AAL-2009-2-049, ALIAS
D4.4
Module for personalized discovery of new contacts on line

Due Date of Deliverable	2012-12-31
Actual Submission Date	2013-02-22
Workpackage:	4.4
Dissemination Level:	Public
Nature:	Report
Approval Status:	Final
Version:	v1.0
Total Number of Pages:	27
Filename:	D4.4-EURECOM-ContactsDiscovery-v1.0.pdf
Keyword list:	LODE, EventMedia, reconciliation, user interface, recommendation, user profiling, social interaction

Abstract

We describe several techniques utilized in personalized people-to-people recommender systems, different from traditional item-to-user recommender systems. We stress on the importance of social influence to build up a cohesive social network. The social relationship can be detected from online and offline interactions such as events co-participating. We propose two SPARQL-based methods for detecting similar users. On one hand, we construct an updated user profile that will be used to discover similar profiles. On the other hand, we leverage the explicit social relationship indicated if two people have somehow interacted with each other.

History

Version	Date	Reason	RevisedBy
0.1	2012-11-30	created [EURECOM]	Houda Khrouf
0.2	2013-01-28	updated [EURECOM]	Houda Khrouf
0.3	2013-01-31	first review [TUM]	Tobias Rehrl
0.4	2013-02-06	second review [YOUSE]	Sebastian Glende
1.0	2013-02-22	final [EURECOM]	Raphael Troncy

Authors

Partner	Name	Phone / Fax / Email
EURECOM	Raphaël Troncy	Tel: ++33 4 9300 8242 Fax: ++33 4 9300 8200 Email: Raphael.Troncy@eurecom.fr
EURECOM	Houda Khrouf	Email: Houda.Khrouf@eurecom.fr
EURECOM	Vuk Milicic	Email: vuk.milicic@eurecom.fr

Table of Contents

1 Executive Summary

Discovering new contacts in event-based social network is a key factor to enhance the offline and online social interactions. This task requires an efficient people-to-people recommender system taking into account the two-way relationship between users, also called *reciprocal* recommendation. In this deliverable, we first describe our framework designed to crawl, reconcile and visualize data in real-time. Then, we provide a comprehensive view of the important user profiling techniques and recommender systems, and we describe how these systems have been adapted for people discovery. We highlight the importance of two aspects to build a cohesive social network: the similarity between users' profiles based on their recent activities, and the co-participation in offline activities reflecting stronger social ties. Hence, we propose a SPARQL-based method that uses the data collected and infers the user profile based on an ontological representation of user's activities. This method allows to incrementally update the user's interests that vary over the time. Finally, we propose another method that leverages the social relationship indicated if two people have somehow interacted with each other such as co-participating social events.

2 Abbreviations and Acronyms

CF	Collaborative Filtering
FOAF	The Friend Of A Friend
JSON	JavaScript Object Notation
LODE	An ontology for Linking Open Descriptions of Events.
MVC	Model-View-Controller
OWL	The Web Ontology Language is a knowledge representation language based on description logics. It has an RDF syntax and in its dialect OWL-Full (one the three flavors of OWL) includes the RDF/S semantics.
RDF	The Resource Description Framework is a knowledge representation language based on a triple model, and serves as foundation for other semantic web languages such as RDFS or OWL.
RDFS	The RDF Schema is a knowledge representation language that has an RDF syntax.
REST	REpresentational State Transfer
SIOC	Semantically-Interlinked Online Communities
SPARQL	The Semantic Web query language
XML	Extensible Markup Language

3 Introduction

People to people recommender systems have always been a key feature of many online social networks. They exploit personalized information and behavioral patterns to discover new potential relationships between users. Integrating such systems in an event-based service has a valuable advantage to identify users sharing some interests, and to promote face-to-face social interactions. Like conventional online social networks, event-based social networks provide an online virtual world where users exchange thoughts and share experiences. But, the distinction comes from the fact that they also capture the face-to-face social interactions in participating events in the offline physical world [1]. In this work, we aim to expand and enhance our event-based social network by recommending new contacts with which the user would like to communicate, share experience and attend future events. For this purpose, the system must first construct a sufficiently-detailed model of the user's interests through preference elicitation. User profiling can be obtained either explicitly from user's metadata or implicitly by observing the user's behavior. Then, an adapted recommender systems for social networks will be employed to discover similar users sharing same "tastes".

In this deliverable, we first describe our framework designed to crawl and reconcile data (Chapter 4). Indeed, we use this framework to collect data about events and participants. Once data retrieved, we interlink it with external datasets that could be exploited to enrich participants' profiles. Data are then visualized using an interactive interface with the aim to meet the end-user needs: relive experiences based on media, and support decision making for attending upcoming events (Chapter 5). Secondly, to build a recommendation system, we propose two methods for modeling participants' profiles either explicitly or implicitly (Chapter 6). We then describe some existing recommender system that could be used in the context of social networks characterized by two-way relationship between people (Chapter 7). Finally, we give our conclusions and outline future work in Chapter 8.

4 Data Crawling and Reconciliation

The advent of the Social Web rises a need to provide tools that crawl data from multiple social services with less effort. Such tools should be able to deal with many tasks such as policy management, requests chaining, data integration or merging responses schemas [2]. We propose a framework that supports those tasks and unifies information sources into a meaningful data model [3]. Once data collected, we reconcile it in real-time with external datasets such as Wikipedia, Musibrainz and BBC that could be harnessed for user profiling. We also provide a Web dashboard that easily handles data crawling and reconciliation, and provides useful statistics about the dataset.

4.1 Data Crawling

The data crawling framework is illustrated in Figure 4.1. It is composed of two main components: the Unified REST Module and the Scraping Processor.

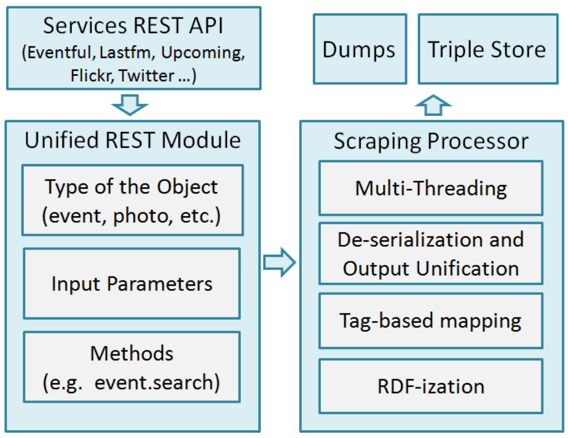

Services REST API
(Eventful, Lastfm, Upcoming, Flickr, Twitter ...)
Dumps
Triple Store
Unified REST Module
Type of the Object (event, photo, etc.)
Input Parameters
Methods (e.g. event.search)
Scraping Processor
Multi-Threading
De-serialization and Output Unification
Tag-based mapping
RDF-ization

Figure 4.1: The Rest-based Crawler Architecture

The first REST module allows for the unification of various Web APIs exploiting their commonalities in terms of described methods, objects and input parameters. Each source API is attached to one JSON serialized file that describes the related server rules such as root URL and API key, and a set of query objects. Each query object describes an API method and the input parameters. We defined new methods and mapped them with the targeted web API. For instance, the method for collecting events takes as input a set of parameters such as source (e.g. eventful, upcoming, etc.), category, location, date and keywords. A user can specify with a single query multiple sources and request in parallel various information. The RESTful service is flexible enough so that new methods can conveniently be created, and new similar REST-inspired Web APIs can be integrated by adding their JSON file descriptors.

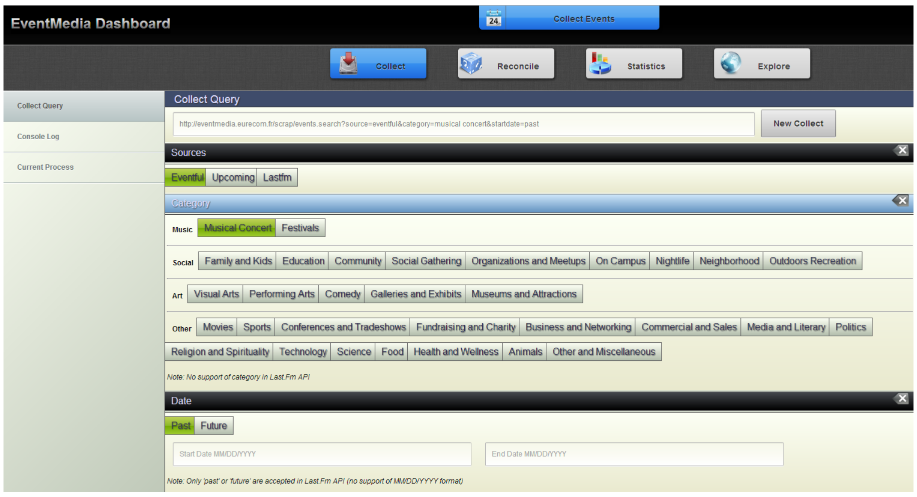
Besides to the REST module, the Scraping Processor manages four important tasks. The first task handles a limited number of multi-thread requests, and many sources can be specified into one query. This will contribute to save a tremendous amount of time usually required to fetch information from web services. The other task mainly deals with data processing starting from JSON de-serialization to RDF conversion into LODE [4] ontology, and finally loading into a triple store. More precisely, the data retrieved is de-serialized and exported into a common schema providing descriptions of events, venues, agents, attendees and photos. Then, we use tag-based mapping consuming some meta-data, not only to establish links between events and media but also to enrich their descriptions with additional information from external datasets.

4.2 Data Reconciliation

At the core of our system is the real-time reconciliation framework that aligns every incoming stream of overlapping but highly heterogeneous data sources. This will sustain a continuous content enhancement, a crucial task to cope with the dynamics of social services. The major gain is to bring context and expand the reach of an event at different stages. In fact, viewing an event page from one event-based service underlines an incomplete content that needs to be further enriched. For instance, we have always detected a real lack of involved agents and their descriptions in the Upcoming directory. While in Last.Fm, people seem to be more responsive to attend events, but only limited to musical concerts without consistent description. We believe that reconciling event-centric data will leverage the benefits of each service and achieve a better overview of an event. Hence, we first explore the connections between events and media using tag-based mapping. Then, we mine meaningful connections between event components, and interlink them with external datasets based on instance matching paradigm. In order to reconcile entities in real-time, we should first select a set of instances of the source dataset by their issuance dates. Then, a SPARQL query is performed for each instance to fetch similar candidates filtering by some blocking keys. Four semantic connections have been addressed: events, media, agents and locations connections (more details are available in [5, 6]).

4.3 EventMedia Dashboard

To easily handle data crawling and reconciliation tasks, we design a user interface 4.2 composed of four main views: collect, reconcile, statistics and explore views. The collect and reconcile views provide a set of graphical widgets that allows for selecting query



Collect Query: http://eventmedia.eurecom.fr/scrap/events.search?source=eventful&category=musical concert&startdate=past

Sources: [✓ Eventful] Upcoming Lastfm

Category: Music [✓ Musical Concert]

Note: No support of category in Last.Fm API

Date: [✓ Past] Future

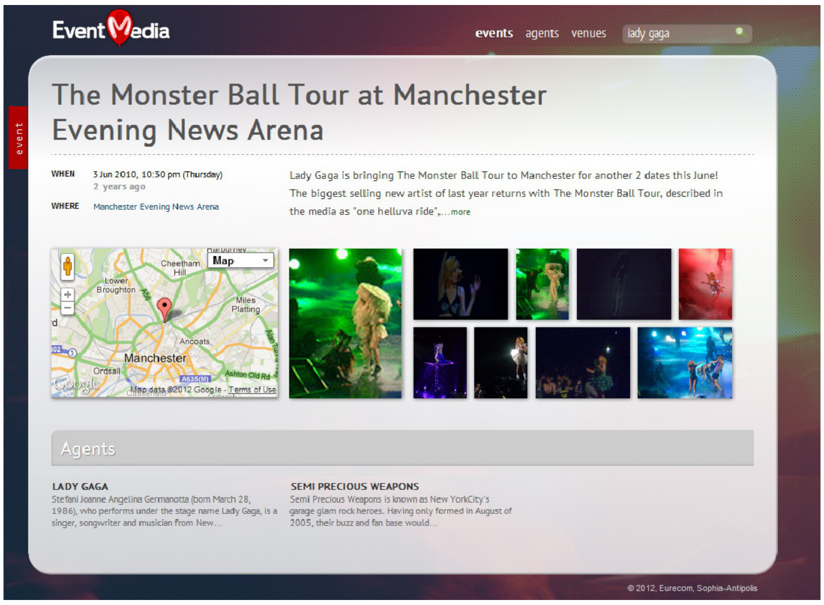
Note: Only 'past' or 'future' are accepted in Last.Fm API (no support of MM/DD/YYYY format)

Figure 4.2: The Rest-based Crawler Architecture

arguments, and track the progress of ongoing tasks. The statistics view provides useful information about the datasets such as the amount of collected data per source and time, and the existing links with external datasets or numbers of events per category. Finally, the explore view provides a set of useful links such as the SPARQL endpoint and the end-user interface. The web dashboard is available on-line at `http://eventmedia.eurecom.fr/dashboard`.

5 User Interface: Live your Event

One of the challenges we want to address is how to enable fluid faceted navigation of a vast event-based space, and to create harmonious views of the interconnected datasets. Users wish to discover events either through invitations and recommendations, or by filtering available events according to their interests and constraints [7]. We provide mechanisms to browse events by location or a period of time. Once an event is selected, media are presented to convey the event experience, along with background information such as category, agents, venues, attendance list, ticket, etc. A typical example is illustrated in Figure5.1.



The Monster Ball Tour at Manchester Evening News Arena

WHEN 3 Jun 2010, 10:30 pm (Thursday)
2 years ago

WHERE Manchester Evening News Arena

Lady Gaga is bringing The Monster Ball Tour to Manchester for another 2 dates this June! The biggest selling new artist of last year returns with The Monster Ball Tour, described in the media as "one helluva ride",...more

LADY GAGA
Stefani Joanne Angelina Germanotta (born March 28, 1986), who performs under the stage name Lady Gaga, is a singer, songwriter and musician from New...

SEMI PRECIOUS WEAPONS
Semi Precious Weapons is known as New YorkCity's garage glam rock heroes. Having only formed in August of 2005, their buzz and fan base would...

Figure 5.1: Interface illustrating a concert of *Lady Gaga* in 2010

Apart from the inspection of the event instance, other conceptual classes (e.g. venues, agents, users) have also accessible views, so that the user can obtain more information about these instances and explore events related to them. We also leverage data from open datasets to be displayed in an infobox separated from the main information, but some parts of this data are interwoven with the main data as well, or used to replace missing data. Finally, we incorporate recommendation mechanism sorting events by popularity. Intuitively, we consider that more attendees an event has, more popular it is. The demonstration of EventMedia is available at `http://eventmedia.eurecom.fr`.

5.1 Mismatch Between Front-end and Back-end

EventMedia is a large knowledge base containing descriptions of events and media derived from several public directories. It focuses on data interlinking to connect events with their associated media documents, and reinforcing the cross-linking with other data sources. Data in EventMedia is stored in an RDF triplestore and can be accessed via a SPARQL endpoint. Data is also published according to the Linked Data principles. Unfortunately, this model doesn't work well with the modern approach to web development which requires REST API. Although there are similarities and Linked Data partially provides the functionality of the API, it has several drawbacks.

Linked Data does not provide features needed by a typical web application. There is no concept of collection resources, and no way of manipulating a number of items and their offset, or sorting and filtering. A developer has to choose between accessing data either in a highly limited (Linked Data) or a very sophisticated way (SPARQL). The second main problem is that there is a mismatch between the graph-based RDF model and the object (hierarchical) model used in object-oriented programming languages.

Dealing with RDF data on the frontend side demands specialized libraries that make a process harder and against the standard methods and best practices. In addition, in a model component of JavaScript MVC (Model-View-Controller) framework, data is represented and serialized using JSON. Although there are ways to represent RDF using JSON, such a JSON significantly differs from a simple idiomatic JSON typically used in JavaScript and must be additionally parsed and reduced in expressivity.

5.2 Technologies

Linked Data API provides a solution for the described problem. We implemented Elda[1], which enables a configurable way to access RDF data using simple RESTful URLs that are translated into queries to a SPARQL endpoint. Besides the RDF syntax (Turtle and RDF/XML), it also provides a simplified XML and JSON representations of RDF data, suitable for use in the context of JavaScript Frameworks.

We used a popular Backbone.js JavaScript framework[2] to facilitate developing the complex user interface. It is a simple but powerful MVC framework, providing Model, Collection, View and Router constructor, together with Event constructor for supporting Pub-/Sub pattern. For this project, its high level of flexibility is especially important due to provided elegant way of customization the framework in order to satisfy specific needs. Finally, it provides an elegant REST integration that make dealing with Elda REST implementation painless.

[1] http://code.google.com/p/linked-data-api/wiki/Specification

[2] http://backbonejs.org/

5.3 User Interface Challenges

High variability of data. From the user interface perspective, the first challenge we have faced is the high variability of data. Both amount and structure vary significantly. Therefore, the user interface must be flexible enough to support entities containing large amounts of data, as well as the ones that are practically empty. The solution is well organized yet flexible grid-based layout in which each piece of data is positioned in certain location. If some piece of data is missing, that is indicated by white space, while the main structure is preserved. Potentially long textual descriptions are shortened to fixed height and length, with a buttons that reveals the whole text. A collection of images of unknown number and proportions are realized in similar fashion, using fixed height and overflow property set to “hidden”, forcing hiding images that could otherwise “break” the layout. Finally, in certain situations the layout adjusts itself according to the content. Concretely, if an agent lacks both infobox and gallery, placed in the right hand side, the width of its description is extended to the whole page width instead of the half of it.

Asynchronous data retrieving. EventMedia uses data from different sources. The main source is the EventMedia dataset, which already contains reconciled and interlinked data from different sources. On the frontend side, the `owl:sameAs` properties are followed in order to fetch additional data from DBpedia, MusicBrainz, etc. Most of data from external sources is rendered in an infobox separated from the main information, but some parts of it are interwoven with the main data as well, or used to replace missing data. For instance, if the agent is missing the description, the one from DBpedia is used instead. Another problem is the fact that in DBpedia there is a significant number of broken links of images. The value of `foaf:depiction` property is used for showing the image, falling back to the `foaf:thumbnail` or removing image completely if that link is broken too.

From the technical standpoint, the challenge here is to catch all the relevant events and act accordingly. In addition, the images’ “error” events that can take some time, making changes in layout in surprising moments of time. Because of the random order of HTTP responses, knowing exactly when the page is finished rendering can also be a challenge.

Adding content dynamically to the upper part of the page causes lower parts of to change the vertical position, i.e. “jump” up and down. This is annoying, especially if happens unexpectedly some time after the initial page rendering. The solution is to fix the height of the whole upper (main) part of the page, preventing the movement of the lower part. This way, we are simultaneously solving the problem described in the section 4.1. as well, preventing that the lower part of the page goes too far away. The user is provided with the option to show the whole information at will.

Making UI responsive. One of the biggest challenges of making EventMedia user interface has been making it work on the small screens of mobile devices. This constraint has significantly influenced the user interface. The *mobile-first philosophy* has led to highly modular design and encouraged a simple design with every unnecessary detail removed.

The modules are following the grid, allowing organizing the content into the columns of various widths and recombining their position when necessary. On the smallest screens, all modules have width set to 100%. When the screen width increases, depending on the available size and situation, more pleasing layout is generated with modules' width reduced to 50% or 33% of the page width. In general, the responsive design is achieved using Media Queries, relative units for 'width' properties and changing the flow of elements.

Visual design. Having all the imposed constraints discussed in the previous sections, the final challenge is to come up with functional visual design. We already have several parameters that direct the final concept: simplicity, modularity, precise grid-based layout. The data (events) coming from EventMedia dataset is highly diverse, ranging from festivals, small concerts, conferences, venues and bands of different genres. Achieving consistency despite all the variability of data is the goal here. As an example, a visual design that works well with a trash-metal band would look odd on the page showing some jazz musician. As a solution, we choose to go with a neutral design, with dominant black and white and the minimal use of other colors. We attempt to use elegance as a "common denominator", which is emphasized with high contrasts, semi-transparency and elegant, magazine font. However, to prevent the design from being too neutral (and perhaps boring), we have attempted to personalize pages, by using available images for page background, evoking the atmosphere from the event or other entity in question.

Having elderly people as users entail additional constraints on the interface, such as enabling magnificence for increasing the font size of the textual description of the events, artists and locations. The minimal and responsive design discussed above enables to fulfill those requirements.

6 User Profiling

A user profile represents all kind of information related to a user and user's context, which are required in order to provide personalized user experience. It can hold various features of the user (user characteristics) as an individual such as demographic information (age, education level, interests, preferences, knowledge etc.), or can represent the overall context of the user's work, including platform, bandwidth or location [8]. The motivation of building user profiles is that users differ in their preferences, interests and background when using a specific application. Discovering these differences is vital to providing users with personalized services. A profile can be expressed through a set of predefined arguments, in which the user explicitly identifies his general interests to particular topics. Interests can represent news topics, web page topics, document topics or hobbies-related topics. In addition, a profile can be constructed by integrating information from user's activities, such as attended events, browsing histories, etc.

To represent a user profile, different techniques have been used. The most common representations is the keyword-based model, in which interests are represented by weighted vectors of keywords in general computed using TF-IDF (term frequency/inverse document frequency) [9, 10]. A more advanced representation of user profile is topic hierarchies [11]. Each node in the hierarchy represents a topic of interest for a user, which is defined by a set of representative words. In this work, we propose two methods to construct a profile: (i) the first method is based on an explicit description of the user's characteristics such as hobbies. (ii) the second method uses SPARQL queries to analyze the user's behavioural patterns and infer some interests such as the favorite artists and venues.

6.1 Explicit Profiling

The explicit profiling can be constructed from the input entered via forms or other user interfaces. The information collected are in general demographic such as the user's age, birthday, gender and hobbies. Another way to obtain explicitly users' interests is to analyze the direct expressions materialized as likes/dislikes buttons, ratings and feedbacks. For example, the user can express opinions about past events, and rate artists according to his preferences. These ratings are also used to recommend interesting items to other similar users.

Although the explicit profiling seems a straightforward way to collect information about users, it raises some problems usually encountered in recommender systems. In fact, the information obtained can be inconsistent and incomplete since users are not willing to fill in long forms, and they might be confused to express their interests. Thus, the most widely used method for obtaining information about users is observing their actions with the underlying application [12].

6.2 Implicit Profiling

Observing the user's activities is a key solution to construct implicitly the user profiling. The system should be able to identify behavioral patterns reflecting repetitive actions in different time points. In the following, we describe different techniques existing in the literature, and we present our semantic-driven approach.

6.2.1 Data Mining Techniques

A key characteristic of learning through observation is that of adapting to the user's changing interests, preferences, habits and goals. The user profiling techniques used have to be able to adapt the content of the user profile as new observations are recorded [12]. For example, the system in [11] observes the browsing behavior to estimate the interest of the user to each visited web page. Some implicit indicators include the time consumed in reading a page, the amount of scrolling, and whether it is added to bookmarks or not.

To automatically build the user profile, several techniques have been employed in the areas of machine learning, data mining and information retrieval such as probabilistic models, neural networks or association rules. These techniques attempt to convert raw data into usable information about the user, and most of them classify a user into a certain group. For example in [13], the authors use Bayesian networks to detect the preferred learning style according to the modeled behaviors in e-learning systems. Indeed, Bayesian Networks have been widely applied to learn and reason probabilistic relationships among variables with the aim to predict user preferences. In [14], the authors use Association Rules, a data mining technique for discovering interesting relationships in large volume of transactions. It captures the relations among items based on patterns of co-occurrence across transactions. An example of an association rule is that 90% of transactions that purchase bread and butter also purchase milk. Although the Association Rule seems promising to learn behavioral patterns, it may lead to some 'nonsense' rules difficult to filter out.

One problem commonly faced when using the above mentioned techniques is the cold-start problem, where a minimum amount of information has to be acquired to discover behavioral patterns. In [15], the authors classify the cold-start problem into two types: (i) The new-system cold-start problem is where the recommender systems have no initial ratings by users and no basis on which to recommend, and hence perform very poorly; (ii) The new-user cold-start problem is where the recommender system has no enough information about a new user, and hence they perform poorly too. The authors highlight one solution to alleviate this problem through the use of ontoloy-based profiling.

6.2.2 Ontological Profiling

One increasingly popular technique to accurately identify the user context is the ontological profiling, recently used for personalized search. The idea behind is to use existing ontologies to match the initial user behavior with existing concepts and relationships between these concepts. Systems such as Quickstep [15] or ARCH [16] are based on domain ontologies as source of semantic knowledge. They use ontological relationships between topics of interest to infer other topics of interest, which might not have been browsed explicitly. The ontological approach to user profiling has proven to be successful in addressing the cold-start problem in recommender systems where no initial information is available early on upon which to base recommendations. However, to the best of our knowledge, there is no previous work based on an ontological approach for recommendation in event-based social network.

6.2.3 SPARQL-based Inferred Profiling

In this work, we propose a novel approach to model the user's activities around events, so that the user's profile could straightforwardly be inferred. In order to conveniently describe the user context in event-based service, we propose the ontological model in Figure 6.1, in which a set of known vocabularies have been used such as LODE[1], SIOC[2] and FOAF[3]. This model provides an insight about user's past activities formalized in a machine-readable format. It depicts the semantic relationships linking the user identified by "Gabriela Pirela" on Last.fm with other items such as event, location and agent. By inferring what events, locations, artists in which the user has been interested, we identify a set of social interests that could be exploited in recommender systems.

Using the SPARQL queries, we mine the meaningful and semantic relationships that link a user with different items. We attempt to discover three social dimensions of interest respectively related to event, location and agent, thus constituting the user's profile. We consider that the weight of user's interest in one dimension will depend on the attendance frequency. For example, more the user attends to "Coldplay" concerts, more he is likely interested to this artist. Indeed, the attendance frequency allows to incrementally update the scores of user's interests that vary over the time. We define the following dimensions of interest that, of course, could be enriched with reconciled external datasets (Wikpedia, BBC, MusicBrainz and Foursquare).

Event-topics of interest. An event is related to specific topics represented as categories or tags. These topics can be considered as part of interests of involved users. Listing 6.1

[1] `http://linkedevents.org/ontology/`

[2] `http://sioc-project.org/ontology`

[3] `http://xmlns.com/foaf/spec/`

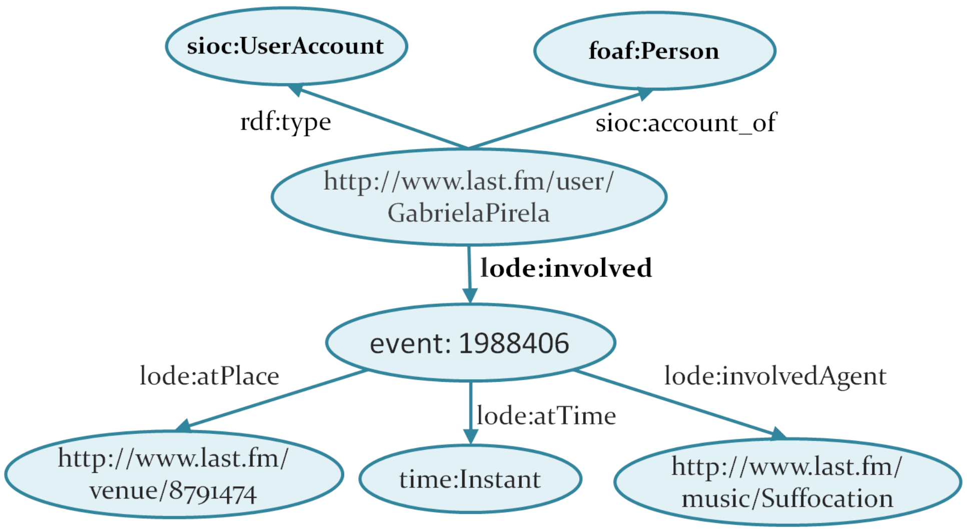


Figure 6.1: The metadata attached to the user named *Gabriela Pirela*

illustrates the SPARQL query to retrieve the list of topics of events to which the user has attended.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lode: <http://linkedevents.org/ontology/>
PREFIX dc: <http://purl.org/dc/elements/1.1/>
SELECT ?user (sql:group_digest(?topic , ', ', 1000, 1)) as ?topics
WHERE {

  ?event rdf:type lode:Event .
  ?event lode:involved ?user .
  { ?event lode:hasCategory ?topic } union { ?event dc:subject ?topic }

} GROUP BY ?user
```

Listing 6.1: SPARQL query to retrieve event topics of users' interests

Agent of interest. An agent is considered as one of important features that attract users to attend an event. Obviously, this agent can reflect the user's interests, for example in terms of his preferred music. Listing 6.2 illustrates the SPARQL query to retrieve the list of agents of interest.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lode: <http://linkedevents.org/ontology/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT ?user (sql:group_digest(?label, ', ', 1000, 1)) as ?agents
FROM <http://data.linkedevents.org/eventful/>
WHERE {

 ?event rdf:type lode:Event .
 ?event lode:involved ?user .
 ?event lode:involvedAgent ?agent .
 ?agent rdfs:label ?label

} GROUP BY ?user
```

Listing 6.2: SPARQL query to retrieve agents of users' interests

Location of interest. A location in which an event has been happened can be an indicator about the most visited place by a given user. This enables to detect users located in same geo-graphical areas. Listing 6.3 illustrates the SPARQL query to retrieve the list of locations that reflect the user's whereabouts.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lode: <http://linkedevents.org/ontology/>
PREFIX vcard: <http://www.w3.org/2006/vcard/ns#>
SELECT ?user (sql:group_digest(?locality, ', ', 1000, 1)) as ?cities
WHERE {

 ?event rdf:type lode:Event .
 ?event lode:involved ?user .
 ?event lode:atPlace ?place .
 ?place vcard:adr ?node .
 ?node vcard:locality ?locality

} GROUP BY ?user
```

Listing 6.3: SPARQL query to retrieve cities of users' interests

7 Contacts Discovery

The Contacts Discovery is a key factor to enhance the social interactions in EventMedia. It is also seen as people-to-people personalized recommendation based on user's activities and preferences. The basis is to predict a potential social link reflecting high probability to connect two similar users. This prediction can take into account different dimensions such as the user's preferences (ratings, likes, etc), the user's activities (visited places, most listened songs), or even the social interactions (two users attend the same event). Nowadays, in online applications with a very large number of choices where customer taste is important in making selections, personalized recommendation of items or people becomes essential [17]. Thus, a large variety of recommender systems has been proposed to cope with the ever increasing information about the users' habits on the web. Well-known examples of recommending systems are Amazon[1], Pandora Radio[2], Netflix[3] and Last.fm[4]. However, different from the traditional items-to-people recommenders, the people-to-people recommenders must satisfy both parties, which is called reciprocal [18] recommender. In the following, we describe the traditional approaches of recommender systems and how they have been adapted to cope with the reciprocal nature of the user-to-user relationship. We finally propose a SPARQL-based recommendation that takes into account the social relationships.

7.1 Content-based Recommendation

Broadly speaking, the recommender systems are based on one of two strategies: the content-based filtering and the collaborative filtering. In the former strategy, the recommendations are based on the content about the items or users. They analyze the information collected explicitly or implicitly in form of users or items profiles. The matching between both profiles is quantified using a variety of similarity distances such as Cosine similarity, Pearson correlation or Latent Semantic Analysis [19]. This kind of matching is also applied to discover people sharing similar interests. It is closely related to detecting documents of similar content in information retrieval field. A known successful realization of content-based filtering is the Music Genome Project, which is used for the Internet radio service Pandora.com. A trained music analyst scores each song in the Music Genome Project based on hundreds of distinct musical characteristics. These attributes, or genes, capture not only a song's musical identity but also many significant qualities that are relevant to understanding listeners' musical preferences [20]. An interesting eval-

[1] `http://www.amazon.com/`
[2] `http://www.pandora.com/`
[3] `http://www.netflix.com/`
[4] `http://www.last.fm/`

uation in [21] has been conducted to compare between different recommender algorithms in the IBM's enterprise social networking service ”Beehive”. The authors find out that the pure user-content matching is the most effective to recommend unknown friends. However, a recommendation is considered good only if there is a minimum threshold of information. In other words, the similar users should share a minimum number of interest that must be enough to get connected.

7.2 Collaborative Filtering Recommendation

The second strategy of recommender system is based on collaborative filtering(CF), a technique that do not need an explicit profiling and purely rely on past user behavior [22]. It has been widely employed in many well-known services such as Amazon, Faceboook, LinkedIn, MySpace and Last.fm. The basis is to analyze the relationships between users and interdependencies among items to identify new user-item associations. In other words, the system makes automatic predictions (filtering) about the interests of a user by collecting preferences from many users (collaborating). The intuition behind is that if a person A has the same preference as a person B on an item, A is more likely to have B's preference on another item. The two primary categories of collaborative filtering are memory-based and model-based approaches. The memory-based approaches compute the similarity between users or, alternatively, between items based on users preferences data, thus detecting the neighbors of a given user or item. Indeed, the unknown rating value of the active user u for an item m is an aggregation of the ratings of users similar to u for the same item m, or an aggregation of the ratings of the user u to similar items of m. The model-based approaches, on the other hand, use data mining and machine learning algorithms to estimate or learn a model from observed ratings to make predictions. For instance, the Clustering model estimates the probability that a particular user is in a particular cluster C, and from there computes the conditional probability of ratings. Another typical technique is the latent factor model that discovers unobserved factors from ratings patterns. The underlying assumption is that there is a set of common hidden factors which explain a set of observations in co-occurrence data. More precisely, the similarity between users and items is simultaneously induced by some hidden lower-dimensional structure in the data. For example, the rating that a user gives to a movie might be assumed to depend on few implicit factors such as the user's taste across various movie genres [23]. Recently, several matrix factorization methods [20] have been proposed as a successful realization of latent factor model. The users and items are simultaneously represented as unknown feature vectors within a user-item matrix. These feature vectors are learnt using low-rank approximations, so that they approximate the known preference ratings with respect to some loss measure.

Although the success of CF recommendation, it suffers from three serious limitations: the sparsity problem where there are few rating of total number of items, the cold-start problem explained in the previous section, and finally the scalability where large amount of users and items have to be analyzed.

Despite of the wide application of CF approach, its application for people-to-people recommendation in social networks is not trivial. The limitation is that in the traditional CF approach, only the preference of users counts and the items are passive. However in social networks, the *items* are also users who are part of the social interactions. This needs to consider the reciprocal effect from two sides, instead of from only one side. In other words, for a given user u, the system has to find interesting users that will respond favorably to him. This two-way nature of user-to-user recommendation has been the subject of some studies [21, 17, 24]. In [21], the collaborative filtering is a typical friend-of-a-friend approach and uses only linking information from the social network. It is based on the intuition that if many of A's connections are connected to B, then A may like to connect to B too. This approach was more successful in finding known contacts compared with content-based approach. In [17], a bilateral CF algorithm called SocialCollab has been proposed. It is based on user similarity in taste and attractiveness. Two users are similar in attractiveness if they are liked by a common group of users, and these two users are similar in taste if they like a common group of users. The intuition behind is that the attractiveness of the recommended user (*resp.* active user) should match the taste of the active user (*resp.* recommended user). This approach was shown to outperform the standard collaborative filtering, confirming the importance of reciprocity in people-to-people recommenders. In the same area, the work in [24] reported improvements over SocialCollab algorithm. The approach used combines the content-based information and relies on tensor [25] decomposition to generate recommendations, a method mainly employed in context-aware recommendation.

7.3 Link Recommendation

The contacts discovery can also be seen as a link recommendation in social network. Some studies [26, 27] propose a graph-based strategy that combines analysis of link structure in social networks with analysis of content, such as shared interests. The work in [27] propose a recommender system called LJMiner for the popular weblog service LiveJournal. The results suggest that features constructed only from the common interests of two users are ineffective at predicting friend relationships. In [26], the authors propose to augment the social graph with attributes (user's interest) as additional nodes, and use the random walk algorithm on this augmented graph. Then, they define a set of weights that increase the probability of random walk according to some well-defined desiderata for link relevance such as homophily, social closeness and common friendship.

7.4 SPARQL-based Recommendation

The social interactions plays an important role to determine the strength of tie between two users. A form of social interactions is the collaborative participation such as co-authoring a paper or commenting each others' blog. In [21], a recommendation algorithm based on such interaction has been evaluated. The system incorporates all information that implies an explicit acquaintance between pairs of people, and ranks them based on the strength and frequency of their interactions on record. The evaluation highlights the role of algorithm based on explicit social relationship to recommend known people, followed by friend-of-friend algorithm and compared to content-based algorithm. In addition, the users rated as good the most of recommendations proposing known people, compared to unknown people. One reason is that more strangers an algorithm recommends, the more likely users will reject or not like the recommendations.

Following the same intuition for the event-based social network, we consider that two users involving in the same event can potentially have a stronger tie than others users. Obviously, more events in which they involve, more stronger is their relationship. The involvement could be offline by directly attending the event, or online by sharing media illustrating this event. This fact has also been highlighted in [1], where the authors stress on the importance of co-participating social events to build an offline social network, but naturally less denser compared to online social network. Based on these observations, we use two SPARQL queries that straightforwardly discover similar users based on events participation (see Listing 7.1 and Listing 7.2).

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lode: <http://linkedevents.org/ontology/>
PREFIX sioc: <http://rdfs.org/sioc/ns#>
SELECT DISTINCT  ?user1 ?user2 COUNT(DISTINCT ?event) as ?NbEvents
FROM <http://data.linkedevents.org/upcoming/>
WHERE {

 ?event rdf:type lode:Event .
 ?user1 rdf:type sioc:UserAccount .
 ?user2 rdf:type sioc:UserAccount .
 ?event lode:involved ?user1 .
 ?event lode:involved ?user2 .
 FILTER (?user1!=?user2)

} ORDER BY DESC (?NbEvents)
```

Listing 7.1: SPARQL query to retrieve the number of attended events by two users

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX lode: <http://linkedevents.org/ontology/>
PREFIX sioc: <http://rdfs.org/sioc/ns#>
SELECT DISTINCT ?user1 ?user2 COUNT(DISTINCT ?event) as ?NbEvents

WHERE {
 ?photo1 lode:illustrate ?event .
 ?photo1 sioc:hasCreator ?user1 .
 ?photo2 lode:illustrate ?event .
 ?photo2 sioc:hasCreator ?user2 .
 FILTER (?user1 != ?user2)

} ORDER BY DESC (?NbEvents)
```

Listing 7.2: SPARQL query to retrieve the number of events, about which two users shared photos

8 Conclusion

The integration of event-centric information from social services using linked data technologies gives rise to EventMedia, an open dataset continuously updated and reconciled with external datasets. To augment the social interactions and enhance the usability of EventMedia, we have been interested in people-to-people recommender systems. Several studies have been reported in this research area highlighting the impact of reciprocal recommendation on the design of traditional recommender algorithms. We also describe a novel approach for ontological user profiling in event-based network taking into account three dimensions of interest: topics, agents and locations. Moreover, the offline and online social interactions around events plays an important role to discover known people. One potentially promising way to build up a social network is to leverage the relationship-based algorithm to find known people, and then extend the network using users profiles similarity. In the future, we aim at creating and integrating a real and rich social network in EventMedia. We also plan to evaluate the side effects of the reconciliation and the cross-linking of users profiles from social services on the recommendation.

Bibliography

[1] X. Liu, Q. He, Y. Tian, W.-C. Lee, J. McPherson, and J. Han, "Event-based social networks: Linking the online and offline social worlds," in *18th ACM SIGKDD conference on Knowledge Discovery and Data Mining*, KDD'12, (Beijing, China), 2012.

[2] G. Gouriten and P. Senellart, "API BLENDER: A Uniform Interface to Social Platform APIs," in *21st World Wide Web Conference (WWW'12)*, (Lyon, France), 2012.

[3] H. Khrouf and R. Troncy, "Eventmedia live: a lod dataset of events illustrated with media," in *Semantic Web journal, Special Issue on Linked Dataset descriptions*, 06 2012.

[4] R. Shaw, R. Troncy, and L. Hardman, "LODE: Linking Open Descriptions Of Events," in *4th Asian Semantic Web Conference (ASWC'09)*, 2009.

[5] H. Khrouf, V. Milicic, and R. Troncy, "Eventmedia live: Exploring events connections in real-time to enhance content," in *Semantic Web Challenge (International Semantic Web Conferrence)*, (Boston, United States), 2012.

[6] H. Khrouf and R. Troncy, "EventMedia Live: Reconciliating Events Descriptions in the Web of Data," in *6th International Workshop on Ontology Matching (OM'11)*, (Bonn, Germany), 2011.

[7] R. Troncy, B. Malocha, and A. Fialho, "Linking Events with Media," in *6th International Conference on Semantic Systems (I-SEMANTICS'10)*, (Graz, Austria), 2010.

[8] M. Barla, "Towards social-based user modeling and personalization," in *Information Sciences and Technologies Bulletin of the ACM Slovakia*, vol. 3, pp. 52–60, 2011.

[9] H. Lieberman, C. Fry, and L. Weitzman, "Exploring the web with reconnaissance agents," in *Communications of the ACM*, pp. 69–75, Aug. 2001.

[10] D. Billsus and M. J. Pazzani, "A personal news agent that talks, learns and explains," in *3rd Annual Conference on Autonomous Agents*, (Seattle, Washington, United States), 1999.

[11] D. Godoy, S. Schiaffino, and A. Amandi, "Interface agents personalizing web-based tasks," in *Cognitive Systems Research Journal (Special Issue on Intelligent Agents and Data Mining for Cognitive Systems)*, vol. 5, pp. 207–222, 2004.

[12] M. Bramer, ed., *Artificial Intelligence: An International Perspective*, vol. 5640 of *Lecture Notes in Computer Science*. Springer, 2009.

[13] P. García, A. Amandi, S. Schiaffino, and M. Campo, "Evaluating bayesian networks' precision for detecting students' learning styles," in *Computers in Education Journal*, vol. 49, pp. 794–808, 2007.

[14] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *ACM SIGMOD international conference on Management of data*, pp. 207–216, 1993.

[15] S. E. Middleton, N. R. Shadbolt, and D. C. De Roure, "Ontological user profiling in recommender systems," in *ACM Transactions on Information Systems*, vol. 22, pp. 54–88, 2004.

[16] A. Sieg, B. Mobasher, and R. Burke, "Learning ontology-based user profiles: A semantic approach to personalized web search," in *IEEE Intelligent Informatics Bulletin*, vol. 8, pp. 7–18, 2007.

[17] X. Cai, M. Bain, A. Krzywicki, W. Wobcke, Y. S. Kim, P. Compton, and A. Mahidadia, "Collaborative filtering for people to people recommendation in social networks," in *23rd Australian Joint Conference on Artificial Intelligence*, (Adelaide, South Australia), pp. 476–485, 2010.

[18] L. Pizzato, T. Rej, J. Akehurst, I. Koprinska, K. Yacef, and J. Kay, "Recommending people to people: the nature of reciprocal recommenders with a case study in online dating," *User Modeling and User-Adapted Interaction*, pp. 1–42, 2012.

[19] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[20] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," in *IEEE Computer Society*, vol. 42, pp. 30–37, 2009.

[21] J. Chen, W. Geyer, C. Dugan, M. Muller, and I. Guy, "Make new friends, but keep the old: recommending people on social networking sites," in *27th International Conference on Human Factors in Computing Systems*, (Boston, MA, USA), 2009.

[22] J. L. Herlocker, J. A. Konstan, and J. Riedl, "Explaining collaborative filtering recommendations," in *the 2000 ACM Conference on Computer Supported Cooperative Work*, (Philadelphia, Pennsylvania, United States), pp. 241–250, 2000.

[23] P. Melville and V. Sindhwani, "Recommender systems," in *Encyclopedia of Machine Learning*, pp. 829–838, 2010.

[24] S. Kutty, L. Chen, and R. Nayak, "A people-to-people recommendation system using tensor space models," in *27th Annual ACM Symposium on Applied Computing*, (Trento, Italy), pp. 187–192, 2012.

[25] T. G. Kolda and B. W. Bader, “Tensor decompositions and applications,” *SIAM REVIEW*, vol. 51, no. 3, pp. 455–500, 2009.

[26] Z. Yin, M. Gupta, T. Weninger, and J. Han, “A unified framework for link recommendation using random walks,” in *International Conference on Advances in Social Networks Analysis and Mining*, (Odense, Denmark), 2010.

[27] W. H. Hsu, A. L. King, M. S. R. Paradesi, T. Pydimarri, and T. Weninger, “Collaborative and structural recommendation of friends using weblog-based social network analysis,” in *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, pp. 55–60, 2006.

[28] R. Nayak, “Utilizing past relations and user similarities in a social matching system,” in *15th Pacific-Asia Conference on Knowledge Discovery and Data Mining*, (Shenzhen, China), pp. 99–110, 2011.

[29] R. Rawat, R. Nayak, and Y. Li, “Identifying interests of web users for effective recommendations,” in *International Journal of Innovation, Management and Technology*, vol. 2, January 2011.