A Location-aware Architecture for Heterogeneous Building Automation Systems

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

***Abstract*— Smart Objects and Smart Environments are expected to become two of the leaders of the future Internet of Things. In this context, the Smart Homes are getting more and more attention, since people are very attracted by the idea of a home environment able to automatically meet their needs. However, the heterogeneity of the smart devices, the difficulty in automating the user-home interaction, and the poor involvement of the users in the development process of new services and applications still represent very debated issues. So, in this paper, we propose an architecture able to overcome the heterogeneity of smart devices and that can be easily extended to new future technologies. To maximize the User Experience, the proposed architecture automatically manages the environment basing on users-defined rules and on people movements, by exploiting an indoor location service based on Bluetooth Low Energy. Finally, the system also provides a simplified development tool that allows even common users to develop new services for Smart Homes and mobile applications to directly interact with the home environment. As a proof-of-concept, the first development steps are described in this paper.**

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

I. INTRODUCTION

Next Internet will be focused on the Internet of Things (IoT) concept. It involves the extension of the Internet to small and low-cost "things" that are thought to realize smart environments in order to provide new services to the users. The continuous attention towards this new vision puts an extraordinary stress on the so-called smart cities. They aim at increasing the effectiveness and efficiency of energy supply, healthcare, home and building automation, by integrating various systems through Information and Communication Technology (ICT) [1]. In the literature, there are many examples that highlight the research effort towards the real applicability of these new technologies, especially from the power consumption point of view [2, 3]. Simultaneously, also Smart Home is becoming a very hot topic, as people are increasingly interested in solutions able to make more comfortable and controllable their living ecosystem. For this reason, commercial home automation systems are gradually gaining wide market sectors and have also inspired some research works aimed at providing innovative solutions for the interaction with the home environment. For example, in [4], authors develop and validate an architecture, both hardware and software, able to monitor and manage a Konnex Networks-based (KNX-based) home automation system. Unfortunately, due to the heterogeneity of smart objects, both the academic and industrial solutions are often focused on specific technologies. On the contrary, it would be desirable moving from *vertical* solutions to *horizontal* ones. To do so, one of the approaches is based on the concept of middleware. It is an architectural component able to present high-level interfaces to the upper layers, in order to shield the heterogeneity of underlying technologies. In this way, new services can be developed more easily and the interaction with the physical objects can be more flexible. For example, in [5], the authors focus the attention on various integration styles for non-IP based devices already deployed in home and building automation systems. Instead, in [6], the readiness and compatibility of existing building automation system technologies with IPv6 are investigated, and the integration challenges of IPv6 for these technologies are presented. Finally, in [7], the authors leverage the principles of the Web of Things approach to implement a gateway able to expose capabilities of a KNX system as Web Services.

Moving to the upper layers, it is worth noting that, to improve the User Experience (UE) in a Smart Home, the Human-to-Machine (H2M) communication paradigm should be replaced by Machine-to-Machine (M2M) communications, i.e., devices cooperate to change the status of the environment without human intervention. To achieve this result, multiple applications (services) should constantly manage the interaction among physical devices, in accordance with a predefined business logic. The current trend is to configure these services as location-aware services, i.e., applications driven by location information. Therefore, the tracking of users in indoor environments to support location-aware services is a major challenge for the creation of effective Smart Homes. As an example, in [8], the authors present a 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 Smart Home is [9]. 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 [10]. 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.

Finally, another increasingly important aspect in the IoT is the growing desire of users to be not only consumers of technology but also designers and creators of services and applications for both themselves and other users. Unfortunately, this possibility is often denied to "common users", since only tech-savvy experts have the needed technological and programming skills. Therefore, enabling different-skilled users to autonomously create IoT software systems represents an important added value for an IoT-based architecture. Two examples of this trend are presented in [11, 12], where authors propose a distributed architecture that allows to graphically create and control mash-up applications, without specific knowledge on both hardware and programming languages. Finally, in [13], 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.

Taking into account all these considerations, in this paper, an architecture able to address the above described issues has been designed. More in detail, the proposed system allows a transparent access to the heterogeneous IoT technologies through a multi-protocol middleware that hides the heterogeneity of the underlying technologies. Its structure is designed to be easily extended to new technologies, in order to improve flexibility and scalability. Then, the system provides a distributed indoor location service to supports location-aware applications, which manage the status of the environment in accordance to both user-defined rules and users' movements. Both the location-aware services and the multi-protocol middleware are deployed in a home gateway. Finally, it provides simplified tools to support users in 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 Smart Home environment. As proof-of-concept, the first steps of the implementation of the architecture are described.

The rest of the paper is organized as follows. Section II provides an overview on the main design challenges. A detailed description of the proposal is presented in Section III. In Section IV, a proof-of-concept is described. Conclusions and future works are summarized in Section V.

II. Design Challenges

The design and development of the described architecture involve addressing several challenges, as explained below:

- The multi-protocol middleware has to communicate with 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 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 of the architecture manage the home environment on the basis of the user location, the system has to provide a service that continuously tracks, in the background, the location of people moving in the house, so as to make this localization information available to the other services. Therefore, it is necessary to identify a low-cost and low-power technological solution able to perform this task with the required accuracy.
- 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., when the user want to remotely act on his/her home system). 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 the users with simplified tools, such as graphic editors or step-by-step wizard, to easily define both new services for the home gateway 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 the home gateway and the user's smartphone.

III. Architecture Design

Fig. 1 shows the overall structure of the proposed system. 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 distributed location service 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.

A. Development tools

The development tool available to the user is substantially composed of three main sections, in a wizard-like fashion. In the first section, the user can define the physical devices installed in the Smart Home. The basic idea is to model a device as a set of elementary functions, each of which is characterized by several attributes (Fig. 2): attributes for physical and network reachability (i.e., physical location and URIs/addresses), attributes that declare both the kind of the interaction accepted by the function (e.g., read or write commands) and the type of the expected payload (e.g., numeric, string, boolean, etc.), and attributes that define the main feature of the function (e.g., temperature indication, switch actuation, dimming actuation, etc.). This parametric

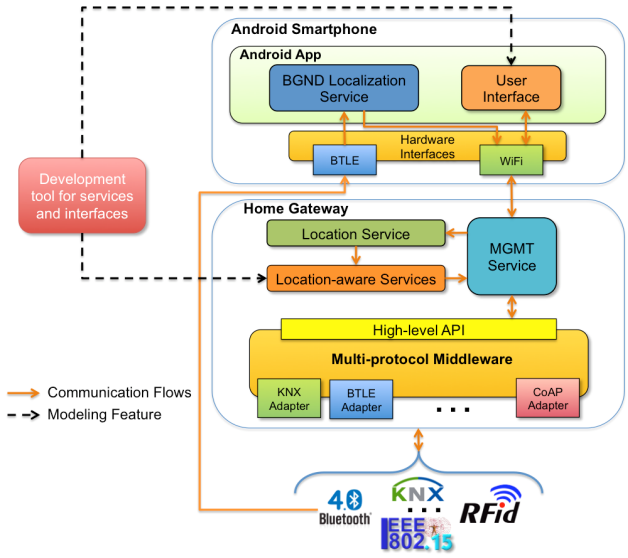

Fig. 1. The overall system architecture.

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. Indeed, this "technology code" is then used by the multi-protocol middleware to map the high-level requests, coming from the services and user interface, 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. More in 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.

In the second section, the user can define the business logic of the services that manage the Smart Home. 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.

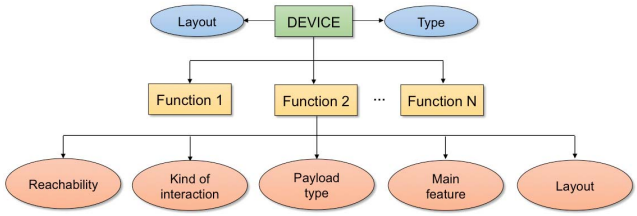

Fig. 2. The basic model of a generic home device.

Finally, in the third section of the development tool, the user can build the graphical user interface of the mobile application used in H2M communications. 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. The flexibility of this approach allows to have a highly customized navigation structure. 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 fallen on a RESTful interface, since the Representational State Transfer (REST) paradigm [14] is actually the core interaction

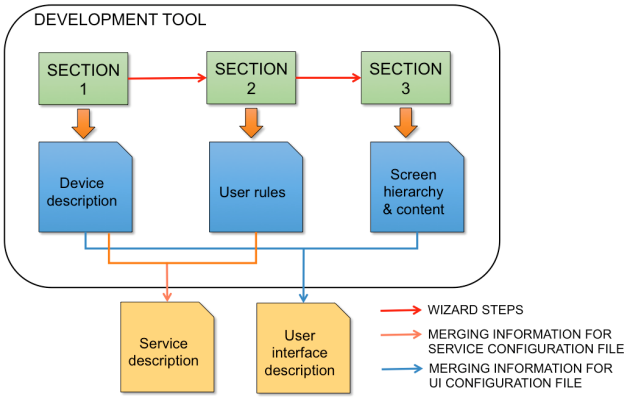

Fig. 3. The overall view of the development tool.

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 Smart Homes, 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 system includes both location-aware services managing the home environment and the distributed location service that tracks users' movements. Each location-aware service is defined through the development tool described in Section III.A and it is actually instantiated when the multi-protocol middleware receives the configuration file that describes the service. Since these services have to run continuously, they are basically infinite loops in which the user-defined rules are cyclically evaluated. More in 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 of the system depend on the location service, it is definitely the more 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 mobile device located on the user 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. In 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 located within its listening range and then determines the room in which it is located. To do so, it computes a proximity index d, for each landmark, using the corresponding value of the RSSI:

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

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 [15]. 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 steps 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 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 this graphical interface is highly customizable, since it is built by parsing the configuration file generated by the user through the development tool. More in 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 screen of the application, 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

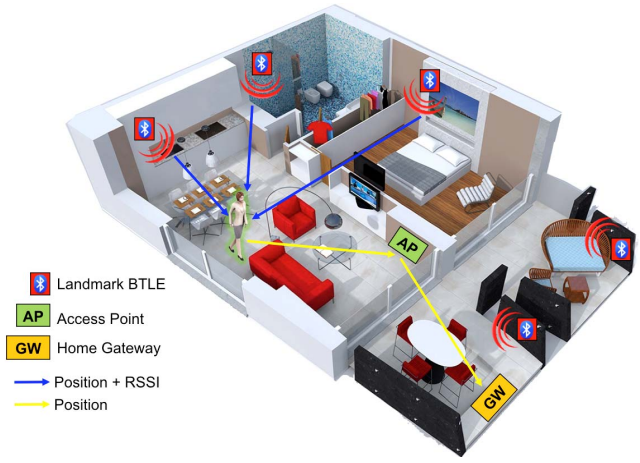

Fig. 4. The main steps of the user localization process.

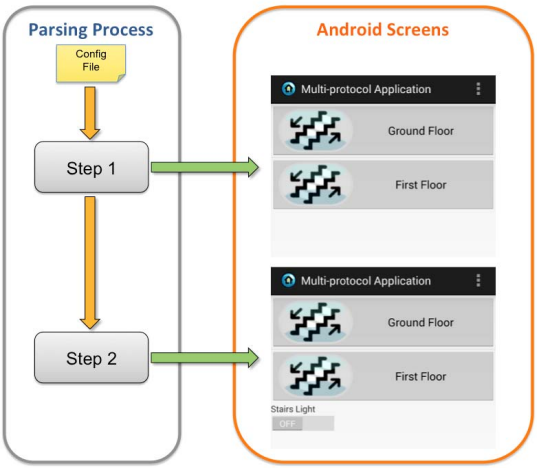

Fig. 5. The main steps of the parsing process.

Fig. 6. Test environment.

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.

About the user application, it is important to highlight another interesting aspect. Since this application bypasses the business logic of the location-aware services, it is essential that no conflicts be generated with the messages sent to the system 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 status of the home devices is exclusively managed through the mobile application.

IV. 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 [16] and Constrained Application Protocol (CoAP) [17], the distributed location service, and the development tool that allows to define the interface of the mobile application (but that does not yet allow the definition of 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 Wireless Sensor Network (WSN) nodes on which a CoAP thin server is running, i.e. a minimum and lightweight server exposing the physical devices through a RESTful interface.

The initial choice of KNX and CoAP is due to their diffusion in both commercial and academic solutions available in the literature. KNX is the worldwide standard for home and building control, whereas CoAP is one of the most used application protocol in the IoT, which provides a lightweight access to physical resources. However, as said, any other technology that characterizes a Smart Home can be integrated into the system. The software tools used to build the KNX and CoAP adapters are Calimero [18] and Californium, respectively. The first one is a Java library that provides a collection of APIs for interacting with KNX-compliant devices, hiding the network protocol details. The second one is a modular CoAP implementation written in Java that allows easy creation of CoAP-based client and server applications.

Regarding the distributed location service, the network of wireless landmarks is realized by transforming several Raspberry Pi boards in iBeacon (called in this case piBeacon), that is, low-powered and low-cost transmitters, developed by Apple, able to notify nearby mobile devices of their presence. To realize these piBeacon boards, a BTLE dongle is integrated into each Raspberry Pi [19], whilst the BlueZ stack [20] is exploited to provide them with the iBeacon 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 is responsible to collect this information in order to calculate its position, as described in the Section III.C. 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 interested in the position of certain users. In this way, it can promptly notify such services about the movement of the target users.

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. More in 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 can exploit the other section, which allows to build the screens hierarchy and their content. Also this section leads to 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

```
<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.

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, in order to actually instantiate the mobile application, an appropriate parser has been implemented in the Android smartphone to process the configuration file. The main task of this component is to perform the parsing process described in Section III.D.

V. Conclusion and Future Work

In this paper, an architecture that allows the transparent management of the future Smart Homes was presented. This capability is provided by a multi-protocol middleware that hides the low-level communication details to the upper layers. In addition, the system manages the interaction among physical devices by automatically changing the state of the environment in accordance with both user-defined rules and users' location. To trace people movements, the architecture exploits a location service based on Bluetooth Low Energy, which ensures the proper level of accuracy, whilst being a low-cost and low-power solution. In addition to the M2M communication paradigm, the architecture also allows a H2M interaction, in order to provide the user with the ability to directly manage his/her home. Finally, the system provides a simplified development tool that enables any user to create new software solutions without any in-depth knowledge of specific technologies or programming languages.

As future works, we first expect the final development of the basic features of the architecture. Next, some critical situations need to be addressed, such as the potential inconsistency among the rules of concurrent services. Then, to increase the system appeal, 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.

Acknowledgment

This work is partially supported by project "PON04a2_D - DICET LivingLab Di Cultura e Tecnologia – INMOTO – OR.C.HE.S.T.R.A.", funded by the Italian Ministry of University and Research (MIUR).

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. Catarinucci, S. Guglielmi, L. Mainetti, V. Mighali, L. Patrono, M.L. Stefanizzi, and L. Tarricone, "An Energy-Efficient MAC Scheduler based on a Switched-Beam Antenna for Wireless Sensor Networks," Journal of Communication Software and Systems, vol. 9, no. 2, pp. 117–127, June, 2013.

[3] 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, pp. 88–104, 2014.

[4] 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. International Conference on Software, Telecommunications and Computer Networks, 2013, pp.1-7.

[5] M. Jung et al., "A Transparent IPv6 Multi-protocol Gateway to Integrate Building Automation Systems in the Internet of Things," in IEEE Int. Conf. Green Computing and Comm., Besancon, 2012, pp. 225 – 233.

[6] 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 Serv. in Ubiquitous Comp., Palermo, 2012, pp. 683 – 688.

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

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

[9] 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. Int. Conf. on Awareness Science and Tech., Seoul, 2012, pp. 109-114.

[10] M.V. Moreno, J.L. Hernandez, and A.F. Skarmeta, "A New Location-Aware Authorization Mechanism for Indoor Environments," in Proc. 28th Int. Conf. on Adv. Inf. Net. and App., Victoria, 2014, pp. 791-796.

[11] L. Mainetti, V. Mighali, L. Patrono, P. Rametta, and S.L. Oliva, "A novel architecture enabling the visual implementation of web of Things applications," in Proc. 2013 International Conference on Software, Telecommunications and Computer Networks, 2013, pp. 1-7.

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

[13] 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 Comp. Net., vol. 64, 2014.

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

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

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

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

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

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

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