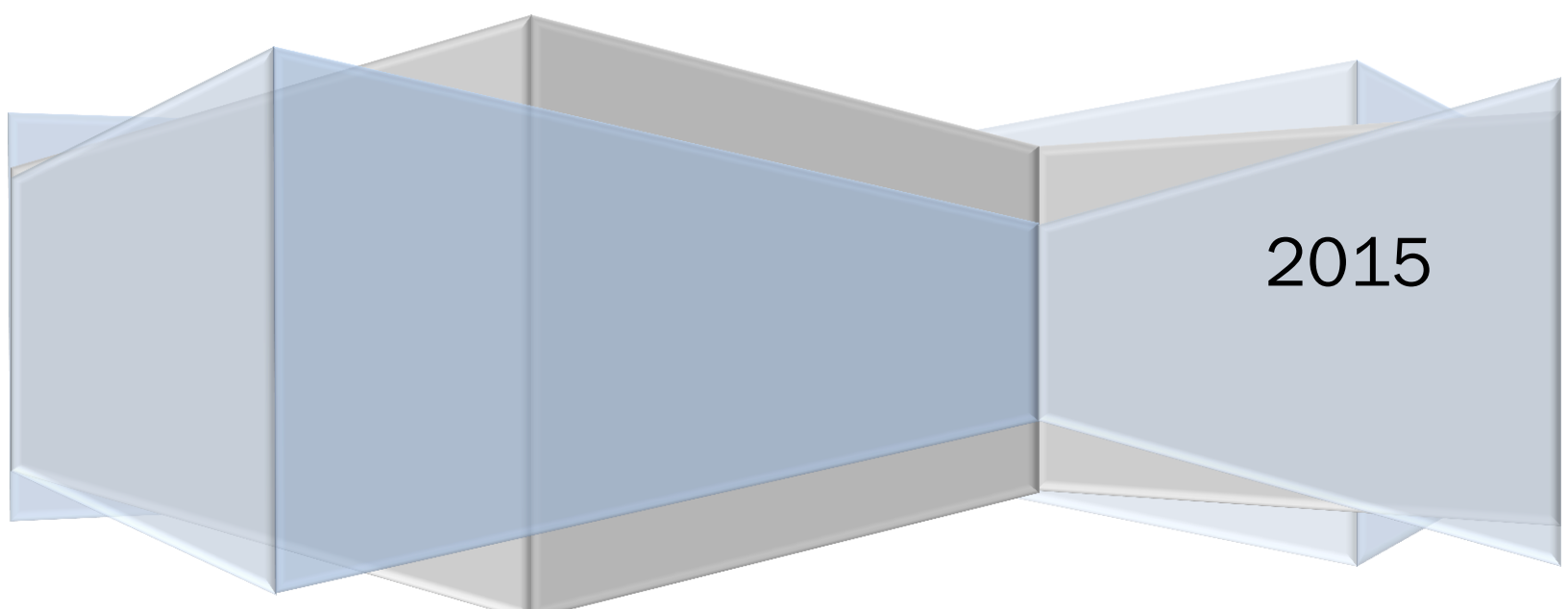


UNIVERSIDAD DEL NORTE

UNDomoDiscovery

Manual de Hardware y Software

Alejandra Fernández Castro



2015

UNDomDiscovery:

Manual de Hardware y Software

Alejandra Fernández Castro
Universidad del Norte
Barranquilla. Colombia

Tabla de contenido

.....	2
1. UNDomDiscovery: Manual de Hardware	2
1.1 Componentes del sistema	2
1.1.1 Home Gateway	2
1.1.2 Dispositivos Domésticos.....	4
1.2 Configuración de los componentes	5
1.2.1 Beaglebone Black	5
1.2.2 Arduino UNO	8
1.2.3 Arduino YUN.....	8
2. UNDomDiscovery: Manual de Software	9
2.1 Descripción del Home Gateway.....	9
2.1.1 Procesos de automatización doméstica.....	9
2.1.1.1 Proceso de descubrimiento de dispositivos.....	9
2.1.1.2 Proceso de configuración de dispositivos	12
2.1.1.2.1 Tipos de dispositivos soportados	12
2.1.1.2.2 Características y comandos.....	13
2.1.1.2.3 Mensaje de configuración.....	15
2.1.1.2.4 Ejecución de control	16
2.1.2 Gestión de bases de datos	17
2.2 Descubrimiento de los dispositivos domésticos	18
2.2.1 Programación en Arduino	21
2.2.1.1 Arduino UNO	21
2.2.1.2 Arduino YUN	21
2.3 Archivos de programación	22
2.3 Iniciando el sistema UNDomDiscovery	23
ANEXO A.....	25
ANEXO B.....	28
ANEXO C.....	32

UNDomoDiscovery:

Manual de Hardware y Software

UNDomo es un servicio de automatización doméstica o domótica diseñado por la Universidad del Norte. “UN” está dado por UNinorte y “Domo” por domótica. Dicho sistema administra los dispositivos instalados en una casa, con respecto a las preferencias de un usuario y a través de un dispositivo electrónico (celular, Tablet, portátil, etc). Dicho dispositivo gestiona la infraestructura de red doméstica a través de la conectividad WiFi con el Home Gateway, el cual se encarga de realizar el descubrimiento y las configuraciones de los equipos. Lo anterior, está representado con dos sistemas: UNDomoserver y UNDomosDiscovery.

- UNDomoserver corresponde al sistema que administra la interfaz gráfica de usuario. Gestiona los perfiles de los dispositivos y permite que el usuario pueda configurarlos de acuerdo con sus atributos. Recibe los perfiles de los dispositivos por parte del Home Gateway, al cual envía peticiones de configuración.
- UNDomosDiscovery corresponde al sistema que administra la infraestructura de red doméstica de los dispositivos instalados en una casa. Realiza el descubrimiento y la configuración de los dispositivos de acuerdo con las peticiones de usuario enviadas por el UNDomosServer.

A continuación se realiza una descripción de Hardware y Software del sistema UNDomosDiscovery.

1. UNDomosDiscovery: Manual de Hardware

La implementación en Hardware del Sistema UNDomosDiscovery está dada por: un Home Gateway, el cual opera como el cerebro del sistema, y los dispositivos domésticos, los cuales corresponden a los elementos finales que se desean controlar. El protocolo de comunicación usado para la conectividad entre estos dos componentes corresponde a WiFi o Wireless Fidelity. Por lo tanto, para garantizar un correcto funcionamiento ambos deben estar conectados a la misma red WiFi doméstica. Al ser la primera versión de este sistema UNDomos, se usan tarjetas de desarrollo que permiten simular una pasarela doméstica y los dispositivos finales.

1.1 Componentes del sistema

1.1.1 Home Gateway

El Home Gateway o pasarela doméstica es un sistema que permite el descubrimiento de los dispositivos domésticos. Una vez descubiertos, éstos pueden ser configurados de acuerdo con las preferencias del usuario, es decir, se puede realizar cambios en su estado (encendido/apagado), el nombre asignado por el usuario (identificativo), atributos del dispositivo (nivel, color, bloqueo), etc.

Este componente del sistema es simulado con una tarjeta de desarrollo llamada **Beaglebone Black**. Esta tarjeta posee un microprocesador incorporado de estándar abierto y bajo costo,

diseñado por Texas Instruments e implementado por desarrolladores de software y aficionados. Posee las funcionalidades básicas de un computador con diferentes opciones de sistema operativo. Posee los siguientes componentes.

- ☐ Procesador ARM Cortex-A8 con Sitara.
- ☐ Memoria RAM DDR3 de 512MB.
- ☐ Desempeño de 1GHz.
- ☐ Memoria Flash de almacenamiento de 4GB eMMC.
- ☐ Acelerador de gráficas 3D.
- ☐ Compatibilidad de software con Android, Debian, Android, Ubuntu y Cloud9.
- ☐ Interfaces USB, USD, Ethernet, HDMI, microSD y 5V.
- ☐ 2 puertos SPI, 2 puertos I2C, 4.5 puertos seriales UART.
- ☐ 7 entradas analógicas de 1.8V.
- ☐ 4 PWMs y 4 temporizadores.
- ☐ 65 interfaces digitales I/O

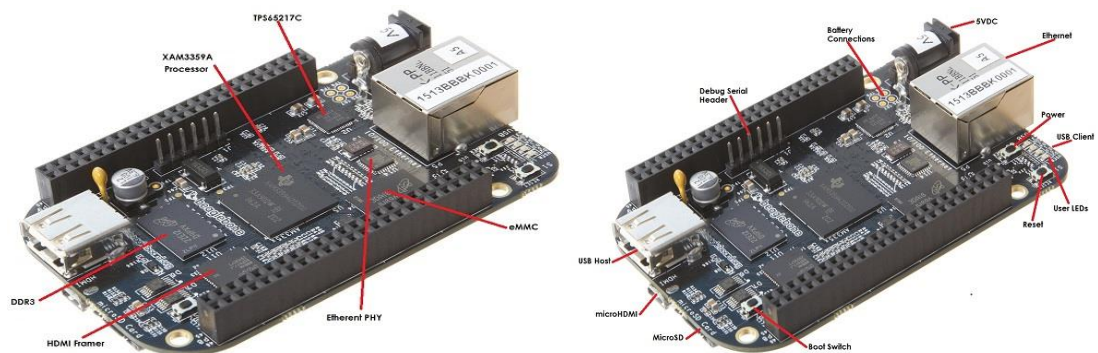


Fig. 1. Beaglebone Black: Componentes e interfaces.

Esta tarjeta de desarrollo no posee conectividad WiFi, sin embargo a través del puerto USB disponible se puede conectar un adaptador WiFi que le permita conectarse a una red, enviar y recibir datos. El **adaptador USB nano inalámbrico** implementado para esta tarjeta corresponde a la versión EW-7811Un de Edimax. El cual opera con los estándares IEEE 802.11b/g/n de WiFi. Posee una velocidad de transmisión de datos de hasta 150Mbps y soporta un rango de frecuencia de 2.4 a 2.4835GHz. Es usada en redes WiFi no seguras (sin clave), seguras (con clave y sin/con encriptación). Posee un sistema de ahorro de energía que le permite autoajustarse.



Fig. 2. Adaptador WiFi conectado a la Beaglebone Black.

1.1.2 Dispositivos Domésticos

Los dispositivos domésticos corresponden a los electrodomésticos o elementos electrónicos que desempeñan una tarea en el hogar. Son capaces de conectarse a una red WiFi, a través de la cual envían y reciben datos de parte de la pasarela doméstica. Estos elementos son simulados por **tarjetas de desarrollo Arduino**, de las cuales se implementan dos: una de referencia UNO y otra de referencia YÚN.

La tarjeta de desarrollo Arduino UNO es de desarrollo libre, fácil para programar y con bajo consumo de potencia. Está basado en el microcontrolador ATmega328 con 14 puertos digitales para entrada/salida, 6 entradas análogos, un resonador cerámico de 16MHz, 2Kbyte de memoria RAM, posee una interfaz USB y un botón de reset. Soporta una interfaz de desarrollo de programación llamada IDE de Arduino. Es considerada adecuada para proyectos de controles sencillos que requieren interacción y manipulación de variables, tales como la humedad, lux, etc. Arduino UNO ofrece la posibilidad de extender sus capacidades a través de caparazones o 'shields', las cuales se conectan a sus puertos disponibles. Las capacidades extendidas pueden ser conectividad Ethernet o WiFi. Por lo tanto, para obtener conectividad WiFi, se requiere adquirir el caparazón WiFi o 'WiFi Shield' para Arduino UNO.



Fig. 3. Tarjeta de desarrollo: Arduino UNO.

Además de lo anterior, se utiliza la tarjeta de desarrollo Arduino YÚN. Dicha tarjeta combina la potencia del sistema operativo Linux con las características generales de desempeño de Arduino. Opera con una distribución de Linux llamada Linino la cual está basada en OpenWRT. Soporta la red cableada Ethernet de 10/100Mbps, y la red inalámbrica WiFi que soporta los estándares IEEE 802.11b/g/n. El chip de Arduino está conectado al módulo de Linux por lo cual la comunicación entre ambos se encuentra habilitada para distribuir los procesos según convenga. Posee un procesador Atheros AR9331, una memoria RAM de 64MB DDR2, memoria flash de 32MB, una interfaz USB y lector de tarjetas Micro-SD.



Fig. 4. Tarjeta de desarrollo: Arduino YÚN.

En la Figura 5 se presenta un modelo general de la implementación de Hardware del sistema de automatización doméstica UNDomó, en la cual se presenta la interacción entre el Home Gateway y los dispositivos domésticos a través del protocolo de comunicación WiFi.

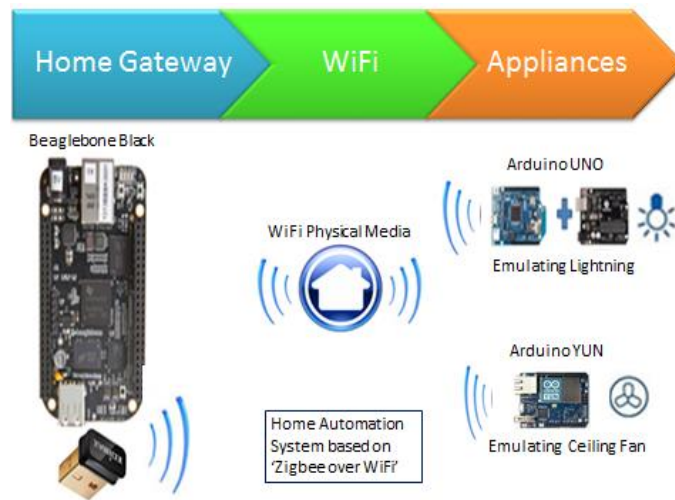


Fig. 5. Implementación de Hardware de UNDomó.

1.2 Configuración de los componentes

1.2.1 Beaglebone Black

La tarjeta de desarrollo Beaglebone Black opera con una versión de Linux beaglebone 3.8.13-bone 50. Posee el lenguaje de programación Java habilitado en su versión 1.8.0_06 y opera con una base de datos Mysql versión 5.5.40-0+wheezy.

La conectividad con la BBB se puede realizar a través de una conexión serial, SSH a través de cableado Ethernet o SSH a través de un cable USB. Se utiliza la conectividad SSH a través de USB, para ello debemos tener en cuenta la instalación de los drivers de acuerdo con el sistema operativo (SO) desde el cual se realiza la gestión de la tarjeta. El SO correcto es requerido para el reconocimiento de la tarjeta de desarrollo. Estos pueden ser descargados en el siguiente enlace: <http://beagleboard.org/getting-started>.

Una vez reconocida la tarjeta se procede a acceder al servidor disponible en la dirección IP /192.168.2.7/, en la cual se confirma la correcta instalación de los drivers y la disponibilidad de la tarjeta para ser configurada.

Se implementa el software Putty para realizar la conectividad vía SSH con la Beaglebone Black (BBB). Inicialmente se desea actualizar la tarjeta con el último software o sistema operativo disponible. Para ello se sigue el procedimiento descrito en la página web mencionada anteriormente. Se realiza la instalación de la imagen que contiene el firmware correspondiente a 'Debian-2GB eMMC' para BBB a través de una tarjeta microSD. Una vez finalizado el proceso de instalación se verifica con los comandos presentados en la Tabla 1.

Tabla 1. Códigos para la verificación de la instalación

Código	Respuesta	
lsb_release -a	Consultar el sistema operativo	Distributor ID: Debian Description: Debian GNU/Linux 7.7 (wheezy) Release: 7.7 Codename: wheezy
uname -a	Consultar la version de Linux	Linux beaglebone 3.8.13-bone50 #1 SMP Tue May 13 13:24:52 UTC 2014 armv7l GNU/Linux

A continuación se procede a configurar la conectividad a Internet de la BBB, lo cual se puede realizar a través de cableado USB o un adaptador WiFi conectado en el puerto USB disponible. Para ello se implementa el adaptador WiFi de la tecnología Edimax que soporta Wifi 802.11b/g/n. El uso de este adaptador requiere la implementación de la alimentación de 5V de la BBB.

Una vez instalado el módulo y conectada la alimentación procedemos a verificar que la BBB reconoce el dispositivo, y a realizar las configuraciones apropiadas para obtener la conexión a Internet. Dicho proceso se visualiza en la tabla 2.

Tabla 2. Reconocimiento y funcionamiento de adaptador WiFi

Código	Respuesta	
lsusb	Bus 001 Device 002: ID 7392:7811 Edimax Technology Co., Ltd EW-7811Un 802.11n Wireless Adapter [Realtek RTL8188CUS] Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub	
ifconfig wlan0	wlan0	Link encap:Ethernet HWaddr 80:1f:02:bf:12:cd UP BROADCAST MULTICAST MTU:1500 Metric:1 RX packets:0 errors:0 dropped:0 overruns:0 frame:0 TX packets:0 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)
iwconfig wlan0	wlan0	IEEE 802.11bgn ESSID:off/any Mode:Managed Access Point: Not-Associated Tx-Power=20 dBm Retry long limit:7 RTS thr=2347 B Fragment thr:off Encryption key:off Power Management:off
iwlist wlan0 scan	#Detecta las redes Wifi disponibles Cell 01 - Address: 68:B6:FC:2A:CD:0F Channel:11 Frequency:2.462 GHz (Channel 11) Quality=70/70 Signal level=-35 dBm Encryption key:on ESSID:"UNE-CD0C" Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 9 Mb/s	

```

18 Mb/s; 36 Mb/s; 54 Mb/s
Bit Rates:6 Mb/s; 12 Mb/s; 24 Mb/s; 48 Mb/s
Mode:Master
Extra:tsf=0000000d43a6c837
Extra: Last beacon: 62ms ago

```

Una vez verificado que la BBB reconoce el módulo de WiFi representado a través de la interfaz **wlan0**, se procede a configurarla. Dicho procedimiento se puede realizar de diferentes formas, presentadas a continuación.

A. Se realiza la modificación directa en el documento que identifica las configuraciones de las interfaces de la BBB, incluyendo la wlan0. Ver Tabla 3.

Tabla 3. Modificación de la configuración de las interfaces	
Código	Configuración
Sudo nano /etc/network/interfaces	#Configuración de la interfaz wlan0 auto wlan0 iface wlan0 inet dhcp wpa-ssid "Mi_Wifi" wpa-psk "Clave"
sudo /etc/init.d/networking restart	#Se reinician los servicios
iwconfig wlan0	#Se verifica la conexión exitosa

B. Se realiza la configuración WiFi directamente desde consola. Ver Tabla 4.

Tabla 4. Proceso de configuración en terminal	
Código	
sudo iwconfig wlan0 essid "Mi_Wifi" key s:Clave	
sudo dhclient wlan0	
#Se verifica la conexión exitosa iwconfig wlan0	

C. Se utiliza una herramienta de Debian llamada 'wicd' que permite la administración de redes alámbricas e inalámbricas. Ver Tabla 5.

Tabla 5. Herramienta wicd	
Código	
Wicd-curses	
#Conexión a una red C	
#Configuración de una red para ingresar contraseña ->	

Se puede verificar la conectividad a Internet realizando pings a estaciones cercanas o al punto de acceso de WiFi. Se procede a realizar la conexión a la red WiFi en la cual se va a realizar el control y configuración del sistema de automatización doméstica.

1.2.2 Arduino UNO

La tarjeta de desarrollo Arduino UNO no posee conectividad WiFi, por lo tanto se requiere un WiFi shield que lo habilite. Una vez conectada la shield en el arduino UNO, se procede a verificar la versión de operación del Firmware de la misma. Normalmente, se obtiene fallos en los primeros pasos con la shield, por lo tanto se requiere la actualización de la misma haciendo uso de su puerto serial. La WiFi shield implementada en este proyecto posee instalada la versión 1.0 y requirió del proceso de instalación mencionado. Se verifica la versión de Firmware de la WiFi shield a través del código presentado en la Figura 6.

```
#include <SPI.h>
#include <WiFi.h>

void setup() {
  //Initialize serial and wait for port to open:
  Serial.begin(9600);

  // check for the presence of the shield:
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue:
    while(true);
  }

  Serial.print("Firmware version: ");
  Serial.println(WiFi.firmwareVersion());
}

void loop() {}
```

Fig. 6. Verificación de la versión del Firmware del Arduino UNO.

Una vez se obtenga una versión apropiada de funcionamiento de la WiFi shield se procede a verificar su funcionamiento con las últimas versiones del IDE de Arduino. La versión implementada es la 1.5.8. Se verifica el correcto funcionamiento de la WiFi shield al implementar códigos para el escaneo de redes WiFi disponibles y la conectividad a una red especificada. Dichos códigos se presentan en el ANEXO A.

1.2.3 Arduino YUN

La tarjeta de desarrollo Arduino YUN posee conectividad WiFi, por lo tanto el procedimiento para conectarse a través de WiFi es diferente a la del Arduino UNO. Inicialmente se conecta la YUN al dispositivo de configuración: portátil o Pc. Visualizaremos en el escaneo de las redes WiFi del equipo que aparece una red llamada '**Arduino-Yun:XXX**' a la cual se procede a conectar. Una vez conectados podremos acceder a la dirección IP 192.168.240.1 para configurar la red WiFi, en la cual se especifica la red a la cual nos queremos conectar y su contraseña. Posteriormente, se requiere conectar inmediatamente a la red WiFi especificada en la configuración.

Se verifica la conectividad WiFi a través de pings al punto de acceso de WiFi o a equipos de trabajo cercanos. La programación del Arduino YUN se puede realizar a través del IDE de

Arduino o a través del intérprete de Python instalado por defecto. Sin embargo, el envío y recepción de paquetes a través de WiFi es llevado a cabo a través del Python, por lo tanto se utiliza este lenguaje para la programación.

Teniendo en cuenta las configuraciones de hardware presentadas anteriormente, se procede a describir las configuraciones de software implementadas en cada uno de los componentes del sistema de automatización doméstica.

2. UNDomDiscovery: Manual de Software

La implementación en software del sistema UNDomDiscovery está dada por un Home Gateway que envía mensajes UDP tipo broadcast a un puerto predefinido, a través de WiFi. Este puerto se encuentra previamente configurado en los dispositivos domésticos, por lo tanto éstos responderán al mensaje tipo broadcast de acuerdo con un protocolo de interacción previamente definido. Dado lo anterior, otros dispositivos que no posean ese protocolo habilitado, no será capaces de responder ni analizar los mensajes que recibe por parte del Home Gateway. A continuación se describen las implementaciones en software llevadas a cabo en el home Gateway y en los dispositivos domésticos, simulados por la Beaglebone Black y Arduino (UNO y YUN), respectivamente.

2.1 Descripción del Home Gateway

El Home Gateway simulado por la tarjeta de desarrollo Beaglebone Black (BBB) posee implementado un servicio de automatización doméstica que posee dos procesos: uno de descubrimiento y otro de configuración de los dispositivos domésticos. Estos procesos están definidos como una secuencia de pasos que se llevan a cabo para alcanzar la meta planteada. La programación de los procesos se realiza a través del lenguaje Java en la plataforma Cloud9 instalada por defecto en la BBB.

2.1.1 Procesos de automatización doméstica

Los procesos de automatización describen el procedimiento secuencial que se debe llevar a cabo para descubrir y configurar los dispositivos domésticos. Dichos procesos se ejecutan constantemente y en conjunto, mediante el envío y recepción de paquetes de datos en un puerto previamente especificado por el Home Gateway. El archivo principal del servicio de automatización corresponde a **UNDomDiscovery.java**, el cual define los puertos a utilizar, realiza envío de mensajes Broadcast, analiza los paquetes recibidos y ejecuta procedimientos de acuerdo con el contenido de los paquetes.

2.1.1.1 Proceso de descubrimiento de dispositivos

Se define el puerto UDP 9000 como puerto predefinido para el envío y recepción de paquetes de datos. A través de este puerto se realiza el envío de paquetes tipo Broadcast a todos los dispositivos pertenecientes a una red. Es necesario mencionar que si el Home Gateway posee cableado Ethernet y Wifi, entonces los paquetes tipo Broadcast serán enviados en ambas redes. Por lo tanto, dispositivos conectados en la red Ethernet también pueden ser descubiertos. Lo anterior, teniendo en cuenta la operación del estándar en la capa de aplicación conocido como 'Zigbee sobre WiFi', sin el cual el dispositivo no es capaz de responder ni analizar los paquetes de datos recibidos.

El proceso de descubrimiento de dispositivos se lleva a cabo a través de un conjunto de pasos descritos a continuación.

1. Definición de puertos UDP y envío de mensajes tipo Broadcast: El mensaje es enviado a todas las direcciones IP conectadas en las redes Ethernet o WiFi a las cuales pertenece el Home Gateway. Los mensajes se envían al puerto 888, el cual es previamente definido en los dispositivos de tal forma que éstos consulten constantemente la recepción de paquetes en este puerto. El mensaje enviado corresponde a **“SERVER BROADCAST”**.
2. Análisis de mensajes recibidos: Se analizan los mensajes recibidos en el puerto 9000 de la BBB. Existen diferentes tipos de mensajes que son aceptados por el Home Gateway, los cuales se presentan y describen a continuación:

A. Mensaje de respuesta de Broadcast.

Los dispositivos domésticos responden al Broadcast con la misma premisa, es decir, **“SERVER BROADCAST”**. Al recibir esta respuesta el Home Gateway procede a analizar si la dirección IP del dispositivo ya se encuentra guardada en la base de datos.

- Si la dirección IP se encuentra guardada se procede a realizar un procedimiento de **“REFRESH”** de la dirección IP, como verificación de que ésta se encuentra activa y en operación. Este procedimiento a su vez verifica que la operación activa de todas las direcciones IP sean verificadas.
- Si la dirección IP no se encuentra guardada en la base de datos se procede a enviar un mensaje tipo **“DISCOVER”**, el cual indica al dispositivo que debe enviar sus datos característicos.

B. Mensaje de respuesta de Discovery.

Este tipo de mensaje es identificado a través del indicativo inicial **“ID”**. Dicho indicativo permite inferir que el siguiente dato corresponde al identificador del dispositivo provisto por el fabricante. Por lo tanto, se procede a analizar el resto de atributos entregados por el dispositivo. La cantidad y tipo de atributos recibidos depende del tipo de dispositivo descubierto. Todos los dispositivos soportan características de encendido y apagado, mientras que algunos soportan características de nivel, color o bloqueo.

Al analizar los atributos de cada tipo de dispositivo se procede a guardar la información en las bases de datos, teniendo en cuenta el tiempo en el que se ha descubierto el dispositivo. Se guardan los atributos referentes a nivel, color, bloqueo y encendido/apagado. Las bases de datos serán descritas y mencionadas en la sección de gestión de bases de datos.

Además de lo anterior, se procede a generar un perfil de cada dispositivo el cual es posteriormente enviado al Home Server para su administración en la aplicación Web/Móvil.

El sistema de automatización soporta 10 tipos de dispositivos, los cuales son a su vez soportados por el protocolo de automatización de Zigbee. Estos dispositivos son simulados de acuerdo con su funcionalidad y operación. Los tipos de dispositivos soportados se describen en la sección de descripción de los dispositivos domésticos. En el ANEXO B se presentan ejemplos de los perfiles de los dispositivos generados, en los cuales se definen características

básicas de identificación, estado de encendido/apagado, tiempo de generación del perfil, características de bloqueo, nivel, color, etc. En la Tabla 6 se presenta una descripción de todos los posibles atributos que se especifican en el perfil de los dispositivos.

Tabla 6. Atributos del perfil de los dispositivos	
Atributo	Descripción
ID	Identificación del dispositivo. Es definido por el fabricante y pretende ser único para cada dispositivo.
NETWORK TYPE	Identifica el tipo de red soportada por el dispositivo.
DEVICE TYPE	Identifica el tipo de dispositivo.
NAME	Nombre por defecto o nombre configurado por el usuario.
SCENE	Escena en la que se ubica el dispositivo.
ON/OFF STATE	Estado de encendido/apagado del dispositivo.
LAST DATE	Última fecha registrada de la modificación del dispositivo.
MAX VALUE	Máximo valor de nivel permitido.
MIN VALUE	Mínimo valor de nivel permitido.
VALUE	Valor de nivel configurado.
LOCK STATE	Estado de bloqueo/desbloqueo del dispositivo.
LUMINANCE	Configuración de la iluminación del color.
HUE	Configuración de la tez del color.
SATURATION	Configuración de la saturación del color.

En la Figura 7 se presenta un modelo descriptivo del funcionamiento del proceso de descubrimiento de los dispositivos, en el cual se representa la operación en software del sistema, la interacción de sus procesos y los mensajes que los describen. En dicha representación se supone que la dirección IP del dispositivo no es conocida, por lo tanto se procede a realizar el descubrimiento del mismo. Al descubrir el dispositivo se procede en almacenar su información, crear su perfil y enviarlo al Home Server, para que éste pueda ofrecerle información al usuario sobre los dispositivos conectados en su área local inalámbrica.

C. Mensaje de respuesta de Configuration.

Este tipo de mensaje es identificado a través del indicativo inicial “**DEV**”. Dicho indicativo permite inferir que el mensaje es proveniente del Home Server y que contiene una petición de configuración que debe ser realizada sobre un dispositivo a petición de un usuario. El análisis y ejecución de este mensaje corresponde al proceso de configuración de dispositivos, descrito en la sección de mensaje de configuración.

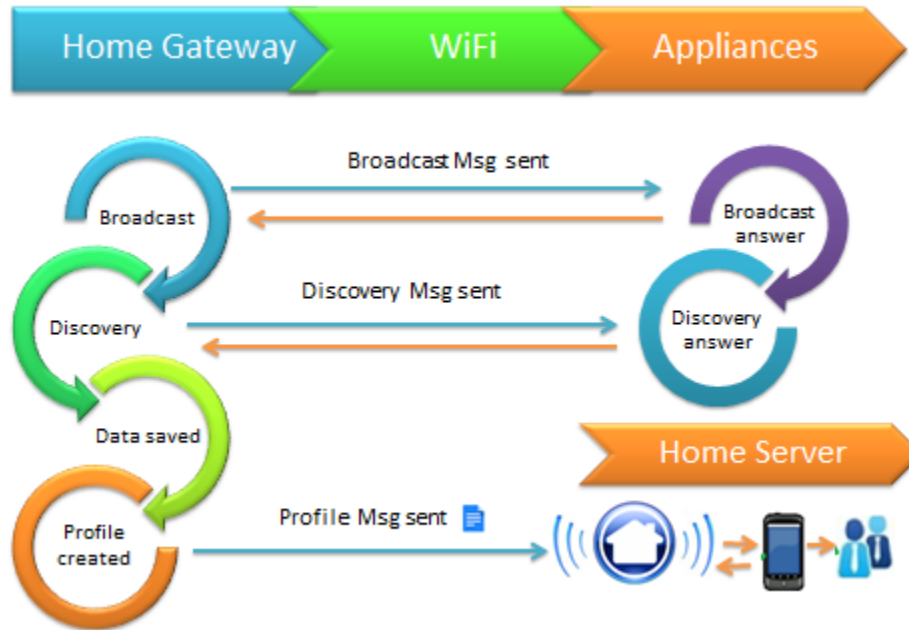


Fig. 7. Proceso de descubrimiento de un dispositivo.

2.1.1.2 Proceso de configuración de dispositivos

Una vez el home server posee información sobre el perfil de los dispositivos, este es capaz de crear a su vez un perfil en el que el usuario es capaz de modificar atributos como estado, nivel, bloqueo y color. Cuando los usuarios realizan una modificación en el atributo de un dispositivo, esto se traduce en una petición de mando de configuración que es enviada por el home server al Home Gateway. El Home Gateway a su vez, envía el mensaje de configuración al dispositivo respectivo, el cual confirma que su estado ha cambiado. Una vez se confirma el cambio, se procede a actualizar las bases de datos y el perfil del dispositivo. Este procedimiento, se explica con detalle posteriormente.

El análisis de los mensajes de configuración enviados requiere el conocimiento del tipo de dispositivo y los atributos que posee, de tal forma que se identifiquen los comandos que permiten modificar sus atributos.

2.1.1.2.1 Tipos de dispositivos soportados

Se seleccionaron 10 de los dispositivos soportados por el protocolo de automatización de Zigbee. Dichos dispositivos poseen funcionalidades de encendido/apagado, configuración de nivel, bloqueo y color. En la Tabla 7 se presentan los dispositivos soportados, su ID y su descripción. El ID definido por Zigbee Home Automation (HA) es mantenido por UNDomó.

Tabla 7. Tipos de dispositivos soportados		
Dispositivo	ID	Descripción
On/Off Switch	0x0000	Interruptor que permite el encendido y apagado.
Level Control Switch	0x0001	Interruptor que permite definir el nivel.
Door Lock	0x000A	Bloqueo de puerta.
Simple Sensor	0x000C	Interruptor digital que permite el encendido y el apagado.

On/Off Light	0x0100	Permite el encendido y apagado de la luz.
Dimmable Light	0x0101	Configuración del nivel de iluminación.
Color Dimmable Light	0x0102	Configuración del nivel de iluminación y el color de la luz.
On/Off Light Switch	0x0103	Interruptor de luz que permite el encendido y apagado.
Dimmer Switch	0x0104	Interruptor que permite la configuración del nivel.
Color Dimmable Switch	0x0105	Interruptor que permite configurar el nivel y el color de un atributo.

Notemos que algunos de los dispositivos sólo soportan características de encendido y apagado, mientras que otro soportan bloqueo, nivel o color. Cada una de estas características o clusters definen un grupo de comandos que permiten las configuraciones de los dispositivos. Por lo tanto, el home server es capaz de identificar los comandos soportados por los dispositivos y enviarlos de acuerdo con el atributo que desea modificar. A continuación se presentan las características y los comandos soportados por cada una.

2.1.1.2.2 Características y comandos

Zigbee HA define clusters que permiten caracterizar los atributos configurables que posee un dispositivo. Entre la gran cantidad de cluster definidos se selecciona la adecuación de 6 clusters, los cuales permiten caracterizar los dispositivos domésticos seleccionados. Las características o clusters soportadas por los dispositivos pueden ser: encendido/apagado, bloqueo, nivel o color. Todos soportan el encendido/apagado, sin embargo los otros clusters dependen de la descripción y atributos del dispositivo. En algunos dispositivos se da una mezcla de estos como es el caso del “Color Dimmable Switch” el cual posee atributos de encendido/apagado, nivel y color.

En la Tabla 8 se presentan las 6 características que son soportadas por el sistema UNDDomo. El ID de estas características corresponde al mismo identificador asignado por Zigbee HA, sin embargo los comandos especificados son diseñados de acuerdo a los atributos modificables que posee un dispositivo.

Tabla 8. Características soportadas			
Característica	ID	Descripción	Comandos
Scenes	0x0005	Define la escena o ubicación.	SET SCENE
On/Off	0x0006	Estado del dispositivo. Permite definir estado de encendido, apagado o cambio de estado.	SET NAME SET ON SET OFF SET TOGGLE
Level Control	0x0008	Modifica el nivel de una característica.	SET VALUE
Binary Input	0x000F	Soporta estados binarios de encendido y apagado.	SET NAME SET ON SET OFF
Door Lock	0x0101	Bloquear o desbloquear un atributo.	SET LOCK SET UNLOCK
Color Control	0x0300	Controla atributos de color, tales como iluminación, saturación y tez.	SET LUMINANCE SET HUE SET SATURATION

En el ANEXO C se ofrece una descripción detallada de las características de los dispositivos y los comandos que soportan. Los comandos poseen longitudes de texto diferente, por lo tanto se definen códigos de una longitud de texto definida que permite identificar el comando que se desea modificar. Dado lo anterior. El Home Server envía los códigos que representan modificaciones en los atributos de un dispositivo. En la Tabla 9 se presenta la relación de códigos y comandos. Además, indica si el comando requiere un valor adicional. En el caso del código CMD004, se requiere un valor dado que pretende configurar el nivel de una característica. Por lo tanto, el valor que lo acompañe es el nuevo atributo que define ese nivel.

Tabla 9. Comandos y códigos		
Código	Comando	Valor
CMD00	SET NAME	No requerido
CMD01	SET ON	No requerido
CMD02	SET OFF	No requerido
CMD03	SET TOGGLE	No requerido
CMD04	SET VALUE	Requerido
CMD05	SET SCENE	Requerido
CMD06	SET LOCK	No requerido
CMD07	SET UNLOCK	No requerido
CMD08	SET LUMINANCE	Requerido
CMD09	SET HUE	Requerido
CMD10	SET SATURATION	Requerido

En la Tabla 10 se presenta la relación entre los dispositivos domésticos implementados y las características que los soportan, lo cual a su vez indica los códigos que permiten configurar a un dispositivo. Notemos que todos los dispositivos soportan la característica de escenas configurables. La mayoría es configurable con “On/Off”, esto porque el dispositivo “Simple Sensor” lo modifica por la característica “Binary Input”.

Tabla 10. Relación de dispositivos y características	
Dispositivo	Características
On/Off Switch	Scenes - On/Off
Level Control Switch	Scenes - On/Off - Level Control
Door Lock	Scenes - On/Off - Door Lock
Simple Sensor	Scenes - Binary Input
On/Off Light	Scenes - On/Off
Dimmable Light	Scenes - On/Off - Level Control
Color Dimmable Light	Scenes - On/Off - Level Control - Color Control
On/Off Light Switch	Scenes - On/Off
Dimmer Switch	Scenes - On/Off - Level Control
Color Dimmable Switch	Scenes - On/Off - Level Control - Color Control

La relación de los dispositivos, sus características y los comandos que los soportan debe ser entendida tanto por el Home Gateway como por el Home Server. Lo anterior, permite que exista interoperabilidad en el sistema automatizado.

2.1.1.2.3 Mensaje de configuración

Los mensajes de configuración se construyen con base en los conocimientos de las características y comandos soportados por un dispositivo. Cuando un usuario ejecuta un cambio en la configuración, esto repercute en el envío de un mensaje que notifica el cambio de atributo. Dicho mensaje posee 4 campos que los describen, los cuales se presentan en la Tabla 11.

Tabla 11. Campos de un mensaje de configuración	
Campo	Descripción
ID	ID del dispositivo sobre el cual el usuario requiere ejecutar una configuración. Ej: ONF123
CODE	Especifica el código que modifica un atributo del dispositivo. Este código es soportado por uno de los clusters que lo caracterizan. Ej: CMD01
VALUE	Algunos códigos requieren valores adicionales. Cuando no es requerido no es evaluado. Ej: 1
DATE	Fecha y tiempo de la ejecución del mensaje de configuración. Ej: 2015-04-15 10:22:18

El Home Gateway recibe un mensaje con los campos presentados anteriormente y procede a evaluar la validez de cada campo, es decir, a verificar que el mensaje sea correcto. El procedimiento de validez se describe a continuación:

- Inicialmente verifica si el ID existe en la base de datos.
 - Si no existe, envía un mensaje de error: **FAIL ID**.
 - Si existe procede a verificar el siguiente campo.
- Verifica si el código presentado es soportado por el dispositivo. Con base en el ID obtiene el tipo de dispositivo y las características que soporta. Obtiene así mismo los códigos soportados.
 - Si el código no es soportado, envía un mensaje de error: **FAIL CODE**.
 - Si es soportado procede a verificar el siguiente campo.
- Verifica la validez del campo VALUE. De acuerdo con el código identifica si el campo es relevante o no.
 - Si el campo es relevante verifica que sea una representación numérica del atributo que va a modificar. Al verificarlo procede a identificar si el valor numérico pertenece al rango de valores permitidos para el atributo. Si existe un error en estas validaciones se envía un mensaje de error: **FAIL VALUE**.
 - Si el campo no es relevante procede a verificar el siguiente campo.
- Finalmente se valida si la fecha y el tiempo son correctos, es decir, si son mayores que la fecha y el tiempo registrados en la última modificación. Lo anterior, para descartar mensajes repetidos o que hayan llegado con retraso.

- Si la fecha y el tiempo no son válidos, se envía un mensaje de error: **FAIL DATE**.
- Si la fecha es correcta se procede a ejecutar el mensaje de configuración.

En la Figura 8 se modela el proceso de verificación de un mensaje de configuración recibido.

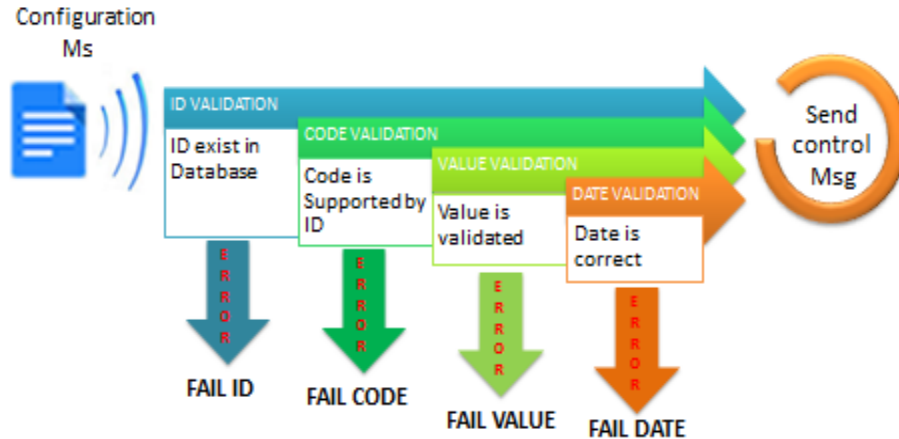


Fig. 8. Proceso de validación de un mensaje de configuración.

2.1.1.2.4 Ejecución de control

Una vez se ha verificado que el mensaje de configuración es correcto se procede a enviar dicho mensaje al dispositivo sobre el cual se requiere efectuar un cambio. Con base en el ID se obtiene la dirección IP y con base en el código se realiza el envío del comando correspondiente. Es decir, si el código es CMD01, se envía SET ON. Los dispositivos están previamente configurados para ejecutar una acción de acuerdo con los comandos recibidos. En la Figura 9 se presenta el proceso general requerido para la configuración de los dispositivos domésticos.

Al recibir el comando, el dispositivo procede a ejecutar el cambio en el atributo especificado y envía un mensaje de acknowledge o ACK al Home Gateway. Dicho mensaje indica que la acción ha sido ejecutada, sin embargo el Home Gateway procede a enviar un mensaje de **REFRESH**, a través del cual captura todos los estados de las variables del dispositivo, de tal forma que verifica que efectivamente el atributo ha sido modificado. Después de lo anterior, el Home Gateway procede a actualizar la información en las bases de datos y a actualizar el perfil del dispositivo. Así mismo, envía un mensaje de **OK** al Home Server indicando que su petición ha sido efectuada con éxito.

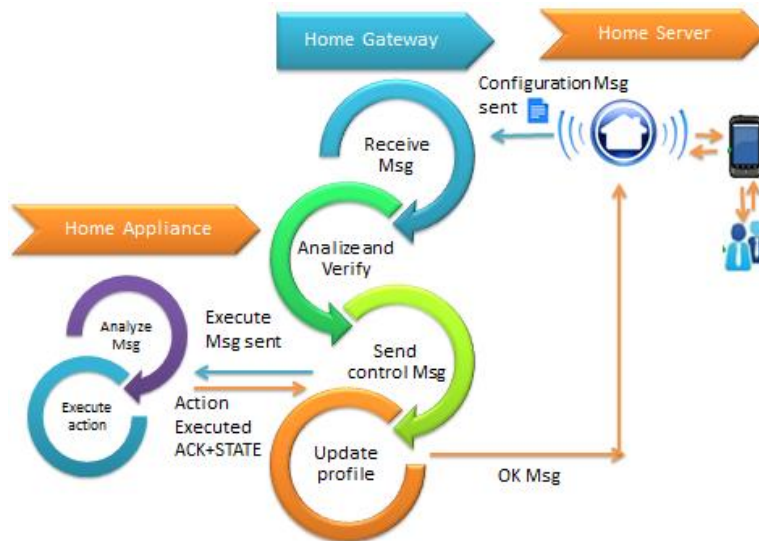


Fig. 9. Proceso de ejecución de mando sobre el dispositivo.

2.1.2 Gestión de bases de datos

Las bases de datos definidas en el Home Gateway son accedidas a través de consola con el comando:

Mysql -h localhost -u root -p

Password: mydebian

Se utiliza la base de datos UNDOMO con el commando:

USE UNDOMO;

Para visualizar todas las tablas pertenecientes a esta base de datos se digita el commando:

SHOW TABLES;

Para observar la información de una tabla en particular se escribe:

SELECT * FROM <NombredelaTabla>;

Las tablas existentes en la base de datos UNDOMO se mencionan y describen a continuación:

- **ADDRESS:** Guarda la información corresponde a la dirección IP del dispositivo asociada a un número de identificación asignado por el Home Gateway.
- **DEVICES:** Guarda información general de los dispositivos: identificación dada por el Home Gateway, identificación dada por el fabricante, red a la cual pertenece (de acuerdo a la tecnología o protocolo de comunicación), nombre asignado por el usuario, escena en la que se ubica el dispositivo, estado de encendido/apagado, fecha y hora de la última modificación.

- **APPLIANCES:** Indica el tipo de dispositivos que soporta el sistema, sus ID de acuerdo con lo establecido por Zigbee HA y el nombre predeterminado que se le asignan a los dispositivos.
- **CODES:** Relaciona los códigos enviados por el Home Server con los comandos que permiten la configuración de los dispositivos. Ej: CMD01→ SET ON.
- **CLUSTERS:** Especifica el grupo de características soportadas por el sistema y los comandos que operan con cada cluster dado. La presentación de la información de estos clusters corresponde a la que se presenta en el ANEXO C.
- **LEVEL:** Guarda la información referente a la característica de nivel de un dispositivo. Indica el identificador dado por el Home Gateway, los valores máximos y mínimos permitidos y el valor actual configurado.
- **COLOR:** Guarda la información referente a la característica de color de un dispositivo. Indica el identificador dado por el Home Gateway, los valores actuales de iluminación, tez y saturación.
- **LOCKCON:** Guarda la información referente a la característica de bloqueo de un dispositivo. Indica el identificador dado por el Home Gateway y el estado de bloqueo.
- **SCENES:** Especifica las escenas soportadas por el sistema de automatización.

2.2 Descubrimiento de los dispositivos domésticos

Los dispositivos domésticos son simulados por las tarjetas de desarrollo de Arduino, referencias UNO y YUN, las cuales son programadas en entornos de desarrollo diferentes. Arduino UNO es programada en el IDE de Arduino, mientras que Arduino YUN es programada a través de Python. El funcionamiento general de ambas programaciones corresponde a la habilitación de la conectividad WiFi con una red específica, la definición de las características de los atributos del dispositivo simulado, la definición de los tipos de mensajes que puede recibir, sus respuestas y las acciones que debe tomar. Una vez definido lo anterior, se procede a monitorear constantemente los paquetes de datos que llegan al puerto UDP 8888, este puerto es configurado para todos los dispositivos simulados.

A continuación se procede a explicar un proceso de interacción entre el Home Gateway y un dispositivo doméstico. En las Figuras 10 y 11 se presentan los mensajes de respuesta del Home Gateway ante un proceso de descubrimiento y configuración, respectivamente. En la Figura 12 se presenta las respuestas del dispositivo doméstico ante ambos procesos.

Se tiene un dispositivo del tipo “On/Off Switch” cuyo programa en Python se encuentra operando y monitoreando el puerto 8888 y el Home Gateway que se encuentra en proceso de descubrimiento de los dispositivos, se desarrolla el proceso de descubrimiento y configuración. El cual se presenta a continuación:

1. **Home Gateway:** Inicialmente el Home Gateway inicia un proceso de descubrimiento, a través del cual realiza el envío de mensajes tipo “SERVER BROADCAST”, que indica que se está haciendo un descubrimiento de los dispositivos instalados en la misma red de área local.

Dispositivo doméstico: Ante este mensaje, todos los dispositivos están programados para

responder con el mismo script, lo cual lo identifica como perteneciente a la red y apto para ser controlado y configurado por el Home Gateway.

2. **Home Gateway:** El Home Gateway al recibir el mismo script, verifica si el dispositivo ha sido previamente descubierto. En caso negativo, procede a descubrir las características del dispositivo a través del mensaje “DISCOVER”.

Dispositivo doméstico: Ante este mensaje, el dispositivo responde con un script donde se involucran todos sus atributos de identificación y configuración. Se utiliza el modelo de texto HTTP para la separación de los atributos entre sí, es decir, se usa el carácter “SP” para separar los datos. Por ejemplo: en el caso del switch los parámetros que debe enviar son:

- a. ID=ONF123 → Identificador provisto por fabricante.
- b. NET= WF → identificador de la red soportada. WF indica WiFi.
- c. TYPE=000 → Tipo de dispositivo. 000 indica On/Off Switch.
- d. STATE=0 → Estado del dispositivo. 0 indica apagado.

Estos atributos son enviados en un mensaje de esta forma: **IDONF123SPWFSP000SPO**

El “ID” inicial del mensaje permite que el Home Gateway identifique que se trata de un mensaje de descubrimiento, es decir, que contiene las características del dispositivo. Por lo tanto, procede a analizarlo, guardar la información en las bases de datos y a crear el perfil apropiado.

En la Figura 10 se visualiza el proceso de descubrimiento y análisis que realiza el Home Gateway, mientras que en la Figura 12 se visualizan los mensajes recibidos por el dispositivo doméstico.

3. El usuario al recibir la información del descubrimiento del dispositivo desea cambiar el estado a <encendido>, por lo tanto se crea un mensaje de configuración para cambiar el atributo estado del dispositivo. El mensaje de configuración posee los siguientes atributos:

- a. ID=ONF123 → Id del dispositivo que se desea modificar.
- b. CODE=CMD01 → Código de mando sobre el dispositivo. Indica SET ON.
- c. VALUE=1 → Valor no válido en el caso del comando CMD01.
- d. DATE=2015-03-25 10:29:49 → Fecha y hora de la petición de configuración.

Home Gateway: recibe el mensaje de configuración por parte del Home Server. Dicho mensaje es analizado, de tal forma que se verifique todos los atributos son correctos. Una vez verificados, se procede a enviar el mensaje al dispositivo identificando el comando que debe ser ejecutado.

Dispositivo doméstico: Recibe el comando “SET ON” y procede a ejecutar la acción requerida. En este caso, realiza el encendido visual de un Led de la tarjeta de desarrollo. Además, envía un ACK para confirmar que se ha ejecutado la acción.

4. **Home Gateway:** Recibe el mensaje ACK y procede a verificar que el estado del dispositivo sea de acuerdo con lo mandado. Para ello envía un mensaje de “REFRESH”, el cual solicita que el dispositivo envíe un mensaje con todos sus estados modificables.

Dispositivo doméstico: Recibe el mensaje de actualización de los estados y procede a enviar el estado de su único atributo modificable, es decir, el estado encendido/apagado. En este caso envía el mensaje: **REF1**, donde el indicativo “REF” indica que se trata de un mensaje de actualización.

5. **Home Gateway:** Recibe la información de actualización y procede a actualizar la información en las bases de datos y actualizar el perfil del dispositivo.

En la Figura 11 se observa el análisis realizado por el Home Gateway, el proceso de envío al dispositivo y las respuestas esperadas para confirmar que el procedimiento ha sido exitoso. En la Figura 12 se presentan todos los mensajes recibidos por el dispositivo simulado por Arduino YUN.

```

Discovering devices in the network:
Sending Broadcast Packet
Waiting for a received packet:
Message received SERVER BROADCAST
Waiting for a received packet:
Message received IDONF123SPWFSP000SP0
Discovery message received
Message: IDONF123SPWFSP000SP0
ID ONF123
NET WF
TYPE 000
String 4 state 0
STATE 0
Data saved!
Builder {"ID": "ONF123", "NETWORK_TYPE": "WF", "DEVICE_TYPE": "data", "NAME": "data", "
SCENE": "Not Assign", "ON_OFF_STATE": "0", "LAST_DATE": "2015-3-9 8:28:58", "CLUSTERS":
"data"}
File Created!
Waiting for a received packet:

```

Fig. 10. Proceso de descubrimiento en el Home Gateway

```

ID ONF123
CODE CMD01
VALUE 1
DATE 2015-03-25 10:29:49
VERID true
VERCODE true
VERTIME true
IPresul /192.168.2.228
Waiting for a received packet:
Message received ACK
ACK message received
Waiting for a received packet:
Message received REF1
REF message received REF1
New state 1
Waiting for a received packet:

```

Fig. 11. Proceso de configuración en el Home Gateway

```

root@Arduino:/mnt/sda2# python switch.py
UDPServer listening on port 8888
('Received message: ', 'SERVER BROADCAST')
('Received from: ', ('192.168.2.197', 9000))
After data
('Received message: ', 'DISCOVER')
('Received from: ', ('192.168.2.197', 9000))
After data
('Received message: ', 'SET ON')
('Received from: ', ('192.168.2.197', 9000))
After data

```

Fig. 12. Proceso de descubrimiento y configuración en un dispositivo doméstico.

2.2.1 Programación en Arduino

Dados los diferentes entornos de desarrollo en las tarjetas Arduino, se procede a describir por separado la programación realizada en cada una de las tarjetas.

2.2.1.1 Arduino UNO

Se implementa el IDE de Arduino para realizar la programación. Para ello se hace uso de un WiFi shield que opera con librerías WiFi para su programación. Esta librería ofrece las herramientas necesarias para obtener la conexión a una red WiFi disponible, así mismo, permite enviar y recibir mensajes a través de un puerto UDP configurado. Permite además verificar el contenido de los mensajes y ejecutar acciones si el contenido corresponde a un comando previamente especificado.

Además de lo anterior, el IDE permite controlar el encendido de leds de apoyo para identificar visualmente cuando el atributo estado del dispositivo es modificado.

2.2.1.2 Arduino YUN

Se implementa Python para realizar la programación. Para ello se hace uso de los módulos de Python que permiten utilizar las funcionalidades de la red, es decir, conectarse a una red WiFi, enviar y recibir mensajes a través de un puerto UDP configurado. Permite además verificar el contenido de los mensajes y ejecutar acciones si el contenido corresponde a un comando previamente especificado.

El encendido de los leds de la tarjeta se realiza a través de la programación en el IDE de Arduino, para ello se hace una programación con la librería de la consola, lo cual permite controlar los leds a través de la conexión por Telnet. Por ello, el encendido y apagado del led de la tarjeta a través de Python se realiza haciendo uso del protocolo Telnet. En la Figura 13 se muestra la programación realizada en el IDE de Arduino para el control de los leds, en el cual se inicializa la variable ledPin en el pin número 13, el cual corresponde a un led.

```

void setup() {
  //initialize bridge and console
  Bridge.begin();
  Console.begin();
  while (!Console);
  Console.println("type 1 or 0 to turn pin 13 on or off");
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // see if there's incoming serial data:
  if (Console.available() > 0) {
    // read the oldest byte in the serial buffer:
    incomingByte = Console.read();
    // if it's a capital H (ASCII 72), turn on the LED:
    if (incomingByte == '1') {
      digitalWrite(ledPin, HIGH);
    }
    // if it's an L (ASCII 76) turn off the LED:
    if (incomingByte == '0') {
      digitalWrite(ledPin, LOW);
    }
  }
  delay(100);
}

```

Fig. 13. Programación en el IDE de Arduino para el control del led.

2.3 Archivos de programación

El archivo de programación en el Home Gateway es desarrollado con Java. Corresponde a un paquete llamado **UNDomoDiscovery**. Dicho paquete posee varios archivos, cuya funcionalidad es descrita a continuación:

- a. **UNDomoDiscovery.java**: Es el archivo principal. Envía los mensajes tipo broadcast y luego monitorea el puerto 9000 para el análisis de la recepción de los mensajes. De acuerdo con el tipo de mensaje ejecuta acciones de almacenamiento, configuración de equipos o actualización de direcciones IP.
- b. **Bcast.java**: es una clase. Realiza el envío de paquetes de broadcast a todas las direcciones de broadcast a las cuales pertenece el Home Gateway.
- c. **SearchDB.java**: Es una clase. Define la búsqueda de la dirección IP del dispositivo en las bases de datos para verificar su existencia. Define además la fecha de creación del perfil del dispositivo. Analiza el mensaje del dispositivo resultante del proceso del descubrimiento. Realiza la actualización de las direcciones IP de los dispositivos para asegurar que aún se encuentran activas en el sistema.
- d. **StoreinDB.java**: Es una clase. Permite el almacenamiento de la información en las bases de datos. Almacena la información general y la información respectiva de las características de los dispositivos, tales como bloqueo, nivel y color. Permite crear el perfil de los dispositivos.
- e. **ConfMesg.java**: Es una clase. Permite analizar y verificar el mensaje de configuración proveniente del Home Server. Posee las funciones de verificación de

ID, Código, Valor y fecha. Además de otras funciones relevantes para la efectividad del proceso de validación del mensaje.

- f. **Execute.java:** Es una clase. Permite ejecutar el envío del mensaje de configuración hacia el dispositivo especificado. Además, verifica que el proceso se haya llevado a cabo con éxito, para lo cual almacena en las bases de datos la nueva configuración y actualiza el perfil de los dispositivos.
- g. **EditFile.java:** Es una clase. Permite realizar las actualizaciones en el perfil de los dispositivos de acuerdo con el atributo modificado.

2.3 Iniciando el sistema UNDomDiscovery

La operación completa del sistema UNDomDiscovery se da de la siguiente manera:

Arduino UNO: Se conecta el Arduino UNO al PC de configuración y se inicia el IDE de Arduino, preferiblemente la versión más reciente, y se carga el archivo que represente al dispositivo que se desea emular. Se debe tener en cuenta que en el menú herramientas se tenga seleccionado “Arduino UNO” y el puerto correcto en el cual se encuentra conectado. Una vez cargado, se procede a observar las salidas en el monitor serie de Arduino para monitorear las entradas y salidas detectadas. En la tabla 12 se presenta los nombres de los archivos correspondientes para cada dispositivo que se desea simular.

Arduino YUN: Se conecta el Arduino YUN al PC de configuración y se inicia la conexión a través del protocolo SSH. Se obtiene la dirección IP de la tarjeta en la información de puertos del menú herramientas del IDE de Arduino. Los perfiles de los dispositivos se encuentran todos en una carpeta llamada **devices**. Por lo tanto, se accede a la carpeta así:

```
cd devices/
ls
```

El comando **ls** permite observar los 10 archivos presentes en la carpeta, los cuales representan los 10 dispositivos simulados por las tarjetas Arduino. En la Tabla 12 se presenta una relación de los dispositivos con el nombre de los archivos que se presentan.

Tabla 12. Dispositivos y archivos que lo contiene		
Dispositivo	Arduino UNO	Arduino YUN
On/Off Switch	Switch.ino	Switch.py
Level Control Switch	Lcswitch.ino	Lcswitch.py
Door Lock	Door.ino	Door.py
Simple Sensor	Sensor.ino	Sensor.py
On/Off Light	Onflight.ino	Onflight.py
Dimmable Light	Dimlight.ino	Dimlight.py
Color Dimmable Light	Coldim.ino	Coldim.py
On/Off Light Switch	Lswitch.ino	Lswitch.py
Dimmer Switch	Dimswitch.ino	Dimswitch.py
Color Dimmable Switch	Colorswitch.ino	Colorswitch.py

Para ejecutar la simulación de los dispositivos se escribe en consola:

python <nombredearchivo>

Después de lo anterior, notamos que la tarjeta se encuentra monitoreando el puerto 8888 para responder activamente al descubrimiento y a las solicitudes del Home Gateway.

ANEXO A

1. Código para la verificación del Firmware de la WiFi Shield.

```
#include <SPI.h>
#include <WiFi.h>

void setup() {
  //Initialize serial and wait for port to open:
  Serial.begin(9600);

  // check for the presence of the shield:
  if (WiFi.status() == WL_NO_SHIELD) {
    Serial.println("WiFi shield not present");
    // don't continue:
    while(true);
  }

  Serial.print("Firmware version: ");
  Serial.println(WiFi.firmwareVersion());
}

void loop() {}
```

2. Código para el escaneo de redes WiFi disponibles.

```
#include <SPI.h>
#include <WiFi.h>

void setup() {
  // initialize serial and wait for the port to open:
  Serial.begin(9600);
  while(!Serial) ;

  // attempt to connect using WEP encryption:
  Serial.println("Initializing Wifi...");
  printMacAddress();

  // scan for existing networks:
  Serial.println("Scanning available networks...");
  listNetworks();
}

void loop() {
  delay(10000);
  // scan for existing networks:
  Serial.println("Scanning available networks...");
  listNetworks();
}

void printMacAddress() {
  // the MAC address of your Wifi shield
```

```

byte mac[6];

// print your MAC address:
WiFi.macAddress(mac);
Serial.print("MAC: ");
Serial.print(mac[5],HEX);
Serial.print(":");
Serial.print(mac[4],HEX);
Serial.print(":");
Serial.print(mac[3],HEX);
Serial.print(":");
Serial.print(mac[2],HEX);
Serial.print(":");
Serial.print(mac[1],HEX);
Serial.print(":");
Serial.println(mac[0],HEX);
}

void listNetworks() {
  // scan for nearby networks:
  Serial.println("** Scan Networks **");
  byte numSsid = WiFi.scanNetworks();

  // print the list of networks seen:
  Serial.print("number of available networks:");
  Serial.println(numSsid);

  // print the network number and name for each network found:
  for (int thisNet = 0; thisNet<numSsid; thisNet++) {
    Serial.print(thisNet);
    Serial.print(" ");
    Serial.print(WiFi.SSID(thisNet));
    Serial.print("\tSignal: ");
    Serial.print(WiFi.RSSI(thisNet));
    Serial.print(" dBm");
    Serial.print("\tEncryption: ");
    Serial.println(WiFi.encryptionType(thisNet));
  }
}

```

3. Código para la conectividad a una red WPA2 segura.

```
#include <WiFi.h>
#include <SPI.h>

char ssid[] = "XXXXXX"; // your network SSID (name)
char pass[] = "*****"; // your network password
int status = WL_IDLE_STATUS; // the Wifi radio's status

void setup() {
  // initialize serial:
  Serial.begin(9600);

  // attempt to connect using WPA2 encryption:
  Serial.println("Attempting to connect to WPA network...");
  status = WiFi.begin(ssid, pass);
  Serial.println(status);

  // if you're not connected, stop here:
  if ( status != WL_CONNECTED) {
    Serial.println("Couldn't get a wifi connection");
    while(true);
  }
  // if you are connected, print out info about the connection:
  else {
    Serial.println("Connected to network");
  }
  Serial.println(status);
}

void loop() {
  // do nothing
}
```

ANEXO B

Se describen los perfiles de los dispositivos de acuerdo al tipo.

1. ON/OFF DEVICE

Característica de encendido/apagado

```
{
  "ID" : "ON01", // Id del equipo
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "On/Off Switch", //Equipo especificado y soportado por Zigbee
  "NAME" : "Switch Room1", // Nombre predeterminado y editado por el usuario
  "SCENE" : "Master Bedroom", // Escenas especificadas por Zigbee
  "ON_OFF_STATE" : "0",
  "LAST_DATE" : "02/03/2015 10:58:25"
}
```

2. LEVEL CONTROL SWITCH

Característica de encendido/apagado y de nivel

```
{
  "ID" : "LEVEL02",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Level Control Switch",
  "NAME" : "Level switch Room2",
  "SCENE" : "Closet",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25",
  "LEVEL" :
  {
    "MAX_VALUE" : "1",
    "MIN_VALUE" : "0",
    "VALUE" : "0.5"
  }
}
```

3. DOOR LOCK

Característica de encendido/apagado y de bloqueo

```
{
  "ID" : "DOOR03",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Door Lock",
  "NAME" : "Principal Door",
  "SCENE" : "Entry",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25",
  "LOCK" :
  {
    "LOCK_STATE" : "1"
  }
}
```

4. SIMPLE SENSOR

Característica de encendido/apagado

```
{
  "ID" : "SEN04",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Simple Sensor",
  "NAME" : "Window",
  "SCENE" : "Office",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25"
}
```

5. ON/OFF LIGHT

Característica de encendido/apagado

```
{
  "ID" : "ON05",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "On/Off Light",
  "NAME" : "Bath Light",
  "SCENE" : "Bathroom",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25"
}
```

6. DIMMABLE LIGHT

Característica de encendido/apagado y nivel

```
{
  "ID" : "LI06",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Dimmable Light",
  "NAME" : "Dimm Dining",
  "SCENE" : "Dining Room",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25",
  "LEVEL" :
  {
    "MAX_VALUE" : "1",
    "MIN_VALUE" : "0",
    "VALUE" : "0.5"
  }
}
```

7. COLOR DIMMABLE LIGHT

Característica de encendido/apagado, nivel y color

```
{
  "ID" : "LI07",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Color Dimmable Light",
  "NAME" : "Color Light",
  "SCENE" : "Theater",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25",
  "LEVEL" :
  {
    "MAX_VALUE" : "1",
    "MIN_VALUE" : "0",
    "VALUE" : "0.5"
  },
  "COLOR" :
  {
    "LUMINANCE" : "100",
    "HUE" : "50",
    "SATURATION" : "80"
  }
}
```

8. ON/OFF LIGHT SWITCH

Característica de encendido/apagado

```
{
  "ID" : "ONF08",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "On/Off Light Switch",
  "NAME" : "Switch Light",
  "SCENE" : "Elevator",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25"
}
```

9. DIMMER SWITCH

Característica de encendido/apagado y nivel

```
{
  "ID" : "SW09",
  "NETWORK_TYPE" : "Wifi",
  "DEVICE_TYPE" : "Dimmer Switch",
  "NAME" : "Dimm Cellar",
  "SCENE" : "Cellar",
  "ON_OFF_STATE" : "1",
  "LAST_DATE" : "02/03/2015 10:58:25",
  "LEVEL" :
  {
```

```
"MAX_VALUE" : "1",  
"MIN_VALUE" : "0",  
"VALUE" : "0.5"  
}
```

10. COLOR DIMMER SWITCH

Característica de encendido/apagado, nivel y color

```
{  
  "ID" : "SW10",  
  "NETWORK_TYPE" : "Wifi",  
  "DEVICE_TYPE" : "Color Dimmer Switch",  
  "NAME" : "Color Switch",  
  "SCENE" : "Game Room",  
  "ON_OFF_STATE" : "1",  
  "LAST_DATE" : "02/03/2015 10:58:25",  
  "LEVEL" :  
  {  
    "MAX_VALUE" : "1",  
    "MIN_VALUE" : "0",  
    "VALUE" : "0.5"  
  },  
  "COLOR" :  
  {  
    "LUMINANCE" : "100",  
    "HUE" : "50",  
    "SATURATION" : "80"  
  }  
}
```


ANEXO C

Descripción detallada de las características de los dispositivos y los comandos que soportan.

1. Encendido/apagado

On/Off cluster

SENDING ON OFF AND TOGGLE COMMANDS TO DEVICES TO CHANGE THEIR STATE

```
{
    "CL_ID" : "06",
    "CL_NAME" : "On/Off",
    "CL_TYPE" : "General",
    "CL_HELP" : "Change the current state of the device",
    "CL_COMMANDS" :
        [
            {"CMD00" : "SET NAME",
            "CMD01" : "SET ON",
            "CMD02" : "SET OFF",
            "CMD03" : "SET TOGGLE"
            }
        ]
}
```

2. Control de Nivel

Level Control CLuster

CONTROL THE LEVEL OF A CHARACTERISTIC.

```
{
    "CL_ID" : "08",
    "CL_NAME" : "Level Control",
    "CL_TYPE" : "General",
    "CL_HELP" : "Set level state of the switch",
    "CL_COMMANDS" :
        [
            {
            "CMD04" : "SET VALUE",
            }
        ]
}
```

3. Configurador de escena de los dispositivos.

Scene selector Cluster

SETTING UP AND SELECT SCENES ON DEVICES

```
{
    "CL_ID" : "05",
    "CL_NAME" : "Scene Selector",
    "CL_TYPE" : "General",
    "CL_HELP" : "Set scene",
    "CL_COMMANDS" :
        [
            {"CMD05" : "SET SCENE",
            }
        ]
}
```

4. Bloqueo de un atributo, esencialmente de puertas.

Door Lock Cluster

```
LOCK OR UNLOCK THE SPECIFIED DOOR
{
    "CL_ID" : "101",
    "CL_NAME" : "Door Lock",
    "CL_TYPE" : "Closures",
    "CL_HELP" : "Change Door State",
    "CL_COMMANDS" :
        [
            {"CMD06" : "SET LOCK",
             "CMD07" : "SET UNLOCK"
            }
        ]
}
```

5. Características binarias de entrada y salida.

Binary Input Cluster

```
SUPPORTING A BINARY INPUT ON/OFF
{
    "CL_ID" : "F",
    "CL_NAME" : "Binary Input",
    "CL_TYPE" : "General",
    "CL_HELP" : "Set states",
    "CL_COMMANDS" :
        [
            {"CMD01" : "SET ON",
             "CMD02" : "SET OFF"
            }
        ]
}
```

6. Configuración de color.

Color Control Cluster

```
LUMINANCE, HUE AND SATURATION MAY BE CONTROLLED
{
    "CL_ID" : "300",
    "CL_NAME" : "Color Dimmable Light",
    "CL_TYPE" : "Lighting",
    "CL_HELP" : "Set color",
    "CL_COMMANDS" :
        [
            {"CMD08" : "SET LUMINANCE",
             "CMD09" : "SET HUE",
             "CMD10" : "SET SATURATION"
            }
        ]
}
```