Smart Home IoT System

Irina-Ioana Pătru, Mihai Carabaş, Mihai Bărbulescu
University POLITEHNICA of Bucharest
Bucharest, Romania
Emails: ioana.patru@cti.pub.ro, mihai.carabas@cs.pub.ro, mihai@roedu.net

Laura Gheorghe
Research and Development Department
Academy of Romanian Scientists
Bucharest, Romania
Email: laura.gheoghe@cs.pub.ro

Abstract— **In recent years, there has been a huge growth in the world of intelligent devices for home automation. Such gadgets are designed in order to ease the interaction between people and daily home duties. Although individually they are simple to work with, each appliance has its own configuration interface which adds overhead to the general user experience.**

This paper presents a solution for connecting more devices into a signal entity which can be easily accessed at any time. The implementation integrates the functionalities of different home automation devices into a single application.

Keywords—smart; automation; remote; security; communication; sensor

I. Introduction

Home is the place where personal and confidential information of each individual can be found and it represents one of the greatest investments in life. It is an essential part of people's lives and an improvement in this area means more comfort for any individual.

People always try to find new methods to increase their comfort. This includes ideas for making daily tasks easier or even eliminating some of the duties. Nowadays people can install smart appliances inside their homes in order to control some of the house tasks [1]. This type of intelligent devices have the possibility of remote control, which eliminates the necessity of being near the device.

Besides daily home tasks like changing light color or room temperature, house security is another aspect. Since it can deposit personal and valuable objects and documents, it is natural that people want to protect it. Classic mechanical door locks have been used starting from Ancient Egypt.

The typical mechanism is now adapted to modern days. But these new methods for controlling what happens inside the house need to be easy to use and understand. People need ways for simplifying their actions and not only adapting them to modern world. The smart devices must be easy to integrate in the home environment.

There is no clear definition for Internet of Things (IoT). Despite this, it is one of the most popular topics and many companies invest for research and development in this area [2]. At the same time, more and more connected devices are planned to be released. This influenced the growth in IoT.

Since it is rapidly evolving and is present in many products, there is a need for understanding its meaning and challenges. The term refers to connecting devices that are not directly controlled by humans to the Internet [2]. This type of devices are "things". By connecting such devices, an intelligent and invisible network is created [3], which can be accessed through the cloud.

Besides the features which each device is capable of implementing, the network must offer an interface. The smart devices are controlled and programed remotely through that interface. All IoT products benefit from embedded technology which allows them to communicate between each other or with the users through the Internet [4].

Intelligent appliances have been produced in many fields. Almost each day a company announces the release of a new IoT enabled product [5]. Some of the most popular applications are present in wearables, connected cars, smart healthcare and smart farming. Consequently, through IoT every individual can monitor their own activities with wearables and smart medical devices or they can control cars using connected car solutions.

Smart home is another IoT application which stands out as the most active field. A statistical report from 2015 states that more than 60000 people search this "smart home" term each month and more than 250 companies and startups invest in this area [5].

This paper presents a solution for connecting more types of home appliances like Nest Thermostat [6] and Philips Hue Light Bulb [7] in one system which can be remotely accessed. The main advantage of having a gateway is that all configurations are done on a single device. In addition, mechanisms for ensuring security are then necessary only on that device.

The solution proposed in this paper offers the modular implementation of a new open source smart home application. Although there are other projects which address this field, the system presented in this paper is easy to be integrated inside a real house. The proposed design offers users the possibility of easily adding new functionalities.

The next sections are structured as follows. In section 2 is presented the state of the art, other projects which promise to create a smart home environment. The design of the solution is described in section 3. Implementation details are given in section 4. Section 5 presents the results and gives an evaluation of the smart home system. The conclusions are given in the last section, which makes a summary of the presented system and lists possible further improvements of the system.

II. State of the Art

Home automation is one of the most common IoT applications. Since it also improves people's lives, many research papers have addressed this idea. From open source standards, to proprietary protocols and applications, there are various implementations that support the smart home concept.

A. Smart Home Protocols

The Open Interconnect Consortium (OIC) tried to create an open source universal project called IoTivity [8]. Other technologies can be easily integrated in this project using plugins. As a result, many types of devices with different communication methods can interact with each other through IoTivity. The design's objective is to create a standard used by both wireless and wired devices in order to communicate over the Internet. This can facilitate the development of smart home projects.

IoTivity tries to ensure that each new smart device can connect into the IoT ecosystem. Since it is an open source project, it encourages developers to contribute and extend the framework with the necessary information for connecting all types of devices in the proper profile like Consumer, Enterprise, Automotive and Health [8].

The IoTivity API framework is based on a RESTful architecture model [8]. It contains four blocks that make communication possible:

- Discovery – supports multiple methods for device discovery, both in proximity or remotely;
- Data transmission – communication based on messaging and streaming model;
- Data management – storage and computation or data;
- Device management – configuration of devices.

Therefore, with this API multiple technologies for transporting information can be incorporated. For example they can be based on Bluetooth, Wi-Fi or cloud communications.

There are also protocols that allow smart devices to communicate with each other and over the Internet. Examples in this area include ZigBee and Z-Wave [9]. These are wireless technologies designed for remote control. Although the idea is similar, they differ in the frequency they use, range of transmission and data rate.

ZigBee is an open wireless standard based on IEEE 802.15.4 network radio standards. The difference between the two standards is that while the last one provides Layer 1 and 2 communication (physical layer and media access control), the other one addresses network and application layers [7]. Therefore, ZigBee offers a software stack for higher levels.

It can either be used for direct connections between devices or in networks based on a start or tree mesh topology. In these scenarios one of the entities acts as a coordinator and dictates the communication between the other nodes. If two nodes cannot communicate directly, then they can pass the information through other entities until it reaches the destination. The maximum range between two nodes is of 10 meters.

The main advantage of this standard is its flexibility. Specific software is developed to work with ZigBee. This is known as a profile and can be integrated by developers into their products. ZigBee stack already includes a home automation profile.

Unlike ZigBee, Z-Wave is not an open standard. It is only available to Zensys/Sigma Design customers [9]. The connection between nodes is based on the same idea, but it uses different ranges. Although ZigBee supports more nodes (65000 vs 232), Z-Wave has a higher range of communication between them of up to 30 meters.

Another difference is that Z-Wave is a proprietary wireless standard which functions at the physical and media access control layers. This is achieved by incorporating a physical module into the products. Z-Wave is primarily used for monitoring and controlling home functions like lights, locks, temperature or security sensors.

ZigBee and Z-Wave have the same IoT target application. Between these two solutions, ZigBee is the one preferred by developers [9]. The available profiles minimize the product development time. The disadvantage of both protocols is that developers need a high number of devices in order to get good coverage since their ranges are low.

Tudose et al. [10] used the 6LoWPAN protocol on top of IEEE 802.15.4 running on Sparrow sensor nodes in order to build a smart home monitoring system. The sensor nodes transmit periodically the collected values through UDP/IPv6 connections to the gateway device.

B. Smart Home Applications

An example of a smart home system is Apple HomeKit [11]. Using this, home devices can be controlled by a smartphone running iOS. There is no need for an additional device which acts as gateway and transferred data is encrypted.

HomeKit benefits from over 50 brands of products with software integrated [11]. Some of them are found in fields like lightning, locks, switches, temperature controls and even shades. After enabling them on the smartphone using the corresponding software from App Store, the user can control them and automate their behavior. They can set scenes and even give direct voice commands using Siri.

In this area of smart home products, a significant percent is represented by security gadgets. Danalock Bluetooth Z-Wave Smart Lock [12] is a door locking appliance which can be easily installed into the deadbolt hole. With this, the door can be opened from a smartphone, working either with Android or iOS enabled devices. It also has the possibility of granting permission to other persons to unlock the door. Another locking solution is offered by Oplink Connected CMPOPG2204OPL01 Alarm Shield [13].

Samsung offers a kit which combines more functions for home control and monitor. It is called SmartThings Home Monitoring Kit [14] and it includes gadgets for traditional home monitors to appliances that can be controlled remotely from a mobile device. Therefore, it has cameras, smoke monitors and even lights control systems.

Many other security appliances have been developed and most of them use cameras for home monitoring like Dropcam Pro WiFi Wireless Video Monitoring Camera, Belkin NetCam HD Wireless IP Camera or D-Link – cloud Camera 5000 [15].

These are all products on the market, but this paper describes the implementation steps and advantages for a proof of concept open source solution which also integrates home security automation.

III. SMART HOME DESIGN

At the basis of the design for this smart home system stands the idea of making an easy to use product which integrates more types of tasks inside a house. It has a simple interface with a few configuration steps. The user connects to the cloud, sends the desired commands to the smart home gateway from the house and the corresponding physical component it activated. This generic flow is presented in Figure 1.

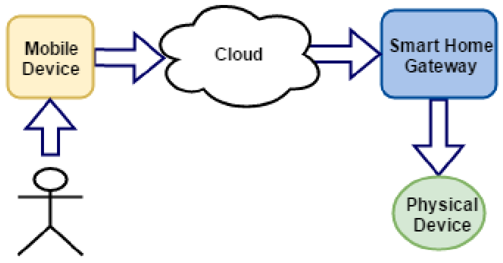

Fig. 1. User Scenario

The interaction between the user and the cloud is done using an Internet browser that can be installed on any mobile device like smartphones, tablets or laptop. The user can choose which type of devices it prefers on the client side.

The communication between the elements from the house is made in two ways. On the one hand are the directly connected elements: sensors, buzzers and motors. On the other hand are the smart products: the Nest Thermostat [6] and the Philips Hue light bulb [7]. Both types of connections are described in Figure 2. The elements shown in the figure are detailed in the next section.

Each smart device has its own way of sending information to the gateway. The light bulb connects to the network through a Philips Hue Bridge. It is used in order to change color, brightness and saturation of the light.

The change in target temperature is monitored through Nest Thermostat. This is a smart device which can receive information from exterior and then announce a Heat Link that it needs to adapt the heating source. Usually a Heat Link is directly connected to the component of the house which can change the temperature, the heating central system. The thermostat connects directly to the internal Wi-Fi. The communication between Nest Thermostat and the Heat Link is done through a Nest protocol.

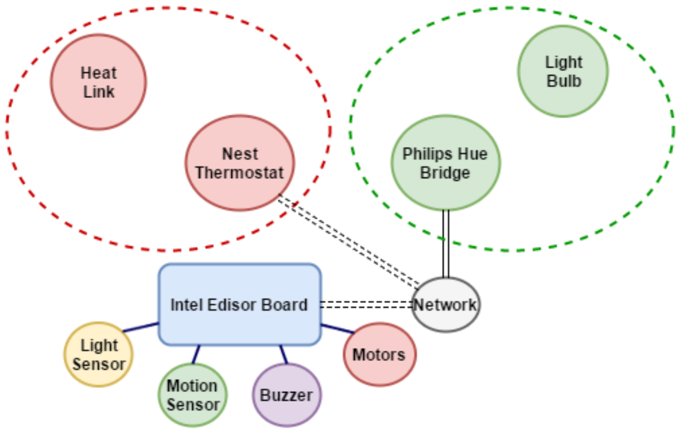

Fig. 2. Communication Design

In the middle of all the mentioned elements stands the gateway. In the described solution, this is represented by the Intel Edison board [16] which is a powerful board that can be easily integrated with open source hardware. This small device is powered by an Intel Atom processor, benefits from 1 GB RAM and 4 GB EMMC storage, integrates Wi-Fi and Bluetooth and supports I2C, SPI and GPIO signals.

Besides its basic characteristics, the board can be connected with Grove Starter Kit [17] since it is Arduino compatible. The kit includes the sensors, shields and connectors that are used in this project: sensors for light and motion detection, buzzers for alarm trigger and motors for the lock system

The application of this smart home system runs on top of the Brillo operating system [18] from the Intel Edison board. This is a newly launched embedded OS and it is developed by Google. It has been designed especially to facilitate the development of products in the IoT world. It integrates core services, system Hardware Abstraction Layers (HALs) and Weave [19]. This protocol is a secure communication mechanism used for connecting the board with mobile devices.

The application's architecture is illustrated in Figure 3. As it can be seen, it is separated into multiple modules which are responsible for implementing different features. The application is the one in charge of communicating with the sensors, buzzers, motors and smart devices. On the other side it connects with the owner of the house through the cloud.

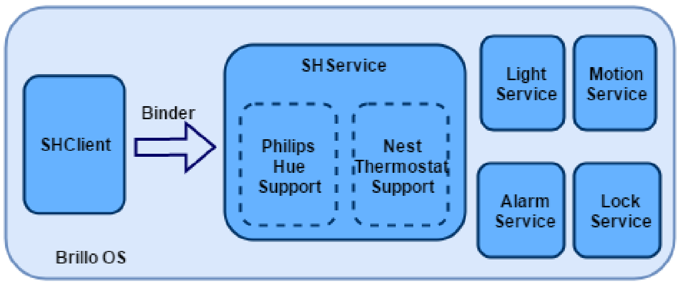

Fig. 3. Application Architecture

The main service, SHClient, is responsible for coordinating the other elements and for connecting with the cloud and, in this way, with the user. Therefore, this service which needs to have the architecture for exporting the smart home possible commands. The other services implement the home related functionalities of the application. They export their methods only to the client service.

The available functions of the application are following:

- Register the application in the cloud;
- Receive user commands for doors/windows locks;
- Receive information from sensors and trigger alarms;
- Change motor state;
- Set light color, saturation and brightness;
- Set target temperature of the house;
- Send current house status to the owner.

IV. SMART HOME SYSTEM IMPLEMENTATION

The application running on Brillo is composed from five services written in C++ that interact with each other in order to achieve the desired security mechanisms.

A. SHClient

SHClient is the service that collects information from all other services in order to send information to the user. This component is also responsible for registering the application with the Weave protocol and receiving commands from the cloud. The communication through Weave is secured using OAuth 2.0 Authentication.

Registering the application in the cloud involves passing a JSON schema through Weave. This schema contains information for all available commands that a user can send remotely and all states that this security product can show. The JSON is parsed in the cloud and then is used for updating the fields for the Weave enabled web applications on the client side.

In the schema each command that this security product exposes is specified through a given name, number of arguments and their types. All commands are associated with methods from SHClient. Therefore, the JSON is the external interface of the product. When the user sends such a command from the interface, SHClient verifies the received arguments and passes them to the corresponding implementation. An example of such flow can be seen in Figure 4. In this scenario the user sends from their mobile device a command for locking the door. In SHClient, the command is passed to the corresponding method which activate the motors at home.

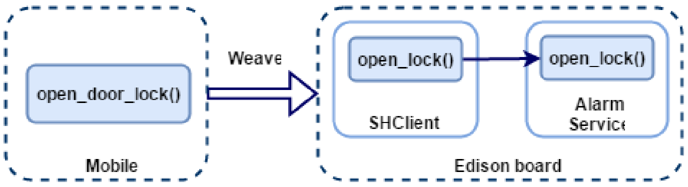

Fig. 4. Example of a command flow

The list of commands that the other services implement is available through a common library libsh-common. This library contains an interface written in AIDL [20] that is updated with information from all involved components in this project. If another functionality is added, then this library is updated with the new commands. Libsh-common contains public information from all the involved services.

B. SHService

Another service is the one which contains the implementation for the smart appliances communication, SHService. In the first place this is achieved by integrating libcurl [21] in order to use the REST interface of the smart devices. Exposure of the available commands to the other service is done through the Binder interface mentioned earlier which is written in AIDL [20].

The implementation separated the two smart devices functionalities into two components. There is a separate class, PhilipsHue, which implement the light bulb features and another one, NestThermostat, which controls the target temperature.

The registration parameters for Philips Hue and Nest Thermostat are read from configuration files inside the application. These files need to be updated when the parameters expire. For the light bulb the user must configure: bridge IP address, registered user name and the light ID that they want to change. On the Nest side, the thermostat device ID is required and the user authentication token.

After ensuring that all parameters are correctly configured and the connection to the smart appliance can be done, the HTTP header can be composed. For both devices it consists from PUT methods.

C. Sensors Services

The Light Service is the one responsible for detecting changes in light. It constantly checks the light value and it triggers the alarm if it exceeds a certain threshold. This project uses the sensor from Grove Starter Kit, the light dependent resistor (LDR) [17].

In the same manner, the Motion Service announces when it detects a modification in the house. It receives information from the PIR Motion Sensor from the Starter Kit [17]. This sensor can adapt its detecting distance from only a few centimeters to almost 6 meters.

The service receives the sensors list for the HAL and identifies the desired sensor based on its type. For light, it needs to match SENSOR_TYPE_LIGHT and for the motion detector one SENSOR_TYPE_SIGNIFICANT_MOTION. These are definitions from the HAL. The libraries upm [22] and libmraa [23] are used by the HAL to contact the hardware elements.

Upm provides software drivers of commonly used sensors. These drivers interact with the hardware elements through libmraa API. This way, sensors usage in application development is very simple. Programmers need to include the corresponding header file, instantiate the sensor class they need and then manipulate the object.

The next step is to activate and open the chosen sensor. Then, it starts retrieving data. When it detects changes greater than the default threshold, it means that an unexpected activity is happening. So, it triggers the buzzer from the alarm service.

D. *Alarm Service*

There is a separate service which implements the alarm mechanism. This Alarm Service is used for controlling the buzzer of this project. It has methods for instantiating the Buzzer object from the upm [22] library which sends information to the hardware device from the Grove Starter Kit.

When the other sensor services decide they need to trigger an alarm, they call a StartAlarm() public method from this service which opens the buzzer. It can play either for a certain amount of time or indefinitely, depending on the arguments passed to the buzzer.

E. *Lock Service*

For implementing the lock system, this service also uses objects from the upm library. It needs StepMotor and GroveRelay in order to trigger a motor movement. The two objects control the stepper motor and relay hardware from Grove Starter Kit.

This locking service initializes the objects from HAL, sets the speed and specifies the direction in which the motor move: either forward or backward. This is the basic idea for how it could lock and unlock a real door or window.

As it has been presented in this section, each service contains the implementation necessary for controlling their specific hardware component or smart device. All services have their own .rc configuration file which mention their name, where they are located and when they should start. In this manner it is ensured that the operating system has finished all the necessary initializations before staring the smart home services.

The smart home application running on Brillo does not have access to any resource from the system like storage or the sensors' drivers. Every required access with a hook in the kernel has to be explicitly allowed through policies [24]. These are specified for each service and are placed in .te files under sepolicy directory from the application. Using this method, the services have access to only the resources they need.

V. Results and Evaluation

Throughout this paper it has been described how to obtain a smart home project which integrates the functionalities of two intelligent appliances, Philips Hue light bulb and Nest Thermostat, and also several types of sensors, buzzers and motors. This system can be controlled from a mobile device with Internet access.

The resulting miniature smart home system is represented by a concept house with small walls made out of cardboard. All the mentioned elements are combined inside the cardboard walls in order to reproduce the smart home features. This is the proof of concept which presents how more types of functionalities from a house can be integrated into a single gateway, the Edison board. It exposes all the available commands in the cloud.

Since it is an open source project, any individual can integrate this system inside their house without having to pay for it. The motors can be attached to doors and windows in order to lock or unlock them and the sensors can be placed in sensitive places inside the house where they could detect unusual events.

Using this project, the user must configure only a single application for setting up the smart home environment. Although some of the appliances used have separate configuration applications on the market, the usage is simplified with the solution presented in this paper. In addition, it offers the possibility of remote home control.

The board application runs on a fast and light operating system designed especially for embedded products. It starts in a few seconds so the user can immediately use its features by connecting the smart appliances and then controlling them. The configuration steps include Internet communication and obtaining the secure parameters for the gadgets registration. Then, users can start exploring the smart home application features.

This solution offers security as well by using Weave protocol for all exchanged data between the board and the web client application. Users must register and then authenticate before accessing the smart home product that is associated with their account. Such a product must be assigned to only one user and then it can be shared with others if this is desired by the main account.

The outcome of the proposed home solution has been analyzed based on the interaction between the user and the end elements: sensors, buzzers, motors and smart appliances. The chosen components behaved as expected during tests. This is a proof of concept implementation of a miniature smart home system which can be integrated inside a real house.

The light sensor is able to detect whether the lights have been opened or closed at any time of the day. Besides this, the value received from this sensor can also inform the owner of the house if they forgot to close the lights.

Detection of other presences inside a room is achieved through the motion sensor. It has an area of impact for up to 6 meters, which is more than enough for a standard room if it is placed appropriately.

Regarding the Philips Hue light bulb usage, there are four parameters which can be changed. One is related to the on and off states of the light and the others reflect the ambient color: saturation, brightness and hue.

On the Nest Thermostat side, the user can set target temperature. As a result, they can remotely specify to what value the temperature should be at the time they get home. This adds more comfort to the home environment.

With the help of this smart home project, the user has a single application for monitoring light, temperature or other presences inside the house and controlling their door and windows locks. In addition, this solution triggers an alarm when unusual activities are detected.

The application which integrates all functionalities starts in a few seconds after the Edison board gets powered on. This is the result of using Brillo, a lightweight operating system that was developed for IoT products.

On the client side, any type of operating system can be used. The requirement is to support an Internet browser and to have access in the cloud. After the user authenticates in the Weave Developer Console, it can immediately start to control the home security application.

Therefore, using the smart home product described in this paper improves daily home activities without raising security concerns. Users' home experience gets better with minimal configuration steps. It includes different functions meant to cover many aspects present at home which could be remotely controlled.

VI. Conclusions and Further Improvements

The paper proposes a proof of concept for a smart home system that can be easily adapted to a real house. The miniature house presented in this paper gives a solution for implementing a security mechanism that can be remotely controlled. Besides this, it integrates the functionalities of two intelligent devices for changing light parameters and room temperature through a single interface.

The application can be accessible from any kind of mobile device with an Internet browser access and the correct access rights. This adds flexibility and eases the interaction with the presented application. Since the interface of the application is easy to understand, this project can be used by both system engineers and people with little knowledge in technology.

This paper explains how the smart home mechanism is implemented. Here it is described what physical components it needs, the communication between them, the architecture of the application and details of implementing the mentioned functionalities.

There are other house security systems on the market that also incorporate doors and windows locks, sensors for detecting presence and devices that activate alarms. An advantage that this proof of concept gives is the fact that it is an open source project which can be easily adapted to control elements in a real house.

Regarding the smart devices it uses, this system has another advantage. It creates a single interface for the management of both devices, the thermostat and the light bulb. Therefore, the user needs to access a single application in order to configure their home.

As a conclusion, this paper shows how to connect smart devices into a single entity using Brillo operating system on an Edison board. The presented idea has a flexible and extendable configuration which can easily be adapted to new appliances and features.

Future development of this smart home solution can be concentrated on more areas. On the one hand it could integrate more types of intelligent appliances in order to incorporate various kinds of home activities. In addition to this, on the board device can be developed more features for the currently added devices. These include setting default scenes and themes that automate the smart appliances duties from home based on user's configurations.

The security solution can be integrated into actual products with other functions besides the available ones from the sensors. Users could control their home security without having to install specialized devices from security companies.

Acknowledgements

This work has been funded by program Partnerships in priority areas – PN II carried out by MEN-UEFISCDI, project No. 47/2014.

References

[1] J. Stragier, L. Hauttekeete, L. Marez, Introducing Smart Grids in Residential Contexts: Consumers' Perception of Smart Household Appliances, Belgium, pp. 1-2, 2010

[2] P. Waher, Learning Internet of Things, Birmingham, pp. 1-3, 2015

[3] J. Chase, The Evolution of the Internet of Things, Texas Instruments, Texas, pp. 1-3, 2013

[4] D. Zhang, L. Yang, H. Huang, Searching in Internet of Things: Vision and Challenges, Busan, 2011

[5] K. Lueth, The 10 most popular Internet of Things applications right now, https://iot-analytics.com/10-internet-of-things-applications/, pp. 1, February 2015

[6] Nest Developers Documentation, http://developers.nest.com/, Last Access: April 20th 2016

[7] Philips Hue Developer Program, http://developers.meethue.com/, Last Access: April 20th 2016

[8] IoTivity, http://www.iotivity.org/, Last Access: April 24th 2016

[9] Lou Frenzel, Electronic Design, What's the difference between ZigBee and Z-Wave?, pp. 1, March 29th 2012

[10] D. S. Tudose, A. Voinescu, M.-T. Petrareanu, A. Bucur, D. Loghin, A. Bostan, and N. Tapus, "Home automation design using 6LoWPAN wireless sensor networks," in 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011, pp. 1–6.

[11] Apple HomeKit, http://www.apple.com/ios/homekit/, Last Access: April 24th 2016

[12] Z-Wave Smart Lock, http://danalock.com/, Last Access: May 2016.

[13] The Oplink Connected Alarm Shield, http://www.oplinkconnected.com/, Last Access: May 2016.

[14] Samsung SmartThings Home Monitoring Kit, http://www.samsung.com/, Last Access: May 2016.

[15] 50 Best smart home Security Products, http://safesoundfamily.com/blog/50-best-smart-home-security-products/, Last Access: May 2016

[16] Hardware Guide, Intel Edison Kit for Arduino. February 2015

[17] What is Grove Starter Kit Plus – Intel IoT Edition? http://www.seeedstudio.com/, Last Access: May 20th 2016

[18] Brillo, http://developers.google.com/brillo/, Last Access: April 15th 2016

[19] Weave, http://developers.google.com/weave/, Last Access: April 15th 2016

[20] Android Developer, Android Interface Definition Language Guide. http://developer.android.com/guide/components/aidl.html/, Last Access: April 16th 2016

[21] Libcurl API, http://curl.haxx.se/libcurl/c/, Last Access: April 16th 2016

[22] Upm Documentation, http://iotdk.intel.com/docs/master/upm/, Last Access: April 20th 2016

[23] Mraa, http://iotdk.intel.com/docs/master/mraa/edison.html, Last Access: April 20th 2016

[24] SELinux, http://source.android.com/security/selinux/implement.html, Last Access: April 29th 2016