



UNIVERSITÀ DI PISA

FACOLTÀ DI LETTERE E FILOSOFIA

Corso di Laurea in Informatica Umanistica

TESI DI LAUREA

Bibliografia online dell'*Edda* di Snorri

**Collaborazione al progetto, creazione del sito web e
sviluppo dell'interfaccia utente**

Relatore

Prof. Roberto Rosselli Del Turco

Candidato

Alessandra Spadaccia

Anno Accademico 2011/2012

INDICE

Introduzione	2
I – L’EDDA DI SNORRI	4
1.1 – Caratteristiche generali e struttura dell’opera	4
1.2 – Ricezione e fortuna.....	6
1.3 – Bibliografia dell’ <i>Edda</i> di Snorri.....	7
1.3.1 – <i>Problemi specifici di una bibliografia dell’Edda di Snorri</i>	10
1.3.2 – <i>La soluzione: una bibliografia online</i>	11
II – IL PROGETTO “ONLINE BIBLIOGRAPHY OF SNORRA EDDA”	13
2.1 – Il progetto	13
2.2 – Il software <i>OpenBMS</i>	14
2.3 – Lo sviluppo di <i>OpenBMS</i>	15
2.3.1 – <i>Costituzione del database</i>	16
2.3.2 – <i>Implementazione e gestione del database</i>	17
2.3.3 – <i>L’interfaccia utente</i>	22
III – LA COLLABORAZIONE CON IL PROGETTO	28
3.1 – Il sito web della “Bibliografia online dell’ <i>Edda</i> di Snorri”.....	28
3.2 – Metodologie, strumenti e contesto organizzativo.....	35
3.3 – L’evoluzione dell’interfaccia utente.....	36
3.3.1 – <i>L’interfaccia versione 0.8.0</i>	37
3.3.2 – <i>L’interfaccia versione 2.0</i>	47
3.4 – Debug della GUI e del back-end	51
IV – IL FUTURO DEL PROGETTO	54
4.1 – Sviluppi futuri e obiettivi a breve e lungo termine.....	54
V – CONCLUSIONI	55
Bibliografia	58
Sitografia	59

INTRODUZIONE

Questa tesi è stata scritta per illustrare com'è nata e si è evoluta la collaborazione con il progetto "Online Bibliography of Snorra *Edda*" iniziata durante il tirocinio e maturata poi in un progetto di tesi.

Il progetto, avviato nel 2008 su impulso della Prof.ssa Adele Cipolla dell'Università di Verona è nato con l'obiettivo di creare un database bibliografico online dell'*Edda* di Snorri che fosse esaustivo e fruibile dalla comunità accademica per rispondere a quelle necessità di ricerca di materiale specifico a cui una bibliografia "tradizionale" non avrebbe potuto far fronte. La soluzione ottimale è stata individuata nella creazione di un nuovo tipo di database relazionale e nella realizzazione di una scheda bibliografica molto più ricca di dettagli rispetto a quelle realizzabili con i software già presenti sul mercato. Il tutto è stato fatto creando un software *ex-novo* che ha permesso così di catalogare ogni singola entrata bibliografica oltre che per le sue caratteristiche, anche per le sue relazioni con il resto del materiale presente.

Il software sviluppato ha integrato un nuovo motore di ricerca, denominato OpenBMS (Open Bibliography Management System), che è già operativo e che include sia un'interfaccia di ricerca, sulla quale mi sono concentrata in modo particolare, sia un modulo per l'inserimento di nuovi dati direttamente dal web.

STRUTTURA DELLA TESI

La tesi prevede una parte introduttiva in cui si spiega cosa sia L'*Edda* di Snorri e quale sia la sua importanza nell'ambito letterario, ponendo l'accento sulla sua vastissima bibliografia, su quali siano i problemi specifici che s'incontrano nella stesura della stessa e sulle soluzioni adottate (Capitolo 1).

Dopo questa parte si illustra com'è stato concretamente concepito il progetto e realizzato il software: si chiarisce quali sono state le scelte di programmazione e le sue fasi. A seguire si mostrano i risultati raggiunti al momento dell'inizio della mia collaborazione (Capitolo 2).

Il terzo capitolo descrive quali sono state le fasi della mia collaborazione, in quali parti del progetto sono intervenuta e in che misura. È descritto il modus operandi e quali sono stati gli obiettivi che ho cercato di raggiungere (Capitolo 3).

Nella penultima sezione è descritto l'attuale stato del progetto, quali sono gli obiettivi attesi a breve e lungo termine, e quali sono gli sviluppi futuri che sono previsti per la bibliografia online (Capitolo 4).

L'ultima parte della tesi raccoglie alcune considerazioni di carattere generale e personale maturate durante e dopo la realizzazione del progetto (Capitolo 5).

L'EDDA DI SNORRI

1.1 – Caratteristiche generali e struttura dell'opera

I testi che portano il nome di *Edda* all'interno della letteratura islandese medievale sono due, legati da un rapporto d'interdipendenza stretta. Si tratta infatti dell'*Edda di Snorri*, o *Edda in prosa*, e dell'*Edda di Sæmundr*, detta anche *Edda poetica* o *Edda antica*, così chiamata dallo scopritore del manoscritto, il vescovo Brynjólfur Sveinsson, perché attribuita a Sæmundr Sigfússon e considerata come una delle fonti usate da Snorri per la composizione della “sua” *Edda*.

L'*Edda* di Snorri è un manoscritto medievale attribuito a Snorri Sturluson, famoso storico, studioso e poeta islandese vissuto tra il XII e il XIII secolo. L'*Edda*, nome non ancora del tutto chiarito, ma che si pensa derivante o dal legame con Oddi, il paese di formazione di Snorri (con *Edda* che significherebbe quindi: “Libro di Oddi”) o per l'accostamento al sostantivo norreno *óðr* “poesia”¹, è un manuale di arte scaldica per aspiranti poeti e, in secondo luogo, un trattato di mitologia norrena. L'opera è un testo composito, formato da un prologo (*Formáli*) e da tre parti principali: il *Gylfaginning*, “l'Inganno di Gylfi”, lo *Skáldskaparmál*, il “Discorso sull'arte scaldica” e l'*Háttatal*, il “Catalogo dei metri”.

Il primo libro, il *Gylfaginning* è una narrazione organica dei principali miti nordici. Cornice della narrazione è il viaggio compiuto da Gylfi, mitico re svedese il quale si reca nel paese degli *Æsir* per scoprire la fonte della loro sapienza e del loro potere. La materia del libro si dipana attraverso le domande di Gylfi sulla creazione e sul divenire dell'universo, sulla natura degli dei e sulla fine del mondo, a cui danno risposta tre misteriosi personaggi nei quali si può identificare lo stesso Óðinn (Odino) sotto tre aspetti diversi. Snorri dà una visione d'insieme delle credenze pagane, racconta i miti di creazione e svela la struttura dell'universo e dei mondi che lo compongono, enumera gli dei e di ciascuno fornisce una descrizione e un

¹ Cfr. Faulkes, *Snorri Sturluson: Edda*, Introduzione, xvii.
Cfr. Isnardi, *Snorri Sturluson: Edda di Snorri*.

resoconto dei principali miti che lo riguardano. Questo perché gli antichi miti pagani erano considerati parte dell'irrinunciabile bagaglio culturale di ogni poeta, materiale propedeutico alla comprensione della poesia scaldica e repertorio necessario per la composizione di nuove opere. Tuttavia, nella prospettiva del lettore moderno, la parte mitologica dell'*Edda* in prosa risulta molto più interessante delle pagine dove si spiegano specificatamente i vari tipi di metri della poesia scaldica; questa visione parziale dell'opera, unita al fatto che le moderne traduzioni favoriscono le parti mitologiche, (in italiano non esiste una versione dell'*Edda* di Snorri completa anche del prologo e dell'*Háttatal*) ha portato al diffuso fraintendimento che l'*Edda* di Snorri sia essenzialmente un trattato di mitologia, cosa che non era probabilmente nell'intenzione dell'autore.

Il secondo libro, lo *Skáldskaparmál*, più vicino invece all'intento "didattico" dell'autore, adotta come cornice un dialogo tra il gigante marino Ægir e il dio della poesia Bragi, in cui il secondo spiega al primo il significato delle più importanti figure retoriche scaldiche e degli *úkend heiti* (semplici denominazioni o epiteti), dando così al lettore molte importanti informazioni sull'origine e sul significato dei nomi mitologici citati nelle opere degli scaldi. A volte Snorri non si contenta di citare una semplice metafora poetica, ma fornisce lunghi estratti dei poemi che la contengono, conservando così frammenti di importanti opere che altrimenti sarebbero andate completamente perdute.

Il terzo libro, l'*Háttatal* o "Catalogo dei metri", è un saggio tecnico che appare nella forma di un poema laudatorio per Hákon Hákonarson – l'allora re di Norvegia (1217-1263) – e per Skúli Bárðarson, in cui Snorri fornisce un esempio dei centodieci differenti tipi di metro che conosce. Come parte integrante di questo libro, c'è anche una lista di *pulur* (*heiti* in versi allitterativi) nei quali sono presentate le diverse maniere di indicare gli dei, i giganti, gli uomini e le donne, le battaglie e le armi, gli elementi naturali e via dicendo.

L'*Edda* in prosa di Snorri ci è stata tramandata in quattro codici più o meno completi: Il *Codex Regius*², redatto nel 1325 e considerato il manoscritto più

² Questo manoscritto non va confuso con un altro *Codex Regius* che contiene a sua volta un'*Edda*, l'*Edda* poetica.

affidabile nella redazione delle edizioni critiche dell'opera³, il *Codex Wormianus*, composto probabilmente tra il 1340 e il 1350, il *Codex Trajectinus*, molto più tardo e redatto probabilmente intorno al 1600 e il *Codex Uppsaliensis*, composto nel 1300 e testimonianza più antica dell'*Edda* in prosa, nonché l'unico che faccia riferimento all'autore e che abbia il titolo. I quattro codici restituiscono ognuno una versione originale dell'*Edda*, oltre che un'organizzazione variabile delle singole parti. Questa è una delle caratteristiche fa dell'*Edda* una delle opere più complicate da organizzare dal punto di vista bibliografico visto l'intrecciarsi di complicate relazioni fra le varie, e diverse, redazioni dell'opera.

1.2 – Ricezione e fortuna

La storia della ricezione dell'*Edda* è stata caratterizzata da molti problemi testuali, sia dei codici originali, sia delle pubblicazioni seguenti. L'*Edda* di Snorri infatti è stata spesso modificata, tradotta e conosciuta per parti, selezionate secondo interpretazioni soggettive o a criteri storici di ricostruzione più o meno attendibili (ancora una volta ne è testimonianza la mancata esistenza a oggi di un'edizione italiana completa). L'*Edda* iniziò a diffondersi in Europa tra il XVII e il XVIII secolo, in seguito al movimento di Panscandinavismo di Svezia e Danimarca, che contribuì alla volontà di far conoscere alcune opere fondanti della tradizione scaldica, come l'*Edda*, in altre lingue, tra cui il latino, il tedesco e in seguito l'inglese. La traduzione dell'opera, o di parte di essa, in lingue diverse dallo scandinavo, antico o moderno, sarà lo strumento primario della notorietà internazionale del testo e del suo assurgere, per noi, a fonte primaria per la conoscenza della mitologia nordica e della tradizione poetica islandese, di cui rimangono solo poche altre testimonianze.

L'attendibilità dell'*Edda* di Snorri come fonte dei miti scandinavi è stata spesso messa in dubbio⁴, ma è evidente che Snorri attinse con scrupolo quasi “filologico” alle fonti pagane, tant'è vero che cita a più riprese lunghi passi degli antichi poemi mitologici da lui utilizzati per la compilazione dell'opera. A volte tuttavia, un

³ Tra i manoscritti medioevali che contengono il *Gylfaginning*, il Regius ha il testo più coerente e mostra minori segni di alterazioni o interpolazioni successive ad opera di eventuali scribi.

⁴ Cfr. Isnardi, *I Miti nordici*.

confronto tra i versi citati e il resoconto prosastico fornito da Snorri mostra una non perfetta coincidenza, come se l'autore abbia interpretato o in parte travisato le sue fonti. Che questo sia avvenuto perché Snorri non conoscesse perfettamente la materia o perché abbia operato delle licenze letterarie, non c'è dato saperlo.

Un approccio realistico all'*Edda* deve tener conto che Snorri, nonostante fosse un uomo erudito e un ottimo conoscitore della cultura, della poesia e della letteratura islandese, visse e scrisse in un'epoca in cui il cristianesimo si era ormai affermato in Islanda e la conoscenza tradizionale era in declino. Bisogna quindi considerare che questi fattori influirono sicuramente sull'utilizzo che fece delle proprie fonti.

La rilevanza dell'*Edda* è ad ogni modo inestimabile per quanto riguarda la nostra conoscenza dei miti scandinavi, Snorri infatti fornì nel suo libro spiegazioni e interpretazioni approfondite di ogni mito e metafora che citò. Senza il suo contributo, buona parte dei miti cui i poemi eddici accennano, sarebbero per noi del tutto incomprensibili.

1.3 – Bibliografia dell'*Edda* di Snorri

Con “bibliografia” s'intende l'insieme sistematico di documenti, codici, libri, saggi o articoli su un dato argomento. Nel caso dell'*Edda* di Snorri la dimensione e la quantità di titoli inerenti è indubbiamente notevole, ma ciò che più caratterizza la bibliografia così particolare di quest'opera sono principalmente i rapporti, non sempre chiari, che intercorrono tra i quattro codici originari e le altre fonti minori.

I quattro codici menzionati: Il *Codex Regius*⁵, il *Codex Wormianus*⁶, il *Codex Trajectinus*⁷ e il *Codex Uppsaliensis*⁸ si possono considerare come quattro manoscritti dal valore testuale indipendente, ma tutti contenenti almeno le quattro parti fondamentali dell'*Edda*: il prologo, il *Gylfaginning*, lo *Skáldskaparmál* e l'*Háttatal*. Il *Codex Regius* e il *Trajectinus* hanno testi abbastanza simili tra loro,

⁵ Conservato presso l'Istituto Árni Magnússon di Reykjavík in Islanda.

⁶ Conservato nell'Arna-Magnæanske Samling, la biblioteca universitaria di Copenaghen.

⁷ Conservato nella biblioteca di Utrecht.

⁸ Conservato nella biblioteca dell'Università di Uppsala.

mentre il *Wormianus* e l'*Uppsaliensis* differiscono sia tra di loro sia dagli altri due codici, in alcuni punti lievemente, in altri in modo piuttosto consistente, sia nella formulazione che nel contenuto.

Anche le lacune e le mancanze sono diverse da codice a codice, Nel *Codex Regius* infatti è andato perduto il primo foglio e manca anche l'inizio del prologo. Inoltre dopo l'*Hattatal*, il *Regius* riporta due poemi: *Jómsvíkingadrápa* e *Málsháttakvæði*, che non fanno parte dell'*Edda*. Il *Trajectinus* manca anch'esso dell'inizio del prologo ma anche della fine dell'*Háttatal*. Il *Codex Wormianus* invece include nel prologo lunghi passaggi che non sono negli altri manoscritti; questi passaggi in particolare contengono materiale tratto dalla tradizione biblica e classica e si pensa quindi che possano essere delle interpolazioni. Sempre il *Codex Wormianus* contiene poi anche quattro *Málfróðiritgerðar* (trattati grammaticali), il poema *Rígsþula* e una raccolta di *þulur* che non fanno parte dell'*Edda*. Allo stesso tempo vi è una vasta lacuna nello *Skáldskaparmál*. Questa parte inoltre era stata oggetto di revisione da parte di un redattore del XIV secolo, ma essendone andate perse le pagine non è possibile osservare se e in quale misura fossero state effettuate delle modifiche⁹. Sia il *Codex Wormianus* sia l'*Uppsaliensis* mancano poi dei *þulur* che si trovano al termine dello *Skáldskaparmál* negli altri due codici. Malgrado ciò, per quando riguarda il contenuto, il *Wormianus* ha un testo in parte corrispondente al *Regius* e al *Trajectinus*, ben diverso invece, è quello del *Codex Uppsaliensis*.

Tutto il testo del *Codex Uppsaliensis* è infatti molto diverso dagli altri. Esso è stato oggetto di un'ampia opera di sintesi e di abbreviazione di molti termini, con il risultato che in molte parti è piuttosto difficile dare un senso al contenuto. Vari passaggi comuni con gli altri manoscritti sono lacunosi, e gran parte del materiale presente è inserito in un ordine diverso. L'*Háttatal* inoltre è in parte incompleto. Il *Codex Uppsaliensis* contiene inoltre svariato materiale che, pur non facendo direttamente parte dell'*Edda*, è collegato con Snorri Sturluson o riflette i suoi interessi, per questo motivo si pensa che possa provenire dalle sue carte. Tra questi vi sono lo *Skáldatal* (una lista di poeti di corte scandinavi, di cui se ne trova una versione anche in uno dei manoscritti dell'*Heimskringla*), una genealogia della

⁹ Cfr. Faulkes, *Snorri Sturluson: Edda*, Introduzione, xxviii.

famiglia Sturlung a cui Snorri apparteneva, un elenco di *lögsögumaður* islandesi¹⁰ che termina con il nome di Snorri stesso, e infine una versione del secondo dei quattro trattati grammaticali (*Málfróðiritgerðar*).

Sono inoltre sopravvissuti dei frammenti di tre altri manoscritti medievali, facenti parte dell'*Arnarnagnaean Codex*, che contengono parti dello *Skáldskaparmál* e dei *þulur*, anche se è impossibile dire se una volta contenessero altre sezioni dell'*Edda* oltre a quelle sopravvissute. Il testo di uno di questi manoscritti, presumibilmente composto intorno al 1400, è molto simile a parti del *Codex Regius*, a cui un secondo scriba avrebbe in seguito aggiunto una genealogia della famiglia Sturlung. Altri due manoscritti, il primo composto nel primo quarto del XIV secolo e il secondo circa nel 1400, contengono entrambi parti di una redazione dello *Skáldskaparmál* e una raccolta di *þulur* piuttosto differenti dalla versione che si trova nei quattro codici principali, inoltre presentano un altro elenco di *kenningar* che probabilmente non erano originariamente parte dell'*Edda* di Snorri.

Un buon numero dei versi scaldici dell'*Edda* si trovano anche in varie altre opere indipendenti come le saghe dei re (tra cui l'*Heimskringla*), le saghe degli islandesi, e i menzionati trattati grammaticali. La maggior parte dei poemi eddici citati nel *Gylfaginning* si trova poi anche nel *Codex Regius* dell'*Edda antica*, o *Ljóða Edda* (*Edda* poetica) redatto nella seconda metà del XIII secolo, e a lungo ritenuto l'ispiratore o la fonte dell'*Edda* di Snorri, nonostante il fatto che il nucleo dell'*Edda antica* sia stato composto circa nello stesso periodo in cui Snorri scriveva la sua *Edda*, e sia stato terminato quasi mezzo secolo dopo¹¹. Parti o brani dell'*Edda* di Snorri sopravvivono anche in numerosi manoscritti successivi in gran parte derivati da testimoni medievali o dai codici originari, malgrado in alcuni casi contengano versioni di parti dell'*Edda* che non trovano riscontro in nessun altro testo; per esempio, alcune redazioni differenti della seconda parte dello *Skáldskaparmál* del *Codex Wormianus* e l'inizio del prologo del *Regius*. I testi di molti di questi manoscritti più tardi sono stati oggetto di ampie modifiche e interpolazioni, che in particolare riguardano lo *Skáldskaparmál*, una delle sezioni che sia nel periodo

¹⁰ Coloro che nelle assemblee generali dei popoli nordici erano incaricati di recitare, con l'aiuto della poesia, tutte le regole, le leggi e le consuetudini vigenti in una determinata comunità.

¹¹ Cfr. Bellows, *The Poetic Edda*, Introduzione, xiv-xv.

medievale sia in quello rinascimentale hanno attirato maggiormente l'attenzione dei lettori.

Ci sono state molte discussioni su quale, tra i vari codici, (in particolare l'*Uppsaliensis*, il *Regius* e il *Trajectinus*) conservasse nel modo migliore l'*Edda* come venne scritta da Snorri, ma effettivamente ci sono solo pochi dati su cui basare un giudizio oggettivo. Il *Codex Uppsaliensis* è probabilmente il manoscritto più antico, anche se solo per pochi anni, e il materiale che accompagna l'*Edda* in questo codice implica una stretta connessione con Snorri stesso, ma la disposizione in questa versione sembra, almeno per i lettori moderni, decisamente poco logica. Gli studiosi sono poi restii ad attribuire quello che sembra essere un miglioramento, rispettivamente del *Regius* e del *Trajectinus*, all'alterazione di un eventuale scriba, eppure, se i due codici fossero le due versioni più vicine a quella originale risulta difficile capire che cosa avrebbe potuto condurre un eventuale scriba a modificare un assetto che sembrava così soddisfacente e produrre l'illogicità dell'*Uppsaliensis*. Una terza possibilità, che potrebbe essere attraente, ma che non è al momento dimostrabile, è che il *Codex Uppsaliensis* sia derivato da un progetto dello stesso Snorri, il quale avrebbe riunito nel codice svariati fogli sciolti, e di cui il *Regius* e il *Trajectinus* (e il *Wormianus* ovviamente) sarebbero una conseguente versione riveduta e ampliata, e che le differenze tra le versioni sarebbero poi state causate da rimaneggiamenti o interpolazioni successive. Tuttavia nel complesso pare meglio ammettere che i manoscritti conservano diverse raccolte tutte basate sul lavoro perduto di Snorri Sturluson, e ognuno di essi ha il suo proprio interesse e valore intrinseco.

1.3.1 – Problemi specifici di una bibliografia dell'*Edda* di Snorri

Il corpus della bibliografia riguardante l'*Edda* di Snorri costituisce un problema complesso ed estremamente vasto per quanto riguarda una sua pubblicazione, implementazione costante e diffusione integrale. Questa a causa sia della quantità dei testi inerenti (certamente non trascurabile), ma soprattutto per via dei legami tra i testimoni primari dell'opera (quattro manoscritti completi, tre frammenti medievali e

un apografo moderno completo), le edizioni critiche¹² e la ricchissima letteratura secondaria che è stata dedicata a un testo di primaria importanza per gli studi letterari di ambito nordico. Una bibliografia dell'*Edda*, anche se incompleta, mostra infatti la presenza di una letteratura snorriana in diversi paesi, attraverso diverse epoche, e come questa abbia inciso nello sviluppo di movimenti culturali propri della Modernità.

La forma aperta che caratterizza, oltre che i manoscritti completi, anche le versioni a stampa realizzate a partire dal XVII secolo, è un altro dei grandi ostacoli a un'organizzazione organica dei testi relativi all'*Edda*. Le edizioni a stampa, in un'epoca in cui la riproduzione chirografica del testo non si era comunque ancora esaurita, erano sovente accompagnate da traduzioni più o meno accurate (dapprima in latino e nelle lingue scandinave moderne, quindi nelle lingue della filologia accademica, soprattutto inglese e tedesco) che saranno lo strumento primario della notorietà internazionale del testo e che incrementeranno notevolmente il numero di versioni esistenti dell'opera.

Per la sua natura composita, l'*Edda* di Snorri è implicata in qualsiasi studio sui miti e le tradizioni poetiche dell'Islanda e della Scandinavia medievali rendendola quindi un testo onnipresente in tutte le ricerche riguardanti l'argomento. Infatti, come sostenuto anche da J.S. Hannesson: «An exhaustive bibliography of works bearing on the Eddas is impossible»¹³. Una bibliografia dell'*Edda* e di tutto ciò che le concerne è un'opera praticamente impossibile, almeno finché viene sviluppata e organizzata secondo i metodi classici.

1.3.2 – La soluzione: una bibliografia online

La soluzione ottimale per ovviare alla mole di dati e alla fittissima interconnessione tra gli stessi è stata trovata nella realizzazione di una bibliografia online, termine con cui indichiamo un database che può essere interrogato attraverso un'interfaccia web, consentendo la consultazione di articoli, testi, libri e collegamenti relativi in maniera

¹² Emblematico il caso dell'*editio princeps* del 1665, ad opera di Peter Johannes Resen, che si rivelò più una riscrittura che una vera e propria edizione.

¹³ Cfr. Hannesson, *A supplement to Bibliography of the Eddas*, (Islandica XIII).

piuttosto semplice. Il database dell'*Edda* di Snorri, in particolare, è stato pensato come in continua evoluzione (una delle sue caratteristiche più importanti) e aperto alla comunità accademica grazie all'uso di un software appositamente creato. È stato così creato un database relazionale in cui ogni singola voce è stata archiviata in modo da indicarne le relazioni all'interno della letteratura primaria e secondaria. Questo ha consentito di restituire per ogni ricerca non solo i risultati richiesti, ma anche le necessarie informazioni metatestuali. Ciò favorisce tra le altre cose la comprensione di quale sia la versione dell'*Edda* di Snorri a cui ciascun titolo si riferisce, offrendo così all'utente la possibilità di tracciare una storia degli studi snorriani (ad esempio, nell'indicare i paesi, le lingue e il genere di pubblicazione).

La rete di nodi che rappresenta la bibliografia può così essere letta come un'infrastruttura di relazioni percorribile in ogni direzione per reperire più efficacemente le informazioni necessarie. Inoltre, la creazione di una bibliografia online ha permesso di definire una scheda bibliografica molto più ricca di dettagli di quanto sarebbe stato altrimenti possibile seguendo gli standard classici, eliminando al contempo i problemi legati ai lunghissimi tempi per la distribuzione e la catalogazione delle nuove entrate e alla difficile reperibilità di alcuni testi.

L'esistenza di una bibliografia indicizzata insieme a keyword "tradizionali" (parole chiave, riferite al contenuto di ogni singola entrata), e a keyword "relazionali", capaci di indicare le relazioni tra i titoli indicizzati all'interno del corpus raccolto, unita alla perfettibilità intrinseca del supporto digitale, ha permesso di effettuare ricerche bibliografiche utilizzando i più moderni strumenti di interazione con l'utente, quali tastiere virtuali per facilitare l'inserimento di caratteri speciali e suggerimenti per il completamento della ricerca. Questo ha consentito di ovviare allo stesso tempo ai problemi insormontabili che si ponevano nel momento in cui era necessario affrontare una bibliografia "fisica" dell'*Edda* di Snorri.

IL PROGETTO “Online bibliography of Snorra Edda”

2.1 – Il progetto

Il progetto “Online Bibliography of Snorra Edda” è stato avviato nel 2008 su impulso della Prof.ssa Adele Cipolla del dipartimento di lingue e letterature straniere dell’Università di Verona. Il progetto nasce con l’obiettivo primario di creare un database sull’*Edda* di Snorri e su tutta la bibliografia inerente che fosse esaustivo, capace di rispondere a ricerche specifiche e aperto alla comunità accademica.

Questo fine è stato raggiunto grazie all'uso di un software creato specificatamente con quest’obiettivo, il cui codice sorgente è stato rilasciato come software open source, sotto licenza GPL, così da favorire lo sviluppo di funzionalità aggiuntive e la riusabilità. La soluzione ottimale individuata per effettuare un'adeguata raccolta e consultazione dei dati bibliografici è stata quella di creare un database relazionale, impostato sui collegamenti tra le varie voci: in questo modo ogni entrata bibliografica è stata archiviata così da registrare, contestualmente, i rapporti e i possibili collegamenti con i manoscritti, le edizioni critiche e il complesso della letteratura secondaria.

La scelta di creare un database *ex-novo* piuttosto che fare ricorso a programmi di questo tipo già disponibili sul mercato, inoltre, ha permesso di definire una scheda bibliografica molto più ricca e fruibile dal punto di vista della ricerca di materiale interconnesso. Il software ha integrato un nuovo motore di ricerca: OpenBMS, il quale è già operativo e include sia un'interfaccia di ricerca, sia un modulo per l'inserimento dei dati direttamente dal web, consentendo così ai singoli utenti (con privilegi amministrativi) di interagire direttamente con il database e aggiornare le varie entrate bibliografiche integrandole o correggendole secondo i parametri disponibili (*Country, Journal, Keyword, Language, Names, Role, Title, Type*). Ogni modifica è poi alla fine sottoposta al controllo di un amministratore per verificarne coerenza e correttezza.

2.2 – Il software *OpenBMS*

Il progetto “Online bibliography of Snorra *Edda*” mira a realizzare un'archiviazione strutturata del materiale bibliografico relativo all'*Edda* di Snorri, consultabile tramite un'applicazione web, e quindi completamente accessibile, che permetta di effettuare ricerche e aggiornare il database in modo diretto ed efficace. È per raggiungere questo scopo che è stato sviluppato un software specifico: *OpenBMS* (Open Bibliography Management System).

OpenBMS è una Java Web Application, sviluppata con tecnologia Java5¹⁴ e JSTL¹⁵ (Java Server Pages e Java Standard Tag Library), installata su un Web Container Java messo a disposizione dall'Università degli Studi di Pisa. L'applicazione implementa il pattern MVC (Model - View – Controller, che è basato sulla separazione dei compiti fra i componenti software per seguire separatamente i tre incarichi principali), per permettere future implementazioni di funzionalità legate all'esportazione dei dati e all'esposizione di servizi che non facciano uso dell'interfaccia web. Per questa ragione la presentazione dei dati è delegata alle JSP (Java Server Page) e al framework javascript ExtJS, che propone un ambiente di sviluppo completo di quasi tutti i componenti presenti nelle normali applicazioni desktop. Per quanto riguarda l'Object Relational Mapping, (una tecnica di programmazione che favorisce l'integrazione di sistemi software legati agli “oggetti”) è stato utilizzato il framework di persistenza Hibernate¹⁶.

L'interfaccia utente utilizza dunque ExtJS per facilitare la ricerca tramite richieste asincrone, gestite sui singoli componenti delle pagine. In questo modo il motore javascript può interrogare il server mentre l'utente digita parte dei parametri di ricerca per ottenere i risultati (in formato JSON, JavaScript Object Notation, o XML). Questo metodo consente di non dover cambiare diverse pagine per impostare i parametri di ricerca.

L'applicazione è composta quindi da due parti fondamentali: un back-end per il popolamento del database e un front-end che consente la ricerca. Le entrate

¹⁴ “Oracle technology networks for Java developers”, <http://java.sun.com>.

¹⁵ “Java Server Pages Standard Tag Library”, <http://java.sun.com/products/jsp/jstl>.

¹⁶ “Hibernate – Jboss Community”, <http://www.hibernate.org>.

bibliografiche sono caricate nel database attraverso il back-end, utilizzabile anche grazie a un'interfaccia grafica sviluppata per consentire persino agli utenti senza competenze di SQL di interagire comunque con la banca dati. Questa soluzione da un lato consente di aggiornare la bibliografia in tempo reale, dall'altro facilita la navigazione all'interno del corpus.

L'accesso al portale infine è consentito (al momento) a due tipi di utenti, autenticati e non autenticati: gli utenti autenticati (che per adesso sono solo utenti con anche privilegi amministrativi) possono sia compiere ricerche sia inserire nuove voci nella bibliografia tramite il back-end, usando i campi preimpostati (*Country, Journal, Keyword, Language, Names, Role, Title, Type*) come guida. Gli utenti autenticati possono quindi fare integrazioni o correzioni del database che in ogni caso l'amministratore potrà modificare, in modo da rendere disponibili gli aggiornamenti, corretti, in tempo reale. Gli utenti non autenticati invece possono solo fare ricerche all'interno del database attraverso l'interfaccia di ricerca.

2.3 – Sviluppo di *OpenBMS*

Le aspettative enunciate richiedevano uno strumento facilmente utilizzabile e continuamente perfezionabile, che consentisse la migliore interazione tra amministratore e utenti, anche allo scopo di ottimizzare e arricchire il database, sollecitando i contributi della comunità scientifica e consentendo il collegamento con altre banche dati o singoli "indirizzi" per la consultazione online di quelle tra le risorse che erano disponibili in rete.

L'evoluzione del progetto, e quindi di *OpenBMS* stesso, ha dovuto obbligatoriamente attraversare alcune fasi di sviluppo, divisibili tra la parte riguardante il database, la sua creazione, il suo popolamento e la sua gestione e l'interfaccia utente, ossia il veicolo attraverso il quale la bibliografia è diventata effettivamente fruibile da tutti. Di seguito è illustrata questa evoluzione nelle sue varie parti.

2.3.1 – Costituzione del database

La creazione del database ha inizio molto prima della nascita del progetto “Online bibliography of Snorra *Edda*” nel 2008, partendo effettivamente circa dieci anni fa. Il progetto infatti è nato ufficialmente nel 2000, presso il Dipartimento di Germanistica e Slavistica dell'Università di Verona. Da quel momento si è iniziato a raccogliere un database bibliografico esaustivo dell'*Edda* di Snorri. Oltre ai repertori “snorriani” di “*Islandica*” (Hermannsson 1920, Hannesson 1955), sono state consultate le principali bibliografie sulla letteratura islandese antica (Warmholtz 1782, Möebius 1856, BONIS “Bibliography of old Norse-Icelandic studies” 1964-1988, Gippert, Laursen e Röhn 1991) e sulla mitologia nordica (Simek 1987, Lindow 1988, etc.); per gli stadi primitivi della riscoperta e degli studi su Snorri sono stati decisivi gli apporti delle pubblicazioni apparse nell'ambito del progetto “*Norse Muse*” (come Krömmelbein 1998, Lönnroth, Clunies Ross 1999, etc.). Sono state inoltre consultate le principali riviste scientifiche, dapprima quelle di settore, poi quelle appartenenti all'ambito più ampio della letteratura nordica. I dati della bibliografia sono in costante aggiornamento e a oggi (Luglio del 2012), contano più di 2500 entrate.

I criteri di selezione delle voci che sono stati applicati si adeguano a quelli stabiliti nelle due bibliografie specifiche dell'*Edda* apparse in “*Islandica*” XIII e XXXVII:

- sotto l'etichetta di “letteratura primaria” sono state incluse edizioni, traduzioni e parafrasi dell'opera;
- sotto quella di “letteratura secondaria” «All works of immediate importance for Eddic studies» (Jóhann S. Hannesson, 1955).

Definizione, quest'ultima, generica e applicabile a qualsiasi corpus bibliografico. In effetti, nel numero potenzialmente illimitato dei titoli pertinenti alla “letteratura secondaria”, sono state incluse:

- monografie e pubblicazioni miscellanee espressamente intitolate a Snorri e alla sua *Edda*;
- articoli su riviste specializzate, volumi tematici o atti di convegni, con specifica menzione o plausibile allusione all'argomento trattato.

Sono poi state incluse, nonostante l'alto rischio di darne un elenco incompleto, le opere di consultazione che avessero sezioni dedicate all'*Edda*, contrariamente a quanto invece fece Jóhann Hanneson nella sua bibliografia su "*Islandica*".

Come nei volumi di "*Islandica*", ristampe, riedizioni e recensioni sono elencate in calce a ciascun'entrata. Le recensioni più rilevanti sono inoltre riportate autonomamente tra le entrate e annotate come le altre.

2.3.2 – Implementazione e gestione del database

Alla raccolta di un numero congruo di entrate bibliografiche che soddisfacessero i criteri stabiliti per rientrare nella bibliografia dell'*Edda*, è seguita la popolazione vera e propria del database online attraverso l'interfaccia che era stata appositamente creata (Nella versione 0.8.3 esisteva ancora anche la Tabella *Manuscript*, eliminata nella versione seguente).



Figura 1 – Screenshot del back-end

L'interfaccia è stata pensata per permettere all'utente (con privilegi amministrativi) sia di aggiungere nuovi records cliccando sul bottone "Add New" posto sotto ognuna delle entità, sia di sfogliare il database in tutte le sue parti (*Names, Keywords, Language, Type, Country, Journal, Role, Title*), tramite il comando "List All", consentendogli così di visualizzare tutte le voci aggiunte fino a quel momento per ogni categoria.

OpenBMS ver. 0.8.3 Last ACTION >			
Logged in as: admin Logout			
virtual keyboard risultati: totali: 1389 pagine: 47 corrente: 1			
Names	<input type="checkbox"/>	1487	Rosén
<input type="checkbox"/>			Helge
Keywords	<input type="checkbox"/>	908	Aall
<input type="checkbox"/>			Jacob
<input type="checkbox"/>		1	Abrahamson
<input type="checkbox"/>			Werner Hans Frederik
<input type="checkbox"/>		534	Abram
<input type="checkbox"/>			Christopher
Language	<input type="checkbox"/>	1415	Acker
<input type="checkbox"/>			Paul
<input type="checkbox"/>		1332	Adelung
<input type="checkbox"/>			Johann Christoph
<input type="checkbox"/>		2	Adlerbeth
<input type="checkbox"/>			Jakob
Type	<input type="checkbox"/>	397	Ågren
<input type="checkbox"/>			Yngve
<input type="checkbox"/>		1319	Aguirre
<input type="checkbox"/>			Manuel
<input type="checkbox"/>		1173	Ahlbäck
<input type="checkbox"/>			Tore
Country	<input type="checkbox"/>	544	Åkerblom
<input type="checkbox"/>			Alex
<input type="checkbox"/>		3	Albano Leoni
<input type="checkbox"/>			Federico
<input type="checkbox"/>		811	Alexandersson
<input type="checkbox"/>			Henrik
Journal	<input type="checkbox"/>	1471	Allen
<input type="checkbox"/>			N.J.
<input type="checkbox"/>		409	Almqvist
<input type="checkbox"/>			Bo
<input type="checkbox"/>		1209	Aloe
<input type="checkbox"/>			Stefano
Role	<input type="checkbox"/>	1210	Ambrosi
<input type="checkbox"/>			Paola
<input type="checkbox"/>		4	Amory
<input type="checkbox"/>			Frederic
<input type="checkbox"/>		413	Andersen
<input type="checkbox"/>			Harry
Title	<input type="checkbox"/>	1112	Andersen
<input type="checkbox"/>			Stig Toftgaard
<input type="checkbox"/>		5	Anderson
<input type="checkbox"/>			George K.
<input type="checkbox"/>		1195	Anderson
<input type="checkbox"/>			Linda
<input type="checkbox"/>		6	Anderson
<input type="checkbox"/>			Rasmus B.
Manuscript	<input type="checkbox"/>	7	Andersson
<input type="checkbox"/>			Theodore Murdock
<input type="checkbox"/>		1244	Andersson Palm
<input type="checkbox"/>			Lennart
<input type="checkbox"/>		798	Andrén
<input type="checkbox"/>			Anders

Figura 2 – L'elenco dei records all'interno di "Names"

Dall'interfaccia amministrativa si possono non solo aggiungere, ma anche modificare le diverse entità e relazioni che costituiscono la base dati. Nel caso di *Names, Keywords, Language, Type, Country, Journal* e *Role* l'inserimento e la modifica contemplano solo attributi stretti dell'entità omonima tramite un processo che è lo stesso per tutte le entità. Per la gestione di *Title* invece, a differenza delle entità semplici, si distingue la gestione degli attributi propri del "titolo" stesso dalle relazioni che questo ha con le altre entità. Questa distinzione serve per permettere di collegare un numero arbitrario di entità semplici al "titolo" scelto (ad esempio più *Keywords*). Dal punto di vista dell'inserimento, l'attività è quindi gestita in due fasi: la creazione del "titolo" con l'attribuzione delle sue proprietà e la modifica delle sue relazioni.

OpenBMS ver. 0.8.3 || Last ACTION >
 Logged in as:admin Logout

virtual keyboard

Edit Country

Names
[List All](#)
[Add New](#)

Keywords
[List All](#)
[Add New](#)

Language
[List All](#)
[Add New](#)

Id: 25
 Name: Argentina

submit

Figura 3 – Il modulo di modifica dei campi di una voce in “Country” (il modulo è identico anche per tutte le altre entità che non siano “Title”)

OpenBMS ver. 0.8.3 || Last ACTION >
 Logged in as:admin Logout

virtual keyboard

Edit Title 531

Names
[List All](#)
[Add New](#)

Keywords
[List All](#)
[Add New](#)

Language
[List All](#)
[Add New](#)

Type
[List All](#)
[Add New](#)

Country
[List All](#)
[Add New](#)

Journal
[List All](#)
[Add New](#)

Role
[List All](#)
[Add New](#)

Title
[Search](#)
[List All](#)
[Add New](#)

Manuscript
[List All](#)
[Add New](#)

Title:
 Approcci metodologici alla Snorra Edda Sturlusonar: l'edizione di Anthony Faulkes

Place:

Pages: 125-142

Year: 2004

Publisher:

Publication type:

- Article
- Article in Miscellaneous Work
- Article in Proceedings
- Article in Reference Work

Contents:

Language Notes:

Figura 4 – Il modulo di modifica dei campi di una voce in “Title”

OpenBMS ver. 0.8.3 || Last ACTION >
 Logged in as:admin Logout

virtual keyboard

risultati: totali: 1 pagine: 1 corrente: 1

delete ID

select to delete 531 Approcci metodologici alla Snorra Edda Sturlusonar: l'edizione di Anthony Faulkes

[EDIT FIELDS](#)

[EDIT RELATIONS](#)

Place:

Pages: 125-142

Year: 2004

Publisher:

Figura 5 – Screenshot dei bottoni per la modifica di campi e relazioni di un record in “Title”

Cliccando su “Edit Relations” nella pagina dell’entrata scelta è possibile modificare le relazioni del “titolo” con le altre entità. Tra le relazioni si distinguono le persone che hanno un legame con la voce: autori, traduttori e illustratori sono solo alcuni esempi di “ruoli” che possono essere associati con una tupla¹⁷ “ruolo-nome-titolo”. I ruoli e le anagrafiche sono inseribili rispettivamente tramite le entità *Role* e *Name*. È quindi possibile scegliere quale ruolo una persona ha ricoperto nella realizzazione dell'opera che è censita all'interno del database. Questo tipo di approccio permette di eseguire ricerche per autore basate sul lavoro complessivo di un nome inserito nella base dati e non dei soli titoli di cui è autore. Le altre relazioni, invece che essere triple “entità-entità-titolo” sono coppie “entità-titolo”, e quindi prevedono la semplice associazione di *n* entità a un “titolo”.

OpenBMS ver. 0.8.3 | Last ACTION >
 Logged in as: admin Logout

virtual keyboard

Names
[List All](#)
[Add New](#)

Keywords
[List All](#)
[Add New](#)

Language
[List All](#)
[Add New](#)

Type
[List All](#)
[Add New](#)

Country
[List All](#)
[Add New](#)

Journal
[List All](#)
[Add New](#)

Role
[List All](#)
[Add New](#)

Title
[Search](#)
[List All](#)
[Add New](#)

Manuscript
[List All](#)
[Add New](#)

Edit Title Relationship for title n° 531

Title: Approcci metodologici alla Snorra Edda Sturlusonar: l'edizione di Anthony Faulkes

Place: null

Pages: 125-142

Year: 2004

Publication type: Article

Related Elements:

<input type="checkbox"/>	Name:	Role:
<input type="checkbox"/>	Gianluca Sandrini	Author

[Add Name](#)

Language:

<input type="checkbox"/>	Italian
--------------------------	---------

[Add language](#)

Keyword:

<input type="checkbox"/>	Snorri's Edda
<input type="checkbox"/>	studies (modern)

[Add keyword](#)

Manuscript:

[Add manuscript](#)

Figura 6 – Il modulo di aggiunta (e modifica) delle relazioni di una voce in “Title”

I nomi e i ruoli che si vogliono associare al “titolo” devono necessariamente già esistere all’interno del database e la selezione avviene tramite i menu a tendina a completamento automatico. Nel caso del ruolo, la lista permette sia la digitazione sia

¹⁷ Una **tupla** è un generico elemento di una relazione con attributi in un database relazionale. Essa rappresenta l’entità base che deve essere memorizzata nella base di dati.

la selezione dal menù a tendina. Nel caso del nome invece è necessario inserire almeno tre caratteri per permettere il popolamento della tendina tramite una chiamata asincrona al server¹⁸. Inserire un autore significa quindi selezionare il ruolo “Author” dalla tendina “Select Role” e selezionare il nome dell'autore digitandone alcuni caratteri nella tendina “Search Name”.



Figura 7 – Il modulo di aggiunta di *Name* e *Role* di una voce in “Title”

Infine per eliminare delle relazioni è sufficiente cliccare su “Edit Relations” dove viene presentata la lista di tutte le relazioni associate al “titolo”. Ciascuna riga ha un checkbox sulla sinistra che se selezionato permette l'eliminazione della relazione tramite il pulsante delete in cima alla tabella (vedi fig. 6).

Il modello ER del database alla versione 0.6.3 dell'applicazione (quando ancora esisteva anche la tabella *Manuscript*) è poi rappresentato nel seguente diagramma:

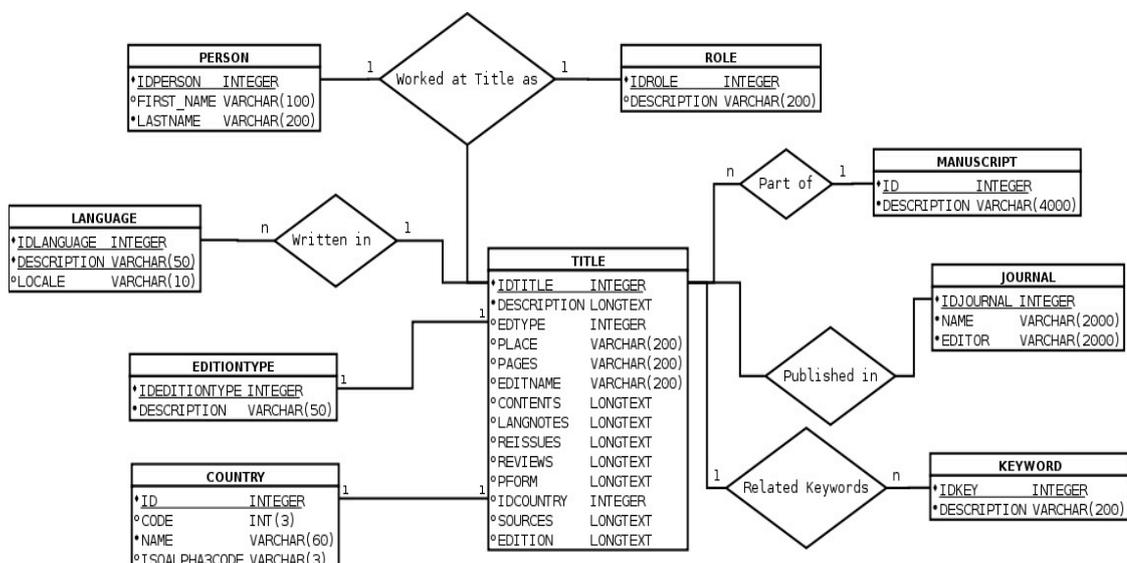


Figura 8 – Diagramma ER del database (ver. 0.6.3)

¹⁸ Cfr. capitolo II, paragrafo 2.3.3.1

2.3.3 – L'interfaccia utente

Le interfacce di OpenBMS, sia quella di ricerca sia il back-end per popolare il database, sono state realizzate attraverso il framework Javascript ExtJS. La decisione di utilizzare questa libreria nel progetto è stata determinata dalla varietà di elementi messi a disposizione dal framework, tra i quali ce n'erano alcuni che si adattavano perfettamente alle nostre esigenze. ExtJS è infatti una libreria Javascript molto potente per realizzare GUI per applicazioni web, ma fornisce anche classi per agganciare funzionalità AJAX di aggiornamento per le interfacce, creare moduli wizard o textarea e agganciare le web application a ulteriori strumenti (jQuery, Prototype, e molti altri) in modo da avere rapidamente software vicini alle applicazioni desktop con cui lavoriamo quotidianamente. La libreria possiede quindi molte caratteristiche vantaggiose:

- è cross-browser e quindi supporta tutti i maggiori web-browser tra cui: Internet Explorer 6+, Firefox 1.5+ (Pc, Mac), Safari 3+ (Pc, Mac) e Opera 9+ (Pc, Mac);
- possiede un'ampia documentazione scaricabile o consultabile sul web che illustra ogni classe ed elemento disponibile, con le rispettive proprietà, metodi ed eventi;
- ha un component model estendibile e modificabile in base alle proprie necessità;
- possiede una licenza non solo commerciale ma anche Open Source;
- è fornita di un forum e altri dispositivi di supporto per gli utenti e gli sviluppatori che usano la libreria.

La programmazione delle pagine che utilizzano ExtJS segue inoltre gli standard dell'ambiente Web 2.0. Pertanto gli script Javascript seguono le tecniche dell'OOP (Object Oriented Programming) e si fa largo uso di tecniche AJAX, di lettura e gestione DOM e di oggetti JSON, cosa che contribuisce a rendere ExtJS uno strumento utile e versatile per realizzare interfacce web.

L'interfaccia qui presentata è la ver. 0.7.9, ossia la versione implementata quando è iniziata la mia collaborazione con il progetto.

The screenshot shows a web-based search interface for OpenBMS version 0.7.9. The interface is titled "Search" and is divided into two main sections: "Basic Search" and "Publication Details". The "Basic Search" section contains three input fields: "Title:", "Name:", and "Keyword:". The "Name:" and "Keyword:" fields have dropdown arrows. The "Publication Details" section contains seven input fields: "Role:", "Journal:", "Type:", "Country:", "Language:", "Year - From:", and "Year - To:". At the bottom of the search area, there are three buttons: "Virtual Keyboard", "Search", and "Clear". The footer of the interface reads "OpenBMS, a GNU GPL 3 software".

Figura 9 – Screenshot dell’interfaccia di ricerca

Attraverso il front-end, l’applicazione permette di compiere fondamentalmente due tipi di ricerca, una semplice e una di tipo “avanzato”. Per la ricerca denominata “Basic Search” basterà inserire il titolo (o parte di esso), o il nome e cognome dell’autore oppure una parola chiave. Qualora l’utente desideri invece eseguire una ricerca più dettagliata, potrà, oltre che dare un valore ai suddetti campi, specificare:

- *Role*: che ruolo ha avuto la persona che inserita nel campo “Name” (autore, curatore, traduttore, etc.);
- *Journal*: effettuare una ricerca per Riviste;
- *Type*: ricercare per tipo di edizione;
- *Country*: ricercare per Paese di pubblicazione;
- *Language*: ricercare per Lingua;
- *Year*: ricercare “Dall’anno” - “All’anno” in cui l’oggetto della ricerca sarebbe stato pubblicato.

I campi sopra riportati rispondono inoltre a due logiche differenti: *Name*, *Keyword* e *Journal* prevedono il completamento automatico a seguito della digitazione di

almeno tre caratteri tramite chiamata asincrona al server¹⁹. *Role*, *Country* e *Language* invece hanno una *cache* di dati locali generata al caricamento della pagina che consente la scelta solo tra un numero limitato di scelte preimpostate.

Cliccando infine su “Search” viene effettuata la ricerca e i risultati vengono caricati nella pagina esposti secondo lo stile bibliografico Chicago²⁰. In alto l'utente può visualizzare il numero di risultati trovati e il tempo impiegato per la ricerca, in più cliccando sul tasto “More” è possibile visualizzare la scheda completa del record.

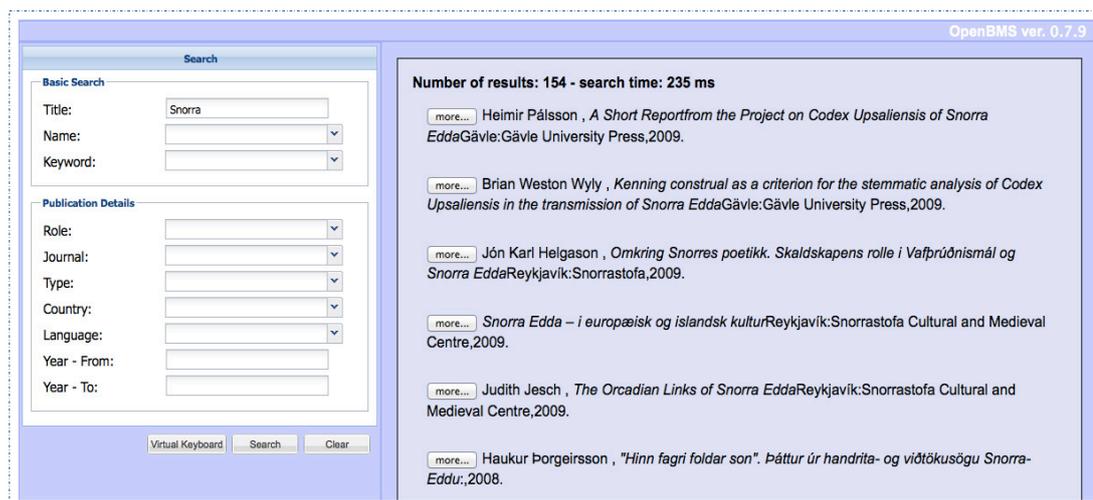


Figura 10 – Esempio di una ricerca semplice

Poiché inoltre è molto probabile che l'utente debba utilizzare dei caratteri speciali non presenti sulla tastiera, è stata infine implementata una tastiera virtuale in cui l'utente può selezionare la lingua di cui desidera utilizzare i caratteri. Cliccando col mouse sulla lettera desiderata questa comparirà nella text-area della tastiera; l'utente potrà così fare un “copia e incolla” o un “drag and drop” della stessa dall'area di testo della tastiera virtuale al campo interessato.

¹⁹ Cfr. capitolo II, paragrafo 2.3.3.1

²⁰ “The Chicago manual of style”, http://www.chicagomanualofstyle.org/tools_citationguide.html.

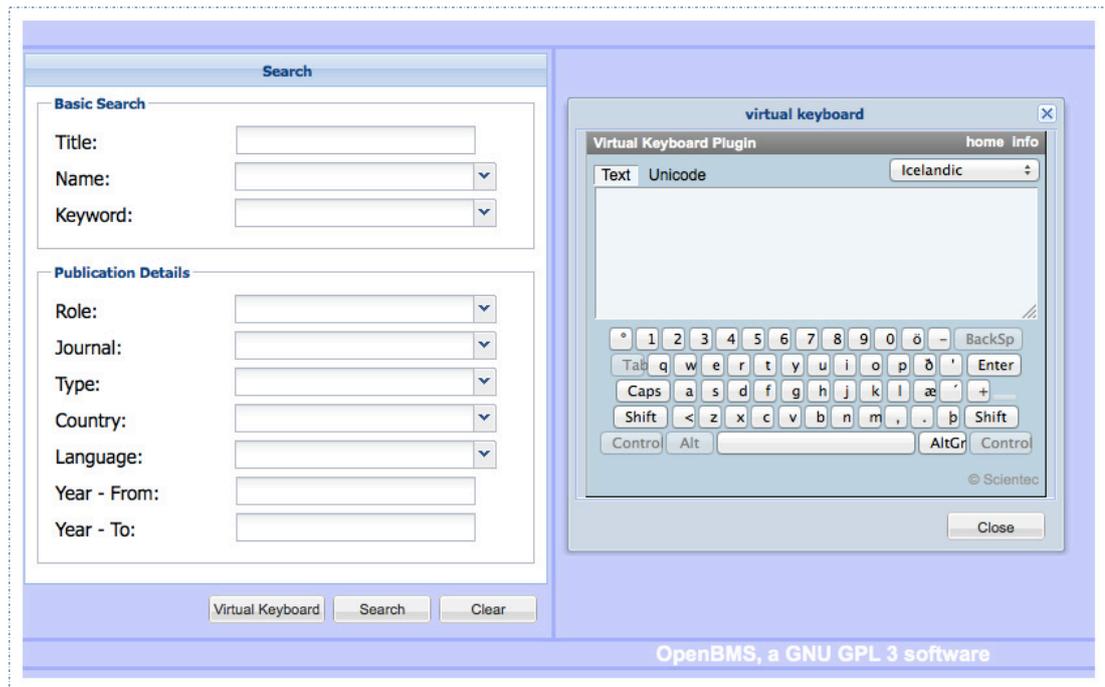


Figura 11 – La tastiera virtuale

LE CHIAMATE ASINCRONE

L'utilizzo del framework ExtJS ha consentito oltre che la creazione del layout con elementi complessi, anche di effettuare chiamate asincrone al server. Le chiamate asincrone nel caso di OpenBMS sono una delle funzionalità più utili e caratterizzanti dell'interfaccia di ricerca.

ExtJS infatti, tramite chiamate XMLHttpRequest consente di ottenere informazioni dal server in maniera diretta al primo caricamento del sito memorizzandole in cache. Queste informazioni saranno poi gestite in Javascript direttamente nella pagina (in questo senso la chiamata al server è "asincrona", perché non ha la necessità di ricaricare la pagina per aggiornarne il contenuto: il menu a tendina si popola automaticamente man mano che si scrive senza bisogno di avviare alcuna ricerca). Questo tipo di sistema è stato adottato in particolare per la funzione di auto-completamento del campo *Name* nell'interfaccia di ricerca, la cui tabella nel database è popolata oltre che da un numero notevole di elementi, soprattutto da nomi stranieri, con caratteri speciali dell'alfabeto latino e quindi difficili da inserire correttamente (anche con l'ausilio della tastiera virtuale).

Il sistema di chiamata asincrona permette infatti, una volta inserito un numero minimo di caratteri (in questo caso tre come definito direttamente nel codice Javascript) di effettuare una ricerca nella cache creatasi al primo caricamento della pagina, restituendo un array in formato Json contenente tutti i nomi trovati che corrispondano ai caratteri inseriti, popolando così il menu a tendina e consentendo all'utente di completare il campo in maniera semi-automatica.

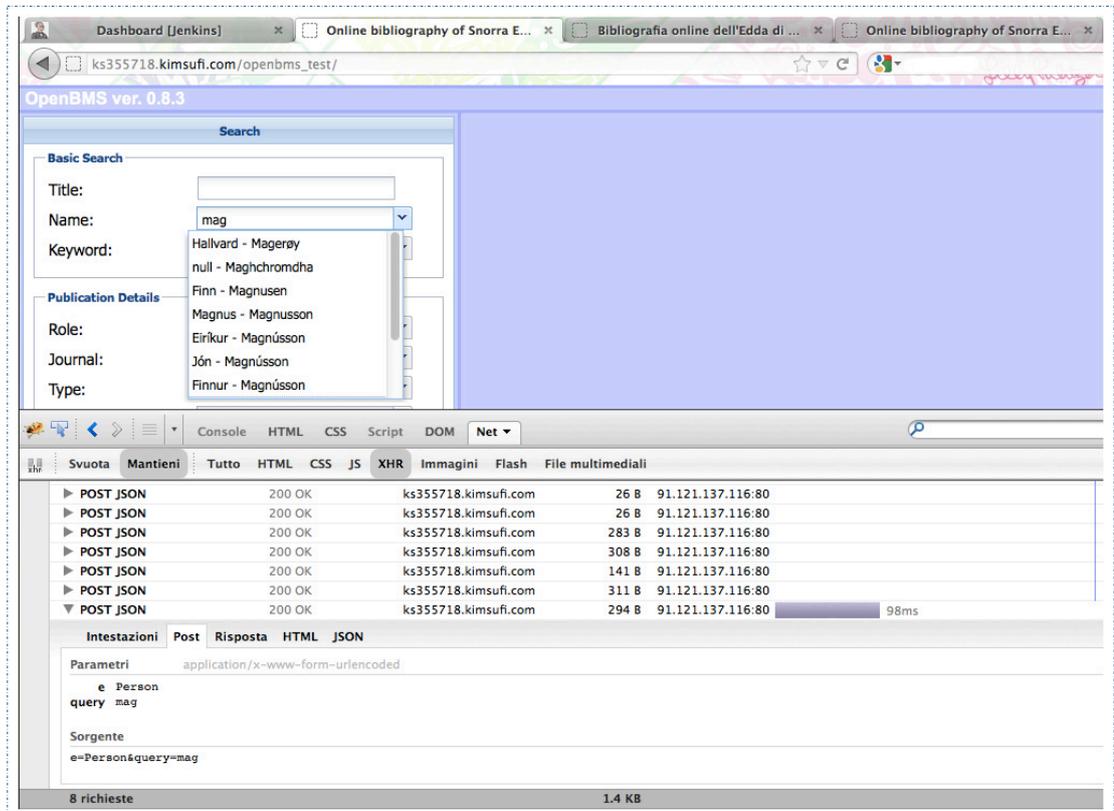


Figura 12 – Screenshot di una chiamata asincrona vista con Firebug

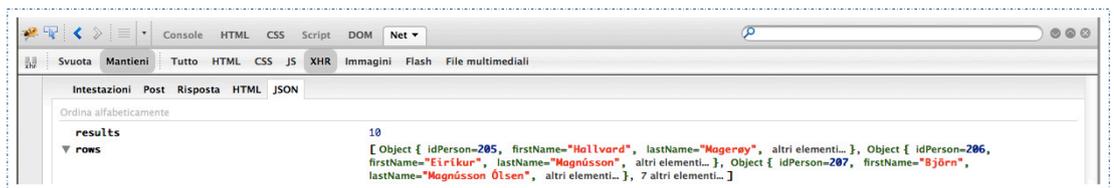


Figura 13 – Screenshot della stringa Json restituita come risposta alla query

```

function richiestaAsincrona(){
    ...
    var person_value = Ext.get('Person').dom.value;
    ...

var person_store;
var person_combo;
...

    person_store = new Ext.data.JsonStore({
        // store configs
        autoDestroy: true,
        url: APP_CONTEXT+"/JSON",
        storeId: 'person_store',
        sortInfo: {
            field: 'lastName',
            direction: 'ASC'
        },
        autoLoad: true,
        // reader configs
        root: 'rows',
        idProperty: "idPerson",
        fields: ['idPerson','firstName','lastName','cn'],
        baseParams: { 'e': 'Person'
    });

    person_combo = new Ext.form.ComboBox({
        id: "combobox_person",
        fieldLabel: 'Name',
        mode: 'remote',
        lazyInit: false,
        minChars: 3,
        displayField: 'cn',
        valueField: 'idPerson',
        store: person_store,
        hiddenName: 'Person'
    });

    ...

```

Tabella 1 – Funzione di richiesta asincrona con ExtJS

LA COLLABORAZIONE CON IL PROGETTO

3.1 – Il sito web della “Bibliografia online dell’*Edda* di Snorri”

La mia collaborazione con il progetto “Online bibliography of Snorra *Edda*” si è articolata in due periodi distinti: nel primo mi sono occupata della realizzazione del sito web di presentazione del progetto. Nel secondo invece ho collaborato in maniera più stretta con i partecipanti, e in particolare con gli sviluppatori di OpenBMS, per realizzare la nuova interfaccia di ricerca e occuparmi di parte del debug e del test dell’applicazione.

Il sito web “Bibliografia online dell’*Edda* di Snorri”²¹ è stato da me ideato e realizzato durante il periodo del mio tirocinio, svoltosi da Settembre 2011 a Marzo 2012, ed è stato anche il punto di partenza della mia collaborazione. La realizzazione del sito si è articolata in circa tre fasi: analisi dei requisiti e del contesto d’uso, creazione di alcuni prototipi dell’interfaccia e valutazione dell’usabilità, e infine implementazione del design scelto. Dopo essere stata introdotta al progetto, alla sua genesi e a quali fossero i suoi obiettivi dal professor Roberto Rosselli del Turco, il mio referente principale in questa fase, mi è stato spiegato quali fossero le aspettative per il sito che doveva presentare il lavoro svolto: sostanzialmente si richiedeva un’interfaccia semplice, che fosse intuitiva da “sfogliare” e mostrasse nella maniera più chiara possibile i vari contenuti, rimanendo allo stesso tempo graficamente accattivante. Ho deciso così di realizzare un sito web statico, che se ritenuto necessario, sarebbe migrato in seguito verso il web dinamico.

Specificato il contesto d’uso del sito, ossia quasi unicamente in ambito accademico e/o comunque da persone già a conoscenza dell’argomento, ma non necessariamente con competenze informatiche, mi sono concentrata nel rendere l’interfaccia semplice da utilizzare e lineare da vedere: i contenuti sarebbero dovuti essere accessibili nella maniera più immediata per andare incontro all’utente preso come modello.

²¹ “Bibliografia online dell’*Edda* di Snorri”, <http://www.snorri.humnet.unipi.it>.

Ho quindi realizzato due prototipi dell'interfaccia web dell'home page (due pagine HTML con i relativi fogli di stile), basilari ma già navigabili, graficamente differenti l'uno dall'altro, ma entrambi corrispondenti ai criteri indicatimi così da creare un punto di partenza per discutere su cosa fosse più indicato per il sito. Le immagini utilizzate per questi primi sketch bisogna precisare, sono state trovate tramite Google²² a puro scopo esemplificativo, e nella versione finale sono state sostituite da immagini facenti parte della bibliografia online e di cui quindi era già stata autorizzata la riproduzione.



Figura 14 – Prototipo dell'interfaccia ver. 1

²² "Google", <http://www.google.it>.



Figura 15 – Prototipo dell'interfaccia ver. 2

Dopo la presentazione delle due interfacce ho avuto dei feedback sostanzialmente positivi e ho così potuto iniziare a realizzare direttamente il layout definitivo del sito: “Bibliografia online dell’*Edda* di Snorri”. In entrambi i prototipi infatti, sono stati trovati degli elementi ritenuti interessanti e da integrare nella versione finale, come per esempio il menu a “etichette” (realizzato tramite CSS3²³) del primo modello e il range di colori scelti per il secondo. Ho così deciso di realizzare una nuova interfaccia partendo dalle due precedenti e integrando gli elementi risultati più apprezzati. Con l’obiettivo di rendere più compatta la struttura del sito si decise

²³ “CSS3. Info – Everything you need to know about CSS3”, <http://www.css3.info>.

inoltre di integrare il titolo della pagina con il banner²⁴ e di spostare il menu di navigazione al si sotto dello stesso.

Ho quindi modificato la struttura delle pagine HTML creandone due versioni: un modello senza colonne, ossia con solo testo (scelto poi per l'home page) e un modello diviso su due zone, il menu di sezione e il testo puro usato poi per le pagine “L’Edda di Snorri” e “Il progetto”.

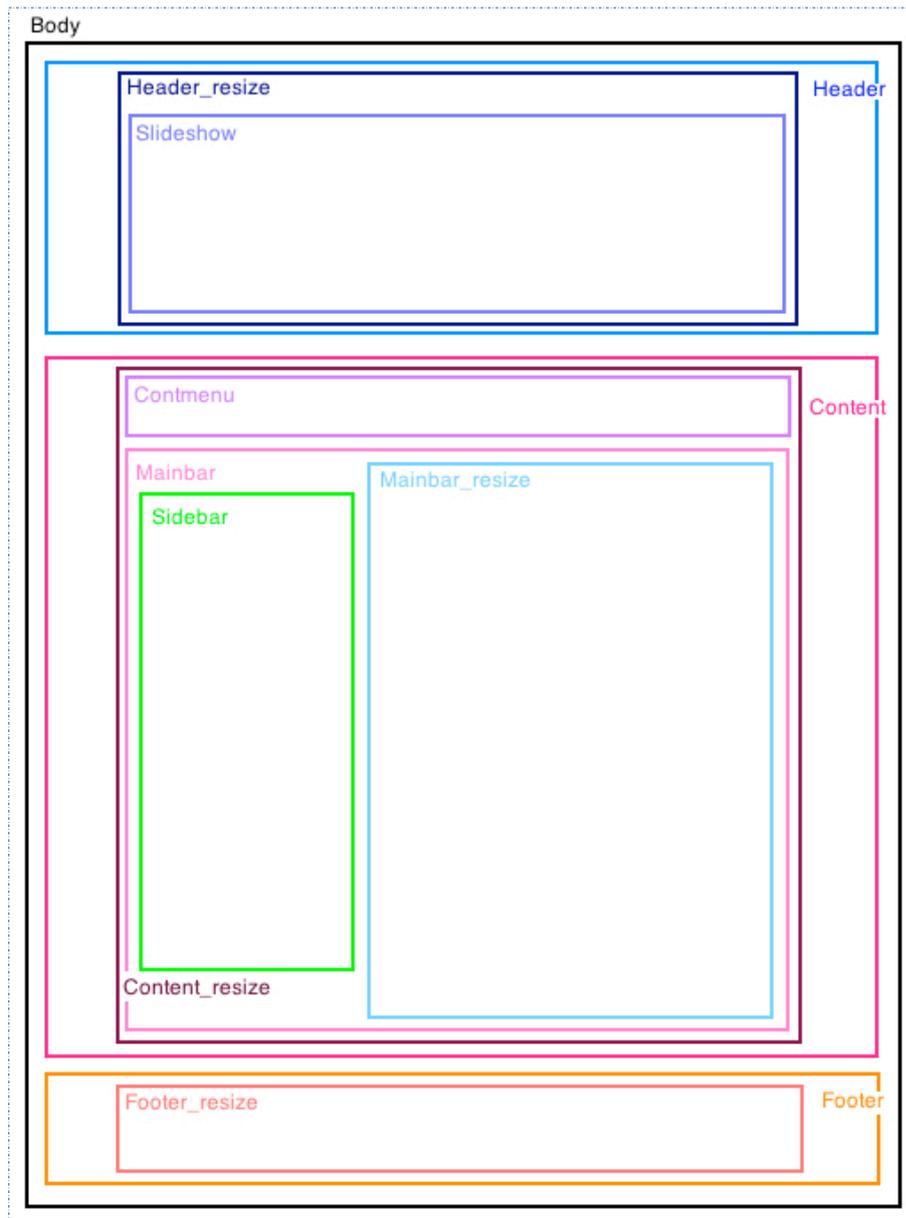


Figura 16 – Schema dello scheletro HTML delle pagine del sito

²⁴ Una striscia o un'immagine che solitamente compare all'inizio di una pagina web e che riporta il nome del sito.

Seguendo le richieste che mi erano state fatte, ho inoltre spostato il menu sotto all'immagine e mi sono dedicata alla realizzazione del nuovo banner.



Figura 17 – Il menu nella prima versione (sopra il banner e con “News”) e il menu attuale

Una volta ottenute le immagini definitive, ho deciso di realizzare un banner dinamico, in contrasto con la pagina statica, in cui la parte a sinistra, quella occupata dal titolo, rimanesse ferma, e la parte a destra fosse invece caratterizzata dalla rotazione di tre immagini diverse. Mi sono servita in questo caso di alcune linee di codice Javascript che ho modificato per andare meglio incontro alle esigenze della pagina.

```

window.addEventListener?window.addEventListener('load',so_init,false):
window.attachEvent('onload',so_init);

var d=document,
    imgs = new Array(),
    zInterval = null,
    current=0,
    pause=false;

function so_init(){
    if(!d.getElementById || !d.createElement) return;

    css = d.createElement('link');
    css.setAttribute('href','slideshow2.css');
    css.setAttribute('rel','stylesheet');
    css.setAttribute('type','text/css');
    d.getElementsByTagName('head')[0].appendChild(css);

    imgs = d.getElementById('slideshow').getElementsByTagName('img');
    for(i=1;i<imgs.length;i++) imgs[i].xOpacity = 0;
    imgs[0].style.display = 'block';
    imgs[0].xOpacity = .99;

    setTimeout(so_xfade,3000);
}

function so_xfade(){
    cOpacity = imgs[current].xOpacity;
    nIndex = imgs[current+1]?current+1:0;
    nOpacity = imgs[nIndex].xOpacity;

    cOpacity-=.05;
    nOpacity+=.05;

```

```

imgs[nIndex].style.display = 'block';
imgs[current].xOpacity = cOpacity;
imgs[nIndex].xOpacity = nOpacity;

setOpacity(imgs[current]);
setOpacity(imgs[nIndex]);

if(cOpacity<=0) {
    imgs[current].style.display = 'none';
    current = nIndex;
    setTimeout(so_xfade,3000);
} else {
    setTimeout(so_xfade,50);
}

function setOpacity(obj) {
    if(obj.xOpacity>.99){
        obj.xOpacity = .99;
        return;
    }
    obj.style.opacity = obj.xOpacity;
    obj.style.MozOpacity = obj.xOpacity;
    obj.style.filter = 'alpha(opacity='+obj.xOpacity*100)+'';
}
}

```

Tabella 2 – Il codice Javascript usato per creare l’effetto “slide” e “fade” del banner

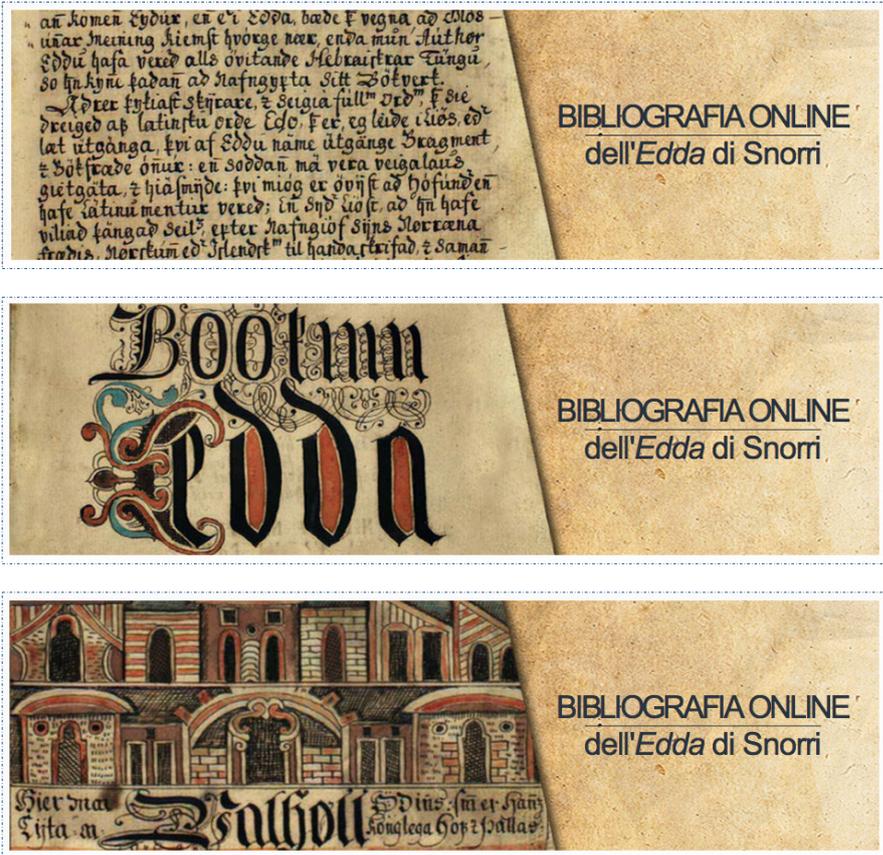


Figura 18 – Il banner nelle sue tre versioni

Successivamente ci sono state solo poche altre modifiche alla versione proposta come definitiva, come il cambiamento dell'immagine scelta per l'area "Contatti", che inizialmente era una scansione di una pagina in bianco e nero. Il sito completo e online è illustrato dalle immagini esemplificative qui sotto.

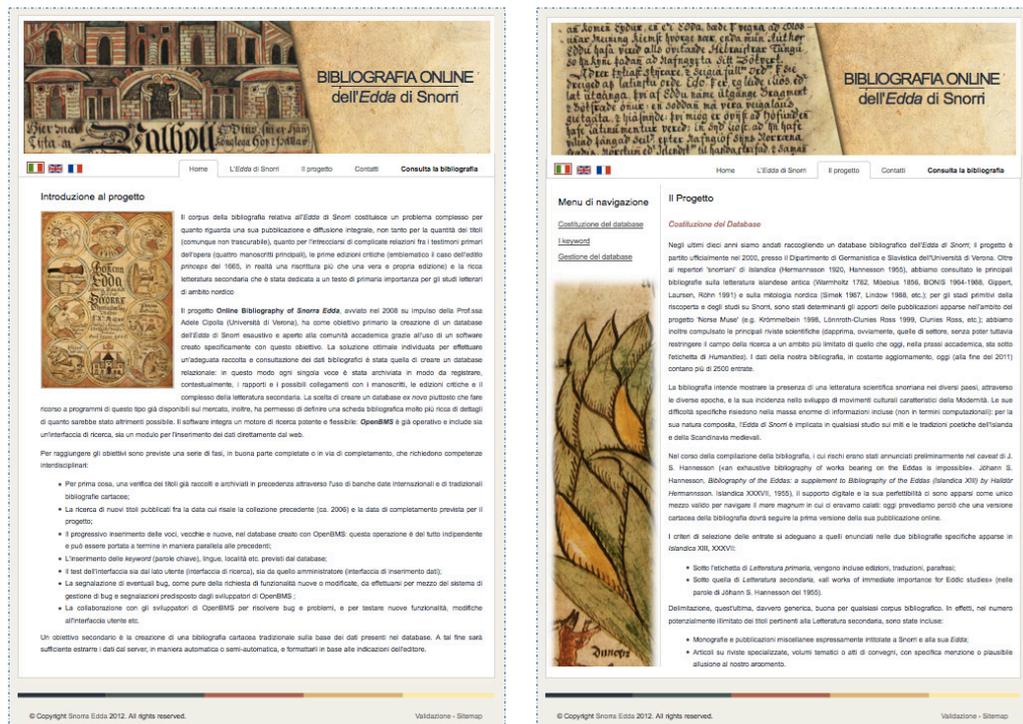


Figura 19 – Il sito alla versione attuale. L'Home page, la pagina sul progetto e i Contatti

3.2 – Metodologie, strumenti e contesto organizzativo

A differenza dello sviluppo del sito, la creazione della nuova interfaccia utente e le operazioni di debug della GUI (*Graphic User Interface*) e del back-end dell'applicazione si sono svolte, per tutta la durata del lavoro, in stretto contatto con gli sviluppatori del software e con i responsabili dei contenuti. A causa della distanza e della dispersione dei partecipanti tra Pisa, Verona e Torino ci siamo serviti di strumenti di sviluppo collaborativo e di comunicazioni molto frequenti, sia attraverso e-mail, sia dov'era possibile, direttamente sulle piattaforme predisposte.

Tra i vari strumenti che abbiamo utilizzato quello più sfruttato e che si è rivelato più efficace nel semplificare il nostro lavoro è stato sicuramente il server di test usato per provare l'applicazione nelle sue varie parti prima di implementarla sulla piattaforma ufficiale fornita dall'Università di Pisa. Il server scelto per eseguire i test è stato fornito dal servizio di server dedicati *Kimsufi*²⁵, del gruppo *OVH*²⁶.

Un altro strumento, potenzialmente molto utile, ma che abbiamo utilizzato in maniera minore, avendolo implementato quando il progetto era già in fase avanzata, è stato *GitHub*²⁷, un servizio web-based di hosting per progetti di sviluppo software, gratuito per i progetti open source. Il sito fornisce inoltre funzionalità di social networking, come ad esempio i feed e un grafico di rete per visualizzare come gli sviluppatori lavorano sulle loro versioni di un repository.

Per tutto il tempo dello sviluppo del software e dell'interfaccia ci siamo inoltre serviti di uno strumento semplice ma efficace: *Bugzilla*²⁸, un bugtracker online, dove previa registrazione è possibile utilizzare un ambiente utile alla catalogazione dei vari bug e alla loro gestione.

Lo sviluppo della nuova interfaccia prima, e il test e il debug della stessa poi, si sono svolti seguendo un percorso “a tappe”, in cui ogni nuova proposta, prima di essere accettata e implementata veniva discussa, testata e sottoposta al parere di tutti i membri del progetto. Sia che si trattasse di un *mockup* di una delle versioni

²⁵ “Kimsufi” – Gamma di server dedicati, <http://www.kimsufi.it>.

²⁶ “OVH” – Soluzioni server dedicati e hosting internet, <http://www.ovh.it>.

²⁷ “GitHub – Social coding”, <https://github.com>.

²⁸ “Bugzilla”, <http://www.bugzilla.org>.

dell'interfaccia, sia che fosse una proposta per la risoluzione di un bug, questa veniva prima presentata ad alcune figure chiave del progetto: Francesco Golia, il programmatore che ha sviluppato OpenBMS, Raffaele Cioffi, responsabile dei contenuti e il Professor Roberto Rosselli Del Turco, che ha supervisionato tutti gli sviluppi del progetto e che è stato il mio collegamento e primo supervisore oltre che per il tirocinio anche per tutto il tempo della mia collaborazione. Infine, per ogni risultato rilevante raggiunto, questo era presentato direttamente alla direttrice scientifica del progetto, la prof.ssa Adele Cipolla, per avere un parere definitivo.

Questo sistema ha consentito di lavorare in maniera comunitaria a tutti gli sviluppi, anche se non strettamente legati al proprio campo di competenza, in cui si è articolata la realizzazione del progetto “Online bibliography on Snorra Edda”, permettendo così di avere sempre una visione d'insieme, fondamentale per una buona riuscita del lavoro.

3.3 – L'evoluzione dell'interfaccia utente

Quando iniziai a collaborare con il progetto l'interfaccia di ricerca era alla versione 0.7.9 (vedi fig. 9), già funzionante ma ancora priva di alcuni widget essenziali.

Mancavano infatti diverse funzionalità per arrivare a una versione dell'interfaccia che fosse abbastanza soddisfacente da essere rilasciata pubblicamente. Tra le nuove funzioni che dovevano ancora essere implementate (anche a livello di database) c'erano per esempio:

- la possibilità per l'utente “comune”, di creare un profilo sull'applicazione con cui salvare le ricerche fatte e inviarne i risultati via e-mail, e per l'utente con privilegi amministrativi, di effettuare il login e accedere al back-end. Doveva quindi essere inserita nell'interfaccia un'area di “Login” e di “Opzioni utente” dove fosse possibile oltre che autenticarsi, anche cambiare le tipiche impostazioni di default (indirizzo e-mail, password, nome utente, etc.);
- le opzioni di visualizzazione, ossia un'area dove modificare le impostazioni legate strettamente alla ricerca di una voce, come una funzione di

ordinamento dei risultati (alfabetico, cronologico, etc.), la possibilità di vedere i risultati per pagina e di esportare sul proprio computer o stampare il risultato di una ricerca;

- la possibilità durante una ricerca di inserire più keyword differenti.

Per rispondere a queste e altre richieste abbiamo affrontato il problema da due punti di vista diversi: da una parte potevamo lavorare sulla versione 0.7.9 fino a renderla pubblicabile, dall'altra potevamo invece creare una nuova interfaccia, ripartendo da zero. Decidemmo così di portare avanti due progetti paralleli su due versioni differenti dell'interfaccia, denominate rispettivamente 0.8.0 e 2.0, e che io avrei lavorato su entrambi. Le due interfacce nascevano da una logica diversa: la 0.8.0 (o futura 1.0) mirava a rendere la 0.7.9 utilizzabile e completa di tutte le caratteristiche richieste senza andare però a modificare l'idea originaria. La 2.0 invece puntava a creare un'interfaccia del tutto diversa da quella di partenza, dandoci così la possibilità di rivederne completamente anche la parte grafica.

L'idea originaria era quindi di arrivare a sviluppare al più presto la 0.8.0 e di implementarla non appena pronta, lavorando contemporaneamente sulla 2.0, che se valutata poi più produttiva avrebbe a sua volta sostituito la versione 0.8.0 al momento del rilascio ufficiale.

3.3.1 – L'interfaccia versione 0.8.0

Lo sviluppo dell'interfaccia è avvenuto tramite una serie di mockup realizzati con il software Pencil²⁹, che ho di volta in volta sottoposto ai partecipanti al progetto così che potessero esprimere il loro parere e suggerire nuove soluzioni.

Il primo mockup che ho realizzato rimaneva molto fedele alla versione 0.7.9 dell'interfaccia; mi ero infatti limitata a implementare le funzioni mancanti senza fare valutazioni di sorta sulla chiarezza e sull'usabilità di questo tipo di layout. Il prototipo però si rivelò utile per iniziare a discutere sulla posizione che avrebbero dovuto occupare i widget.

²⁹ "Pencil Project", <http://pencil.evolus.vn/en-US/Home.aspx>.

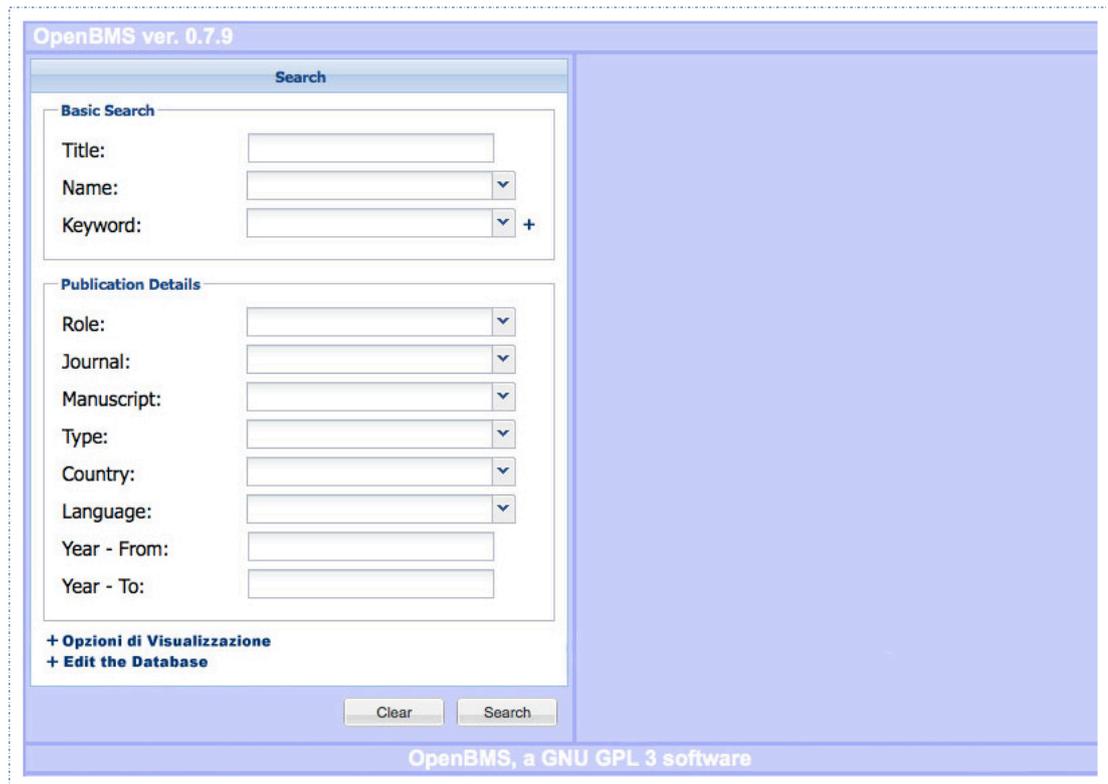


Figura 20 – Il primo mockup dell’interfaccia 0.8.0

Come si vede dalla figura, mi ero limitata a introdurre la possibilità di aggiungere keyword multiple tramite un pulsante “+” accanto al campo omonimo (il quale, se cliccato, avrebbe dovuto inserire un nuovo campo Keyword sotto il precedente), e due zone espandibili: una relativa alle opzioni di visualizzazione, l’altra alla modifica del database. Ancora non avevamo chiaro come volessimo far interagire l’utente (con privilegi amministrativi) con la banca dati, quindi avevo immaginato autonomamente un’area a espansione dentro alla quale fosse possibile fare il login per accedere al back-end.

Il prototipo proposto ovviamente non era soddisfacente sotto molti punti di vista, ma ciò che su tutto risultava poco funzionale era la posizione data all’area del login e l’uso del pulsante “+” accanto ad essa. “Edit the database” infatti, si decise che dovesse essere solo un collegamento che portava a un’area separata dall’interfaccia dove effettuare il login, e non una zona “a espansione” come invece le opzioni di visualizzazione (nelle quali, tra l’altro, era stata inserita la tastiera virtuale, che però così risultava introvabile per un utente impegnato in una prima ricerca).

The screenshot shows the OpenBMS 0.8.0 search interface. The title bar at the top reads 'OpenBMS' on the left and 'ver 0.8.0' on the right. The main content area is a search form with the following elements:

- Search section:**
 - Title:
 - Name:
 - Keyword: +
- Advanced search section:**
 - Role:
 - Journal:
 - Manuscript:
 - Type:
 - Country:
 - Language:
 - Year -From:
 - Year - To:

Below the search fields are two buttons: 'Clear' and 'Search'. At the bottom of the form, there is a link '+ Opzioni di visualizzazione'. The footer of the interface contains the text 'Edit the Database' on the left and 'OpenBMS, a GNU GPL 3 software' on the right.

Figura 21 – Il secondo mockup dell’interfaccia 0.8.0

La mia seconda proposta è stata quindi di spostare l’area di login nel “footer” della pagina, dove però, non solo non aveva ancora il risalto necessario, ma con la sua dicitura non permetteva di chiarirne la funzione.

A questo punto dello sviluppo dell’interfaccia la discussione si spostò nello specifico su quei punti che fino ad ora non era ben chiaro come volessimo sviluppare. In particolare era necessario chiarire quale sarebbe stato il funzionamento della sezione utente e di come questa dovesse essere collegata alla pagina di ricerca. Da questo confronto è stato chiarito che la “pagina utente”, in cui registrarsi, effettuare il login e dove in seguito modificare le proprie impostazioni dovesse essere in maniera chiara una sezione esterna rispetto all’interfaccia di ricerca. Decidemmo inoltre di dividere le categorie di utenti autenticati in due sottogruppi che fino a quel momento erano stati solo ventilati ma non definiti: l’utente semplice (capace di fare ricerche, salvarne il risultato e inviarlo), e l’utente/amministratore, che una volta autenticato avrebbe potuto accedere al back-end e modificare il database.

Questa distinzione sarebbe stata però effettuata nella pagina di login e non nell'interfaccia di ricerca, per ridurre il più possibile i collegamenti visualizzati nella pagina.

The screenshot displays the OpenBMS 0.8.0 web interface. At the top left, it shows 'ver 0.8.0'. The main title is 'OpenBMS', and at the top right, there is a 'Login / Register' link. The interface is divided into two main sections. The left section contains a search form with the following fields: 'Title:' (text input), 'Name:' (dropdown menu), 'Keyword:' (dropdown menu with a '+' icon), and an 'Advanced search' section with 'Role:', 'Journal:', 'Manuscript:', 'Type:', 'Country:', and 'Language:' (all dropdown menus), and 'Year - From:' and 'To:' (text inputs). Below these fields are three buttons: 'Virtual Keyboard', 'Clear', and 'Search'. The right section is currently empty. At the bottom of the left section, there is a 'View Options' section with 'Sort by:' and 'Results for page:' (both dropdown menus), and two buttons: 'Stamp' and 'Export'. At the bottom center of the page, it says 'OpenBMS, a GNU GPL 3 software'.

Figura 22 – Il terzo e ultimo mockup dell'interfaccia 0.8.0

Nel terzo e ultimo mockup ho quindi inserito l'area di Login in alto a destra, dove di solito si trova la zona di autenticazione nelle pagine web, rappresentata da un semplice collegamento "Login/Register", che se cliccato avrebbe portato a una sezione esterna unicamente legata alla gestione delle funzionalità "utente". Ho inoltre deciso di riportare la tastiera virtuale fuori dalle opzioni di visualizzazione per consentire di trovarla immediatamente. Le opzioni di visualizzazione sono poi state semplificate al massimo, rendendo disponibili solo funzioni essenziali e rendendo i due campi "Sort by" e "Results for page" non compilabili, ma legati a due menu a cascata a scelta multipla.

Una volta definita l'interfaccia sono quindi passata all'implementazione della stessa sul server di test servendomi di ExtJS per generarne i contenuti.

IMPLEMENTAZIONE DELL'INTERFACCIA SUL SERVER DI TEST

Per poter implementare l'interfaccia definita attraverso i mockup sul server di test, ho dovuto prima dotarmi di un client SVN e un client SQL (rispettivamente Versions³⁰ e Squirrel SQL³¹) per gestire i file dell'applicazione e il database. In seguito, dopo aver studiato la versione dell'interfaccia online (0.7.9) ho iniziato a replicarne la struttura di base per poi poterla modificare secondo quanto progettato grazie agli elementi messi a disposizione dalla libreria ExtJS.

In concreto ho modificato il file `openbmsfunctions.js` contenente quasi tutto il codice Javascript e la pagina `index.jsp` (Java Server Page) che lo utilizza. Come si vedrà, il codice HTML è piuttosto semplice: utilizzando gli elementi della libreria Javascript è sufficiente definire dei contenitori HTML ed assegnargli lo stesso attributo `id` usato nel file Javascript per identificare il contenuto di un elemento. Vediamo ora nel dettaglio il codice della pagina `index.jsp`.

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
  <link rel="stylesheet" type="text/css" href="ext/resources/css/ext-all.css">
  <link rel="stylesheet" type="text/css" href="base.css">
  <link rel="stylesheet" type="text/css" href="search.css">
  <script type="text/javascript" src="ext/adaptor/ext/ext-base.js"></script>
  <script type="text/javascript" src="ext/ext-all.js"></script>
  <script type="text/javascript" src="constants.js"></script>
  <script type="text/javascript" src="openbmsfunctions.js"></script>
  <title id="page-title">Online bibliography of Snorra Edda</title>

  <script type="text/javascript">
    Ext.BLANK_IMAGE_URL = 'ext/resources/images/default/s.gif';
    var resp;
    var formPanel;
    var vk_visible=false;
    var win;
    Ext.onReady(build_stores);
  </script>
  <%
boolean appenaSloggato= false;
if(request.getParameter("logout")!=null ){
session.invalidate();
appenaSloggato=true;
}
boolean utenteLoggato = ( request.getRemoteUser()!=null  && !
appenaSloggato ); %>
</head>
```

³⁰ “Versions – SVN client”, <http://versionsapp.com>.

³¹ “Squirrel SQL”, <http://squirrel-sql.sourceforge.net>.

```

<body>

  <div id="header" align="left">
    <span id="newheader">OpenBMS ver. ${initParam.swversion}</span>
    <span id="newheader2"><a href="/openbms_test/admin/"
>Login</a><%if (utenteLoggato) {%>
      || Logged in as: <%= request.getRemoteUser() %> ||
      <a href="admin/index.jsp">Admin Area</a> ||
      <a href="index.jsp?logout=true" > Logout</a>
      <%}%></span>
    </div>
    <div id="content">
      <div id="content-row">
        <div id="menu" align="center" style="">
          <div id="search"></div>
          <div class="x-hidden" id="keyboard-win">
            <div class="x-window-header">Hello Dialog</div>
            <div id="keyboard-panel">

<iframe src="http://www.virtualkeyboard.ws/plugin.shtml"
frameborder="0" marginheight="0" marginwidth="0" scrolling="no"
width="350" height="280" style="border:thin outset">Free virtual
keyboard plugin from <a href="http://www.virtualkeyboard.ws"
target="_blank">virtualkeyboard.ws</a></iframe>
          </div>
          </div>
          <div class="optionstyle">
            <div id="ext-gen4a" class="x-panel-header x-unselectable">View
Options</div>
            <div id="opt">
              <div id="ext-gen363" class="x-form-field-wrap x-form-field-
trigger-wrap" class="sort">Sort by: <input id="inputone" type="text"
size="24" maxlength="183"/>
              
              </div> <br/>
              <div id="ext-gen364" class="x-form-field-wrap x-form-field-
trigger-wrap" class="result">Results for page: <input id="inputtwo"
type="text" size="24" maxlength="183"/>
              
              </div>
              <span id="st-ex">
                <a class="stp" href="#">Stamp<a/>
                <a class="exp" href="#">Export<a/></span>
              </div>
              </div>
              </div>
              <div id="core" style=""></div>
            </div>
          </div>
          <div id="footer" align="center" >OpenBMS, a GNU GPL 3 software</div>

</body>
</html>

```

Tabella 3 – Il codice HTML della pagina Index.jsp

Nella pagina ho aggiunto i contenitori delle Opzioni di visualizzazione e ho modificato parte della zona di ricerca e dell'header per adattarla al nuovo layout.

Il codice Javascript chiarisce ulteriormente come ho inserito all'interno dei <div> i vari elementi dell'interfaccia. La libreria ExtJS infatti mette a disposizione vari tipi di layout³², cioè vari tipi di strutture predefinite per organizzare i contenuti della pagina. Il layout di tipo “formpanel³³” da me scelto per il progetto e in particolare per organizzare i campi di ricerca sulla sinistra, ad esempio fornisce un container standard per i *form*. Esso crea automaticamente una BasicForm per la gestione di tutti gli oggetti Ext.form.field che vengono aggiunti come discendenti del pannello, facilitandone la gestione.

```
function build_stores(){
    define_stores();
    formPanel = new Ext.form.FormPanel({
        standardSubmit: true,
        id: 'form-panel',
        formid: 'formTag',
        url: APP_CONTEXT+ '/TitleSearch',
        method: 'POST',
        waitMsgTarget: true,
        frame:false,
        title: 'Search',
        bodyStyle:'padding: 5px',
        width:400,
        defaults: {width: 380},

        items: [{
            xtype:'fieldset',
            title:'Basic Search',
            collapsible: false,
            autoHeight:true,
            defaults: {width: 200},
            defaultType: 'textfield',
            items: [{
                fieldLabel: 'Title',
                name: 'Title',
                id: 'Title',
                width: 183,
                allowBlank: true
            },
            person_combo,
            keyword_combo]
        }, {
```

³² Una panoramica dei layout e degli elementi messi a disposizione da ExtJS è consultabile alla pagina: <http://docs.sencha.com/ext-js/4-0/#!/example>.

³³ “Sencha Docs – ext.form.Panel”, <http://docs.sencha.com/ext-js/4-0/#!/api/Ext.form.Panel>.

```

        xtype:'fieldset',
        title: 'Advanced search'
        collapsible: false,
        autoHeight:true,
        defaults: {width: 200},
        defaultType: 'textfield',
        items: [
            role_combo,
            journal_combo,
            editiontype_combo,
            country_combo,
            language_combo,

            {
                fieldLabel: 'Year - From',
                name: 'fromYear',
                id: 'fromYear',
                width: 183,
                allowBlank: true
            },
            {
                fieldLabel: 'Year - To',
                name: 'toYear',
                id: 'toYear',
                width: 183,
                allowBlank: true
            },
            {
                inputType: 'hidden',
                id: 'submitbutton',
                name: 'myhiddenbutton',
                value: 'hiddenvalue'
            },
            {
                inputType: 'hidden',
                id: 'action',
                name: 'action',
                value: 'NEWSEARCH'
            }
        ]
    }],
    buttons: [
        {
            text: 'Virtual Keyboard',
            handler: function(){
                if(!vk_visible){
                    vk_visible=true;
                    if(!win){
                        win = new Ext.Window({
                            applyTo:'keyboard-win',
                            title: 'virtual keyboard',
                            layout:'fit',
                            width:380,
                            height:350,
                            x: 150,
                            y: 150,
                            closeAction:'hide',
                            plain: true,

```

```

        items: new Ext.Panel({
            applyTo: 'keyboard-panel',
            border:false
        }),
        buttons: [{
            text: 'Close',
            handler: function(){
                win.hide();
                vk_visible=false;
            }
        }
    ]});
    });
    win.show(this);
}
}
}, {
    text: 'Search',
    handler: richiestaAsincrona
},
{
    text: 'Clear',
    handler: function() {
        formPanel.getForm().reset();
    }
}
}
});
formPanel.render('search');
}

```

Tabella 4 – Il codice Javascript usato per generare l’interfaccia.

Nel codice si vede chiaramente come ExtJS permetta di definire valori e contenuti direttamente attraverso Javascript, consentendo allo stesso tempo il collegamento con le funzioni usate per la ricerca. Come si può vedere le opzioni di visualizzazione non sono ancora collegate al database da nessuna funzione ma sono solo implementate graficamente, questo in attesa dell’approvazione definitiva dell’interfaccia per renderla funzionante.

Rispetto alla versione 0.7.9 dell’interfaccia ho deciso di utilizzare degli elementi della libreria Javascript differenti per due motivi, *in primis* ero interessata a comprendere le funzionalità di questo framework sotto più aspetti, e in secondo luogo perché dall’implementazione della versione 0.7.9 ExtJS era stato aggiornato ed erano stati introdotti elementi più specifici rispetto al semplice “GridPanel” utilizzato nella scorsa versione.

ExtJS però, per quanto ricco di elementi potenzialmente combinabili tra loro in innumerevoli abbinamenti, difetta di flessibilità nell'ambito della personalizzazione. I fogli di stile standard associati al framework sono piuttosto scarni, e, a causa della caratteristica della libreria di assegnare un *id* ad ogni elemento e di sfruttare ampiamente il metodo di attribuzione di caratteristiche tramite la discendenza dei *tag*, anche di difficile gestione. Per questo, andando a modificare parti del layout dell'interfaccia ho dovuto attribuire nuove *classi* e *id* agli elementi interessati (rovinando anche in parte la pulizia della pagina HTML) così da forzare ExtJS ad accettare i nuovi comandi CSS.

L'immagine qui sotto illustra l'interfaccia attualmente online sul server di test³⁴.

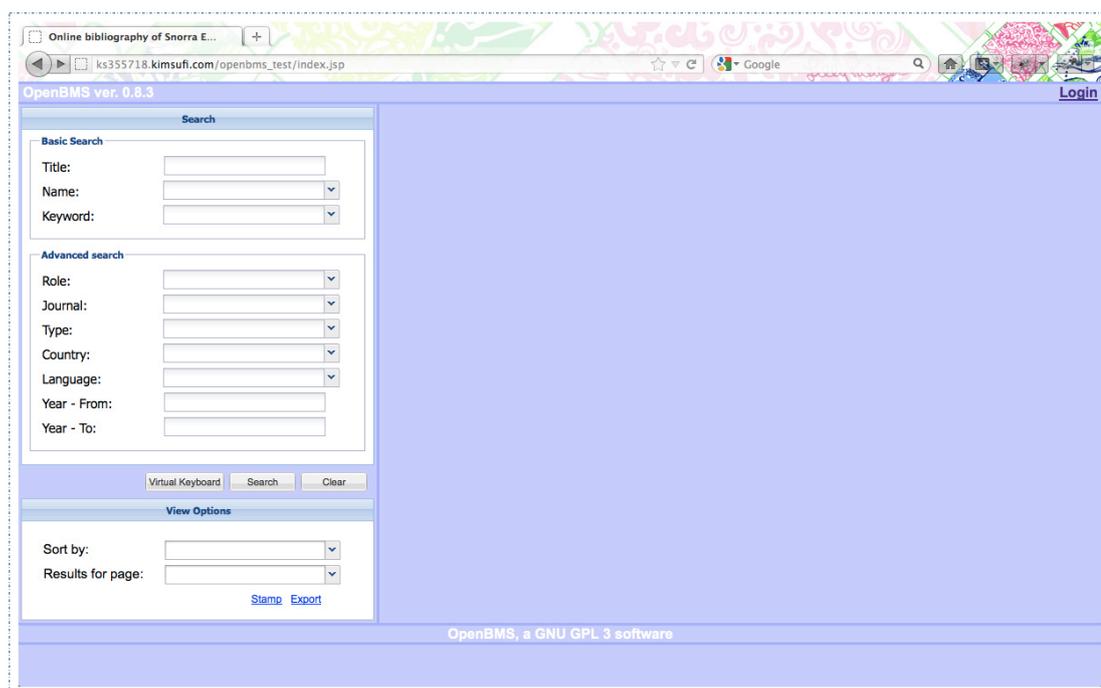


Figura 23 – L'interfaccia online sul server di test

³⁴ “Online bibliography of Snorra Edda”, www.ks355718.kimsufi.com/openbms_test/

3.3.2 – L’interfaccia versione 2.0

Una delle proposte avanzate durante la fase di analisi e di elaborazione dell’interfaccia 2.0 è stata quella di suddividere le possibilità di ricerca su tre livelli in base alla specificità voluta, creando così tre interfacce separate chiamate: “Base search”, “Advanced search” ed “Expert search”.

Seguendo questa proposta ho realizzato i primi tre mockup, (uno per ogni livello) in cui le tre interfacce erano contrassegnate da etichette, come le schede in un browser. La ricerca base avrebbe avuto un campo unico in stile motore di ricerca, ma senza la funzione di auto-completamento. L'applicazione avrebbe poi cercato in tutto il database le occorrenze e avrebbe proposto dei risultati organizzati per entità (es: Names, Keywords, Titles, etc.). La ricerca avanzata invece avrebbe avuto a disposizione un numero ristretto di campi, tre, alcuni con completamento automatico altri a testo libero, e la ricerca approfondita invece avrebbe avuto un’interfaccia con tutti i campi disponibili per la compilazione.

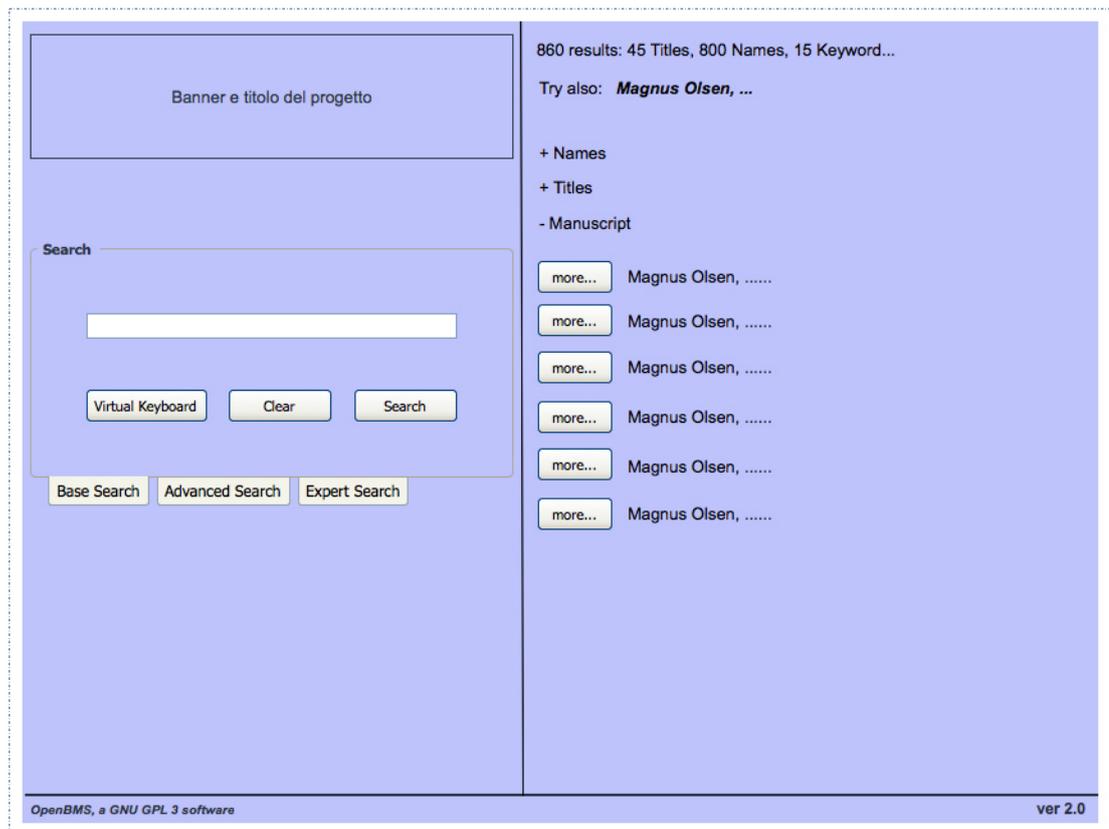


Figura 24 – Il primo mockup dell’interfaccia 2.0 (Base Search)

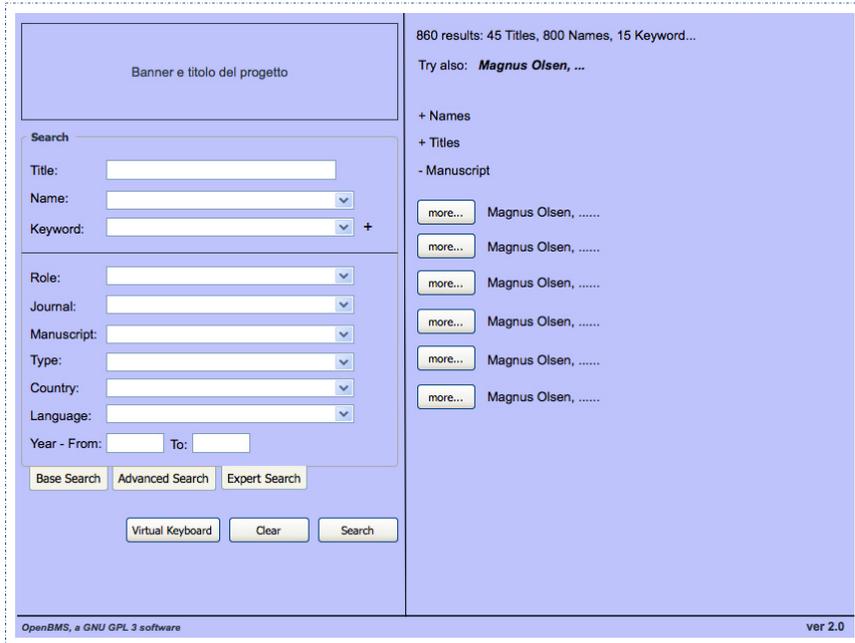
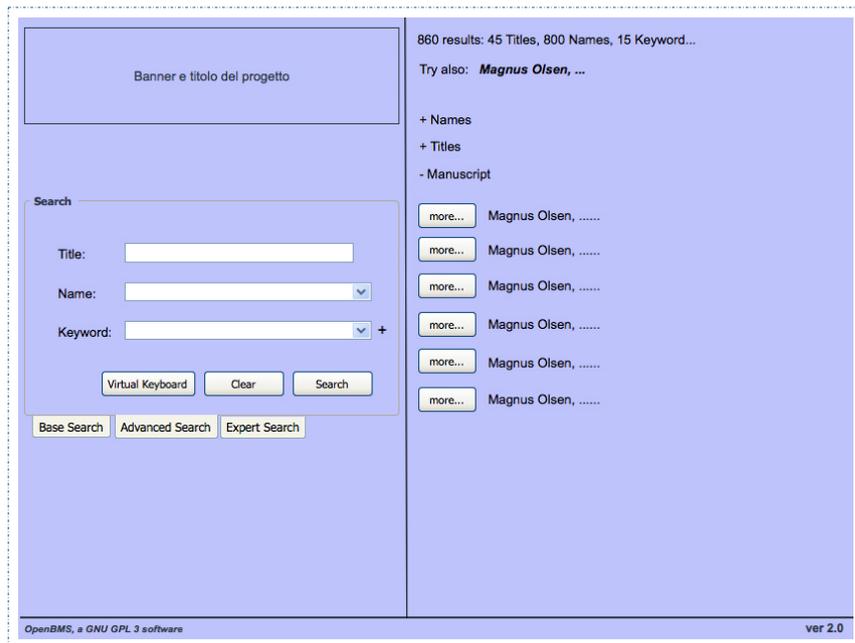


Figura 25 – Il primo mockup dell’interfaccia 2.0 (Advanced ed Expert Search)

La struttura da me presentata era però molto vicina al layout a due colonne della versione 0.79, che mal si adattava alla nuova suddivisione su livelli. Come disposizione, infatti, era ottimale per i tipi di ricerca più avanzati, ma per la ricerca base evidenziava uno spreco di spazio. Spazio che era sacrificato dai parametri di ricerca, togliendone al raggruppamento dei risultati per tipologia. Inoltre non c’era una postazione soddisfacente per il pannello delle opzioni di visualizzazione (vedi fig. 24 e 25).

Cogliendo appieno i limiti di questo layout abbiamo provato a sperimentarne una versione “a righe”, alternativa a quella “a colonne” adottata fino ad adesso. Ho così posizionato in cima alla pagina i due pulsanti delle opzioni (utente e di visualizzazione) e le etichette relative al tipo di ricerca, con sotto un'area riservata ai campi compilabili, diversi a seconda del tipo selezionato. Così facendo abbiamo recuperato spazio lateralmente da usare per un layout a due colonne solo nella zona di visualizzazione dei risultati.

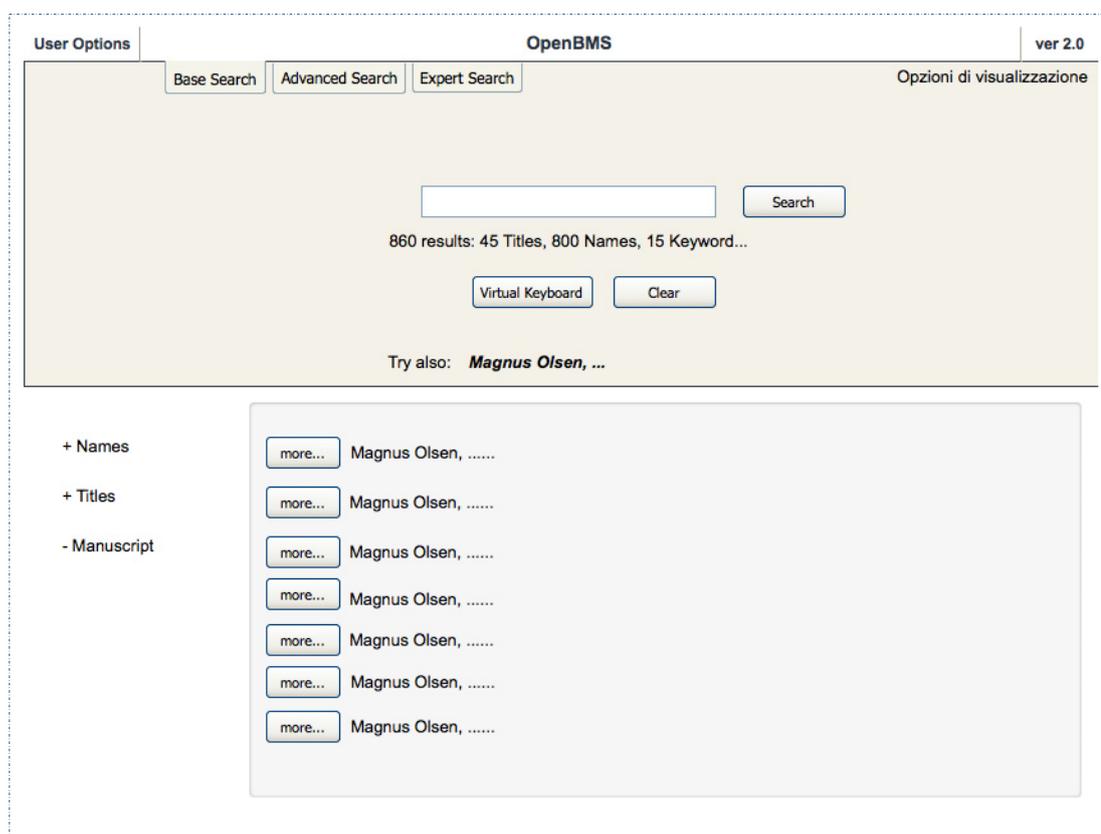


Figura 26 – Il secondo mockup dell’interfaccia 2.0 (Base Search)

Un layout di questo tipo, a differenza del precedente, era più efficace, specialmente per la ricerca base, ma presentava ancora alcuni problemi per le ricerche più avanzate. Il difetto principale era l'ingombro in verticale dei campi nel caso in cui questi fossero di numero elevato. Avremmo dovuto quindi ipotizzare una sorta di auto-hide del pannello dei campi di ricerca, che permettesse di leggere comodamente i primi risultati senza dover scorrere la pagina, ma che altrettanto agevolmente permettesse di cambiare i criteri nel caso di una nuova ricerca. Questa soluzione però, per quanto praticabile non era semplice da raggiungere.

Una delle possibilità infatti, era di creare uno pseudo-linguaggio che al momento della restituzione dei risultati riportasse sulla pagina della ricerca base e creasse una stringa delle parole inserite nei vari campi che sarebbe stata visualizzata nel campo di ricerca unico della “Base Search”.

A fronte della mia proposta successiva però, più pratica e di più semplice realizzazione, si decise di accantonare questa possibilità. La mia idea, che ha poi portato alla versione attuale dell’interfaccia 2.0 (ancora in valutazione) era infatti di un modello non a schede, ma detto “a espansione”. Come si vede nell’ultimo mockup, mantenendo il layout “a righe” ho inserito nella zona di ricerca solo tre campi; rispettivamente Title, Name e Keyword, eliminando così la tripartizione tra Base/Advanced ed Expert Search, e mettendo sotto a questi una zona cliccabile che si sarebbe espansa mostrando anche i campi rimanenti (Role, Journal, etc.). Inoltre, nonostante l’eliminazione delle “etichette” dei tre tipi di ricerca avevo mantenuto l’idea per le opzioni di visualizzazione, spostate in questa versione in un pannello ad espansione identificato, appunto, da un’etichetta, posto sotto l’area di ricerca.

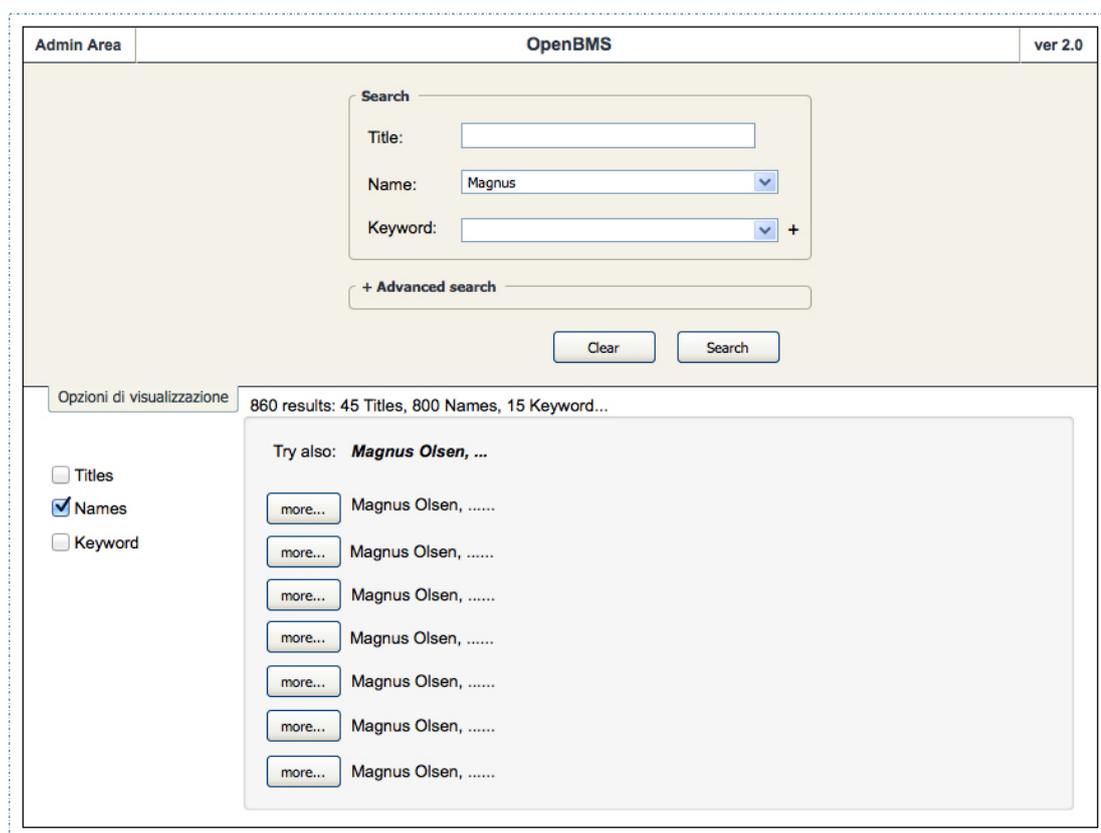


Figura 27 – L’interfaccia 2.0 nella versione attuale

3.4 – Debug della GUI e del back-end

La parte finale della mia collaborazione con il progetto è consistita soprattutto nel testing e nell'analisi di alcuni dei bug dell'interfaccia utente (GUI) e del back-end. Per tutto il periodo dello sviluppo del software e dell'interfaccia ci siamo serviti di uno strumento semplice ma efficace per tenere traccia dei problemi da risolvere: *Bugzilla*, un bugtracker online, dove previa registrazione è possibile utilizzare un ambiente utile alla catalogazione dei vari bug e alla loro gestione. Ogni bug è infatti identificato da un numero progressivo, è possibile assegnargli una priorità e uno stato (“Confirmed”, “In progress”, “Resolved”). Si può esportare la lista dei bug in formato XML o solo testo e sotto ogni bug è disponibile anche una zona in cui inserire commenti per discutere sull'eventuale soluzione, funzionalità molto utile vista la distanza fisica tra i vari partecipanti al progetto.

Tutti i bug rilevati durante la fase di sviluppo e di test sono stati “tracciati” con *Bugzilla* e man mano risolti attraverso le cinque fasi canoniche di risoluzione di un bug: identificazione dell'errore, replicazione dello stesso, individuazione della causa, progettazione di una correzione per il bug e implementazione e testing della suddetta correzione. Al momento attuale sono rimasti solo circa venti bug ancora attivi, ma quasi tutti in via di risoluzione.

Tra i bug più interessanti ai quali mi sono dedicata, ce ne sono stati un gruppo riguardanti l'interfaccia di ricerca che più che bug tecnici o errori di programmazione erano scelte da compiere in ambito progettuale, comunque segnalate in *Bugzilla* come problemi da risolvere, che sono stati eliminati attraverso l'elaborazione delle nuove versioni dell'interfaccia di ricerca. Il debug dei problemi incontrati e la progettazione dei vari aspetti del software infatti, non sono stati momenti uno conseguente all'altro, ma fasi che sono procedute parallelamente, lasciando la priorità di volta in volta alla necessità più immediata.

Una delle prime difficoltà incontrate è stata proprio come rappresentare i dati bibliografici; se per intero, in parte, in un pannello a espansione o simili. Alla fine si è deciso di mostrare solo la prima parte della voce bibliografica, ordinata secondo lo stile Chicago, lasciando che il resto dei dati fosse mostrato solo cliccando su un pulsante che è poi diventato il “more” posto accanto ad ogni voce (vedi fig. 10)

tramite la funzione “Toggle” qui sotto presentata in una mia versione leggermente modificata.

```
function toggleVisibility(id) {
    var el = Ext.get(id);
    var button = Ext.get(id.replace("_", "_button_"));
    if (button.dom.innerHTML == 'more...') button.dom.innerHTML = 'less';
    else button.dom.innerHTML = 'more...';
    el.setVisibilityMode(2);
    el.toggle();
}
```

Tabella 5 – Funzione “Toggle” usata per aprire e chiudere una scheda con il pulsante “more...”

Un altro aspetto simile su cui abbiamo lavorato è stato il problema delle keyword multiple che ci siamo resi conto essere necessarie per affinare ricerche che in alcuni casi sarebbero state troppo vaghe. La soluzione dell’incremento dei campi cliccando sul pulsante “+” mostrata nei mockup della versione 0.8.0 dell’interfaccia, è stata solo l’ultima dopo varie proposte (tra l’altro ancora da implementare ma realizzabile con una funzione simile a “Toggle”), tra cui il metodo “a tag”, con le varie keyword separate da una virgola in un unico campo. In quel modo però, le parole chiave non avrebbero beneficiato dell’auto-completamento dato dal menu a tendina.

Ci sono poi stati problemi di natura più tecnica, ma risolti in modo semplice e senza la necessità di intervenire sul codice. Un esempio è quello denominato “delle url multiple”. Nelle voci in cui c’era più di un collegamento a una pagina web esterna, questi venivano visualizzati come un link unico e quindi non funzionante. La cosa si è risolta spostando ogni collegamento su una riga separata, impedendo così a ExtJS di vederli come un blocco unico.

Bug in cui invece è stato necessario intervenire sul codice sono stati invece quelli che hanno interessato la ricerca vera e propria, come quello per cui la ricerca, se senza risultati, non solo non restituiva nulla, ma non avvisava neppure della fine della stessa. Per questo motivo è stato introdotto nell’*header*³⁵ della pagina di risposta un campo, *results*, che riportasse il numero di risultati trovati con la ricerca. In questo modo la funzione di ricerca una volta modificata, se rilevato il campo con valore

³⁵ L’**header** è quella parte del pacchetto che contiene informazioni di controllo.

diverso da zero avrebbe popolato la zona dei risultati correttamente, in caso contrario l'avrebbe popolata con zero risultati, lasciandola giustamente vuota e avrebbe mostrato un pop-up per avvisare che la ricerca non aveva dato risultati.

```

Intestazioni di risposta
Accept-Ranges none
Connection Keep-Alive
Content-Length 88
Content-Type text/html;charset=UTF-8
Date Fri, 29 Jun 2012 17:52:53 GMT
Keep-Alive timeout=15, max=100
Server Apache/2.2.3 (Red Hat) mod_ssl/2.2.3 OpenSSL/0.9.8e-fips-rhel5 mod_fcgid/2.3.7 DAV/2 mod_jk/1.2.28 PHP/5.2.17 with Suhosin-Patch mod_python/3.2.8 Python/2.4.3 mod_perl/2.0.4 Perl/v5.8.8
Set-Cookie JSESSIONID=E0C95AC7BD9F8B1818D004489F64436A; Path=/bibliografia
results 0

```

Figura 28 – L’header della pagina di risposta con il campo “results” uguale a zero

```

function research(){
    var editiontype_value = Ext.get('Editiontype').dom.value;
    var role_value = Ext.get('Role').dom.value;
    var person_value = Ext.get('Person').dom.value;
    var language_value = Ext.get('Language').dom.value;
    var keyword_value = Ext.get('Keyword').dom.value;
    var journal_value = Ext.get('Journal').dom.value;
    var country_value = Ext.get('Country').dom.value;
    var title_value = Ext.get('Title').dom.value;
    var fromYear_value = Ext.get('fromYear').dom.value;
    var toYear_value = Ext.get('toYear').dom.value;

    Ext.Ajax.request({
        //form: formPanel.getForm() ,
        method: 'POST',
        url: APP_CONTEXT+"/TitleSearch",
        params: {action:'NEWSEARCH',
            Editiontype: editiontype_value,
            Role: role_value,
            Person: person_value,
            Language: language_value,
            Keyword: keyword_value,
            Journal: journal_value,
            Country: country_value,
            Title: title_value,
            fromYear: fromYear_value,
            toYear: toYear_value
        },
        success: function(o){
            resultdiv = Ext.get('core');
            resultdiv.update(o.responseText);
            if(o.getResponseHeader("results") == 0){
                alert("No results");
            }
        }
    });
}

```

Tabella 6 – La funzione di ricerca e la parte aggiunta per notificare una ricerca senza risultati

IL FUTURO DEL PROGETTO

4.1 – Sviluppi futuri e obiettivi a breve e lungo termine

Il progetto, ormai in via di completamento, dovrà comunque affrontare alcune altre fasi per potersi considerare ultimato e pronto alla presentazione pubblica. Presentazione che probabilmente avverrà con un intervento della coordinatrice del progetto, la professoressa Maria Adele Cipolla dell'Università di Verona, durante una delle prossime conferenze della Saga Society.

Il lavoro svolto fino ad ora ha richiesto numerose competenze interdisciplinari da parte di tutti i partecipanti, competenze che saranno ora necessarie anche a completarne le ultime fasi:

- sarà necessario infatti fare un'ulteriore ricerca di nuovi titoli, pubblicati fra la data cui risale la collezione precedente (ca. 2006), e oggi, così da aggiornare il più possibile il database per il momento della pubblicazione;
- andranno inserite delle nuove keyword (parole chiave) su lingue, località etc. previste al momento della progettazione ma non ancora attive;
- dovremo implementare l'interfaccia 0.8.0 sul server ufficiale e finire di testarla, così da rinominarla 1.0 e procedere al rilascio al pubblico del software funzionante. Parallelamente continuerà lo studio dell'interfaccia 2.0, che se ritenuta idonea potrebbe poi sostituire a sua volta la versione 1.0;
- il sito della presentazione del progetto sarà probabilmente trasferito su CMS (Wordpress³⁶) se si riterrà necessaria l'introduzione di una sezione news, così che possa essere gestita più agevolmente che dal sito attuale;
- un obiettivo secondario, ma non meno importante, sarà la creazione di una bibliografia cartacea "tradizionale" grazie ai dati presenti nel database. Per realizzarla sarà sufficiente estrarre i dati dal server, in maniera più o meno automatica, e formattarli secondo lo stile bibliografico scelto anche per la presentazione dei dati online: Chicago.

³⁶ "Wordpress", <http://it.wordpress.org/>

CONCLUSIONI

Il presente lavoro è partito con un chiaro obiettivo: apportare un contributo effettivo a un progetto già in corso di sviluppo, inserendomi all'interno di un team strutturato e cercando così di collaborare alla realizzazione un prodotto caratterizzato da precisione, versatilità e facilità d'utilizzo.

Il progetto "Online bibliography of Snorra Edda" può inoltre essere una chiara e tangibile dimostrazione di quello che l'informatica applicata alle discipline umanistiche consente di ottenere: si tratta di un messaggio rivolto a tutti coloro che ancora sottovalutano il potenziale delle nuove tecnologie, in particolare quelle *web-based*.

Nell'approccio multidisciplinare è infatti possibile trovare uno dei punti di forza di questo progetto: sono stati utilizzati gli strumenti messi a disposizione dall'informatica e dalla filologia per creare un prodotto completo e di qualità. Una qualità che è garantita da diverse caratteristiche: il rigore con il quale sono state condotte le ricerche e raccolti i dati bibliografici, la sequenzialità delle fasi di lavoro e la facilità con la quale è possibile utilizzare il sistema (sia per gli utenti che vogliono collaborare alla creazione del corpus, sia per gli utilizzatori finali che devono solo usufruirne).

È doveroso inoltre sottolineare che quando si cerca di elaborare un progetto innovativo, questo non può mai dirsi veramente "completo". La "Bibliografia online dell'Edda di Snorri" è infatti a tutti gli effetti "aperta", realizzata anche in vista di successivi miglioramenti e aggiornamenti che saranno apportati nel corso del tempo.

Gli ostacoli incontrati non sono stati pochi poiché, comunque, mi sono confrontata con una realtà che non mi era familiare: ci sono state difficoltà nella fase di progettazione, sia del sito sia dell'interfaccia, specialmente nel mettere a punto un modello che potesse soddisfare ogni tipo di esigenza e punto di vista.

Dalle complessità affrontate sono tuttavia giunta a diverse conclusioni: *in primis* ho capito quanto sia importante il confronto e la discussione per la messa a punto di un

progetto di questo genere. Confrontarsi con opinioni ed esperienze diverse inizialmente può disorientare, ma consente anche di non rimanere chiusi nella propria visione, con vantaggi per la progettazione e la realizzazione di un prodotto in grado di soddisfare ogni tipologia di utente. Ho capito anche che è bene non sottovalutare questa fase di preparazione che precede la programmazione vera e propria: iniziare infatti la fase di implementazione senza aver valutato e messo a punto un modello può portare solo a fare errori di valutazione e a sprecare tempo, risorse ed energie.

Dalla necessità di realizzare un software che fosse cross-browser e accessibile, invece ho compreso quanto sia assolutamente necessario utilizzare gli standard internazionali e le linee guida per la programmazione e che, nonostante tutto, il lavoro di progettazione è un lavoro che prevede continui cambiamenti e messe in discussione.

Personalmente questo progetto ha arricchito le mie conoscenze sotto ogni punto di vista: sono entrata in contatto con un mondo, quello della letteratura nordica e dello studio dei testi antichi, per me finora sconosciuto e che, negli ultimi tempi, sta conoscendo uno sviluppo continuo grazie alle nuove tecnologie. Questo progetto mi ha permesso anche di approfondire le mie conoscenze informatiche, accostandomi alla programmazione Javascript in maniera più professionale e avvicinandomi per la prima volta alla progettazione e gestione di database a un livello molto più profondo di quello didattico. Considero quindi questo lavoro un valido esempio di applicazione pratica delle conoscenze apprese durante il mio corso di studi: le conoscenze informatiche possono convivere con le conoscenze umanistiche e soprattutto possono aiutare nella loro divulgazione, rendendole accessibili e condivisibili.

BIBLIOGRAFIA

- Bellows, Henry Adams. *The Poetic Edda. Translated from the Icelandic with an Introduction and Notes*. New York: American-Scandinavian Foundation, 1923.
- Hannesson, Jóhann S. "Bibliography of the Eddas: a supplement to bibliography of the Eddas" (Islandica XIII) by Halldór Hermannsson, Islandica XXXVII, 1955.
- Isnardi, Gianna Chiesa. *I Miti nordici*. Milano: Longanesi, 1991.
- Isnardi, Gianna Chiesa (a cura di). *Snorri Sturluson: Edda di Snorri*. Milano: Tea, 2003.
- Faulkes, Anthony. *Snorri Sturluson: Edda*. Londra: Dent, 1995.
- Mullet, Kevin, and Sano, Darrel. *Designing visual interface*. NJ: Prentice Hall, 1995.
- Niederst, Jennifer. *Web Design in a nutshell: a desktop quick reference*. Sebastopol: O'Reilly, 2006.
- Nielsen, Jacob. *Web Usability*. Tradotto da Walter Vanini. Milano: Apogeo 2000.
- Nikolassy, Riccardo. *Manuale di Javascript*. Milano: Hoepli, 2007.
- Polillo, Roberto. *Il check-up dei siti web*. Milano: Apogeo, 2004.
- Pollock, Jhon. *Fondamenti di Javascript*. Milano: McGraw-Hill, 2004.

SITOGRAFIA

Tutti i siti sono stati visitati l'ultima volta nel mese di Giugno 2011.

- Software “Adobe Photoshop”:
<http://www.adobe.com/it/products/photoshopfamily.html>.
- Libreria Javascript “ExtJS”: <http://extjs.com>.
- Oracle technology networks for Java developers: <http://java.sun.com>.
- Java Server Pages Standard Tag Library: <http://java.sun.com/products/jsp/jstl>.
- Hibernate – Jboss Community: <http://www.hibernate.org>.
- Standard W3C, World Wide Web Consortium: <http://www.w3.org>.
- Linee guida per l’accessibilità: <http://www.w3.org/TR/WCAG>.
- CSS references: <http://www.w3schools.com/cssref/default.asp>.
- CSS3.info: <http://www.css3.info>.
- Wordpress (Content Management System): <http://it.wordpress.org>.
- Kimsufi, Gamma di server dedicati: <http://www.kimsufi.it>.
- OVH, Gruppo di hosting internet: <http://www.ovh.it>.
- GitHub: <https://github.com>.

- Bugzilla – bugtracker: <http://www.bugzilla.org>.
- Firebug – web development tool: <http://getfirebug.com>.
- Pencil project – The opensource GUI prototype tool:
<http://pencil.evolus.vn/en-US/Home.aspx>.
- Versions – SVN client for Mac: <http://versionsapp.com>.
- Squirrel SQL – SQL client: <http://squirrel-sql.sourceforge.net>.
- Documentazione API su ExtJS: <http://docs.sencha.com/ext-js/4-0/#!/api>.
- The “Chicago” manual of style:
http://www.chicagomanualofstyle.org/tools_citationguide.html.
- The “APA” (American Psychological Association) style:
<http://www.apastyle.org>.
- Dell’Orso, Francesco. *Citazioni bibliografiche secondo il Chicago manual of style (e con appunti da “Come si fa una tesi di laurea” di Umberto Eco)*:
<http://www.aib.it/aib/contr/dellorso1.htm>.
- Gnoli, Claudio. *Le citazioni bibliografiche, una guida introduttiva per interpretare e redigere correttamente le citazioni delle fonti bibliografiche*:
<http://www.aib.it/aib/contr/gnoli2.htm>.

