# Managing Presence and Policies in Social Network Dependent Systems

Ahmed Hasswa
Telecommunications Research Lab, School of Computing
Queen's University
Kingston, Ontario, Canada, K7L 3N6
hasswa@cs.queensu.ca

Hossam Hassanein
Telecommunications Research Lab, School of Computing
Queen's University
Kingston, Ontario, Canada, K7L 3N6
hossam@cs.queensu.ca

***Abstract* — Social Networks have recently experienced a significant increase in popularity and are now an integral part of millions of people's daily lives. Through these Social Networks, users create profiles, build relationships, and join groups forming intermingled sets and communities. There is a wealth of information within Social Networks, which if exploited properly and combined with rules and policies, can lead to a whole new level of smart contextual services. A mechanism is therefore needed to extract data from heterogeneous Social Networks, link profiles across different networks and aggregate the data obtained. We design a Presence and Policies Server that manages the information exchange between Social Networks, services and the environment and passes along the relevant information and rules to different entities. The Presence and Policies server is capable of querying, importing and aggregating data from across multiple Social Networks and services and then converting that data into standardized semantic information that can be interpreted and translated into meaningful information by other users and services.**

***Keywords-Presence, Policies, Social Networks, SIP, IMS***

## I. INTRODUCTION

Social Networks have recently experienced a significant increase in popularity and evolved to become an integral part of millions of people's daily lives. Social Networking websites such as Facebook [1], MySpace [2], Bebo [3] and Twitter [4] have captured the attention of millions of users and received billions of dollars in investor funds. Facebook for instance has more than 350 million active users [5], is the second most visited website in the world (and the second most visited website in Canada too) [6], has over 55,000 regional, work-related, collegiate, and high school networks, and is used by 85% of U.S. university students [5]. Social networks in general are used for a variety of purposes ranging from staying in touch with friends, sharing information, planning events to meeting new people with similar interests and hobbies. Through Social Networks, users are able to create a social profile that includes a wealth of information ranging from hobbies, interests, favourites, education, and career to emotions, thoughts and feelings. Furthermore, Social Network users build relationships amongst each other and join groups forming intermingled sets and communities.

There is a wealth of information within social networks, which if exploited properly, can lead to a whole new level of mobile social environments and targeted contextual services. In order to make use of the data provided by social networks in a mobile environment it is crucial to be able to somehow identify users and their personal profiles and then import their social content into a system through some sort of standardized format. By having such a reliable portability module, any system can utilize this information to provide smarter and more adaptive services. However, not all users join the same social networking websites. There is a variety of different Social Networks and users will join different ones based on their age groups, or simply interests and preferences. Furthermore, it is not out of the ordinary for a user to have multiple profiles on multiple social networks. A mechanism is therefore necessary to extract data from heterogeneous Social Networks and also link profiles across different networks and aggregate the data obtained. Furthermore, identifying social ties between different people and objects and classifying them into groups can create better-targeted services to cater to that group.

A well defined Social data portability module should therefore be able to query, import and aggregate data from across multiple Social Networking websites and then convert that data into standardized semantic information that can be further interpreted and converted into meaningful information by other entities within a system. Nonetheless, to create such modules there are significant challenges. The main problem with information extraction from different heterogeneous social networking sources is the availability of different data formats. Each social network exports its data in a different style and format. It is impossible to simply compare diverse un-matching data formats without having some common vocabulary. There must be some kind of mediator format that allows easier interoperability, integration and computations to be performed.

On top of representing social network user profiles in a standardized format, it is essential for all other entities within the system to interact via a unified ontology. This applies to preferences and policies set by the users, including privacy ones. The environment also creates rules defining which services and activities are allowed and which are not within the space. Services need to also communicate with users and

the environment and exchange capabilities, features, and more. It is therefore necessary for users, environments and services to share a common communication language or ontology and have it managed through a policies and presence server. The server filters out information and shares only the information that other entities need and are allowed to access.

In this paper we propose a new presence and policies management system and semantic ontology for social networks. We design a Presence and Policies Server (PPS) that manages the information exchange between social networks, users, services and the environment and passes along the relevant information and rules to different entities. PPS is capable of querying, importing and aggregating data from across multiple Social Networks and services and then converts that data into standardized semantic information that can be further interpreted and converted into meaningful information by other users and services.

The rest of this paper is organized as follows. Section II discusses related work and some of the major problems with current solutions. Section III then presents the overall architecture, shows how the Presence and Policies Server will function and describes event notification within the system. Section IV describes the social context representation through a standardized format. Section V describes a complete social ontology for exchanging data within the system. This is followed by the conclusion and possible future research directions in Section VI follow this.

## II. BACKGROUND & RELATED WORK

In this section we present the criteria needed to create a successful Social-Network based system. We then analyze some related work and discuss which criteria they meet and how they can be improved. The requirements for creating a social network dependent system are different from those of any other system because of the heterogeneity of social networks and the large amount of services they are expected to offer. Here are some of the criteria we propose for creating a truly reliable and practical social network based system:

**Social Network Independent and non-standardized** – There are hundreds of social networks and each have their own users and features. Allowing the different social networks to connect to the system is necessary. In order to do so, there must be a common language that all social profiles, no matter which social network they are a part of, can be presented in.

**Networking Technology Independent** – There is a multitude of heterogeneous networks each offering different coverage ranges and speeds. All these networks are here to stay and there is a common belief that the future network is a heterogeneous network that allows different networks to co-exist and benefits from the advantages of each. Ensuring that the system is accessible via any kind of network technology is therefore necessary.

**Service Independent** – In the past, services were tightly coupled with system but as we shift from a communication centric world to a service oriented one, we realize that different users have different requirements and preferences. Creating a handful of services can no longer satisfy all users. Third part developers should be able to create services and easily plug it into a social network system.

**Device-independent** – There is no killer device. Just like everything else in life, people will always have different tastes and preferences. Platforms will always evolve and operating systems will change. Creating a system that is device specific will only lead to its demise and failure. It is important that users can access their social network system and services from whatever device they are using.

There have been several projects that focus on mobile systems for social networks. However, most of these projects have focused on allowing users to access Social Networks through their mobile devices [8]. Other projects [9] [10] have focused on creating new local social groups and providing services that allow text messaging and exchange of data such as photos and music within that local group. These groups are often very specific in purpose and no interaction is made with global Social Networks. Beyrem Chelly et al. [11] present an algorithm aiming at connecting members of a community equipped with mobile devices based on their location to create cooperative groups. However they only focus on network layer connectivity and barely establish any kind of social networking or interaction between the users nor do they use any semantic definitions to ensure generality and extensibility. Similarly, Yao-Jen Chang et al. [12] propose an interactive multimedia location-based application that provides supportive social services in a work field. The system enables social workers, their colleagues, and other participating professionals to keep in touch with each other and stay informed and organized as a single mobile community. Although this may prove to be an invaluable application in the workplace, there is no mention of utilizing currently available Social Networks or representing social contexts. Furthermore, the system seems to be limited by many factors including networking interfaces.

Marco von Arb et al. [13] take Social Networking one step further and introduce a decentralized system, VENETA, that is able to explore the social neighbourhood of a user by detecting friends of friends. They develop a distributed algorithm that is able to locate friends of friends that are in the user's current physical proximity. They also address privacy related issues. Although VENETA tackles some of the issues relating to locating mobile friends in the region, contextual information is not utilized and thus the system lacks any context-awareness. Furthermore, VENETA uses a local relational database to share the information and does not try presenting the data in a format that can be interpreted across multiple platforms. No information is mentioned as to how the system can tackle these issues to provide a feasible service. Google had an attempt at penetrating the Mobile Social Networks market by purchasing DodgeBall [14], a Social Networking Service founded by Dennis Crowley and

Alex Rainert [15]. Google however failed to realize that in order to succeed it is necessary to interface with other Social Networks rather than build a mobile Social Networking platform from scratch. People were reluctant to provide new profile information and in many cases never found any of their friends on DodgeBall.

Emiliano Miluzzo et al. [16] design a personal sensing system called CenceMe that enables members of social networks to share their sensing presence with their friends in a secure manner. Sensing presence captures a user's status in terms of their activity (e.g., sitting, walking, meeting friends), disposition (e.g., happy, sad, doing OK), habits (e.g., at the gym, coffee shop today, at work) and surroundings (e.g., noisy, hot, bright, high ozone). CenceMe then injects sensing presence into popular social networking websites such as Facebook, MySpace, and Instant Messaging services such as Skype. These improved user statuses provide more information about a user to their online friends but it does not make use of the contextual information to enhance the mobile user's real-world experience. It also has a limited set of statuses and does not allow for the customizability that most users prefer.

So far all of the above-mentioned research projects have failed to realize the true potential behind Mobile Social Networking. It is not about using the contextual information to improve your online status but rather enhance you real-world presence and experience. The only two research projects which realize this vision to a certain extent are Serendipity [17] and WhozThat [18]. Serendipity is based on the belief that the mobile phone market is at a critical tipping point, where functionality will shift from the traditional telephone paradigm to a much broader socio-centric perspective. Serendipity therefore utilized contextual social information to provide services such as dating or knowledge management systems. It does so by facilitating interactions between physically proximate people through a centralized server. Unfortunately, Serendipity does not harness the wealth of information available in Social Networks but rather creates its own user profile database. This database lacks a generic standardized data format. It is therefore very limited in scope and functionality and cannot be extended to interact with any Social Network or deliver any kind of service.

One of the most promising Context-aware systems that reply on Mobile Social Networks is WhozThat. WhozThat harnesses mobile and wireless technology to help answer the age-old human social question "Who's that?" It builds a local wireless networking infrastructure using mobile devices to share each individual's social networking ID, then connects to the Internet via wireless to look up the advertised identities. It then consults an online social network with the identity to import the relevant social context into the local context to enrich the local human interaction. WhozThat scheme provides major improvements context-aware mobile social networks. However it does not meet the four criteria mentioned earlier in this Section. WhozThat relies on the Social ID of a certain network as identification for its system. This makes it limited in its ability to encompass multiple heterogeneous social networks. Although the authors mention the possibility of creating a gateway between the Social Network and WhozThat, there are no details on how such a gateway could be designed. WhozThat deploys a specific Bluetooth component to exchange information between the mobile device and the context aware server. This limits the possibility of deploying WhozThat on any device due to specific technology constraints.

We address the limitations and weaknesses of the abovementioned schemes and propose a novel and truly smart adaptive ecosystem that utilizes contextual social information and enhances people's real-world experience through technology.

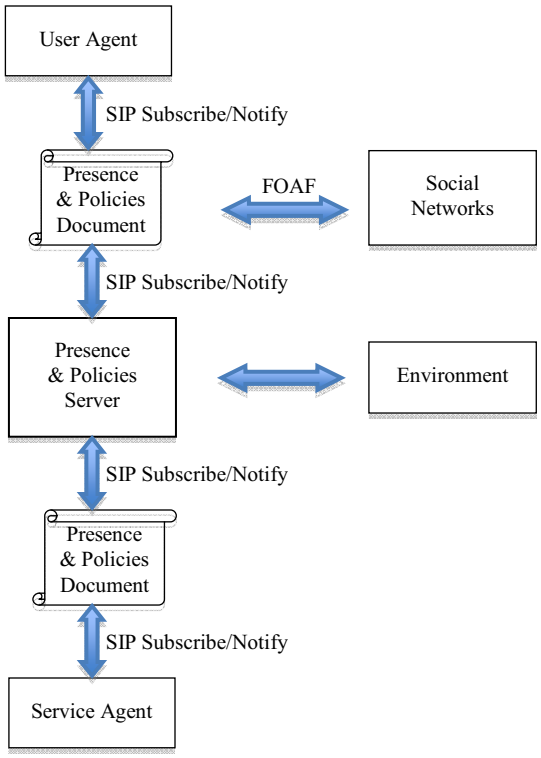


**Figure 1:** Social Network Presence and Policies Architecture

## III. ARCHITECTURE

In this section the overall architecture is presented. We show how different modules interact and how data is retrieved from social networks. Figure 1 shows the overall generic architecture of the system.

### *A. The Presence and Policies Server (PPS)*

Managing presence and policies is a major requirement for providing a truly omnipresent smart social network services platform. Presence services should be capable of providing a multitude of different information to all users within the space. Changes to existing states within the

system should be detected by the server and propagated to all users interested in that information. New services added to the space by third-party developers should be able to read, understand and interact with Presence and Policies Servers. Ultimately, services should be able to deliver presence information by understanding users' preferences and requirements.

Presence can be interpreted as a service that allows a user to be informed about the availability of another user. Users can set their status to online, offline, busy, away or anything they like. Users can also share information about their device and its multimedia and other capabilities. We extend this definition of presence to encompass more than just users: We treat services too as "users" that have states, features and capabilities. The same applies to environments. This enables users and services to easily interact with one another. Services can notify users of changes to their status and information and vice versa. Status refers to the current online state of a user and whether they are available for communication.

We develop a presence and policies framework that consists of a presence server and assigns different roles to different entities within the system. Users and services provide presence information and are therefore defined as presentities. The presence information supplied by users and services include properties and attributes, statuses, network capabilities, features, etc. Every device acts as a Presence User Agent and every service has a Presence Service Agent module. Presence User and Service Agents provide information about a user or service's current presence.

A presence user agent knows whether the user is logged in or not and has information about the user's registration status and what kinds of sessions they are currently running. If the user is engaged in any kind of communication the user agent is also aware of that. User agents can even have more information such as user's preferences, what kinds of communications the user is interested in or capable of accepting. And even when a user will be available or what the user is doing right now. All information gathered by User Agents is then forwarded to the Presence and Policies Server.

PPS gathers all information about a single user or service and creates a complete picture of that user. It does the same for all users and services within the space and hence creates a complete presence picture of the whole space. The server controls all the information flowing through the presence framework. It coordinates between presentities, presence user agents, service agents and watchers. PPS also acts as a proxy server for subscription requests.

## IV. Representing Social Context

For a social network dependent system to make use of users' social profiles, a mechanism for identifying different users, reading their profiles is necessary and then translating this information into a format that can be read by other entities within the system. It is better to use an agreed upon semantic representation format such as the Extensible Markup Language (XML). XML is a general-purpose specification for creating custom markup languages. In particular the Resource Description Framework (RDF) could be used. RDF is a family of World Wide Web Consortium (W3C) specifications, originally designed as a metadata data model, which has come to be used as a general method of modeling information through a variety of syntax formats. RDF can be combined with other technologies to describe people, objects, and the relationships between them in an abstract manner. Social networking websites can become rich data sources that are interpreted by the semantic RDF module. This information now becomes machine-readable and can be used to provide an enhanced view and rich set of information about a social network dependent system. How the standardized data is then interlinked and aggregated to form a meaningful picture is beyond the scope of this module. This module focuses on retrieving the information from a heterogeneity of social networks, identifying a single user's profile on multiple networks and translating the profile data into a unified language.

RDF defines a resource as any object that is uniquely identifiable by an Uniform Resource Identifier (URI). The properties associated with resources are identified by property-types, and property-types have corresponding values. Property-types express the relationships of values associated with resources. In RDF, values may be atomic in nature (text strings, numbers, etc.) or other resources, which in turn may have their own properties. A collection of these properties that refers to the same resource is called a description. At the core of RDF is a syntax-independent model for representing resources and their corresponding descriptions.

If additional descriptive information regarding the member were desired, e.g., the member's email address and affiliated network, an elaboration on the previous example would be required. In this case, descriptive information about Jane Smith is desired. Before descriptive properties can be expressed about the member Jane Smith, there needs to be a unique identifiable resource representing her.

The syntactic vocabulary to be used to describe the profile data extracted is FOAF (Friend of a Friend). FOAF is a decentralized semantic web technology, and has been designed to allow for integration of data across a variety of applications, web sites, services, and software systems. FOAF syntax is written in XML syntax, and adopts the conventions of RDF. By using FOAF syntax, modules within a social network dependent system will be able to exchange and represent social network data in a unified and machine readable manner since FOAF specifies ontology for representing people, objects and the relationships that they share. Using FOAF, people, their attributes and relationships can be modeled using these concepts:

- Person (Object): Each person is represented as a foaf:Person instance.

- Profile (Attributes): Each person has certain properties such as a name foaf:name, gender foaf:gender and profile picture foaf:img
- Relationships: Each person has friends and relationships which are represented using foaf:knows

The foaf:Person class represents people. An element is tagged as a foaf:Person if it is a user of the system. foaf:name and foaf:knows are examples of an attribute and a relationship a foaf:Person. These attributes and relationships can also be called properties. FOAF will enable other modules within a system to exchange a large number of properties such as name, gender, age, interests, email, etc. Figure 2 shows a simple FOAF representation for a social network user profile.

```
<rdf:RDF xml:lang="en">
    <foaf:Person>
        <foaf:name>Jane Smith</foaf:name>
     <foaf:nick>jsmith</foaf:nick>
     <foaf:interest dc:title="Comedy Movies"
   rdf:resource="http://www.imdb.com/Sections/Genre
s/Comedy"/>
     <foaf:knows>
          <foaf:Person>
          <foaf:name>Jessice Lee</foaf:name>
     </foaf:knows>
    </foaf:Person>
</rdf:RDF>
```

Figure 2: FOAF representation of a Social Network user profile

## V. PRESENCE AND POLICY ONTOLOGY

In this section we describe the full ontology that can be used by a presence and policies server to co-ordinate between different entities within a Social Network based system.

### A. Ontology Core

We use The Presence Information Data Format (PIDF) as a reference format for our extended presence description. PIDF was designed by the IETF as a guide to presence implementations and can be easily extended to create more complex ontology. This flexibility makes PIDF suitable for current and future presence systems. It is based on the Common Profile for Presence (CPP). It is therefore crucial that our presence ontology is compliant with PIDF and hence CPP. PIDF is also based on the Instant Messaging and Presence (IMPP) protocol. We therefore divide the ontology into presence data entries. Each entry consists of multiple tuples. In PIDF these statuses are limited to open and closed. We extend this to allow for custom defined values. By doing so, we can incorporate members' statuses from Social Networks.

Because of PIDF's extensible format and its standardized representation in XML, it provides a very reliable base on top of which many extensions have been built. However, none of these extensions are perfectly suited for our system, but many of them provide very useful ideas that our system shall inherit and then further extend and enhance. We describe these different extensions and explain why they are insufficient. We also explain which parts are inherited and extended by our ontology.

### B. Inheriting CIPID Elements

Contact Information in Presence Information Data Format (CIPID) is a basic PIDF extension. CIPID addresses the issue of PIDF's limit capability of representing contact information. PIDF has a single element for representing contact that is usually limited to the email address. CIPID defines a full set of contact elements that allow the user's contact information to be represented in more detail. These elements include display-name, card, homepage, icon, msg and sound.

The <card> element includes a URI pointing to a business card such as a vCard [14] format. The <display-name> element includes the name identifying the tuple or person that the presentity suggests should be shown by the watcher user interface. It is left to the watcher user interface design to choose whether to heed this suggestion or to use some other suitable string. The <homepage> element provides a URI pointing to general information about the tuple or person, typically a web home page. The <icon> element provides a URI pointing to an image (icon) representing the tuple or person. The watcher can use this information to represent the tuple or person in a graphical user interface. The <map> element provides a URI pointing to a map related to the tuple or person. The watcher can use this information to represent the tuple or person in a graphical user interface. The map may be an image, an HTML client-side image map, or a geographical information system (GIS) document. The <sound> element provides a URI pointing to a sound related to the tuple or person. The watcher may use the sound object, such as a midi or mp3 file, referenced by the URL to inform the watcher that the presentity has assumed the status open. The sound object might also be used to indicate how to pronounce the presentity's name.

Although these CIPID elements help include more information about the user, they are limited to basic contact information and some messaging-related features. They define contact information rather than logical properties such as the user's mood, and activities. Furthermore, they do not exploit the wealth of information present in social networks. Therefore CIPID alone is not sufficient for providing a comprehensive ontology, however it does provide some features that can be inherited by our ontology. The <card> element is unnecessary for our ontology since the objective of the presence module is not to exchange personal contact information but rather social context information. It is also preferable to avoid extracting personal contact information

for privacy reasons. <display-name> element is used by our ontology to provide the name identifying the person that the presentity suggests should be shown by the watcher user interface. <homepage> is unnecessary since our system will identify users through URIs and most Social network users do not have a local homepage. If they do, it does not add any useful context to the system. The <icon> element is used but is named <avatar> instead and it provides a URI pointing to an image (icon) representing the person. The watcher can use this information to represent the person in a graphical user interface. The avatar can be an picture of the user. Instead of a <map> element we shall have a more sophisticated <location> element. <sound> element is unnecessary and was created mainly for a more Instant Messaging oriented service. We have therefore inherited certain elements from CIPID but this leaves a lot to be desired in order to create a truly sophisticated ontology that can represent social context, provide policies and describe user preferences.

### C. *Inheriting RPID Elements*

The Rich Presence Information Data (RPID) format is a more advanced extension to PIDF that allows the representation of a large amount of properties in the presence description. This includes physical information such as properties of the surrounding environment as well as logical properties such activities and mood. Because of the larger amount of information managed by RPID as opposed to PIDF and CIPID, it consists of a more complex structure. We analyze each element and evaluate its usefulness for a social network dependent system.

The <place-is> element describes properties of the place the person is currently at. This offers the watcher an indication of what kind of communication is likely to be successful. Each major media type has its own set of attributes. This element allows an environment to be described. For instance, if the place requires silence, this is where such restriction would be conveyed. Unfortunately, this element does not enforce any rules or policies but simply describes the place. In a social network system so we will need to describe the environment or space within which a mobile client exists but this should be described in a more concrete manner than enables the enforcing of policies.

The <place-type> element describes the type of place the person is currently at. This offers the watcher an indication of what kind of communication is likely to be appropriate. Such kind of classification can be very useful in describing social network based environments. At the same time, each environment has its unique properties, requirements and policies and therefore it is necessary to allow for custom property definitions as well.

The <privacy> element indicates which types of communication third parties in the vicinity of the presentity are unlikely to be able to intercept accidentally or intentionally. This does not in any way describe the privacy properties of the electronic communication channel, e.g., properties of the encryption algorithm or the network protocol used. SIP session management renders this element useless in our system. It should be impossible for a third party to intercept a communication between two clients and it is necessary to implement the security and privacy features that ensure that.

The <relationship> element designates the type of relationship an alternate contact has with the presentity. This element is provided only if the tuple refers to somebody other than the presentity. Relationship values include "family", "friend", "associate" (e.g., for a colleague), "assistant", "supervisor", "self", and "unknown". The default is "self". So if say a friend is using the user's mobile device, the relationship value would be set to "friend". If a relationship is indicated, the URI in the <contact> element refers to the entity, such as the friend, that has a relationship to the presentity, not the presentity itself. For our system the services and applications are device independent. Each user can login to the space using a unique username. If someone borrows another person's device, they should log on using their ID. Hence, the user properties, preferences and policies follow the user around and are independent of device, network and location.

The <service-class> element designates the type of service offered. The possible entries include electronic, postal, courier, freight and in-person. This element is focuses on describing physical services and this is irrelevant to a social network kind of system and is therefore omitted.

The <sphere> element designates the current state and role that the person plays. It is a very useful property that allows the user to split their life into several "spheres" For example, it might describe whether the person is in a work mode, at home, or participating in activities related to some other organization such as university. CIPID does not define names for these spheres except for two common ones, "work" and "home", as well as "unknown". Sphere element would be a very powerful property to provide in our social network oriented system however the ability to define custom spheres would be necessary. By using spheres, a person would be able to easily turn on or off certain rules that depend on what groups of people should be made aware of the person's status. For example, if the person is a student at Queen's University, she might set the sphere to "queenscampus" and then have a rule set that allows other Queen's students to see her presence status. Or it can be more specific, e.g. "queenspsychology14" and have rules that allows Queen's Psychology students class of 2014 only to see her presence status. As soon as she switches her status to "res", "home", or some other sphere her fellow psychology students would lose access. Spheres offer services and

The <status-icon> element includes a URI pointing to an image (icon) representing the current status of the person or service. The watcher may use this information to represent the status in a graphical user interface. This element is unnecessary in our system since users can define custom statuses or present Social Network based statuses.

The <time-offset> element describes the number of minutes of offset from UTC at the person's current location. A positive number indicates that the local time-of-day is

ahead (i.e., east of) Universal Time, while a negative number indicates that the local time-of-day is behind (i.e., west of) Universal Time. We can utilize this basic element in our system to manage time within different spaces in different geographical locations.

The <user-input> displays the amount of time that has passed by since a user last used the service based on human user input, e.g., keyboard, pointing device, or voice. This element is not very useful for our system which is based on minimizing user-device interaction to provide a more seamless user experience.

### D. *Social Network Semantics Schema*

Now that we have discussed all the different elements and building blocks needed to describe the system, Figure 4 presents the completed schema. Please note that due to space limitations, some basic parts of the schema have been removed.

```
<xs:element name="uri" type="xs:anyURI"/>
<xs:element name="userid" type="xs:integer"/>
<xs:element name="userid" type="xs:integer"/>

<!—Sample of schema: />

<xs:element name="whocanfind">
  <xs:annotation>
    <xs:documentation>
      Designates the current state and role that
the person plays.
    </xs:documentation>
  </xs:annotation>
  <xs:complexType>
    <xs:choice minOccurs="0">
      <xs:element name="contacts" type="empty" />
      <xs:element name="sphere" type="empty" />
      <xs:element name="sphereandcontacts"
type="empty" />
      <xs:element name="everyone" type="empty" />
      <xs:element name="noone" type="empty" />
      <xs:any namespace="custom"
maxOccurs="unbounded" processContents="lax"/>
    </xs:choice>
  </xs:complexType>
</xs:element>

<!— More elements go here based on discussion in
Section V  />
```

Figure 3: Partial representation of social network schema.

We developed the social network based system to test our ontology. The Presence and Policies manager filters information and only forwards necessary information to other entities within the system. Users are able to connect to any kind of social network, make sure of any service and connect via any access technology. We therefore meet the four criteria mentioned in Section II. A screenshot of some of the functionalities is shown in Figure 6.


Figure 4: User Client

## VI. Conclusion and Future Work

We proposed a mechanism to extract data from heterogeneous Social Networks, link profiles across different networks and aggregate the data obtained. We also designed a Presence and Policies Server that manages the information

exchange between Social Networks, services and the environment and passes along the relevant information and rules to different entities. The Presence and Policies server is capable of querying, importing and aggregating data from across multiple Social Networks and services and then converting that data into standardized semantic information that can be interpreted and translated into meaningful information by other users and services.

## REFERENCES


[1] (2009, Nov.) Facebook. [Online]. http://facebook.com

[2] (2009, Nov.) MySpace. [Online]. http://myspace.com

[3] (2009, Nov.) Bebo. [Online]. http://bebo.com

[4] (2009, Nov.) Orkut. [Online]. http://twitter.com

[5] Facebook. (2009, Nov.) Facebook Statistics. [Online]. http://www.facebook.com/press/info.php?statistics

[6] A. I. Inc. (2009, Nov.) Facebook.com traffic details from Alexa.com. [Online]. http://www.alexa.com/data/details/traffic_details/facebook.com

[7] K. Seada and C. Perkins, "Social Networks: The Killer App for Wireless Ad Hoc Networks?," Nokia Research Centre NRC-TR-2006-010, 2006.

[8] Joe. (2008, Nov.) Facebook for iPhone. [Online]. http://www.facebook.com/apps/application.php?id=6628568379

[9] S. Counts and K. E. Fisher, "Mobile Social Networking: An Information Grounds Perspective," in *Proceedings of the 41st Annual Hawaii International Conference on System Sciences*, Hawaii, 2008, pp. 153-153.

[10] S. Farnham and P. Keyani, "Swarm: Hyper Awareness, Micro Coordination, and Smart Convergence through Mobile Group Text Messaging," in *Proceedings of the 39th Annual Hawaii International Conference on System Sciences, HICSS '06*, Hawaii, 2006, pp. 59a-59a.

[11] B. Chelly and N. Malouch, "Movement and Connectivity Algorithms for Location-Based Mobile Social Networks," in *IEEE International Conference on Wireless and Mobile Computing Networking and Communications, WIMOB '08*, Avignon, France, 2008, pp. 190-195.

[12] Y.-J. Chang, H.-H. Liu, L.-D. Chou, Y.-W. Chen, and H.-Y. Shin, " A General Architecture of Mobile Social Network Services," in *International Conference on Convergence Information Technology*, Gyeongju, 2007, pp. 151-156.

[13] M. von Arb, M. Bader, M. Kuhn, and R. Wattenhofer, "VENETA: Serverless Friend-of-Friend Detection in Mobile Social Networking," in *IEEE International Conference on Wireless and Mobile Computing Networking and Communications, WIMOB '08*, Avignon, France, 2008, pp. 184-189.

[14] G. Inc. (2009, Nov.) Dodgeball - Mobile Social Software. [Online]. http://dodgeball.com

[15] J. Dixit, " The artful- and mobile-dodger [location-based social networking]," *IEEE Spectrum*, vol. 42, no. 3, pp. 59-60, Mar. 2005.

[16] A. T. Campbell, et al., "CenceMe: Injecting Sensing Presence into Social Network Applications using Mobile Phones," in *In Proc. of Ninth ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc '08)*, Hong Kong, 2008.

[17] N. Pentland and E. a. A., "Social Serendipity: Mobilizing Social Software," *IEEE Pervasive Computing*, vol. 4, no. 2, pp. 28-34, 2005.

[18] A. Beach, et al., "WhozThat? evolving an ecosystem for context-aware mobile social networks," *IEEE Network*, vol. 22, no. 4, pp. 50-55, Jul. 2008.

[19] D. Liben-Nowell, J. Novak, R. Kumar, P. Raghavan, and A. Tomkins, "Geographic routing in social networks," *National Academy of Sciences*, vol. 102, no. 33, pp. 11623-11628, 2005.