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

[Extended Abstract] *

Nicola Parrello†
Institute for Clarity in Documentation
1932 Wallamaloo Lane
Wallamaloo, New Zealand
trovato@corporation.com

Wally Will‡
Institute for Clarity in Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
webmaster@marysville-ohio.com

Carmen Sandiego§
The Thørväld Group
1 Thørväld Circle
Hekla, Iceland
larst@affiliation.org

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 senilitÃă e utili nella trasmissione ai posteri di quel che era. Proprio l'utilitÃă di questo fenomeno spontaneo, chiamato reminiscenza, fa sorgere una domanda: come ÃĺÍ 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.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms

Theory

Keywords

ACM proceedings, LaTeX, text tagging

*A full version of this paper is available as *Author's Guide to Preparing ACM SIG Proceedings Using LaTeX2ε and BibTeX* at `www.acm.org/eaddress.htm`

†Dr. Trovato insisted his name be first.

‡The secretary disavows any knowledge of this author's actions.

§This author is the one who did all the really hard work.

1. INTRODUZIONE

EâĂŹ successo a tutti, almeno una volta, di ritrovarsi a raccontare a qualcuno (amico, familiare o altro), episodi della propria vita passata: lâĂŹatto di raccogliere dalla memoria esperienze passate per condividerle con qualcuno, rendendo significativo il rapporto trai due attori della conversazione, ÃĺÍ chiamato reminiscenza. Se questo fenomeno ÃĺÍ importante durante buona parte della vita, assume rilevanza massima per gli anziani; ÃĺÍ infatti studiato anche come aiuto per i malati di Alzheimer, in quanto il ricordare diventa come unâĂŹisola nella quale la persona puÃš trovare degli appigli. Ma non ÃĺÍ solo questo: la reminiscenza, in quanto interazione con un altro soggetto, puÃš essere utile anche per combattere lâĂŹisolamento in cui gli anziani spesso si trovano, favorendo allo stesso tempo un rapporto faccia a faccia, positivo per entrambi gli attori. Indicata la positivitÃă del fenomeno, ÃĺÍ doveroso fare i conti con il fatto che nella maggior parte dei casi nasce in maniera spontanea, dando vita a un quesito importante: come trovare gli stimoli giusti per stimolare la reminiscenza? Crediamo che la fonte piÃź adatta a fornire lâĂŹinput adatto a farla scattare sia quello in cui lavoriamo piÃź o meno tutti i giorni, cioÃĺÍ il web. Dopo piÃź di ventâĂŹanni di internet, la nostra storica collettiva ÃĺÍ quasi tutta online: se oltre a questo consideriamo lâĂŹespansione dei dataset disponibili in forma strutturata, ÃĺÍ chiaro che raccogliere da lÃň materiale audiovisivo, eventi e personaggi storici puÃš fornire delle importanti fondamenta da cui partire. Ma possedere questi dati non ÃĺÍ abbastanza: perchÃĺÍ gli stimoli siano effettivamente significativi, ÃĺÍ necessario ideare un meccanismo atto a identificare quale parte dei contenuti raccolti dal web ÃĺÍ 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, ÃĺÍ importante iniziare a calarsi nel concreto, rendendosi conto a quali campi ispirarsi per costruire unâĂŹarchitettura funzionale alla soluzione della questione.

Web semantico: con questo termine, coniato dal suo ideatore Tim Berners-Lee, âĂIJsi intende la trasformazione del World Wide Web in un ambiente dove i documenti pubblicati (pagine HTML, file, immagini, e cosÃň via) sono associati ad informazioni e dati che ne specificano il contesto semantico in un formato adatto all'interrogazione e l'interpretazione (es. tramite motori di ricerca) e, piÃź in generale, all'elaborazione automatica.âĂİ [Wikipedia] Questo non significa limitare le numerosissime forme in cui Ãĺpossibile presentare i contenuti sul web, bensÃň Ãĺun 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 facilitÃă con la quale Ãĺpossibile estrarre informazioni e adattarle alle proprie necessitÃă; sono quindi queste la scelta piÃź ovvia come semplice punto di partenza per la ricerca di contenuti atti a stimolare la reminiscenza. Una delle realtÃă piÃź grosse che opera nel web semantico Ãí DBpedia, il cui approccio Ãíquello di parsare periodicamente i 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 Ãídefinita tramite un URI, un identificatore unico dellâĂŹentitÃă presentata. Il lavoro di DBpedia Ãíeccellente, ma Ãíinvolontariamente 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 Ãíinterrogabile tramite SPARQL, un linguaggio SQL-like costruito per leggere RDF. Dopo il rilascio del primo dataset al pubblico nel 2007, la situazione attuale Ãíquella di un grosso grafo di altri set di dati collegati tra di loro tramite lâĂŹaccoppiamento di risorse appartenenti a insiemi diversi, ma rappresentanti la stessa entitÃă. Pubblicare questi Linked Data rende molto piÃź facile la ricerca di informazioni sul web, rendendole molto piÃź precise e complete. Alcuni altri progetti di web semantico sono: Freebase, che a differenza di DBpedia Ãíun progetto proprietario e orientato al profitto; 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. A settembre 2011, lo stato dei collegamenti tra datasets semantici Ãírappresentato dal grafo visualizzabile a (http://lod-cloud.net/versions/2011-09-19/lod-cloud.html). Un approccio diverso nellâĂŹattribuzione di un significato ai contenuti presenti sul web Ãíquello dei microformati, unâĂŹestensione di markup che, tramite lâĂŹutilizzo degli attributi HTML class, rel e rev, consente lâĂŹattribuzione di regole semantiche a normali pagine web: date le informazioni sul contatto di Nicola Parrello

```
<div>
    <div>Nicola Parrello</div>
    <div>University of Trento</div>
    <div>123-123123</div>
    <a href="mailto:hi@everybody.com">Email</a>
</div>
```

queste, se riscritte con il microformato hCard (specifico per i contatti), diventano

```
<div class="vcard">
    <div class="fn">Nicola Parrello</div>
    <div class="org">University of Trento</div>
    <div class="tel">123-123123</div>
    <a class="email" href="mailto:hi@everybody.com">Em
</div>
```

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

Web search e contestualizzazione dellâĂŹinformazione: Il passo successivo nella ricerca di una soluzione ad problema che Ãítema di questo documento, dopo aver scelto da dove reperire i dati, Ãíquello di decidere come permettere a un eventuale utente di accedere ai contenuti che vogliamo proporre. LâĂŹapproccio che sembra piÃź naturale Ãíquello 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 Ãíuna buona idea quella di indicizzare il materiale ricercabile. In questo campo Ãídifficile non considerare Google come lo stato dellâĂŹarte, essendo loro il search engine piÃź utilizzato al mondo. PerchÃl' 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 puÃš essere riassunto cosÃň: Il Server URL parte da un URL e, leggendo il Document Index, invia gli URL ai Crawler. I Crawler scaricano le pagine web e le mandano nello Store Server. Lo Store Server comprime le pagine nel formato zlib (RFC1950), riducendo la loro dimensione a un terzo dellâĂŹoriginale, che vengono poi di un docId univoco e immagazzinate nel repository. Il repository viene letto dallâĂŹIndexer, 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, lâĂŹIndexer 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. Lo URL Resolver combina i documenti con i dati dei link, per costruire tabelle di coppie di docId, utilizzati per calcolare il PageRank. Il Sorter riordina lâĂŹindice (ordinato per docId) in un indice inverso (ordinato per wordId), aggiungendo altri dati per rendere la ricerca piÃź precisa. Infine viene calcolato il PageRank, una misura dellâĂŹimportanza di una pagina web calcolata in base a quali e quante sono le pagine che hanno dei link che puntano a essa.

Negli anni Google Ãídiventato colmo di pubblicitÃă, mostrata e scelta in base ai dati raccolti dalle ricerche degli utenti, ma certamente non ÃílâĂŹunica possibilitÃă in quanto a motori di ricerca: un esempio su tutti e quello di Duck Duck Go che, a differenza di Google, Ãíun Semantic Search Engine. Il sistema quindi si occupa di valutare lâĂŹeffettivo significato

dei termini di ricerca, eliminando in maniera piÃź precisa i risultati irrilevanti. Ma quello che lo contraddistingue in maniera maggiore dagli altri ÃÍ 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. (https://duckduckgo.com/privacy) Quello che puÃš infastidire gli utenti, cioÃÍ gli annunci di cui sopra, sono perÃš molto interessanti per la nostra ricerca, in quanto introducono un altro aspetto molto importante: perchÃÍ la ricerca sia semplice da utilizzare, e soprattutto piacevole, puÃš essere necessario renderla per cosÃň dire automatica, nascondendo lâĂŹazione manuale dellâĂŹinserimento di parametri e mostrando i risultati direttamente come contesto di qualche altra azione. Premiata nel 2012 dal magazine Popular Science come innovazione dellâĂŹanno (http://www.popsci.com/bown/2012/product/google-now), Google Now ÃÍ unâĂŹestensione dellâĂŹapplicazione mobile Google Search che, oltre ad essere un assistente vocale, analizza abitudini, ricerche e posizioni ricorrenti per fornire dati e risultati contestuali, mostrando allâĂŹutente le informazioni prima che egli ne faccia richiesta esplicita allâĂŹapp.

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 puÃš essere riassunto in una lista di eventi, di storie di vita; allo stesso tempo, un evento puÃš essere identificato da una coppia <t,s>, dove t ÃÍ la coordinata temporale e s descrive la posizione nello spazio. Utilizzando questo modello, il criterio di cui sopra diventa quello della distanza spazio-temporale tra due entitÃă, e il calcolo di questa distanza diventa il mezzo per effettuare unâĂŹindicizzazione preliminare delle risorse.

Un interessante esempio di sfruttamento estensivo delle informazione su tempo e spazio al fine di restituire dei risultati a una query ÃÍ TimeTrails. TimeTrails ÃÍ un sistema per lâĂŹestrazione e lâĂŹesplorazione 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 (http://it.wikipedia.org/wiki/Wikipedia:Vetrina)), si occupa di estrarre le date e i luoghi contenuti nel testo, normalizzarli (e.g. da âĂIJ25 luglio 1991âĂİ a âĂIJ25/07/1991âĂİ per le prime e da âĂIJPergine ValsuganaâĂİ a âĂIJ46.06853620, 11.23528960âĂİ 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 dellâĂŹelaborazione, cioÃÍ il documento originale unito a una sequenza ordinata di tuple <t,s>, viene poi salvato in (2) un database ottimizzato per contenere dati su luoghi (http://postgis.net/), dal quale (3) unâĂŹinterfaccia permette la ricerca testuale di documenti, con un risultato che verrÃă mostrato su una mappa sotto forma di traiettoria, rappresentata da tutte le tuple lette dal db. Nel caso la query dovesse restituire piÃź di un documento, puÃš risultare interessante vedere dove e quando le traiettorie si incrociano: se i documenti sono biografie, lâĂŹintersezione delle traiettorie in uno dei punti segnati dalle tuple <t,s> puÃš indicare che i personaggi si sono incontrati.

3. PROBLEMA

Nelle ultime righe dellâĂŹintroduzione ÃÍ stato presentato il problema che il lavoro presentato ha tentato di risolvere, cioÃÍ come sfruttare lâĂŹenorme quantitÃă di informazione reperibile sul web per innescare automaticamente la reminiscenza in un individuo. La questione si fa spinosa, in quanto ÃÍ necessario raggiungere un compromesso tra qualitÃă dellâĂŹinformazione, facilitÃă nel reperimento della stessa, e velocitÃă di elaborazione, questâĂŹultima indispensabile se lâĂŹobiettivo ÃÍ la costruzione di un sistema accessibile al pubblico e capace di servire un buon numero di utenti contemporaneamente. Inoltre, tralasciando tutto ciÃš che non ÃÍ visibile allâĂŹutilizzatore finale, ÃÍ indispensabile formalizzare il fulcro del problema in termini di input e di output, e partire da questa formalizzazione per unâĂŹanalisi piÃź 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 lâĂŹafflusso di ricordi nel soggetto a cui vengono presentati. LâĂŹoggetto della trattazione diventa perciÃš, a partire da informazioni geografiche e cronologiche rappresentanti degli eventi nella vita dellâĂŹutente, quello di restituire il contesto piÃź vicino possibile alla vita di questa persona. Utilizzando un formalismo matematico, il problema, a partire dalle entitÃă

PREMESSE,

ÃÍ rappresentabile con

ALGORITMO LATEX,

dove LEGENDA. 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 ÃÍ un indice di risorse storiche, da consultare per ottenere le entitÃă piÃź pertinenti alla vita di un individuo; proprio per decidere cosa ÃÍ attinente e cosa no, un algoritmo deve essere pensato e implementato; affinchÃÍ il risultato dellâĂŹimplementazione sia scalabile e flessibile, deve essere scelto un meccanismo di accesso aperto e utilizzabile in piÃź 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 etÃă, che, rappresentando una fetta consistente dei possibili utilizzatori, richiedono una progettazione piÃź attenta.

Nella realizzazione di un sistema capace di rispondere allâĂŹutente con dati relativi al tempo e al luogo in cui egli ha vissuto alcuni momenti della sua vita, la prima difficoltÃă

ÃÍ riscontrabile nella scelta riguardo quale meccanismo utilizzare per recuperare le entitÃă con cui costruire il contesto; per ottenere questo risultato le possibilitÃă si possono ricondurre a tre approcci principali:

Sfruttare i linked data per recuperare a runtime le informazioni necessarie: questo ÃÍ 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. Utilizzare i linked data, collegandoci 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 puÃš 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 entitÃă 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.

QuestâĂŹultima ÃÍ la proposta di soluzione piÃź 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 . Per poter essere meno dipendenti dalle specifiche RDF e SPARQL, certamente ancora in evoluzione, lâĂŹapproccio che abbiamo scelto ÃÍ di per sÃl' il piÃź semplice: perchÃÍ 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? CosÃň nasce Reminiscens, il cui funzionamento e struttura sono spiegati qui sotto.

4.1 Architettura generale

Reminiscens ÃÍ un sistema strutturato di varie parti, che lavorano insieme per fornire numerosi servizi. Le funzionalitÃă di raccolta e memorizzazione dati sono svolte da due database, dei moduli software e unâĂŹinterfaccia grafica: Il primo database ÃÍ la componente fondamentale di Reminiscens, contenente tutto il materiale utilizzato dal sistema per restituire un contesto (di cui si parlerÃă piÃź avanti) allâĂŹutente, mentre il secondo serve a memorizzare eventuali contesti personali (FUTURE WORK); perchÃÍ la Knowledge Base contenga dei dati, questi devono essere raccolti da qualche parte, nel nostro caso il lavoro ÃÍ fatto da dei moduli ETL, ognuno dei quali si occupa di interrogare una determinata risorsa sul web (e.g. DBpedia); la stessa funzione ÃÍ svolta da CrowdMemories, una UI sviluppata da Francesco Maturi che utilizza il crowdsourcing per reperire materiale storico direttamente dagli utenti. LâĂŹaccesso ai dati invece, ÃÍ 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 nella SEZIONE CONCLUSIONI

4.2 Il mio lavoro

Durante il mio lavoro, mi sono trovato a seguire i dati per tutto il percorso, dalla loro raccolta alla loro visualizzazione, passando per tutta la fase di elaborazione. Di seguito ecco le parti che ho sviluppato:

Moduli ETL (Extract, Transform, Load): LâĂŹidea ÃÍ quella di un insieme di moduli 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 ÃÍ vitale perchÃÍ, oltre allâĂŹovvia funzione di recuperare gli elementi che andranno poi restituiti allâĂŹutente in forma di contesto, il 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 maniera testuale). Essendo la lista delle risorse consultabili sul web sempre in crescendo, questa scelta ÃÍ stata fatta per rendere semplice lâĂŹaggiunta e la sostituzione dei moduli, che attualmente supportano la consultazione di Flickr, DBpedia, Catinabib (catalogo di cartoline e incisioni a tema regionale di proprietÃă della biblioteca comunale di Trento) e di dataset del progetto OpenData Trentino. Proprio grazie a questâĂŹultima risorsa, si spera che con il passare del tempo, grazie allâĂŹaggiunta di nuovi datasets, le informazioni contenute nella Knowledge Base e di conseguenza fornite allâĂŹutente possano crescere di quantitÃă e di qualitÃă.

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

Api REST: Per permettere la consultazione della KB senza dover interrogarla direttamente, ho sviluppato queste 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 lâĂŹindice per restituire un contesto. Questo argomento ÃÍ descritto formalmente a [4.3] Per questioni piÃź tecniche, a [nonloso] ÃÍ disponibile la documentazione delle api.

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

Booklet: UnâĂŹultima parte riguarda il lavoro per comporre unâĂŹinterfaccia che potesse essere utile a visualizzare i dati restituiti dalle API, indipendente dai client di Reminiscens: chiedendo allâĂŹutente una decade e un luogo, quello che viene mostrato ÃÍ un libricino sfogliabile tramite browser, e contenente un contesto completo in una forma che possa essere familiare a chi lo guarda, soprattutto nel caso dei meno giovani, che mancano di dimestichezza con i mezzi piÃź tecnologici. Il libro restituito ÃÍ perÃš un insieme di dati impersonali, in quanto per avere un risultato personalizzato la scelta giusta ÃÍ il vero e proprio client di

Reminiscens.

4.3 La costruzione di un contesto

Come definito sopra, il contesto ÃÍ un insieme di immagini, eventi, personaggi e musica che hanno a che fare con il tempo e il luogo di cui lâĂŹutente potrebbe voler scoprire qualcosa, e che potrebbe aiutarlo a raccontare della sua vita. La nostra scelta ÃÍ stata quella di comporre questo contesto utilizzando cinque entitÃă per tipo, selezionate secondo alcuni criteri allâĂŹinterno dellâĂŹindice. Partendo dalla knowledge base, la Lifeincontext API crea un contesto intorno ad una lista di eventi. LâĂŹelaborazione si divide due casi principali, allâĂŹinterno dei quali si snodano diversi sottocasi: se le coppie lat-lon possono essere racchiuse allâĂŹinterno di un cerchio virtuale di raggio 50 km: se la maggior parte delle decadi ÃÍ uguale, viene creato un unico contesto a partire da una delle coppie lat-lon e da quella decade. questo si puÃš ottenere con una singola query che, ordinando i risultati per pertinenza, seleziona direttamente i primi cinque. se il numero degli eventi di input ÃÍ minore di tre, viene creato un contesto componendo dei âĂIJsotto-contestiâĂİ, uno per ogni coppia formata dalle decadi di input e da una delle coppie lat-lon. lâĂŹultimo caso ÃÍ unâĂŹestensione 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 cosÃň come indicate in input, senza elaborazioni addizionali. Oltre al calcolo a partire dalle sole indicazioni spazio-temporali, un ulteriore versione dellâĂŹalgoritmo ÃÍ stata pensata per cercare, filtrare e personalizzare i risultati ottenuti anche in base a delle parole chiave, siano esse argomenti, generi musicali, interessi dellâĂŹutente e oggetto della risorsa: ad esempio, una richiesta riguardante Trento nel decennio 1980, con keyword âĂIJcalcioâĂİ, 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 dallâĂŹutente in fase di configurazione del profilo, potrebbero consegnargli del materiale interessante ed evocativo di momenti del suo passato, stimolando il racconto di episodi della sua vita.

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. Api REST: la scrittura di queste librerie mi ÃÍ stata facilitata dallâĂŹutilizzo di Ruby unito al framework Sinatra, con lâĂŹausilio 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 (http://home.hiwaay.net/ taylorc/toolbox/geography/geoutm.html) e ho adattato un implementazione Ruby dellâĂŹalgoritmo lineare di Megiddo per il calcolo del Minimum Enclosing Circle di un set di punti (http://www.dseifert.net/code/me CRUD: Questa ÃÍ unâĂŹapplicazione 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. Booklet: LâĂŹinterfaccia del booklet si basa su âĂIJ20 things I learned about browsers and the webâĂİ (http://www.20thingsilearned.com/), applicazione web HTML5 resa open source e riadattata da me per le nostre necessitÃă.

5. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

APPENDIX

A. HEADINGS IN APPENDICES

The rules about hierarchical headings discussed above for the body of the article are different in the appendices. In the **appendix** environment, the command **section** is used to indicate the start of each Appendix, with alphabetic order designation (i.e. the first is A, the second B, etc.) and a title (if you include one). So, if you need hierarchical structure *within* an Appendix, start with **subsection** as the highest level. Here is an outline of the body of this document in Appendix-appropriate form:

A.1 Introduction

A.2 The Body of the Paper

A.2.1 Type Changes and Special Characters

A.2.2 Math Equations

Inline (In-text) Equations

Display Equations

A.2.3 Citations

A.2.4 Tables

A.2.5 Figures

A.2.6 Theorem-like Constructs

A Caveat for the TEX Expert

A.3 Conclusions

A.4 Acknowledgments

A.5 Additional Authors

This section is inserted by LaTeX; you do not insert it. You just add the names and information in the `\additionalauthors` command at the start of the document.

A.6 References

Generated by bibtex from your .bib file. Run latex, then bibtex, then latex twice (to resolve references) to create the .bbl file. Insert that .bbl file into the .tex source file and comment out the command `\thebibliography`.

B. MORE HELP FOR THE HARDY

The acm_proc_article-sp document class file itself is chockfull of succinct and helpful comments. If you consider yourself a moderately experienced to expert user of LaTeX, you may find reading it useful but please remember not to change it.