Home automation system with web interface in the JdeRobot framework

Daniel Castellano, José María Cañas

***Abstract*—Despite the current real estate crisis, the ratio of houses that include home automation systems is increasing. For the seller it is an aditional value, for the buyer it is a trustworthy technology that will help to improve their comfort, power efficiency and safety. The interest of companies in this field is also growing. Home automation systems share with physical agents the usage of sensors and actuators, but they are usually installed in the facilities instead of on mobile platforms. In this work we present the fourth generation of a home automation system that uses the opensource JdeRobot framework. The system consists of a central node and a distributed set of *ZigBee* wireless nodes (sensor or actuator ones) and video camera nodes. A web interface has been deployed for the human user access and interaction with the system, so the user can remotely read the current sensor data, see statistics, receive alarms and deactivate them, see the video in streaming and send commands to the actuators. This ongoing work builds a framework to develop home automation applications for comfort, safety, monitoring and alarms involving sensors and actuators at home. It has been validated and tested in a real home.**

***Index Terms*—Home automation, sensors, actuators, software framework**

I. INTRODUCTION

The home automation systems use devices (sensors, actuators and/or control elements) that allow the automation of different tasks related with the control of different aspects at home or other buildings. Generally speaking the object is to increase comfort, reduce energy consumption and increase security. These devices are usually integrated using indoor and/or outdoor communication networks, both wired or wireless, and their control can be done either within or outside the home.

This field has many applications. As a security element it can be oriented toward personal security or material objects. As an energy efficiency element, light intensity sensors can be used to keep the light level withing a range controlling shutters to reduce the power consumption. Combining light sensors with presence sensors lights in bedroom can be turned off when not required significantly reducing the power consumption. As a comfort element a reasonable temperature can be maintained in different rooms with changing outdoor conditions.

The most commonly used home sensors are the regular RGB cameras (they do not work in the dark) and motion sensors (currently they not distinguish between people, animals or other objects). The most commonly used home actuators are the acoustic alarms and relays to turn on/off lights as people walk passed, saving electricity.

As the need to connect different home devices arose, standards were proposed. The most important ones are reviewed. X10 [6], developed in 1975, is the initial communication standard to remotely control electric devices. It uses the power network to communicate the different composing devices through predefined commands as specified in the standard. It is still in use today, although it has been redefined to include increased functions.

Universidad Rey Juan Carlos.

E-mails: castellanobonilla@gmail.com, jmplaza@gsyc.es

KNX [7] is the european standard for the automation of homes and offices. Internet, wireless, high voltage (electric power) and low voltage (the most commong) cabling can be used for data transmission. It is an open standard that allows a larger number of commands and more varied sensor types than *X10*. KNX is not compatible with *X10* but there are many gateways that allow for the use of multiprotocol systems.

SCP (Simple Control Protocol) is a *Microsoft* and *General Electric* standard and is currently royalty free. It requires point-to-point connections and low voltage wiring, so it is not viable for systems with a large number of sensors. It is uncompatible with European legislation, more specificaly with *EN-50065*[8].

In the last few years the EU has financed various projects which research on home automation systems, aiming to ease the introduction of this technology in homes and to increase the standarization of platforms, gateways, sensors, etc.. This effort, when applied to the elderly care, coincides with another high priority research line in the last European Research Framework: *Ambient Assisted Living (AAL)*. AAL aims at increasing artificial intelligence and services in houses and Old People's Homes.

Moreover the number of companies that offer home automation systems which are interoperable is increasing, allowing for the flexible creation of systems at low cost. A recent sample of this interest is the purchase of Nest Labs[1], the developer of thermostats and smoke alarms, by Google for 3200 M$. There are companies that sell isolated sensors that do not require interconnection (for instance, vibration sensors which emit sounds), or that have a limited connection (mainly by X10) to link several sensors to a central node, which will sound the alarm, or visualize the IP camera. Those systems are generally cheap and easy to install and maintain, but offer little flexibility. Other companies offer design services and implementation of made to measure home automation systems, that can not be reused and tend to be expensive.

[1]`https://nest.com`

The purpose of this research was to design, construct and program a wireless, distributed, internet accessible home automation system whih also allows for the use of security cameras. More specifically the aim was to include it in the software framework JdeRobot for physical agents, extending the number of supported sensors and actuators. In addition the introduction of ZigBee technology in sensor nodes was another goal. The wireless technologies provide more flexibility when distributing the sensors, as well as a simpler and cheaper installation. Also some requirements were considered such as low cost, low power consumption, aesthetics, ease of use, access from cell phones, interoperability (that allows for different types of sensors to be used) and small size.

In the second section sample of various home automation products currently available is presented. The system developed is described in the third section along with the chosen design and each one of its components. The fourth section is devoted to some experiments which validates the implementation undertaken. The paper ends with the main conclusions of the research.

II. Related works and products

Some representative samples of the home automation market in Spain are described in this section. Z-Wave [9] includes several interoperable modules from different providers, with large variety of sensors and actuators. It allows the creation of wireless systems and their management using smartphones. They are proprietary systems and some of its modules are still on development. It works with different providers and so their prices vary. For instance, a central controller node costs between 60 and 850 euros (the most basic one does not include ethernet connection), a sensor between 40 and 60 euros, and a camera between 70 and 130 euros. The modules are not usually very expensive and Z-Wave providers also offer subscription payment per services.

Verisure [10] is the alarm system of *SecuritasDirect* firm. It is not a complete home automation system, as it only includes alarm sensors. The sensor kit is fixed, no made to measure system is available. The kit includes one central module, two camera-based motion sensors, an open door sensor, an NFC reader and a siren. Its main advantage is the use of volumetric sensors that work even without triggering false alarms in presence of pets. It monthly charges 99 euros and offers a police call service (and videosurveillance in the *SecuritasDirect* headquarters).

Prosegur [11] provides surveillance and home automation services. They offer a complete home automation system (security, comfort and power saving), although their web site only shows surveillance modules (IP cameras), motion detectors, magnetic switches and switchboards with NFC label. They monthly charge 30 or more euros. The included equipment is similar to that of *Verisure*, except for an additional motion sensor instead of an open door one.

Delta Dore [12] is a generic, multipurpose, wireless home automation system. It consists of several modules depending on the customer needs. The system may manage the air conditioning, the home lights, alarms and automatic door opening. The system is commanded from a remote controller, from a display or from a smartphone app. It is expensive. For instance, the central node costs more than 200 euros, there are sensors for 65 euros, etc.. although several kits are offered that reduce the final price.

The Spanish firm Libelium [13] provides a sensor network system which implements a multiprocol router that may interact from the same device with several technologies: WiFi, ZigBee, GPRS, Bluetooth and GPS. It is a system mainly oriented towards development (it is based on Arduino and XBee) but the firm may develop customized systems using its base boards and sensors.

III. HOME AUTOMATION SYSTEM IN JDEROBOT

The home automation system developed at the URJC Robotics group is known as Surveillance[2]. Its most up to date version, presented in this paper, is Surveillance-4. It was built upon the distributed video Surveillance-2[3] [1] and upon Surveillance-3 [2], which uses Arduino processors for all the system nodes, including the central one, and which does not have any video node.

A. Design

Surveillance-4 consists of a central node and several wireless satellite nodes, in a star configuration (Fig. 1). The central node has a low cost processor, a RaspberryPI one with GNU/Linux inside, WiFi and ZigBee. There are several satellite types. ZigBee satellites include a low cost Arduino processor, home automation sensor or actuator devices and ZigBee connectivity. Video satellites include a RaspberryPi processor and a webcam. The central node is the only one which requires internet access, either wired or wireless.

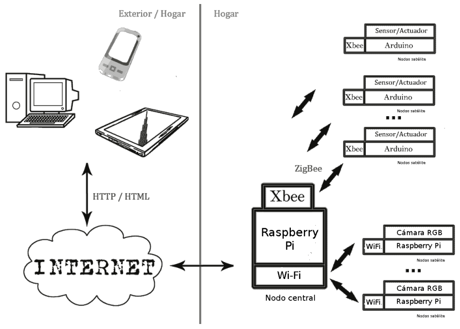


Fig. 1: Designed architecture

ZigBee satellites may be either sensory or actuation nodes. In the first case they include

[2]`http://jderobot.org/Surveillance`

[3]`http://jderobot.org/Rocapal_Enabling_Technologies_LS`

sensors that detect flood, gas, temperature, humidity, etc.. In the second, they typically include relays. They have been designed to be low cost and low power consumption, and hence, the Arduino was chosen as the processing unit and the ZigBee as the networking technology. In addition, these nodes may be slept most of the time, saving batteries.

In order to decrease the size of processors and to distribute the computation load we chose a star topology, where the satellites do some complex tasks, reducing the central node CPU requirements. This star architecture is flexible and scalable for future developments. The main risk of this architecture is the single point of failure at the central node. The wireless satellite nodes may be distributed along the home.

The human user interacts with the system through a web interface, from the web browser, either on a PC, a laptop, a tablet or a smartphone. This interaction allows seeing the current sensor data and sending commands to the actuators. This web interface was chosen to have operating system and platform independence.

B. ZigBee satellite node

The satellite nodes are composed of an *Arduino UNO*, a *XBee* shield with its corresponding Z24-B module [20], a LED, a 6lr6 battery, a sensor and small electronic components required for its operation. They are wireless nodes required to have low power consumption, so they have to be simple. The supported sensors in this version are temperature plus humidity, humidity (for pots), gas (or smoke), light intensity, water, vibration and magnetic switch (open door). They are shown in Fig. 2.

Initially there are two types of ZigBee satellite nodes: alarm and pure sensory ones. The last may be additionally divided into analog and special ones. The alarm nodes generate or stop voltage difference on two of their pins when certain event occurs. In analog sensory nodes the voltage between two of their pins varies as the measured physical quantity does. The software reads such voltage using the A/D converter inside the Arduino chip. Such reading is translated into a digital level in a byte (0-255 for a voltage between 0-5V). The sensors are not calibrated, therefore there is no lineal correspondence between the digital level and the real value of the measured physical quantity. In special sensory nodes the DHT protocol is required to locally obtain the readings. This is the case of the temperature plus humidity sensor.

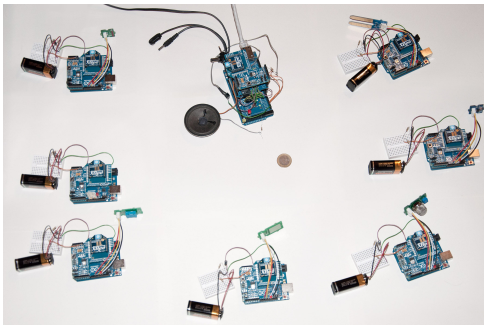


Fig. 2: Several ZigBee satellite nodes

The processing inside the nodes can be divided into local sensor reading and communication with the central node. Each sensory node sends the sensor measurements and its battery level every 2 minutes. Analog sensory nodes use their A/D converter to read both the sensor value and the battery level. Alarm nodes send only the battery level every 2 minutes and the alarm generation is sent asynchronously, just as it takes place: the sensor itself will wake the processor up when the corresponding event occurs and the processor will immediately send the alarm to the central node.

In order to minimize the power consumption the wireless nodes are slept most of the time, awaking only to: first, collect the sensor reading; second, to send their data to the central node; and third, to receive the possible messages from the central node.

The satellite nodes only remain awake, with their ZigBee module actively listening, in the case of alarm or warning state. This way they can synchronously receive messages from the central node (for instance, that the alarm has been deactivated by the human user).

Although several elements (alarms or sensors) may be included in a single node, for the sake of simplicity, only one element per node is permitted, except for the temperature plus humidity sensor, which physically is a single device providing two different sensor readings at the same time.

A third ZigBee node type is the actuator node. The most common actuators are the alarm notifiers (for instance a loud-speaker), and relays to switch on/off a power plug and hence its connected device.

C. *WHAP ZigBee protocol*

The communication between the central node and the satellites is done through ZigBee. We have created the WHAP (*Wireless Home Automation Protocol*) protocol for the sending of alarms, information exchange, etc. between the nodes. Due to the low bandwidth of ZigBee this protocol is not appropriate for massive data sending (like the video streaming). It is specially designed to work upon ZigBee (it fits its constraints), although it can be used upon other less demanding standards with the number of bytes per packet such as X10, KNX, Ethernet, etc. Depending on the context it can be used either as a transport protocol (similar to UDP because it does not provide delivery guarantee) or as an application protocol, just as it has been used in this project (the ZigBee protocol inside the *XBee* already implements the network and transport levels and provides delivery guarantee).

The WHAP protocol allows a maximum of 255 sensors/actuators of the same type inside the same network, up to 255 different types of sensors or actuators, and data up to 6MB (a maximum of 96 bytes in a single packet).

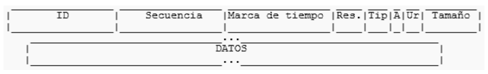


Fig. 3: WHAP ZigBee communication protocol

The fields inside the WHAP frame (Fig. 3) are:

- *ID*: 2 bytes for uniquely identifying the final destination node; inside this field, the first byte indicates the type of sensor/actuator and the second one is the node identifier inside its type. Currently the allowed node types are: (0) central node, (1) temperature sensor (from the temperature plus humidity sensor), (2) humidity sensor (from the temperature plus humidity sensor), (3) light intensity sensor, (4) humidity sensor for plants, (5) gas and smoke sensor, (6) vibration sensor, (7) flood sensor, (8) open door/window sensor, (9) PIR sensor (movement using infrared signals), (10) camera and (11) actuator. If a satellite receives a packet, this ID field specifies the destination node (a zero value means the message is a broadcast for all the nodes). If the central node receives a packet, this ID field shows the sender node.
- *Sequence*: 2 bytes for the number inside the packet sequence. This field will increase one unit per packet coming from the same data. The data can be split in several packets to be transmitted. For data smaller than 96 bytes this field is always zero.
- *Timestamp*: 4 bytes with the date and time (seconds from January 1st, 1970) of when the data was sent. In the case of a sequence each packet of the sequence must have the same timestamp, the same as the first packet.
- *Miscellaneous byte*: 2 bits are reserved (Res) and must be zeros, 3 bits (Tip) to

indicate the type of data (0 for data from a sensor to the central node, 1 for data from the central node to an actuator, 2 for data request from the central node to a sensor/actuator, 3 for answer with the requested data, 4 for alarm reset –the human user considers the alarm already solved–, 5 for actuation request –only in messages to actuators–), 1 bit (A) to distinguish between alarms and other messages, and 2 bits (Ur) to specify priority (from 0 meaning not urgent up to 3 meaning very urgent).

- *Data size*: 1 byte showing the total size of the data inside the packet. Maximum 96 bytes.
- *Data*

D. Central node

The central node consists of one RaspberryPI processor with GNU/Linux, one USB-WiFi converter to talk with video satellite nodes and one XBee module to talk with ZigBee satellite nodes. Typically it is directly connected to the power line and has an internet connection. Besides receiving the data and alarms from the satellites it includes the software that specifies how to react after each alarm and the source code of the home automation applications themselves (presence simulation, alarm propagation, etc.).

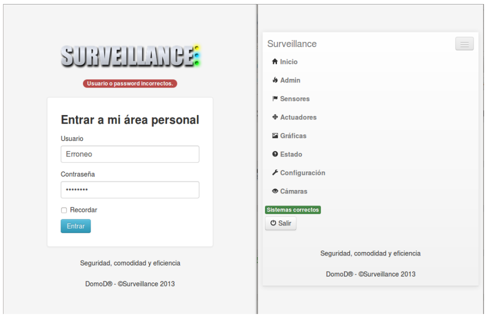
1) Communication with satellites: Each satellite sends the information from its sensor and its remaining battery level every 2 minutes and sends the alarms when they ocurr. To receive them the central node uses its XBee module through the GPIO serial port included in the RaspberryPi, and follows the WHAP protocol. When the arriving packet is a simple sensor reading it checks whether the data are more recent than the stored one (using the *timestamp* field inside the WAHP frame). If not, the packet is discarded. If it is more recent stores it overwritting the previous one. All the data are stored in the database shared with the web application. The central node also checks whether the remaining battery level falls below 6.5V and initiates the low battery warning in such a case.

When the arriving packet is an alarm it fires the alarm protocol. Several LEDs are activated, both in the central node and in the corresponding satellite, an acoustic siren sounds and a visual alarm notification triggers in the web interface. When the alarm protocol is started, the red LEDs in the central node and in the corresponding satellite are turned on permanently, the loud speaker in the central node starts to sound intermittently. Once the alarm has been managed and solved the home automation system may return to its normal state when the human user pushes the hardware alarm reset button or clicks on the alarm reset widget on the web interface. Both actions cause the LED and loud speaker deactivation in the central node until a new alarm takes place and cause the sending of a WHAP alarm deactivation message to the source satellite. On the reception of such message through ZigBee the satellite deactivates its own red LED and resumes its normal operation. In the case of resetting the alarm without solving the real problem the satellite will detect it once more, send a new alarm packet and fire the alarm protocol again.

The central node periodically checks whether its stored data are fresh enough and whether there is no satellite node with lost connection. If it has not received any data from a given satellite in the last five minutes the central node executes the 'inactive node detected' warning.

In order to get the sensor readings from other JdeRobot application components and to send commands to ZigBee actuator nodes from those components, a `ZigBeeServer` has been developed. This software component talks to the ZigBee nodes, either sensory, alarm or actuator nodes and offers their data through a standard high level ICE interface, following the JdeRobot [4] policy for intercomponent communication. It can be seen as the driver to manage the ZigBee nodes hiding their low level details.

2) Web interface: The central node includes a web server and a web application that shows the sensor data collected from the satellite nodes, either ZibBee or video nodes. It allows sending commands to the actuators of the corresponding ZibBee satellites. In additions it displays the active detected alarms and allows their reset. This web interface significantly increases the usability of the home automation system.



Usuario o password incorrectos.

Entrar a mi área personal

Usuario: Erroneo

Contraseña: ••••••••

☐ Recordar

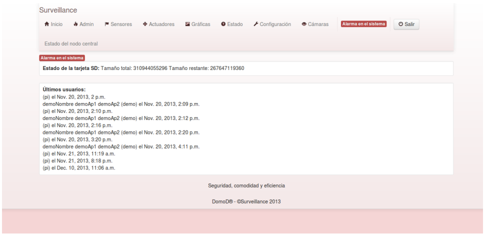
Seguridad, comodidad y eficiencia

DomoD® - ©Surveillance 2013

Fig. 4: Initial page of the web interface

The web server has been implemented using de *Django* [24] web framework (1.6 release), with its data model, templates, views, automatic administration interface and all the power of Python to program the logic of the application. This framework allowed a fast web application development using third party functionality, widely tested, like the *Bootstrap*[31] and *JQuery*[30] *JavaScript* APIs. With them the visual appearance of the application is appealing. Many templates have been used, they are dynamically filled with data coming from the satellite nodes.

In the presentation layer of the web application the simplicity and ease of use have been prioritized, with small texts and self explanatory images, using a proper size to improve visualization without hindering the navigation. The web interface use is quite simple, even for non-technical users. First, the user has to login in the initial page (Fig. 4) and then a menu with all the choices appears in the upper side of the web page. Several admin pages have been developed to operate the system (only available to users with permissions), some pages to see the system and node's state (the last sensor reading and the remaining battery), to see the active alarms, when they occurred, a history alarm log, to display statistic from the sensor readings, one page to send commands to the actuator nodes and another to see the images from the video nodes in streaming. Several CSS style sheets have been created, including some for active alarm state with the red background. This way, when an alarm occurs the web interface changes to red background allowing an intuitive notification to the user (Fig 5).



Alarma en el sistema

Estado de la tarjeta SD: Tamaño total: 310944055296 Tamaño restante: 267647119360

Últimos usuarios:
(pi) el Nov. 20, 2013, 2 p.m.
demoNombre demoAp1 demoAp2 (demo) el Nov. 20, 2013, 2:09 p.m.
(pi) el Nov. 20, 2013, 2:10 p.m.
demoNombre demoAp1 demoAp2 (demo) el Nov. 20, 2013, 2:12 p.m.
(pi) el Nov. 20, 2013, 2:16 p.m.
demoNombre demoAp1 demoAp2 (demo) el Nov. 20, 2013, 2:20 p.m.
(pi) el Nov. 20, 2013, 3:20 p.m.
demoNombre demoAp1 demoAp2 (demo) el Nov. 20, 2013, 4:11 p.m.
(pi) el Nov. 21, 2013, 11:19 a.m.
(pi) el Nov. 21, 2013, 8:18 p.m.
(pi) el Dec. 10, 2013, 11:06 a.m.

Seguridad, comodidad y eficiencia

DomoD® - ©Surveillance 2013

Fig. 5: Web interface with generated alarms

Because the chosen user interface is HTTP it can be accessed from any device with a web browser: smartphones, computers, tablets or even smartTVs. This provides large access flexibility to the user. In the typical home configuration two IP ports must be open to use the system from outside: 80 port for the central node and 8080 for the video satellite node.

E. Video satellite node

Video satellite nodes consist of a RaspberryPI processor, a webcam and network connectivity, either WiFi or Ethernet (ZigBee does not offer enough bandwidth). To send the image stream the *M-JPEG Streamer*[19] open source software has been used. It allows capturing the images from the USB camera in JPEG format and serving them in a stream inside a web page without adding any plugin or extra component into the browser. Some web pages from the *M-JPEG Streamer* project have been modified to make them look like the other pages of Surveillance-4.

In addition, the Python program `listener.py` has been created. It is executed when the satellite starts and allows the communication with the central node to initiate the video stream only when it is going to be visualized, and to stop it when such visualization has finished. This way the CPU is not continuously requesting images from the camera, it remains silent when the human user does not require any video at all. The `listener.py` program opens a TCP socket with the central node using the local network and its private IP. When a command is received through this socket from the central node the program gets the command ("init" to initiate the video stream, "stop" to halt it, "switch" to change the state), checks the current node state (transmitting or stopped) and executes it. In order to start the image transmission `listener.py` creates a second socket connection where listen the requests coming directly from the web browser (following the architecture shown in the left side of Fig. 6). Once the *TCP bind* message comes directly from the browser machine the video satellite node starts the video streaming using such connection. In order to halt the video stream `listener.py` kills the process that gets the camera images and closes the TCP socket used for the streaming.

IV. Experiments

The system has been installed in a real home and tested with two ZigBee satellite nodes at the same time. Both ZibBee nodes have been located in a bedroom 16 meter away from the central node in the living room, with three walls in between. The script "data_test.py" has been created to introduced test data in the database, simulating the

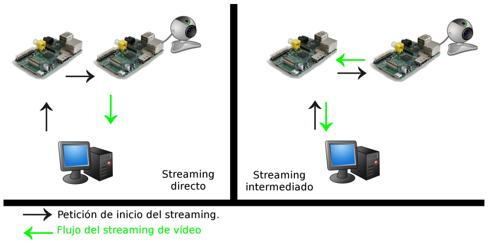

Streaming
directo
Streaming
intermediado
Petición de inicio del streaming.
Flujo del streaming de vídeo

Fig. 6: Video transmission architectures

presence of additional satellites. Due to XBEE module limitations only up to ten satellites can simultaneously connect to the same central node. The central node has been up and running for a month without interruptions, performance drops nor heating. Several tests have been performed including accesses from a computer, from an Android device, from the private LAN and from the outside. Simultaneous accesses have also been checked reporting no problem. The same tests have been also succesful with the video streaming.

One test consisted in suddenly shut the power of the nodes down, simulating a power blackout. All the nodes recovered without problems and restored the previous configuration.

There are several illustrative videos of the system working at the Surveillance-4[4] web page.

A. Comparing wireless technologies

We wanted a wireless system and so we studied three options (WiFi, Bluetooth and ZigBee) in terms of speed, power consumption, price and scope (Fig. 7). Once the speed was enough for our application requirements we made emphasis on the power consumption. Finally we chose ZigBee for the nodes with a low data transmission rate and and WiFi for nodes with high data throughput (video).

[4]`http://jderobot.org/Surveillance`

Technology	Speed	PowerConsumption	Scope	Price
WiFi	300 MBps	85 mA	100 m	Medium
Bluetooth	1 MBps	40 mA (0.2mA idle)	10 m	Small
ZigBee	250 KBps	30 mA (3uA idle)	50 m	Medium

Fig. 7: Wireless technologies comparison

B. *Alarm generation*

Fig 8 shows the behavior of a ZigBee satellite node when the human user poures a water drop over the flood sensor. The system properly detects the alarm, turns the alert LEDs on and activates the loud speaker sound until the alarm is deactivated.

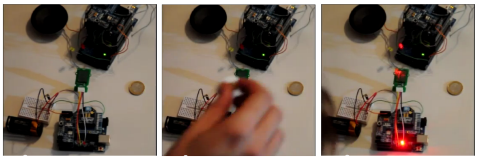
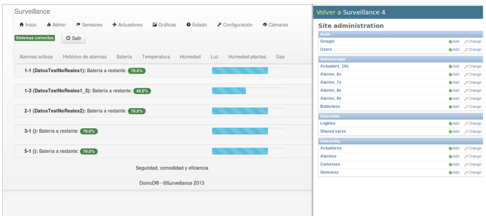


Fig. 8: Alarm generated by the water sensor node

C. *Using the web interface with ZigBee sensor nodes*

The sensor readings are displayed in the normal web interfaz, as shown in Fig. 9. A graphical widget has been also included to visually show the battery level of each node.



1-1 (DatosTestNoReales1): Batería a restante: 76.0%

1-3 (DatosTestNoReales1_3): Batería a restante: 46.0%

2-1 (DatosTestNoReales2): Batería a restante: 76.0%

3-1 (): Batería a restante: 76.0%

5-1 (): Batería a restante: 76.0%

Seguridad, comodidad y eficiencia

DomoD® - ©Surveillance 2013

Site administration

Auth: Groups, Users

Datastorage: Actuators_10s, Alarms_6s, Alarms_7s, Alarms_8s, Alarms_9s, Batteriess

Ownstate: Loginss, Shared varss

Satconfig: Actuatorss, Alarmss, Camerass, Sensorss

Fig. 9: Web interface showing last sensor data and the node battery state

D. *Using the web interface with video node*

Fig. 10 shows the web interface for a video satellite node as displayed from two different web browsers, one in a personal computer and another in an Android smartphone.

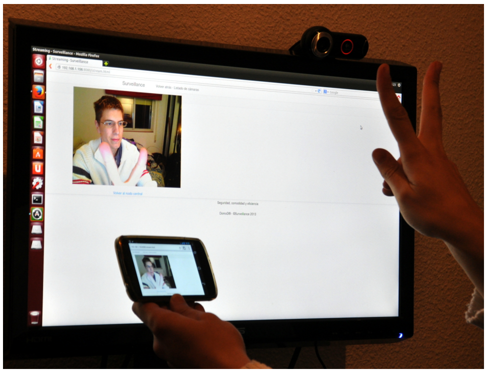


Fig. 10: Video streaming

E. *Using the web interface with ZigBee actuator node*

Fig 11 shows the proper system operation when a command is sent to a ZigBee actuator node. In this case the node relay was connected to a light. To illustrate the efect a witness camera was also included in a video node. Initially the light is off, but after the corresponding command is sent from the web interface it arrives at the actuator node, which executes it and hence the light is turned on.

V. Conclusions

In this paper we have presented a complete home automation system, distributed, wireless and with web interface. The system has a star architecture with several satellite nodes distributed throughout the house, which communicate with the central node using ZigBee. A specific protocol has been developed over ZigBee to shape the wireless dialog between the satellites and the central node. The satellites include the most common sensors in home automation systems: gas, flood, temperature, vibration... Actuator satellites with relays have been also created. In addition video satellite nodes have been developed to receive in real time the images from cameras.

The whole system is low cost. Arduino microprocessor in the ZigBee nodes and

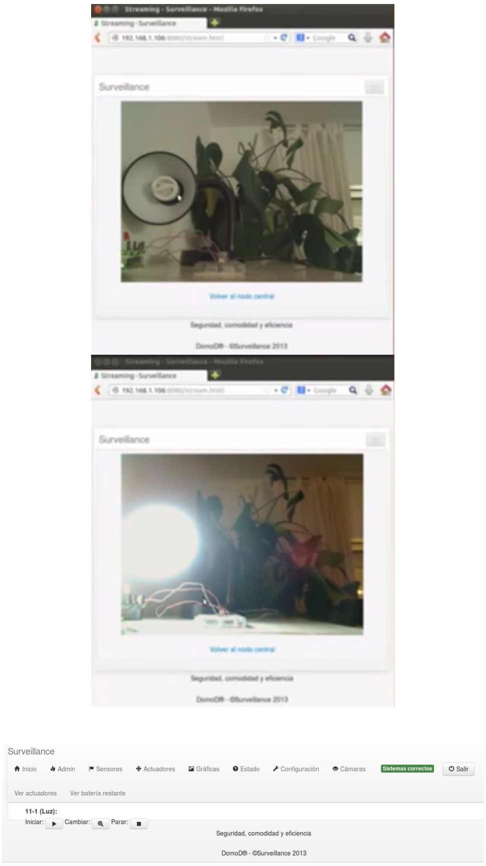


Fig. 11: An actuator node used through the web interface to turn on/off a lamp

RaspberryPi for the central node were chosen. Every two minutes the satellite nodes send sensor readings and asynchronously notify the alarms just when they occur. The human user deactivates them once they have been managed and solved.

The central node includes a web application developed on Django that allows the human user to comfortably and remotely interact with the home automation system, regardless the operating system and platform used (smartphone, laptop, etc.).

As shown in section IV the system has been experimentally validated. It was installed on a real home and several tests were performed to prove all its functions. Surveillance-4 is a step forward from the previous home automation projects of the group and a complete proof of concept. It also extends the available sensors and actuators ready to use from any JdeRobot application, including the developed ZigBee devices.

We are actively working on the integration of new home automation devices like X10 nodes, sensors that measure individual power consumption and other sensors already supported in JdeRobot like the RGBD (Kinect, Asus Xtion) sensors. The approach is to extend the capabilities of the central node to exchange data using standard ICE interfaces with all the already existing JdeRobot components like `CameraServer` for cameras or `OpenniServer` for RGBD sensors.

ACKNOWLEDGEMENTS

This research has been partially sponsored by the Community of Madrid through the RoboCity2030-II project (S2009/DPI-1559).

REFERENCES

[1] Sistema distribuido de videovigilancia basado en Android. Roberto Calvo Palomino, Master Thesis dissertation, ETSIT Universidad Rey Juan Carlos 2010.

[2] Sistema domótico distribuido, inalámbrico, interoperable y *opensource*. Daniel Castellano, Proyecto Fin de Carrera, ETSIT Universidad Rey Juan Carlos, 2013.

[3] ORELLANA GALLOSO, RUBÉN,*Sistema de monitorización y control de riego e iluminación basado en tecnologías Arduino, ZigBee, Android y Bluetooth para un emplazamiento en Cádiz*, España, 2013.

[4] Proyecto JdeRobot. Middlelware para el desarrollo de componentes robóticos y domóticos. http://jderobot.org/index.php/Main_Page

[5] Detección de caídas en hogar basado en cámaras y kinects. http://jderobot.org/index.php/ElderCare

[6] Monográfico sobre el estándar X10. Historia, protocolo, implementaciones, etc. de este estándar. https://forja.rediris.es/docman/view.php/414/761/X10.pdf

[7] Página web del estándar KNX. http://www.knx.org/es/

[8] Norma UNE-EN 50065-1:2012 para la transimisión de sañales por la red eléctrica. http://www.aenor.es/aenor/normas/normas/fichanorma.asp?tipo=N&codigo=N0048827

[9] Página web del la alianza empresarial para la difusión de *Z-wave*. http://www.z-wavealliance.org/

[10] Sistema de seguridad domótica de la empresa *Securitas Direct*. http://www.securitasdirect.es/myverisure.html

[11] Alarma para el hogar *Prosegur Proview* +. http://www.alarmahogarprosegur.com/?o=web

[12] Sistema domótico textitDelta Dore. http://www.deltadore.com/

[13] Página web de la compañía *Libelium*. http://www.libelium.com/company/

[14] Estándar de la última especificación (2011-04-12) del lenguaje de programación C. http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf

[15] Página oficial del lenguaje de programación Java SE. http://www.oracle.com/technetwork/java/javase/overview/index.html

[16] Documentación de las funciones del lenguaje Java SE versión 7 (*javadoc*). http://docs.oracle.com/javase/7/docs/api/

[17] Página oficial del lenguaje de programación Python. http://www.python.org

[18] FIELDING, ROY T.,*Architectural Styles and the Design of Network-based Software Architectures*, EE.UU., 2000. http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm

[19] Página oficial del programa *M-JPEG Streamer*. http://sourceforge.net/projects/mjpg-streamer/

[20] *XBee* Multipoint RF Modules, *Digi International Inc*, 2011. http://www.digi.com/pdf/ds_xbeemultipointmodules.pdf

[21] Página oficial del proyecto Arduino. http://arduino.cc/

[22] Página ofical de la *Raspberry Pi*. http://www.raspberrypi.org/faqs

[23] Esquema hardware de la *Raspberry Pi*. http://www.raspberrypi.org/wp-content/uploads/2012/10/Raspberry-Pi-R2.0-Schematics-Issue2.2_027.pdf

[24] API de programación en Python para Django. https://docs.djangoproject.com/en/1.6/py-modindex/

[25] Wiki para el sistema de sensores de *Seeedstudio* utilizados en Surveillance 3 y 4. http://seeedstudio.com/wiki/GROVE_System

[26] Página oficial de OpenGL ES. http://www.khronos.org/opengles/

[27] Especificaciónes de los estándares *IEEE 802.11* (*Wi-Fi*). http://standards.ieee.org/about/get/802/802.11.html

[28] Especificación del estándar *IEEE 802.15.4* para la creación de redes inalámbricas. http://standards.ieee.org/getieee802/download/802.15.4a-2007.pdf

[29] Página web de la ZigBee Alliance, con información sorbe el consorcio y la especificación ZigBee. http://www.zigbee.org/Specifications/ZigBee/Overview.aspx

[30] Librerías de JavaScript "*JQuery*", utilizadas en la interfaz web de Surveillance 4. http://jquery.com/

[31] Librerías de JavaScript "*Bootstrap*", utilizadas en la interfaz web de Surveillance 4. http://getbootstrap.com/

[32] Plugin de JQuery utilizado para la visualización de las gráficas en la interfaz web. http://www.jqplot.com