UNIVERSITÀ DI PISA

Corso di Laurea in Informatica Umanistica

RELAZIONE

Progetto Comitatus: un database anagrafico di Marchesi Conti e Visconti nel Regno Italico (secc. IX-XII)

Candidato: *Paola Andriani*

Relatore: *Paolo Rossi*

Correlatore: *Maria Simi*

Anno Accademico 2008-2009

a mamma e papà (e fratello)

1. Introduzione

1.1. Il progetto Comitatus

Il progetto è nato con l'intenzione di definire e realizzare una base di dati inerenti al periodo storico fra il 774 d.C. e il 1115 d.C., ossia fra l'avvento dei regni "barbari" (longobardi prima, franchi poi) e la costituzione delle prime entità comunali.

In particolare, si vuole rappresentare la situazione politica e sociale di tale periodo (tra il IX e l'XII sec.), ponendo l'attenzione sulle divisioni territoriali in contee e marchesati, sugli amministratori di tali territori (vassalli nominati dal re, famiglie indipendenti, vescovi nominati dal papa), sulle loro famiglie, sulle fonti che li citano.

Si tratta di un database che copre quindi circa un migliaio di persone, distribuite tra duecento comitati e marche, delle quali si dovranno dare, ove possibile, indicazioni anagrafiche, legami di parentela e appartenenza familiare e repertorio bibliografico relativo, articolato fra studi e fonti. Il progetto prevede la realizzazione di un'interfaccia che renda possibile l'inserimento di tali dati da parte di studiosi e ricercatori interessati all'argomento.

1.2. La metodologia e scelta degli strumenti

Il percorso metodologico per la realizzazione della base di dati è passato attraverso tre fasi:

1. progettazione concettuale: analisi dei requisiti, definizione dei bisogni informativi del committente, modellizzazione dei dati;
2. progettazione logica: la traduzione delle specifiche dei requisiti in una struttura logica dei dati, la scelta è caduta sul modello relazionale;
3. progettazione fisica e implementazione: scelta della strumentazione, organizzazione del contenuto dell'interfaccia, realizzazione pratica (Atzeni e altri 2002, p. 185 e sgg).

Una volta progettata la basi di dati è stata scelta la strumentazione sia per la progettazione fisica del database, sia per la realizzazione dell'interfaccia. La scelta è caduta sull'utilizzo di un framework, Ruby on Rails e il linguaggio di programmazione sul quale RoR si basa, Ruby, per la connessione con il database. La restante parte di codifica è stata realizzata utilizzando Xhtml 1.0 con DTD Strict e Css 2.0, con particolare attenzione agli standard W3C.

Javascript, con le sue librerie Prototype e Script.aculo.us, è stato utilizzato per implementare le parti dinamiche del progetto.
Il setup dell'ambiente di sviluppo è stato fatto dapprima in locale e successivamente è stato spostato il progetto sui server del dipartimento, dove ora è disponibile all'indirizzo <http://di.unipi.it:3000>.

1.3. Finalità del progetto

La speranza è che il progetto possa essere proposto come strumento di supporto al ricercatore interessato ai «secoli IX-XII, specialmente per il Regno Italico e per i suoi ceti dominanti. (Violante 1988, p. 3)»
Egli può, una volta inserita la documentazione pertinente ai suoi studi, avere un database bibliografico a disposizione sul quale effettuare differenti tipi di interrogazioni e visualizzare in maniera dinamica collegamenti tra i personaggi storici inseriti, variando all'occorrenza il punto di vista dell'osservazione del dato.
Il primo esempio di interrogazione proposta è la correlazione fra le cariche amministrative assunte dai personaggi storici inseriti nel database e i territori di competenza. Si può quindi visualizzare in maniera immediata l'avvicendamento delle cariche in un dato territorio o la successione di cariche assunte da un dato personaggio, il tutto corredato dalla fonte da cui l'informazione è tratta.
La flessibilità del sistema permette di impostare ricerche diverse, come ad esempio quella genealogica, in parte delineata ma non ancora del tutto implementata.

1.4. Conclusioni e sviluppi futuri

Il progetto vorrebbe inserirsi nell'ambito degli studi sui grandi gruppi familiari (gli Obertenghi, i Bernardingi, gli Aldobrandeschi, i conti di Bologna, i conti Alberti ecc) protagonisti dell'ambito politico-economico dei sec. IX-XII. Si propone come aiuto nell'esaminare la

> continuità che si distende dalle famiglie marchionali comitali e vicecomitali carolinge, e da quelle della prima metà del secolo X, a quelle dei signori rurali (...): un'ininterrotta continuità di sangue che fu nel suo svolgimento storico molto

> dinamica per i profondi e numerosi cambiamenti e che non escluse apporti di stirpi nuove che assicurarono il ricambio sociale. (Violante 1988)

Interessanti le possibilità di ampliamento delle ricerche, dall'integrazione della già accennata indagine genealogica, alla definizione più dettagliata dei rapporti fra cariche comitali e marchionale e i vescovadi, «che esercitavano potenti interferenze rispetto agli ordinamenti di origine carolingia (Violante 1988, p. 5)»

Il progetto vuole quindi essere un punto di partenza, una proposta da ampliare e arricchire a seconda delle esigenze dei ricercatori che vorranno utilizzarlo, sperando che possa aiutarli nella trasposizione digitale dei contenuti delle loro ricerche.

Ringraziamenti:
a tutti quelli che hanno avuto molta pazienza, tutti. Chi ha atteso, chi ha sopportato, chi ha consolato, chi ormai non ci credeva più. Grazie di cuore, alla fine ce l'abbiamo fatta!
A chi mi ha aiutato quando non sapevo più a chi chiedere. Giorgio, Luca, senza di voi sarebbe stato più difficile.
A tutta Casa Tango, al cuoco greco eccezionale, alla matematica tanguera che mi ha aiutato a risolvere scherzi del codice. Trovarsi in casa con voi è stata una delle più belle sorprese di questo nuovo anno.
Alle mie svalvoline adorate, siamo tutte e tre innamorate ed è bellissimo!
Un gancho, un voleo, una sciabola e chi più ne ha più ne metta, al gruppetto felice o come qualcuno lo chiama al The Brand New Pisa's Tango y Cinema Group. Vi voglio bene, avete migliorato la mia vita!
Una carezza dietro l'orecchio al mio cagnolone, Popper.

Più di tutti, a chi al momento giusto ha saputo conquistarmi e affascinarmi, parlandomi di fiori in tasca e bolle di sapone! <3

2. Cenni storici e problematiche derivate

2.1. Il periodo storico

Il regno Longobardo di re Desiderio fu conquistato da Carlo Magno nel 774, grazie ad una sanguinosa campagna militare cominciata nel 773 e terminata con la conquista di Pavia.

Re Carlo procedette quindi alla sistematica immissione di ufficiali pubblici e di vescovi reclutati tra i propri vassalli, cercando comunque di non sconvolgere eccessivamente il preesistente ordinamento politico e amministrativo, non escludendo del tutto duchi e funzionari longobardi.

La guerra tra Franchi e Longobardi portò alla decadenza dei piccoli proprietari terrieri e all'affermazione delle grandi proprietà fondiarie. Infatti larga parte dell'esercito longobardo era costituita da piccoli proprietari terrieri che, contribuendo di tasca propria alle spese di guerra, vi esaurirono tutte le loro risorse e, costretti a vendere le loro terre ai grandi possidenti per poter sopravvivere, divennero contadini dipendenti. Cominciò così ad affermarsi la grande proprietà fondiaria, sia laica che ecclesiastica.

L'espansione dell'aristocrazia fu facilitata dalla discontinuità e dalla inconsistenza del potere centrale: lo stesso Carlo Magno poté trascorrere solo brevi periodi in Italia a causa dei suoi impegni imperiali, e ne affidò il comando al figlio Pipino prima e poi, alla morte di questi nel'810, al nipote Bernardo. Questi si ribellò allo zio Ludovico il Pio, nel frattempo asceso al trono imperiale dopo la morte di Carlo Magno nel 814, ma venne sconfitto e ucciso. La sua morte aprì una fase di vuoto di potere, durante la quale si affermarono le iniziative autonome di conti e vescovi, liberi di soggiogare e depredare di tutti i loro beni gli abitanti dei territori da loro controllati.

Solo nel'822 Lotario, figlio di Ludovico il Pio, fu incoronato re d'Italia, ma nonostante i suoi provvedimenti per arginare l'oppressione dell'aristocrazia, la situazione di instabilità non mutò; anzi si accrebbe quando otto anni dopo anche Lotario si oppose al padre, dando luogo ad una lunga serie di scontri che si conclusero solo nel'843 quando Lotario, dopo strenue lotte contro i fratelli per la successione al padre conclusasi con il trattato di Verdun, ottenne il titolo imperiale e con esso il Regnum Italiae, delegandone però l'amministrazione al figlio Ludovico II.

Ludovico II, che diventerà imperatore nel'855, tentò di frenare il processo di dissoluzione dell'ordinamento carolingio che però era ormai in atto in tutti i territori dell'impero e che

giunse a compimento nel′888 con la deposizione dell′ultimo imperatore carolingio, Carlo il Grosso.

Dopo la morte di Carlo il Regno d′Italia restò senza eredi diretti e l′inesistenza di una casata nettamente superiore alle altre portò ad una lunga serie di guerre e devastazioni causate dalla competizione per il trono dei titolari dei grandi ducati di Spoleto, Friuli e Toscana, raramente riuscendo a governare per lunghi periodi: tra i più importanti, ci sono il regno di Berengario di Friuli, di Ugo di Provenza e di Berengario d′Ivrea.

A ciò vanno aggiunte le sanguinose incursioni degli Ungari che a partire dal 899, cominciarono a flagellare l′Italia settentrionale.

Questo periodo di guerra continua portò alla trasformazione dei centri abitati, che diventarono sempre più fortificati e isolati, costretti a contare solo su loro stessi per sopravvivere; si andarono così a costituire comunità autonome, separate del resto del Regno e governate quasi totalmente dal vescovo o nobile di turno, rendendo il Regnum Italiae uno stato solo di nome, diviso in numerosi frammenti non comunicanti tra loro.

Nel 967 sale al trono Ottone II del Sacro Romano Impero e a cui successe Ottone III, ponendo momentaneamente fine ai conflitti dei casati Italiani. Durante il loro governo, gli Ottoni cercarono di far riaffermare l′autorità del potere centrale, unitamente al tentativo di moralizzazione dei costumi della Chiesa. Un obbiettivo difficilissimo da raggiungere nella società in disfacimento che era quella italiana, ma che gli Ottoni portarono avanti con a risultati apprezzabili, fino alla morte prematura di Ottone III nel 1002.

Ultimi re d′Italia furono Arduino d′Ivrea ed Enrico II del Sacro Romano Impero, che si contesero la corona fino alla morte di Arduino nel 1015. Arduino fu l′ultimo re proveniente da una casata italiana, dopo di che il potere fu sempre esercitato da imperatori stranieri.

Anche grazie alla lontananza del potere centrale, in numerose zone dell′Italia centro-settentrionale le comunità urbane cominciarono a evolversi in quelli che sarebbero poi diventati i Comuni.

2.2.Le analisi possibili

Come abbiamo visto nei cenni storici (v. 2.1), il progressivo passaggio tra l'impianto degli ordinamenti pubblici carolingi e lo sviluppo delle istituzioni signorili arricchì il ceto

dominante di nuove componenti: all'aristocrazia franca si affiancarono famiglie eminenti, locali, di sangue longobardo o anche romano; alla sistematica articolazione circoscrizionale, politica e amministrativa, facente capo al re, si intrecciarono strutture e istituzioni che esprimevano nuovi modelli di radicamento famigliare e di esercizio del potere.
Il sec. IX fu quello in cui maggiormente il ceto dominante si affermò, rendendo concreta la sua egemonia, esercitando uffici pubblici (marchionali e/o comitali), che ebbero un ruolo nel promuovere certi tipi di strutture familiari aristocratiche.

La ricerca in questo ambito si sviluppa su diverse linee:

- quella genealogica, arricchendo e aggiornando vecchie genealogie;
- sull'individuazione delle strutture familiari e delle loro dinamiche, politiche, professionali, patrimoniali;
- sul comportamento delle grandi famiglie nei confronti delle circoscrizioni pubbliche in cui operavano;
- sulla politica familiare attuata, i programmi per l'avvenire e i progetti dinastici.

Dalle marche il discorso è estensibile a tutti i distretti comitali e a tutte quelle zone del Regno Italico più incerte e mobili nei confini. Qui l'analisi di territori, famiglie e personaggi deve essere integrata con l'analisi negli stessi termini dei vescovadi, che esercitavano potenti interferenze rispetto agli ordinamenti di origine carolingia.

Per proporre una ricerca il più possibile puntuale è utile analizzare la denominazione di ciascuna "marca" o "comitatus", dell'uso dei titoli di "marchio" e di "comes", osservando il contesto lessicale e strutturale in cui i singoli termini si trovano inseriti nella documentazione archivistica e nelle fonti narrative.
Interessanti sono le espressioni come "terra comitis" o "fiscus comitis", "res comitis", ecc o riferimenti ai "beneficia" del "comes" o del "marchio" e soprattutto le qualifiche di "vassi" o "fideles" o "milite maiores" attribuite a "comites" o a "marchiones".

L'indagine si sofferma anche sulla struttura interna della "marca" e del "comitatus": marchesi che sono titolari di una intera "marca", o che lasciano a conti, o anche a visconti, il governo di

tutte o solo di alcune contee che la compongono. È importante anche rilevare i "comites" che pasano da un "comitatus" a un altro o che riuniscono nelle loro mani più contee.
A questo proposito è preziosa la segnalazione dei distretti pubblici dove avvengono mutamenti di famiglie marchionali o comitali e dove si effettua la compresenza di più dinastie di conti o di marchesi.

Ritornando alle strutture familiari, risulta importante l'individuazione del capostipite di ciascuna famiglia, specificando in base a quali elementi lo si stabilisca e appurare, ove possibile quale memoria del passato abbia il gruppo familiare e a quante generazioni risalga.
Importante è anche l'indagine sull'inizio dell'ereditarietà del titolo nella stessa famiglia e come l'appellativo si sia trasmesso agli eredi.

In ultimo vedere se il titolo comitale sia legato ad una circoscrizione di "comitatus" con al centro una città o se il titolo sia connesso con una signoria territoriale e con un castello.

Indagini ulteriori interessanti riguardano i coniugi degli aventi titolo e come questi sono definiti e il vasto problema dei visconti e del loro rapporto con il titolare della "marca" o del "comitatus".

2.3. Le problematiche relative

Il periodo storico preso in considerazione porta con se numeroso problematiche all'atto pratico della modellizzazione dei dati.

La prima riguarda l'incertezza dei dati con i quali ci troviamo a trattare.

2.3.1. Incertezze anagrafiche

Nel realizzare un database anagrafico di soggetti contemporanei, avere un nome, un cognome, una data di nascita e una di morte, insieme a molti altri dati attinenti, come il genere o il codice fiscale, non è facilissimo ma è possibile. Si possono consultare documenti pubblici (come gli atti dello stato civile, gli atti parrocchiali o gli atti notarili) o privati (come cronache, lettere ecc) in cui reperire dati genealogici verificabili. Più ci si allontana nel tempo, più i dati saranno incerti.

Se i soggetti sono di epoca medievale, bisogna innanzitutto decidere quali sono i dati anagrafici pertinenti e quali di questi sono reperibili o opzionali.

La prima difficoltà sta nel reperire fonti anagrafiche antecedenti al 1564, anno di istituzione del registro parrocchiale da parte della Chiesa Cattolica. Ci si rivolge quindi principalmente a fonti documentarie che avevano tuttavia lo scopo di riportare una determinata attività in un preciso momento e luogo e non lo stato civile del personaggio citato. Da questi documenti si possono estrapolare dei dati cronologici che definiscono l'esistenza in vita della persona in questione. La prima attestazione e l'ultima attestazione rappresentano l'arco temporale in cui il personaggio storico si muove, meglio di un anno di nascita e un anno di morte. Spesso si conosce l'anno di morte sicura, ossia si può sapere che è morto sicuramente dopo un determinato anno (l'espressione che si usa è *quondam*).

Una volta individuato l'arco di esistenza di una persona, le incertezze riguardano il suo nome. Era infatti invalsa la tradizione di chiamare il figlio con il nome di un avo precedente, per delineare la linea familiare. Per distinguere un Bonifacio da un altro occorrono quanti più dati collaterali possibili. A volte, a seconda delle cariche acquisite in vita, si può trovare una numerazione romana accanto al nome (Adalberto I Marchese di Toscana), che facilita la collocazione temporale dell'individuo e permette di 'numerare' i discendenti se ricoprono la carica assunta dal padre.

L'altra incertezza riguarda la forma in cui il nome si trova, nelle fonti si può reperire *Hugo comes* tanto quanto *Hugonis comitiis* o *Hugus qui fuit comes* o *Hugo Alberici Marchio*.

La scelta è caduta sull'inserimento del nome latino al nomitativo per la persona trattata ed è stato impostato un altro campo per le altre variazioni del nome, sempre al nomitativo.

Ulteriori campi sono stati aggiunti per l'inserimento della famiglia o del cognome familiare (Aldobrandeschi, Bonifaci ecc) e del patronimico.

Riassumendo una persona può avere:

- Nome, in forma latina, al nominativo
- Variazioni del nome, in forma latina, al nominativo
- Patronimico
- Nome della famiglia
- Sesso
- Una data di nascita, se conosciuta
- Una data di morte, se conosciuta
- Una prima attestazione, da reperire nell'elenco delle attestazioni registrate
- Un'ultima attestazione,
- Un anno di sicura morte, quondam
- Note aggiuntive

2.3.2.La bibliografia

Nell'interfaccia che stiamo delineando, una volta visualizzati i dettagli relativi alla persona, è necessario corredare le informazioni inserite con le fonti da cui queste informazioni sono state prese e con gli studi effettuati.

Lo schema sarà quindi il seguente:

- Dettagli anagrafici
- Attestazioni in forma tabellare cronologica, con citazione della fonte da cui è stato estrapolato il dato e bibliografia della fonte citata.
- Elenco degli studi relativi alla persona

Se, come nel nostro caso, è stata realizzata un ulteriore interpretazione dei dati (tra gli scopi primari del progetto) può essere anch'essa posizionata nella nostra visualizzazione.

Ecco quindi comparire l'elenco delle cariche amministrative assunte dalla personalità.

- Dettagli anagrafici;

- Cariche amministrative ricoperte e territorio di competenza, in ordine cronologico, con riferimento alla fonte da cui è stato estrapolato il dato;
- Attestazioni in forma tabellare cronologica, con citazione della fonte da cui è stato estrapolato il dato e bibliografia della fonte citata;
- Elenco degli studi relativi alla persona.

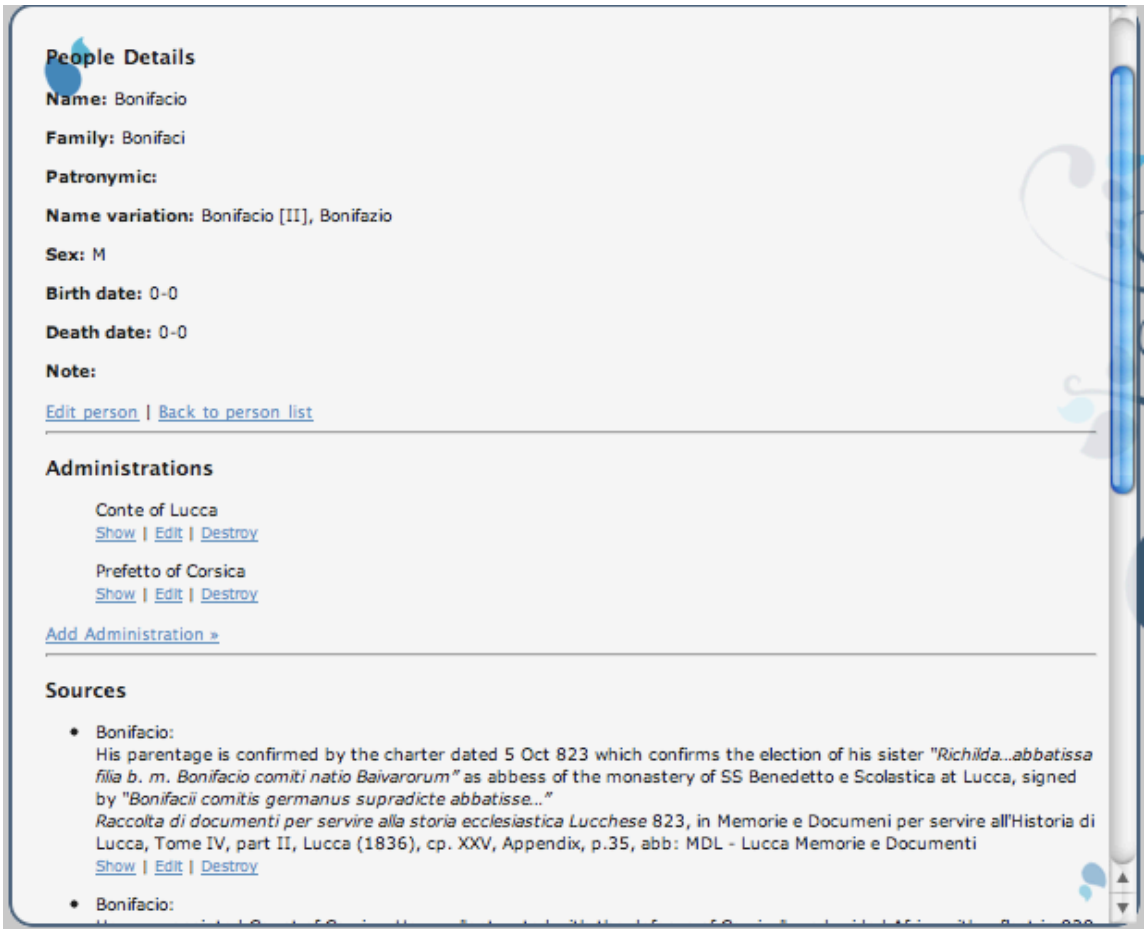


Figura 1. Schermata dei dettagli anagrafici della persona e delle fonti che lo citano.

La bibliografia viene suddivisa in fonti e studi, come è stato detto. Le tipologie bibliografiche per il periodo trattato sono sei: la monografia, il saggio in miscellanea, la tesi e l'articolo di rivista, che rientrano negli studi, il manoscritto e la raccolta di fonti primarie o edizione critica che rientrano nelle fonti.

Il problema della catalogazione bibliografica è ampio e sterminato. Lo scopo che volevamo raggiungere era quello di dare una forma normale alle citazioni bibliografiche e il conseguente lavoro di normalizzazione dei campi bibliografici si è rivolto verso questa direzione.

Una volta ridotte a sei le tipologie, sono stati individuati degli attributi comuni e successivamente gli attributi specifici per ogni entità particolare.

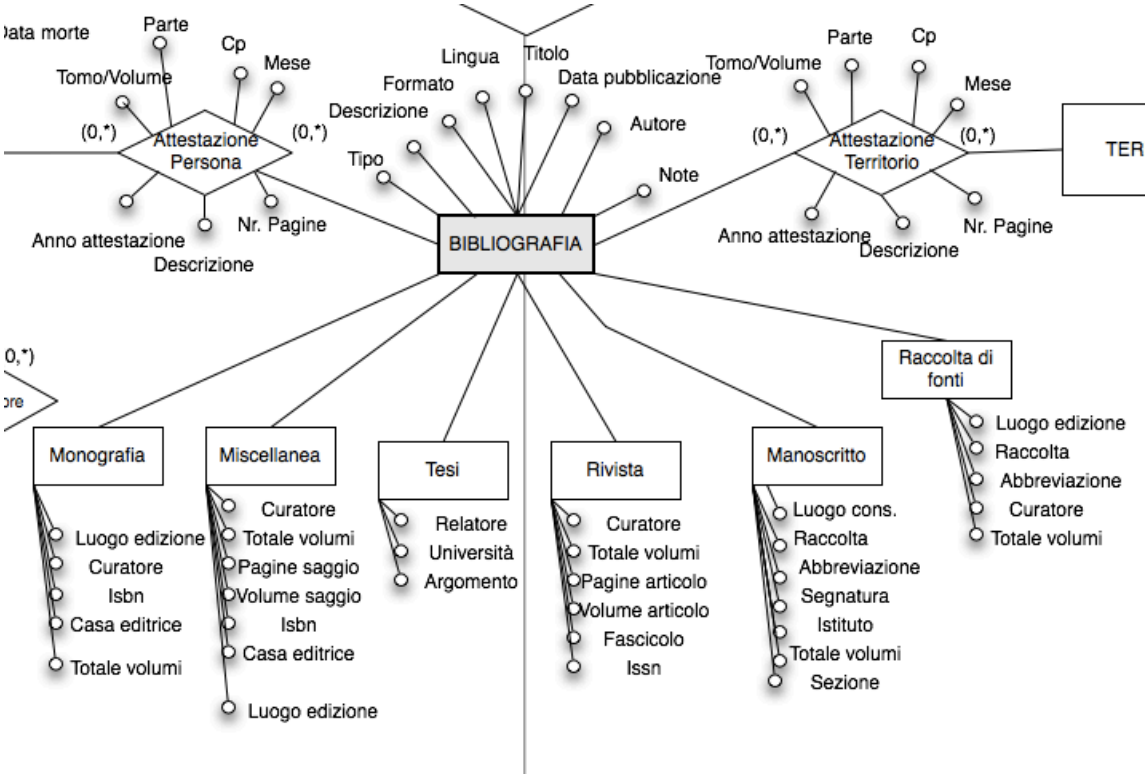


Figura 2 . Particolare dello schema concettuale relativo al problema bibliografico.

Per tutte le categorie sono stati individuati:

Tipo (Fonte o Studio)

Autore

Titolo

Lingua

Formato

Descrizione

Data di pubblicazione

Immagine

URL Uno o più indirizzi web dove è possibile trovare immagini o riferimenti al testo citato

Note aggiuntive

Per la monografia sono stati aggiunti:

Luogo di edizione

Curatore

Codice Isbn

Casa editrice

Volumi totali, nel caso di un'opera in più volumi

Per il saggio in miscellanea:

Luogo di edizione

Curatore

Codice Isbn

Casa editrice

Volumi totali, nel caso di un'opera in più volumi

Intervallo delle pagine del saggio

Indicazione del volume in cui è inserito il saggio, nel caso di opera in più volumi

Per la tesi di laurea:

Relatore

Argomento di laurea

Nome dell'università

Anno accademico

Per l'articolo in rivista:

Curatore della rivista

Codice Issn

Volumi totali, nel caso di un'opera in più volumi
Intervallo delle pagine dell'articolo
Fascicolo, in cui è conservato l'articolo
Indicazione del volume in cui è inserito l'articolo, nel caso di una rivista raccolta in più volumi

Per il manoscritto:
Istituto, dove è conservato
Sezione
Segnatura
Raccolta, eventuale nome della raccolta in cui è inserito
Abbreviazione, nome breve con cui è conosciuto il manoscritto
Luogo, di conservazione, città dell'istituto
Volume totali, eventuali numero di volumi in cui è raccolto

Per la raccolta di fonti o edizione critica:
Luogo di edizione
Curatore
Codice Isbn
Casa editrice
Raccolta, eventuale nome della raccolta in cui è inserito
Abbreviazione, nome breve con cui è conosciuta la raccolta
Volumi totali, nel caso di un'opera in più volumi

In questo modo si è cercato di coprire tutte le eventualità di citazione da un testo.

Sono state quindi differenziati i moduli di immissione dei dati, uno per ogni tipologia di risorsa, in modo da non mandare in confusione l'utente che si trova ad effettuare il lavoro di inserimento. Per ogni fonte o studio introdotto l'elenco dei dettagli visualizza solo i dati pertinenti. Tutto questo è stato ottenuto tramite gli *helpers* (v. 4.5) , un'altro utile strumento messo a disposizione dall'ambiente di sviluppo che abbiamo usato.

3. Progettazione concettuale

La progettazione concettuale di una base di dati consiste nella costruzione di uno schema Entità-Relazione in grado di descrivere al meglio le specifiche sui dati di una applicazione.

3.1. Analisi dei requisiti

Una volta individuato il periodo storico di riferimento e le sue caratteristiche, si passa all'analisi dei requisiti, ossia «la completa individuazione dei problemi che l'applicazione da realizzare deve risolvere e le caratteristiche che tale applicazione dovrà avere (Atzeni 2002, p. 226)».
Nel nostro caso:

Si vuole realizzare una base di dati anagrafica di personalità appartenenti ai ceti dominanti nel Regno Italico, vissuti fra il 774 d.C e il 1115 d.C e dei territori da loro amministrati, nonché delle attestazioni relativi ad entrambi. Vogliamo rappresentare quindi i dati delle persone, dei territori e della bibliografia che ne tratta. Per le persone (circa 1000), identificate da un codice, rappresentiamo il nome latino al nominativo, il nome della famiglia di appartenenza, il patronimico, le variazioni latine del nome, altre variazioni conosciute del nome, il sesso, la data di nascita, la data di morte, il quondam e note aggiuntive. Per i territori rappresentiamo il codice identificativo, la denominazione, la variazione del nome, l'area e una descrizione.

Per la bibliografia rappresentiamo il tipo (monografia, saggio in miscellanea, articolo di rivista, manoscritto, tesi, la raccolta di fonti), la tipologia (se fonte o studio), una descrizione, il formato, la lingua, il titolo, l'autore, la data di pubblicazione, l'editore, una immagine, uno o più link di riferimento e note aggiuntive. Per le attestazioni/citazioni rappresentiamo il codice identificativo, l'anno di attestazione, la descrizione, le pagine che contengono la citazione. Per le relazioni coniugali rappresentiamo il marito, la moglie e l'intervallo di matrimonio. Per le relazioni tra genitore e figlio rappresentiamo il genitore e il figlio. Per le cariche amministrative, rappresentiamo la persona che ricopre la carica, il territorio di competenza, l'inizio e la fine dell'amministrazione del territorio e il tipo di carica ricoperto.

Tabella 1. Analisi dei requisiti espressi in linguaggio naturale

È utile, ai fini di migliorare la precisione dei termini usati, definire un glossario dei termini.

Termine	Descrizione	Sinonimi	Collegamenti
Persona	Amministratore di uno o più territori o persona generica con un legame di parentela con un amministratore	Conte, Marchese, Vescovo	Persona autoreferenziale, Attestazione, Fonte bibliografica, Territorio.

Termine	Descrizione	Sinonimi	Collegamenti
Territorio	Territorio amministrato	Contea, Marchesato, Episcopato	Persona, Fonte, Attestazione, Amministrazione.
Bibliografia	Fonti e studi relativi alla persona, possono essere monografie, miscellanee, riviste, tesi, manoscritti, raccolte di fonti.	Monografia, Miscellanea, Rivista, Tesi, Manoscritto, Raccolta di fonti.	Persone, Attestazioni, Territori.
Attestazione	Riferimenti cronologici alle persone e ai territori inseriti nel database, citazioni bibliografiche.	Rif. cronologici Testimonianze	Persone, Fonti, Territori.
Amministrazione	Indicazioni temporali e tipologia di carica assunta da una persona in un territorio	Carica amministrativa	Persona, Territorio.

Tabella 2. Glossario dei termini

Il passo successivo consiste nell'individuare le specifiche sulle operazioni da effettuare sui dati. Segue un elenco delle principali operazioni individuate:

Operazione 1: inserire una nuova persona indicando tutti i suoi dati

Operazione 2: assegnare il grado di parentela di una persona con un'altra

Operazione 3: inserire una nuova fonte bibliografica

Operazione 4: inserire una nuova fonte bibliografica o un nuovo studio

Operazione 5: inserire un nuovo territorio

Operazione 6: inserire una nuova citazione o attestazione, collegando una persona ad una fonte o ad uno studio

Operazione 7: assegnare ad un territorio un amministratore, la durata della carica e il titolo assunto.

Operazione 8: stampare l'elenco delle persone in ordine alfabetico

Operazione 10: stampare di una persona tutte le informazioni che lo riguardano, territori di competenza e cariche assunte, fonti che lo citano, studi che lo riguardano.

Operazione 11: stampare tutti gli amministratori di un determinato territorio

3.2. Modellizzazione dei dati

Abbiamo quindi individuato i concetti rilevanti della struttura dei dati. Il passo successivo è la traduzione in un modello Entità-Relazione attraverso una serie di trasformazioni e di raffinamenti dello schema stesso, per arrivare in ultima istanza allo schema finale.

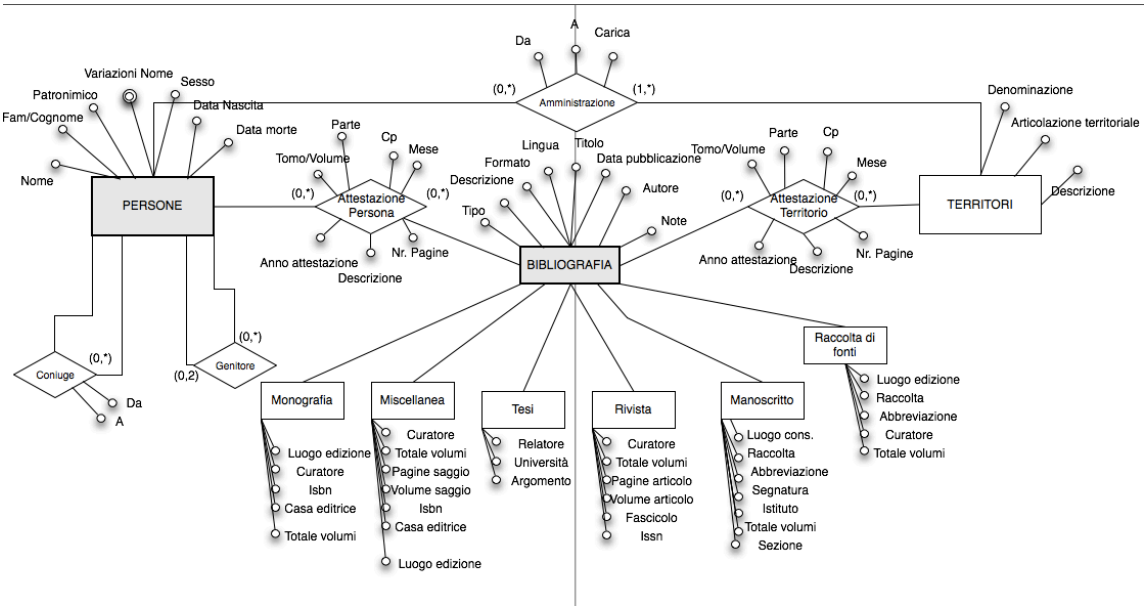


Figura 3. Schema E-R della base di dati

Come si può evincere dalla figura 3 le entità principali sono tre, Persone, Bibliografia e Territori. Bibliografia a sua volta è una generalizzazione di altre sei entità distinte, nello specifico, Monografia, Miscellanea, Tesi, Rivista, Manoscritto, Raccolta di fonti. La distinzione fra fonti e studi è stata ridotta ad un attributo 'tipo' dell'entità Bibliografia. A esse sono state

aggiunte le proprietà (attributi) relativi. Le entità Attestazione e Amministrazione, sono due relazioni fra entità. La prima è scissa in due relazioni separate, Attestazioni_Persona e Attestazioni_Territorio. Entrambe associano ad una persona la fonte o lo studio relativo, aggiungendo attributi alla relazione. La relazione Amministrazione associa invece una persona ad un territorio di competenza, aggiungendo proprietà pertinenti al tipo di relazione, come il tipo di carica e le date di inizio e fine dell'incarico.
Come si può osservare le relazioni sono di tipo Molti a Molti fra Persona e Bibliografia e fra Territorio e Bibliografia. Allo stesso modo la relazione fra Persone e Territori.

3.3.Traduzione del modello concettuale in modello logico

Una volta completata la modellizzazione concettuale, si passa alla modellizzazione logica, che non è una «semplice traduzione da un modello ad un altro [poiché] lo schema E-R va ristrutturato per soddisfare due esigenze: quella di "semplificare" la traduzione e quella di "ottimizzare" il progetto. (Atzeni 2002, p. 255)».

La ristrutturazione di uno schema E-R, passa attraverso una serie di passaggi. Nel nostro caso uno è quello risultato importante: l'eliminazione delle generalizzazioni.
Delle tre alternative possibili - accorpamento delle figlie della generalizzazione nel padre, accorpamento del padre nelle figlie, sostituzione della generalizzazione con associazioni - è stata preferita la prima. Le proprietà delle entità che subiscono l'accorpamento (Monografia, Miscellanea, Tesi, Rivista, Manoscritto, Raccolta di fonti) vengono riversate nell'entità padre (Bibliografia). A tale entità viene aggiunto un ulteriore attributo che serve a distinguere la "tipologia" di un'occorrenza, cioè a quale delle entità figlie apparteneva la tale occorrenza. Si è scelto di lasciare poi la distinzione fra tipologie di Fonti bibliografiche e Studi all'interfaccia di implementazione (v. 4.3).

4. Progettazione fisica e implementazione

4.1. Modellizzazione dei task e scenari d'uso

Una volta individuate la struttura dei dati e le operazioni da effettuare, si possono ipotizzare degli scenari d'uso, utili per creare un'interfaccia il più possibile aderente alle esigenze degli utenti finali.

Come abbiamo detto il nostro target d'utenza è rappresentato dal ricercatore di Storia Medievale, quindi l'interfaccia che andremo a realizzare deve soddisfare delle esigenza piuttosto alte, soprattutto dal punto di vista storico e storiografico.

Sono stati valutati i seguenti scenari:

4. Leonardo, ricercatore. Leonardo, sta conducendo delle ricerche sulla famiglia degli Aldobrandeschi, ha reperito una serie di documenti, vuole mettere in correlazione la figura di Ildebrandus con i territori da lui amministrati. Per lui è fondamentale partire dal dato storico, dalla fonte e successivamente effettuare un'interpretazione dei dati. Decide quindi di utilizzare l'applicazione, registra le proprie generalità sul sito, accede alla pagina relativa alle personalità storiche, inserisce i dati anagrafici di Ildebrandus, dal nome latino, alle variazioni del nome, il sesso, la data sicura di morte. Prima di aggiungere le cariche amministrative ricoperte, inserisce nella parte relativa agli studi, qualcuno dei testi di studio sulla famiglia in questione, dopo aver controllato che non ci siano già. Decide successivamente di completare la scheda informativa sul personaggio in questione, allegando alcune delle citazioni che ha reperito durante i suoi studi. Controlla che le fonti da cui inserirà le citazioni non siano già presenti e le introduce. Nel caso di quelle invece già inserite evidentemente da un altro ricercatore, semplicemente immette il testo della citazione e la pagina in cui essa si trova. In questo modo i dati inseriti vanno a posizionarsi in una struttura tabellare in cui sono visualizzate le attestazioni registrate in ordine cronologico. Ogni attestazione ha il suo riferimento bibliografico. Una volta completata l'immissione delle citazioni, Lorenzo si decide ad inserire le cariche amministrative di Ildebrandus. Introduce i dati relativi al tipo di carica, al periodo e il riferimento alla fonte che attesta Ildebrandus comes del territorio interessato. Può quindi scorrere la pagina dei dettagli della persona e vedere tutti i dati appena aggiornati, in alternativa può andare su

una pagina di uno dei territori inseriti, per esempio Lucca e vedere così l'avvicendamento delle cariche territoriali sul territorio. Esce dal sito, facendo il log out.

Task: Implementazione di una pagina di inserimento dei dati per le fonti bibliografiche, una pagina di inserimento degli studi, una pagina di inserimento dei dati anagrafici di personalità storiche nonché una pagina di inserimento dei territori. Al fine una parte deputata all'inserimento delle associazioni fra le entità appena presentate e di visualizzazione delle stesse.

5. Lorenzo, ricercatore. Lorenzo, sta completando i suoi studi sulla famiglia dei Bonifaci. Ha già inserito nel sito tutti i dati relativi ai membri della famiglia, vuole controllare l'avvicendarsi delle cariche nel territorio di Lucca e vedere l'evoluzione delle cariche amministrative che il territorio ha subito negli anni. Accede quindi alle pagine del territorio, e visiona l'indice dei territori presenti nel database. Ricerca il territorio di Lucca, ne vede i dettagli e vede i nominativi di chi ha ricoperto le cariche. Stampa i dati relativi al territorio, dunque esce dal sito.

Task: Implementazione di una pagina di ricerca. Possibilità di stampare i dati raccolti.

4.2. Organizzazione del contenuto

Una volta definiti i task si può passare ad una prima organizzazione del contenuto del sito.

4.2.1. Contenuto generale

L'interfaccia dovrà quindi avere una pagina di presentazione del progetto, una serie di pagine di primo livello in cui sarà possibile visualizzare una lista dei dati presenti, che siano Persone, Territori, Fonti e Studi e sui quali si potranno fare delle ricerche. Nella pagina di primo livello delle entità principali, Persone, Territori, Fonti e Studi sarà possibile inserire nuove istanze delle medesime entità, se si posseggono i privilegi di modifica dei dati. Sarà quindi necessaria una pagina di log in e log out dal sito e una di registrazione dell'utente.

Al secondo livello ciascuna di queste pagine mostrerà il dettaglio del record selezionato. Se si hanno i privilegi di "sola lettura" sarà possibile scorrere i dati e visionarli, nel caso di un utente con possibilità di log in, egli avrà accesso a un terzo livello di modifica o inserimento di altri dati.

Nello specifico al terzo livello delle pagine Persone e Territori sarà prevista una pagina con un form di associazione tra entità, ad esempio di creazione delle associazioni tra Persone e Fonti o tra Territori e Fonti. Nella pagina Persone è previsto un ulteriore allargamento all'inserimento dei dati relativi alle cariche amministrative e agli studi relativi alla persona selezionata.

4.3.Implementazione

4.3.1.Linguaggio di codifica e strumentazione

Il lavoro è stato implementato usando come linguaggio XHTML 1.0 con DTD Strict. La parte grafica è stata realizzata con un foglio di stile (CSS 2.0) esterno.

Per la sezione più impegnativa relativa alla connessione al database, alla creazione di pagine dinamiche successive all'interazione con l'utente, nonché del database stesso l'ambiente di sviluppo è stato utilizzato il framework Ruby on Rails, basato sul linguaggio di scripting Ruby. La versione di Rails utilizzata è la 2.3.5. La versione di Ruby la 1.8.7.

Ruby è un linguaggio di scripting orientato agli oggetti. «Ruby è coinciso, senza essere inutilmente stringato: (...) consente di esprimere le idee in modo naturale e chiaro. (Thomas, 2009)». È stato creato da un giapponese, Yukihiro Matsumoto, con l'intento di sintetizzare ciò che di meglio si potesse trovare negli altri linguaggi di programmazione. È inoltre totalmente open source.

I punti di forza del linguaggio:

1. ha una sintassi chiara e leggibile, elegante, ispirata parzialmente da linguaggi come Eiffel e Ada;
2. è linguaggio orientato agli oggetti, come Smalltalk o Java, a differenza di linguaggi come PHP o Perl dove il supporto Object Oriented è stato aggiunto in un secondo momento, con tutti gli svantaggi derivati.
3. la gestione delle eccezioni è agile e potete, derivata da Java e Python;
4. sono disponibili blocchi e closure, sulla falsariga delle funziona lambda in Lisp;

5. prevede classi, metodi e funzioni per la gestione del testo e delle espressioni regolari, migliorando Perl in questo campo, e dispone di una classe (Regexp) interamente dedicata alle espressioni regolari.

È inoltre un linguaggio interpretato, come Perl o Python, che comporta grande versatilità, assenza di compilazione e portabilità del codice.
Dapprima ostacolato nella sua diffusione a causa della documentazione prevalentemente scritta in giapponese, ha avuto una nuova distribuzione a seguito del rilascio della versione 1.6, corredata da documentazione in lingua inglese. Fu "adottato" e ripreso da due progammatori statunitensi (Andy Hunt e Dave Thomas) che godevano di ottima considerazione nell'ambiente informatico e che con il loro interesse ne decretarono la conseguente popolarità.
Ad ampliare il successo di questo linguaggio arrivò a cavallo tra il 2004 e il 2005, un framework per lo sviluppo di applicazioni web, chiamato appunto Ruby on Rails.

Ruby on Rails, spesso chiamato RoR o semplicemente Rails, è un framework open source per applicazioni web la cui architettura è fortemente ispirata al paradigma Model-View-Controller (MVC).

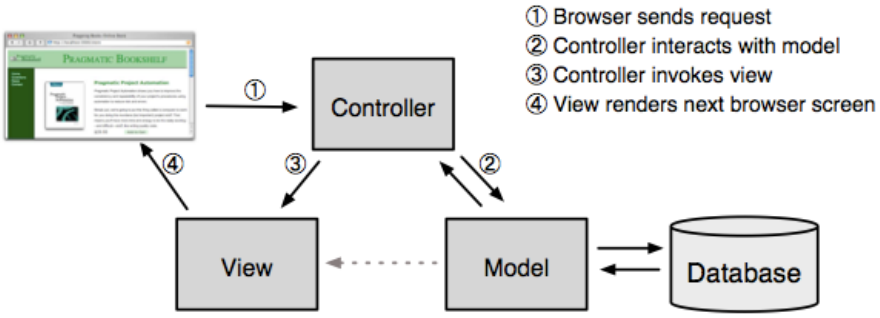


Figura 4. Schema dell'architettura MVC (Ruby 2009)

Questo tipo di architettura prevede la separazione tra

1. il model, che fornisce i metodi per accedere ai dati utili all'applicazione;

2. il view, che visualizza i dati contenuti nel model e si occupa dell'interazione con utenti e agenti;
3. il controller, che riceve i comandi dell'utente (in genere attraverso il views) e li attua modificando lo stato degli altri due componenti.

Il modello si occupa di mantenere lo stato dell'applicazione, non rappresenta soltanto i dati ma stabilisce le regole applicative cui devono attenersi i dati stessi.
La vista si occupa di generare l'interfaccia utente, basata generalmente sui dati del modello. Accede all'elenco dei dati e li formatta per per l'utente.
I controller coordinano l'applicazione, ricevono gli inpute dell'utente, interagiscono con il modello e forniscono all'utente la vista appropriata.

Le applicazioni risultato quindi strutturate in modo più lineare, ogni porzione di codice ha la sua collocazione e si parte con uno scheletro dell'applicazione già pronto.
Due concetti chiave hanno governato la progettazione di Rails, DRY (*Don't Repeat Yourself*) e *convention-over-configuration*. DRY significa che ogni elemento di informazione nell'ambio di un sistema dovrebbe essere espresso in un solo punto e, per adattarsi a questo principio, Rails utilizza la potenza di Ruby. Difficilmente si troveranno duplicazioni di codice in un'applicazione.
Convention-over-configurarion indica che Rails offre soluzioni predefinite per quasi tutti gli aspetti che compongono un'applicazione, in questo modo sarà possibile realizzare un'applicazione web com molto meno codice rispetto a quello necessario con un altro linguaggio, ad esempio Java.
Rails integra numerosi altri servizi e ne rende facile l'utilizzo, attraverso i set di librerie messe a disposizione di default (integrazione con AJAX e con le interfacce RESTful, la gestione della posta, l'ambiente completo per effettuare i test, ambienti separati per lo sviluppo, i test e la produzione, ecc) e ai numerosi plug-in che aggiungono funzionalità al progetto [1].

[1] http://agilewebdevelopment.com/

È stato quindi deciso di utilizzare RoR in quanto ben si adattava alle esigenze del progetto, grazie alla flessibilità del sistema, alle numerose librerie di codice che permettono di ampliarne le funzionalità e alle allettanti prospettive di sviluppo futuro della piattaforma.

Rails è stato estratto dal suo autore, David Heinemeier Hansson, dal codice di Basecamp, un'applicazione Web per il project management di 37signals, e rilasciato come progetto open source nel 2004; intorno al progetto si è sviluppata nel tempo una community molto attiva, tanto che la versione 2.1, rilasciata a giugno del 2008, ha visto la partecipazione di 1400 collaboratori che hanno contribuito con oltre 1600 patch. L'interesse verso il framework si è espresso anche con la pubblicazione di numerosi libri, blog, e l'organizzazione di conferenze in tutto il mondo[2].

4.4.Funzionalità

Dopo aver rapidamente installato e aggiornato RubyGem, il sistema di installazione e aggiornamento delle librerie Ruby tramite una serie di comandi da Terminale[3],

```
sudo gem update --system
sudo gem install rails -y
sudo gem update rake
sudo gem update sqlite3-ruby
```

possiamo creare la nostra prima applicazione, nel nostro caso "comitatus".

```
rails comitatus
```

---

[2] Alcuni link da cui partire:
http://api.rubyonrails.org/, dove si può trovare tutta la documentazione ufficiale
http://wiki.rubyonrails.org/, dove si trovano gli articoli scritti dalla comunità RoR
http://railscasts.com/ e http://asciicasts.com/, dove si possono trovare degli ottimi video e tutorial
http://www.ruby-forum.com/, uno dei forum più completi

[3] Come sistema operativo è stato usato Mac Os X. Si rimanda alle guide specifiche per l'installazione in altri sistemi operativi come Linux o Windows.

I file generati dopo la digitazione di questo comando, costituiscono lo scheletro dell'applicazione.

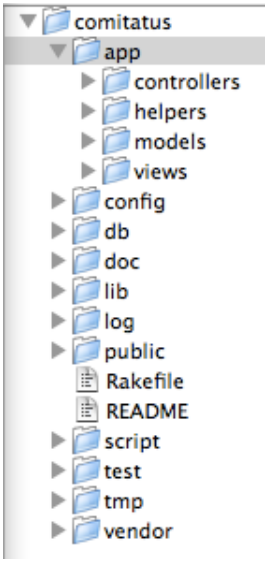


Figura 5. Struttura dell'applicazione

Come possiamo notare, all'interno della cartella `app` compaiono quelli che saranno poi i protagonisti della nostra applicazione, le sottocartelle `controllers`, `models` e `views`.

Come abbiamo precedentemente detto l'architettura di Ruby on Rails implementa il pattern MVC :

- i modelli si occupano di rappresentare i dati che l'applicazione vuole trattare, e di interagire con il sistema persistenza utilizzato per salvare i dati dell'applicazione;
- le viste si occupano di interagire con gli utenti, raccogliendo gli input e mostrando i risultati delle elaborazioni (le pagine HTML, in definitiva)
- i controller contengono la logica applicativa, ovvero il comportamento che l'applicazione adotta a seconda degli stimoli esterni (view) e dei dati (model).

Vediamo come tutto questo funziona nel progetto che abbiamo realizzato.

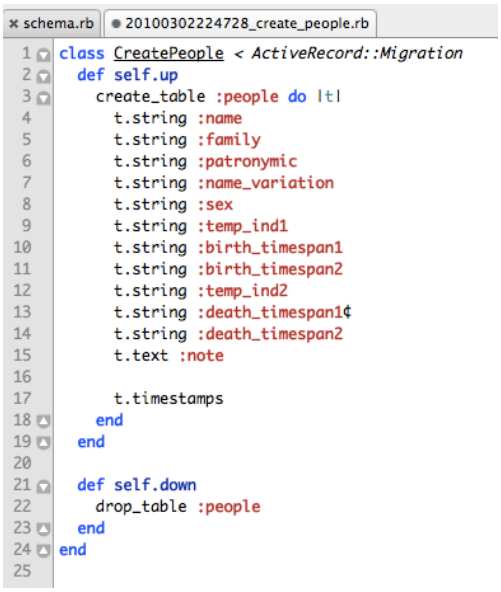
Figura 6. Creazione della tabella Persone

4.4.1. Modelli

I modelli sono in pratica quelle che poi saranno le "tabelle" del database nelle quali inseriremo i dati. Del database relazionale viene data una visione a oggetti, sia nella definizione di una nuova tabella che corrisponde alla creazaione di una nuova classe, sia nella visualizzazione, sia nella creazione di una nuova ennupla che corrisponde alla creazione di un'istanza.

Per crearle andiamo ad utilizzare l'istruzione `generate scaffold`.

In questo caso partiamo dal modello Persone e creiamo la risorsa indicando gli attributi che ogni persona deve avere e il relativo tipo, che abbiamo individuato nella progettazione logica:




```
ruby script/generate scaffold person name:string family:string patronymic:string
```

Il risultato, come si può vedere è l'elenco degli attibuti relativi alla tabella creata, nei quali si può ancora intervenire, aggiungendo limitazioni o definendo le chiavi esterne.

Successivamente utilizziamo il comando `rake db:migrate` per preparare il database e la tabella necessaria alla gestione della risorsa. Questo andrà a creare fisicamente la tabella all'interno del database e la inserirà insieme alle altre già precedentemente create.

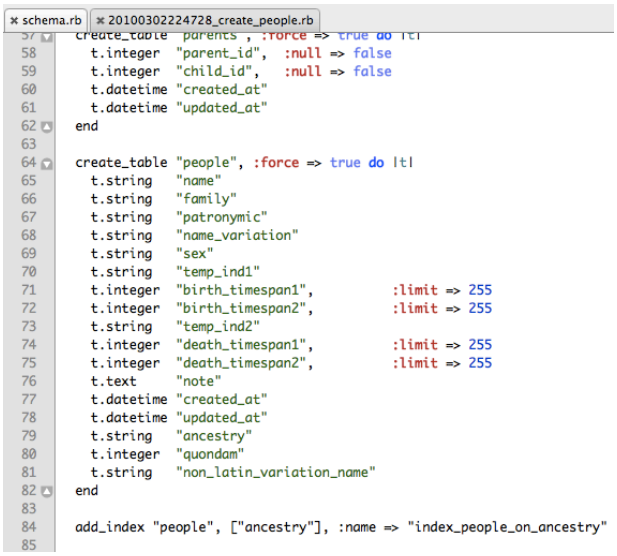


Figura 7. Porzione del database.

La stessa operazione viene effettuata per tutte le tabelle che vogliamo generare, incluse quelle di relazione fra due entità principali, dove andremo ad inserire ad esempio come attirbuti i riferimenti alle chiavi esterne.

Qualsiasi intervento successivo alla struttura del database, dalla aggiunta di attributi alle tabelle, alla rimozione o rinominazione delle "colonne" o delle tabelle stesse avviene attraverso le "migrazioni" o migration, che vengono dapprima generate con il comando `ruby script/generate migration <nome migrazione>` e poi applicate con il comando visto precedentemente `rake db:migrate` appunto. Il sistema prevede anche la possibilità di tornare indietro nel tempo nella struttura del db e di ritornare ad una versione precedente della struttura.

Una volta definita la struttura del db, si agisce sui modelli per impostare le relazioni con le restanti tabelle, si definiscono così le relazioni uno a molti, uno a uno o molti a molti che abbiamo trovato nello schema logico.

Nello schema si vede come ci sia un'associazione molti a molti tra la persona e la bibliografia, dato che una persona può avere molte attestazioni che la citano e una fonte può citare molte persone. L'associazione è mediata dalla relazione attestazione_persona, la quale mette in

correlazione il codice identificativo della persona con il codice identificativo della fonte che la cita.
Nel modello tutto questo si traduce in :

```
class Person < ActiveRecord::Base
  has_many :person_attestations,:dependent => :destroy
  has_many :bibliographies, :through => :person_attestations
end
```

e dall'altra parte:

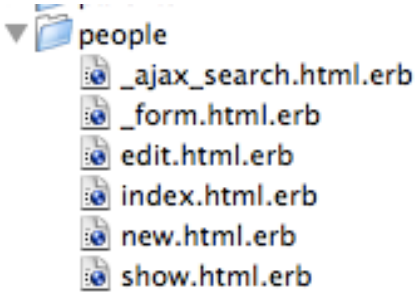
```
class Bibliography < ActiveRecord::Base
  has_many :person_attestations, :dependent => :destroy
  has_many :people, :through => :person_attestations
end

class PersonAttestation < ActiveRecord::Base
  belongs_to :person
  belongs_to :bibliographies
end
```

e così via per tutte le relazioni individuate nello schema. Le parole chiave sono `has_many` e `belongs_to`.

4.4.2.Viste

A seguito del comando `ruby script/generate scaffold` ecc, Rails genera automaticamente una serie di pagine. Tra le tante genera anche le viste, o i file con estensione `.html`.



Solitamente i file sono quattro, `index`, `show`, `new` e `edit`, rispettivamente il primo visualizza un elenco dei dati inseriti, il secondo il dettaglio di un istanza della tabella, il terzo e il quarto creano una nuova istanza o la modificano. Nell'esempio riportato ci sono inoltre alcuni file relativi a funzionalità aggiuntive.

Figura 8. Esempio di file generati dallo scaffold

A video questo si traduce con:

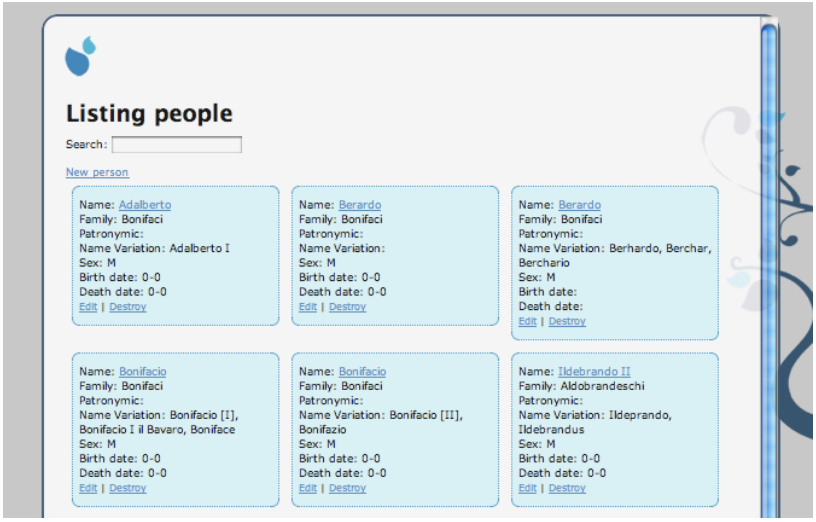


Figura 9. index.html.erb

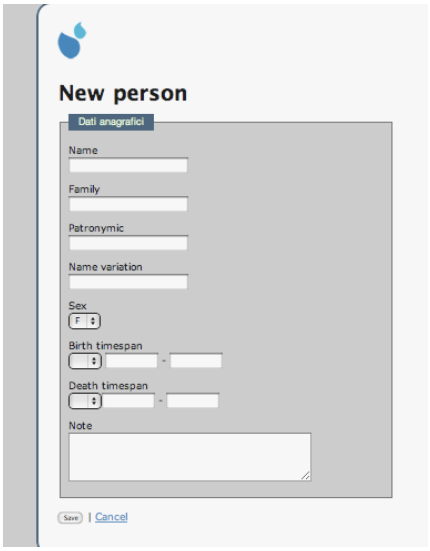


Figura 10. new.html.erb

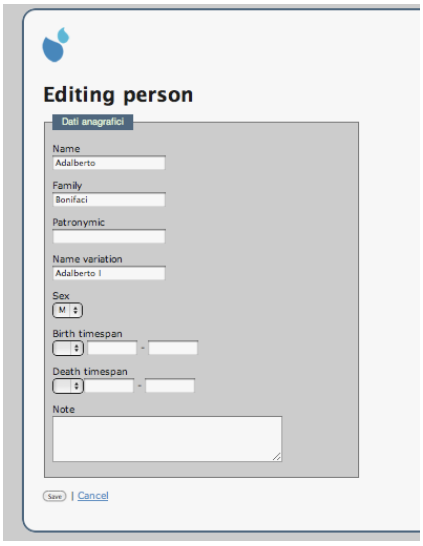


Figura 11. edit.html.erb


```
<h3>People Details</h3>
<p>
  <b>Name:</b>
  <%=h @person.name %>
</p>

<p>
  <b>Family:</b>
  <%=h @person.family %>
</p>

<p>
  <b>Patronymic:</b>
  <%=h @person.patronymic %>
</p>

<p>
  <b>Name variation:</b>
  <%=h @person.name_variation %>
</p>
```

Figure 11 e 13. show.html.erb

La parte di codice sottostante è un misto fra HTML e Ruby annotato, delimitato da un tag di apertura e chiusura, esattamente come in molti altri linguaggi come PHP o ASP. Nello specifico il tag utilizzato per l'inclusione di Ruby è <%= . . . %> quando il risultato dell'esecuzione del

codice Ruby deve essere stampato all'interno dell'HTML, `<% ... %>` quando serve per cicli e condizioni.

Nel codice dell'esempio si nota l'uso della variabile `@person`. Questa notazione nello specifico recupera dal database il dato inserito e lo stampa nella pagina html. L'impostazione della variabile e del codice che estrae i dati del database si trova nel controller.

Per rimanere fedeli alla filosofia DRY (*Don't repeat yourself*), la pagina di impostazione del layout dell'intero sito è unica, si chiama `application.html.erb` e si trova nella cartella `views/layout`. Viene impostata una volta sola l'intestazione del sito, il titolo il menù e quant'altro fa parte del layout portante del sito. In luogo del contenuto c'è un *placeholder* che carica i vari file html posizionati nelle cartelle di appartenza.

Un'altro utile sistema è quello di usare i "partial", si può utilizzare più volte ed evita di ripetere porzioni di codice. Con i partial è possibile estrarre una porzione di codice da una vista e salvarla in un file esterno; il codice estratto può essere condiviso da differenti viste e può essere tanto codice statico che codice dinamico.

È utile ad esempio per sostituire con un'unica pagina (`_form.html.erb` - l'underscore davanti è una convenzione per indicare che è un partial) le due di creazione e modifica di una persona.

Sia people/new.html.erb

```
<h1>New person</h1>
 <%= render :partial => 'form' %>
```

che people/edit.html.erb

```
<h1>Editing person</h1>
 <%= render :partial => 'form' %>
```

rimandano a _form.htm.erb

```
<% form_for(@person) do |f| %>
  <%= f.error_messages %>
<fieldset>
      <legend>Dati anagrafici</legend>

  <p>
    <%= f.label :name %><br />
    <%= f.text_field :name %>
  </p>
```

```
  <p>
    <%= f.label :family %><br />
    <%= f.text_field :family %>
  </p>
  <p>
    <%= f.label :patronymic %><br />
    <%= f.text_field :patronymic %>
  </p>
  <p>
    <%= f.label :name_variation %><br />
    <%= f.text_field :name_variation %>
  </p>
  <p>
    <%= f.label :sex %><br />
    <%= f.select(:sex, %w[F M]) %>
  </p>

  <p> Birth timespan <br />
     <%= f.select(:temp_ind1, %w[ v. p. d.], { :include_blank =>
true }) %> <%= f.text_field :birth_timespan1, :size => 10 %>
           -
         <%= f.text_field :birth_timespan2, :size => 10 %>
       </p>
       <p>Death timespan <br />
              <%= f.select(:temp_ind2, %w[ v. p. d. q.],
{ :include_blank => true }) %><%=
f.text_field :death_timespan1, :size => 10 %>
           -
         <%= f.text_field :death_timespan2, :size => 10 %>
       </p>
  <p>
    <%= f.label :note %><br />
    <%= f.text_area :note, :rows => 4 %>
  </p>

</fieldset>


  <p>
    <%= f.submit 'Save' %> | <%= link_to 'Cancel', @person %>
  </p>
<% end %>
```

In questo modo il codice relativo al form viene scritto una volta sola. Tutti questi accorgimenti rendono la codifica del progetto facile da mantenere sul lungo tempo e immediata nelle modifiche.

4.4.3.Controllers

I controllers sono il trattino d'unione fra modelli e viste. Il controller è dove vengono gestiti i dati estratti dal database e resi disponibili alla vista. Al loro interno vengono impostate le variabili utilizzate dalle viste e vengono altresì impostati i passaggi che il browser invia a seguito di una richiesta dell'utente sull'interfaccia.

```
class PeopleController < ApplicationController
  # GET /people
  # GET /people.xml
  def index
    @people = Person.all :order => :name

    respond_to do |format|
      format.html # index.html.erb
      format.xml  { render :xml => @people }
    end
  end

  # GET /people/1
  # GET /people/1.xml
  def show
    @person = Person.find(params[:id])

    respond_to do |format|
      format.html # show.html.erb
      format.xml  { render :xml => @person }
    end
  end
```

Figura 14. Parte del controller

C'è un controller per ogni modello, così come a sua volta c'è una cartella views per ogni modello. Prendiamo in considerazione il controller del modello Persona. Come si può notare vengono definite le funzioni `index` e `show`, che gestiscono la visualizzazione dei file `views/people/index.html.erb` e `views/people/show.html.erb`.

La funzione `index` assegna alla variabile `@people` tutti gli attributi di ogni istanza `Person`. Se si vuole intervenire sulla selezione dei dati da visualizzare nella vista, è qui che si deve agire. Si può notare come sia stato impostato l'ordinamento alfabetico per il campo nome. La riga di codice che stiamo prendendo in considerazione, equivale ad un istruzione SQL tipica di un interrogazione ad un database, come `SELECT * from PEOPLE, Order by People.Name`.

Questa sono i concetti base di funzionamento di Rails, il controller non fa solo questo ma molto di più, recupera parametri, gestisce vari tipi di interrogazioni al db, imposta le ricerche, redireziona le pagine a seconda delle esigenze, imposta le sessioni degli utenti, il log in e il log out degli stessi. Per la documentazione esaustiva sul framework si rimanda ai manuali preposti, l'intezione era di fare una panoramica veloce sulle funzionalità e su alcune delle caratteristiche implementate nel progetto.

4.4.4.Gli helpers

Un'altro utile strumento messo a disposizione dal framework che stiamo utilizzando sono gli *helpers*. Sono dei moduli, che contengono metodi che aiutano le viste. Servono per evitare di mettere troppo codice annotato nei file .html che devono restare il più possibile file html.

Nel nostro caso sono stati utilizzati per costruire le citazioni. Come si può vedere dalla figura viene definita la funzione, la funzione costruisce un array i cui elementi vengono estrapolati, se presenti, dal database. Viene altresì aggiunta la punteggiatura utile alla costruzione della citazione, personalizzata per ogni tipo di testo. Vengono utilizzati i cicli per costruire l'array.

```
module PrimariesHelper
  def primary_bibliography primary
    case
    when primary.type == "Monography"
       result = []
       result << " #{h primary.author }" if primary.author.present?
       result << ", <em>#{h primary.title }</em>" if primary.title.present?
       result << ", ed. by #{h primary.editor }" if primary.pub_place.present? and primary.language == "e
       result << ", a cura di #{h primary.editor }" if primary.pub_place.present? and primary.language ==
       result << ", #{h primary.pub_place }" if primary.pub_place.present?
       result << ", #{h primary.publisher }" if primary.pub_place.present?
       result << ", #{h primary.pub_date_b }" if primary.pub_date_b.present?
       result << "- #{h primary.pub_date_e }" if primary.pub_date_e.present?
       result << ", #{h primary.volume_tot }" if primary.volume_tot.present?
       result << "."
       result.join ''

     when primary.type == "Miscellany"
       result = []
       result << "<em>#{h primary.title }</em>" if primary.title.present?
       result << ", ed. by #{h primary.editor }" if primary.pub_place.present? and primary.language == "e
       result << ", a cura di #{h primary.editor }" if primary.pub_place.present? and primary.language ==
       result << ", #{h primary.pub_place }" if primary.pub_place.present?
       result << ", #{h primary.publisher }" if primary.pub_place.present?
       result << ", #{h primary.pub_date_b }" if primary.pub_date_b.present?
       result << "- #{h primary.pub_date_e }" if primary.pub_date_e.present?
       result << ", #{h primary.volume_tot }" if primary.volume_tot.present?
       result << "."
       result.join ''
```

Figura 15 . Esempio di utlizzo di un helpers

Nella vista corrispondente viene richiamata la funzione.

```
  <ul>
    <li><%= primary_bibliography primary %> <br />
<small>
    <%= link_to 'Show', primary %>
    <% if session[:user_id] %>
    |
    <%= link_to 'Edit', edit_polymorphic_path(primary) %>   |
    <%= link_to 'Destroy', primary, :confirm => 'Are you sure?', :method => :delete %>
    <% end -%>
</small>
</li>
```

Figura 16. Richiamo all'helper nel codice html.

Gli helpers possono essere impostati secondo le esigenze dell'utente e riutilizzati nel codice semplicemente richiamandoli all'occorrenza. Esistono vari tipi di helpers, da quelli personalizzati a quelli generici che il framework usa per generare codice html in modo automatico.

4.4.5. Plug-in aggiuntivi

Oltre ai moduli che implementano il modello MVC, Rails comprende altri moduli che facilitano la realizzazione di particolari funzionalità, come la realizzazione di Web Service con ActiveResource o la generazione e la spedizione di email con ActionMailer. Come tutti i moduli del framework, anche questi sono indipendenti, utilizzati dal controller nella logica applicativa.
Quando necessario è possibile estendere le funzionalità di Rails installando plugin sviluppati da terzi o librerie Ruby.
Mentre l'utilizzo di librerie Ruby permette di accedere a funzionalità di basso livello, i plugin implementano funzionalità complete come la gestione degli utenti o il versioning dei dati sul database, e possono estendere un modulo specifico oppure più moduli del framework.
Lo stesso team di sviluppo del core di Rails ha adottato la forma del plugin per estrarre alcune funzionalità considerate non strettamente indispensabili al framework e renderle disponibili solo se richieste.
L'utilizzo di plugin rappresenta non solo un'opportunità per risparmiare codice e concentrarsi sulle particolarità della propria applicazione, ma anche l'occasione per studiare le soluzioni applicate ai problemi più ricorrenti della programmazione Web.
Nel progetto sono stati utilizzati tre plug-in aggiuntivi:

- Paperclip[4], per la gestione delle foto da abbinare alla bibliografia;
- Rdiscount[5], per gestire le citazioni;
- Ancestry[6], per gestire i rapporti di parentela e la creazione di alberti genealogici.

Quest'ultimo in particolare è stato installato ma non approfondito.

Per preservare il progetto da un malfuzionamento dovuto agli aggiornamenti del sistema in generale o di una qualsiasi delle "gemme" che compongono il sistema, è possibile "congelare" il tutto, in modo da rendere l'applicazione autosufficiente. In questo modo i plugin installati saranno sempre con l'applicazione e non dipenderanno dall'ambiente di sviluppo dove il progetto verrà spostato.

4.5.Grafica

4.5.1.Layout

Il sito presenta una struttura omogenea per tutte le pagine in maniera da risultare coerente graficamente e non generare nell'utente ambiguità.

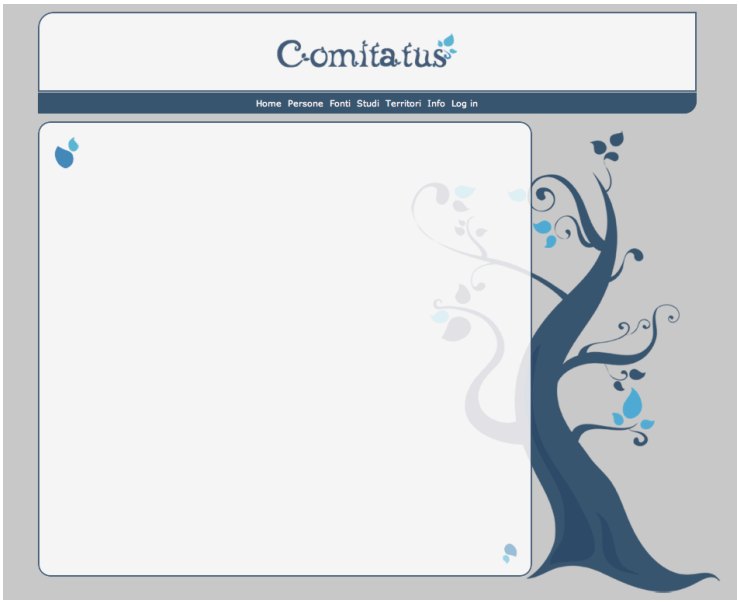


Figura 17. Esempio del layout dell'interfaccia

[4] http://github.com/thoughtbot/paperclip

[5] http://github.com/rtomayko/rdiscount

[6] git://github.com/stefankroes/ancestry.git

Come si può vedere nella figura 13, il titolo è posizionato in alto, il menù subito sotto e spostato sulla sinistra si può trovare il riquadro dove andranno tutti i contenuti del sito, sia dinamici che statici.

Tutti gli elementi della struttura hanno le dimensioni espresse in modo relativo (con le percentuali) e non assoluto in modo da permettere il ridimensionamento e l'adattamento della pagina a più risoluzioni dello schermo. Sembrava riduttivo dover progettare solo per la risoluzione 800 x 600 andando così a penalizzare chi utilizza schermi più grandi (ad esempio 1280x1024). La visualizzazione ottimale è comunque per uno schermo 1024 x 768.

4.5.2.Testo

È stato scelto per la parte testuale un carattere senza grazie, in particolare il Lucida Sans Unicode (di base, poi altri caratteri senza grazie in alternativa), lo stesso per ogni elemento del progetto. Per facilitare la lettura delle parti testuali l'interlinea è stata impostata ad una volta e mezzo il corpo del carattere. Anche il corpo del carattere è espresso in percentuale, in modo da adattarsi anch'esso alla risoluzione dello schermo. Per il CSS per la stampa il carattere di base scelto resta il Lucida Sans pur essendo un carattere senza grazie e quindi non indicato in teoria per la lettura su cartaceo ma capace di mantenere un'ottima leggibilità.

4.5.3.Navigabilità

I link sono tutti testuali e definiti dall'attributo 'title' per agevolare gli screen readers. Inoltre seguono tutti uno stesso sistema di caratterizzazione: sono bianchi e di carattere minuscolo all'inizio, al passaggio del mouse cambiano colore e vengono sottolineati in modo da non veicolare l'informazione solo tramite il colore ma anche attraverso un segno grafico. Una volta visitato il link diventa più scuro per indicare appunto lo stato di 'già visitiato'.
Dove è stato necessario è stata resa la navigazione tramite 'avanti e indietro', in particolare per agevolare la navigazione nelle sezioni di immissione dei dati e poterne così usufruirne in maniera continuativa senza dover sempre tornare al menù iniziale per passare alla pagina successiva (o precedente).

Infine, per finire la panoramica sugli accorgimenti utilizzati per migliorare l'accessibilità del sito tutte le immagini sono state dotate dell'attributo "alt" per avere un'alternativa testuale all'immagine stessa e sono anche accompagnate da un testo descrittivo sottostante. Inoltre è stata portata particolare attenzione al tag title nello head per segnalare in modo corretto la pagina visualizzata nel menù della finestra del browser.

4.5.4. Adattività

Il layout grafico è stato testato con quattro browser (Explorer 7.0, Google Chrome 5.0, Mozilla Firefox 3.6 e Opera 10.10) ed è risultato coerente su tutti e quattro. Come già detto tutte le misure sono espresse in percentuale, in questo modo il sito si adatta alla risoluzione dello schermo dell'utente, senza che l'utente debba intervenire.

4.6. Accessibilità

4.6.1. Validazione CSS

Tutti i fogli di stile associati sia al sito sono stati validati con il validatore del W3C all'indirizzo http://jigsaw.w3.org/css-validator.

4.6.2. Validazione XHTML

Tutte le pagine sono state validate per l'XHTML 1.0 con DTD Strict con il validatore del W3C all'indirizzo http://validator.w3.org/.

4.6.3. Valutazione con filtro per il contrasto di colori

È stata fatta inoltre una valutazione per il contrasto dei colori con il sito http://colorfilter.wickline.org/ e il risultato è apparso buono con i filtri di simulazione delle quattro tipologie di daltonismo, incluse le forme atipiche.

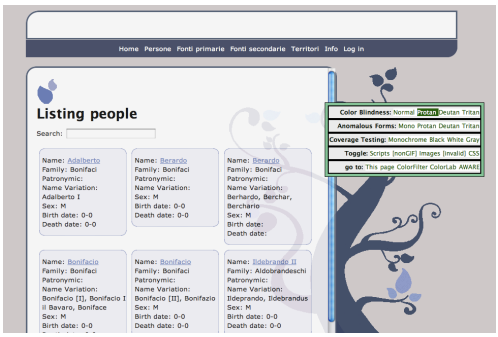


Figura 14. Pronatopia

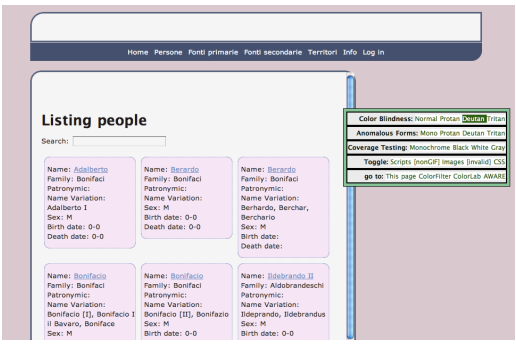


Figura 15. Deuteranopia

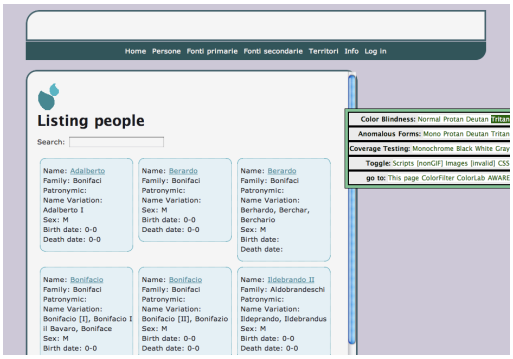


Figura 16. Tritanopia

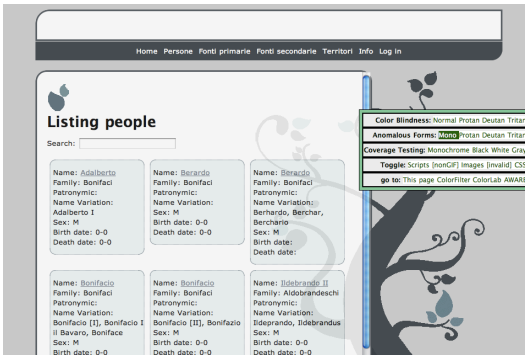


Figura 17. Acromatopsia

5. Conclusioni

La speranza del progetto è di essere, come ho già detto, un valido aiuto nell'analisi delle relazioni fra territori e grandi gruppi familiari del Regno Italico, tra il IX e XII sec.

Data la peculiarità delle informazioni, spesso frammentarie e di provenienza incerta, avere a disposizione un valido strumento che permetta una facile catalogazione, archiviazione dei dati e il loro confronto, può essere di grande aiuto per chiunque studi tale argomento.

Il progetto Comitatus svolge efficacemente tale funzione, permettendo, come abbiamo visto durante la dissertazione, l'inserimento dei dati anagrafici relativi alle personalità storiche del periodo in esame, dei territori da essi amministrati, degli studi relativi ai ceti familiari dominanti e delle attestazioni che ne trattano.

Il progetto è aperto a numerosi sviluppi futuri, come ad esempio l'indagine genealogica, che è stata solo parzialmente integrata.

Un ambito di studio correlato potrebbe essere lo studio dell'origine e della distribuzione dei cognomi attraverso l'uso di modelli matematici, che il Dipartimento di Fisica "E. Fermi" sta portando avanti

La flessibilità del sistema utilizzato e la possibilità di ampliamento, tramite plug-in esterni dell'ambiente di sviluppo del progetto, ne permette una dinamicità tale da poter esser facilmente modellato e modificato per ambiti di studio differenti.

6. Bibliografia

SAGGI E MANUALI

Violante, Cinzio (a cura di). 1988. *Formazione e strutture dei ceti dominati nel medioevo: marchesi, conti e visconti nel Regno Italico (secoli IX-XIII). Atti del convegno.* Roma, Istituto Palazzo Borromini.

Spicciani, Amleto (a cura di). 2003. *Formazione e strutture dei ceti dominati nel medioevo: marchesi, conti e visconti nel Regno Italico (secoli IX-XIII). Atti del convegno*. Roma, Istituto Palazzo Borromini.

Fumagalli, Vito. 1978. *Il Regno italico*. Torino, UTET.

Prosdocimi, Luigi, e Piero Zerbi (a cura di). 1977. *Le istituzioni ecclesiastiche della "Societas Christiana" dei secoli XI – XII. Diocesi, pievi e parrocchie. Atti della sesta settimana di studio.* Milano, Vita e Pensiero.

Redondi, Pietro. 2003. Leggere e scrivere. *Un ABC della ricerca bibliografica e della composizione di ricerca.* Milano, Università degli studi di Milano – Bicocca.

Delogu, Paolo. 1994. *Introduzione allo studio della storia medioevale.* Bologna, il Mulino.

Atzeni, Paolo, Stefano Ceri, Stefano Paraboschi e Riccardo Torlone. 2002. *Basi di dati. Modelli e linguaggi di interrogazione.* Milano, McGraw-Hill.

Ruby, Sam, Dave Thomas e David Heinemeter Hansson. 2009. *Agile Web Development with Rails.* Dallas (Texas, USA), The Pragmatic Bookshelf.

Beaumont, Andy, Jon James, Jon Stephens e Chris Ullman. 2002. *Form per il web. Progettazione e usabilità del sito.* Milano, Ulrico Hoepli Editore.

Briggs, Owen, Steven Champeon, Erico Costello e Matt Patterson. 2002. *Cascading Style Sheet (CSS). Fogli di stile per il web.* Milano, Ulrico Hoepli Editore.

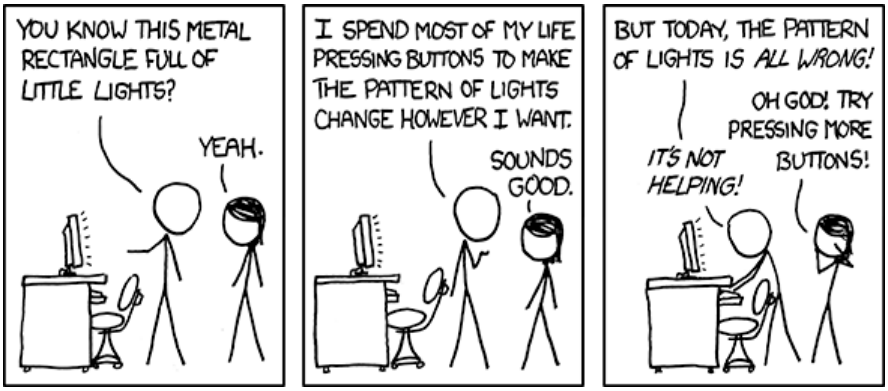
St. Laurent, Simon, e Edd Dumbill. 2008. *Learning Rails*. Sebastopol (California, USA), O'Reilly Media.

Guida Ruby On Rails 2 di Luca Bozzo su html.it.
http://ruby.html.it/guide/leggi/151/guida-ruby-on-rails-2/ (visitato il 13 aprile 2010)

Clark, Mike. 2008. *Advanced Rails Recipes.* Dallas (Texsas, USA), The Pragmatic Bookshelf.

Ceresa, Marco. 2006. *Ruby per le applicazioni web.* Milano, APOGEO.

COMPUTER PROBLEMS



Alt = This is how I explain computer problems at my cat. My cat usually seems happier than me.

http://xkcd.com/722/