An IoT-based User-centric Ecosystem for Heterogeneous Smart Home Environments

Luca Mainetti, Vincenzo Mighali, Luigi Patrono
Dept. of Innovation Engineering
University of Salento
Lecce, ITALY
{luca.mainetti, vincenzo.mighali, luigi.patrono}@unisalento.it

***Abstract*— Recent innovations in the ICT field are strongly focused towards the Internet of Things, which will definitely lead to an enhancement also in the domestic environments. Low-power and low-cost devices are expected to create a network of interconnected smart objects able to transform our homes into real Smart Homes. However, the heterogeneity of the underlying technologies prevents these smart objects to natively interoperate for adapting the environment to users' needs. In addition, common users are often excluded from the development of new applicative services that exploit physical devices, as they do not have sufficient programming and technological skills. To overcome these limitations, we propose a software ecosystem that allows different-skilled users to develop location-aware services able to autonomously manage the Smart Home. These services control the environment in accordance with user-defined rules and the users' location, calculated by exploiting an indoor localization mechanism. In addition, to directly interact with smart devices, users can also define customized interfaces for mobile devices. Finally, a multi-protocol middleware allows both the services and the mobile applications to access the physical network hiding the underlying heterogeneities. As a proof-of-concept, the first implementation steps are presented.**

Keywords*— *Indoor localization, IoT, middleware, M2M, Smart Home.

I. INTRODUCTION

The Internet of Things (IoT) is leading to the development of a plethora of smart objects that are intended to transform homes into real Smart Homes (SH). In this perspective, taking advantage of the latest innovations in the Information and Communication Technology (ICT) [1], the home automation devices are gaining increasing attention both in industry and in academia, since they are able to improve the comfort, safety and energy efficiency of our homes. However, their widespread diffusion is still limited by both the poor maturity of the controlling software systems and energy issues. While for the last problem there are already several meaningful solutions in the literature [2], the first issue is still very debated. The vast devices' heterogeneity leads to strictly *vertical* software solutions, meaning that they are focused on specific contexts and only on certain technologies. On the contrary, a *horizontal* approach able to involve heterogeneous technologies would be much more useful, since it would enable rapid and efficient development of applicative services.

Moving to the upper layers, another debated issue relating to the SHs concerns their capability to act as autonomously as possible, with minimal human intervention. The "remote control function", which allows users to directly interact with the physical devices, is undoubtedly very interesting and, in some cases, necessary for security reasons, but the ability of the environment to automatically react to some events is even more attractive. In other words, the transition from the Human-to-Machine (H2M) to the Machine-to-Machine (M2M) communication paradigm would definitely lead to an improvement of the User Experience (UE). The current trend is to give location-awareness to the environment, so that it can change in accordance with location information (i.e., users movements).

Finally, another increasing trend within the SH context and, more generally, in the IoT is the greater involvement of the end-users in the development of the services and applications they will have to use. Therefore, an infrastructure that allows different-skilled users to autonomously enrich the system with new applications and services is more and more important for any IoT solution.

In this paper, we propose a system able to address all the above-described issues. In more detail, the proposed system provides a multi-protocol middleware that allows a transparent access to the heterogeneous IoT technologies hiding the low-level communication details. It is able to guarantee flexibility and scalability, since it can be easily extended to new technologies. The business logic that manages the environment status is, instead, executed by several location-aware services, which act in accordance with both user-defined rules and users' movements. These services rely on a distributed indoor localization mechanism that tracks the position of people in the house. Finally, the system provides simplified tools for the implementation of both location-aware services and graphical interfaces to interact with the environment by a mobile device. This way, also "common users" can develop and customize their SH environment. As proof-of-concept, the first implementation steps are described.

The rest of the paper is organized as follows. In Section II an overview on the current state-of-the-art is presented. Section III provides a description of the main design challenges to address. The details of the proposed architecture are presented in Section IV. In Section V, a proof-of-concept for evaluating

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

the effectiveness of the proposed solution is described. Conclusions and future works are summarized in Section VI.

II. Related Works

In the literature, there are several works addressing the aforementioned issues, but none of them provides a flexible and scalable solution that can solve all the problems in one system. Some solutions are focused on specific technologies and are mainly aimed at simplifying the development and customization of user applications. For example, in [3], authors develop and validate an architecture, both hardware and software, able to monitor and manage a Konnex-based (KNX-based) home automation system. On the other hand, to solve the problem of devices heterogeneity, most of the approaches are based on the concept of middleware. For example, in [4], the authors focus the attention on various integration styles for non-IP based devices already deployed in home and building automation systems. Instead, in [5], the readiness and compatibility of existing building automation system technologies with IPv6 are investigated, and the integration challenges and new opportunities of IPv6 for these technologies are presented. Finally, in [6], the authors leverage the principles of the Web of Things approach to implement a gateway able to expose capabilities of a KNX home automation system as Web Services.

The other key feature of this work is represented by the indoor localization mechanism, which is a very hot topic in the recent literature. As an example, in [7], the authors present a personalized smart control system that: (i) localizes the user using the magnetic field of the smartphone, and (ii) controls appliances present at the user's location. Another example of location-aware services in a SH is [8]. Here, the authors propose a system that collects information from the environment and then provide services to improve the lifestyle of the users, mainly from the energy point of view. The last example is provided in [9]. In this case, the authors propose an access control mechanism, which consists of an engine embedded into smart objects, able to take authorization decisions by considering both user location data and access credentials. User location data are estimated using magnetic field measured and sent by phone.

Also the last issue, that is, the involvement of different kind of users in the development of services and applications, has led to meaningful solutions in the literature. An example is presented in [10], where authors propose a distributed architecture that allows to graphically create and control mash-up applications in an easy and scalable way, without specific knowledge on both hardware and programming languages. Finally, in [11], the authors try to solve some previously mentioned issues through three main components: a Domain Specific Language for abstracting the application generation problem, a graphic editor that simplifies its creation, and an IoT platform for interconnecting heterogeneous objects.

III. Design Challenges

The design of an architecture able to satisfy the requirements previously described needs to address several challenges, as explained below:

- The multi-protocol middleware has to communicate with all the underlying technologies, hiding their intrinsic heterogeneity. This aspect implies a deep study of the IoT technologies and standards, in order to identify the common models that characterize both the physical devices and their interactions. Exploiting these general models, it is possible to define the high-level interfaces that the middleware exposes to the upper layer. Furthermore, in order to make the architecture highly scalable, the structure of the middleware should be modular, so that new technologies can be easily integrated into the system without modifying the interface provided to the services.
- Since the main services provided by the proposed architecture manage the home environment basing on the user's location, the system has to provide a service that continuously tracks, in the background, the location of people moving in the house. Then, this information has to be promptly provided to the other services. Therefore, it is necessary to identify a low-cost and low-power technological solution able to perform this task with an accuracy appropriate to the target scenario.
- Although the proposed architecture has to automatically manage the environment, it is however desirable that the user can still send commands to the devices bypassing the business logic of the location-aware services (e.g., for remotely acting on his/her home system). The best solution seems to be an application for mobile devices, since smartphones are almost ubiquitous in our society. In this case, the messages sent from user's mobile device and those coming from home services have not to interfere.
- The proposed solution provides users with simplified tools, such as graphic editors or step-by-step wizard, to easily define both new location-aware services and graphic interfaces for mobile devices. These tools should be based on an abstract language able to describe the physical devices, the business logic of the services, and the layout of the user interfaces. These abstract descriptions have to be transformed into concrete implementations within a home gateway and the user's smartphone.

IV. Architectural Design

The overall architecture of the proposed system is shown in Fig. 1. Macroscopically, it is possible to group the building blocks into two main parts, a static part and a dynamic part. The multi-protocol middleware and the location service (which is distributed among the user's smartphone and the home gateway) represent the static part, since they cannot be modified by the end user. The location-aware services and the user interfaces are the dynamic components of the system, as they can be created and customized by the end user at any time. To do so, the user is provided with the *development tool for services and interfaces*. It allows to easily describe the home devices as well as the services and user interfaces that involve these devices. Finally, the *Management* (MGMT) *Service* is a system service that coordinates the data exchange among the other system components.

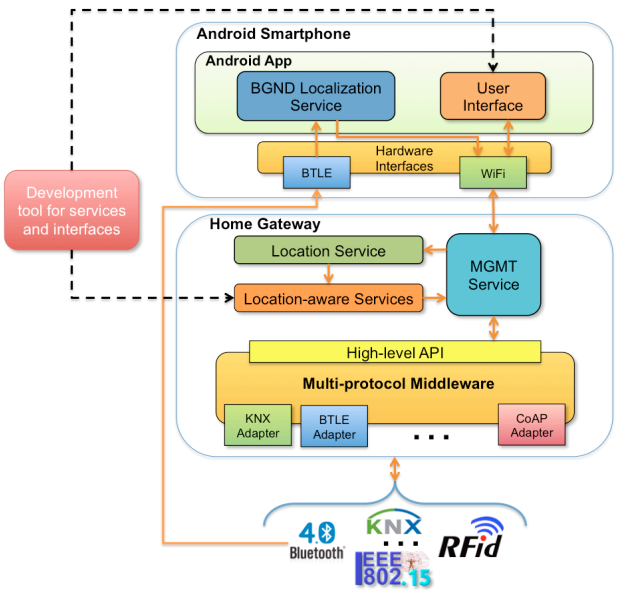

Fig. 1. The overall system architecture.

A. Development tools

The development tool is substantially constituted by three main sections. In the first section, the user can define the physical devices installed in the SH. The basic idea is to model a device as a set of elementary functions, each of which is characterized by several attributes (Fig. 2): physical and network reachability, interaction accepted by the function (e.g., read or write commands), expected payload (e.g., numeric, string, boolean, etc.), and main feature of the function (e.g., temperature indication, switch actuation, dimming actuation, etc.). This parametric description allows to abstract the heterogeneity of devices and to see them only from a functional point of view. However, it is also necessary to associate a *type attribute* to each device to identify the technology of the device itself. This "technology code" is then used by the multi-protocol middleware to map the high-level requests on the low-level messages for the physical network. In addition to the functional characteristics of each device, the user can also define their graphical appearance, that is, how they will be represented in the mobile application. In more detail, for each device, the user can define the size (as number of rows and columns) of a matrix and then the position of each function in the matrix. These design choices allow to improve the scalability and flexibility of the system, since any new device can be immediately integrated by simply describing it as a set of basic functions. The descriptions of the devices are stored in a file, in order to be available for the definition of both services and interfaces, as shown in the following.

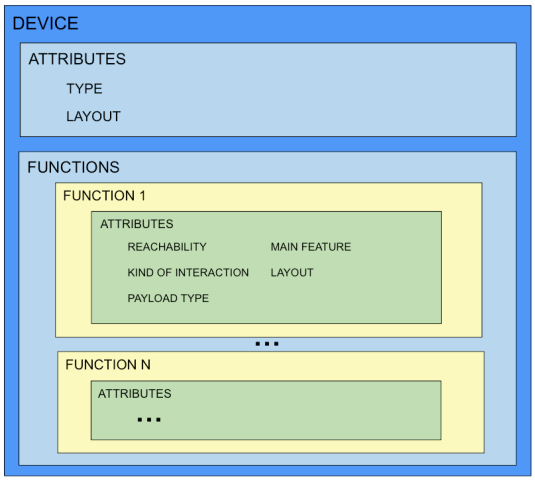

Fig. 2. The basic model of a generic home device.

In the second section, the user can define the business logic of the services that manage the SH. To do so, a simple graphical interface allows to generate the rules that characterize this business logic. A simple example rule is:

$$user(x) \,\&\, isInRoom(x,y) \,\&\, temp(y,z) \,\&\, isGreaterThan(z,t) \Rightarrow cooling(y,on)$$

It states that "if the user x is in the room y and the temperature z of the room y is greater than a threshold t, then the cooling service of the room y has to be turned on". The data entered by the user are then exploited to create part of a configuration file that fully describes the desired service. In particular, in addition to the user-defined rules, also the descriptions of the devices involved in these rules are included in the file. The complete service description file is then sent to the home gateway, where the service is actually instantiated.

Finally, in the third section of the development tool, the user can build the graphical user interface of the mobile application. In particular, s/he can define the navigation structure by describing the hierarchy and the content of the application screens. For each screen, it is possible to add both references to other screens and graphical controls for interacting with the home devices. Also in this case, the configuration process leads to a description file that contains the screens hierarchy and the descriptions of the devices included in each screen. Fig. 3 shows an overall view of the procedures described in this Section.

B. Multi-protocol Middleware

The multi-protocol middleware is the software component through which both the location-aware services and the mobile application are able to interact with the home devices. It is free from any business logic, since it only has to act as an interface for sending commands to the physical devices or for soliciting updates from the network. The main feature of the middleware is the ability to abstract the heterogeneity of the underlying technologies, providing a common interface to the upper layers. This concept is the core of any software architecture dedicated to the IoT, since it allows to transparently communicate with heterogeneous technologies. To make possible this kind of approach, the middleware is equipped with appropriate software modules, the *adapters*, which are able to communicate with the IoT technologies according to the respective standards and protocols. This feature allows to easily extend the gateway to new technologies by developing the corresponding adapter.

Regarding the high-level interface provided to the location-aware services and to the mobile application, the choice has

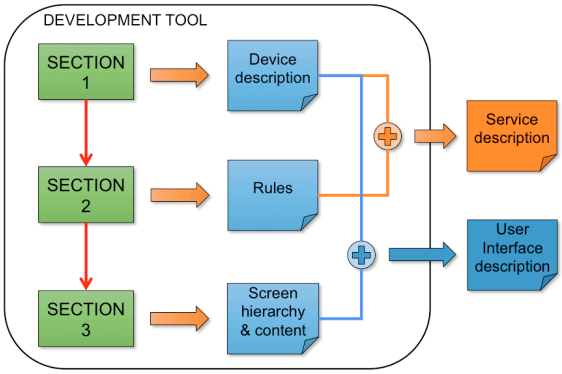

Fig. 3. The overall view of the development tool.

fallen on a RESTful interface, since the Representational State Transfer (REST) paradigm [12] is actually the core interaction paradigm of the Web. It sees each sources of data as a resource addressable by a Uniform Resource Identifier (URI) and it allows the manipulation of this resource by means of the basic methods of the Hypertext Transfer Protocol (HTTP), that is, GET, POST, PUT, and DELETE. This vision fits perfectly in the context of SHs, in which each network device (e.g., sensors and actuators) can be seen as a resource whose status can be requested or updated. For example, if the location-aware service that manages the temperature of the living room needs to know the current value of the temperature sensor, it has to send a GET request to the resource representing that sensor. Once received the request, the middleware has to send it to the physical sensor by using the proper adapter. To do so, it can exploit the "technology code" embedded into the requesting message, which is specific for each technology.

C. Services

The services of the system include both the location-aware services and the distributed location service. The first ones are defined through the development tool described in Section III.A and are actually instantiated when the home gateway receives the corresponding configuration file. Since these services have to run continuously, they are basically infinite loops in which the user-defined rules are cyclically evaluated. In more detail, when a service is instantiated, it subscribes to the location service in order to keep itself up to date on the position of the "target users". Moreover, it also instantiates, in its source code, all the "target devices" by exploiting the corresponding description embedded in the configuration file. The target users and the target devices represent the users and the devices involved in the rules that characterize that service. After this setup phase, every time the location-aware service receives an update on the position of a target user, it evaluates its own rules for determining whether to interact with the physical devices or not.

Since all the services depend on the location service, it is definitely the most important service of the architecture. It consists of three main components: an infrastructure of wireless landmarks that periodically send localization information, a service installed on the user's mobile device that collects the information of the landmarks to determine its location, and the service running on the home gateway that receives the location of the mobile device and notifies the other services about it. More specifically, the network of wireless landmarks consists of embedded devices equipped with Bluetooth Low Energy (BTLE) interface and placed individually in the different rooms of the house. The choice of BTLE is mainly due to its low energy consumption in front of a communication range comparable with that of the traditional Bluetooth. This way, the wireless landmarks can be battery powered, making the localization mechanism more flexible and less invasive. Each device of the BTLE infrastructure sends its location indication together with the Received Signal Strength Indication (RSSI) value. The service running on the user's mobile device collects location information from all the landmarks that are within its listening range and then determines the room in which it is located. To do so, for each landmark, it computes a proximity index d, using the corresponding value of the RSSI in the following formula:

$$RSSI = -(10n\log_{10}d + A) \qquad (1)$$

where A is the received signal strength at 1 m, n is a signal propagation constant depending mainly on the environment, and d is the distance from the sender [13]. The landmark that has the lowest value of d represents the user's current location. This information is then sent to the service running on the gateway, which stores it and sends a notification to any interested services. Fig. 4 shows the main components of the user localization process.

Finally, the management service enables the communications among the user's mobile device, the system services, and the multi-protocol middleware.

D. Android Application

The main components of the mobile application are: a background service that calculates the user's current location, and the graphical interface through which the user interacts with the home environment. The first component is part of the distributed location service, and it is launched in background at the mobile device startup. The procedure for calculating the user's position and for communicating it to the home gateway

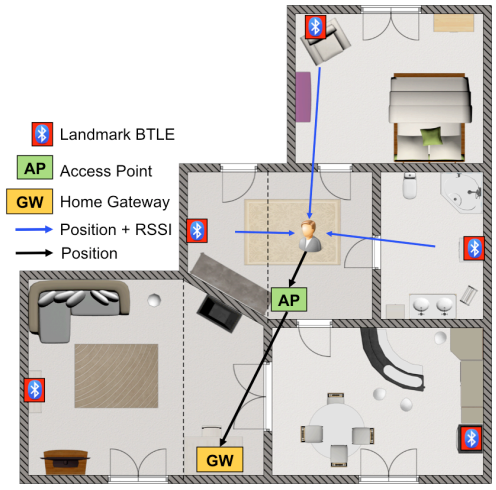

Fig. 4. The main components involved in the user localization process.

has already been described in the previous Section.

The second component represents the real means of interaction between the user and the smart devices. The appearance of the graphical interface is highly customizable, since it is built by parsing the configuration file generated by the user through the development tool. In more detail, the parsing process can be logically divided into two steps (Fig. 5). During the first step, the configuration file is scanned to build the navigation structure of the interface. In particular, for each application screen, the links to other screens are created. In the second step, the configuration file is scanned looking for references to physical devices. For each of them, the corresponding device description is retrieved from the file to generate the graphical element representing the device. In particular, the layout attributes are used to set the graphical appearance of the device, whereas the other attributes are exploited to define its business logic. At this step, also the *type attribute* is set to define the "nature" of the device (i.e., the technology it is compliant with). This way, the multi-protocol middleware could understand which adapter is needed to satisfy the requests coming from that graphical element.

It is worth noting that, since the application bypasses the business logic of the location-aware services, no conflicts have to be generated with the messages sent by the services. For this purpose, the mobile application provides a simple option through which it is possible to pause the execution of the location-aware services. In this "manual" mode, the home status is exclusively managed through the mobile application.

V. Proof-of-Concept

In this section, the first implementation results are presented. In particular, the architectural components currently developed and validated are: the multi-protocol middleware with support for KNX [14] and Constrained Application Protocol (CoAP) [15], the distributed location service, and the development tool for defining the interface of the mobile application (but not yet the location-aware services). Fig. 6 shows the simple scenario used for the validation phase. It consists of a heterogeneous building automation system in which there are both KNX-compliant devices and CoAP-based

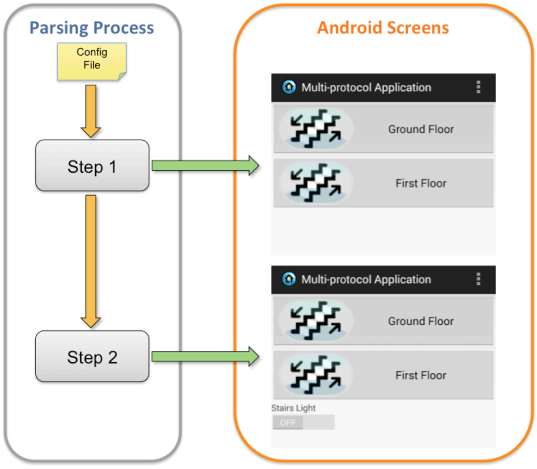

Fig. 5. The main steps of the parsing process.

Fig. 6. Test environment.

Wireless Sensor Network (WSN) nodes. The initial choice of KNX and CoAP is due to their diffusion in both commercial and academic solutions available in the literature. However, as said, any other technology that characterizes a SH can be integrated into the system. The software tools used to build the KNX and CoAP adapters are Calimero [16] and Californium, respectively. The first one is a Java library that provides a collection of APIs for interacting with KNX-compliant devices. The second one is a modular CoAP Java implementation that allows the creation of CoAP-based clients and servers.

Regarding the distributed localization service, the network of wireless landmarks is realized by transforming several Raspberry Pi boards [17] in iBeacon (called in this case piBeacon), i.e., low-powered and low-cost transmitters able to notify nearby mobile devices of their presence. To realize these piBeacon boards, a BTLE dongle is integrated into each Raspberry Pi, whilst the BlueZ stack [18] is exploited to provide them with BTLE functionalities. On each piBeacon, a dedicated Java application broadcasts the information about the position of the device and the RSSI value. On the other side, the mobile application collects this information in order to calculate its position, as described in the Section III.C. This application was implemented in two versions: an Android mobile application, and a Java application deployed on a prototype wearable device, namely the Odroid Xu [19]. This second option is useful for adapting the environment according to the movement of people who should not (or can not) directly interact with the environment (e.g., children, elderly, disabled, etc.). Then, on the home gateway, a simple service able to receive information from the mobile application and store it locally has been configured. This service has a list of other services that are interested to be promptly notified about target users' movement.

Finally, some components of the development tool have been implemented, namely the section for defining the devices, and the section for building the graphic interface of the mobile application. In more detail, to describe the physical devices, the user could exploit a friendly interface that allows to define all the functions of a device and the attributes of each function. The result of this process is an XML file that consists of a set of *device nodes*, each of which has one or more *function*

childNodes. Moreover, each *device node* has the layout attributes that specify the size of the matrix into which the basic components of the device should be placed. Of course, each *function childNode* specifies the pair <row, column> identifying its position into the grid (Fig. 7).

To define the navigation structure of the application and the content of the graphical interface, the user has to exploit the other section, which allows to build the screens hierarchy and their content. Also this section leads to the creation of an XML file, which consists of one or more *screen nodes*, each of which may contain both *screenRef* and *deviceRef childNodes*, i.e., references to other *screen nodes* and *device nodes*. In the resulting XML file, also the referenced *device nodes* are added, picking them from the XML file built in the first section.

Obviously, an appropriate parser has been implemented in the Android smartphone to process the configuration file created by the development tool. This component is in charge to perform the parsing process described in Section III.D.

VI. Conclusion and Future Work

In this paper, we have presented a system for SHs, which provides support at various architectural levels, from the interaction with the heterogeneous physical devices, to the automatic management of the home status, also including the implementation of new services and applications. In more detail, at the lowest level, a multi-protocol middleware provides the upper layers with an interface for the transparent access to physical devices. For the environmental management, an indoor localization service and a set of user-defined rules enable various location-aware applications to adapt the status of the environment to the users' needs and their movements. In addition, the H2M communication is still guaranteed through highly customizable mobile applications. Finally, to allow users to independently create new services and applications, a simplified development tool allows them to design and develop new solutions without specific programming knowledge.

As future works, beyond the final development of the basic features, some critical situations need to be addressed, such as the potential inconsistency among the rules that characterize concurrent services. Then, in order to increase the appeal of the proposed solution, we will provide the opportunity to develop user interfaces for different platforms. Finally, the issue of security must be addressed to prevent the violation of data transferred to and from the smart objects.

```
<Device Id="SWITCH-00002">
    <Name>Stairs Light</Name>
    <ConnectionRefId>KNXIP-1</ConnectionRefId>
    <Type>KNX</Type>
    <Layout>
        <LayoutType>grid</LayoutType>
        <LayoutParameters>1,1</LayoutParameters>
    </Layout>
    <Function>
        <URI>1.1.5</URI>
        <WriteAddress>0/0/2</WriteAddress>
        <ReadAddress>0/1/2</ReadAddress>
        <PayloadType>numeric</PayloadType>
        <Feature>light switch</Feature>
        <GUI>
            <Control>switch</Control>
            <LayoutPosition>1,1</LayoutPosition>
        </GUI>
    </Function>
</Device>
```

Fig. 7. Description of a KNX switch actuator.

References

[1] M. Naphade, G. Banavar, C. Harrison, J. Paraszczak, and R. Morris, "Smarter cities and their innovation challenges," Computer, vol. 44, no. 6, pp. 32–39, 2011.

[2] L. Anchora, A. Capone, V. Mighali, L. Patrono, and F. Simone, "A novel MAC scheduler to minimize the energy consumption in a Wireless Sensor Network", Ad Hoc Networks, vol. 16, 2014, pp. 88–104.

[3] G. De Luca, P. Lillo, L. Mainetti, V. Mighali, L. Patrono, and I. Sergi, "The use of NFC and Android technologies to enable a KNX-based smart home," in Proc. 2013 Int. Conf. on Software, Telecommunications and Computer Networks, SoftCOM 2013, 2013, pp.1-7.

[4] M. Jung, J. Weidinger, C. Reinisch, W. Kastner, C. Crettaz, A. Olivieri, and Y. Bocchi, "A Transparent IPv6 Multi-protocol Gateway to Integrate Building Automation Systems in the Internet of Things," in Proc. 2012 IEEE Int. Conf. Green Computing and Communications, Besancon, 2012, pp. 225 – 233.

[5] M. Jung, C. Reinisch, and W. Kastner, "Integrating Building Automation Systems and IPv6 in the Internet of Things," in Proc. Sixth Int. Conf. Inn. Mobile and Internet Services in Ubiquitous Computing, Palermo, 2012, pp. 683 – 688.

[6] G. Bovet and J. Hennebert, "A Web-of-Things Gateway for KNX Networks," in Proc. European Conference on Smart Objects, Systems and Technologies, Germany, 2013, pp. 1 –8.

[7] K.P. Subbu and N. Thomas, "Poster abstract: A location aware personalized smart control system," in Proc. 13th International Symposium on Information Processing in Sensor Networks, Berlin, 2014, pp. 309 – 310.

[8] J. Wang, C. Zixue, L. Jing, O. Yota, and Y. Zhou, "A location-aware lifestyle improvement system to save energy in smart home," in Proc. 4th International Conference on Awareness Science and Technology, Seoul, Korea, 2012, pp. 109 – 114.

[9] M.V. Moreno, J.L. Hernandez, and A.F. Skarmeta, "A New Location-Aware Authorization Mechanism for Indoor Environments," in Proc. 28th International Conference on Advanced Information Networking and Applications, Victoria, 2014, pp. 791 – 796.

[10] L. Mainetti, V. Mighali, L. Patrono, and P. Rametta, "Discovery and Mash-up of Physical Resources through a Web of Things Architecture", Journ. of Communications Software and Systems, vol. 10, no. 2, June 2014, pp. 124-134.

[11] C. González García, B.C.P. G-Bustelo, J.P. Espada, and G. Cueva-Fernandez, "Midgar: Generation of heterogeneous objects interconnecting applications. A Domain Specific Language proposal for Internet of Things scenarios", Journ. of Computer Networks, vol. 64, May 2014, pp. 143 - 158.

[12] L. Richardson and S. Ruby, "RESTful web services," O'Reilly Media, May 2007.

[13] E. Lau and W.Y. Chung, "Enhanced RSSI-Based Real-Time User Location Tracking System for Indoor and Outdoor Environments," in Proc. Enhanced RSSI-Based Real-Time User Location Tracking System for Indoor and Outdoor Environments Convergence Information Technology, Gyeongju, 2007, pp. 1213 – 1218.

[14] http://www.knx.org/. Retrieved: July, 2014.

[15] The Constrained Application Protocol (CoAP), RFC 7252, June 2014.

[16] http://calimero.sourceforge.net/. Retrieved: July, 2014.

[17] http://www.raspberrypi.org. Retrieved: September, 2014.

[18] http://www.bluez.org. September, 2014.

[19] http://www.hardkernel.com. September, 2014.