Lightweight Architecture for IoT Devices with Context-aware Autonomous Control

Bruno Serra
GECAD
Polytechnic of Porto (ISEP/IPP)
Porto, Portugal
1151400@isep.ipp.pt

Luis Gomes
GECAD
Polytechnic of Porto (ISEP/IPP)
Porto, Portugal
https://orcid.org/0000-0002-8597-3383

Zita Vale
GECAD
Polytechnic of Porto (ISEP/IPP)
Porto, Portugal
https://orcid.org/0000-0002-4560-9544

***Abstract*—The integration of the Internet of Things (IoT) devices in our buildings is already a reality and the dissemination of such devices would grow in the future. However, much of these devices deal with remote monitoring and/or control without an intrinsic context-aware control. This paper proposes hardware and software architectures for the development of IoT devices with context-aware autonomous control. The efficient and effective control, of the proposed architectures, is demonstrated using two scenarios where one television and one air conditioner unit are controlled according to their contexts. The context-aware control can increase users' comfort while decreases the use of appliances and resources, resulting in a decrease in energy consumption.**

Keywords — Internet of things, context awareness, autonomous control

I. INTRODUCTION

Internet of Things (IoT) devices are spreading in our homes, buildings, and cities, creating the opportunity to have smart homes, smart buildings and smart cities. In the current market is easy to find a vast application of IoT devices from door lockers to garden monitoring devices. However, these devices mainly provide remote monitoring and control.

The word 'smart' was been placed in front of old, and usually common, names, such as smart homes [1]. This concept promises to revolutionize our homes and the way we interact with them. Smart homes can improve, between others, home automation, living, and energy management. In [2] an environmental awareness smart plug with shared knowledge is proposed to decrease the energy consumption in a fully distributed system using resource automation. In [3] a system is proposed for elderly tracking, supporting people with dementia. In [4] an ontology is proposed as part of a constraint satisfaction problem for resource optimization in a smart home.

As stated before, the current market is full of these relatively new products that can bring a new life to our homes and pave the way to smart homes. Is expected that by 2022 a total of 216.9 million homes will have at least one smart device [5]. This is a market in expansion and therefore it is easy to find the biggest companies trying to get their piece. From the conventional smart plugs to smart rice machine of Xiaomi, almost everything is getting smarter.

The IoT devices in the market are many and increasingly cheaper. However, they have a problem regarding compatibility with each other. The current solution is the integration of all our devices in a centralized system. This demands another centralized system and does not guarantee the integration of all IoT devices. Smart voice control systems, like Google Home, Amazon Alexa, and Apple HomePod can usually do the job of aggregating multiple IoT devices from several manufacturers.

If an open source solution is required, is possible to use the Home Assistant platform that enables the integration of multiple devices and communication protocols [6]. Home Assistant does not provide voice control, but it allows the integration of Google Assistant and Amazon Alexa. An open source solution for voice assistant is also possible using Mycroft AI [7].

Another option to integrate several IoT devices is by buying everything from the same manufacturer, limiting the available market offer. By integrating several IoT devices, is possible to have context-aware control over some devices. For instance, is possible to turn off a heater's smart plug if the room's temperature rises above a set value. Using IFTTT service (If This Then That) is also possible to define some actions regarding context. However, all these solutions are centralized and somewhat limited, being basically *if* actions.

Context-Aware control and ambient intelligent systems can change the way we see systems. The ability to change their status according to its context provide a better fit within its environment. Moreover, systems with such capabilities can improve users' comfort and experience. An example of that is presented in [8] where an energy management system is proposed to minimize energy consumption by solving an optimization problem with multiple comfort constraints to maintain the multiuser comfort. In [9], is proposed a top-level architecture for IoT devices to provide services to users. Gateway-enabled architectures are proposed in [10] and [11], simplifying the development of horizontal platforms. Another approach is proposed in [12], where a decentralized architecture is used for resources' utilization optimization.

This paper proposes a hardware architecture for an IoT device with context-aware autonomous control over an electrical appliance/resource. The proposed architecture focus on simplicity and efficiency regarding context-aware control. This paper also presents the software architecture that should be implemented inside the processing unit of the hardware architecture. The software architecture provides the necessary layers to enable context-aware autonomous control. The main contribution of this paper is the proposal of a lightweight, functional and complete solution for an IoT device with context-aware autonomous control over an individual electrical resource.

In this paper, are presented two case studies where the hardware and software architectures are deployed. The two developed IoT devices were successfully deployed in one of our buildings. The deployment and results are presented in this paper. One case study will control the brightness of one television to reduce its energy consumption, while the other case study will control one air conditioner unit to prevent it from working without having persons inside its room.

After this first introductory section, Section II will detail both proposed architecture; the hardware and the software. Section III will describe the deployment of the two IoT devices used. In Section IV are presented the results of both deployments and finally, in Section V a discussion and main conclusions are presented.

II. Proposed IoT Device for Context-aware Autonomous Control

Market available IoT devices promote remote control and monitor, they usually are small pieces of a system that the user can expand and create. A single IoT device is usually limited in its functionalities but by integrating it in an IoT community, with other devices, then new control possibilities arise. This paper will go beyond the state of the art regarding IoT devices by proposing a generic architecture that enables the development of IoT devices for context-aware autonomous control.

An IoT device enables the integration of the physical world in the internet world. To provide this integration, a close collaboration between hardware and software must be built. Therefore, the proposed architecture is, in fact, a combination of one hardware architecture and one software architecture. The developed devices, shown in this paper, have these two architectures.

The proposed hardware architecture, shown in Figure 1, is divided into four main modules: a processing unit, sensors, an actuator, and a communication module. The software of the proposed IoT device is divided, as seen in Figure 2, in four layers: contextual data monitoring, context awareness (re)action, correction monitoring, and resource control.

The proposed architecture is able to obtain and deliver every information needed to create a context-aware autonomous control. The minimalist architecture enables the development of cheap and easy to use IoT devices.

A. Hardware modules

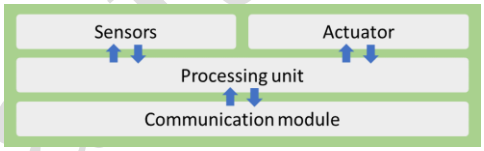

Figure 1. Hardware architecture

The main novelty of this architecture, when compared to market available IoT devices, is the integration of sensors and one actuator in the same device. This integration enables the context-aware control inside the hardware device and not in the cloud. Normally, market available solutions and scientific proposed solutions use the cloud-side to process heavy algorithms. However, the proposed work will function entirely inside the IoT device without needing a cloud connection.

In the centre of the architecture, it is found the main component of the device, the processing unit. The processing unit consists of a microcontroller, and it is responsible for the workflow of the device. Everything that involves information processing, receiving or sending information from or to some component, it is done inside the microcontroller. The choose of microcontroller must take into account the communication protocol intended for the IoT device and the sensors and actuator that will be integrated into the device.

The sensors module integrates all the sensors needed for the context-aware control. Each chosen sensor must make sense in the desired control. The sensor will measure, in real-time, values that the software will read and process in the processing unit. The measured values, provided by sensors, must be directly used in the context-aware autonomous control. Therefore, sensors have the objective of knowing what is happening on the device's surroundings.

The actuator module is responsible to control the desired resource, the type of actuator used will dictate the type of control allowed over the resource. There are a variety of possible actuators, they may consist on a simple relay to just turn on or off a circuit or they may consist on something more complex, it can use communication protocols like infrared emitters to provide a detailed control.

The Communication module provides the opening of the IoT device to the outside world and truly creates an IoT compatible device. In our work, we used the Message Queuing Telemetry Transport (MQTT) [13] protocol over TCP/IP. However, other protocols can be used, such as the Advanced Message Queuing Protocol (AMQP) or the Simple Text Oriented Messaging Protocol (Stomp). Other base protocols, different than TCP/IP, are also valid options, such as ZigBee or Z-Wave.

The Communication module, of Figure 1, is the base for modules integration. The Communication module works as a base while the others lay on its top. Sensors, Processing unit and Actuator work side by side to provide readings and actuation features. Although Communication is the most important layer to turn this into an IoT device, the focus of this paper is the ability to perform context-aware autonomous control. Therefore, the Communication layer will work to enable and disable the context-aware control; the focus of the paper will be the performance of the control itself.

B. Software modules

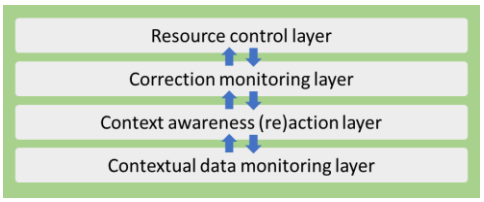

Figure 2. Software architecture

The contextual data monitoring layer is responsible to read and process all the data feed from the sensors. The interpretation of such signals is essential to provide the system with real-time contextual data that will be used in the context-

The present work has been developed under the EUREKA - ITEA2 Project M2MGrids (ITEA-13011), Project SIMOCE (ANI|P2020 17690), and has received funding from FEDER Funds through COMPETE program and from National Funds through FCT under the project UID/EEA/00760/2019 and SFRH/BD/109248/2015.

aware control. This layer is directly dependent on the hardware sensors used.

The resource management will be done in the context awareness action layer that interprets the sensors' data and defines the right control procedure. This layer should be built to enable the desired action. The action can be at least one of several types, such as partial control, power control or a warning system. However, this layer is tailormade for the desired functionality that the developer wants to give to the IoT device. In this paper, it will be shown two deployments, one for status control actions and another for safety actions.

The proposed IoT device as the goal to be integrated with a resource that can be controlled in a manual or external way; without exclusively using the IoT to control it. This way, our IoT device should monitor the controllable resource in order to detect, identify and react to the users' changes. The correction monitoring layer is responsible to identify the current controllable resource status. The monitor of the resource can demand hardware sensors that must be included in the hardware sensors module. If the device is unable to fully understand the resource's context or status the context-aware autonomous control will fail over time. The actions of the user must be considered and cannot be ignored in a context-aware system. Therefore, this correction monitoring software layer is extremely important to promote the right function of the system.

The resource control layer is responsible to control the actuator hardware in an efficient and coordinated way. Similar to contextual data monitoring layer, the resource controller layer will build the bridge between the software world and the hardware world. This layer will be used by the two middle layers: the context awareness (re)action layer, and the correction monitoring layer.

The software architecture, of Figure 2, was designed using hierarchical layers. Because the normal microcontrollers do not provide multi-thread, the layers are executed sequentially by their hierarchic order. It starts in the Contextual data monitoring layer and ends in the Resource control layer.

III. Deployment Scenarios

For the propose of this paper, two IoT devices, using the proposed hardware and software architectures, were developed and deployed in our facilities for uninterrupted context-aware autonomous control of two resources: a television; and an air conditioner.

Both deployment applications have equal processing units and communication protocols. However, the sensors and actuators are different from each application. The processing unit used is the NodeMCU Module [14] and MQTT is used as the communication protocol. This combination of the NodeMCU with MQTT protocol was successfully tested and used in [15].

The NodeMCU Module is based on ESP8266 Wi-Fi system on a chip from Espressif Systems. This is a low-cost module that integrates a processing unit with 128 kBytes memory and a Wi-Fi chip. The integration of processing unit and Wi-Fi chip is ideal for IoT devices. The NodeMCU Module is an open source platform that uses Lua as programing language and gives the developer the opportunity to create a custom firmware build using only the necessary modules (e.g. GPIO, MQTT, DHT or DS18B20) [16].

In our research centre, we have the open source MQTT broker Eclipse Mosquitto [17]. Therefore, the obvious communication protocol was to implement the MQTT over the TCP/IP protocol in the NodeMCU Module. The MQTT protocol, is each application deployed, is implemented as subscriber and publisher. The following communications are possible in each application:

- *Publish sensor data* – the data from each sensor connected to the IoT device is published in the MQTT broker, for each sensor is created a new MQTT topic;
- *Subscribe to control actions* – the actuator of the IoT device can be directly controlled by external users using the proper MQTT topic "control";
- *Subscribe to activation actions* – the IoT device context-aware autonomous control can be activated or deactivated by using the proper MQTT topic "activate".

Both IoT devices use infrared Light Emitting Diodes (LED) to control their controllable resources – mimicking the typical television remote control. However, the used NodeMCU framework timer function cannot provide a fast-enough clock to enable the direct connection and use of an infrared LED emitter. The framework can only provide a 1 kHz clock. To achieve the necessary frequency of 38 kHz, an astable multivibrator circuit must be created. Figure 3 shows the circuit that enables the use of infrared emitters with the NodeMCU module. The circuit uses an NE555 to work as the oscillator to reproduce the infrared signal.

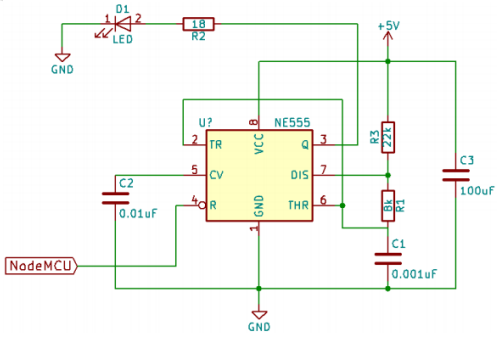

Figure 3. Astable multivibrator circuit for the infrared emitter

A. Television control

The proposed IoT device architecture was used to create a system able to control the brightness of a television (TV). The TV is used in our building to show the real-time building's consumption and generation, as well as some weather measurements. The TV is located in a hall and the idea behind the IoT device deployment is to minimize TV's consumption by monitoring the presence of persons inside the hall.

As a requirement, to prevent system failures, the IoT device will only control TV's brightness. The IoT device will not control the turn-off and turn on the TV. However, a smart plug, connected to the TV, is responsible to turn on the TV at 8:00 a.m. and turn it off at 8:00 p.m. during the working days. The used smart plug is a TP-Link HS110 with energy monitoring capabilities.

To control TV's brightness, the IoT device will change the TV's energy saving mode, from the minimum to maximum. The minimum energy saving mode uses a high brightness and produces a consumption of around 95 W. The maximum

energy saving mode will decrease the TV's brightness and consumption, where the consumption will be around 35 W.

Regarding software implementation, the flowchart of Figure 4 was implemented. The context awareness (re)action layer will put energy saving mode at minimum every time a person is detected in the hall and after 7 minutes without movement, the IoT device will change the TV to its maximum energy saving mode. When the PIR sensor detects someone in the hall, the IoT device sends a signal to change the energy saving mode, making it go to the maximum brightness mode (i.e. energy saving at minimum).

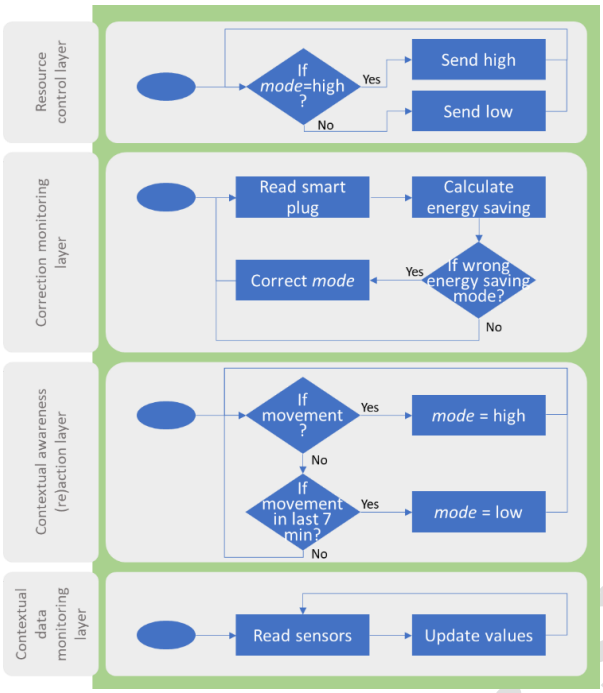

Figure 4. IoT device for television with the software layers

The correction monitoring layer will use the external energy sensor provided by the TP-Link HS110 smart plug. Using the TV's consumption, the IoT device can detect the current TV's energy saving mode and if the energy saving is not the desired, the IoT device can correct the TV's energy saving mode. The correction monitor layer checks TV's consumption every minute.

Figure 5 shows the TV's menu where energy saving is chosen. In our system, only two energy saving modes are used. The LG TV used did not provide a direct infrared signal to switch from the energy saving mode.

Figure 5. Television's energy-saving menu

To change the energy saving mode, the following sequence of signals are needed to be emitted by the IoT device:

i. *Open the energy saving menu* – this infrared signal, opens the energy saving menu (Figure 5);

ii. *Energy saving signal* – after the menu is open, the same infrared signal is sent repeatedly to change the selected energy saving mode, the IoT device will use the data from the smart plug to identify the current energy saving mode and then calculate the necessary repetitions to achieve the desired energy saving mode;

iii. *Ok signal* – after the right selection, it is sent the ok signal to activate the selected energy saving mode.

B. Air conditioner control

The IoT device for air conditioner control was developed for security reasons to prevent unnecessary consumption while protecting the air conditioner unit from continuous use. In our laboratories, the air conditioner units are old and sometimes students and researchers let the units working through the night, weekends and holidays. For more than once, problems with the units appear because of the continuous operation. To prevent future problems, an IoT device was developed and deployed to detect if the room is closed and then turn off the air conditioner unit.

The IoT device is only able to turn off the air conditioner unit when it is operating in the context-aware autonomous control. If the room is empty and dark for 7 minutes, and if there is some consumption produced by the air conditioner unit, then the IoT device will send the turn-off infrared signal. The IoT device uses the room's light to detect if the room is dark – meaning that the room was closed and there is not artificial or natural light. The IoT device also has a PIR sensor to detect movement in the room. Figure 6 shows the developed board used to control the air conditioner.

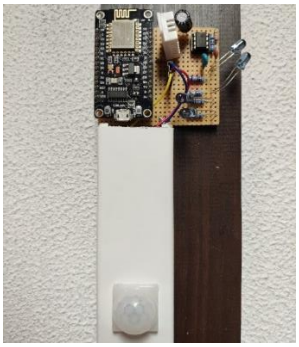

Figure 6. IoT device for the air conditioner unit

Because the hardware board for the television is similar to the air conditioner – only differentiate themselves because the use of DHT22 – the same board can be used in both case studies. However, the software is entirely different from each case study.

Following the modules of Figure 1, the air conditioner IoT device has: as processing unit a NodeMCU, as integrated sensors a PIR and a Light Dependent Resistor (LDR), as actuator four infrared LED, and has communication protocol MQTT.

To take the right advantages of such an IoT device, the board also included a DHT22 temperature and humidity module. All the sensor data is published in MQTT topics. Also, besides the turning off signal used in the context-aware autonomous control, the IoT device is able to send a complete set of signals to allow the entire remote control of the air conditioner unit.

Because NodeMCU has a limit flash memory size of 4 Mbytes; in its default version, the storage of all the air conditioner infrared signals was simply not possible. The solution was to use an external server to store all the infrared signals. Figure 7 shows the sequence diagram of the IoT device when receiving an MQTT message for air conditioner control, where the IoT device will query the external server for the right infrared code and then will emit the signal using the infrared LED. The external server was developed in Node.js using the Express module for a fast and easy implementation of a RESTful server.

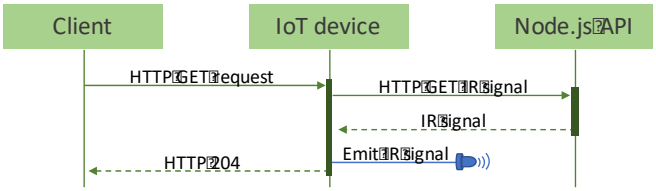

Figure 7. Infrared signal request

The IoT device for the air conditioner was built for security and energy saving. It differs from the previous TV IoT device that demands a continuous operation. The air conditioner IoT device uses an alarm logic of detecting a specific situation and then act. The context awareness (re)action layer will identify the moment when the room is dark and without movement for more than 7 minutes and then it will turn the air conditioner off if it has consumption.

For this IoT device, the correction monitoring layer was not implemented. This layer is for continuous control in order to correct failures along the controlled period. Because the air conditioner IoT device assumes an alarm logic, there is no need for having a correction monitoring layer.

IV. Deployment Results

This section presents the results in each deployed IoT device. As will be seen, the proposed architecture was successfully developed, and the results demonstrate the efficiency of context-aware autonomous control in the television and the air conditioner unit.

A. *Television deployment*

As stated before, the television smart plug imposes a schedule from 8:00 a.m. to 8:00 p.m. Figure 8 shows the television consumption between 7:30 a.m. and 8:30 p.m. When the television is turned on using the smart plug scheduled control, their consumption goes to the maximum of 95 W. However, the IoT device's correction monitoring layer will automatically detect the wrong consumption and change the energy saving to the minimum consumption of 35 W.

During the day, is visible the movement of the people inside the building. The chart in Figure 8 also presents, in green bars, the movement sensor activation. The researches arrive near 9:00 a.m. and have a clear impact on television consumption. Because the television is located near the kitchen, the movement sensor is triggered every time a researcher arrives or leaves the kitchen, goes for a coffee, goes to stores his/her food or have some lunch or snack.

In Figure 8, is visible the 7 minutes that the IoT device waits until the end of the last movement to act on television. Therefore, every time a movement is detected the television stays at the highest brightness mode for 7 minutes straight.

The deployed of the IoT device enabled the context-aware autonomous control resulting in a decrease of consumption energy of 56% – from 1.14 kWh/day to 0.64 kWh/day. Previously the consumption of television was the same during the day – from 8:00 a.m. to 8:00 p.m. – using the lowest energy saving mode.

B. *Air conditioner deployment*

The air conditioner unit goal is to detect and prevent situations where the air conditioner was left on. Figure 9 shows the results of the air conditioner control in a day that the unit was left on. The chart of Figure 9 shows the air conditioner consumption, the lux intensity inside the room and the movement sensor between 3:00 p.m. and 7:00 p.m.

Has can be seen in Figure 9, the air conditioner unit works nearly every 10 minutes. The PIR sensor only detects significant heating movements. Therefore, the movement sensor is usually not triggered even though there are users inside the room. At 3:45 p.m. all but one lamp inside the room are turned off, living only the external light and a single lamp turned on. At this moment, the room only had a person working inside. The lux sensor is in one wall of the room.

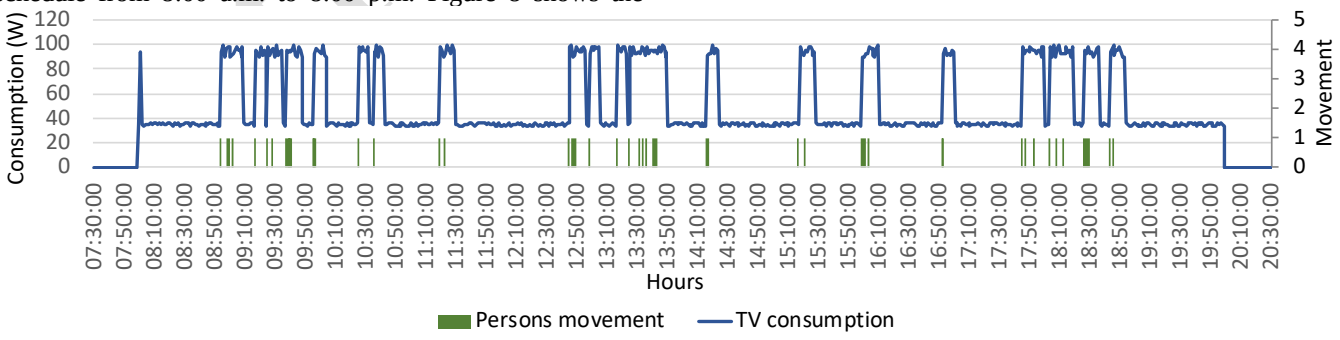

Figure 8. Television context-aware autonomous control

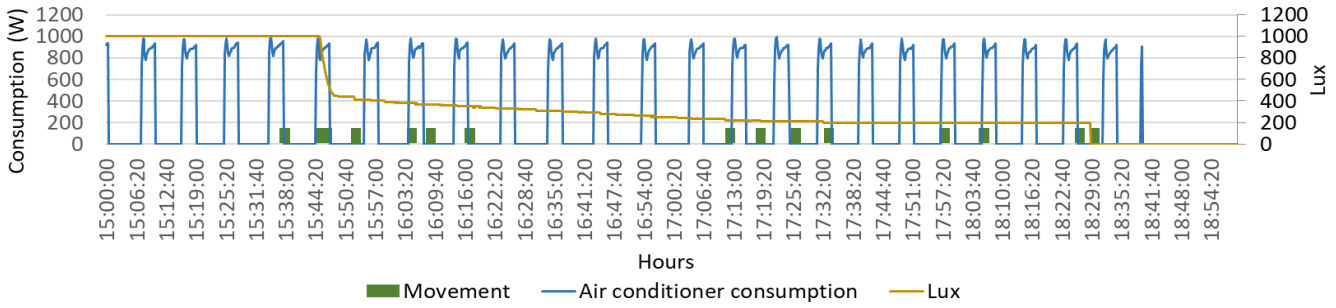

Figure 9. Air conditioner contextual actuation

At 6:28 p.m., the last person leaves the room and the last lamp is turned off. At this moment, the PIR sensor detects the person movement and stays triggered until 6:30 p.m. After the light is zero and there is no movement inside, the IoT device waits 7 minutes to see if the room remains closed – during this time the air conditioner has energy consumption because the IoT device did not yet detect that the room is closed. At 6:39, when the IoT device detects that the room is closed, there is no air conditioner consumption, and because of that, the system does not perform any action, being in the monitoring mode. When at 6:39 p.m. the air conditioner starts its motor and produces energy consumption, the system detects it and send right away a turn off infrared signal. Therefore, at 6:39 p.m. the air conditioner starts consuming but it is almost immediately stopped.

V. Discussion and Conclusions

The use of IoT products in our homes are more and more normal. These products take several forms and functionalities, from smart televisions to smart pots passing by the smart plugs. They allow users to have a better understand and control over their home's appliances and resources. They also enable the dissemination of concepts such as smart homes and energy management demand response programs.

However, much of IoT devices have the intention to produce remote monitoring and/or control. They, by themselves, are not able to autonomously control a resource or area according to its context. They have some functionalities for autonomous control and they can interact with other IoT devices to produce a contextual control but are unable to have efficient and good quality context-aware autonomous control.

The current approach in IoT devices for home control to achieve contextual control demands multiple IoT devices with a centralized point where the contextual control is performed. This centralized architecture is debatable and demands the users to buy multiple devices. This paper proposes a distributed solution where each IoT device has its own context-aware autonomous control without needing external hardware.

This paper proposed hardware and software architectures for a context-aware autonomous IoT device. The simple architecture demonstrated to be very effective for the two different case studies presented. Both IoT devices deployments use the MQTT protocol for remote control and monitor. They can be integrated with Home Assistant to work with market solutions such as TP-Link HS110 smart plug.

The proposed architecture was used in television and in an air conditioner unit. In the television case, the IoT device monitors the brightness according to the users' position. In case of the air conditioner, the IoT device is able to prevent the unit from work during the night and during periods where the room is closed and without anyone.

The proposed approach has the advantage of the context-aware autonomous control being inside the IoT device without the need for a centralized solution. However, the IoT architecture gives the possibility to remotely (de)activate the contextual control, control the actuator and monitor the connected sensors. Other advantages are its simplicity and modularity that enables the use of the proposed architectures in multiple situations where a context-aware autonomous control is desired or necessary. As a disadvantage, the proposed architectures required more hardware and should be tailor-made for each situation.

The main concern that should be considered is the security of communications and data. The users' data must be protected to maintain security and privacy. The presented work was developed and deployed in a research building where no outside server connection was used. The data was stored locally in the research group network. Nonetheless, future implementation of the proposed architectures must consider security and privacy issues in each development.

The proposed hardware and software architectures can provide, as proven in this work, context-aware autonomous control using a unique IoT device in an efficient and effective control.

References

[1] Min Li, Wenbin Gu, Wei Chen, Yeshen He, Yannian Wu, Yiying Zhang, "Smart Home: Architecture, Technologies and Systems," Procedia Computer Science, vol. 131, pp. 393-400, 2018. Doi: 10.1016/j.procs.2018.04.219

[2] L. Gomes, F. Sousa, Z. Vale, "An Intelligent Smart Plug with Shared Knowledge Capabilities," Sensors, vol. 18, 2018. Doi: 10.3390/s18113961

[3] E. Demir, E. Köseoğlu, R. Sokullu, B. Şeker, "Smart Home Assistant for Ambient Assisted Living of Elderly People with Dementia," Procedia Computer Science, vol. 113, pp. 609-614, 2017. Doi: 10.1016/j.procs.2017.08.302

[4] V. Vujović, M. Maksimović, "Raspberry Pi as a Sensor Web node for home automation," Computers & Electrical Engineering, vol. 44, pp. 153-171, 2015. Doi: 10.1109/TASE.2018.2789658

[5] Statista. Smart Home Report 2018 – Control and Connectivity; Statista: Hamburg, Germany, 2018.

[6] Home Assistant documentation. [Online] Available: https://www.home-assistant.io/docs/ (access on 9 January 2019)

[7] Mycroft AI documentation. [Online] Available: https://mycroft.ai/documentation/ (access on 9 January 2019)

[8] C. Lu, C. Wu, M. Weng, W. Chen and L. Fu, "Context-Aware Energy Saving System With Multiple Comfort-Constrained Optimization in M2M-Based Home Environment," IEEE Transactions on Automation Science and Engineering, vol. 14, pp. 1400-1414, July 2017. Doi: 10.1109/TASE.2015.2440303

[9] S. Prasad Gochhayat, P. Kaliyar, M. Conti, P. Tiwari, V.B.S. Prasath, D. Gupta, A. Khanna, “LISA: Lightweight context-aware IoT service architecture,” Journal of Cleaner Production, vol. 212, 2019. Doi: 10.1016/j.jclepro.2018.12.096

[10] S. K. Datta, C. Bonnet and N. Nikaein, "An IoT gateway centric architecture to provide novel M2M services," *2014 IEEE World Forum on Internet of Things (WF-IoT)*, Seoul, 2014, pp. 514-519. Doi: 10.1109/WF-IoT.2014.6803221

[11] Vallati, C., Mingozzi, E., Tanganelli, G. et al. Wireless Pers Commun, vol. 87, 2016, pp. 1071-1091. Doi: 10.1007/s11277-015-2639-0

[12] J. Mocnej, W. K.G. Seah, A. Pekar, I. Zolotova, “Decentralised IoT Architecture for Efficient Resources Utilisation,” IFAC-PapersOnLine, vol. 51, 218, pp. 168-173. Doi: 10.1016/j.ifacol.2018.07.148

[13] MQTT protocol specification. [Online] Available: http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html (access on 9 January 2019)

[14] NodeMcu documentation [Online]. Available: https://nodemcu.readthedocs.io/en/master/ (access on 9 January 2019)

[15] M. Kashyap, V. Sharma, N. Gupta, “Taking MQTT and NodeMcu to IOT: Communication in Internet of Things,” Procedia Computer Science, vol. 132, pp. 1611-1618, 2018. Doi: 10.1016/j.procs.2018.05.126

[16] NodeMCU custom builds [Online] Available: https://nodemcu-build.com/ (access on 9 January 2019)

Mosquitto MQTT broker [Online] Available: https://mosquitto.org/ (access on 9 January 2019