SUP – A Service Oriented Framework for Semantic User Profile Extraction and Representation

Alexandru-Lucian Gînscă, Emanuela Boroş, Sabin-Corneliu Buraga, Lenuța Alboaie
Faculty of Computer Science
"Al. I. Cuza" University
Iași, Romania
{lucian.ginsca, emanuela.boros, busaco, adria}@info.uaic.ro

***Abstract*—We present in this paper a novel semantic model for a user's profile based on multiple social network accounts and services that compute the user's influence on these networks. Also, a Web application was developed in order to resolve the tasks of data acquisition, to offer an accessible interface for accessing the knowledgebase and allow the user to have his/her social graph semantically modeled. In the modeling process we make use of common vocabularies, but we also create our own model especially for information regarding influence. The semantic similarity between the main topics associated with two users is computed by our implemented Web service.**

Keywords- user profile modeling; social computing; semantic Web; services

I. INTRODUCTION

An increasing number of users want to do more than simply query or extract statistics about their personal data, looking instead to process their own data, visualize and analyze it, in a better way that extends the modularity of applications on the Web. By collecting the information that social networks, such as Twitter and Facebook, offer, our framework for *Semantic User Profile Extraction and Representation* (SUP) builds a large collection of data about people and their interests in order to create a social profile for a given user. A profile is semantically modeled and exposed via the current Web standards.

SUP also provides means for estimating a user's reputation based on multiple criteria, using social scoring services such as Klout [1] and PeerIndex[1]. A user's social graph can be visualized and also queried using a SPARQL service endpoint. The core principles behind this application are built around the visually attractive method of seeing a user's semantic profile.

Regarding the architectural features of the application, SUP extends a standard CRUD (create, read, update and delete) architecture into a web application. The presentation and data model logic are properly separated and the storage is handled by Virtuoso triple store. SUP also provides end-to-end consistency in data (JSON/ JavaScript), smooth communication and interaction between client and server, clean encapsulated interfaces and lightweight RESTful (conforming to the Representational state transfer constraints) Web services.

SUP became a project that is easily implemented, leveraging existing vocabularies for semantic augmentation of data and significantly improving the functionality of interactive data visualization. Users have more control on their semantic data by visualizing or querying it, also with a feedback regarding their influence score on social media networks.

In section II, other similar approaches are depicted. Section III presents the general architecture of the SUP framework. We describe our global model and our influence model in section IV, while details of implementation are offered in section V. Finally, in sections VI and VII we give actual usage examples of SUP and we discuss the main problems that may occur and, in section VIII, we draw several conclusions and present further directions of research.

II. RELATED WORK

We identify two directions of work that share similar aspects with our approach. The first one refers to abstract models for defining user interactions in social networks while the second one concerns actual implementations of semantic models for a certain social network.

In [2], the authors address the topic of modeling social networks through RDF at a high level of conceptualization. They first give a formal definition for a social network data model as a triple consisting of nodes, edges and labels, after which they provide definitions for a social network triple representation and a SPARQL query. They also provide research results regarding query language semantics. The authors do not offer examples of known vocabularies that can be used in an implementation of their model.

A similar approach to ours regarding linking user profiles from multiple sources is described in [3] , where the authors worked on merging profiles and identities used on different sites modeled using the FOAF [2] (Friend Of A Friend) vocabulary. Although they use data from 11 sources, their main goal is to analyze the number of connections that can be made between profiles of the same user from different sites, whereas our application focuses not only on a user's profile, but also on the interactions between users, user

[1] http://dev.peerindex.com/

[2] http://xmlns.com/foaf/spec/

generated content, user influence scores and targets a broader audience by offering accessible methods of querying and visualizing the information stored accordingly to our model.

A framework that exploits the RDF representations of social networks is described in [4]. The authors provide formal definitions in SPARQL of social network analysis operators parameterized by the ontologies underlying these representations. Also, they present an ontology of social network analysis characteristics used to annotate social networks.

Regarding delivering social network content in a RDF format, we give as a reference SemanticTweet[3], a web service that generates a RDF document using FOAF from a user's list of Twitter friends and followers. Using the public Twitter API to collect public data only, this service does not require user authentication.

Another application that targets Twitter data is ShreddedTweet[4], which aims to model the latest tweets returned by a keyword search. The results are modeled using the RDF/XML format. This application uses several well-known vocabularies, including FOAF, SIOC[5] and DCMI[6].

III. ARCHITECTURE

Our general approach, as described in the introduction, focuses on creating a system that requires gathering and processing a broad range of data from different people – in this case, users from social media networks – to optimally serve individuals, according with the current trends and principles of linked data.

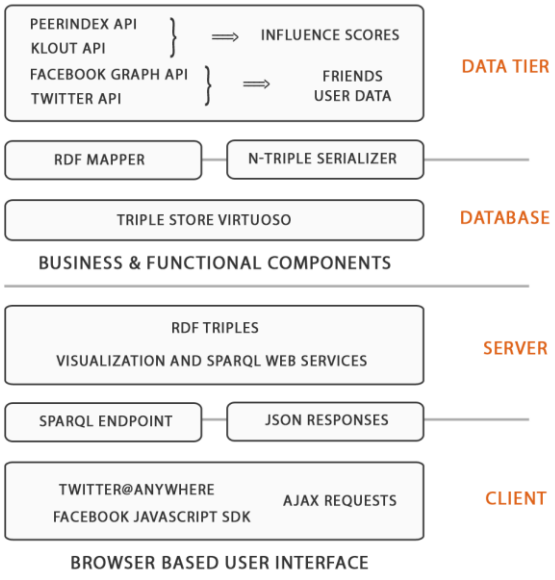

Figure 1. SUP general architecture

This type of system requires methods for data processing and a scalable infrastructure that can handle high data rates. In order to obtain this, the architecture of SUP has been designed following a three-tier approach such as a light model-view-controller as seen in Figure 1, with the advantages of modular web services that provide an extensible and interoperable framework for application-to-application communication, thanks to the use of universal data formats and protocols, a back-end server that handles data processing and an appropriate data storage engine.

IV. MODEL

A. Global model

Besides our own vocabularies developed with the purpose of modeling influence information, we mainly use the foaf and sioc vocabularies.

TABLE I. VOCABULARY TERMS

SIOC	FOAF	FOAF
sioc:user	foaf:Agent	foaf:birthdate
sioc:follows	foaf:onlineAcount	foaf:firstName
sioc:userAcount	foaf:knows	foaf:lastName
sioc:avatar	foaf:nick	foaf:homepage
sioc:creatorOf	foaf:img	
sioc:post	foaf:mbox	

In Figure 2, we present a small sample containing some attributes used to describe a user in our model. The prefix "suser" addresses one of our own vocabularies used to semantically describe and to uniquely identify a user and his multiple online accounts.

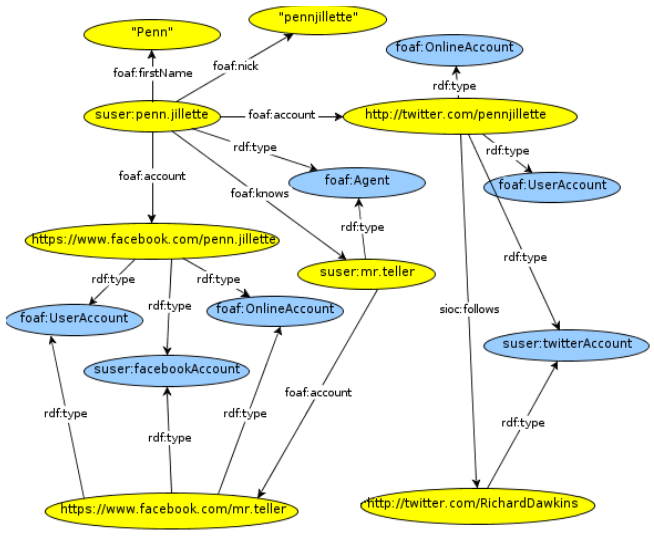

Figure 2. Global model example

For example, let us consider "Penn Jillette", a user that has two social network accounts (on Facebook and Twitter), follows Richard Dawkins on Twitter and is friend with Mr. Teller on Facebook. A fragment of the corresponding RDF model is:

```
suser:penn.jillette     a               foaf:Agent;
                        foaf:firstName  "Penn";
                        foaf:nick       "pennjillette";
```

[3] http://semantictweet.com/
[4] http://shreddedtweet.org/
[5] http://rdfs.org/sioc/spec/
[6] http://dublincore.org/documents/dcmi-terms/

	foaf:account	http://twitter.com/pennjillette;
	foaf:account	https://facebook.com/penn.jillette.
	foaf:knows	suser:mr.teller
suser:mr.teller	a	foaf:Agent;
	foaf:account	https://facebook.com/mr.teller.
http://twitter.com/pennjillette	a	sioc:UserAccount;
	a	suser:twitterAccount;
	sioc:follows	http://twitter.com/RichardDawkins.
http://twitter.com/RichardDawkongs	a	suser:twitterAccount.

B. *Influence model*

We are interested in discovering features related to a user's influence on a certain social network, the influence of his/her friend(s) and creating a model using RDFS and OWL for these influence components. We use two services that are known for their work in social network influence analysis, Klout and PeerIndex. Next, we give descriptions of the different scores exposed by these services that we encapsulated in our influence model.

1) Klout: We included in our model, besides the Klout score, other influence related concepts that Klout offers. Next, we present the four influence scores that Klout provides.

- Klout Score: is the measurement of the user's overall online influence. The score ranges from 1 to 100 with higher scores representing a wider and stronger sphere of influence.
- Amplification Probability: Klout describes it as "the likelihood that your content will be acted upon. The ability to create content that compels others to respond and high-velocity content that spreads into networks beyond your own is a key component of influence."
- Network: the network effect that an author has and it is a measure of the influence of the people the author is reaching.
- True Reach: measures how many people an author influences.

2) PeerIndex: Although PeerIndex relies on fewer data sources than Klout, we desired to have an alternative to the klout score. Next, we will present descriptions of the four influence scores given by PeerIndex.

- PeerIndex score: a user's overall PeerIndex score is a relative measure of his/her online authority. The PeerIndex Score reflects the impact of his/her online activities.
- Authority Score: is the measure of trust calculating how much others rely on the user's recommendations and opinion in general and on particular topics.
- Audience Score: is a normalized indication of the user's reach taking into account the relative size of his/her audience to the size of the audiences of others.
- Activity Score: is the measure of how much the user does that is related to the topic communitie(s) he/she is part of.

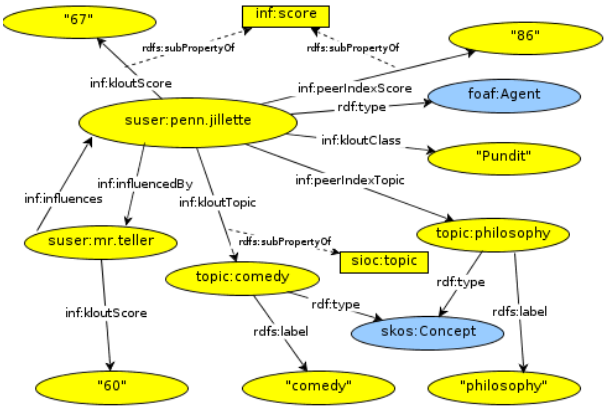

Figure 3. Influence model example

In Figure 3 we give an example of influence data from our model. We rely mostly on our vocabularies created to describes concepts form the influence domain ("inf" prefix) and topics related with influence ("topic" prefix).

C. *Topic semantic similarity*

A user has associated different topics drawn from multiple sources which give an overview image of his mostly discussed concepts or his interests. In our current implementation, topics are gathered from the Klout and PeerIndex services. While PeerIndex returns a straight-forward list of topics for a certain user, Klout has a particular understanding of the concept of "topic". Next, we will present Klout's method of finding topics.

Klout topics are gathered from the Twitter stream and in some cases they seem to have nothing to do with what the tweets about. Klout looks for specific keywords in the user's tweets that received a certain amount of attention, such as numerous replies to the user's tweet or retweets of that tweet. If the user replies to someone's tweet and the response generated lots of interest, then Klout will look back to the original tweet for keywords. Once the keywords that draw influence are obtained, Klout uses a dictionary to identify relevant terms. More details regarding this dictionary and how the terms are correlated seem not to be available for public disclosure. Klout then compares the user's influence on these terms to see if you he is generating significant influence within their network. If Klout determines if a user has influence on a specific term, then that term will appear on his/her list of topics. This approach functions on English language only. Unfortunately, Klout does not provide support for other languages.

For computing the semantic similarity between two terms, we use three WordNet [7] semantic similarity algorithms, Wu and Palmer [5], Resnik [6] and Lin [8]. Next, we give more details about these measures and present results computed on five Klout topics extracted from our knowledgebase in Tables II, III and IV.

1) The Wu & Palmer measure: The Wu & Palmer measure [5] calculates semantic similarity by considering the

[7] http://wordnet.princeton.edu/

depths of the two synsets (words grouped into sets of synonyms) in the WordNet taxonomies, along with the depth of the least common subsumer. The formula is as follows:

$$wuPalmerScore(t_1, t_2) = \frac{2 * depth(lcs(s_1, s_2))}{depth(s_1) + depth(s_2)}$$

s_1: the synset of the first term;
s_2: the synset of the second term;
$lcs(s_1, s_2)$: the synset of the least common subsumer.

TABLE II. WU AND PALMER MEASURE RESULTS

Terms	internet	design	web	education	philosophy
internet	1.0	0.631	0.909	0.222	0.21
design	0.631	1.0	0.75	0.8	0.75
web	0.909	0.75	1.0	0.461	0.428
education	0.222	0.8	0.8	1.0	0.8
philosophy	0.21	0.75	0.428	0.8	1.0

2) The Resnik measure: This measure also relies on the idea of a least common subsumer (LCS), the most specific concept that is a shared ancestor of the two concepts. [6]

The Resnik [7] measure simply uses the Information Content of the LCS as the similarity value:

$$resScore(t_1,t_2) = IC(lcs(t_1,t_2))$$

$lcs(t_1,t_2)$: the least common subsumer.

$$IC(t) = -\log\left(\frac{freq(t)}{maxFreq}\right)$$

freq(t): the frequency of term t in a corpus;
maxFreq: the maximum frequency of a term from the same corpus.

The Resnik measure is considered somewhat coarse, since many different pairs of concepts may share the same LCS. However, it is less likely to suffer from zero counts (and resulting undefined values) since in general the LCS of two concepts will not be a very specific concept.

TABLE III. RESNIK MEASURE RESULTS

Terms	internet	design	web	education	philosophy
internet	10.37	0.631	10.37	0.0	0.0
design	2.49	11.76	2.49	3.39	3.39
web	10.37	2.49	11.76	2.87	0.77
education	0.0	3.39	2.87	10.66	3.39
philosophy	0.0	3.39	0.77	3.39	11.76

3) The Lin measure: The Lin measure [8] augments the information content of the LCS with the sum of the information content of concepts A and B themselves. The Lin measure scales the information content of the LCS by this sum.

$$linScore(t_1, t_2) = \frac{2 * resScore(t_1, t_2)}{IC(t_1) + IC(t_2)}$$

TABLE IV. LIN MEASURE RESULTS

Terms	internet	design	web	education	philosophy
internet	1.0	0.28	0.32	0.0	0.0
design	0.28	1.0	27	0.46	0.48
web	0.32	0.27	1.0	0.09	0.09
education	0.0	0.46	0.09	1.0	0.46
philosophy	0.0	0.48	0.09	0.46	1.0

4) Topic set similarity: For computing the semantic similarity between the topics of interest of two users using one of the three measures described above, we first generate the stem of each term, using an open source implementation of the Porter Stemmer. The final similarity score is computed using a weighted average over the maximum score obtained by applying a semantic similarity measure on each combination of a term from the first user's topics set and one from the second user's topic set.

$$globalSim(T_1, T_2) = \frac{\sum_{t_1 \in T_1} \max(\text{sim}(t_1, t_2 \in T_2))}{|T_1|}$$

T1: first user's topics set;
T2: second user's topics set;
$sim(t_1, t_2)$: one of the Wu and Palmer, Resnik or Lin similarity measures.

From the three similarity methods described above, we have selected the Lin measure to be used in the global similarity computation for two sets of topics. The main arguments sustaining our choice are the easy interpretation of the result and the higher penalty for terms with a low degree of similarity. We expose this similarity computation to the end user through a RESTful web service.

V. IMPLEMENTATION

The architecture combines different technologies, such as Javascript, jQuery, and Ajax on the client side and Java, on the server side. The presentation layer is Javascript-driven with Ajax for pushing information while the business and data layers are developed with Java EE technologies. Following this approach, SUP takes the best of both worlds: the dynamic, personalized user experience we expect of immersive Web applications and the simple, scalable architecture we expect from RESTful applications. Below we provide further details about the three specific tiers.

A. Presentation Layer

This layer has been developed using the single Web page design pattern. The parent page has the primary purpose of satisfying the common user of the application that is looking for a creative way of visualizing personal data and the child page regards the specialized users that are looking for a representational state of SPARQL queries. The communication between the two higher tiers is carried on through Ajax, with the client submitting requests to the logic tier and receiving back JSON data representing the content of the response, which is then parsed and used to activate proper interaction in the user interface. The presentation

implies data received from server represented in two ways: one for the social graph and the other one for the raw result for the SPARQL queries result, which comes in xml format.

The business and functional components of the application require minimal information from the main social networks that are used as data providers. These are completed using Twitter@Anywhere [8] and Facebook Javascript SDK[9]. Twitter @Anywhere is an easy-to-deploy solution for bringing the Twitter communication platform to a web page. It is used to build the integration with "Connect to Twitter." The Facebook JavaScript SDK provides simple client-side functionality for accessing Facebook's API calls. The social plugins are used to obtain an access token for the communication with Facebook.

The creation and population of the visualizations for every semantic profile is done with the use of Protovis[10]. The common forms of visualization are the social graph and the timeline. These are provided with JSON results after the RESTful services are also provided with query-specific results.

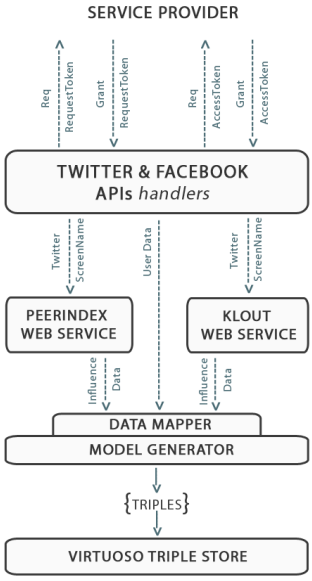

Figure 4. Data acquisition workflow

B. Business Logic Layer

The business logic of the application is implemented through a collection of Java RESTful Web Services which are deployed on Tomcat 6 server. The services are used for sending further SPARQL queries and receiving from the Virtuoso triple store specific responses. These are processed and made more appealing for the user interface to display them. This tier has the great property of using REST web services which are lightweight (no complex markups) with human readable results and easy to build – no toolkits required. We take advantage of using them for a CRUD approach of getting our need data for creating semantic profiles.

C. Data Layer

The data tier is mainly represented by a component for accessing and managing the RDF/OWL model. This component queries and manages RDF triples RDF triples with the OpenLink Software's Virtuoso[11] which is a database server that can also store (and, as part of its original specialty, serve as an efficient interface to databases of) relational data and XML. The primary data which consists of details of users' profiles from different social networks and different scores of their influence in online medium is gathered using implementations of common social networks (Twitter and Facebook) and social scoring applications (Klout and PeerIndex).

For Klout and PeerIndex, we created our personal API's implementations. They are the main providers for influence scoring computing. For Twitter, we used *Twitter4J*[12] which is a library for easily integration of the Twitter service with built-in OAuth support and zero dependency. In the case of Facebook, we chose *RestFB*[13] which is a simple and flexible Facebook Graph API and Old REST API client written in Java.

VI. PROOF OF CONCEPT

We will present in this section two actual uses of SUP concerning a user's access to the knowledgebase. At this point, the data acquisition process for the user "penn.jillette" is completed and personal information, friendship relations from his Twitter and Facebook accounts, and influence data from Klout and PeerIndex were extracted. A sample of how this data appears in the model is displayed in section IV.

In Figure 5, we offer a Protovis powered graph representation of the Twitter accounts that "penn.jillette" follows. The size of the avatar images is directly proportional with each user's PeerIndex score.

Figure 5. User graph

In the second case, we present the situation in which a user wants to query our knowledgebase. He wants to find all the people that the SUP user "penn.jillette" follows on Twitter and how influent are them according to the Klout

[8] https://dev.twitter.com/docs/anywhere/welcome
[9] https://developers.facebook.com/docs/reference/javascript/
[10] http://mbostock.github.com/protovis/
[11] http://docs.openlinksw.com/
[12] http://twitter4j.org/en/index.html
[13] http://restfb.com/

score. The SPARQL query that corresponds to this request and the first three results returned are shown in Figure 6.

Figure 6. SPARQL query response

VII. DISCUSSION

An important issue is the evaluation of our system's capabilities. In particular, the questions that are highlighted are:

- Has the system the ability to support a considerable number of users?
- What makes a profile accurate and when?
- Has the user the privacy that he/she seeks for?
- Does this representation of users' profiles have advantages?

The backbone of the system is the triple store, Openlink Virtuoso, which has been proven to have a successful scalability performance for queries.

Our solution on security and user's privacy is enriched by the requirement of access tokens from the APIs. Some issues still remain to be resolved, such as data privacy, private aggregation communities, and building personalized views and aggregation services for public updates.

Finally, there are a considerable number of advantages for creating semantic profiles and exposing data in RDF/XML. In order to maintain these advantages, accurate information about the user's data must be collected, modeled and also updated incrementally based on user's behavior. The phase that follows the modeling of the RDF-based user profiles and the computation of the semantic similarities between topics is the aggregation of the distributed user profiles.

When merging user profiles it is necessary to avoid duplicate statements (and this is done automatically by a triple-store during the insertion of the statements). Furthermore, in the case of the different accounts for every user, if the same user is present on two different social networks it is necessary to: represent the user only once by creating a SUP user with multiple accounts, recalculate its influence, and update the provenance of the profile account keeping track of the source. As regards the provenance of a social media account, as shown in section IV, we use the properties *hasTwitterAccount* and *hasFacebookAccount* to state that the user was originated by a specific media.

That being assured, the system allows an easy exchange of user profiles within a community, enrich user profiles with visualization techniques or even contribute to a connected, distributed social network that can feed a variety of applications. Moreover, based on the semantic similarities, users can also see how their influence on different topics overlaps with their friends and followers.

VIII. CONCLUSION AND FUTURE WORK

We described a model for social network user accounts and detailed the process of data acquisition, designing and implementing a framework capable of leveraging the acquired knowledgebase. Our work is justified by the need of a thorough semantic model for user details, content and interaction obtained from multiple social networks not only on an abstract level, but actually delivering a user friendly product for accessing and visualizing the data.

Semantic modeling deserves necessary involvement from out team and it is important to continue investigating new means for a more accurate influence computation. We will also focus on improving the semantic model and a larger collection of triples would be needed. We plan to explore further methods of visualizing the information extracted from our model, especially to offer an adaptive solution for generating different types of graphs based on the content of the SPARQL query and by this exploring the semantics of such a query. We will also focus on improving the semantic model.

REFERENCES

[1] I. Anger and C. Kittl, "Measuring Influence on Twitter", in Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies, 2011.

[2] M. San Martin and C. Gutierrez, "Representing, Querying and Transforming Social Networks with RDF / SPARQL", in ESWC, 2009.

[3] L. Goldbeck and M. Rothstein, "Linking social Networks on the web with FOAF", in Proceedings of the twenty-third conference on artificial intelligence, AAA, 2008.

[4] G. Erétéo, M. Buffa., F. Gandon and O. Corby, "Analysis of a Real Online Social Network Using Semantic Web Frameworks", Network, 5823, 180-195. Springer, 2004.

[5] Wu and M. Palmer, "Verb semantics and lexical selection", in 32nd Annual Meeting of the Association for Computational Linguistics, pages 133–138, Las Cruces, New Mexico, 2004.

[6] T. Pedersen, P. Siddharth and J. Michelizzi, "Wordnet::similarity — measuring the relatedness of concepts", in Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04). AAAI Press, Cambridge, MA, pages 1024–1025, 2004.

[7] P. Resnik, "Using information content to evaluate semantic similarity", in Proceedings of the 14th International Joint Conference on Artificial Inteligence, IJCAI-9, 1995.

[8] D. Lin, "An information-theoretic definition of similarity", in Proceedings of the International Conference on Machine Learning, Madison, August 1998.