An Architecture supporting Knowledge flow in Social Internet of Things systems

Orfefs Voutyras, Panagiotis Bourelos, Dimosthenis Kyriazis, Theodora Varvarigou
National Technical University of Athens
Athens, Greece
orfeasvoutiras@gmail.com

***Abstract*— Recently, the idea that the Internet of Things (IoT) systems can be advantaged in many ways by integrating social networking concepts is gaining momentum. In this paper we present the social approach that the COSMOS project introduces. COSMOS supports knowledge flow between Things in order to provide a system that learns, observes and evaluates the usage and communication patterns and generates new knowledge. It focuses on the value of experience and experience-sharing and investigates models and principles designed for the social networks, which would provide it with the potential to support novel applications in more effective and efficient ways.**

Keywords— Internet of Things; Knowledge Management; Social Internet of Thing.

I. Introduction

The Internet of Things (IoT) is very challenging as it leads to networks connecting a huge number of Things that operate on different administrative domains. The scale and the complexity of the formed networks require new approaches that will make objects able to cooperate in an open and reliable way. Taking into consideration the rate at which IoT devices are deployed and used in different applications, one of the main challenges refers to the efficient and optimized management of these entities. Future internet applications tend to exploit a big number of devices, which highlights the need for distributed management approaches given that centralized mechanisms are either non efficient (for a huge number of things) or not applicable (e.g. due to communication problems). Furthermore, the Things are owned and operated by different administrative domains, thus centralized approaches in many cases cannot be used for their management given the diversity in access rights. What is required refers to techniques that will enable the formulation of subsets /sub-networks of Things in which management access is feasible. What is more, management decisions usually do not take into account the context under which the Things operate (e.g. specific object may be used with different configuration parameters in different applications). Approaches are required that will allow management decisions to incorporate situational awareness and propose management actions based on them. Finally, an additional challenge with respect to IoT management relates to the autonomous reasoning of Things on a context-aware basis. Autonomous management will integrate different types of knowledge (e.g. device-specific, situational, application-specific, administration-related, etc) and trigger decisions accordingly.

The COSMOS project [1] will provide a framework for the decentralized and autonomous management of Things based on service-, interaction-, location- and reputation-oriented principles, inspired by social media technologies. COSMOS, following the IoT-A reference model [2], supports real-virtual world integration by representing Things and groups of Things of the real world via their counterparts in the Cyberworld: Virtual Entities (VEs). VEs may have their own goals and be equipped with an internal logic in order to achieve them. They acquire perception through accessing sensor readings via IoT-services and can impact their environment or undertake physical actions using actuators via other IoT-services. Finally, VEs may interact with each other for various purposes like collaboration (sharing a common goal), cooperation (getting help from other VEs in order to achieve specific objectives), advertising of their properties/attributes, offering actuation services etc. Our approach follows the Social Internet of Things (SIoT) paradigm [3], as it defines, monitors and exploits social relations and interactions between the VEs, and uses technologies and services from the domain of the social media.

In order to achieve self-management and autonomicity we follow the MAPE-K model [4], as we estimate that it is very close to the nature of the IoT management. The IoT can provide to the MAPE-K all the data it needs to complete the autonomic cycle, while the adoption of suitable solutions for the implementation of the MAPE-K components can provide to the IoT optimal self-managing functionalities. However, we need to make a new approach dictated by the social view of the Things that is adopted. In this direction, we extend the MAPE-K loop by introducing two new components, Social Monitoring (SM) and Social Analysis (SA) [5].

Ontologies (example in Fig.1) are used for the description of the VEs, as they provide a rich vocabulary for the general domain knowledge, enabling the user to better express his/her requirements and submit queries, leading to greater precision and recall rates. Moreover, formalization of ontologies improves retrieval, similarity adaptation and learning [6]. It is of major importance to enhance the VEs with the key features of a social intelligent entity, which means that a VE has social characteristics, can acquire knowledge through various means, such as learning from experience [7], and can reason with knowledge to make plans, explain observations etc. The latter will allow VEs to learn based on their own experiences or those of other VEs, while situational knowledge acquisition and analysis will make them aware of conditions and events affecting their behavior. Socially-enriched coordination will consider the role and participation scheme of VEs in and across networks. Management decisions and runtime adaptability will be based on Things security, trust, location, relationships, information and contextual properties. Extended complex event processing and social media technologies will extract only the valuable knowledge from the information flows, while workload-optimized data object stores will facilitate efficient storage by also exploring the interplay between storage and analytics on networks of data objects.

The rest of this paper is organized as follows: Section II discusses the knowledge and experience concepts and presents the possible types of learning and communication for VEs. Section III introduces the social properties of the VEs and comments the value of friendship between them. Section IV refers to the management components needed to support the socialization of the VEs. Finally, Section V concludes the paper.

II. Learning through Communication

A. *The concept of Knowledge*

The IoT will create a flood of real world information to the virtual world. Our applications will be considerably enriched, as they will be more and more aware of what happens in the real world, in real time, everywhere. With a trillion sensors [8] embedded in the environment, all connected by computing systems, software and services, the future IoT platforms have to deliver data and information management mechanisms to handle the exponentially increasing "born digital" data. The transformation of this huge amount of raw data into knowledge is one of the biggest challenges behind the IoT. There is an entire cycle of data processing up to the generation of cooperative knowledge networks. These knowledge networks can feed complex hierarchical feedback control loops, since sensorial data is very important for decision making. Decisions made on the virtual side can be reflected on the real environment helping us to better use our resources. Hence, a first step to designing the general architecture of a project on the IoT domain and realizing its capabilities and chances for evolution is the definition of its own Knowledge Management (KM) cycle.

Knowledge management is the process of capturing, developing, sharing and effectively using knowledge and summarizes all activities with the goal of using knowledge in a more efficient and effective manner, achieving certain objectives. A Knowledge Pyramid, the DIKW Pyramid [9], is usually used for the representation of purported structural and/or functional relationships between data (D), information (I), knowledge (K) and wisdom (W). In the literature, typically information is defined in terms of data, knowledge in terms of information and wisdom in terms of knowledge. Generally, when we take data and put it in context we have information, when information becomes actionable it is transformed into knowledge and when pieces of knowledge are consolidated, with the help of experience, wisdom is born. In our DIKW Pyramid, *Data* are the raw-data which are collected from the VEs through their IoT-services. Physical objects like buses or houses which are represented by VEs will have a huge number of embedded sensors, continuously "feeding" COSMOS with data regarding the temperature and humidity of the environment, the velocity of the buses etc. *Information* is the result produced by analyzing the raw-data. Suitable mechanisms make possible the detection of simple or complex events of the physical world around the VEs. For example, analyzing the data offered by the sensors of the buses or the houses, the detection of events like "fire" or "traffic" becomes possible. *Knowledge* includes problems or situations detected (e.g. "fire") associated with specific solutions, implemented through IoT-services. In other words, *Knowledge* includes directions that specify how the VEs are going to react in changes of their environment in a well-defined way. For example, a house may include in its Knowledge Base (KB) the scenario of the problem "fire" and "know" that the solution to the problem is "inform the fire department". Knowledge is a store of information proven useful for a capacity to act. This level gives the VEs the advantage of learning from previous experiences. Finally, *Wisdom* is born using high-level reasoning techniques, such as Case-Based Reasoning (CBR) [10] and Rule-Based Reasoning [11], which give to the VEs the ability to reason and understand their situation and take decisions on their own, thus producing Knowledge on their own. Things attaining this level could be characterized as cognitive, intelligent or Wise, as they have the capacity to acquire, adapt, modify, extend and use knowledge in order to solve problems.

B. *The concept of Experience*

The proposed approach provides the VEs with the advantage of learning from previous experiences. Experience is usable knowledge acquired through the use of collaborating communication techniques between two or more individuals. Different types of experiences are defined, arising from the correlated phases of our control loop approach, which is adopted for the implementation of the project regarding VEs' management. Experience can be a piece of knowledge described by an ontology, a model resulting from Machine Learning or contextual information (Fig. 1). However, we focus mainly on the representation of experience through Cases as defined in the CBR technique. A case can be considered as a combination of a problem with its solution, whereas a problem consists of one or more events. In other words, a case is a kind of rule for an actuation plan, which is triggered when specific events are identified [12].

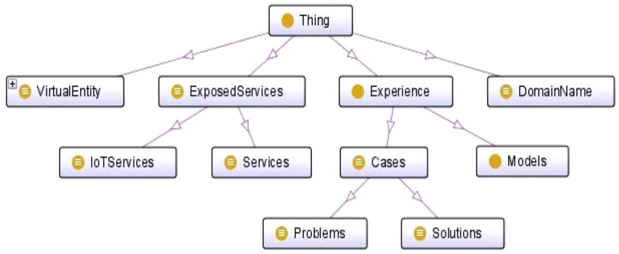

Fig. 1. Classes of the COSMOS ontology.

Each VE may maintain its own Case Base (CB) locally as part of its KB. Storage of experience in a local or central KB [13] depends on whether the individual's knowledge needs are constant or opportunistic. Such a categorization of needs will be primarily based on the "domain" membership of individual VEs as well as technical limitations that may be present. A KB can be shared between VEs with suitable social characteristics, something that improves the decision making mechanisms. Moreover, VEs representing weak devices that do not have their own KB can take advantage of the KB of their social group.

Finally, support of experience sharing gives to a VE the opportunity to ask for help from other VEs and find the most suitable solution by leveraging social features. Of great importance is the fact that the concept of brokers appears in experience sharing between VEs and their Friends.

C. Types of Learning and Communication

VEs have three types of learning cycles which are complementary and may occur in parallel [14]. These may interact with each other in complicated ways and are the following:

- *individual learning*: Individual learning will take the form of Cases creation and storage inside the VEs. By utilizing sensor readings and actuator values, each VE is capable of creating complete Cases of a complexity proportional to its technical abilities. The individual enrichment of the local CB can serve as a basis for the second stage of learning.
- *learning through communication*: This second stage comes into play when the locally stored knowledge is not sufficient for the needs of a VE. Such needs may be constant or opportunistic in nature, a distinction which helps segregate the actions taken on the provided knowledge. In this case, a VE uses the experience sharing (XP-sharing) service and targets a group of Friends that may have the required knowledge. Friends are maintained in a Friend List in the KB. When a VE decides to initiate the experience sharing mechanism with its Friends, it specifies the "depth" of communication (Fig 2). That is important mainly because of the recursive way the experience sharing method works, meaning that if a Friend of the original VE does not locate a suitable case inside its own local CB, it will check the depth required (mentioned TTL/time-to-live of the experience query) and initiate a new version of experience sharing this time directed at its own Friends. Therefore brokers are dynamically designated taking into account that Friends are willing and able to act as such for their respective Friends. It is also worth noting that respective brokers will not claim success as their own since the returning knowledge is also annotated with the id/name of the VE that successfully provided this knowledge. This approach is related to the "six degrees of separation" concept that has become quite popular at the domain of social networks [15].
- *learning through a knowledge repository*: Finally, if both previous knowledge acquisition mechanisms fail, VEs possess the ability to connect to a central KB. It is worth mentioning that experience will be as a final resort stored centrally in the COSMOS repositories so that a "purge" of acquired knowledge does not result in loss of experience.

We can make a distinction between two forms of learning through communication:

- *supply driven learning*: In supply driven learning, an individual VE acquires new experience and communicates it to the Groups of VEs (GVEs) it belongs to.
- *demand driven learning*: In demand driven learning, a VE comes along a new event/problem and asks its Friends whether they have a solution for this problem.

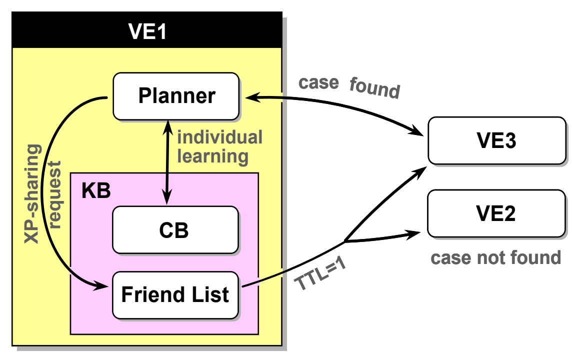

Fig. 2. Learning through communication.

In both cases, two factors should be taken into account:

- *overhead*: the number of useless messages that are acceptable from the recipients side.
- *hit rate*: the amount of VEs that get the message compared to the amount of VEs that should have received it.

Regarding the dissemination mode, there are three options to choose from (adopting the terminology from the advertising, marketing and communications domain):

- *Broadcasting*: Sending the message to every available VE. This way, the hit rate is maximized at the cost of a large communication overhead. An advantage of sending the message to a large audience is that it creates redundancy in the knowledge assets of the system, which facilitates knowledge development through combination. However, for most IoT use-cases, this is not an option due to scalability issues. As such, COSMOS does not provide any broadcasting mechanisms.

- *Narrow casting*: Sending the message to every VE that may be interested in a specific topic. This option combines the advantages of the other two, but it requires the VEs to state beforehand which kinds of messages they are interested in (e.g. by means of a user profile). This in turn requires that there is a predefined set of possible topics or, otherwise, that there are guidelines for creating new topics. In our case, data can flow through the system via a Message Bus which is organized into topics. Each VE can publish and/or subscribe to them. The whole process is supported by a Complex Event Processing (CEP) component which is responsible for processing data and analyzing them in real time, according to applications' specific logic. If a certain event is detected by the CEP component, this may trigger the generation of certain messages to a new topic.
- *Personal casting*: Sending the message only to VEs that are directly involved to its content. This is the most efficient way of communication, as only VEs that can directly help or can offer the new required knowledge are informed. In this way, the communication overhead is kept to a minimum, which is important for maintaining the communication channel alive. That is why the VEs need Friends and we should develop a social environment that can support their discovery.

As a final note, the technical solution used for learning through communication involves RESTful interfaces [16] that connect individual VEs on a peer to peer basis. Such a method increases the hit rate but has the side effect of taxing the network resources when a highly social VE requires knowledge (personal casting). On the other hand accessing knowledge through non targeted means, like topics on a Message Bus, guaranties a low overhead but increases the probability of the request not reaching all intended recipients (narrow casting).

III. SOCIAL PROPERTIES OF THE VIRTUAL ENTITIES

The need for effective and decentralized discovery of Knowledge/Experience by using the Social Internet of Things paradigm [17] brings us to the most important social concept that has to be implemented: *friendship between VEs*. In the spirit of implementing an autonomous and decentralized communication model, it is imperative to understand that communication between VEs will, after a certain point, be completely platform independent. This does not mean that the ontology will never be accessed, but that after social connections have been established, VEs are expected to communicate directly with each other. Friendship between VEs will be a guideline, a road map of communication, as each VE will maintain a group of VEs which have been deemed to be in a position to help it or receive help from it. The choice of Friends will be based on other social criteria, like their domains and trust indexes, with the eventual provision that friend lists are to be dynamically maintained. The social ontology possesses the property "hasFriend" which is object-type and non-symmetric. That means that if VE1 is Friend with VE2, then VE2 will not necessarily have a "hasFriend" property with VE1 as the target. In that sense, the concept of "Friends" matches this of Twitter "Followers" (non-mutual relationship) rather than Facebook "Friends".

It is important to state that even though the general ontology uses individuals to signify VEs, the local VE storage could use other means to store Friends, for example using only their URIs/IP address and ports. Finally, it is worth noting that in a fashion similar to social media, Friends of Friends can also be mined and used for further recommendation. The basic idea is that already established friendship patterns can offer an invaluable aid in determining VEs with similar interests to the VE accessing the ontology, requesting a recommendation.

The choice of suitable Friends is based on two composite criteria: Relevance and Dependability. Relevance includes the concepts of Homophily [18] and Distance Proximity, while Dependability refers to Reliability, Trust and Reputation (Fig.3). The analysis of the corresponding social properties follows:

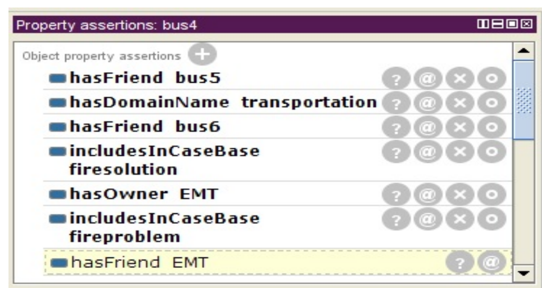

Fig. 3. Example of VEs' properties.

- **Domain (Homophily):** One of the main properties that should be included in the ontology of a VE is its domain. This property is quite important, as it is the first step to identifying the different groups into which VEs are organized and their different relations. The platform should give the developer the option to choose from a variety of diverse domains. As a result, we have to identify the different values that could be given to the social parameter "domain". Some ideas are: Traffic Management, Waste Management, Environment Monitoring, Smart Water, Smart Metering, Security & Emergencies, Logistics, Industrial Control, Home Automation, eHealth. The "domains" of the VEs could accelerate the discovery mechanisms and give more information for further social analysis. On the topic of domains, it is important to state that any VE wishing to register to the COSMOS platform is imperative to have at least one association with a certain domain. The list of possible domain names will be as stated a priori known to VE developers so that input of VEs to the platform's registration component will be efficiently handled. By dividing our ontology's scope into domain-specific parts we also achieve a functioning segregation of available IoT-services. That means that, if a certain VE exposes services with a multitude of purposes, intra-VE communication will be more effective as far as both discovery and service recognition are concerned. Therefore, if a certain VE desires a look-up of services pertaining to traffic management, domain identifiers can be used to limit the time needed for a query response. Also, regarding the VE discovery, if a VE has no connections to initiate communication in order to enrich its case base or its group of accessible services, the platform can initiate a process of recommendation. In the general ontology, membership to a domain is signified by the object-type property "hasDomainName". This property connects individuals belonging to the class "VirtualEntity" with individuals that are domains.

- **Physical Entity (*Homophily*):** This characteristic indicates the type of the actual physical entity represented by the VE. While it is not a strict segregation, similar VEs should be able to formalize friendship relations easier than completely diverse or unrelated ones. This attribute is represented in the social ontology by the use of "isPhysicalEntity" and is possible to take values like "BusStop", "HQ", "TrafficLight", "bus", "car", "house" etc.

- **Location (*Distance Proximity*):** The location property can take the following values:

  Fixed: for entities established in a permanent structure, not intended for portable operation, e.g. house.

  Portable: for entities fitted in a temporary location, e.g. laptop.

  Mobile: for entities that can move and by their nature change their position frequently and continuously e.g. vehicles, mobile phones.

- **Geo-location (*Distance Proximity*):** In specific cases, we can use data of Geo-location to ensure friendship suggestions are valid. If for example the VEs represent houses in a domain topic of "Home Automation" or "Environment Monitoring", then proximity with each other should be a consideration for the platform, as geographical locality implies relatively similar needs in the environmental variables that are to be balanced (temperature, moisture). That, in turn, implies the use of similar IoT-services as responses (solutions) to common problems. Geo-location variables are represented in the ontology through the use of the "hasGeoLat" and "hasGeoLon" data-type properties that use a range of float numbers to accurately store latitude and longitude respectively. We should note that too accurate positioning is not required in most IoT use-cases.

- **Dependability Indexes:** The social ontology contains three social indexes that define the Dependability of a VE:

  Reliability Index: an absolute indicator of the performance of the physical entity that quantifies the efficiency of its sensors and actuators functionalities, relative to their normal operation. The index is represented by the data-type property "ReliabilityIndex" which contains a float from 0 to 1.

  Trust Index: a counter which states how many times a VE has successfully shared its CB and/or IoT-services. Coupled with the concept of feedback and through refinement of its calculation, we can use this index as a means to simulate social mobility in the platform, as Trust will be one of the most important components of friendship recommendation. The index is represented by the data-type property "TrustIndex" which contains an integer.

  Reputation Index: a counter which monitors how many times the VE has received a request (how many "hits" it has). It is a cumulative and comparative indicator. The index is represented by the data-type property "ReputationIndex" which contains an integer.

Finally, a very important property of a VE, which could be characterized as hybrid, since it belongs to both the Relevance and Dependability criteria, is the **Owner ID**. It represents the physical owner (e.g. individual, organization) of a VE and determines friendship prioritization based on common ownership of VEs when other criteria are met too.

Individual VEs base the structuring of their local ontologies on the domains that they belong to. By using auxiliary ontologies provided by the platform they can import the ontology structure needed to store services and individuals such as Friends.

IV. Management Components

The framework supports various components that aim at the "socialization" of VEs. Such components are (Fig. 4):

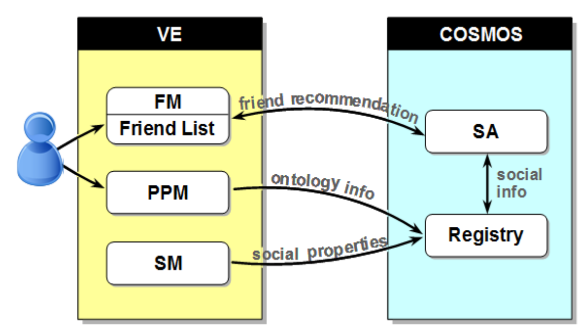

Fig. 4. Management components in the COSMOS architecture.

- The **Profiling and Policy Management (PPM)** component. It assigns a unique ID to the VE and enables the entry of all the information needed for the description of the physical entity through the domain ontology of the corresponding VE. Moreover, it enables the owner to determine the social "openness" of the VE: the IoT-services that can be used by other VEs, the kind of experience that can be shared, the sets of VEs which can access such information etc. However, the "openness" of VEs is affected by the social selfishness, a basic attribute of human beings. Thus, while designing this component, the concept of Opportunistic loT [19] should be taken under consideration.

- The **Friends Management (FM)** component is responsible for creating and maintaining the list of Friends that a VE has. In other words, it allows VEs to initiate, update and terminate their friendship with other VEs on the basis of the owner's control settings. It provides the owner with the option of setting new Friends to his/her VEs, offers friend-recommendation request services and monitors the Friend List of a VE regularly or on demand in order to find any Friends whose Dependability is no more the desired one and thus should be removed. For this purpose, it communicates with the SA component.
- The **Social Monitoring (SM)** component. It contains all the main tools and techniques that are used for the monitoring of the social properties of the VEs, like Trust and Reputation. Its main objective is to collect, aggregate and distribute monitoring data (events) across the decision making components of the collaborating groups. The events are generated by interactions in response to - directly or indirectly - user actions (e.g. registering a new VE) or VEs' actions (XP-sharing). Social Monitoring "feeds" the VE Registry.
- The **Social Analysis (SA)** component. Based on the results of the Social Monitoring component and taking advantage of Social Network Analysis (SNA) [20], the SA component is used for the extraction of complex social characteristics of the VEs (e.g. centrality), as well as models and patterns regarding the behavior of the VEs and the relations between them.

Figure 4 illustrates the way the aforementioned management components are involved when a VE sends to the platform a friend-recommendation request.

V. Conclusion

The COSMOS platform can be characterized as a SIoT platform since it defines, monitors and exploits social relations and interactions between the VEs and uses technologies from the domain of the social media. The social side of COSMOS improves the knowledge flow, which is of great importance for the constant evolution of the IoT systems, and introduces the concept of experience sharing between Things. However, one of the main concerns regarding the success of such an architecture is its potential to maintain an opportunistic IoT system, offering the human users motives to share the knowledge and IoT-services of their VEs.

Acknowledgment

The research leading to these results is partially supported by the European Community's Seventh Framework Programme under grant agreement n 609043, in the context of the COSMOS Project.

References

[1] COSMOS project: http://iot-cosmos.eu/.

[2] IoT-A: http://www.iot-a.eu/public.

[3] Social Internet of Things: http://www.social-iot.org/.

[4] IBM. "An architectural blueprint for autonomic computing.", Autonomic Computing White Paper, June 2005, Third Edition.

[5] O. Voutyras, S.V. Gogouvitis, A. Marinakis, T. Varvarigou, "Achieving autonomicity in IoT systems via situational-aware, cognitive and social things", in press.

[6] N. Guarino, "Formal ontology and information systems", Formal Ontology in Information Systems, FOIS 98, pages 3–15, Trento, Italy, June 1998. Ios Press.

[7] D. Kyriazis and T. Varvarigou, "Smart, Autonomous and Reliable Internet of Things", 4th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN), Niagara Falls, Canada, 2013.

[8] CISCO. "The Internet of Things, Infographic", 2011.

[9] J. Rowley, The wisdom hierarchy: representations of the DIKW hierarchy, Journal of Information Science 33(2) (2007) 163-180.

[10] A. Aamoth and E. Plaza, "Case-Based Reasoning: Foundational Issues, Methodological Variations and System Approaches", Artificial Intelligence Communications, 1994, pp. 39-59.

[11] S. Dutta, and P. P Bonissone, "Integrating case- and rule-based reasoning", International Journal of Approximate Reasoning, Volume 8, Issue 3, May 1993, 163–203, Elsevier.

[12] Z. Budimac and V. Kurbalija, "Case Based Reasoning – a short overview", Proceedings of the Second International Conference on Informatics and InformationTechnology, pp222-233.

[13] V. Supyuenyong and N. Islam,. "Knowledge Management Architecture: Building Blocks and Their Relationships", Technology Management for the Global Future, 2006. PICMET 2006 (Volume:3).

[14] U.M. Borghoff and R, Pareschin, "Information Technology for Knowledge Management", Springer, 18 Mar, 1998 - 232 pp.

[15] J. Guare, "Six Degrees of Separation: A Play", Vintage Books, 1990.

[16] A. Rodriguez, "RESTful Web Services: The basics", Developer works page REST, Nov. 2008.

[17] L. Atzori, A. Iera, G. Morabito, and M. Nitti, "The Social Internet of Things (SIoT) – When social networks meet the Internet of Things: Concept, architecture and network characterization", Computer Networks, Volume 56, Issue 16, 14 Nov. 2012, pp 3594–3608.

[18] H. Al-Qaheri, S. Banerjee, and G. Ghosh, "Evaluating the power of homophily and graph properties in Social Network: Measuring the flow of inspiring influence using evolutionary dynamics", Science and Information Conference (SAI), 2013.

[19] B. Guo, Z. Yu, X. Zhou, and D. Zhang, "Opportunistic IoT: Explorig the social side of the internet of things", Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference.

[20] S. Wasserman and K. Faust, "Social Network Analysis: Methods and Applications", Structural Analysis in the Social Sciences, Chambridge Univeristy Press.