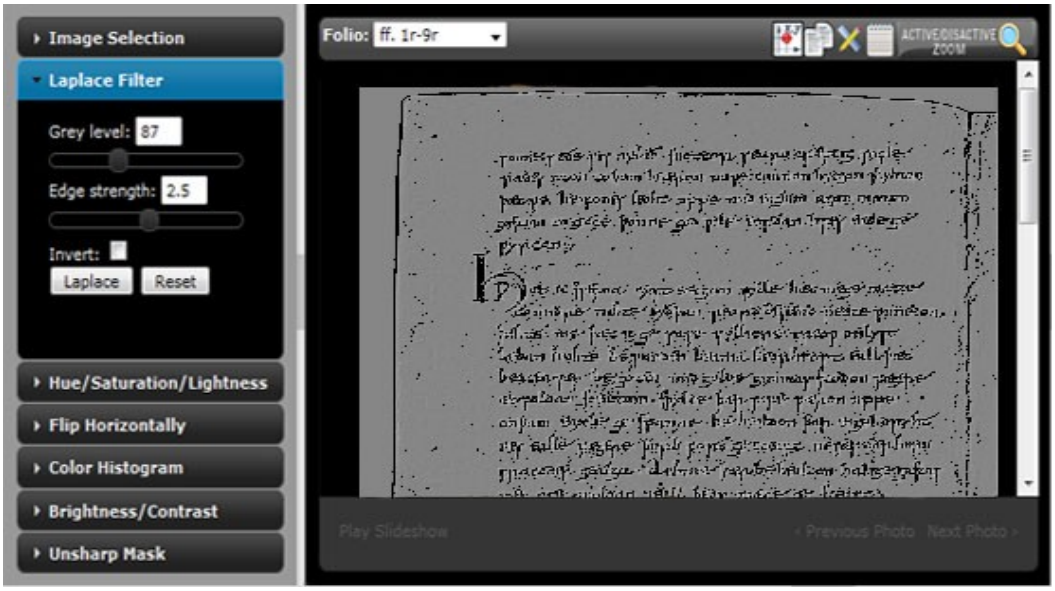


Figura 45 – Attivazione del filtro Laplace

- *Hue/Saturation/Lightnetss.* I parametri inizializzati sono *hue,* con valori da -180 a 180, *saturation* i cui valori sono compresi tra -100 a 100 e *lightness,* con valori impostati come il parametro di saturazione.
- *Flip Horizontally.* Questa azione inverte semplicemente l'immagine orizzontalmente.
- *Color Histogram.* Consta di un convertitore da RGB a scala di grigi, il comando *paint* disegna l'istogramma sull'immagine.
- *Brightness/Contrast.* Questa azione consente di regolare la luminosità e il contrasto dell'immagine. Ci vogliono tre parametri, uno per la luminosità, il contrasto e e uno per uno per selezionare la modalità "legacy". Se la modalità legacy è selezionata, la luminosità è regolata allo stesso modo di Photoshop. Altrimenti, ogni pixel viene moltiplicato per il fattore brightness_value / 150.
- *Unsharp Mask.* Questa funzione presenta l'effetto di contrasto con valore compreso tra 0 e 500, il raggio di sfocatura e una maschera di contrasto con valore compreso tra 0 e 5. La soglia dell'effetto di contrasto è modellabile su un valore compreso tra 0 e 255. La Figura 46 presenta una porzione del manoscritto prima e dopo l'applicazione della maschera.

Figura 46 – Porzione del manoscritto prima e dopo l'applicazione della maschera

Passiamo infine alla pagina di visualizzazione testo/testo (Figura 42c):

- *Pulsante Homelies.* Raccoglie l'elenco delle Omelie (23).
- *Pulsante Poems* . Raccoglie l'elenco dei poemi (6).
- *Pulsante The VB Texts.* Area di testo che presenta un breve commento sui testi del *Vercelli Book.*

4.5. Risultato finale

Le immagini che seguono mostrano in anteprima il risultato finale del progetto presentato secondo le modalità di visualizzazione immagine/testo (Figura 47a), immagine/immagine (Figura 47b) e testo/testo (Figura 47c) secondo la modalità standard. L'accesso all'applicazione avviene attraverso una semplice pagina di presentazione grafica, tutta la navigazione e gli strumenti del software invece vengono presentati tramite un'unica vista principale, così strutturata:

- la modalità immagine/testo è considerata la modalità di *default* dell'applicazione, quella disponibile al momento dell'avvio;
- la modalità immagine/immagine presenta due immagini affiancate, utile per manoscritti con varie riproduzioni di una stessa immagine, legate magari a qualche particolare tipo di elaborazione;

- la modalità testo/testo mostra due testi affiancati, utile nel caso dei manoscritti per presentare edizioni o lezioni diversi dello stesso testo o la relativa traduzione.

La struttura dell'interfaccia grafica cambia a seconda delle esigenze dell'utente: tutti i pannelli utilizzati possono essere collassati o aperti a tutto schermo così che l'utente visualizza i contenuti utilizzando lo spazio che ritiene più consono alle sue esigenze.

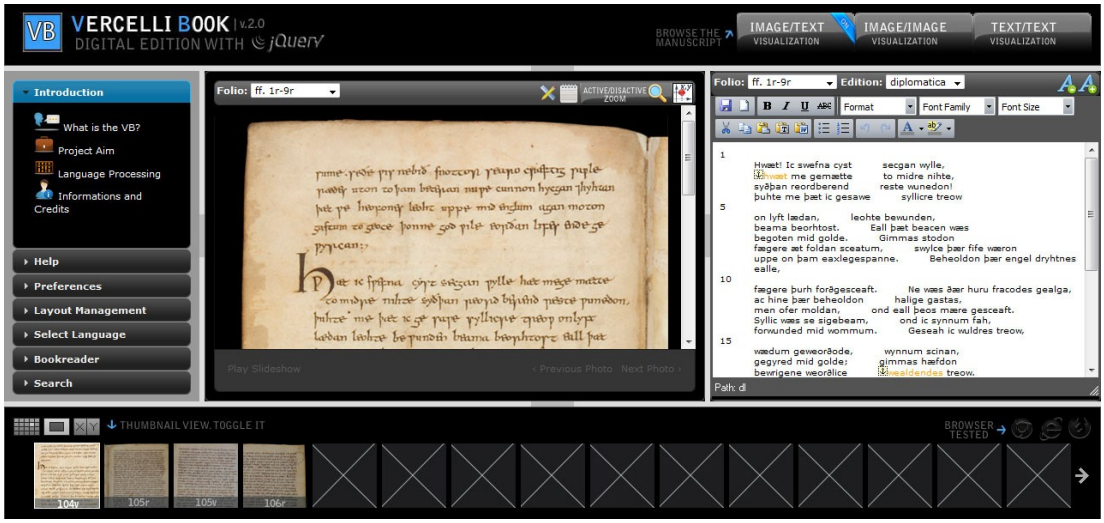


Figura 47a– Screenshot relativo alla modalità di visualizzazione immagine-testo

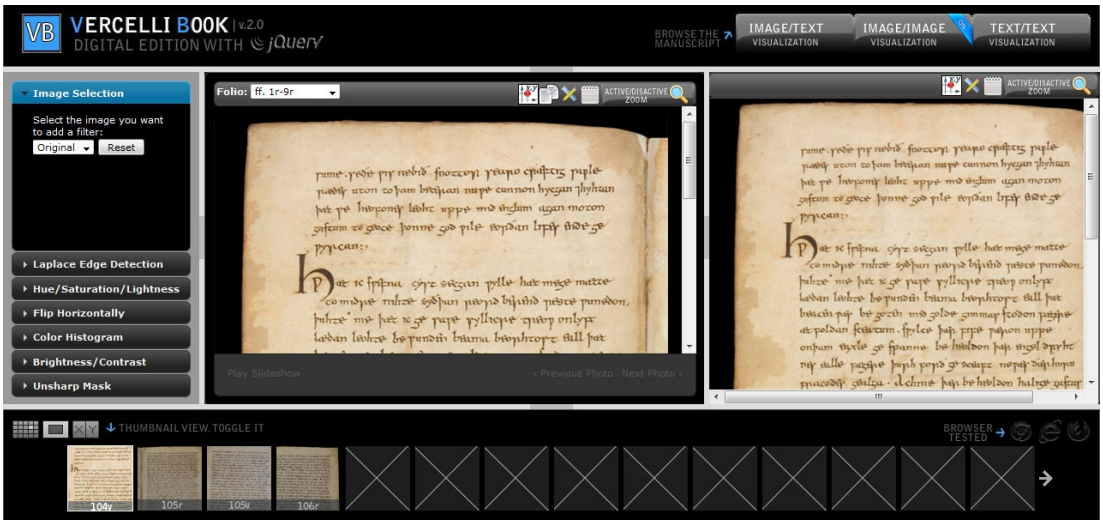


Figura 47b– Screenshot relativo alla modalità di visualizzazione immagine-immagine

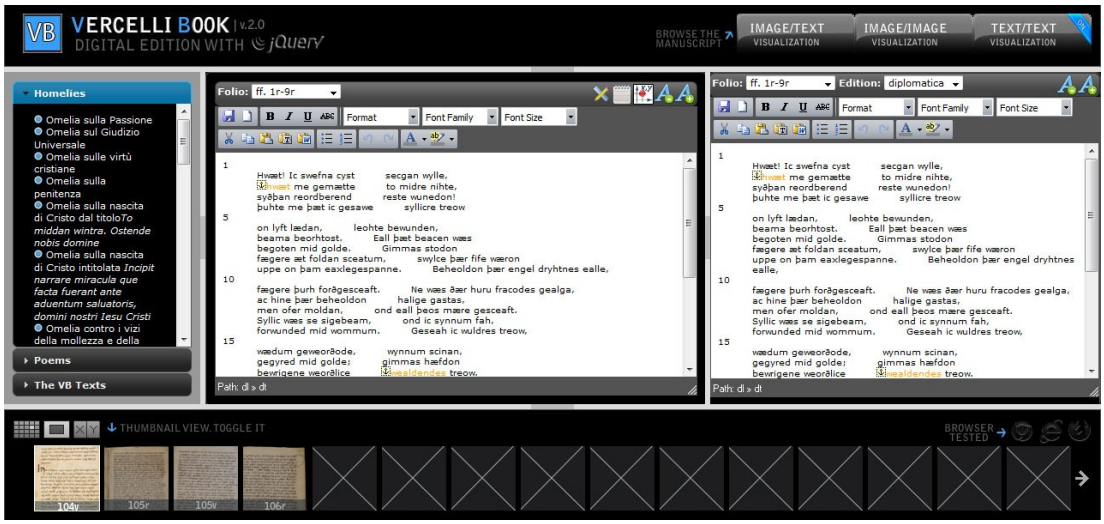


Figura 47c – Screenshot relativo alla modalità di visualizzazione testo-testo

4.5.1 Browser supportati

Il software è stato progettato per un utilizzo ottimale con i seguenti browser:

1. Internet Explorer 9
2. Firefox 2.x / 3.x
3. Safari 3.1
4. Opera 9.5
5. Google Chrome

In caso di utilizzo del software con versioni 6 e 7 di Internet Explorer si possono verificare errori. Gli altri standard utilizzati sono stati XHTML 1.0 per la realizzazione delle pagine web, XML 1.0 per la codifica dei testi, CSS per le impostazioni grafiche delle pagine. Per quanto riguarda i formati di memorizzazione delle immagini sono stati utilizzati i formati più comuni, evitando accuratamente ogni formato proprietario: JPEG, GIF e PNG. Per quanto riguarda invece la parte di programmazione è stato utilizzato il linguaggio Javascript, le teconolgie DHTML che utilizzano lo standard W3C DOM, *Document Object Model* e la libreria jQuery, compatibile con tutti i maggiori browser web.

CAPITOLO V

CONCLUSIONI

5.3 Un software *Open Source*

Il software e l'interfaccia grafica utilizzati nella realizzazione dell' EVT sono state realizzate tenendo presente le indicazioni riguardanti gli standard internazionali del W3C, i vantaggi e le disposizioni per utilizzare e creare codice *Open Source* e le regole di accessibilità.

I programmi utilizzati sono tutti Open Source, ma è bene fare una precisazione al riguard visto che spesso per definire questi software si usa anche l'espressione *free software. I*n realtà *open source* e *free software* non sono sinonimi, al riguardo esistono due organismi, la "*Open Source Initiative*[127]" (OSI) e la "*Free Software Foundation*[128]" (FSF). Le due definizioni sono simili ma nascono da due considerazioni opposte: nel caso della OSI infatti la motivazione è di carattere tecnico-economico difatti l'attenzione è posta sui vantaggi pratici che derivano dal rendere accessibile il codice sorgente; la FSF invece ha sostenuto una motivazione a carattere etico: la libertà e il diritto degli utenti che usufruiscono di un software sono motivi che precedono qualsiasi ragione economica e tecnica.

Anche il software da me sviluppato è Open Source e, per consentire un futuro lavoro di integrazione e riprogettazione del codice, ho deciso di commentare ogni funzionalità messa a punto personalmente anche la parte di codice che riguarda l'utilizzo della libreria jQuery. Il progetto è aperto alla collaborazione con altri studenti e studiosi e rientra fra le attività del LCD. Verrà distribuito su uno dei siti che raccolgono software Open Source come Sourceforge[129].

[127] Documento disponibile all'indirizzo http://www.Open Source.org/docs/definition.php.
[128] Sito Web ufficiale: http://www.fsf.org/.

Creare un software Open Source permette numerosi vantaggi. In particolar modo esso consente:

- la disponibilità di accesso ai codici sorgenti;
- la possibilità di apportare continui miglioramenti al programma attraverso il lavoro delle comunità di sviluppatori;
- una garanzia di sicurezza. Un codice Open Source è garanzia di trasparenza dal momento che è costantemente sottoposto a verifica da parte di migliaia di utenti. Per questo è più semplice quindi individuare e correggere gli errori;
- una libertà di sviluppo. Il futuro di un programma non è legato all'azienda che ne detiene il controllo, ma dipende dal lavoro e dal dialogo tra sviluppatori, utenti, aziende e istituzioni.

5.2 Uno sguardo al futuro

Quando parliamo di "standard" del Web, intendiamo: linguaggi strutturali (XHTML e XML), linguaggi di presentazione (CSS) e modelli a oggetto (DOM).
Naturalmente gli standard in informatica non specificano soltanto lo scambio di informazioni sul Web. Altri standard importanti servono per la codifica dei caratteri e i formati di memorizzazione delle immagini. Questi ultimi servono a garantire l'interoperabilità (PNG, JPEG,TIFF). Quanto invece agli standard UNICODE[130] e formati di codifica come ome UTF-8 e UTF-16 sono stati definiti per ovviare al problema della gestione dei 32 bit.
Progettare e costruire secondo questi standard è più semplice in quanto riduce drasticamente i costi di produzione e consente di realizzare siti accessibili ad un maggior numero di persone. Restano tuttavia ancora delle questioni da risolvere nell'ambito della programmazione di un'edizione digitale:

- Come influiscono le Edizioni Digitali sui livelli di apprendibilità?
- È possibile migliorare l'accesso ai documenti digitali?

[129] La documentazione ufficiale di Sourceforge è reperibile al seguente indirizzo: http://sourceforge.net/.
[130] Standard UNICODE: http://www.unicode.org.

- Esistono delle metodologie tradizionali che non possono essere potenziate dalle tecnologie digitali?

5.3 Conclusioni

Lo scopo di questa tesi è stato quello di definire uno strumento basato sul Web per creare e modificare edizioni elettroniche basate su immagini e archivi digitali di testi umanistici.

Il software e l'interfaccia grafica stati realizzati tenendo ben presente le indicazioni riguardanti gli standard internazionali del W3C, i vantaggi e le disposizioni per utilizzare e creare codice Open Source e le regole di accessibilità.

Progettare e implementare una buona interfaccia non è un lavoro banale. Ci vuole una buona dose di creatività, un approccio strutturato alla progettazione grafica, la comprensione della capacità umana, la conoscenza di principi di progettazione noti, ma anche il coraggio di rompere le regole quando necessario. Ancora più importante, per diventare un buon progettista è l'esperienza.

Le difficoltà incontrate non sono state poche, visto che comunque mi sono confrontata con una realtà a me poco familiare: ci sono state pertanto difficoltà nella fase di progettazione (dovendo mettere a punto un modello che potesse soddisfare il più possibile le esigenze esposte nei requisiti tecnici[131]) e nella fase di implementazione perché ho dovuto acquisire nozioni che non possedevo. Questo progetto mi ha permesso anche di approfondire le mie conoscenze informatiche, accostandomi alla programmazione in jQuery in maniera professionale. Considero quindi questo lavoro un valido esempio di applicazione pratica delle conoscenze apprese durante il mio corso di studi.

[131]Si veda il paragrrafo 3.2 "Fase I. Requisiti Tecnici".

BIBLIOGRAFIA

Framework JavaScript e Interfacce GUI

- Orchard M.L, Pehlivanian A, Koon S, Jones H. (2009), *Professional JavaScript frameworks : Prototype, YUI, Ext JS, Dojo and MooTools* WROX, Indianapolis.
- Steven M. Schafer. (2005), *Web standards programmer's reference : HTML, CSS, JavaScript, Perl, Python, and PHP,* WROX, Indianapolis, O'Reilly.
- Tidwell, J. (2011), *Designing interfaces*, 2nd ed, O'Reilly.
- Nielsen, J. *Ten Usability Heuristics*.
- Tognazzini, B. (2003), *First Principles of Interaction Design Retrieved*
- Scott B., Neil T. (Jan 26, 2009), *Designing Web Interfaces: Principles and Patterns for Rich Interactions,* O'Reilly.
- Rosselli Del Turco, R. (2011), "After the editing is done: designing a Graphic User Interface for Digital Editions".

Codifica dei Testi

- Ciotti F., (2005), *Il manuale TEI Lite – Introduzione alla codifica elettronica dei testi letterari*, Edizioni Sylvestre Bonnard.
- Pierazzo E., (2005), *La codifica dei testi*, Roma, Carocci Editore.

Electronic Edition

- Hockey S., *Electronic Texts in the Humanities*, Oxford UP, Oxford, 2000.
- Muir, B.J (2004), *The Exeter anthology of Old English poetry: an edition of Exeter Dean and Chapter MS 3501 (Exeter Medieval English Texts and Studies),* Edition Exeter: Exeter University Press, 2004.

- Kiernan K.S., *Digital Facsimiles in Editing: Some Guidelines for Editors of Image-based Scholarly Editions*. In John Unsworth, Katherine O'Brien O'Keeffe, and Lou Burnard (ed.) Electronic Textual Editing, 2007.

Documentazione Ext-JS

- Shea F., Ramsay C., Blades S., (2008), *Learning Ext JS : build dynamic, desktop-style user interfaces for your data-driven web applications,* Birmingham, PACKT Publishing.
- García J., (2010), *Ext JS in action,* London, Pearson Education.
- Ramon, Jorge, (2009), *Ext JS 3.0 cookbook 109 great recipes for building impressive Rich Internet Applications using the Ext JS JavaScript library,* Birmingham, PACKT Publishing.

Documentazione JavaScript

- Zakas, N., (2009), *Professional JavaScript for web developers*, 2nd ed, Indianapolis, Wiley Publishing.
- Nikolassy R., (2007), *Manuale di Javascript*, Hoepli.

Documentazione jQuery

- Lindley, C., (2010), *jQuery Cookbook: Solutions & Examples for jQuery Developers,* O'Reilly.
- Bibeault, B., Yehuda Katz Y., (2010), *jQuery in action*, 2nd ed, Greenwich, Manning Edition.
- Harwani, B.M, (2010), *jQuery recipes,* New York, Apress Edition.
- Tidwell, J., (2011), *Designing interfaces*, 2nd ed, Sebastopol, O'Reilly.
- Wellman, D., (2009), *jQuery UI 1.7: The User Interface Library for jQuery*, PACKT Publishing.

WEBLIOGRAFIA

Codifica Digitale dei Testi

- Association for Computers and the Humanities
  http://www.ach.org/
- Association for Computational Linguistics
  http://www.aclweb.org/
- Collaboratory for Research in Computing for Humanities
  http://www.rch.uky.edu/
- Association for Literary and Linguistic Computing
  http://www.allc.org/
- Humanities Research Institute
  http://www.shef.ac.uk/hri/projects/projectpages/accessgrid.html
- TEI: P5 Guidelines
  http://www.tei-c.org/Guidelines/P5/
- TEI: Text Encoding Initiative
  http://www.tei-c.org/index.xml

User Interface Design

- GNOME 2.0 Human Interface Guidelines
  http://developer.gnome.org/projects/gup/
- Smith S.L, Jane N. Mosier N.J., (1986), *Guidelines for Designing User Interface Software*, http://hcibib.org/sam/
- HCI Bibliography : Human-Computer Interaction Resources
  http://hcibib.org/
- *Nielsen's Usability Heuristics Spolsky (2001).*
  http://www.useit.com/papers/heuristic/heuristic_list.html

- KDE Usability Project

  http://techbase.kde.org/Projects/Usability

- *Mac OS X Human Interface Guidelines*
  http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHI

- De Souza F., Bevan N., (1990), *The Use of Guidelines in Menu Interface Design,* http://www.usability.serco.com/papers/useofguidelines.pdf

- *Mac OS X Human Interface Guidelines*
  http://developer.apple.com/documentation/UserExperience/Conceptual/OSXHI

Vercelli Book e altri progetti

- *Vercelli Book Digitale*
  http://islp.di.unipi.it/bifrost/vbd/
- Francesca Fiorentini, *EVT: Edition Visualization Technology; Progettazione e sviluppo un software per la consultazione di edizioni digitali.*
- http://etd.adm.unipi.it/theses/available/etd-09132008-133733/
- Alessandra Giammalva, *Flos - Edizione digitale.*
  http://iu.di.unipi.it/flos/
- Canterbury Tales Project
  http://www.canterburytalesproject.org/
- Collaboratory for Research in Computing for Humanities
  http://www.rch.uky.edu/
- Digital Scriptorium
  http://www.scriptorium.columbia.edu/
- Edition Production & Presentation Technology (EPPT)
  http://www.eppt.org/eppt-trial/EPPT-TrialProjects.htm
- Fondazione Museo del Tesoro del Duomo e Archivio Capitolare

http://www.tesorodelduomovc.it/

- HRI Online: Humanities Research Institute
  http://www.hrionline.ac.uk/hrionline/index.php
- ITSEE: Projects
  http://www.itsee.bham.ac.uk/projects.htm
- Electronic Beowulf: Technology and Editing Methods
  http://beowulf.engl.uky.edu/~kiernan/eBoethius/edit.htm
- Scholarly Digital Editions
  http://www.sd-editions.com/
- Institute for Textual Scholarship and Electronic Editing
  http://www.birmingham.ac.uk/research/activity/.../index.aspx
- TEI Viewer Project
  http://teiviewer.org/

Framework JavaScript e Interfacce GUI

- Don Norman's *Gestural Interfaces: A Step Backwards In Usability*
  http://dl.acm.org/citation.cfm?id=1836228
- UKOLN, (2006) *Good Practice Guide for Developers of Cultural Heritage Web Services* http://www.ukoln.ac.uk/interop-focus/gpg/.
- Kenney A.R, Rieger O.Y., (2000), *Moving Theory into Practice: Digital Imaging for Libraries and Archives*
  http://www.dlib.org/dlib/july00/07bookreview.html
- Northeast Document Conservation Center, (2000), *Handbook for Digital Projects: A Management Tool for Preservation & Access*
  http://nedcc.org/oldnedccsite/digital/dighome.htm.
- Arts and Humanities Data Service (AHDS), *Guides to Good Practice* website
  http://www.ahds.ac.uk/creating/guides/index.htm.
- Washington State Library, *Digital Best Practices* website
  http://digitalwa.statelib.wa.gov/newsite/best.htm.

- Schreibman S., (2007), *Best Practice Guidelines for Digital Collections at University of Maryland Libraries.* http://www.lib.umd.edu/dcr/publication
- Snap-Together Visualization http://www.cs.umd.edu/hcil/snap/
- UDL Spotlight http://udlspotlight.wordpress.com/
- O'Donnell D.P., *Disciplinary Impact and Technological Obsolescence in Digital Medieval Studies* http://www.digitalhumanities.org/companion/view?docId=blackwell/9781405148641/9781405148641.xml&chunk.id=ss1-4-2&toc.id=0&brand=9781405148641_brand
- Wardrip-Fruin N., *Reading Digital Literature: Surface, Data, Interaction, and Expressive Processing* http://www.digitalhumanities.org/companion/view?docId=blackwell/9781405148641/9781405148641.xml&chunk.id=ss1-5-2&toc.depth=1&toc.id=ss1-5-2&brand=9781405148641_brand

Documentazione Ext-JS

- Client-side JavaScript Application Framework http://www.sencha.com/products/extjs/
- Exjts Tutorial http://www.extjstutorial.com/
- Introduzione a Ext.js http://javascript.html.it/articoli/leggi/2640/introduzione-a-extjs/3/
- Sencha - Desktop and Mobile JavaScript Frameworks http://www.sencha.com/
- Sencha Forum http://www.sencha.com/forum/

Progetti realizzati con exist-DB

- Carl-Maria-von-Weber-Gesamtausgabe (WeGA) The correspondence, diaries, writings and works of Carl Maria von Weber
  http://www.weber-gesamtausgabe.de/
- Diary of Robert Graves
  http://graves.uvic.ca/
- OperaLiber
  http://193.204.255.27/operaliber/index.php?page=/operaLiber/home
- The Anglo-Norman Dictionary
  http://www.anglo-norman.net/
- Early Americas Digital Archive
  http://mith2.umd.edu/eada/
- VNS letters online: digital edition of letters concerning the Belgian literary journal 'Van nu en Straks'
  http://vnsletters.org/VNS/
- Corpus Pieter Willems: searchable facsimile edition of the first Dutch dialect survey ever
  http://ctb.kantl.be/corpora/CPWNL/
- Sermones.net : éditions électroniques de sermons latins médiévaux
  http://sermones.net
- Interactive Album of Mediaeval Palaeography: website for training in practical palaeographical skills.
  http://ciham.ish-lyon.cnrs.fr/paleographie/index.php?l=en
- Colonial Despatches: the colonial despatches of Vancouver Island and British Columbia 1846-1871.
  http://bcgenesis.uvic.ca/
- Le mariage sous L'Ancien Régime
  http://mariage.uvic.ca/

- Vienna-Oxford International Corpus of English: a corpus of transcripts of spoken ELF interactions in TEI format.
- http://voice.univie.ac.at/

Documentazione jQuery

- jQuery: The Write Less, Do More, JavaScript Library
  http://jquery.com/
- Improve Your Web App With jQuery
  http://www.scribd.com/doc/34935690/45-Improve-Your-Web-App-With-jQuery
- jQuery Labs -Find fast, do more!
  http://www.jquerylabs.com/
- jQuery UI - Demos & Documentation
  http://jqueryui.com/demos/
- jQuery UI
  http://jqueryui.com/
- jQuery UI – ThemeRoller
  http://jqueryui.com/themeroller/
- jQuery virtual tour
  http://www.openstudio.fr/jquery-virtual-tour/salleformation.html/
- PageTurner with jQuery
  http://www.sitepoint.com/examples/jquery/animate4.php
- Project types: jQuery Plugins
  http://plugins.jquery.com/
- jQuery Italia: La community italiana dedicata a jQuery
  http://www.jqueryitalia.org/

Edizioni Digitali di Manoscritti

- CEEC - Codices Electronici Ecclesiae Coloniensis
  http://www.ceec.uni-koeln.de/
- Codex Sinaiticus
  http://codexsinaiticus.org/
- e-codices Project
  http://www.e-codices.unifr.ch/
- Galileo at Work: His Complete Notes on Motion in an Electronic Representation
  http://www.mpiwg-berlin.mpg.de/texts/Galileo.Nuncius.html
- Gallica Digital Library
  http://gallica.bnf.fr/
- Heliand Project
  http://venus.unive.it/mbuzzoni/heliand.html
- Internet Archive BookReader
  http://openlibrary.org/dev/docs/bookreader
- Internet Archive Digital Library
  http://www.archive.org/index.php
- John Dickinson Writings Project
  http://dickinsonproject.rch.uky.edu/funding.php
- Most viewed virtual books
  http://www.bl.uk/onlinegallery/virtualbooks/viewmostviewed/index.html
- Old Bailey Online - The Proceedings of the Old Bailey
  http://www.oldbaileyonline.org/
- TILE | Text-Image Linking Environment
  http://mith.info/tile/PageTurner
- Panoply – Manuscripts
  http://cbers.shef.ac.uk/manuscripts/index.html
- St Gall Priscian glosses

http://www.stgallpriscian.ie/

- The Edgar Allan Poe Digital Collection
  http://research.hrc.utexas.edu/poedc/
- The Papers of George Washington
  http://gwpapers.virginia.edu/
- The Shakespeare Quartos Archive
  http://www.quartos.org/index.html
- The Domesday Project
  http://www.atsf.co.uk/dottext/domesday.html
- Virtual Vellum
  http://www.shef.ac.uk/hri/projects/projectpages/virtualvellum.html
- Vivarium: the online digital collections of Saint Joh's Univesity and the College of Saint Benedict
  http://cdm.csbsju.edu/cdm4/item_viewer.php
- State Papers Online
  http://gale.cengage.co.uk/state-papers-online-15091714/part-ii.aspx
- Vetus Latina Iohannes - Department of Theology and Religion
  http://www.birmingham.ac.uk/schools/ptr/departments/theologyandreligion/research/projects/vetus-latina-iohannes.aspx
- Digital MappaeMundi
  http://bob.drew.edu/mappaemundi/
- NINES
  http://www.nines.org/
- NT.Virtual Manuscript Room
  http://intf.uni-muenster.de/vmr/NTVMR/viewer/viewerCodex01.php

Progetti ImaP

- IMaP Demonstration
  http://lettuce.tapor.uvic.ca/~taprimap/imapdemo/demo/

- King County ImaP
  http://www5.kingcounty.gov/iMAP/viewer.htm
- The Map of Early Modern London
  http://mapoflondon.uvic.ca/
- p.mapper
  http://www.pmapper.net/demo.shtml

Edition Production & Presentation Technology (EPPT)

- Canterbury Tales Project
  http://www.canterburytalesproject.org/
- CDLI - Cuneiform Digital Library Initiative
  http://cdli.ucla.edu/
- Roman de la Rose
  http://romandelarose.org/
- The Electronic Boethius Project
  http://beowulf.engl.uky.edu/~kiernan/eBoethius/inlad.htm
- Richard Rawlinson Center Medieval Institute
  http://www.wmich.edu/medieval/research/rawl/
- Fine Rolls Henry III
  http://www.finerollshenry3.org.uk/

Edizioni Digitali disponibili su CD-ROM

- EFT : Electronic Facsimiles & Texts
  http://beowulf.engl.uky.edu/
- Evellum
  http://www.evellum.com/index.php
- The Miller's tale
  http://www.sd-editions.com/AnaAdditional/millerEx/images/millerhome.html

- Computers & Texts 12: Frase
  http://users.ox.ac.uk/~ctitext2/publish/comtxt/ct12/fraser.html
- *A Digital Catalogue of the Canterbury Tales*
  http://www.sd-editions.com/AnaAdditional/CTPCATex/CTPCATexhome.html
- DantÈ s *Commedia*
  http://www.sd-editions.com/AnaAdditional/CommediaEx/CommediaExhome.html
- DantÈ s Monarchia
  http://www.sd-editions.com/Monarchia/index.html
- Nun's Priest's Tale on
  http://www.sd-editions.com/AnaAdditional/NPEx/index.html
- Parliament Rolls of Medieval England
  http://www.sd-editions.com/AnaAdditional/PROMEeg/Images/home.html
- Hengwrt Chaucer Digital Facsimile
  http://www.sd-editions.com/AnaAdditional/HengwrtEx/images/hgopen.html

Riviste e Pubblicazioni

- DHQ: Digital Humanities Quarterly: e-Science for Medievalists
  http://digitalhumanities.org/dhq/vol/3/4/000071/000071.html
- Digital Medievalist
  http://www.digitalmedievalist.org/
- Electronic Medievalia *O Captain! My Captain! Using Technology to Guide Readers Through an Electronic Edition*
  http://www.heroicage.org/issues/8/em.html
- Emil Iacob's Publications
  http://www.eppt.org/~emil/

Software utilizzati

- Adobe Photoshop CS5 Extended
  http://www.adobe.com/products/photoshopextended.html
- Dropbox - Simplify your life
  https://www.dropbox.com/
- Feng-GUI: Attention Analysis for Websites and Advertisements
  http://www.feng-gui.com/
- Free Software Foundation: working together for free software
  http://www.fsf.org/
- Aptana Studio
  http://www.aptana.com/
- Pencil Project
  http://pencil.evolus.vn/en-US/Home.aspx
- JSTOR
  http://www.jstor.org/
- Sencha - Desktop and Mobile JavaScript Frameworks
  http://www.sencha.com/
- The UVic Image Markup Tool Project
  http://www.tapor.uvic.ca/~mholmes/image_markup/index.php
- W3C Document Object Model
  http://www.w3.org/DOM/
- Adobe Dreamweaver CS5.5
  http://www.adobe.com/products/dreamweaver.html
- WorldCat
  http://www.worldcat.org/
- DOM – Document Object Model
  http://www.w3.org/DOM/
- Zotero
  http://www.zotero.org/

- jsFiddle: Online Editor for the Web
  http://jsfiddle.net/
- HTML KIT: making HTML coding fun again
  http://www.htmlkit.com/

Software consultati

- Adobe Digital Editions
  http://www.adobe.com/products/digitaleditions/
- AJAX-ZOOM jQuery Image Zoom Viewer
  http://www.ajax-zoom.com/index.php
- Book Flip - JavaScript only
  http://www.coastworx.com/bookflip.php
- FlippingBook - software for creating online publications, magazines, photo albums and flip books with the real page turning effect
  http://page-flip.com/
- FlipViewer.com
  http://www.flipviewer.com/
- Juxta
  http://www.juxtasoftware.org/
- MegaZine3 - Open Source PageFlip
  http://www.megazine3.de/

Standard, Open Source e Accessibilità

- Mission: Open Source Initiative
  http://www.Open Source.org/
- Unicode Consortium
  http://www.unicode.org/
- Web Content Accessibility Guidelines (WCAG) 2.0
  http://www.w3.org/TR/WCAG/

ELENCO DELLE FIGURE

ELENCO DELLE TABELLE

RINGRAZIAMENTI

La mia gratitudine va al mio relatore di tesi, il Dott. Roberto Rosselli Del Turco, costante guida durante lo sviluppo di questo progetto. La sua gentilezza, pazienza e incoraggiamento sono stati una grande risorsa durante la fase di elaborazione e stesura di questo elaborato finale. Ha inoltre il mio più sincero ringraziamento per avermi assistito durante alcuni dei momenti più difficili della mia vita e della stesura di questa tesi.

* * *

Ringrazio la mia correlatrice Dott.ssa Maria Simi per la comprensione e il tempo concessomi nel corso di questa tesi e del mio percorso di studi.

* * *

Sono anche in debito con mia madre, mio padre e mia sorella, fonte costante di amore e di ispirazione continua per tutta la vita - hanno sempre sostenuto le mie aspirazioni e non sarei la donna che sono oggi senza di loro. Li ringrazio per tutto quello che hanno fatto per me.

* * *

Infine vorrei ringraziare Chiara, Letizia, Michele e Alessandra per il loro amore e la felicità e la gioia che portano nella mia vita, per me sempre incoraggianti. Il sostegno della mia famiglia e degli amici hanno contribuito a rendere possibile questa tesi. Voglio dunque estendere il mio ringraziamento a tutti loro.