# A Semantic Service Creation Platform for Social IoT

Victoria Beltran, Antonio M. Ortiz, Dina Hussein, and Noel Crespi
Wireless Networks and Multimedia Services Department
Institut Mines-Telecom, Telecom SudParis
91011 Evry, France
Email: {victoria.beltran, antonio.ortiz_torres, dina.hussein, noel.crespi}@it-sudparis.eu

***Abstract*—The Social Internet of Things (SIoT) is aimed at integrating devices into users' daily life by taking advantage of the interconnectivity and user-friendliness of Social Networks (SNs). To ensure the success of SIoT, we need to provide new attractive services that engage people to socialize their devices. Automation is a requirement to seamlessly integrate such services into user life. Ontologies are used to semantically describe Web services with the aim of enabling the automatic invocation and composition of these services. We envisage semantic Web services as a means to develop automated, value-added applications for SIoT. We propose an SN that can achieve a synergy between the SIoT and semantic Web services based on RESTful principles and ontologies. The proposed SN is a converging point for a user's friends, devices and Web services of interest. Moreover, this SN is a service creation environment through which users can define their own event-triggered services involving devices and Web services.**

## I. Introduction

The Internet of Things (IoT) is a novel paradigm that aims at completely changing the shape of modern wireless communications by making it possible for a broad array of devices around us, such as sensors, actuators, ID-tags, smartphones, tablets, etc., to interact and cooperate to achieve common objectives and improve users' everyday life [1].

Social Network (SN) services are basically promoted as a large network of people where the relationships among those in a certain community are modeled and described. Social networks are composed of nodes of people, and the edges between these nodes represent their relationships [2]. Many types of social networks have been built for different goals based on a range of basic information and different methods.

Objects can not only be part of traditional networks, they can also form an SN of smart connected objects that tends to mimic human behavior towards a scalable and effective service discovery and composition, as well as trustworthiness management [3]. In this sense the three worlds of the Internet, the IoT and SNs are being integrated to bring the physical real world into the virtual one. The resulting paradigm, called the Social Internet of Things (SIoT), has the potential to support novel applications and networking services for the IoT in more effective and efficient ways [4], but it is necessary to find solutions for resource visibility, service discovery, object reputation assessment, crowd-sourcing and service composition [3], [4].

Semantic Web services are an interesting approach to manage SIoT services, since they provide interoperability and automation that can be used by different stakeholders to unambiguously access and interpret data. However, despite the great interest in semantic Web services over the last decade, these services have not yet been realized in the current Web. This is due to a large extent to the complexity of Semantic Web technologies and the absence of attractive uses cases. It is clear that if service providers do not envisage the use of their services by an automated application, they will not take up the effort to semantically mark-up their services. Moreover, it is necessary to develop user-friendly and intuitive tools that will motivate people without technical skills to use semantic services. Through context-aware and proactive platforms that enable users to create their own services, it will be the users themselves who will bring out with new use cases.

In this paper, we propose using an SN as the converging point for people, Web services, and devices. We consider SNs as a meaningful opportunity to finally bring the Semantic Web and the IoT to users. Our proposed SN is a Service Creation Environment (SCE) in which users can create event-based actions involving their devices, friends, and Semantic Web services, as well as several Web APIs currently available on the Web (e.g., Google or Facebook APIs). Thus, we aim to promote SIoT and dissolve the perception of the Semantic Web as a utopian promise by bringing practical applications into a familiar and easy-to-use platform.

The rest of this paper is organized as follows. Section II outlines the evolution of Web services towards the Semantic Web, thereby providing a glimpse of our motivation. Section III describes the proposed SN and its main functionalities. The architecture and key components are detailed in Section IV. Finally, some conclusions are presented in Section V.

## II. Web of services: from interoperability to automation

Web services emerged to enable the access of information distributed on the Web. With the aim of providing interoperability, SOAP and WSDL were proposed to standardize Web service access and metadata. Nevertheless, overly complex and verbose WSDL files, as well as their tedious maintenance are just two of the reasons why these technologies present a significant learning curve for developers. Although SOAP and WSDL provide interoperability, it is not enough for the automation of Web services. It is necessary to model the meaning of Web services in a way that machines can fully understand the services themselves as well as their inputs and outputs. The Semantic Web is aimed at providing automatic Web service discovery, invocation and composition. To semantically describe Web services and their data, the Semantic Web relies on ontologies described by RDF and OWL. However, the complexity of Semantic Web technologies has hindered the emergence of real implementations of semantic Web services,

as well as their automatic discovery and composition. By September 2013, only 1.5% of the total number of Web services registered on the programmableWeb[1] were semantic.

Meanwhile, the Web has evolved on its own; the architectural REST style [5] has proven its success thanks to its alignment with the nature of the Internet. RESTful services basically model Web applications as a set of resources manipulated by HTTP methods (i.e., GET, POST, PUT, DELETE). RESTful services are increasingly made available over the Web; major websites such as Google, Facebook, Flickr, Salesforce and Amazon offer access to their functionality and data through RESTful APIs. By September 2013, 63% of the Web services registered in the programmableWeb were RESTful-based, and only 20% were SOAP-based.

While SOAP-based Web services' composition has not yet been broadly successful, mashups of Web services have rapidly attracted great interest and popularity [6]. The programmableWeb contained 7190 mashups by September 2013. However, Web service communication in such mashups remains hard-coded. Since Web services are not annotated with their semantics, they cannot automatically interoperate. Recently, RESTful services have gained attention as a simpler approach for providing the Web with semantics. Proposals have emerged to semantically annotate RESTful services, such as hRESTS [7] and SA-REST [8]. The main criticism leveled at hRESTS and SA-REST is their adherence to the RPC-like interaction model, thereby disregarding the architectural properties of RESTful services. Instead of representing RESTful services as resources that can be created, updated, retrieved and deleted, these methods describe input-operation-output information as traditional RPC-based services.

Some authors have recently addressed this drawback. SEREDASj [9] uses JSON to describe RESTful services as a set of resources that are semantically annotated by references to ontologies. This solution requires handling the description layer (i.e., a JSON document) and the data separately, which causes duplication of data and complicates the Web services maintenance. These reasons led the authors to propose Hydra [10], which is a vocabulary that merges Web service documentation and operations following the principles of the Linked Data. The authors also proposed JSON-LD [11] as the serialization format for Hydra. RESTdesc [12] describes other work that shares the same motivation but it expresses the operation of RESTful services in Notation3. The authors of [13] propose a set of microformats for HTML service descriptions in a resource-oriented style. They also provide attributes to link resources with the aim of facilitating service discovery and composition.

## III. People, Web Services and Devices: Let's Communicate and Create New Things!

We envisage an SN to converge a user's world; not only their friends but also their appliances and Web services. All of these, as a whole, becomes context-aware, user-centric, and semantic. Devices and Web services become social since they are integrated in the SN. Thus, each thing has a profile into the SN, which determines the thing's graphical interfaces (i.e., walls) to users. We use the term social thing to refer to both a social Web service and a social device. The proposed SN allows users to be continuously connected to and updated about their social things. Figure 1 outlines our proposal. In addition to 'friends', Web services and devices are part of the users' social network. Even logical groups of devices such as those at home or in an office can be seen as a single entity taking part in the user's social network.

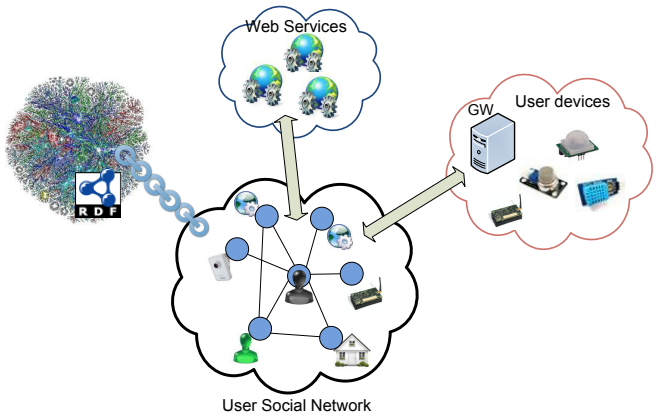


Fig. 1. Semantic SN for people, Web services and devices.

Some studies such as [14] and [15] have proposed service composition frameworks based on social networks. However, all of them extract information from social networks that, in some cases, is used to feed some external SCE. Our approach stands apart from the existing related work; we propose the social network itself as a SCE.

Through the proposed SN, users can browse their social things' walls to check their status or send commands through intuitive graphical interfaces. Users can also create Condition-Action (CA) rules that invoke an action on a device or Web service (i.e., the action part of a rule) when an event of interest occurs (i.e., the condition part of a rule). Rule events represent changes in the context of some social thing. Rule actions can be customized according to the nature of rule events. The proposed SN therefore offers users a simple mechanism for service composition, which can be automated by relying on the underlying data model's semantics.

The SN maintains a set of interconnected OWL ontologies that describe information about people, social things, and their context. As an ontological model, it enables all the intelligence of the SN, as well as the communication with semantic RESTful services. Moreover, the SN works closely with a gateway that exposes semantic RESTful interfaces and is responsible for communicating with the user's devices. Through this gateway, the SN can discover, get information from, and send commands to the user's devices. The gateway keeps the semantic description of devices and is responsible for performing the appropriate translations between the ontological model and the underlying data structures that devices understand.

### A. SN as a Service Creation Environment: Functionality

The SN is the graphical interface that allows users to interact with their things and to create their own personalized automatic services. Therefore, we need to provide users with

[1] http://www.programmableweb.com/

user-friendly and easy-to-use tools to navigate social things (i.e., devices that can interact in an SN), communicate with such things, and create services. To achieve this goal, social thing profiles will contain graphical features that make it easy to understand the meaning of a thing, its functionality and how a user can interact with it. A profile will facilitate the easy creation of simple commands and automatic actions. The proposed SN makes it possible for people to communicate with their devices and Web services of interest by providing the following functionalities:

- **Browse social things**: The user can browse her/his social things' profiles that show information and updates (e.g., provider, location, etc.) from the "thing". The nature of such updates depends on the type of thing. For example a temperature sensor will post temperature updates on its wall and a calendar Web service will post calendar updates on its wall.
- **Get the latest information from a social thing**: The user can type a textual command on a social thing's wall to obtain the latest value of some information provided by the social thing. For example, a user may type on her/his home's wall *give me the current temperature* and the value will be posted on the home's wall.
- **Change the status of a thing**: Users will be able to type textual commands on the wall of social things whose status can be modified, namely actuators. As an example, a user may type *turn air conditioner on* on her/his home's wall.
- **Create automatic actions interconnecting people and social things**: A user can set up event-triggered actions in the form of *IF (condition) THEN (action)* rules. For simplicity, rules contain a single action that is executed when the rule condition turns true. The condition part of a rule is a simple Boolean condition exhibited on one of the user's friends or social things such as *temperature >20* or *Anne's status is not busy*. A social thing's profile will show icons that indicate other social things to which a thing can be connected by means of an automatic action. By clicking on one of these icons, a graphical window to assist the user in creating the rule will appear. This window will present a form with condition and action parts, which will be adapted to the characteristics of the social things to be connected. To enable this rule creation assistant, the system automatically analyzes the ontology of social things so as to infer the possible combinations of conditions and actions that may be used by the user.

## IV. Architecture

The proposed architecture is fully based on an underlying ontology that maintains information about users, Web services and devices in the SN as well as other information such as user-defined automatic rules and context information. Figure 2 shows the architecture of the proposed SN. The whole system (i.e., the SN and a gateway) relies on the ontology database. The ontology-based layer is the highest-level layer built on top of the ontological information. It contains three main modules that strongly rely on the ontology database: the profile handler, the rules handler and the recommendations reasoner.

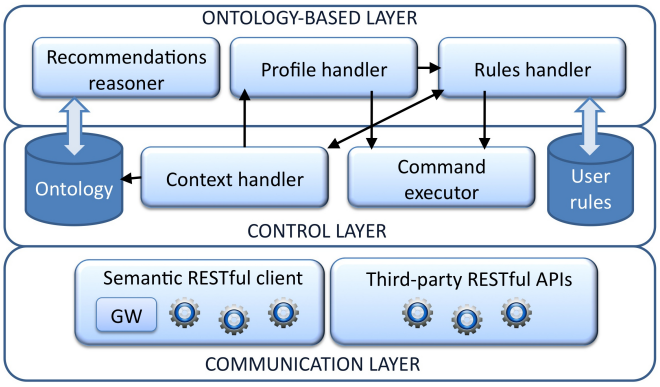


Fig. 2. Three-layered architecture of the proposed SN.

The profile handler manages the information associated with users' and social things' profiles, including how this information is shown to the user. It updates the profile when some new information of interest comes from the context handler in the lower-level layer. The profile handler also receives the user input and communicates with the command executor and the rules handler for commands and automatic actions, respectively. The rules handler controls the event-triggered actions created by users. When an automatic action is created, this module analyzes the context associated with the condition part of the rule and registers a subscription to such context into the context handler. Whenever the context associated with any subscription changes, the rules handler is notified. If the condition is satisfied, the rules handler will call the command executor to invoke the rule action. The third module in the ontology-based layer, that is, the recommendations reasoner, relies on the ontology database's and actual data to infer recommendations about possible automatic actions to users.

The control layer contains the system data model and the basic modules for the performance of the whole architecture such as the context handler and the command executor. The context handler is responsible for updating the ontology database with context information from external sources, that is, Web services and the gateway. This module notifies the upper-level layer about changes to information of interest. The command executor interacts with the communication layer in order to execute user actions. The communication layer contains all the software components for communicating with external services by means of RESTful interfaces. This layer implements a semantic approach to communicate with the gateway or with a third-party Web service. Moreover, this layer contains proxy objects that encapsulate Web APIs that do not follow a semantic approach. Such proxies provide interfaces according to the semantic data model and carry out appropriate translations to the specific data structures of each Web API.

### A. Semantic Web Services

We adopt a semantic approach to automatically integrate RESTful Web services into the SN. We have decided to use JSON-LD [11] as the serialization format for descriptions of RESTful APIs since it is a lightweight, easy-to-parse and easy-to-generate format. This simplicity has led many developers

to adopt JSON, resulting in the increasing growth of JSON-based Web APIs and the decrease of XML-based Web APIs. We therefore decided to use JSON-LD, that is, an extension of JSON for semantic annotations. We plan to use the Hydra [10] ontology to describe RESTful APIs as it is a simple, self-contained vocabulary that aligns with the linked data principles and takes a resource-focused approach. Nevertheless, we will evaluate this approach against others (e.g., SA-REST) in order to design the best solution for interoperating with semantic RESTful services.

The proposed social network will support the dynamic insertion and deletion of semantic RESTful services through an API. To this end, Web services will have to be semantically annotated with the vocabulary that is understood by the system (e.g., Hydra). Once the Web service has successfully registered, the system will learn the service semantics (i.e., for its operations and data types) through its annotations. Thus, the system will be able to build the service's profile and interconnect the service with the rest of the knowledge database. After creating the service's profile, the service will become part of the list of services that supports the recommendation system.

Since the proposed SN is considered as something that will become part of a users' life, we must consider current Web APIs that are not semantically annotated, such as Google Calendar, Twitter, Google Voice and so on. We will manually develop social wrappers for such services so that they can be uniformly integrated into the system. A social wrapper maps a service's data structures to an SN's semantic data model and performs the appropriate run-time translations to enable communication between the SN and the service.

### B. Semantic Modeling

The system's ontology database plays a crucial role in the proposed architecture since it defines the classes of information and their relationships handled by the system. Based on this ontology, the system performs automated reasoning on the system information (i.e., the knowledge database), which enables the automated communication with Web services and devices. We classify the system ontology classes into four inter-correlated groups: social things, user information, social wrappers, and social views. Social things refers to devices and Web services. User Information is information about people such as location, status, profile, preferences, etc. Social wrappers contain the classes and relationships that connect a social thing to the social network's model. They provide uniform interfaces to each social thing's information and actions. These interfaces are used by the SN to build profiles as well as conditions and actions of automatic actions. Social Views represent the instance of a particular social thing for a particular user. Thus, a social thing may have several associated views that contain information such as a user's preferences (e.g., kinds of information that are posted on the social thing's wall), user's credentials, and the automatic actions created by a user.

## V. Conclusions

The Social Internet of Things (SIoT) is a recently adopted term to refer to the integration of people and devices into the social network paradigm. The ultimate goal of SIoT is to enrich human life by seamlessly integrating their devices into their quotidian routine. We contribute to SIoT with a proposal that converges the Web, 'things' and users under a semantic approach. Our proposal goes beyond the SIoT in four ways. First, we include Web services in the loop (i.e., services become part of Social IoT). Secondly, this approach relies on the REST principles, more specifically on the semantic modeling of RESTful services. Thirdly, we evolve a social network into a service creation environment. Through the social network, users can create their own services based on their Web services of interest, their devices and context information. Lastly, we allow Web services to be connected to devices, thereby enabling the integration of the Web with the Internet of Things.

## Acknowledgements

This work was supported by the EU ITEA2 Project 11020, "Social Internet of Things - Apps by and for the Crowd" (SITAC).

## References

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] L. Ding, P. Shi, and B. Liu, "The clustering of internet, internet of things and social network," in *3rd Int. Symp. on Knowledge Acquisition and Modeling (KAM)*. IEEE, 2010, pp. 417–420.

[3] L. Atzori, D. Carboni, and A. Iera, "Smart things in the social loop: Paradigms, technologies, and potentials," *Ad Hoc Networks*, 2013.

[4] 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*, 2012.

[5] R. Fielding and R. Taylor, "Principled design of the modern web architecture," *ACM Trans. on Internet Technology (TOIT)*, vol. 2, no. 2, pp. 115–150, 2002.

[6] J. Yu, B. Benatallah, F. Casati, and F. Daniel, "Understanding mashup development," *Internet Computing, IEEE*, vol. 12, no. 5, pp. 44–52, 2008.

[7] J. Kopecký, K. Gomadam, and T. Vitvar, "hrests: An html microformat for describing restful web services," in *IEEE/WIC/ACM Int. Conf. on Web Intelligence and Intelligent Agent Technology*, 2008, pp. 619–625.

[8] A. P. Sheth, K. Gomadam, and J. Lathem, "Sa-rest: semantically interoperable and easier-to-use services and mashups," *IEEE Internet Computing*, vol. 11, no. 6, pp. 91–94, 2007.

[9] M. Lanthaler and C. Gütl, "Seamless integration of restful services into the web of data," *Advances in Multimedia*, vol. 2012, p. 1, 2012.

[10] M. Lanthaler and G. Christian, "Hydra: A vocabulary for hypermedia-driven web apis," in *6th Workshop on Linked Data on the Web (LDOW)*, 2013.

[11] M. Sporny, G. Kellogg, and M. Lanthaler, "Json-ld 1.0 - a json-based serialization for linked data," *W3C Working Draft*, 2013.

[12] R. Verborgh, T. Steiner, D. Van Deursen, J. De Roo, R. Van de Walle, and J. G. Vallés, "Capturing the functionality of web services with functional descriptions," *Multimedia Tools and Applications*, pp. 1–23, 2013.

[13] D. John and M. Rajasree, "A framework for the description, discovery and composition of restful semantic web services," in *2nd Int. Conf. on Computational Science, Engineering and Information Technology*. ACM, 2012, pp. 88–93.

[14] J. Rana, S. Morshed, and K. Synnes, "End-user creation of social apps by utilizing web-based social components and visual app composition," in *22nd Int. Conf., on World Wide Web companion*, 2013, pp. 1205–1214.

[15] Z. Maamar, N. Faci, Q. Z. Sheng, and L. Yao, "Towards a user-centric social approach to web services composition, execution, and monitoring," in *Web Information Systems Engineering-WISE 2012*. Springer, 2012, pp. 72–86.