applied system innovation

MDPI

Article

A System for Controlling and Monitoring IoT Applications

Zebenzuy Lima [1], Hugo García-Vázquez [2,*], Raúl Rodríguez [3], Sunil L. Khemchandani [3], Fortunato Dualibe [2] and Javier del Pino [3]

1 iTECHierro, 38914 El Hierro, Spain; info@itechierro.com
2 Service d'Électronique et de Microélectronique, Université de Mons, 7000 Mons, Belgium; fortunato.dualibe@umons.ac.be
3 Instituto Universitario de Microelectrónica Aplicada (IUMA), Universidad de Las Palmas de Gran Canaria, 35017 Gran Canaria, Spain; rrodriguez@iuma.ulpgc.es (R.R.); sunil@iuma.ulpgc.es (S.L.K.); jpino@iuma.ulpgc.es (J.d.P.)
* Correspondence: hugo.garciavazquez@umons.ac.be

Received: 14 June 2018; Accepted: 25 July 2018; Published: 27 July 2018

Abstract: In this work, the design and implementation of an open source software and hardware system for Internet of Things (IoT) applications is presented. This system permits the remote monitoring of supplied data from sensors and webcams and the control of different devices such as actuators, servomotors and LEDs. The parameters which have been monitored are brightness, temperature and relative humidity all of which constitute possible environmental factors. The control and monitoring of the installation is realised through a server which is managed by an administrator. The device which rules the installation is a Raspberry Pi, a small and powerful micro-computer in a single board with low consumption, low cost and reconfigurability.

Keywords: Internet of Things (IoT); wireless sensor network (WSN); Raspberry Pi; Arduino; monitor; control

1. Introduction

Wireless Sensor Network (WSN) and Internet of Things (IoT) are two strongly related concepts which are essential in this work.

1.1. Wireless Sensor Network (WSN)

Nowadays, continuous technological advances have encouraged the development of devices with wireless communication capabilities, which are distributed in many locations, and are increasingly small, autonomous, more powerful, and have more efficient battery consumption [1]. With regards to the batteries, they are similar to those in mobile phones. However, they can be recharged through solar cells and from here the idea of the wireless sensor network (WSN) arises [2,3].

A WSN is composed of low cost and low power consumption devices named nodes or motes (see Figure 1). These nodes are able to acquire any information type from the environment where they are located, and process and transmit it by means of wireless links to a central node. The latter acts as a coordinator or a gateway. The nodes act as communication infrastructure elements to send the transmitted packages through more distant nodes to the coordination center [4]. These can be fixed or mobile depending on the need. A WSN is designed to realise a set of high level processing information tasks, such as detection, monitoring or classification.

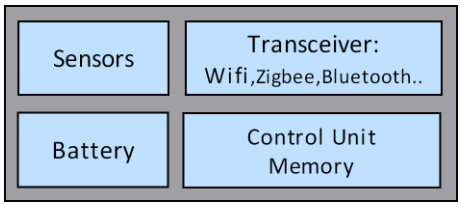

Figure 1. Wireless Sensor Network (WSN) node.

1.2. Internet of Things (IoT)

The Internet of Things is a concept that refers to the digital interconnection between ordinary objects through the Internet. IoT is considered the next Internet evolution as it supposes a breakthrough in its capability to collect, analyse and distribute data which can be used for a large number of purposes [5,6].

This work is a complete system for IoT applications where the users are able to control and monitor different installations (e.g., Wireless Sensor Network). In this paper, the related concepts of WSN and IoT are presented in Section 1. Section 2 describes the full system for controlling and monitoring IoT applications. The different blocks of the system are explained in Sections 3 and 4. Then, the integration of the different blocks is depicted in Section 5. Some tests are carried out in Section 6. Finally, the conclusions are presented in Section 7.

2. Complete System

Figure 2 shows the complete system for IoT that was implemented. This system mainly consists of three different blocks, which are all connected through the Internet. Block 1 represents each installation of a WSN. Block 2 represents the server and the database where the information of each installation is stored. Finally, Block 3 represents the different devices that the different users can use to interact with their corresponding installations through the server by means of a web browser.

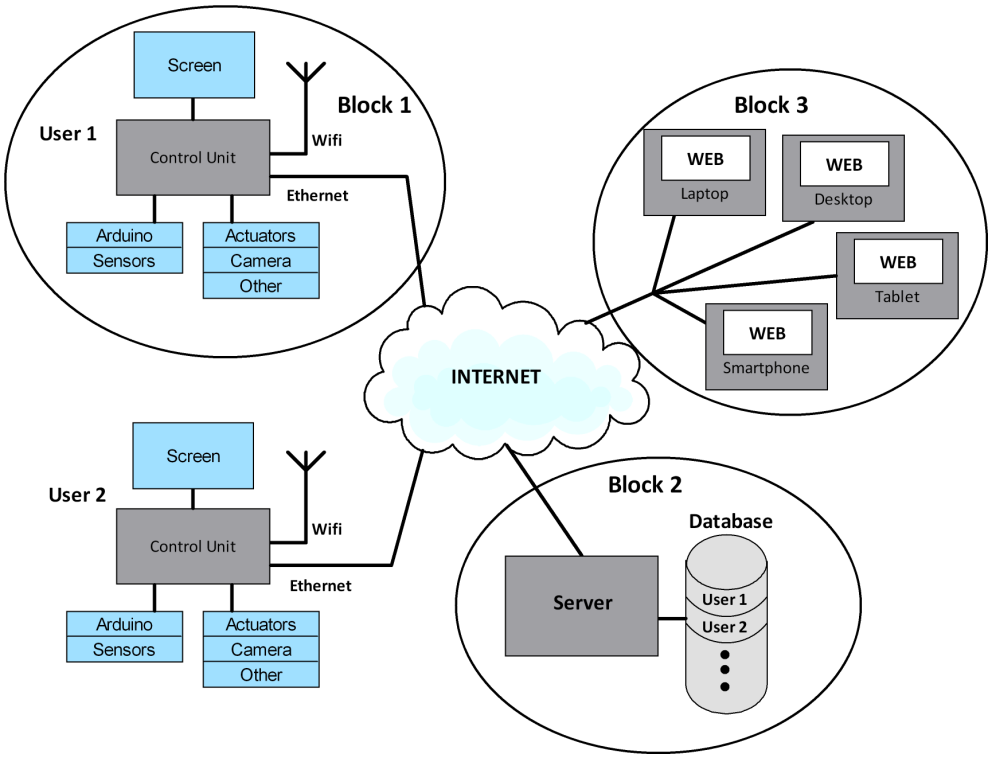

Figure 2. Complete system for Internet of Things (IoT).

3. Block 1

3.1. Control Unit

The control unit (CU) must be able to work as sensor master and control any other device which the installation requires. It must govern the installation and supply the data for users and it has to execute the requested tasks. To achieve this, it is important to have a small size, powerful, low consumption and a low cost device, with a single board. Several devices have been studied [7]. Taking into account the performance/cost ratio, the Raspberry Pi was chosen. In particular, the Raspberry Pi (RPi) used in this work includes a system-on-a-chip (SoC) Broadcom BCM2835, a CPU ARM11 76JZF-S running at 700 MHz, a graphical processor unit (GPU) VideoCore IV, and 512 MB of RAM memory. The device does not include a hard disk or solid state disk by default, instead it uses an SD card for the operating system and for non volatile storage.

The free open source Raspbian Operating System (OS) from the Raspberry Pi official web [8] was chosen for this work. Based on Debian (Linux) [9] this OS was optimised for the RPi hardware. It offers more than 35,000 packages and programs, many of which are necessary to develop this work.

It is possible to have access to the Rpi without needing a screen, a mouse or a keyboard connected through secure shell (SSH) from other devices such as laptops, PCs or mobile phones. The peripherals must be compatible with the device. There is a large list available on the Internet [10] with a large variety of brands and models. The incompatibility problems mainly come from the power supply and the SD card. In this work, a 5 V and 1 A Samsung power supply, and a LEXAR SDHC memory card were selected.

The RPi is connected to the Internet through a high-gain wireless USB adapter. It is a TP-LINK TL-WN722N device which allows a connection of up to 150 Mbps, compatible with the IEEE 802.11n standard, with a 4 dBi outdoor-gain antenna. In addition, this device provides WPA/WPA2 encryption.

To configure a static IP, we have to take into account the Raspbian OS which uses dynamic host configuration protocol (DHCP) to assign a static IP to the RPi. Therefore, we have to specify a static IP, the netmask and the gateway for the eth0 interface. After that, the last step is configuring the domain name system (DNS) server. With the objective of increasing the security of the connection, it is better to avoid using default values for the ports and other parameters.

3.2. Input/Output Ports

The RPi has different ports that can be configured as inputs or outputs that allow users to adapt the system according to the target application requirements. To control the general purpose input/output ports (GPIO) of the RPi, it is necessary to install the GPIO library for the specific language, in our case the Python library.

The serial port (UART) of the RPi comprises two signals, a transmission signal (TxD) and a reception signal (RxD), located in the GPIO header. This port can be used in order to establish a communication channel with other devices or if it is desirable, to connect the serial port to a laptop or PC through a USB port using a USB-Serial cable (logic levels of 3.3 V are permitted to be connected directly to the GPIO header). In addition, USB direct connection is possible through a common USB cable. Some configurations are required in order to use the UART, because by default it can only be accessed by the root and the users associated with the tty group.

3.3. LAMP Server

To remotely control the installations a LAMP server was installed in the RPi. This server is an Internet infrastructure system which uses Linux as the OS, Apache as the web server, MySQL [11] as the DB management, and Perl, PHP [12], or Python as the programming languages. PHP and Python were installed for this work. By means of the LAMP server it is possible to activate processes and/or run scripts by using Java. After some updates and tests, Apache2 and PHP5 were used [13].

Python language was chosen in order to control the GPIO ports in the RPi and collect data which is sent to the main server (Block 2) and monitored through a web browser.

A very simple web interface using HTML was used in order to handle different actuators, for example the LED diodes. This interface is located in the server of the RPi (Block 1). With asynchronous JavaScript and XML (Ajax), the applications can run in the user browser while the asynchronous communication is on hold in the background. In this way, it is possible to modify the web page without reloading it. It improves the interactivity, speed and usability of the applications.

3.4. Connected Devices to the RPi

3.4.1. Webcam

A webcam was connected to the USB port and the necessary libraries were installed. The streaming video function is incorporated into the web interface. The streaming service was configured for a video resolution of 320 $\times$ 240 and 25 frames per second.

3.4.2. Arduino

Arduino is an open electronic platform for developing prototypes based on flexible software and hardware. With its analogue and digital inputs and outputs it can take information from the environment thanks to a complete set of sensors. Moreover, it can interact with its surroundings, controlling lights, engines, and other actuators. Arduino is a single-board microcontroller based on the ATmega328 microprocessor. It has 14 digital inputs/outputs (6 channels can be used as PWM outputs), 6 analogue inputs, a USB connector, a female jack plug, an in-circuit serial programming (ICSP) connector and a reset button. This microcontroller can be powered from the USB connector or from a battery [14,15].

3.4.3. SquidBee Module

SquidBee is a design development kit created by the company Libelium. It allows the change and optimisation of each one of its components due to being completely based on open technologies. SquidBee uses an open code software and hardware design as a platform for remote control and detection. Through its humidity, temperature and light sensors it allows the system to capture environmental data and to send it through the SquidBee net employing the ZigBee wireless communication protocol. The SquidBee activity can be summarised by: acquiring the environmental parameter values (temperature, humidity and light), transmitting those values by ZigBee (low power consumption) or through a serial USB connection and changing to an idle state to save energy until the process is activated again.

3.4.4. XBee Shield Arduino

The Xbee Shield allows the Arduino board to communicate using ZigBee. The module can communicate in a straight line of up to 30 m indoors or 90 m outdoors. It can be used as a replacement for the serial/USB or in command mode to select a variety of broadcast or grid nets.

3.4.5. Sensors

In this work, the monitored physical magnitudes are the temperature, humidity and light. For the temperature, the SquidBee mote incorporates a MCP9700A sensor. This is a low cost sensor which works in the range of -40 °C up to 125 °C. For the humidity, the selected sensor is the 808H5V5. It has a low cost, high precision and long term functionality. The last sensor is a light sensor (LDR), where the resistance decreases as the light increases.

3.4.6. Servomotor

The incorporation of a servomotor into the system provides more functionality and brings a multitude of applications. As is seen previously, the RPi incorporates a PWM pin. This pin (GPIO 18) is used in order to control the position of the servomotor. Also, it is possible to control several servomotors using the PWM generator of the Arduino.

4. Blocks 2 and 3

In this work, the system is controlled and managed by the main server (Block 2). This system has to accommodate the website where different users can consult the state of their installations from a remote distance, through any device which supports a web browser (Block 3). The other main function is the management of users and data of the installations. For this reason, it is necessary to install a server, database (DB) management and programming language to develop the web dynamic content. In this case, the main server was implemented with the WAMP server (version for Windows of the Internet infrastructure system), which includes Apache as the server, MySQL as the DB management and PHP as the programming language. This infrastructure allows it to provide HTML pages through the Internet and also to manage data on these pages. At the same time, it provides programming languages such as PHP, Perl or Python to develop web applications.

A relational DB is the most used model nowadays to implement planned DBs. These allow it to establish interconnections (relationships) between data (stored in tables), and through these connections it can link the data of both tables [16]. A unique DB was created with the name 'installations_management' composed of two related tables:

- users_table: Personal data of the user and the server which corresponds to the installation.
- installations_table: RPi stores the sensor data after asking the Arduino mote.

To protect the information, an automatic backup of the database was implemented. This backup program (mysqldump) permits the backup of the DB in the same server or the transfer of the DB to another MySQL server.

5. Communication between Blocks

Figure 3 shows a more detailed structure of the system with the different devices and services that are involved in the communication.

5.1. Communication between Arduino and RPi

The connection between these devices can be performed by Zigbee or USB (configuring the jumpers), thanks to a USB-Zigbee adapter [17]. The communication is performed by the serial port and Python language was used for this purpose. The Arduino board must be identified in the USB ports, it appears with the name of ttyACM0. This board accepts several serial transmission speeds, from 9600 bps to 115,200 bps [18]. Furthermore, it accepts other optional configuration parameters like the parity and the stop bits [19]. In this work, successful communication at 9600 bps was configured.

5.2. Communication between RPi, Server and DB

Data must be stored in the DB so communications between the RPi, the main server and the DB are mandatory. The RPi-Server communication is carried out through the Internet (Wi-Fi) using the TP-LINK TL-WN722N wireless USB adapter as was explained in Section 3. Users connect to the domain of the main server with the web application. This web application must remotely interact with the RPis. To avoid contracting a static IP or a domain (only one domain is required for server and web application), a program in Python language was made. It identifies the public assigned IP address to the RPi and stores it in the DB each time that it changes.

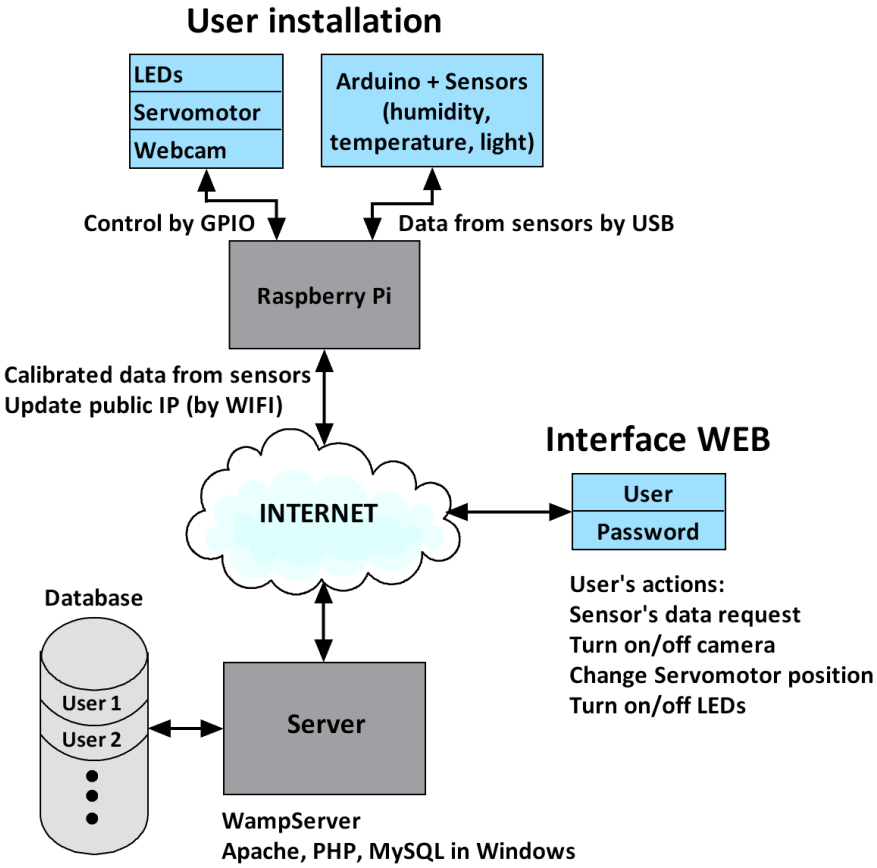

Figure 3. Devices and services involved in the communication.

5.3. Secure Connection through SSH

A minimum level of security is necessary in the RPi-server communication. With this aim a SSH server is mounted in the main server and the RPi is configured as a client. COPSSH was chosen as the SSH server, it is an open SSH software for Windows implementation. It offers SSH client and server functionality and it can be used for remote management Windows systems. COPSSH has a very simple interface where we can activate or deactivate the server, add the users who have system access, and configure the ports for more security.

To have more control over the installation, we have provided log files to the RPi. The scripts have been programmed to save an error message with the warning label and the name of the corresponding incident.

6. Obtained Results

Control tests have been performed to ensure the full management and different aspects of this system. Also, different tests have been performed with a simple script in order to see the behaviour with respect to temperature variation, resource consumption and power consumption. Regarding the consumption of the CPU, the generated processes of LED control (apache2 and Python), the activated processes when a data request is sent by the sensors (apache2 and Python) and the webcam process activation (mjpg_streamer) were taken into account. One of the most remarkable pieces of data is that when the webcam is active, the consumption of the CPU rises to 53.4% compared to a maximum of 17.8% without the webcam. With respect to the total power consumption, the Raspberry Pi demands 15 W including a necessary USB hub. Compared with the average power consumption of a normal laptop of around 65 W, the use of this system for this kind of installation leads to a power consumption saving of 76.9%.

7. Conclusions

A controlling and monitoring system focused on installations of Internet of Things (IoT) was successfully implemented using open source software and hardware based on a Raspberry Pi. To be able to fulfill all objectives, software to request, send, and receive data as well as its posterior storage in a DB has been developed. The security of the communication in the system is guaranteed through SSH between the Raspberry Pi and the server. It includes the server configuration and the DB which manages the system. This open and complete system can be used in several IoT applications saving an important amount of energy compared with using an ordinary computer/laptop.

Author Contributions: Conceptualization, Zebenzuy Lima and Hugo García-Vázquez; Investigation, Zebenzuy Lima; Methodology, Zebenzuy Lima; Supervision, Hugo García-Vázquez and Javier del Pino; Writing—original draft, Hugo García-Vázquez and Raúl Rodríguez; Writing—review and editing, Sunil L. Khemchandani and Fortunato Dualibe.

Funding: This research received no external funding.

Acknowledgments: Thanks to Sarah Grief for her valuable support with relation to the technical English review of this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Aránzazu, C.; Moreno, G.A. Revisión del estado del arte de redes de sensores inalámbricos. *Rev. Politéc.* **2009**, *5*, 94–111.
2. Gascón, D. Redes de Sensores Inalámbricos, la Tecnología Invisible. Available online: www.libelium.com (accessed on 20 September 2017).
3. Yan, Z.; Zhu, Y.; Huang, H.; Yuan, C.; Chen, X. Undermine Intelligent Gas Detection Car Based on INS/WSN and Algorithm Implementation. In *Computer Science and Engineering Technology (CSET2015), Medical Science and Biological Engineering (MSBE2015)*; World Scientific: Singapore, 2015; pp. 188–194.
4. Fernández, J.J.; Martín, M.; Martín, J.; García, A. A wireless sensor network for urban traffic characterization and trend monitoring. *Sensors* **2015**, *15*, 26143–26169. [CrossRef] [PubMed]
5. Meng, F.-Z.; Liu, H.; Tian, T. Low power subsampling architecture for Internet of Things applications. In *Wireless Communication and Sensor Network*; World Scientific: Singapore, 2016; pp. 207–215.
6. Yue, K.-H. A Secure IoT-Based Healthcare System with Body Sensor Networks. *IEEE Access* **2016**, *4*, 10288–10299.
7. Blog Think Big. Alternativas Raspberry Pi. Available online: https://blogthinkbig.com (accessed on 18 September 2017).
8. Raspberry Pi. Section of Downloads. Available online: https://www.raspberrypi.org/downloads (accessed on 15 September 2017).
9. Raspbian Website. Raspbian, OS. Available online: https://www.raspbian.org (accessed on 15 September 2017).
10. Embedded Linux Wiki. RPi Verified Peripherals. Available online: https://elinux.org/RPi_VerifiedPeripherals (accessed on 14 September 2017).
11. Wikipedia Website. MySQL. Available online: https://en.wikipedia.org/wiki/MySQL (accessed on 16 September 2017).
12. Wikipedia Website. PHP. Available online: https://en.wikipedia.org/wiki/PHP (accessed on 18 September 2017).
13. Wikipedia Website. Apache HTTP Server. Available online: https://en.wikipedia.org/wiki/Apache_HTTP_Server (accessed on 18 September 2017).
14. Huang, T.; Zhou, L. Research on Intelligent Irrigation System of Agricultural Base Based on Cloud-Platforms. In *Computer Science and Technology*; World Scientific: Singapore, 2017; pp. 81–90.
15. Adrian, S.; Tan, T.; Iqbal, S. Implementation of INC MPPT and CV Charging using LLC Resonant Converter for Solar Streetlight System. *J. Circuits Syst. Comput.* **2017**, *27*, 1850043.
16. Wikipedia Website. Database. Available online: https://en.wikipedia.org/wiki/Database (accessed on 17 September 2017).

17. Sparkfun Electronics Website. Adaptador Xbee. Available online: https://www.sparkfun.com (accessed on 17 September 2017).
18. Embedded Linux Wiki Website. RPi Serial Connection. Available online: http://docshare.tips/raspberry-pi-low-level-peripherals-elinux_584230cbb6d87fc64e8b4747.html (accessed on 23 September 2017).
19. Arduino Website. Serial Port. Available online: https://playground.arduino.cc/Interfacing/SerialNet (accessed on 21 September 2017).