Reminiscens, una knowledge base di risorse storiche per supportare la reminiscenza

Tesi di laurea triennale

Nicola Parrello
Università degli studi di Trento
Via Sommarive, 14
38123 Povo (Trento)
parrello.nicola@gmail.com

ABSTRACT

Nella vita collezioniamo moltissimi ricordi, innumerevoli esperienze che plasmano le persone che siamo e che saremo. Con il passare degli anni aumenta per ognuno l'importanza della rievocazione di queste memorie, che possono avere una doppia valenza, in quanto terapeutiche per combattere la senilità e utili nella trasmissione ai posteri di quel che era. Proprio l'utilità di questo fenomeno spontaneo, chiamato reminiscenza, fa sorgere una domanda: come è possibile far sorgere la reminiscenza in un individuo? Questo documento contiene le scelte di implementazione che hanno portato alla costruzione di Reminiscens, una piattaforma sociale (ancora in fase di sviluppo) per favorire l'afflusso di ricordi provenienti dalla vita di una persona; in particolare, verrà presentato il lavoro da me compiuto, cioè quello di costruire una Knowledge Base atta a supportare gli scopi della piattaforma, correlata di strumenti per la lettura e la scrittura dei dati che vogliamo utilizzare per stimolare la reminiscenza.

1. INTRODUZIONE

E' successo a tutti, almeno una volta, di ritrovarsi a raccontare a qualcuno (amico, familiare o altro), episodi della propria vita passata. L'atto di raccogliere dalla memoria esperienze passate per condividerle con qualcuno, rendendo significativo il rapporto tra i due attori della conversazione, è chiamato reminiscenza, ed è riscontrabile in tutte le età dell'essere umano. Coleman distinse la reminiscenza in varie tipologie a seconda del modo in cui essa viene evocata e del suo scopo: tra queste troviamo quella detta "life review", che riguarda proprio la rievocazione di memorie lontane nel tempo, al fine di ricostruire quella che è stata la propria vita[3]. Questo fenomeno assume rilevanza massima per gli anziani: è infatti studiato come terapia per i malati di Alzheimer o affetti da altre forme di demenza, dato che il ricordare eventi piacevoli del passato aiuta a ristorare l'autostima e la soddisfazione personale, combattendo la malattia. Ma non è solo questo: la reminiscenza, in quanto interazione con un altro soggetto, può essere utile anche per combattere la depressione e l'isolamento in cui gli anziani spesso si trovano, favorendo allo stesso tempo un rapporto faccia a faccia, positivo per entrambi i partecipanti. Indicata la positività del fenomeno, è doveroso fare i conti con il fatto che nella maggior parte dei casi essa nasce in maniera spontanea, dando vita a un quesito importante: come trovare gli stimoli giusti per innescare la reminiscenza? Crediamo che la fonte più consona a fornire l'input adatto a farla scattare sia quella in cui lavoriamo più o meno tutti i giorni, cioè il web; dopo più di vent'anni di internet, la nostra storia collettiva è quasi tutta online: se oltre a questo consideriamo l'espansione dei dataset disponibili in forma strutturata, è chiaro che raccogliere da lì materiale audiovisivo, eventi e personaggi storici può fornire delle importanti fondamenta da cui partire. Ma possedere questi dati non è abbastanza: affinché gli stimoli siano effettivamente significativi, è necessario ideare un meccanismo atto a identificare quale parte dei contenuti raccolti dal web è rilevante per la storia della vita di una persona. Proprio a partire da questo problema si sviluppa il mio lavoro, esplicitato nei seguenti paragrafi, ma la sua trattazione ha bisogno di una digressione sulle tecnologie che nei diversi ambiti rappresentano il progresso raggiunto.

2. STATO DELL'ARTE

A partire dal problema che cerchiamo di risolvere, è importante iniziare a calarsi nel concreto, rendendosi conto di quali sono i campi a cui ci si vuole ispirare per costruire un'architettura funzionale alla soluzione della questione.

2.1 Web semantico

con il termine "web semantico", coniato dal suo ideatore Tim Berners-Lee, si intende "la trasformazione del World Wide Web in un ambiente dove i documenti pubblicati (pagine HTML, file, immagini, e così via) sono associati ad informazioni e dati che ne specificano il contesto semantico in un formato adatto all'interrogazione e l'interpretazione (e.g. tramite motori di ricerca) e, più in generale, all'elaborazione automatica."[1] Ciò non significa limitare le numerosissime forme in cui è possibile presentare i contenuti sul web, bensì è un modo per fornire una struttura capace di rispondere a delle query, permettendo la lettura automatica di pagine e documenti. Risorse con questa caratteristica sono enormemente utili per la facilità con la quale è possibile estrarre

[1] http://it.wikipedia.org/wiki/Web_semantico

```
<div>
   <div>Nicola Parrello</div>
   <div>UNITN</div>
   <div>123-123123</div>
   <a href="mailto:bar@foo.com">Email</a>
</div>
```

```
<div class="vcard">
   <div class="fn">Nicola Parrello</div>
   <div class="org">UNITN</div>
   <div class="tel">123-123123</div>
   <a class="email" href="mailto:bar@foo.com">Email</a>
</div>
```

Figure 1: Un contatto senza e con il microformato hCard

informazioni e adattarle alle proprie necessità; sono quindi queste la scelta più ovvia come semplice punto di partenza per la ricerca di contenuti atti a stimolare la reminiscenza. Una delle realtà più grosse che opera nel web semantico è DBpedia, il cui approccio è quello di effettuare periodicamente il parsing dei dump rilasciati da Wikipedia, estrarre le uniche informazioni strutturate delle pagine (le cosiddette infobox), unendo alcuni attributi delle versioni localizzate dello stesso articolo e rappresentando le risorse tramite Resource Description Framework (RDF). Ogni risorsa è definita tramite un URI, un identificatore unico dell'entità presentata. Il lavoro di DBpedia è eccellente, ma è involontariamente ostacolato dagli editori di Wikipedia, che per la compilazione delle infobox non hanno dei template ben definiti, finendo per rappresentare lo stesso attributo in articoli diversi con nomi diversi; DBpedia è interrogabile tramite SPARQL, un linguaggio SQL-like costruito per leggere RDF. Dopo il rilascio del primo dataset al pubblico nel 2007, la situazione attuale è quella di un grosso grafo composto da molti altri set di dati collegati tra di loro tramite l'accoppiamento di risorse appartenenti a insiemi diversi, ma rappresentanti la stessa entità. Pubblicare questi Linked Data rende molto più facile la ricerca di informazioni sul web, rendendole molto più precise e complete. Altri progetti di web semantico sono: Freebase, che a differenza di DBpedia è un progetto proprietario; Uniprot, una knowledge base contenente dati liberamente accessibili sulle sequenze di proteine; GeoNames, un database geografico contentente oltre 10 milioni di luoghi, accessibile e scaricabile sotto licenza Creative Commons. È disponibile online lo stato dei collegamenti tra datasets semantici in forma di grafo, aggiornato a settembre 2011[2], mentre una descrizione di DBpedia in particolare è reperibile in [1].

Un approccio diverso nell'attribuzione di un significato ai contenuti presenti sul web è quello dei microformati[3], un'estensione di markup che, tramite l'utilizzo degli attributi HTML class, rel e rev, consente l'attribuzione di regole semantiche a normali pagine web: figura 1 mostra la differenza tra le informazioni sul contatto di Nicola Parrello in maniera canonica e le stesse, se riscritte con il microformato hCard (specifico per i contatti).

Attraverso i microformati, ad esempio, un software come un browser può estrarre facilmente informazioni e navigare le relazioni tra oggetti diversi, mantenendo comunque la normale leggibilità di una pagina web. Oltre ad hCard, solo hCalendar è stato formalizzato: altri microformati, come hAtom, hMedia e hNews (rispettivamente per feed Atom, contenuti multimediali e notizie) sono solo più o meno abbozzati, e non rappresentano quindi uno standard.

2.2 Web search e contestualizzazione dell'informazione

Il passo successivo nella ricerca di una soluzione ad problema che è tema di questo documento, dopo aver scelto da dove reperire i dati, è quello di decidere come permettere a un eventuale utente di accedere ai contenuti che vogliamo proporre. L'approccio che sembra più naturale è quello di costruire un piccolo motore di ricerca, in modo tale da creare una maschera che permetta una richiesta di risorse in maniera uniforme, anche se da fonti diverse; inoltre, per avere una ricerca rapida e reattiva, il buon senso suggerisce che è una buona idea quella di indicizzare il materiale ricercabile. In questo campo è difficile non considerare Google come lo stato dell'arte, essendo loro il search engine più utilizzato al mondo. Perché utenti da ogni parte del globo possano utilizzare un servizio veloce ed efficiente, Google utilizza sette componenti dinamiche che salvano e leggono dati in altre strutture; il procedimento può essere riassunto così:

1. Il Server URL parte da un URL di base e, leggendo il Document Index, invia gli URL ai Crawler;
2. I Crawler scaricano le pagine web e le mandano nello Store Server.
3. Lo Store Server comprime le pagine nel formato zlib (RFC1950[4]), riducendo la loro dimensione a un terzo dell'originale, che vengono poi di un docId univoco e immagazzinate nel repository.;
4. Il repository viene letto dall'Indexer, che decomprime i documenti e li parsa, assegnando a ogni parola (a cui viene assegnato un wordId univoco) informazioni su posizione, grandezza del testo, numero di occorrenze e altro; ogni voce viene poi aggiunta ad un indice parzialmente ordinato. Dalle pagine lette, l'Indexer estrae anche i dati dei link in esse contenuti, oltre a servirsi del testo analizzato per costruire un lessico, utile per la vera e propria funzione di ricerca;
5. Lo URL Resolver combina i documenti con i dati dei link, per costruire tabelle di coppie di docId, utilizzati per calcolare il PageRank;
6. Il Sorter riordina l'indice (ordinato per docId) in un indice inverso (ordinato per wordId), aggiungendo altri dati per rendere la ricerca più precisa;
7. Infine viene calcolato il PageRank, una misura dell'importanza di una pagina web calcolata in base a quali e quante sono le pagine che hanno dei link che puntano a essa.

[2] `http://lod-cloud.net/versions/2011-09-19/lod-cloud.html`

[3] `http://microformats.org/`

[4] `http://www.ietf.org/rfc/rfc1950.txt`

Per una trattazione più approfondita, rimandiamo a [2].

Negli anni Google è diventato colmo di pubblicità, mostrata e scelta in base ai dati raccolti dalle ricerche degli utenti, ma certamente non è l'unica possibilità in quanto a motori di ricerca. Un esempio su tutti è quello di Duck Duck Go che, a differenza di Google, è un Semantic Search Engine. Il sistema cioè si occupa di valutare l'effettivo significato dei termini di ricerca, eliminando in maniera più precisa i risultati irrilevanti. Ma quello che lo contraddistingue in maniera maggiore dagli altri è la sua attenzione alla privacy: Duck Duck Go infatti non conserva e non vende a terzi nessuna informazione sulle ricerche, permettendo una ricerca anonima e al sicuro da data leak, richieste legali da parte delle istituzioni e disonesti.[5] Quello che può infastidire gli utenti, cioè gli annunci di cui sopra, sono però molto interessanti per la nostra ricerca, in quanto introducono un altro aspetto molto importante: perché la ricerca sia semplice da utilizzare, e soprattutto piacevole, può essere necessario renderla per così dire automatica, nascondendo l'azione manuale dell'inserimento di parametri e mostrando i risultati direttamente come contesto di qualche altra azione. Premiata nel 2012 dal magazine Popular Science come innovazione dell'anno[6], Google Now è un'estensione dell'applicazione mobile Google Search che, oltre ad essere un assistente vocale, analizza abitudini, ricerche e posizioni ricorrenti per fornire dati e risultati contestuali, mostrando all'utente le informazioni prima che egli ne faccia richiesta esplicita all'app.

2.3 Algoritmi spazio-temporali

Abbiamo definito la raccolta e la ricerca, ma un ultimo punto rimane fumoso: in base a quale criterio scegliere un set di risorse rispetto a un altro, con la condizione che queste siano effettivamente rilevanti ed efficaci nella stimolazione della reminiscenza? Il vissuto di un individuo può essere riassunto in una lista di eventi, di storie di vita; allo stesso tempo, un evento può essere identificato da una coppia <D,L>, dove D è la coordinata temporale e L descrive la posizione nello spazio. Utilizzando questo modello, il criterio di cui sopra diventa quello della distanza spazio-temporale tra due entità, e il calcolo di questa distanza diventa il mezzo per effettuare un'indicizzazione preliminare delle risorse.

Un interessante esempio di sfruttamento estensivo delle informazione su tempo e spazio al fine di restituire dei risultati a una query è TimeTrails. TimeTrails è un sistema per l'estrazione e l'esplorazione di coordinate spazio-temporali che si possono trovare nei documenti di testo, composto di tre componenti principali: (1) una pipeline che, dopo aver letto i dati forniti da dei moduli che estraggono documenti di testo da varie sorgenti (e.g. la vetrina di Wikipedia[7]), si occupa di estrarre le date e i luoghi contenuti nel testo, normalizzarli (e.g. da "25 luglio 1991" a "25/07/1991" per le prime e da "Pergine Valsugana" a "46.06853620, 11.23528960" per i luoghi), e calcolare il numero e la posizione delle occorrenze trovate, in modo da verificare se la coordinata dello spazio e quella del tempo identificano un evento ben preciso oppure se sono due riferimenti senza alcuna correlazione. Il risultato dell'elaborazione, cioè il documento originale unito a una sequenza ordinata di tuple <D,L>, viene poi salvato in (2) un database ottimizzato per contenere dati su luoghi[8], dal quale (3) un'interfaccia permette la ricerca testuale di documenti, con un risultato che verrà mostrato su una mappa sotto forma di traiettoria, rappresentata da tutte le tuple lette dal db. Nel caso la query dovesse restituire più di un documento, può risultare interessante vedere dove e quando le traiettorie si incrociano: se i documenti sono biografie, l'intersezione delle traiettorie in uno dei punti segnati dalle tuple <D,L> può indicare che i personaggi si sono incontrati. TimeTrails è presentato dai suoi ideatori in [6].

[5] `https://duckduckgo.com/privacy`
[6] `http://www.popsci.com/bown/2012/product/google-now`
[7] `http://it.wikipedia.org/wiki/Wikipedia:Vetrina`

3. PROBLEMA

Nelle ultime righe dell'introduzione è stato presentato il problema che il lavoro presentato ha tentato di risolvere, cioè come sfruttare l'enorme quantità di informazione reperibile sul web per innescare automaticamente la reminiscenza in un individuo. La questione non è scontata, in quanto è necessario raggiungere un compromesso tra qualità dell'informazione, facilità nel reperimento della stessa e velocità di elaborazione, quest'ultima indispensabile se l'obiettivo è la costruzione di un sistema accessibile al pubblico e capace di servire un buon numero di utenti contemporaneamente. Inoltre, tralasciando tutto ciò che non è visibile all'utilizzatore finale, è indispensabile formalizzare il fulcro del problema in termini di input e di output, e partire da questa formalizzazione per un'analisi più approfondita.

Introduciamo quindi il concetto di contesto, definendolo come un set di elementi (immagini, eventi, personaggi famosi, musica, libri, etc..) selezionati in maniera tale da stimolare l'afflusso di ricordi nel soggetto a cui vengono presentati. L'oggetto della trattazione diventa perciò, a partire da informazioni geografiche e cronologiche rappresentanti degli eventi nella vita dell'utente, quello di restituire il contesto più vicino possibile alla vita di questa persona. Formalizzando il problema, esso può essere definito nel seguente modo:

In primo luogo consideriamo l'insieme M, contenente le risorse con le quali costruire il contesto, definito come segue:

$$M = \{\langle nome, URL, categoria \in MCAT, tipo \in MTYPE, D, L\rangle\}$$
$$MTYPE = [foto, canzone, evento, personaggio]$$
$$MCAT = [evento\ sportivo, evento\ politico, cartolina, ...]$$
$$D = \langle decade, anno, mese, giorno\rangle$$
$$L = \langle luogo\ testuale, città, stato, lat, lon\rangle$$

Prendendo in input una timeline T, cioè una lista di coppie data-luogo

$$Timeline\ T = \{\langle D, L\rangle\}$$

il contesto C può essere rappresentato come

[8] `http://postgis.net/`

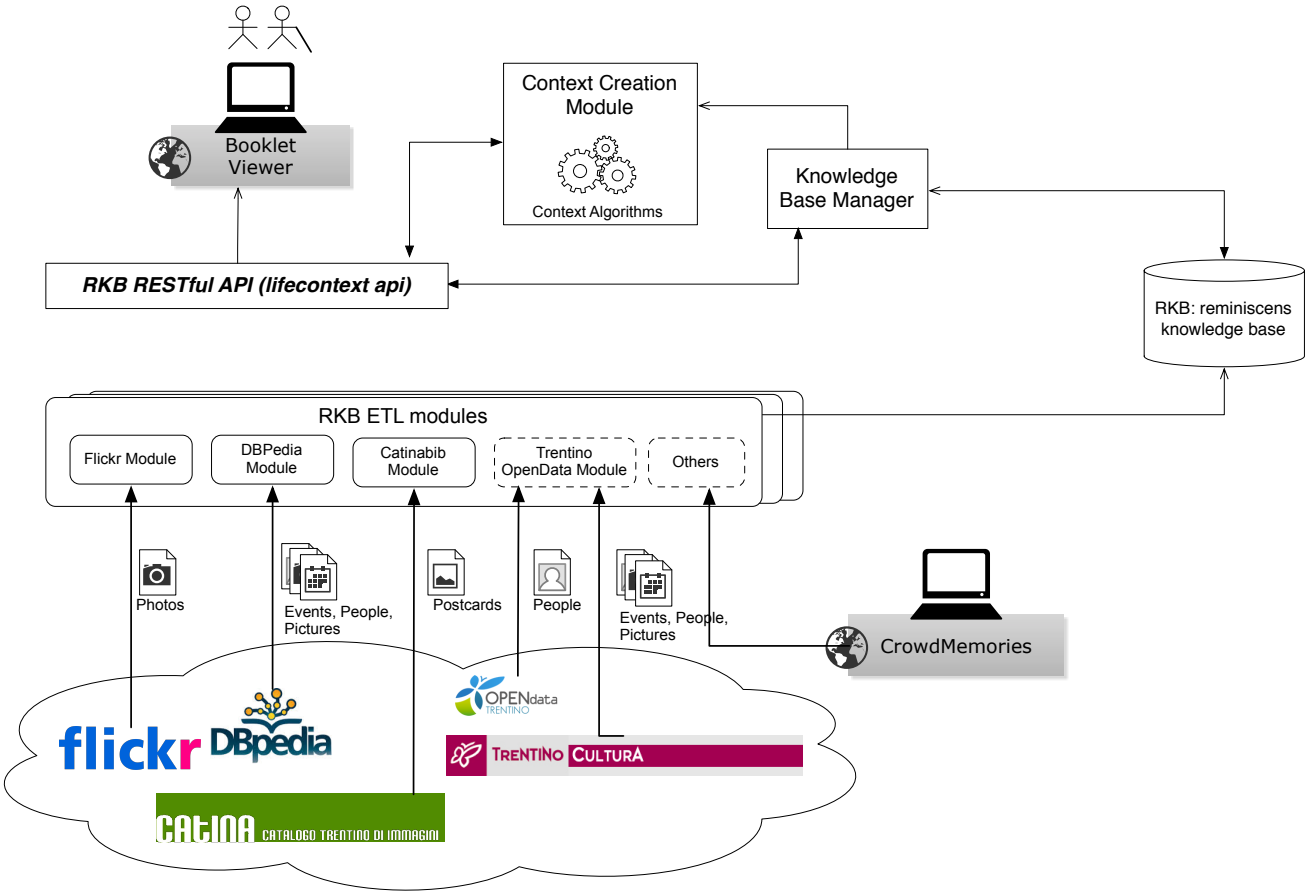

Figure 2: Architettura di Reminiscens

$$
\begin{aligned}
\textit{Contesto } C = \{\langle m, r\rangle : m \in M \\
&\wedge\ x = dist(m.data, t.data \in T) < \delta \\
&\wedge\ y = dist(m.luogo, t.data \in T) < \rho \\
&\wedge\ m.categoria \in MCAT \\
&\wedge\ r = ranking(m, x, y)\}
\end{aligned}
$$

dove:

- δ e ρ sono rispettivamente il limite della distanza spaziale e temporale tra gli oggetti della in T e quelli in M, una sorta di barriera tra gli oggetti da considerare e quelli che sicuramente non possono essere evocativi di memorie per l'utente;
- *dist* è una funzione della distanza tra due oggetti, calcolata spazio e tempo;
- ogni risorsa possiede anche un ranking r, cioè un peso che viene attribuito all'oggetto tenendo in considerazione anche la precisione associata ai luoghi e alle date associate agli elementi in M.

Dopo questa premessa, possiamo passare oltre ed andare ad analizzare le scelte fatte per riuscire a concretizzare la soluzione di cui sopra.

4. SOLUZIONE

Partendo dai paragrafi precedenti, riusciamo ora a delineare quelle che sono le possibili componenti software da costruire per i nostri scopi. La prima e indispensabile è un indice di risorse storiche, da consultare per ottenere le entità più pertinenti alla vita di un individuo; proprio per decidere cosa è attinente e cosa no, un algoritmo deve essere pensato e implementato. Affinché il risultato di tale implementazione sia scalabile e flessibile, deve essere scelto un meccanismo di accesso aperto e utilizzabile in più frangenti; ultima, ma non meno importante, la visualizzazione dei dati raccolti e filtrati, che devono essere presentati in una forma user-friendly, con attenzione particolare agli utenti di una certa età, che, rappresentando una fetta consistente dei possibili utilizzatori, richiedono una progettazione più attenta.

Nella realizzazione di un sistema capace di rispondere all'utente con dati relativi al tempo e al luogo in cui egli ha vissuto alcuni momenti della sua vita, la prima difficoltà è riscontrabile nella scelta riguardo a quale meccanismo utilizzare per recuperare le entità con cui costruire il contesto; per

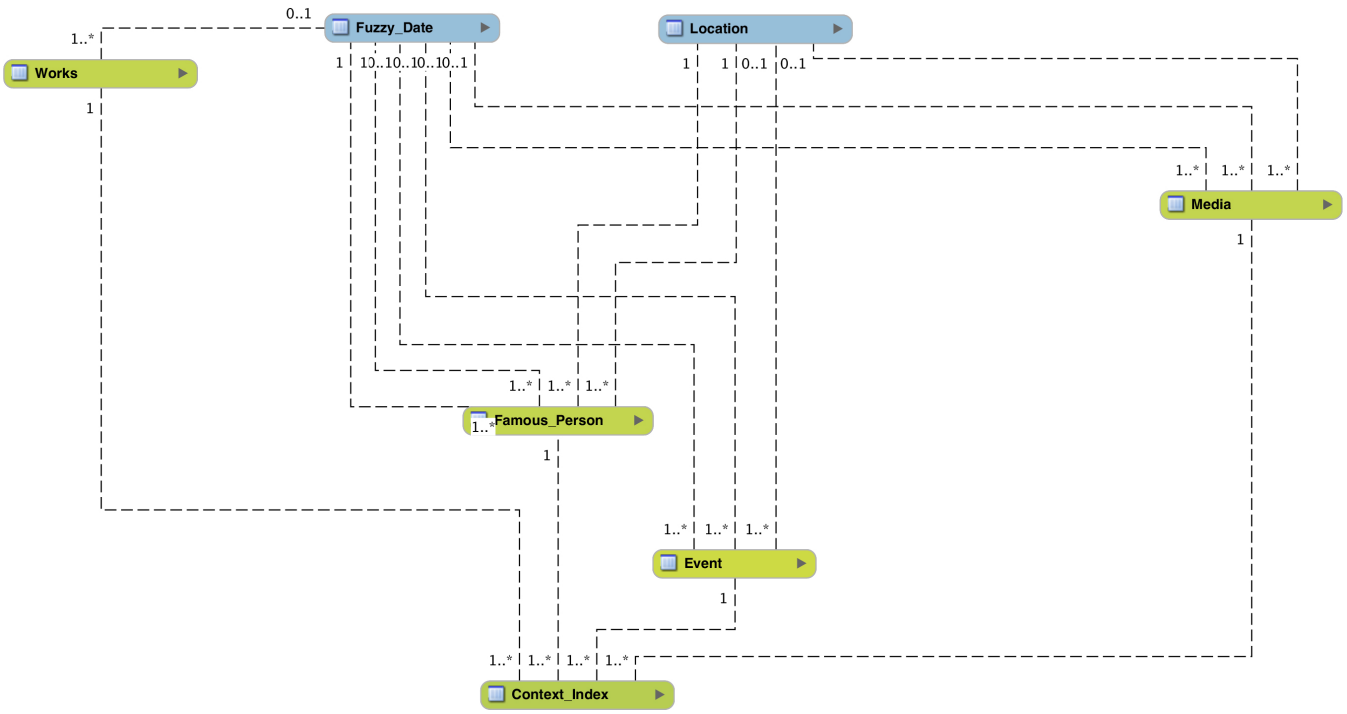

Figure 3: Architettura della Knowledge Base

ottenere questo risultato le possibilità di approccio principali si possono ricondurre a tre:

1. Sfruttare i linked data per recuperare a runtime le informazioni necessarie: questo è una scelta ambiziosa, che pone molta fiducia nel fatto di riuscire a scavare nel modo giusto tra i datasets per trovare abbastanza elementi da restituire, elementi che siano anche inerenti alla richiesta fatta.

2. Utilizzare i linked data, collegandovi un altro dataset, formato da informazioni raccolte da noi: il problema che vogliamo risolvere riguarda la storia delle persone, e chi meglio delle altre persone può aiutarci a raggiungere questo obiettivo? per questo un grosso aiuto nella ricerca di immagini, eventi e altro potrebbe venire da un opera di crowdsourcing che, immagazzinata sotto forma di entità con un significato e, collegata ad altre sorgenti di dati, andrebbe a completare i nostri bisogni e forse quelli di altri utilizzatori del semantic web.

Quest'ultima è la proposta di soluzione più interessante, che si tuffa nel futuro di una rete nella quale informazioni da sorgenti completamente diverse tra di loro vanno a completarsi e impreziosirsi a vicenda, ma che per la natura sperimentale del nostro lavoro va purtroppo considerata solo un possibile sviluppo futuro .

3. Per poter essere meno dipendenti dalle specifiche RDF e SPARQL, certamente ancora in evoluzione, l'approccio che abbiamo scelto è di per sé il più semplice: perchè non costruire un archivio storico in forma di database, indicizzarlo in maniera corretta e renderlo disponibile tramite un algoritmo di creazione di un contesto intorno a delle coordinate spazio-temporali?

Così nasce Reminiscens, il cui funzionamento e struttura sono spiegati qui sotto.

4.1 Architettura generale

Reminiscens è un sistema strutturato di varie parti, che lavorano insieme per fornire numerosi servizi. Le funzionalità di raccolta e memorizzazione dati sono svolte da un database, da moduli software e da un'interfaccia grafica. Il database è la componente fondamentale di Reminiscens, contenente tutto il materiale utilizzato dal sistema per restituire un contesto (di cui si parlerà più avanti) all'utente. Perché la Knowledge Base contenga dei dati, questi devono essere raccolti da qualche parte; nel nostro caso il lavoro è fatto da moduli ETL, ognuno dei quali si occupa di interrogare una determinata risorsa sul web (e.g. DBpedia). La stessa funzione è svolta da CrowdMemories, una UI sviluppata da Francesco Maturi che utilizza il crowdsourcing per reperire materiale storico direttamente dagli utenti. L'accesso ai dati invece, è implementato tramite delle API con architettura REST, che svolgono quindi la funzione di intermediari nella comunicazione tra i client e la Knowledge Base. La parte di visualizzazione consiste attualmente in una semplice interfaccia in forma di libro sfogliabile, a cui nel futuro si affiancheranno altri servizi; questi ulteriori sviluppi verranno trattati alla sezione 6.1.

4.2 Il mio lavoro

Nella costruzione del lavoro, mi sono trovato a seguire i dati per tutto il percorso, dalla loro raccolta alla loro visualizza-

```
SELECT DISTINCT *
        WHERE {
                ?event a <http://dbpedia.org/ontology/SportsEvent> .
                ?event <http://www.w3.org/2000/01/rdf-schema#label> ?title .
                ?event <http://dbpedia.org/ontology/abstract> ?descr .
                ?event <http://dbpedia.org/ontology/date> ?date .
                ?event <http://dbpedia.org/ontology/city> ?city.
                FILTER (lang(?title)='it' && lang(?descr)='it')
        }

SELECT *
        WHERE {
                ?work a <http://dbpedia.org/ontology/Song> .
                ?work <http://www.w3.org/2000/01/rdf-schema#label> ?title .
                ?work <http://dbpedia.org/ontology/abstract> ?descr .
                ?work <http://dbpedia.org/ontology/artist> ?author .
                ?work <http://dbpedia.org/ontology/genre> ?genre .
                {
                        ?work <http://dbpedia.org/ontology/releaseDate> ?relDate .
                } UNION {
                        ?work <http://dbpedia.org/property/released> ?relDate_1
                } FILTER (lang(?title)='it' && lang(?descr)='it')
        }
```

Figure 4: Alcune query SPARQL

zione, passando per tutta la fase di elaborazione. Di seguito ecco le parti sviluppate.

4.2.1 Moduli ETL

L'implementazione da noi scelta è quella di un insieme di moduli ETL (Extract, Transform, Load) standalone che raccolgono ed elaborano dati, dipendenti quasi solo da un modulo che si occupa della comunicazione diretta con la Knowledge Base. La loro importanza è vitale perchè, oltre all'ovvia funzione di recuperare gli elementi che andranno poi restituiti all'utente in forma di contesto, hanno anche il compito di ricostruire dove possibile le coordinate spazio-temporali dei dati estratti (e.g. tramite il Geocoding di una locazione scritta in forma testuale). Essendo la lista delle risorse consultabili sul web sempre in crescendo, questa scelta è stata fatta per rendere semplice l'aggiunta e la sostituzione dei moduli, che attualmente supportano la consultazione di Flickr, DBpedia, Catinabib (catalogo di cartoline e incisioni a tema regionale di proprietà della biblioteca comunale di Trento) e di dataset del progetto OpenData Trentino. Proprio grazie a quest'ultima risorsa, si spera che con il passare del tempo, grazie all'aggiunta di nuovi datasets, le informazioni contenute nella Knowledge Base e di conseguenza fornite all'utente possano crescere di quantità e di qualità.

4.2.2 Indice

Onde consentire una lettura veloce dei dati raccolti, il risultato delle estrazioni dal web viene periodicamente indicizzato, così che non sia necessario leggere tutto il db per ottenere un risultato da spedire ai client visuali.

4.2.3 Api REST

er rendere possibile la consultazione della KB senza dover interrogarla direttamente, ho sviluppato delle semplici api che, tramite messaggi HTTP, permettono di fare ricerche a partire da parametri spazio-temporali. Queste si dividono in due tipi:

- Nel caso sia necessario semplicemente leggere una lista di elementi dal KB, un insieme di chiamate si assicura che al richiedente torni la giusta lista di tutte le immagini o gli eventi o altro combacianti i parametri di input.
- Se invece si tratta della comunicazione con i client, altre chiamate vanno a leggere l'indice per restituire un contesto. Questo argomento è descritto in maniera specifica alla sezione 4.2.5.

Per questioni più tecniche, nell'appendice B è disponibile la documentazione delle api.

4.2.4 Simple CRUD

Affiancato ai moduli ETL e alle api è presente un piccolo e semplice sistema con interfaccia minima che effettua CRUD, per aggiungere e modificare manualmente il materiale del db.

4.2.5 Booklet

Un'ultima parte riguarda il lavoro per comporre un'interfaccia che potesse essere utile a visualizzare i dati restituiti dalle API, indipendente dai client di Reminiscens: chiedendo all'utente una decade e un luogo, quello che viene mostrato è un libricino sfogliabile tramite browser, contenente un contesto completo in una forma che possa essere familiare a chi

Figure 5: Una pagina del Booklet, ottenuto con query 1990 - Trento

lo guarda, soprattutto nel caso dei meno giovani, che mancano di dimestichezza con i mezzi più tecnologici. Il libro restituito è però un insieme di dati impersonali in quanto, per avere un risultato personalizzato, la scelta giusta è il vero e proprio client di Reminiscens. Il caso d'uso del Booklet come applicazione per il testing è descritto nell'appendice A, mentre per una prova pratica è disponibile online una versione della web app[9].

4.2.6 *Algoritmo LifeContext*

Come definito sopra, il contesto è un insieme di immagini, eventi, personaggi e musica che hanno a che fare con il tempo e il luogo di cui l'utente potrebbe voler scoprire qualcosa, e che potrebbe aiutarlo a raccontare della sua vita. La nostra scelta è stata quella di comporre questo contesto utilizzando cinque entità per tipo, selezionate secondo alcuni criteri all'interno dell'indice. Partendo dalla knowledge base, la Lifeincontext API crea un contesto intorno ad una lista di eventi. L'elaborazione si divide due casi principali, all'interno dei quali si snodano diversi sottocasi:

- se le coppie lat-lon possono essere racchiuse all'interno di un cerchio virtuale di raggio 50 km:
  - se la maggior parte delle decadi è uguale, viene creato un unico contesto a partire da una delle coppie lat-lon e da quella decade. questo si può ottenere con una singola query che, ordinando i risultati per pertinenza, seleziona direttamente i primi cinque;
  - se il numero degli eventi di input è minore di tre, viene creato un contesto componendo dei "sotto-contesti", uno per ogni coppia formata dalle decadi di input e da una delle coppie lat-lon;
  - l'ultimo caso è un'estensione del secondo, infatti, a partire da una lunga lista di coppie, viene creato un contesto componendo tre sotto-contesti tra quelli possibili; questo viene fatto per non inquinare troppo il risultato, che senza il filtraggio appena indicato sarebbe una semplice accozzaglia di elementi che in ultima analisi sarebbero poco correlati tra di loro;
- se le coppie non soddisfano la condizione, vengono riproposti il secondo e il terzo sotto-caso indicati sopra, con la differenza che ogni contesto viene computato a partire dalle coppie così come indicate in input, senza elaborazioni addizionali.

Oltre al calcolo a partire dalle sole indicazioni spazio-temporali, un ulteriore versione dell'algoritmo è stata pensata per cercare, filtrare e personalizzare i risultati ottenuti anche in base a delle parole chiave, siano esse argomenti, generi musicali, interessi dell'utente e oggetto della risorsa. Ad esempio, una richiesta riguardante Trento nel decennio 1980, con keyword "calcio", dovrebbe restituire tra le altre una voce sul campionato di calcio serie C2 1984-85, quando la squadra locale venne promossa in Serie C1 dopo uno spareggio vinto contro l'Ospitaletto. Grazie a queste poche informazioni, fornite al sistema dall'utente in fase di configurazione del profilo, l'API dovrebbe consegnare del materiale interessante ed evocativo di momenti del suo passato, stimolando così il racconto di episodi della sua vita.

[9] `http://test.reminiscens.me/lifecontext/booklet/ui/`

4.3 La Knowledge Base

Certamente buona parte della riuscita di questo lavoro è dipesa dalla scelta di modellazione per la Knowledge Base di Reminiscens, per questo abbiamo puntato su uno schema semplice e facilmente modificabile: ogni tipologia di entità (eventi, immagini, lavori artistici come musica e libri, personaggi famosi) è mappata su una singola tabella, e possiede una relazione con altre due tabelle che contengono i dati riguardanti luogo e date associati a ogni elemento, e una con la tabella che funge da indice. Figura 3 mostra in breve l'architettura del nostro database.

Durante la raccolta delle risorse con i moduli ETL, ci siamo resi conto che le coordinate spazio-temporali erano in forme molto diverse, e contenevano dati con precisione diversa: per questo abbiamo costruito la Knowledge Base con i concetti di *Fuzzy Date* e *Fuzzy Location*, cioè date e luoghi che possono essere più o meno definiti (e.g. una data contiene l'anno ma non il giorno, mentre un'altra contiene solo la decade), classificandole tramite due scale di valutazione. Non ci siamo preoccupati troppo di ottenere solo entità con informazioni precise per due semplici motivi: (1) avremmo ovviamente dovuto scartarne molte, e (2) passando tramite una fase preliminaria di pulizia automatica - durante la raccolta - e un'altra di pulizia manuale, riusciremo a rendere consistenti e semplici da cercare gran parte dei contenuti.

4.4 Framework e librerie

Per realizzare queste componenti, mi sono avvalso di diverse tecnologie, elencate qui sotto per sezione.

- Moduli ETL e Indice: Tutti i moduli sono realizzati in Java utilizzando Apache HttpComponents per gli scambi di messaggi HTTP, Hibernate, Apache Jena per poter interrogare tramite SPARQL, le API di Google Maps per risolvere tutti i problemi relativi alla risoluzione sia di coordinate in luoghi che il contrario, Google Gson per il parsing di messaggi JSON e le API di Youtube per associare un video musicale a ogni canzone nella KB. Riguardo SPARQL, in figura 4 sono mostrate due semplici query utilizzate dal sistema per interrogare DBpedia e ottenere rispettivamente canzoni ed eventi sportivi con dati in lingua italiana.

- Api REST: la scrittura di queste librerie mi è stata facilitata dall'utilizzo di Ruby unito al framework Sinatra, con l'ausilio di ActiveRecord e del linguaggio di templating RABL. Al fine di eseguire alcuni calcoli complessi, ho eseguito il porting di uno script per la conversione di coordinate geografiche in UTM[10] e ho adattato un implementazione Ruby dell'algoritmo con complessità lineare elaborato da Megiddo per il calcolo del Minimum Enclosing Circle di un set di punti[11].

- CRUD: Questa è un'applicazione Ruby on Rails che si avvale di una semplice UI per la modifica manuale dei dati, ma che permette eseguire le stesse operazioni programmaticamente, inviando e ricevendo JSON.

[10] `http://home.hiwaay.net/~taylorc/toolbox/geography/geoutm.html`
[11] `http://www.dseifert.net/code/mec/index.html`

Tipo	Numero
Immagini	548
Canzoni	1587
Eventi	630
Persone famose	479

Table 1: Tipi raccolti

Sorgente	Numero
Flickr	340
DBpedia	1168
Catinabib	208
OpenData Trentino	463
Crowd e altro	1065

Table 2: Distribuzione delle sorgenti

- Booklet: L'interfaccia del booklet si basa su "20 things I learned about browsers and the web"[12], applicazione web HTML5 (sviluppata dal team di Google Chrome e resa open source) riadattata per le nostre necessità.

4.5 Stato della Knowledge Base

Per questa prima parte di sviluppo, anche pensando a possibili fasi di testing, abbiamo voluto concentrarci sulla raccolta di materiale relativo alla zona del Trentino, con maggiore attenzione alle zone di Trento, Rovereto e Pergine Valsugana; nella realtà dell'implementazione, questo significa che, dove possibile, i dati sono stati filtrati per luogo, con alcune eccezioni: per quanto riguarda la musica ad esempio, abbiamo fatto la scelta di cercare anche musica straniera, data la facilità con cui un abitante locale può venire a contatto con canzoni che non provengono dall'Italia. Al tempo della scrittura di questo documento, la Knowledge Base di Reminiscens contiene 3244 entità, tra immagini, canzoni, eventi e personaggi famosi. Tabella 1 mostra quante di queste risorse appartengono a ogni tipo, mentre tabella 2 mostra la provenienza delle stesse. Entrambe le tabelle mostrano come, nonostante il lavoro da noi compiuto sia ancora in uno stato di sviluppo, (1) sia stato molto più semplice raccogliere elementi di tipo musicale (un piccolo progetto che ha lavorato con la musica come terapia è illustrato brevemente nella sezione 5.1), (2) Il web semantico e il crowd si siano dimostrate due fonti di valore inestimabile per gli scopi che ci siamo posti, soprattutto considerando il fatto di averle sfruttate in maniera non intensiva; i dati qui mostrati sono solo uno degli input che è necessario analizzare approfonditamente per scegliere quale strada tracciare per il futuro di Reminiscens.

5. LAVORI CORRELATI

Reminiscens non è di certo il primo lavoro compiuto con l'IT al servizio della reminiscenza; discuteremo brevemente alcuni approcci qui di seguito.

5.1 Stimolare la reminiscenza

Pensieve[13] è un sistema ideato per inviare a intervalli regolari degli stimoli a ricordare. Tali stimoli possono essere interattivi (come una domanda, e.g. "Ti ricordi quando hai imparato a cucinare?"), oppure semplicemente visuali: in questo ultimo caso, è richiesto un collegamento dell'account Pensieve con importanti e popolari servizi, come Picasa, Flickr e Twitter, dai quali Pensieve provvede a estrarre le immagini e i post che verranno utilizzati come contenuto per inviare i trigger all'utente. Tra i risultati del progetto,

[12] `http://www.20thingsilearned.com/`
[13] `http://pensieve.cornellhci.org/`

è stato notato che le persone preferiscono un tipo di reminiscenza di tipo sociale, ma è conseguibile in maniera migliore tramite un'interazione faccia a faccia; questo e gli altri risultati di Pensieve sono descritti in [4].

Simile a questo possiamo trovare anche il social network Proust[14], che è basato solo su quesiti riguardanti la vita di un individuo e sulle risposte che lo stesso dà. Le risposte, arricchite con informazioni su luogo e tempo, vengono utilizzate per comporre la storia dell'utente, e visualizzate in forma di libro da mostrare agli altri utenti. Proprio la condivisione di queste storie, unita alla possibilità che determinati utenti hanno di chiedere al diretto interessato qualcosa sul suo passato, costituisce il punto di forza di Proust.

Un progetto molto interessante, che si discosta da quelli appena presentati per il suo non essere un sistema informatico, è Alive Inside[15], che presenta gli effetti per così dire miracolosi, su alcuni anziani malati di Alzheimer e affetti da demenza senile, della musica da loro ascoltata in gioventù. Il risultato è un documentario d'effetto che mostra come essi vengano "svegliati" dalla musica, che fa riaffiorare in un lampo ricordi del passato e li rende in grado di rispondere a domande, in un modo che fino a pochi istanti prima sembrava impossibile: l'obiettivo del film è infatti sensibilizzare le case di cura e gli ospedali, ma anche le famiglie, all'uso di questa terapia semplice ed economica, quanto efficace.

5.2 Raccolta dei dati

Numerosi sono stati i tentativi di costruire una banca dati culturale, e un esempio è Memoro[16], piattaforma tutta italiana ma ormai estesa e localizzata in molte regioni del mondo che, prendendo spunto dalla pratica del racconto delle proprie esperienze, tipica di genitori e nonni nei confronti di figli e nipoti, si propone come un contenitore di clip audio e video raccolti dagli utenti, clip che contengono la memoria di vite vissute secondo usanze e valori di un'altra epoca, rendendola disponibile a chiunque abbia 10 minuti di tempo da spendere.

Una menzione speciale va a Live Memories[17], progetto locale coordinato da FBK, Università di Trento e Università di Southampton e orientato alla costruzione di un archivio multimediale eterogeneo, raccogliendo dati da sorgenti molto diverse tra loro. Il progetto, che ha avuto luogo dal 2008 al 2011, si è concluso dopo aver coinvolto, oltre agli enti accademici, anche il quotidiano l'Adige, la rivista Vita Trentina e il consiglio comunale di Trento.

5.3 Visualizzazione dei dati

I due modi più naturali di rappresentare entità caratterizzate da coordinate spazio-temporali sono (1) tramite punti su una mappa e (2) con l'utilizzo di timeline; il secondo è l'approccio scelto da Project Greenwich[18], realizzato da Microsoft Research Cambridge. Tramite il login a Facebook, permette di creare delle timeline con le proprie fotografie, arricchirle collegandoci delle pagine da Wikipedia, confrontarle con le timeline create da altri utenti e condividerle con famiglia e amici. Il sistema è stato studiato per permettere agli utenti di riflettere sul loro passato, magari guadagnando consapevolezza, fornendo anche gli strumenti per organizzare i contenuti in maniera creativa ed espressiva.

6. CONCLUSIONI

Durante la progettazione del lavoro sono sorti alcuni problemi, dettati soprattutto da necessità pratiche: (1) nonostante la quantità di informazione reperibile sul web sia immensa, il fatto di lavorare per un target che è di lingua italiana restringe enormemente il campo, tanto che molte risorse nei dataset non sono state utilizzate; ciò costringe ad impiegarsi in maniera estremamente maggiore nell'acquisizione di molti più dataset, finendo ovviamente per complicare tutto il sistema. (2) Anche le informazioni disponibili in forma strutturata non sono esenti da difetti: DBpedia, nonostante sia una risorsa utilissima, soffre di un problema che ha Wikipedia in primo luogo. La questione proviene dalle libertà che gli editori hanno nella scrittura dei template per le infobox, e risulta quindi in attributi che, pur essendo semanticamente equivalenti, sono proposti con nomi diversi e possono quindi sfuggire alla progettazione di un sistema di raccolta dati automatico. (3) SPARQL ha il pregio di essere un linguaggio di interrogazione estremamente potente, ma questo consegue naturalmente in una difficoltà nel suo apprendimento da parte dell'essere umano; siamo sicuri che nel prossimo futuro, migliorando le query, riusciremo a ottenere più dati, e soprattutto dati migliori.

Merita una riflessione a parte il discorso sulle categorie di risorse per stimolare la reminiscenza: cosa può essere aggiunto e cosa migliorato? Come già spiegato, attualmente Reminiscens è progettato per lavorare su eventi, immagini, canzoni e personaggi famosi, ma non è detto che questa sia la scelta giusta, e molti altri possono essere i tipi utili a favorire l'afflusso di ricordi; *Pensieve: Supporting Everyday Reminiscence* contiene ad esempio un'interessante categorizzazione di memory triggers[5].

6.1 Lavoro futuro

La natura di Reminiscens è quella di un sistema vasto e in continua evoluzione; molto è quindi il lavoro ancora da fare, per aggiungere funzionalità e migliorare quelle esistenti. Il primo passo è quello di portare a compimento i client di Reminiscens, permettendo ai primi utenti di avvicinarsi alla piattaforma, aggiungendo anche un modulo di confronto tra timeline allo scopo di offrire una componente social a gruppi di persone che hanno caratteristiche, gusti e esperienze di vita simili. Ispirati da altri sistemi che si occupano della reminiscenza, verranno aggiunti anche dei trigger in forma di domande poste all'utente, che andranno a diventare una sorta di cornice alle risorse mostrate. Vi è connesso l'obiettivo di dare la possibilità agli utilizzatori di personalizzare la loro esperienza, caricando del materiale privato in un loro spazio ed editando i booklet per adattarli a specifiche esigenze; nel caso di utenti anziani, ci aspettiamo che operazioni di questo tipo vengano portate a termine con naturalezza con l'aiuto dei loro cari.

Relativamente al backend, un importante step da compiere è il raffinamento dei moduli ETL: seguendo l'esempio di Ti-

[14] http://www.proust.com/
[15] http://www.ximotionmedia.com/
[16] http://www.memoro.org/
[17] http://www.livememories.org/
[18] http://projectgreenwich.research.microsoft.com/

meTrails, numerosi dati spazio-temporali potrebbero essere ricavati dal testo semplice e dagli attributi scritti in forma estesa e normalizzati prima del salvataggio nella Knowledge Base, utilizzando ad esempio il tagger temporale HeidelTime e il Geotagger di MetaCarta. A questi seguiranno ovviamente nuove versioni dell'algoritmo LifeContext, forse dettate da particolari esigenze emerse dalla raccolta di nuovi dati. Proprio riguardo quest'ultima, c'è da discutere sulla rimozione dei duplicati: se più avanti molti dataset verranno aggunti, si profila il rischio di raccogliere delle entità duplicate. Questo problema, senza escludere completamente la possibilità di un intervento manuale, potrebbe essere risolvibile tramite il calcolo della "firma" della risorsa che, considerata la varietà di forme nelle quali si possono trovare i dati sul web, deve essere una pensata come una funzione di informazioni "immutabili". Come già accennato, l'intervento manuale è da trattare con grande importanza: perchè l'esperienza offerta all'utente sia migliore possibile, vorremmo infatti predisporre una piattaforma pensata per un gruppo di possibili "esperti", da utilizzare per effettuare pulizia e arricchimento dei dati, da affiancare magari a CrowdMemories.

7. RICONOSCIMENTI

Ringrazio il mio relatore Prof. Fabio Casati, il correlatore Cristhian Parra e LifeParticipation[19] per avermi seguito durante tutto il percorso dell'internship che ha portato a questa tesi e al lavoro da me fatto. Ringraziamenti estesi a Manuel Cattin Cosso e chi mi ha sopportato nei momenti di stress di quest'anno.

8. REFERENCES

[1] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. Dbpedia-a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1):107–117, 1998.

[3] P. G. Coleman. Measuring reminiscence characteristics from conversation as adaptive features of old age. *The International Journal of Aging and Human Development*, 5(3):281–294, 1974.

[4] D. Cosley, V. S. Sosik, J. Schultz, S. T. Peesapati, and S. Lee. Experiences with designing tools for everyday reminiscing. *Human–Computer Interaction*, 27(1-2):175–198, 2012.

[5] S. T. Peesapati, V. Schwanda, J. Schultz, M. Lepage, S.-y. Jeong, and D. Cosley. Pensieve: supporting everyday reminiscence. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2027–2036. ACM, 2010.

[6] J. Strötgen and M. Gertz. Timetrails: a system for exploring spatio-temporal information in documents. *Proceedings of the VLDB Endowment*, 3(1-2):1569–1572, 2010.

APPENDIX

A. IL BOOKLET COME APPLICAZIONE PER IL TESTING

Nella sezione dedicata alla soluzione è stato presentato brevemente il Booklet, un'interfaccia pensata per svolgere una delle funzioni di base di Reminiscens, cioè quella di mostrare contenuti che possano risultare familiari all'utente in modo da stimolare l'afflusso di ricordi e far partire la narrazione di episodi correlati al contenuto presentato. Tale applicazione è stata intesa per permettere, come sviluppo futuro, alle famiglie degli anziani la personalizzazione del booklet, aggiungendo ad esempio immagini e musica di propria scelta; il tutto per riuscire a creare un "aggregatore di ricordi" che sia il più efficace possibile. Come anticipato in sezione 4.2.4, la scelta da noi compiuta è quella di offrire un'interfaccia che riprende un libro antico, con una grossa copertina in pelle e il titolo dorato, il tutto con uno sfondo che rappresenta la superficie in legno massiccio di un vecchio tavolo, in modo da mostrare un ambiente familiare che possa essere il primo passo per andare oltre il gap che indubbiamente esiste tra gli anziani e le nuove tecnologie; proprio per come è stata pensata, la UI si presta perfettamente ad un'attività preliminare di testing. Per verificare la bontà delle scelte di design, così come l'adeguatezza di alcuni dati raccolti dai moduli ETL, si potrebbe pensare ad un workshop avvalendosi della collaborazione di un centro ricreativo per anziani, strutturato in questo modo: dopo aver diviso i collaboratori in gruppi, ad ogni gruppo verrebbe assegnato un iPad con a bordo una versione del booklet con contenuti diversi; l'idea è quella di comporre ogni piattaforma di test con contenuti riguardanti un diverso periodo della vita, tenendo conto delle diverse età dei partecipanti e ottenendo quindi infanzia, giovinezza, età adulta e presente. Ultimo parametro da considerare nella divisione dei gruppi e nella composizione dei booklet è quello del luogo in cui centrare il calcolo del contesto da presentare ai volontari: perchè il workshop possa restituire dei risultati utili, c'è bisogno che ognuno possa vedere immagini, ascoltare musica e ricordare eventi che almeno in teoria possano essere significative (e.g. una persona cresciuta a Roma difficilmente potrà rievocare memorie passate guardando delle fotografie del Monte Bondone). Il feedback ricevuto da questa e altre prove è indispensabile per capire se la strada che stiamo percorrendo con Reminiscens è quella giusta e, in caso di difetti del nostro approccio, fornirebbe importanti linee guida per correggerlo.

B. DOCUMENTAZIONE API

A seguito del lavoro compiuto, viene qui rilasciata la documentazione delle API REST che si occupano di costruire un contesto a partire dalle entità contenute nella Knowledge Base. Oltre a questo, sono state scritte delle route che permettono anche la semplice lettura dei dati dalla KB, utilizzando delle query più complesse; l'obiettivo è quello di renderle disponibili al pubblico, sperando che possano essere utili. Ogni route è raggiungibile a partire dall'host `http://test.reminiscens.me/lifecontext/api`; la documentazione completa è anche disponibile sulla pagina apiary.io dedicata[20].

B.1 LifeContext API

- **GET** /v2/generalBooklet/media
- **GET** /v2/generalBooklet/events
- **GET** /v2/generalBooklet/works

[19] `http://www.lifeparticipation.org/`

[20] `http://docs.lifecontext.apiary.io/`

Le prime due route accettano i seguenti parametri:

- **decade[]** : un array di decadi nella forma *dddd*. Ogni decade deve avere nome *decade[]* - invece di *decade* - per far sapere all'API che si tratta di un array.
- **lat[]**, **lon[]** : due array di coordinate geografiche, Latitudini e longitudini devono avere nome rispettivamente *lat[]* e *lon[]* - invece di *lat* e *lon* - per far sapere all'API che si tratta di due array.

mentre la terza accetta solo il primo, cioè *decade[]*.

B.2 Lettura dei dati

- **GET** /media
- **GET** /events
- **GET** /people
- **GET** /works

Le prime tre route accettano i seguenti parametri:

- **decade**: una decade nella forma *dddd*.
- **place** : è la forma letterale rappresentante il luogo; quando usato, è cercato nel db; se assente, viene effettuato il Geocoding per ottenere le coordinate corrispondenti; può essere associato a *radius*; se usato insieme a *lat* e *lon*, viene ignorato.
- **lat** e **lon** : ovviamente latitudine e longitudine; devono sempre apparire in coppia; possono essere associate a *radius*; se utilizzate insieme a *place*, quest'ultimo viene ignorato.
- **radius** : rappresenta la distanza nella da *place* / *lat* e *lon* che l'API deve utilizzare per considerare validi o no i possibili risultati; di default ha valore 0.

mentre l'ultima accetta solo il primo, cioè *decade*.