Universidad Zaragoza
1542

Trabajo Fin de Grado

Nodo IoT con cámara web para aplicaciones domóticas

Autor

Heimdal Lahoz Chuliá

Director

Bonifacio Martín del Brío

Escuela de Ingeniería y Arquitectura

2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. Heimdal Lahoz Chuliá,

con nº de DNI 35604952D en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Nodo IoT con cámara web para aplicaciones domóticas,

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, a 18 de Sptiembre de 2018

Fdo: Heimdal Lahoz Chuliá

RESUMEN

En este trabajo se va a diseñar e implementar un nodo IoT en Arduino Yún, que permita a un usuario acceder, mediante PC o teléfono móvil, a cámaras y sensores de bajo coste instalados en su vivienda, utilizando para ello las posibilidades de Arduino Yún, el cual incluye muchas facilidades para su conexión a internet y un microprocesador con sistema operativo Linux. Se trata de diseñar un prototipo que en el futuro pueda emplearse como parte de un sistema de vigilancia del hogar.

En primer lugar, se convertirá Arduino Yun en un servidor web en el que se alojarán las aplicaciones implementadas. Seguidamente se abordan cuatro desarrollos, basados en el acceso al nodo IoT desde internet (desde computador o teléfono móvil) para llevar a cabo las tareas de monitorización indicadas (cámaras, sensores).

El primer desarrollo consiste en la conexión al nodo de diferentes tipos de cámaras "low cost", programando la *captura periódica y transferencia automática* de imágenes a una cuenta de Dropbox, permitiendo elegir al usuario tanto el periodo como la cámara que se utiliza para las capturas. El segundo desarrollo permite que el usuario pueda solicitar *a demanda* la realización de una única captura de imagen, que será enviada como archivo adjunto a su email y además la posibilidad de escribir un Tweet y publicarlo en la red social si se desea. El tercero se basa en la *toma de datos* mediante un sensor de temperatura conectado a Arduino, permitiendo observar la gráfica (temperatura-tiempo) en tiempo real y posteriormente guardándolos en un registro histórico para poder acceder a ellos en un futuro. Por último, la cuarta aplicación da la posibilidad de emitir vídeo en *streaming cuando se detecta presencia mediante un sensor PIR*; una vez se detecta movimiento, se realizan cuatro capturas consecutivas y se envían al email del usuario automáticamente.

Para conseguir estos objetivos, se ha realizado la programación mediante el software Arduino y la aplicación PuTTY, que permite la ejecución por línea de comandos del programa en el procesador Linux, además de la sincronización con las diferentes redes sociales a las que tiene acceso el nodo IoT. A continuación, se lleva a cabo el desarrollo de las páginas web y la aplicación móvil para poder hacer las pruebas del acceso desde internet y comprobar cómo funciona el nodo IoT a través de la interfaz de usuario, además de poder incorporar la programación necesaria para la comunicación a través de internet (protocolo http).

Contenido

Índice de Figuras

Índice de Tablas

Anexos

Capítulo 1.

Introducción

1.1. Contexto

Antes de empezar a hablar en concreto de la “Internet of Things” (IoT), hay que tener en cuenta el impacto que internet ha tenido sobre la humanidad, siendo una de las creaciones más importantes y poderosas de la historia. Básicamente, su objetivo es la conexión de personas (y equipos) a distancia.

El punto de inflexión en el que cambia la concepción de la utilidad de internet a lo que en la actualidad se conoce como IoT es, según el Grupo de Cisco de soluciones empresariales basadas en internet, el momento en el que se conectaron a la red más cosas u objetos que personas. Este nuevo concepto se entiende como la primera gran revolución de internet, ya que deja de centrarse en la simple comunicación y abre una ventana de posibilidades para el desarrollo de aplicaciones en el área de automatización, uso de sensores y la comunicación entre máquinas, que mejorarán significativamente la manera de vivir, aprender, trabajar, etc. [1]

En la actualidad, el concepto de IoT se basa en crear una red global compuesta por redes de sensores diferentes y con distintos fines conectados a internet. Por ejemplo, los automóviles que disponen de múltiples redes para controlar el funcionamiento del motor, las medidas de seguridad, los sistemas de comunicación…; a medida que evolucione el IoT se conseguirá que dichas redes estén conectadas de manera que el sistema pueda tomar decisiones a tiempo con precisión, velocidad y consistencia, dotando de autonomía al coche.

Otro campo en constante desarrollo es el de la medicina, en el que, en la actualidad, podemos encontrar dispositivos conectados a internet que los pacientes pueden ingerir y que permiten a los médicos diagnosticar y determinar las causas de distintas enfermedades; un ejemplo es el biosensor de glucosa.

La evolución del IoT está directamente relacionada con el aumento en gran medida de la manera de recoger datos a partir de los sensores conectados a la red, y dando la capacidad de comunicarlos pudiendo acceder a ellos desde cualquier lugar.

En este trabajo se ha utilizado Arduino Yún (Figura 1). Se trata de un modelo de placa Arduino que, a diferencia del resto de modelos que ofrece la marca, combina un microcontrolador de Atmel con un microprocesador con

sistema operativo Linux, dándole capacidad de conexión a internet mediante Ethernet o WiFi, o vía bluetooth. Arduino Yun será el nodo IoT al que se conectarán tanto las cámaras web como los sensores de este trabajo. Gracias a sus facilidades para conexión a internet, dará la posibilidad de acceder de manera remota a los sensores para recoger datos y también de realizar capturas de las imágenes suministradas por las cámaras.

Otra de las funcionalidades que aporta la integración de un procesador con Linux es poder añadir una memoria SD externa que dote de mayor capacidad de memoria, para no depender sólo de la interna del dispositivo Arduino (muy limitada). Esto permite la configuración de Arduino como servidor web y da la posibilidad de guardar tanto los datos recogidos como las capturas realizadas para acceder posteriormente a ellas y poder analizarlas. Además, el servidor web configurado puede ejecutar tanto las páginas web como los archivos Python, JavaScript, etc., ampliando así en gran medida el número de aplicaciones que permitirá el uso del nodo sensor.

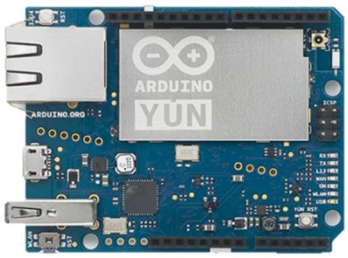

Figura 1. Arduino Yún

1.2. Objetivos y alcance

En este trabajo se va a diseñar e implementar un nodo IoT con Arduino Yún, módulo especialmente diseñado para conexión a internet, que permita a un usuario acceder mediante PC o teléfono móvil, a cámaras y sensores de bajo coste instalados en su vivienda. Se trata de diseñar un prototipo que en el futuro pueda emplearse como parte de un sistema de vigilancia del hogar.

Para ello se realizará la implementación de un nodo IoT con la placa Arduino Yún, utilizando esta plataforma como puerta de acceso a los dispositivos conectados en ella: cámaras de bajo coste (una cámara “estilo Arduino” y una webcam USB estándar) y sensores que puedan ser utilizados en aplicaciones de seguridad en el hogar (temperatura, presencia…). Para el desarrollo de la programación se han utilizado diferentes softwares dependiendo de si el código tiene que ser ejecutado por el microcontrolador o por el procesador. En el caso del microcontrolador, se programará en IDE Arduino, plataforma de desarrollo software de código libre de Arduino, que utiliza lenguaje Arduino AVR o C++, y en el otro caso se utilizará PuTTY, que permite ejecutar código Linux por la línea de comandos en el procesador.

Además, para configurar la memoria interna del servidor web implementado en Arduino, se tiene que acceder mediante un cliente SSH como WinSCP. Para comprobar el correcto funcionamiento del nodo, se han desarrollado páginas web y una aplicación para móviles Android. La descripción detallada tanto del software como de las aplicaciones programadas se refleja en capítulos posteriores.

El primer objetivo de este proyecto es conseguir que a través del módulo Arduino Yún, por conexión WiFi, Ethernet o Bluetooth, el usuario pueda establecer los parámetros necesarios para llevar a cabo satisfactoriamente cualquiera de las aplicaciones que decida ejecutar, y asegurar el correcto acceso a las redes sociales vinculadas a las credenciales de acceso del usuario. La idea es que el usuario pueda visualizar imágenes de su casa, a demanda o cuando se dispara una alarma, desde una aplicación web o móvil, o consultando una carpeta Dropbox, o recibiendo un email o incluso un Tweet.

El nodo deberá guardar la configuración deseada desde el instante en el que arranque una aplicación hasta que el usuario decida detenerla, para así poder funcionar de forma autónoma, sin necesidad de consultar los parámetros como, por ejemplo, el periodo.

El segundo objetivo es que el nodo sensor sepa distinguir mediante el código recibido por cualquiera de los métodos de comunicación, qué dispositivo tiene que usar, en caso de elegir una u otra cámara web, ya que necesitan diferente programación porque tienen diferente manera de procesar la imagen, o en el caso de tener que tomar datos mediante uno de los sensores conectados.

El tercer objetivo es que el nodo IoT pueda acceder a los datos necesarios alojados en el servidor web de Arduino, tanto para poder llevar a cabo las tareas necesarias con las capturas de imagen como para visualizar desde una página web gráficas de evolución de los datos de los sensores conectados. Además, tiene que guardar el instante exacto en el que se realiza cada captura o registro de datos para ponerle una etiqueta.

1.3. Organización de la memoria

El resto del trabajo se ha dividido en los siguientes capítulos:

- **Capítulo 2:** En este capítulo se hace una introducción teórica de los dos conceptos básicos con los que se plantean las bases del trabajo, “Internet of Things” y la plataforma Arduino Yún. A continuación, se explican más detalladamente el microcontrolador y el microprocesador que se combinan en la placa, para conocer cómo funcionan internamente, y las comunicaciones que permite y que serán necesarias para el desarrollo del trabajo. Por último, se abordan teóricamente los

periféricos que se conectarán al nodo IoT y se detalla dicha conexión mediante esquemas hardware.

- **Capítulo 3:** Se explica detalladamente el software utilizado para la programación de las aplicaciones, tanto del lado del microcontrolador como del procesador con sistema operativo Linux, haciendo una breve introducción del funcionamiento del programa y del lenguaje utilizado, y una presentación de la interfaz de cada uno de ellos. A continuación, se explican teóricamente los diferentes lenguajes de programación adicionales que se han usado en el trabajo, para ejecutar en Linux y aportar funcionalidades que no son capaces de llevarse a cabo por el microcontrolador, como Python, HTML o JavaScript para el desarrollo web.

- **Capítulo 4:** En este se explican las aplicaciones implementadas en el nodo IoT, haciendo una descripción detallada del funcionamiento de cada una de ellas y de los pasos que hay que seguir para la correcta ejecución, tanto por parte del usuario como del módulo Arduino Yún. También, se mostrará el método con el que se ha programado cada una de ellas mediante diagramas de flujo, indicando el proceso que se lleva a cabo en cada ejecución de dicha aplicación y las limitaciones de cada programa como, por ejemplo, la necesaria elección de un periodo o de una de las cámaras web para poder continuar.

- **Capítulo 5:** Capítulo donde se muestran los prototipos de página web y aplicación Android desarrollados, para poder comprobar el funcionamiento y analizar los resultados obtenidos de los experimentos realizados a través del acceso al nodo mediante conexión WiFi o Bluetooth.

- **Capítulo 6:** En él se desarrollan las conclusiones obtenidas de la realización del trabajo, las dificultades encontradas a la hora de abordar la implementación práctica del proyecto y las posibles ampliaciones o mejoras de este en un futuro.

1.4. Cronograma

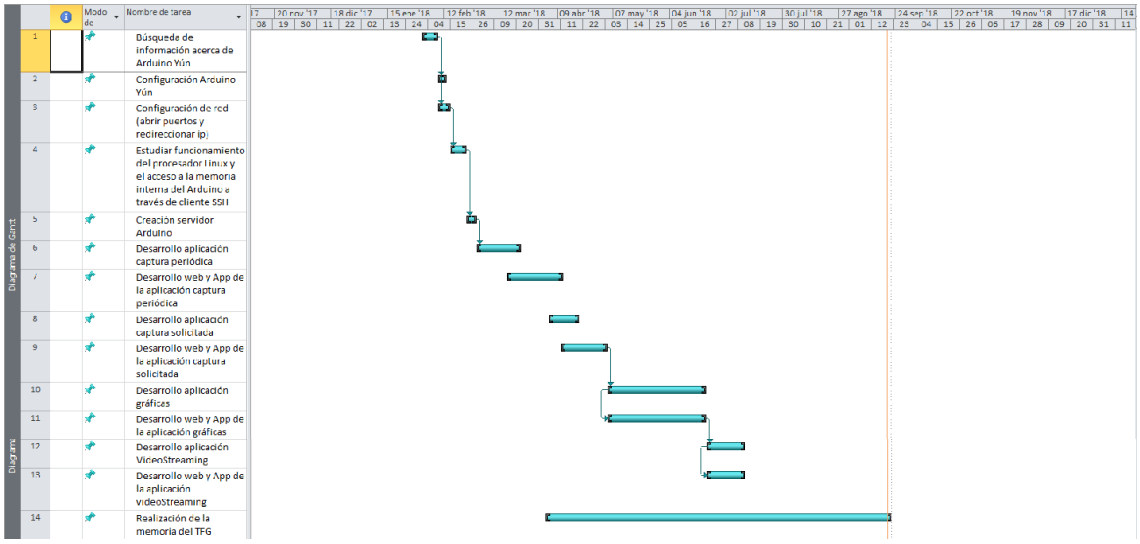

Figura 2. Diagrama de Gantt del proyecto

Como se puede observar en la Figura 3, las primeras tareas están dirigidas al estudio desde el punto de vista teórico de la plataforma Arduino Yún y su funcionamiento interno, a la configuración inicial y la conexión a internet (WiFi o Ethernet), a la ampliación de memoria utilizando una tarjeta SD, y conocer las posibilidades que da el uso combinado del microprocesador y el procesador Linux en el módulo Arduino.

Las tareas en las que más tiempo se ha empleado son las relacionadas con el desarrollo de las aplicaciones prácticas en el nodo IoT, debido a la poca información disponible y la necesidad de avanzar en muchas ocasiones a base de prueba y error. Han requerido mayor dedicación las tareas basadas en las aplicaciones captura solicitada y gráficas, esto se debe a la complejidad que tiene la implementación de la interfaz de usuario en ambas aplicaciones. Por ejemplo, la representación gráfica en tiempo real o la ampliación de la imagen en una ventana emergente y la posibilidad de publicar un Tweet, con necesidad de conocimientos avanzados de JavaScript y el uso de AJAX.

Capítulo 2.

Hardware y comunicaciones

2.1. Definición de “Internet of things”

El concepto está basado en la idea de conectar dispositivos físicos inteligentes (sensor, webcam, smartphone, etc), componentes básicos del “Internet of things”, a un sistema integrado principal que a su vez también está conectado a internet. Esta conexión nos da la posibilidad de acceder a datos y controlar el mundo físico desde la distancia.

En el concepto “Internet of things” el dispositivo inteligente deja de representarse a sí mismo digitalmente para convertirse en algo más grande, ahora forma parte de una red de dispositivos y bases de datos que actúan al unísono, conocido como “ambiente inteligente”. En esta red global, la cual podemos entender como despliegue generalizado de objetos inteligentes, se le asigna una identificación electrónica inteligente de bajo coste a cada dispositivo, con la posibilidad de ser reconocida por otros dispositivos a distancia, como si se tratara de un código de barras.

Gracias al acceso a la captura instantánea de datos por parte de cualquier dispositivo unido a la red y a la recopilación de información alojada en internet anteriormente, se crea una red con la cual convertimos el mundo físico en simples datos, los cuales se pueden analizar y llevar a cabo acciones con un propósito específico, haciendo así que las tareas cotidianas realizadas en el mundo físico sean más eficientes, ahorrando tiempo, coste y posiblemente errores en el proceso. [1] [2] [3]

2.2. Diagrama hardware del sistema

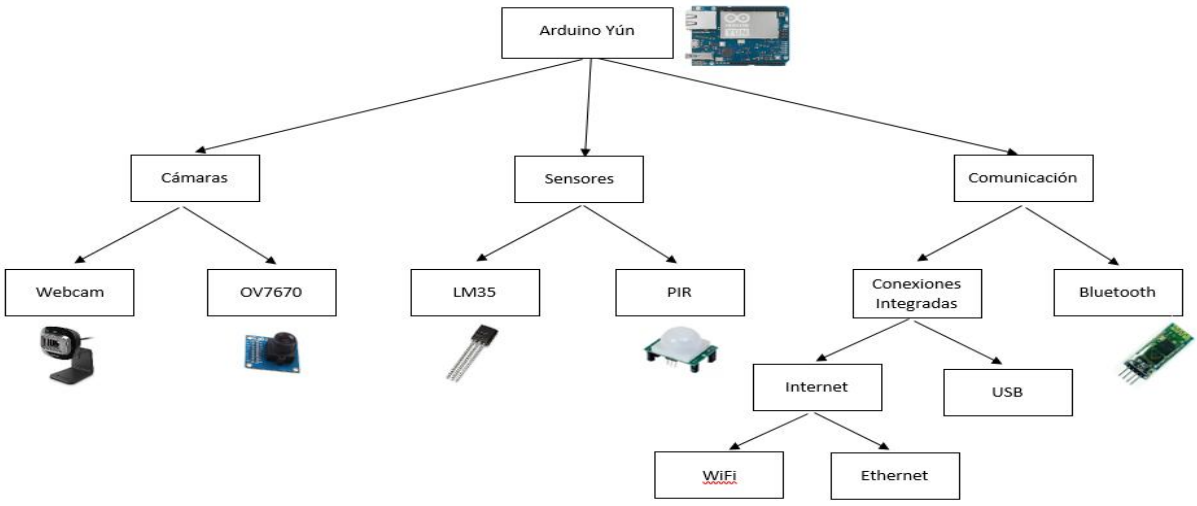

Figura 3. Diagrama de bloques del hardware del sistema

La Figura 3 muestra los dispositivos que componen el sistema y sus conexiones. El nodo IoT estará programado en la placa Arduino Yún, por eso se muestra como el centro desde el que se controlará todo. Por un lado, se conectarán 2 cámaras de bajo consumo (webcam USB estándar y OV7670 de bajo coste), además para controlar la estancia en la que se encuentre el módulo Arduino, se dispone de un sensor de temperatura LM35 y un sensor de proximidad PIR. La conexión del nodo IoT a internet se hará a través de WiFi o Ethernet, ambos integrados en la placa Arduino, y para la comunicación con el computador o el teléfono móvil se utilizará el puerto USB o el módulo Bluetooth Hc-06.

En los siguientes puntos del capítulo se realizará una explicación teórica de la placa Arduino, centro de control del nodo IoT, de los periféricos que se conectarán para supervisar el hogar, y los tipos de comunicaciones con las que el usuario puede conectar el módulo Arduino al ordenador o al teléfono móvil.

2.3. Plataforma Arduino

2.3.1. Introducción

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo (software), diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. Es una forma sencilla de llevar a cabo proyectos interactivos para usuarios con distinto nivel de conocimiento acerca de la electrónica. Es fácil de usar para novatos y a la vez lo suficientemente flexible para usuarios avanzados. Por esto se utiliza en todo tipo de ámbitos, desde usuarios que comienzan como hobby pasando por profesores que inician a sus alumnos en el campo de la programación y la robótica hasta ingenieros, diseñadores o científicos para desarrollar diferentes prototipos. También gracias a la posibilidad de utilizar el software en Macintosh OSX, Linux o Microsoft, en el cual muchos sistemas de microcontroladores están limitados. [4]

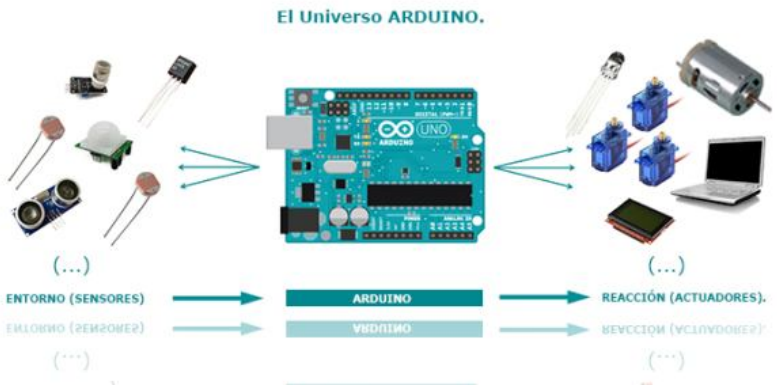

Figura 4. Periféricos, sensores y actuadores de Arduino

Arduino es utilizado por su facilidad de programación del microcontrolador incorporado en la placa dando la posibilidad de crear elementos autónomos interactuando tanto con el hardware como con el software a través de la placa, desde el cableado y el montaje de hardware hasta la captación de datos del mundo físico a partir de los sensores y posterior conversión de esta información en acciones que llevar a cabo por parte de los actuadores, Figura 4. Por ejemplo, una aplicación sencilla como encender un pequeño LED (actuador) cuando el usuario apriete un botón (sensor), u otras más avanzadas como conectar un motor (actuador) que baje una persiana cuando el sensor de luz detecte que el nivel lumínico está por debajo del rango deseado.

El hardware consiste generalmente en una placa formada por un microcontrolador Atmel AVR, puertos de comunicación y puerto de entrada/salida tanto analógica como digital. La principal ventaja es la gran cantidad de creaciones que se pueden llevar a cabo con una inversión económica mínima: el coste del dispositivo Arduino y los periféricos.

El software que nos proporciona Arduino es libre y consiste en un entorno de desarrollo en el que implementar el lenguaje de programación AVR C, el cual puede expandirse por los usuarios a través de librerías de C++, herramientas para transferir el firmware al microcontrolador y el *bootloader* ejecutado en la placa. Se caracteriza sobre todo por su sencillez y su fácil uso. [5]

2.3.2. Arduino Yún

Arduino Yún es el miembro de la familia que combina el poder del procesador Atheros9331, el cual lleva instalado Linino (distribución GNU/Linux basada en OpenWRT). El procesador está embebido en la PCB de la plataforma Arduino por lo que está comunicado con el microcontrolador de Atmel, esto permite mandar comandos del lado Arduino que serán procesados por Linino y viceversa. Permite la conexión entre procesador y microcontrolador gracias a la librería “Bridge” alojada en el software de Arduino (IDE), este añade funcionalidades del procesador aprovechando el poder computacional de Linux. [6]

Además, el modelo Yún permite conectarse a internet a través del WiFi integrado en el chip del procesador o mediante cable de Ethernet. Estas conexiones se pueden configurar a través del Yún Web Panel, funcionalidad integrada en el procesador con la que podremos acceder al sistema de forma remota a través de una web vinculada a la ip del propio Arduino (192.168.240.1), nos permite programarlo sin necesidad de una conexión física mediante el cable USB al PC para enviar el programa desde Arduino (IDE). [7]

Una de las pocas desventajas de este modelo es la no incorporación de regulador de tensión integrado en la PCB por lo tanto hay que llevar especial cuidado y es importante asegurarse de alimentar el módulo Arduino mediante el micro USB desde el PC o desde una fuente de alimentación regulada a 5V.

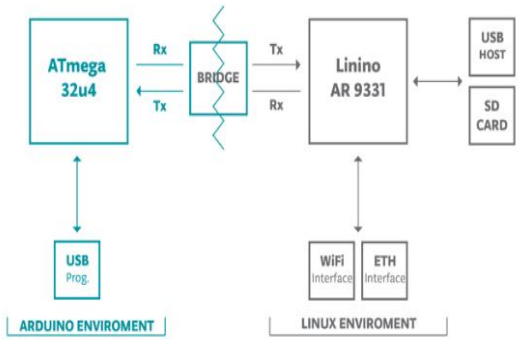

Figura 5. Esquema de conexión entre microcontrolador y procesador

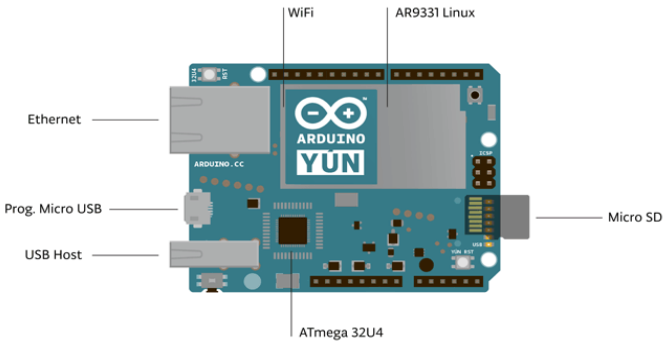

Figura 6a. Esquema de periféricos integrados

La placa cuenta con periféricos integrados que no incluyen otros modelos (Figuras 6ª, 6b, y 6c), y añaden funcionalidades importantes: [7]

- Host USB que permite conectar dispositivos que interactúen con la parte del procesador con sistema Linux.
- Adaptador microSD que tiene conexión física con la parte Linino, pero también permite acceder desde el microcontrolador mediante la librería “Bridge”.
- LED’s de estado del dispositivo
- 3 botones de reset, uno para el WiFi, otro para el microprocesador de Atmel y otro para el procesador Atheros AR9331.

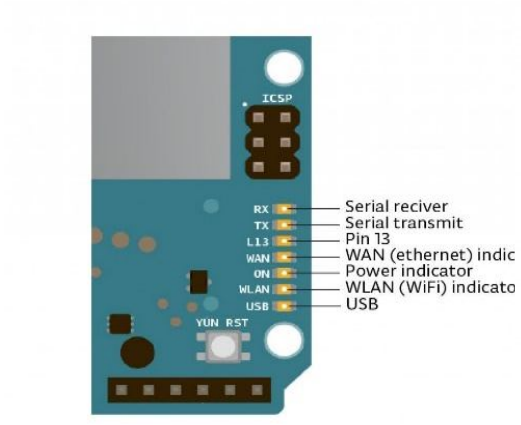

Figura 6b. LED's de estado

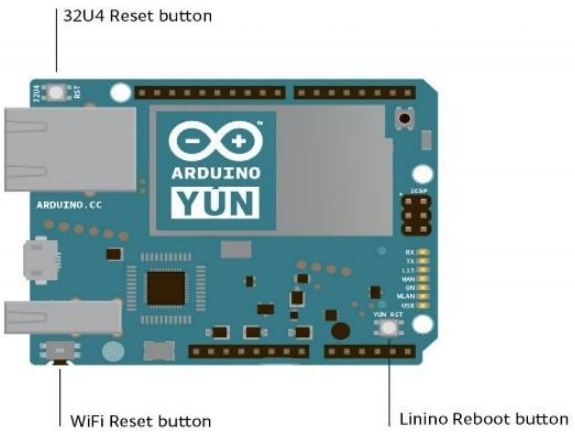

Figura 6c. Botones de reset

2.4. Microcontrolador Atmega32u4

2.4.1. Introducción

Los microcontroladores son circuitos integrados que incorporan todos los bloques funcionales de un Sistema Microprocesador en un único encapsulado. Secuencialmente interpretan ciertas instrucciones y generan señales digitales internas y/o externas que permita controlar un sistema electrónico.

El microcontrolador ATmega32u4 no requiere de oscilador externo porque tiene uno interno de 8MHz y requiere una tensión de 5 v para su alimentación. Este es un microcontrolador de bajo coste, miembro la familia de alto rendimiento de microcontroladores de 8 bits de Atmel, y como se observa en la Figura 7, el modelo elegido tiene 20 pines digitales y 7 analógicos programables como entradas o salidas. [8]

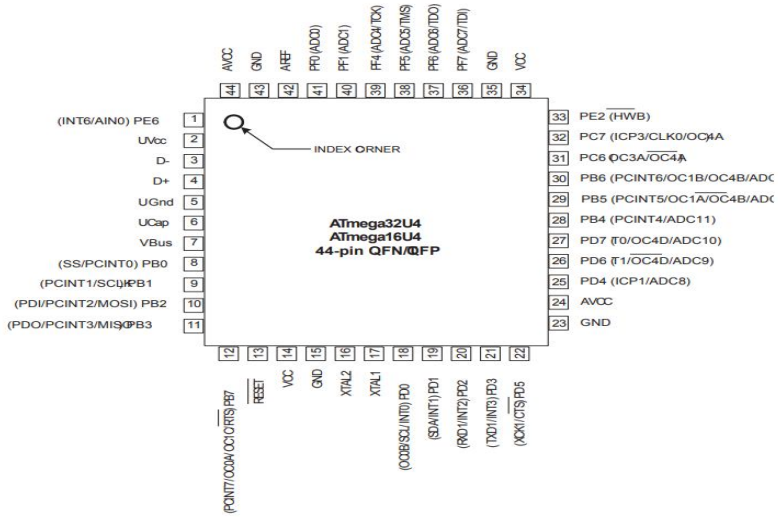

Figura 7. Esquema de pines del Atmega32u4

A continuación, en la Tabla 1 se muestran las especificaciones del microcontrolador

Operating Voltage	5V
Input Voltage	5V
Digital I/O Pins	20
PWM Output	7
Analog I/O Pins	12
DC Current per I/O Pin	40 mA on I/O Pins; 50 mA on 3,3 Pin
Flash Memory	32 KB (of which 4 KB used by bootloader)
SRAM	2.5 KB
EEPROM	1 KB
Clock Speed	16 MHz

Program Memory Type	Flash
Program Memory Size (KB)	32
CPU Speed (MIPS/DMIPS)	16
SRAM Bytes	2,560
Data EEPROM/HEF (bytes)	1024
Digital Communication Peripher...	1-UART, 2-SPI, 1-I2C
Capture/Compare/PWM Periphe...	2 Input Capture, 2 CCP, 12PWM
Timers	2 x 8-bit, 2 x 16-bit
Number of Comparators	1
Number of USB Modules	1, Full Speed
Temperature Range (C)	-40 to 85
Operating Voltage Range (V)	2.7 to 5.5
Pin Count	44

Tabla 1. Especificaciones ATmega32u4

2.4.2. Diagrama de bloques

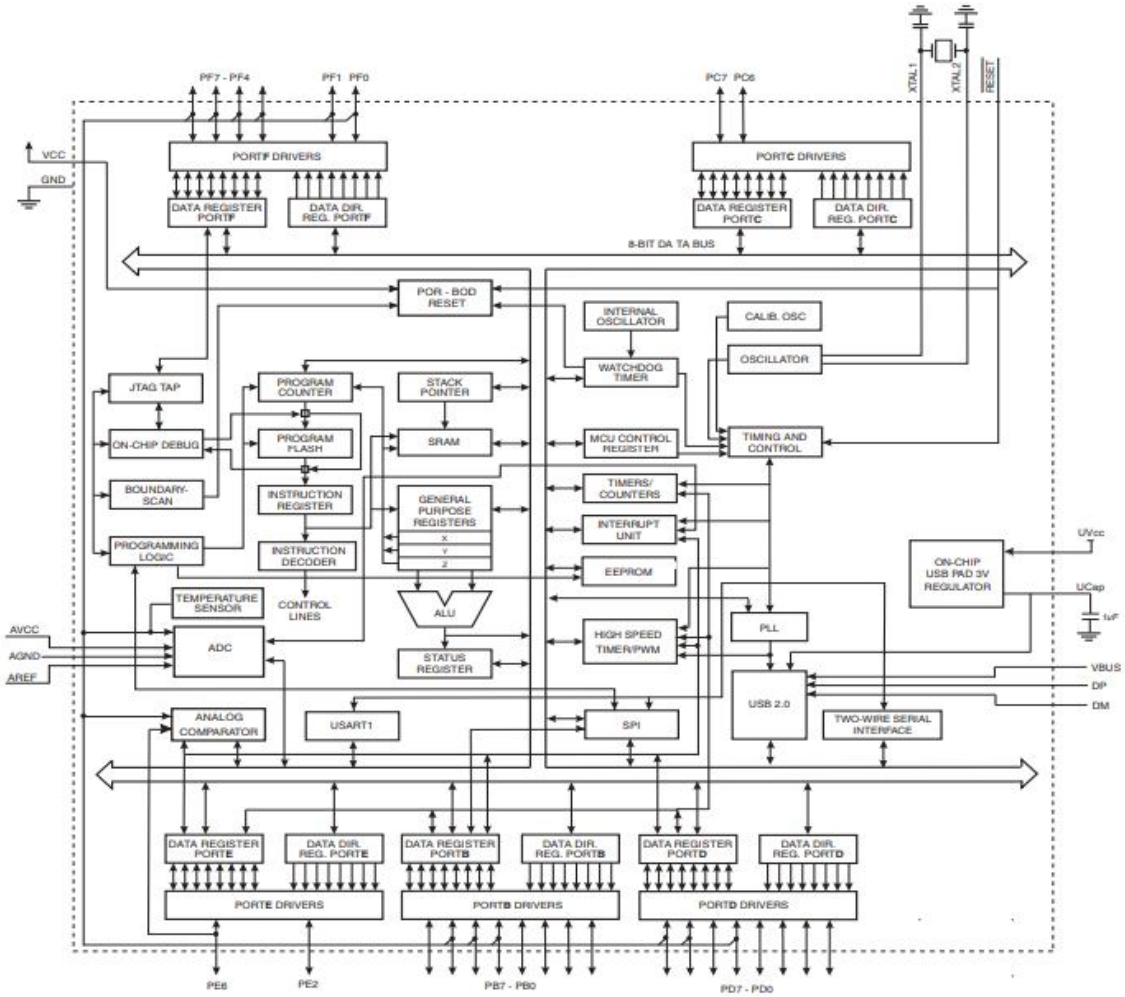

Figura 8a. Diagrama de bloques ATmega32u4

El núcleo AVR, como se muestra en la Figura 8a, combina los registros de instrucción con 32 registros de propósito general. Los 32 registros están directamente conectados a la Unidad Aritmética Lógica (ALU), permitiendo así que pueda acceder dos registros independientes en una instrucción ejecutada en un único ciclo de reloj. La arquitectura resultante es más eficiente, consiguiendo un rendimiento 10 veces más rápido que los microcontroladores convencionales.

Como muestra la Figura 8b, el AVR usa arquitectura Harvard, separa las memorias y los buses dependiendo de que sean dedicados al programa o a datos. Mientras una instrucción de está ejecutándose la siguiente ya está siendo prefijada por la memoria de programa, esto permite que las instrucciones sean ejecutadas en un único ciclo de reloj. [9]

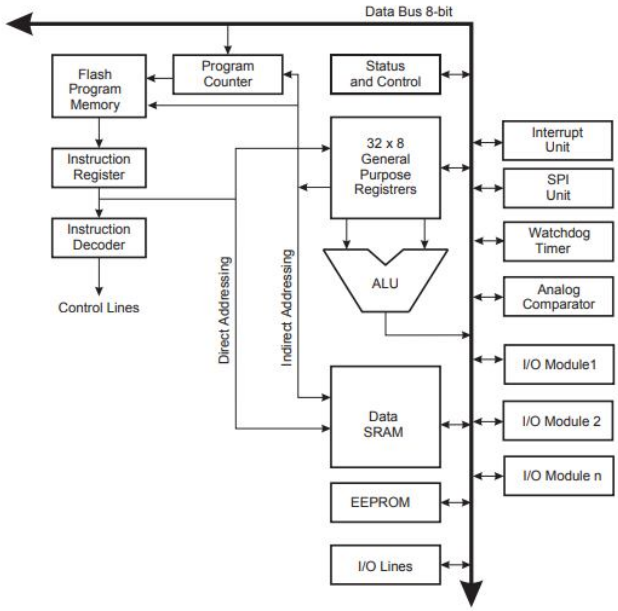

Figura 8b. Diagrama de la arquitectura AVR del microcontrolador

2.5. Microprocesador Atheros AR9331

El microprocesador es la parte encargada de llevar a cabo la ejecución de los programas, por eso podemos decir que es el cerebro de la máquina, procesa y ejecuta las instrucciones codificadas en números binarios, en primer lugar, lee las instrucciones, luego las decodifica, seguidamente lee los operandos, los ejecuta y por último muestra los resultados.

El Atheros AR9331 es un integrado con *WiFi System-on-Chip* (WiSoC) utilizado habitualmente en puntos de acceso o plataformas router. En este microprocesador incluye entre otros un procesador MIPS 24k, una conexión Ethernet, host USB para conectar dispositivos externos a la placa Arduino, interfaz para memorias Flash, UART y GPIO's usados para LED's de control o para la configuración de otros interfaces de propósito general (Figura 9a). [10]

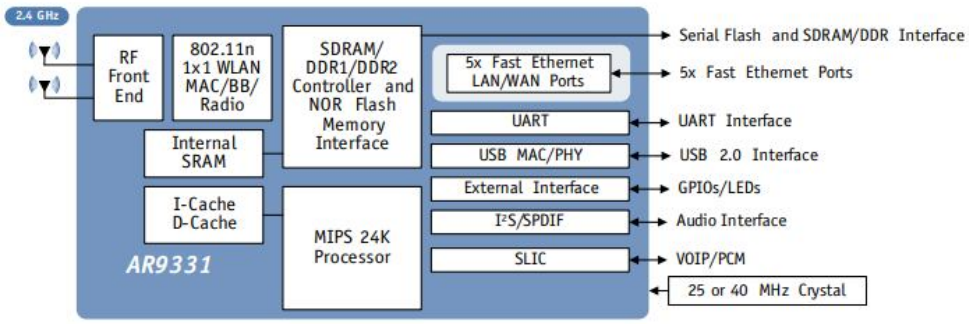

Figura 9a. Diagrama de bloques del sistema del microprocesador

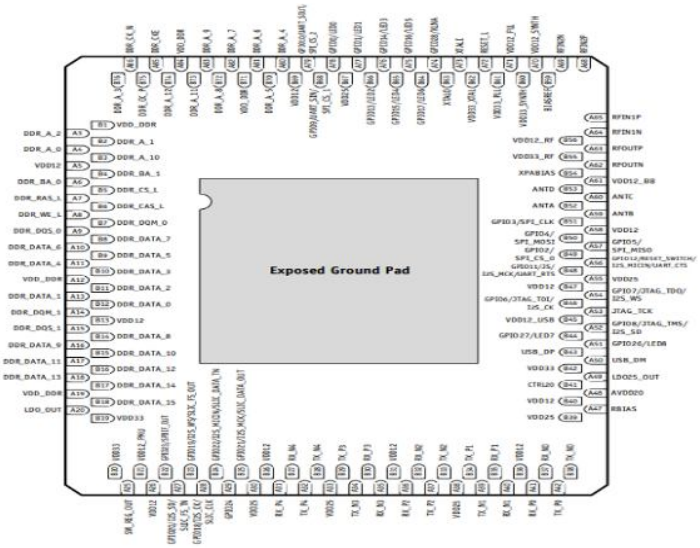

Figura 9b. Diagrama de pines

En la tabla 2 se muestran las características del microprocesador:

Microprocesador Linux	
Procesador	Atheros AR9331
Arquitectura	MIPS @400MHz
Voltaje Operativo	3.3V
Ethernet	IEEE 802.3 10/100Mbit/s
WiFi	IEEE 802.11b/g/n
USB Tipo-A	2.0 Host/Device
Lector MicroSD	Micro-SD only
RAM	64 MB DDR2
Memoria Flash	16 MB

Tabla 2. Características del Atheros AR9331

2.6. Comunicación serie/UART y USB

Los puertos y los buses son los encargados de que haya comunicación entre dispositivos con microcontroladores, el concepto es muy amplio y abarca desde interfaces físicas como el USB hasta los virtuales como puertos virtuales integrados en un chip. Existen dos tipos de puertos que se distinguen según la forma en la que envían la información:

- **Paralelos**: Envían la información a través de múltiples canales de forma simultánea, por lo que son puertos que necesitan buses con mayor número de cables o pistas.
- **Serie:** Envía la información secuencialmente, en forma de secuencias de bits, para ello el puerto necesita de dos canales para realizar la comunicación, RX (receptor) y TX (transmisor). También existen otros conductores para referencia de tensión, sincronización con la señal de reloj, etc. [11]

En este trabajo sólo se abordará teóricamente el puerto serie, ya que estos son la forma principal de comunicar la placa Arduino con otros dispositivos. El puerto serie es un componente fundamental de gran cantidad de proyectos y con el que sacar todo el potencial de la plataforma Arduino. Gracias a él se puede, desde simular la escritura de un usuario por teclado hasta controlar otros dispositivos a través de internet.

Arduino dispone de un bus UART (Transmisor-Receptor Asíncrono Universal) integrado, con el que controla los puertos y dispositivos serie, opera a nivel TTL 0V/5V, es directamente compatible con la conexión USB y está compuesto por 3 pistas fundamentales: TX, RX y GND. De tal manera que el pin TX del dispositivo esté conectado al RX de la placa Arduino y viceversa y que las masas de ambos estén conectadas, permitiendo así la comunicación de manera unidireccional, cuyo esquema se puede observar en la Figura 8, es decir, sólo 1 envía y sólo 1 recibe. El puerto serie en la placa está unido físicamente a los pines 0(RX) y 1(TX), mientras este se utiliza no se pueden utilizar dichos pines como entradas o salidas digitales. [12]

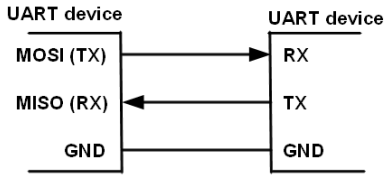

Figura 10. Diagrama conexión a bus UART

El UART comúnmente transmite 8 bits de datos de la siguiente forma: un bit de inicio, a nivel bajo, seguidamente los 8 bits de datos y al final un bit de parada a nivel alto. Como ventajas: no necesita señal de reloj para sincronizarse, la velocidad está limitada a 2Mbps. [13]

2.7. Comunicación WiFi

Arduino Yún tiene 2 interfaces de red separadas, un puerto Ethernet 10/100 Mbit/s periférico, se encuentra unido a la placa y tiene conexión directa con el procesador que se encarga de llevar a cabo la comunicación mediante el sistema operativo Linux, y una interfaz WiFi integrada en el chip microprocesador con un estándar IEEE 802.11 b/g/n, soportando encriptación WEP/WPA/WPA2. La interfaz WiFi puede tener 2 configuraciones diferentes:

- **Conexión red local:** De este modo se establece comunicación entre Arduino y el enrutador (encargado de interconectar redes, por ejemplo, una red en el hogar con internet), dando así acceso a internet al sistema. Al igual que con la conexión por cable Ethernet, se le asigna una dirección IP local al dispositivo, desde la que se podrá acceder tanto a la configuración interna a través del Arduino Web Panel como a la raíz del servidor en Linux mediante el cliente SSH.

- **Punto de acceso:** En este modo cualquier dispositivo WiFi puede conectarse directamente a la red creada en la Yún, aunque no dispondría de acceso a internet, podría funcionar como núcleo para un grupo de sensores WiFi. [14]

El sistema operativo Linux simplifica los medios para acceder utilizando muchas de las herramientas que se utilizan en cualquier computadora, dando así la posibilidad de acceder mediante la comunicación WiFi al servidor creado, desde una ip local o desde la propia red de la Yún.

Para ir un paso más allá en la conexión inalámbrica, una vez configurada la interfaz WiFi o Ethernet de Arduino, el siguiente método consiste en la modificación de los parámetros iniciales del enrutador de la red doméstica, es decir, redireccionar los puertos de entrada de cualquier dirección ip asignada por cualquier red a la que se conecte un dispositivo externo a los puertos de la red local a la cual está conectado el Arduino, permitiendo así el acceso remoto al sistema y dando la capacidad de recibir órdenes a través de los puertos locales desde cualquier red. [15] (VER ANEXO: Config. Red Doméstica)

2.8. LM35

2.8.1. Introducción

El LM35 es un sensor de temperatura en forma de circuito integrado con una salida lineal proporcional en la que cada grado equivale a 10mV. Tiene la ventaja respecto a los sensores calibrados en grados Kelvin de que el usuario no necesita extraer un gran número de medidas a un voltaje constante para conseguir una escala constante en grados centígrados.

Este sensor no requiere de ningún tipo de calibración o ajuste para conseguir una precisión de ¼ºC en una habitación y de 2ºC en el exterior, en un rango de entre -55ºC y 150ºC. También tiene asegurado un bajo coste por el ajuste y la calibración interna, además de su baja impedancia de salida. Por todo esto el LM35 es considerado un sensor de lectura y circuitería especialmente fácil.

El dispositivo se puede alimentar mediante una única fuente de alimentación de continua o por un regulador de tensión positiva y negativa, extrayendo 60 µA de dicha alimentación y provocando un autocalentamiento de menos de 0,1ºC al aire libre. [16]

2.8.2. Acondicionamiento de la señal

Como se ha explicado en el punto anterior, el LM35 es un sensor con una salida analógica, es decir, proporciona un voltaje proporcional a la temperatura, en la que cada grado equivale a 10mV. La medida que tomamos de un sensor analógico con Arduino mediante la función *analogRead* da un valor entre 0 y 1023, 1024 posibles valores. Por lo tanto, si el sensor esta alimentado a 5V, 0V sería un valor de 0 y 5V sería 1023.

A partir de esta información se puede concluir una fórmula matemática:

$$Temperatura = Valor(sensor) * 5 * 100/1023$$

2.8.3. Esquema hardware

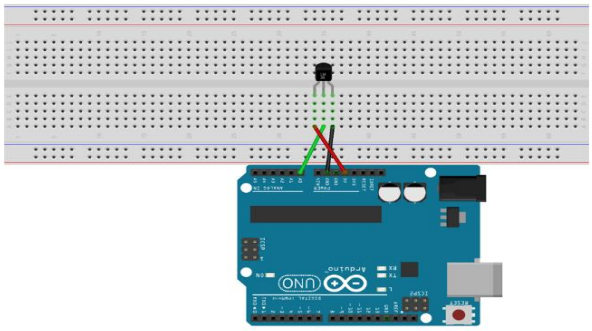

Figura 11a. Esquema hardware LM35

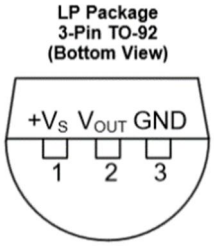

Figura 11b. Esquema pines sensor LM35

En las Figuras 11a y 11b se muestra una posible conexión de los pines a la placa Arduino y el esquema de pines del encapsulado del sensor de temperatura LM35, indicando cual se debe conectar a VCC, cual masa y a través de cuál de las patillas se obtiene la señal de salida que se conectará al pin analógico del Arduino y cuyo dato se transformará para obtener el dato en voltios y así poder calcular la temperatura en grados centígrados.

2.9. Sensor PIR

2.9.1. Introducción

Los sensores infrarrojos pasivos son dispositivos utilizados para la detección de movimiento. Se usan con frecuencia para aplicaciones domóticas o sistemas de seguridad debido a que son baratos, pequeños y fáciles de usar.

Básicamente el PIR está formado por un sensor piezoeléctrico que se encarga de la medición de la radiación infrarroja (Figura 12b). Todos los cuerpos emiten cierta cantidad de energía, siendo esta directamente proporcional a la temperatura de dicho cuerpo. Este sensor es capaz de captar esa radiación emitida y convertirla en una señal eléctrica. [17]

Cada sensor está dividido en dos partes debido a que su cometido es la detección de movimiento, es decir, un cambio en la señal recibida, no una medición del nivel medio de radiación infrarroja. Ambos campos están unidos, y es por eso por lo que si reciben la misma cantidad de infrarrojos en los dos la señal eléctrica resultante es nula. De esta forma, si un cuerpo atraviesa uno de los campos de medición se genera una señal diferencial, que es captada por el sensor, y se emite una señal digital.

El otro elemento que lo compone es la óptica del sensor (Figura 12a). Una cúpula de plástico formada por lentes de fresnel, que enfoca la radiación hacia cada uno de los campos del PIR. [18]

Figura 12a. Cúpula del sensor de movimiento

Figura 12b. Sensor piezo eléctrico del PIR

2.9.2. Diagrama de bloques

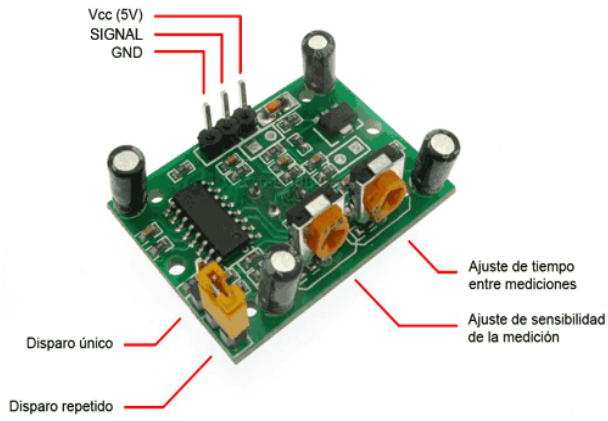

Figura 12c. Patillaje sensor PIR

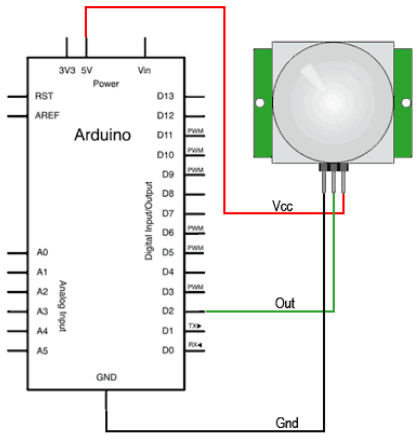

Figura 13. Esquema eléctrico de conexión a placa Arduino

2.10. Módulo bluetooth (Hc-06)

2.10.1. Introducción

Se denomina Bluetoooth al protocolo de comunicaciones que posibilita la transmisión de voz y datos mediante un enlace por radiofrecuencia, diseñado especialmente para dispositivos de bajo consumo, que requieren corto alcance de emisión y basados en transceptores de bajo costo. Siendo los principales objetivos:

- Comunicación fácil entre equipos móviles.
- Eliminar el cableado entre dispositivos.
- Creación de redes inalámbricas y facilitar la sincronización de datos.

Los dispositivos pueden comunicarse entre sí cuando se encuentran dentro de su alcance. Las comunicaciones se realizan por radiofrecuencia, por lo tanto, los dispositivos no tienen que estar alineados y pueden estar en habitaciones separadas si la potencia de transmisión es suficiente. Según la potencia de transmisión los dispositivos pueden clasificarse como “Clase 1”, “Clase2” o “Clase 3”.

El hardware del dispositivo está formado por:

- Dispositivo de radio, modula y transmite la señal.
- Controlador digital, procesador de señales digitales (DSP) llamado controlador de enlace y de las interfaces con el dispositivo anfitrión. Se encarga de las funciones de transferencia tanto asíncrona como síncrona, la codificación de audio y el cifrado de datos

Cuando dos dispositivos bluetooth se vinculan, se identifican por nombre y dirección interna y solicitando la clave PIN para autorizar la conexión. Una vez realizado el emparejamiento con éxito, ambos nodos guardan la identificación para volverse a vincular cuando se encuentren cerca sin necesidad de intervención manual.

Los comandos AT básicos que se utilizan para configurar el módulo Hc-06 son:

- AT+VERSION, versión del Firmware
- AT+NAMEXXX, Programa el nombre con el que se identificara el dispositivo frente a otros nodos bluetooth.
- AT+BAUDX, velocidad de comunicación entre el módulo y la consola según a la siguiente tabla:
    - 1 1200bps
    - 2 2400bps
    - 3 4800bps
    - 4 9600bps (Predeterminado)
    - 5 19200bps
    - 6 38400bps
    - 7 57600bps
    - 8 115200bps
- AT+PINXXXX, configura el número PIN, que se requerirá para establecer la vinculación. [19]

2.10.2. Diagrama de bloques

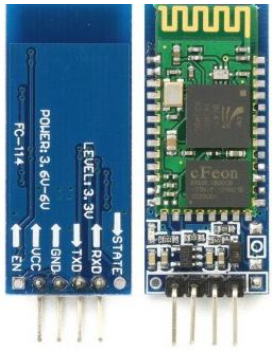

Figura 14. Módulo bluetooth Hc-06

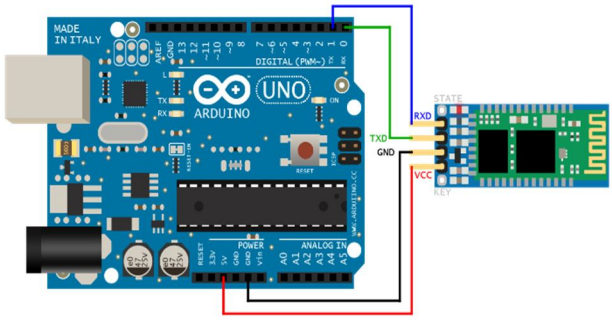

Figura 15. Conexión de módulo bluetooth a la placa arduino

En la Figura 14 se puede ver el módulo bluetooth que se va a utilizar, en este caso es el Hc-06, que dispone de 4 patillas, 2 para alimentación (VCC) y masa (GND) y otras 2 para la transmisión (RXD) y recepción (TXD) de datos.

Este módulo se conectará a la placa Arduino Yún tal y como se muestra en el diagrama de bloques representado en la Figura 15.

2.11. Cámaras

2.11.1. Webcam

2.11.1.1 Introducción

La cámara web que se va a utilizar para la vigilancia del hogar es una estándar USB, en este caso, de la marca Microsoft, modelo Lifecam hd-3000. Sus especificaciones técnicas son:

- Calidad de imagen 720p.
- Pantalla panorámica 16:9.
- Buena calidad de imagen en casi todas las situaciones de iluminación.
- Micrófono con reducción de ruidos.

2.11.1.2 Programa MJPG-Streamer

MJPG-Streamer es una aplicación de línea de comandos que copia cada uno de los frames en JPEG capturados por un dispositivo de entrada o una webcam compatible con Linux-UVC y lo transmite como MJPG vía HTTP. Se usa para emitir a través de una red inalámbrica desde una webcam a diferentes tipos de reproductores como Chrome, Firefox, VLC, etc. [20]

Su funcionamiento está basado en un plugin (de entrada) que copia las imágenes JPEG a un directorio de acceso global, mientras que otro plugin (de salida) procesa las imágenes, transmitiéndolas como simple fichero de imagen, o bien, emitiéndolas en formato MPEG. Estos plugins consisten en:

- **mjpg_streamer:** Herramienta de línea de comandos que copia las imágenes JPEG de un dispositivo de entrada a uno o más plugins de salida.
- **input_uvc.so:** Captura los frames JPEG de la webcam conectada.
- **output_http.so:** Servidor web HTTP que sirve una única imagen JPEG o bien emite en MJPEG. [21]

2.11.2. OV7670

2.11.2.1 Introducción

La OV7670, mostrada en la Figura 16, es un sensor CMOS de bajo voltaje y muy bajo coste y tamaño, para captura de imagen, que ofrece toda la funcionalidad de una cámara VGA y un procesador de señal en un solo integrado. Esta ofrece gran cantidad de formatos de imagen, controlada a través del Serial Camera Control Bus (SCCB) compatible con la comunicación I2C.

Es capaz de captar imágenes a una velocidad de hasta 30 fps en VGA con un control completo por parte del usuario sobre la calidad de la imagen, el formato y su transmisión. Todas las funciones de procesado de la imagen como establecer parámetros gamma, de balance de blancos o de saturación son programables a través del SCCB. Además, incluye tecnología sensor para mejorar la calidad de la imagen y eliminar cualquier ruido que pueda trastocarla. [22]

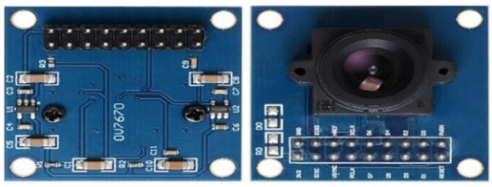

Figura 16. Hardware cámara OV7670

Especificaciones técnicas del módulo OV7670:

- Tamaño de la óptica 1/6 pulgadas
- Resolución 640x480 VGA
- Regulador integrado. Necesita 3.3V
- Lente F1.8 / 6 mm
- Funciones de control automático de imagen incluidas
- Controles de calidad de la imagen como saturación, gamma, brillo…
- Incluye reducción de ruido y corrección de defectos.
- Autoajuste del nivel de saturación.

2.11.2.2 Diagrama de bloques y esquema hardware

La Figura 17a representa el diagrama de bloques de la conexión entre el módulo de la cámara OV7670 y el host, en este caso será la placa Arduino Yún. También se muestra en el esquema de la Figura 17b, un ejemplo de cómo conectar las patillas de la cámara a los pines de una placa Arduino.

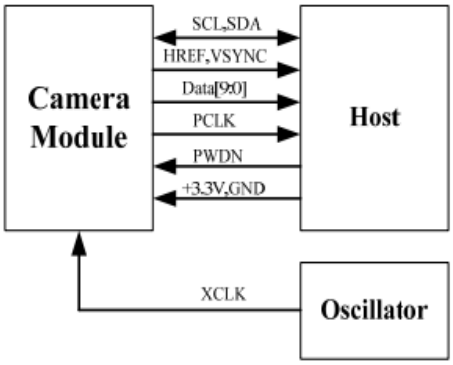

Figura 17a. Diagrama de bloques de cámara OV7670

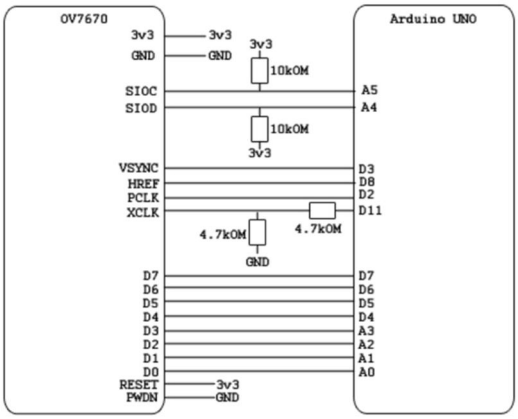

Figura 17b. Esquema conexión a placa Arduino

Capítulo 3.

Software

3.1. Entornos de programación

3.1.1. Arduino (IDE)

Entorno de desarrollo software de Arduino en el cual se escribe el código en AVR C para enviar las órdenes al microcontrolador. Destaca por ser fácil, sencillo e intuitivo por la posibilidad de programar en lenguaje C y por la inclusión de librerías predefinidas para la configuración de los puertos de comunicación o de más alto nivel para llevar a cabo operaciones específicas. El software incorpora compilador, mostrado en la Figura 19a, para poder comprobar el código y un monitor serie (Figura 19b) para ver la respuesta recibida desde el microcontrolador cuando ejecuta el programa. [23]

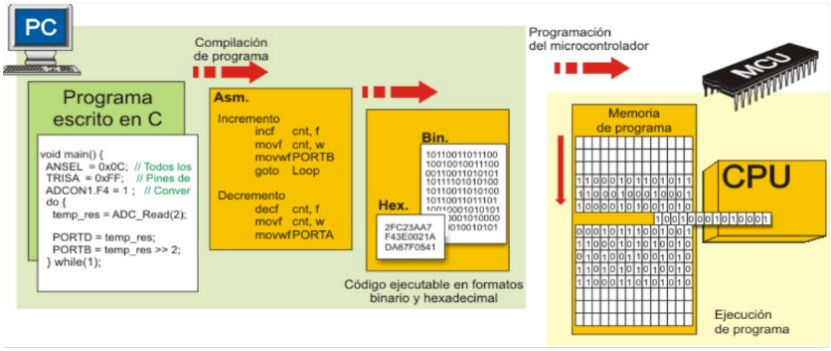

Figura 18. Programación del microcontrolador

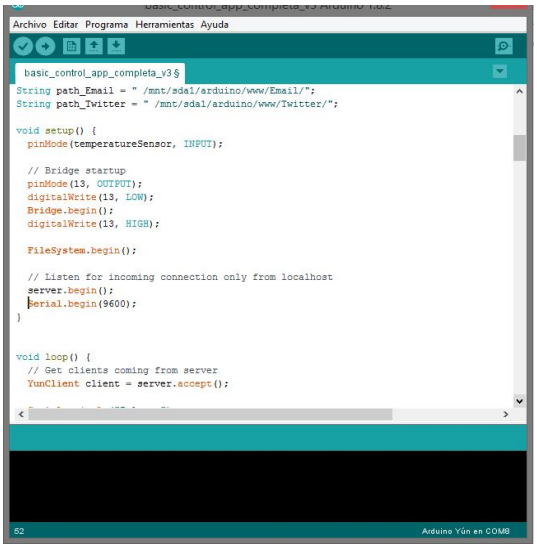

Figura 19a. Interfaz Arduino (IDE)

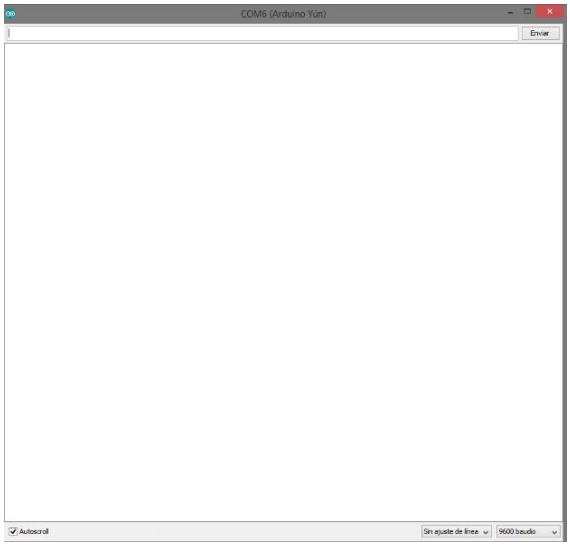

Figura 19b. Interfaz Monitor Serie de Arduino (IDE)

3.1.2. PuTTY (Linux)

Es un cliente SSH y Telnet con el que podemos conectarnos a servidores remotos iniciando una sesión en ellos que nos permite ejecutar comandos. Entre sus ventajas destacan: gratuito y de código abierto, disponible tanto para plataforma Windows como Linux, portable, sencilla y con multitud de opciones. [24]

Es el entorno elegido para comunicarse con el microprocesador con sistema operativo Linino, cuya interfaz se puede ver en la Figura 20a, y a través del cual se iniciará sesión en el servidor instalado en el Arduino Yún, accediendo a él con las mismas credenciales que las que utilizamos en el Arduino Web Panel. Una vez conectado al servidor se crea una línea de comandos, como se muestra en la Figura 20b, que serán procesados por Linux para agregar los programas necesarios para el proyecto o administrar la configuración interna del servidor. [25]

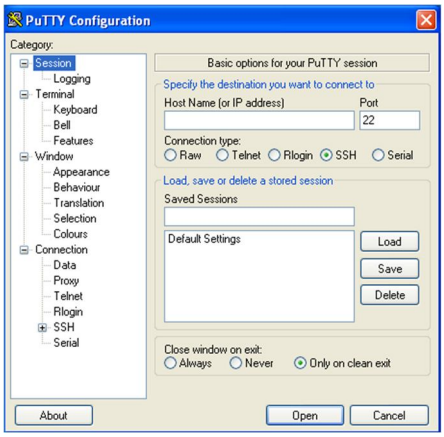

Figura 20a. Interfaz PuTTY

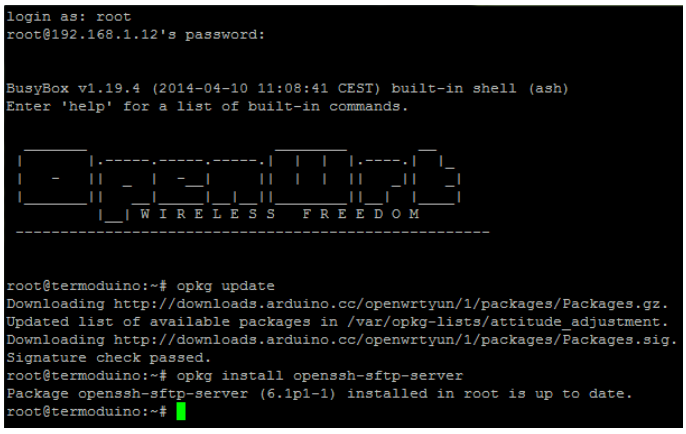

Figura 20b. Línea de comandos de PuTTY

3.2. Lenguajes de programación

3.2.1. Python

Python es un lenguaje de programación de propósito general, cuya expansión y popularidad es relativamente reciente. Está orientado a objetos, con la posibilidad de realizar cualquier tipo de programa, desde aplicaciones a servidores de red o incluso páginas web. Además, es un lenguaje que no necesita que se compile su código fuente para poder ejecutarlo, esto lo dota de ventajas como el rápido desarrollo. Todo esto junto a la gran cantidad de librerías, tipos de datos y funciones incorporadas, la sencillez y velocidad con la que se programa, haciendo que el código tenga entre 3 y 5 líneas menos que en lenguaje java o C, la cantidad de plataformas en las que se puede desarrollar y la gratuidad de Python es lo que ha hecho que en los últimos años se haya hecho tan popular.

Haciendo más hincapié en las características más importantes de este lenguaje:

- **Propósito general:** Capacidad para crear todo tipo de programas. No está concebido específicamente para la web, pero sí que permite el desarrollo de estas.
- **Interpretado**: Es decir, se realiza una compilación antes de ejecutar el código, pero de manera transparente al programador.
- **Interactivo:** Dispone de un intérprete por línea de comandos que permite ejecutar sentencias produciendo un resultado visible, ayudando a entender el lenguaje y a comprobar los resultados de porciones de código.
- **Orientado a objetos:** Ofrece una manera sencilla de crear programas con componentes reutilizables.

- **Funciones y librerías:** Contiene muchas funciones incorporadas en el propio lenguaje para tratamiento de strings, archivos, etc. Además, permite importar librerías en los programas para tratar temas específicos, por ejemplo, en este trabajo se añaden librerías como Dropbox o Twitter para desarrollar sistemas en red.
- **Sintaxis clara:** Tiene una notación identada (con márgenes) de obligado cumplimiento. Para separar porciones de código, en comparación con otros lenguajes en los que se utilizan llaves o palabras clave como *begin* y *end*, sólo se tiene que tabular hacia dentro, poniendo margen al código que iría dentro de una función o bucle. Esto hace que se adopten unas mismas notaciones y que los programas tengan aspecto muy similar.
- **Multiplataforma:** Cualquier sistema es compatible con este lenguaje siempre que contenga un intérprete para él.

En este caso una de las características importantes por las que se ha escogido el lenguaje Python es la capacidad de trabajar en cualquier sistema, esto supone una gran ventaja en Arduino Yún ya que el procesador Atheros AR9331 en Linux lleva instalado el paquete python2.7 el cual permite interpretar el lenguaje y permite la ejecución de programas que se alojarán en el servidor para implementar funciones avanzadas que el microcontrolador de Atmel no puede llevar a cabo. [26] [27]

3.2.2. HTML

Lenguaje de marcado que se utiliza en el desarrollo de páginas web, sus siglas corresponden a *Hipertext Markup Lenguage,* con este él se establece la estructura y el contenido de un sitio web, tanto de texto, objetos e imágenes. Los archivos creados usan la extensión .htm o .html.

Se trata de un sistema de formato abierto, es decir, en el cual se permite ordenar y etiquetar diversos documentos dentro de una lista sin que existan reglas para la organización de dicho contenido.

El lenguaje HTML se encarga de describir los documentos y su estructura complementando el texto con diversos objetos (como imágenes, animaciones, etc.). El contenido de dicha página se crea a partir de etiquetas, también llamadas “tags”, que permiten interconectar conceptos y formatos. Estas están especificadas en el texto a través de paréntesis angulares: < y >, dando forma a la estructura esencial del lenguaje mediante el contenido en sí mismo y sus atributos.

Además, este lenguaje permite enlazar códigos externos que se conocen como *scripts* a los documentos HTML, los cuales mandan instrucciones al navegador que se encarga de procesar el lenguaje. Dentro del contenido de una página web mediante la etiqueta *script* importaremos estos códigos indicando la ubicación en la que se encuentra el archivo, haciendo que el

navegador al procesar la web cargue el contenido como si formara parte de la estructura formal de la página y ejecutando unas órdenes específicas.

Hay que remarcar que HTML no es un lenguaje de programación ya que no cuenta con funciones aritméticas, variables o estructuras de control, por lo que genera únicamente páginas web estáticas, sin embargo, se puede usar en conjunto con diversos lenguajes de programación para la creación de páginas web dinámicas, como PHP o JavaScript. [28] [29] [30]

3.2.3. JavaScript

Se trata de un lenguaje de programación que se utiliza principalmente para la creación de páginas web dinámicas. Este no necesita compilar el código, es decir, el programa se puede probar directamente ejecutándolo desde el navegador sin necesidad de procesos intermedios.

Una web dinámica es aquella que incorpora efectos como animaciones, acciones que se ejecutan al pulsar botones, ventanas emergentes con mensajes de aviso al usuario, etc.

En este trabajo se programarán los scripts necesarios para ejecutar acciones dinámicas dentro de las páginas web creadas a partir de código HTML en lenguaje JavaScript, por eso a continuación se explican las maneras de integrar los scripts en las páginas web:

- **Incluir directamente en el documento HTML:**
  El código JavaScript se encierra entre las etiquetas <script> (para abrir) y </script> (para cerrar) en cualquier parte del documento, aunque se recomienda incluirlo en la cabecera del documento. También es necesario para que sea válido el atributo "*type*" y en este caso con el valor "*text/javascript*".
- **Código en archivo externo:**
  Las instrucciones JavaScript se programan en un archivo externo (con extensión .js) que se enlaza a los documentos HTML mediante la etiqueta <script>, se pueden enlazar todos los archivos que sean necesarios. En este método además del "*type*" hay que definir el atributo "*src*" que indica la URL con la ubicación del archivo JavaScript. Cada etiqueta <script> solamente puede enlazar un archivo, pero se pueden incluir todas las etiquetas con URL necesarias en la misma página web.
- **Código directamente en los elementos HTML:**
  Este método consiste en incluir los trozos de código directamente en los elementos HTML, provocando que se ensucie el código HTML y haciendo dificultoso el mantenimiento del JavaScript.

La sintaxis de este lenguaje de programación está definida por el siguiente conjunto de normas básicas:

- No se tienen en cuenta los espacios en blanco y las nuevas líneas, pudiendo ordenar el código de forma adecuada para entenderlo mejor con tabulaciones de línea, creando nuevas líneas, etc.
- Distinción entre mayúsculas y minúsculas.
- No es necesario definir el tipo de las variables al ser creadas.
- No es necesario terminar cada sentencia con el punto y coma.
- Se pueden incluir comentarios para añadir información en el código fuente del programa añadiendo dos barras oblicuas (//) al principio de la línea que se quiere comentar. [31]

Otra técnica importante para el desarrollo de las páginas web dinámicas es AJAX, este es el acrónimo de "*Asynchronous JavaScript and XML*". Permite que, mediante programas escritos en JavaScript, una vez la web haya sido cargada, el navegador pueda intercambiar información de forma asíncrona con un servidor en múltiples formatos: texto, HTML, JSON o XML.

Una página web ya puede contener un enlace a través del cual se solicite información a un servidor o incluso un formulario a través del cual se envíen datos a un servidor que envía de vuelta una respuesta generada respecto a la información recibida, habiendo en ambos casos una conexión entre cliente y servidor. La diferencia que supone el uso de AJAX es que el envío de información se lleva a cabo asíncronamente, es decir, permite guardar la respuesta del servidor en una variable JavaScript de manera que se pueda acceder a ella en cualquier momento, evitando recargar la página web por completo. [32]

Por ejemplo, cuando en una página es necesario introducir un período en un formulario web y esta información tiene que ser enviada al servidor para tomar datos de un sensor cada cierto tiempo, pero la página tiene que seguir activa para poder ver la gráfica en tiempo real, por lo tanto, tiene que evitarse la recarga de la página.

Capítulo 4.

Aplicaciones

4.1. Diagrama del software del sistema

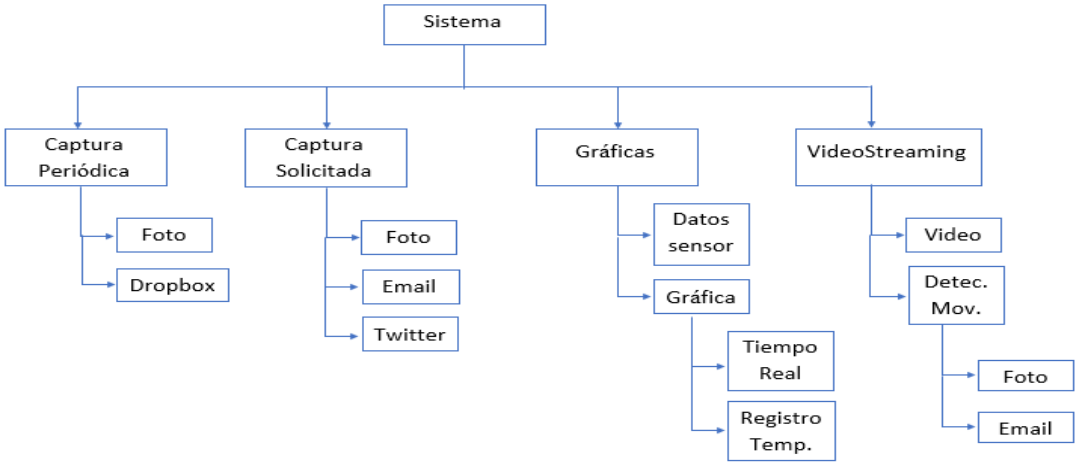

Figura 21. Diagrama software del sistema

4.2. Captura de Imagen Periódica

4.2.1. Introducción

Esta aplicación, como se muestra en el diagrama de la Figura 21, se basa en la captura de imágenes de forma periódica desde una de las cámaras conectadas al nodo IoT y su almacenamiento en una cuenta Dropbox. Para el acceso remoto al nodo sensor se utilizará una conexión a internet Wifi o por cable de Ethernet, con la configuración previa de los parámetros de la red a la que se conecta o por bluetooth mediante el módulo Hc-06, al que se vinculará el móvil mediante la App para móviles Android. Se han desarrollado dos tipos de interfaz de usuario, una página web a la que se accede a través de internet y una aplicación para teléfonos móviles, que a diferencia de la web permite también la comunicación bluetooth.

El sistema estará en reposo a la espera de que el usuario decida cuando utilizar esta aplicación (Figuras 22 y 23). El programa principal está implementado en el entorno de desarrollo de Arduino, de su ejecución se encargará el microcontrolador de Atmel, y se divide en dos partes, según si se reciben las órdenes desde internet o desde bluetooth. Una vez recibida la orden, según la cámara escogida por el usuario, se procede a la realización de la captura con la configuración adecuada.

Figura 22. Captura periódica en reposo.

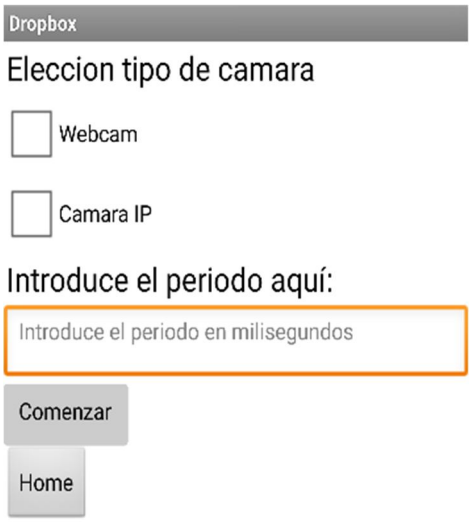

Figura 23. Captura periódica en reposo (App)

Al comenzar la aplicación el usuario escoge la cámara con la que vigilar el hogar, después escoge el periodo y el nodo IoT lo guarda para repetir la ejecución de dicha aplicación cíclicamente hasta que el usuario decida detenerla o escoger un periodo nuevo. A continuación, el microcontrolador conectará con el procesador Linux para enviar el comando con el que se ejecutará la captura (Figuras 24 y 25). Una vez realizada, se etiqueta con la fecha y la hora del instante en el que se ha realizado (con formato DD-MM-AAAA, hh-mm-ss) y se guarda en el servidor web de Arduino. Por último, se ejecuta desde el procesador, el archivo con el que acceder a la cuenta Dropbox del usuario, que contiene las credenciales de acceso programadas en lenguaje Python, y se almacena en la carpeta predeterminada. (VER ANEXO: Config. Redes Sociales).

Figura 24. Captura de imágenes activa.

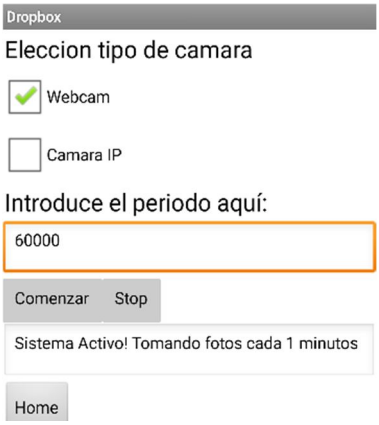

Figura 25. Captura activa cada minuto (App)

La aplicación se ejecutará cíclicamente tomando fotos de la estancia y subiéndolas a Dropbox de manera automática, y una vez pulsado el botón “Stop”, se detiene y da la opción de ver las capturas realizadas o devolver el sistema a su estado de reposo (Figuras 26 y 27).

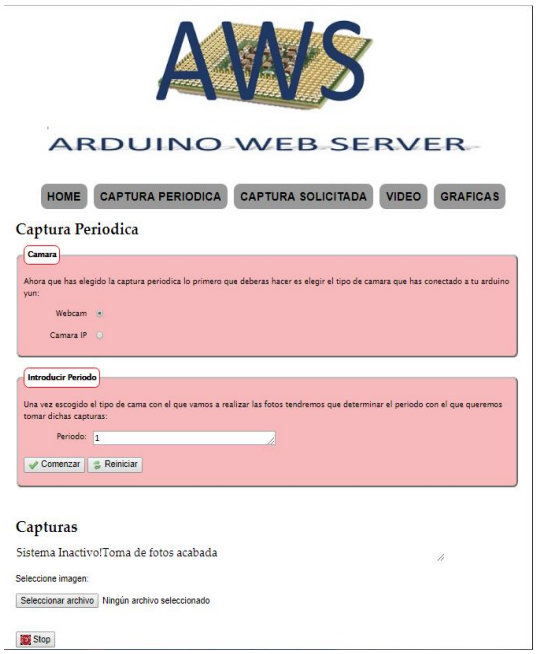

Figura 26. Sistema inactive. Captura periódica detenida

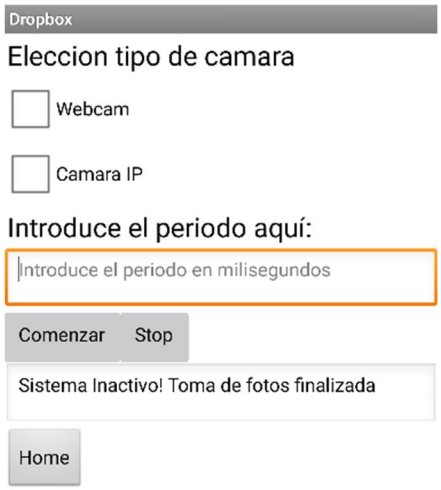

Figura 27. Captura detenida (App)

4.2.2. Diagrama de flujo

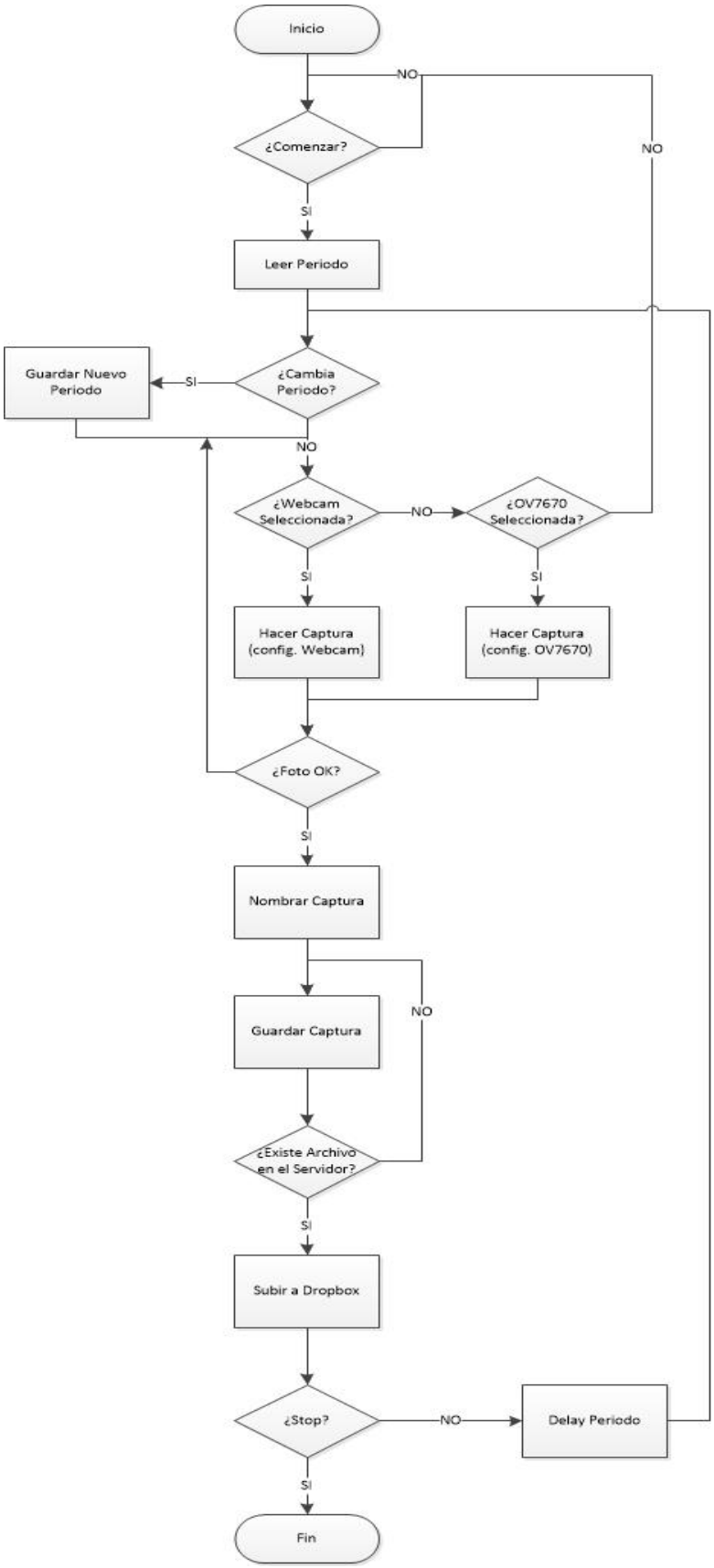

Figura 28. Diagrama de bloques de la captura periódica

4.3. Captura de Imagen a Demanda del Usuario

4.3.1. Introducción

La siguiente aplicación, cuyo esquema básico de funcionamiento puede verse en la Figura 21, consiste en la captura de imágenes de la vivienda desde las cámaras conectadas al nodo IoT cuando lo solicita el usuario; después se muestra la imagen en la interfaz y se envía automáticamente al email del usuario, además da la opción de escribir un Tweet que se publicará en la cuenta de Twitter junto a la captura realizada. (VER ANEXO: Config. Redes Sociales).

Como en el resto de las aplicaciones implementadas en el Arduino Yún, el programa principal se implementará en Arduino (IDE), y cuando sea necesario se accederá desde el procesador a los archivos con funciones descritas en lenguaje Python.

En primer lugar, se decide con que cámara se vigila la estancia donde se encuentra el módulo Arduino, y el sistema se queda en reposo hasta que el usuario pulse el botón “Comenzar” (Figuras 29 y 30). Una vez pulsado, el nodo IoT ejecutará la captura con la configuración correspondiente a la cámara escogida (Figuras 31 y 32). Además, se guarda la imagen en el servidor Arduino con la etiqueta del instante en el que se ha realizado, con el formato fecha y hora (dd-MM-aaaa, hh-mm-ss).

Figura 29. Captura solicitada en reposo.

Figura 30. Captura solicitada en reposo (App)

Figura 31. Sistema activo. Captura realizada.

Figura 32. Captura realizada (App)

Después, a través del procesador Linux se ejecutan los archivos Python con los que se envía la imagen al email del usuario, con el formato que se muestra en la Figura 33, además de mostrar la captura a través de la interfaz. Si el usuario pulsa sobre la imagen mostrada, se abrirá una ventana nueva con la foto ampliada (Figuras 34 y 35). A continuación, pulsando sobre la casilla “Escribir Tweet”, aparecerá un campo en blanco para introducir los 140 caracteres. Una vez se ha escrito el tweet, y pulsando el botón “Publicar tweet”, automáticamente se publica en el perfil de Twitter. Además, el sistema manda un mensaje directo a los seguidores para avisar de la nueva publicación en la cuenta (Figura 36).

Figura 33. Imagen recibida en el email desde la App.

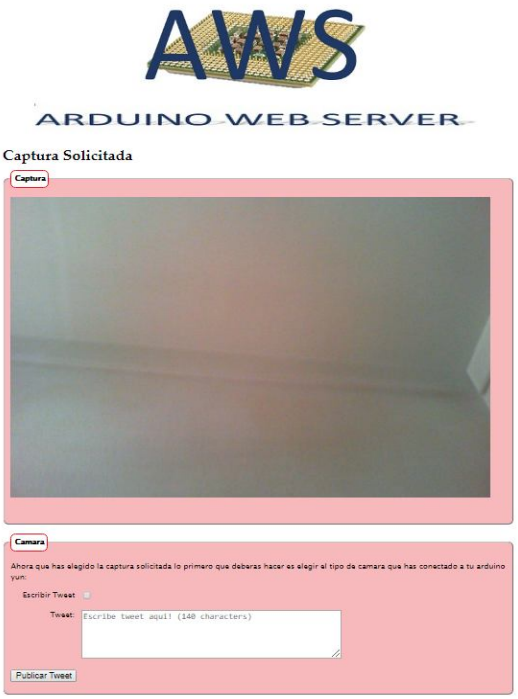

Figura 34. Ventana emergente para publicar tweet.

Figura 35. Escribir tweet (App).

Por último, la ventana emergente se cerrará automáticamente y volverá a la interfaz inicial, a la espera de que se solicite una nueva captura o que se escriba un nuevo Tweet con la imagen tomada de la vivienda.

Figura 36. Tweet publicado.

4.3.2. Diagrama de flujo

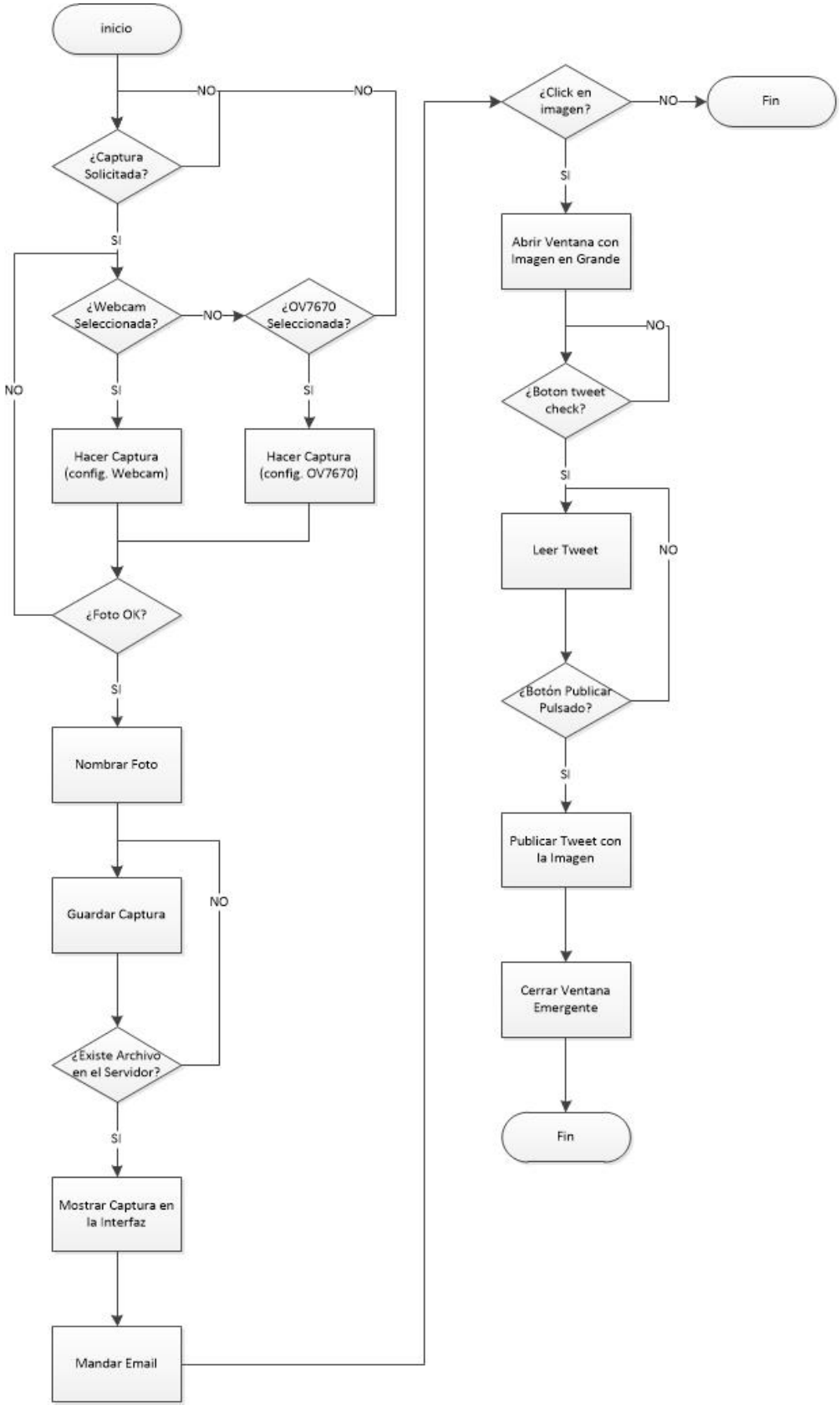

Figura 37. Diagrama de bloques de la captura solicitada

4.4. Gráficas de Datos de Sensores

4.4.1. Introducción

Esta aplicación, como muestra el esquema de la Figura 21, es la que más se diferencia del resto. Mediante el sensor de temperatura LM35 se recogerán datos de temperatura de la vivienda y se guardarán en un registro histórico. A través de la web el sistema puede representar la gráfica de temperatura en tiempo real, y una vez detenido el sistema, permite acceder al registro de archivos con mediciones en el servidor Arduino, para mostrar la gráfica histórica con los datos recopilados la última vez que se ejecutó la aplicación.

El funcionamiento de la aplicación comienza con la elección del periodo por parte del usuario, dejando el sistema en reposo hasta que se pulse el botón "Comenzar", como se muestra en las Figuras 38 y 39. Cuando se activa el sistema, habilita el sensor de temperatura para que comience a tomar datos, además cada vez que capta un dato lo guarda automáticamente en un archivo de texto en el servidor Arduino, a su vez en otro archivo guarda el valor de la hora a la que ha sido tomado dicho dato, en formato hh-mm-ss.

Mientras la aplicación siga activa, el nodo IoT seguirá guardando datos de temperatura cíclicamente y dando la opción de mostrar la gráfica en tiempo real, representando la variación de los valores recogidos frente al tiempo, y actualizándose con cada ejecución del programa (Figura 40). La lectura del sensor se detiene al pulsar el botón "Stop" y automáticamente el microcontrolador se encarga de borrar los archivos que guardaban los datos en tiempo real en el servidor Arduino. A continuación, crea un histórico en el que incluye todos los datos recopilados y se etiqueta con la fecha y la hora de la última ejecución de la aplicación (dd-MM-aaaa, hh-mm-ss). Por último, cuando el registro histórico se ha creado, la interfaz permite mostrar la gráfica con todos los datos guardados en el último histórico (Figura 41).

La principal diferencia de la implementación de esta aplicación respecto a las demás es que más allá de una programación relativamente sencilla del microcontrolador, hay otra parte importante desarrollada en la interfaz web. Se trata de la posibilidad de mostrar tanto la gráfica de temperatura en tiempo real, mientras el sistema esté activo, como la gráfica con los datos recopilados en el registro histórico de la última ejecución. Para ello se han creado archivos con funciones programadas en JavaScript que aportan a la web la capacidad de enviar peticiones HTTP con los datos de los formularios web sin la necesidad de recargar la página, lo que posibilita la lectura de los datos de los archivos y la actualización en tiempo real sin volver a iniciar la página.

Figura 38. Gráficas en reposo

Figura 39. Lista de periodos que el usuario puede escoger

Figura 40. Sistema activo. Tomando datos y representándolos cada medio minuto

Figura 41. Toma de datos detenida. Muestra de registro de la última vez que el sistema estuvo activo.

4.4.2. Diagrama de flujo

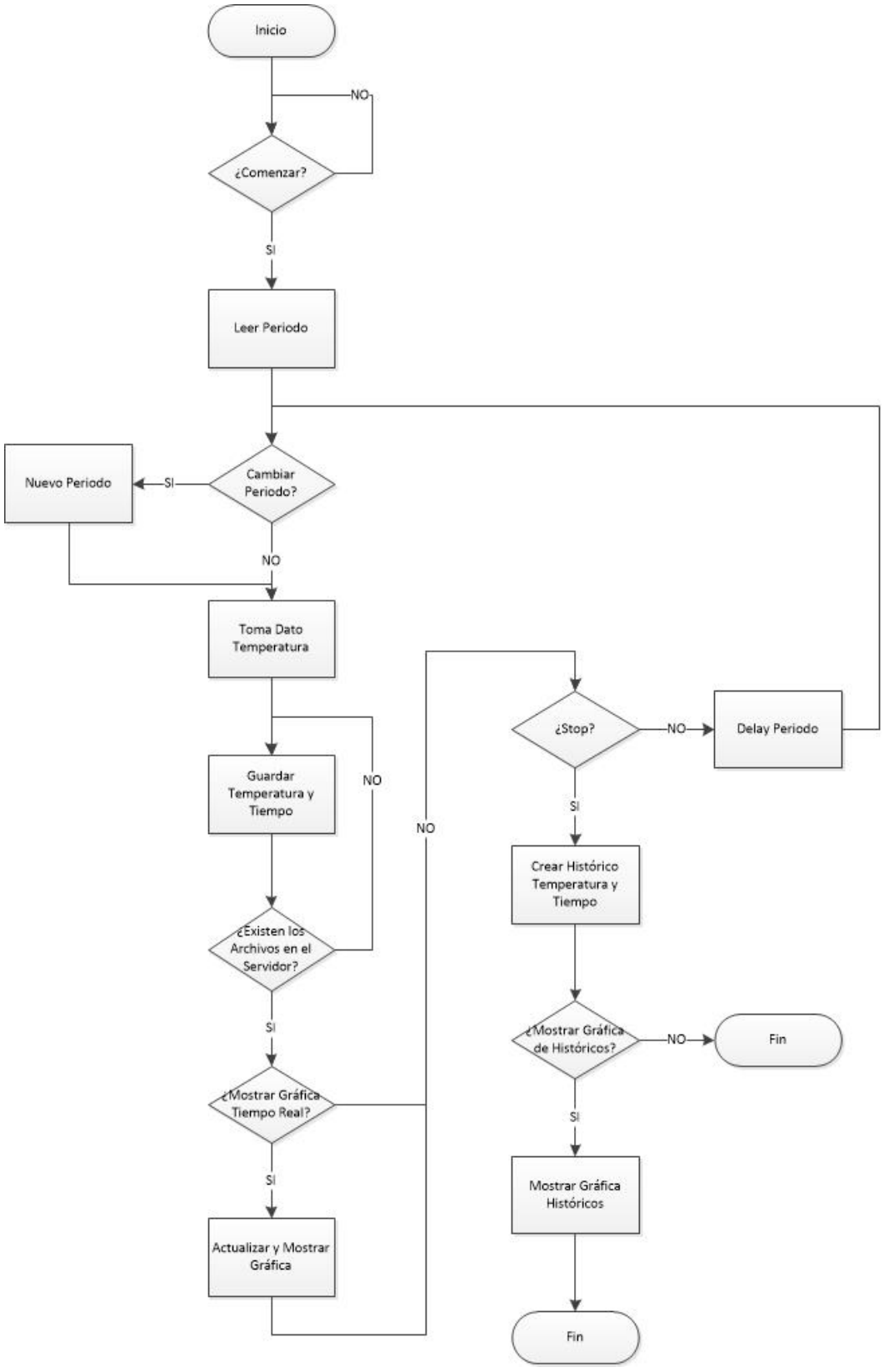

Figura 42. Diagrama de bloques de las gráficas

4.5. VideoStreaming y Disparo por Sensor de Presencia

4.5.1. Introducción

En esta aplicación del nodo IoT, como se muestra en el diagrama de bloques de la Figura 21, se emitirá vídeo en directo. A la vez, el sistema se mantiene alerta de que el sensor de proximidad PIR detecte movimiento en la vivienda, momento en el que se capturará una serie de cuatro instantáneas consecutivas que se almacenarán en el servidor Arduino, y también se mandarán adjuntas en un email que notificará al usuario que se ha detectado alguna anomalía en el hogar, indicando la fecha y la hora a la que se realizaron dichas capturas.

La aplicación comienza con la elección de la cámara con la que se va a visualizar la estancia del hogar, y manteniendo el sistema en reposo hasta que el usuario pulse el botón “Comenzar” (Figuras 43 y 44). Cuando se activa la aplicación, el nodo IoT habilita el sensor de proximidad PIR y comienza la captura de imágenes a través de la cámara conectada al Arduino Yún.

Figura 43. VideoStreming en reposo

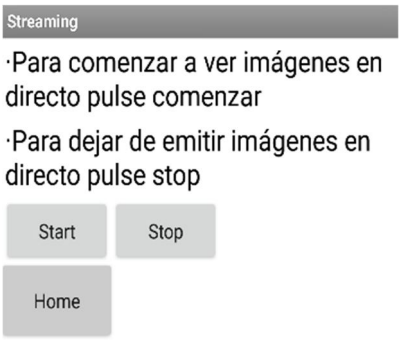

Figura 44. VideoStreaming en reposo (App)

Mientras el sensor no detecte se emitirá vídeo en *streaming* a través de la página web o de la App para teléfonos móviles (Figuras 45 y 46). Si se detecta una perturbación en la vivienda se activará el sensor de proximidad y detendrá el vídeo en directo, momento en el que se realizan cuatro capturas consecutivas que se guardan automáticamente en el servidor Arduino, y a continuación, se envía un email al usuario con las imágenes adjuntas junto con la fecha y la hora a la que se detectó dicha perturbación, sirviendo así de aviso por parte del sistema de que algo ocurrió.

Figura 45. Sistema activo. Imágenes en streaming.

Figura 46. VideoStreaming activo (App)

A continuación, si no se ha detenido la aplicación, de manera predeterminada se vuelve a emitir el vídeo en *streaming* hasta que el sensor detecte de nuevo. Cuando el usuario desee detener la aplicación de forma permanente tendrá que presionar el botón “Stop”, y así cesará el funcionamiento la captura de imágenes y la detección, volviendo a entrar en reposo el nodo IoT.

4.5.2. Diagrama de flujo

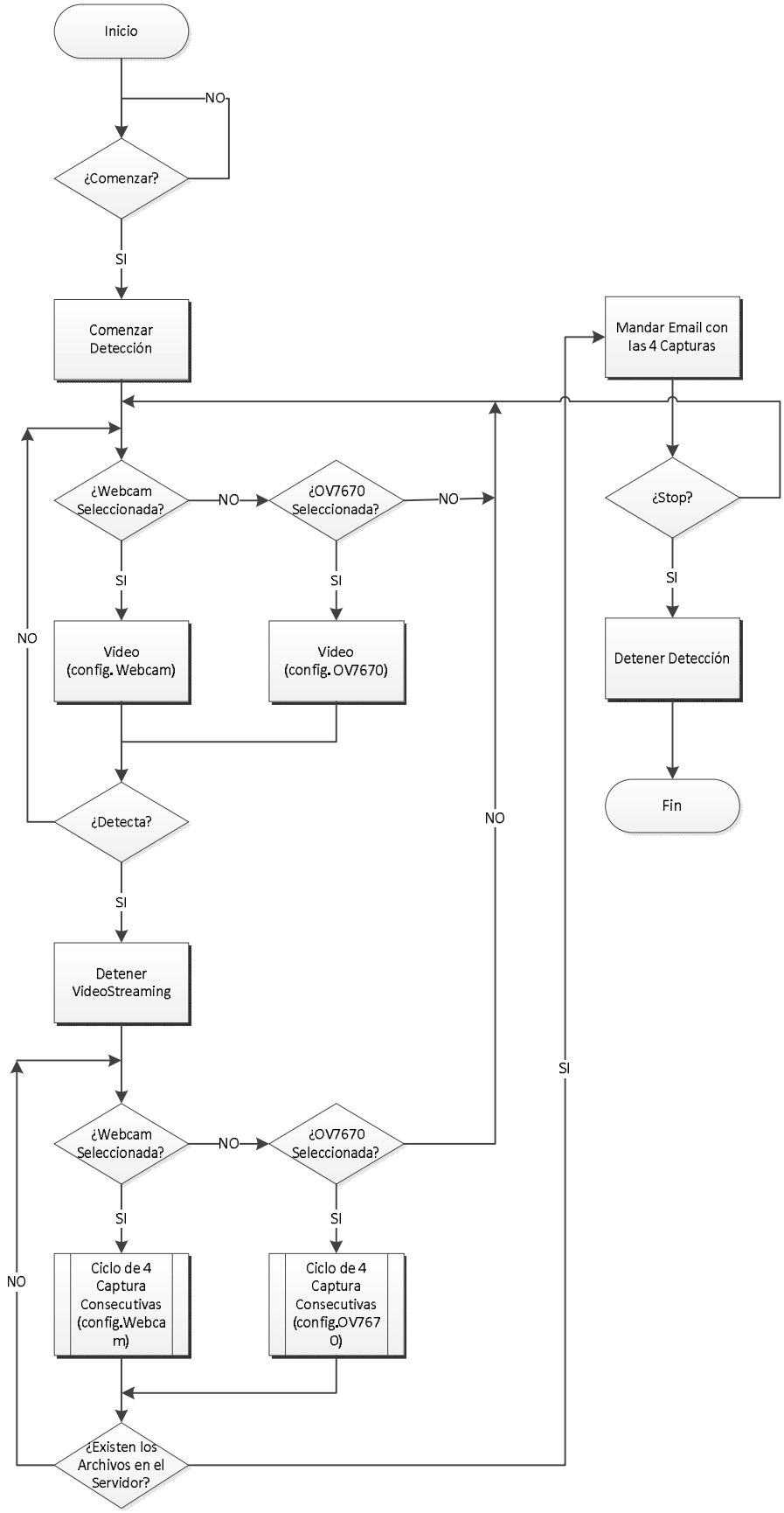

Figura 47. Diagrama de bloques de Videostreaming

Capítulo 5.

Prototipado Hardware

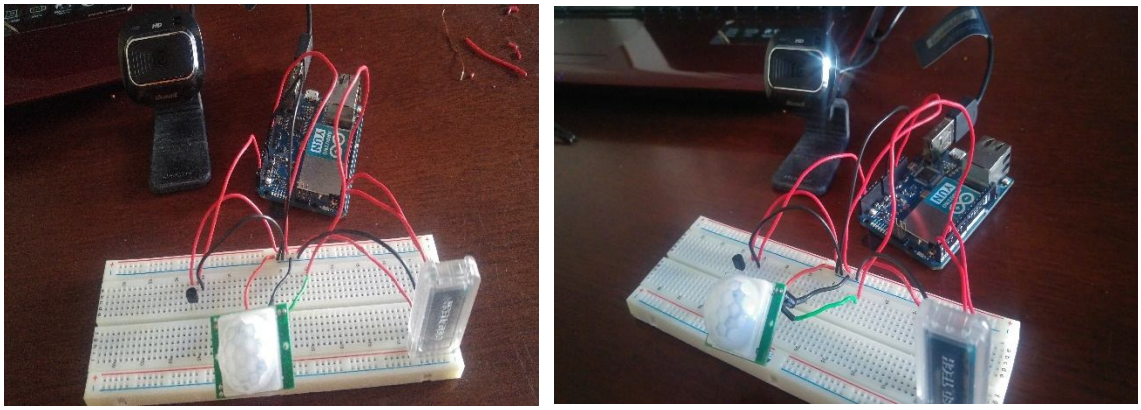

Figura 48. Periféricos conectados al nodo IoT

En la Figura 48 se muestra la conexión de 4 periféricos al nodo IoT Arduino Yun: sensor de temperatura, sensor de proximidad, módulo bluetooth (con el que nos conectaremos al dispositivo Arduino desde el teléfono móvil) y la cámara web (que conectaremos al puerto USB integrado en la placa). Los dispositivos conectados comparten señal de alimentación y de masa, aportadas por el módulo Arduino Yún, y la señal de salida de cada uno la conectaremos a un pin diferente, analógico o digital, ya que en el entorno de desarrollo podemos configurar los analógicos como entradas digitales.

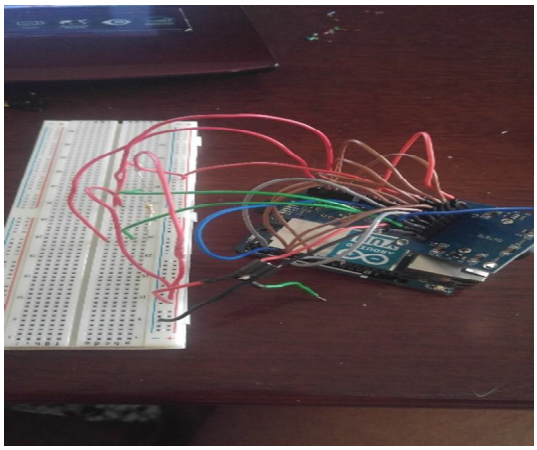

Figura 49. Conexión cámara de bajo coste OV7670

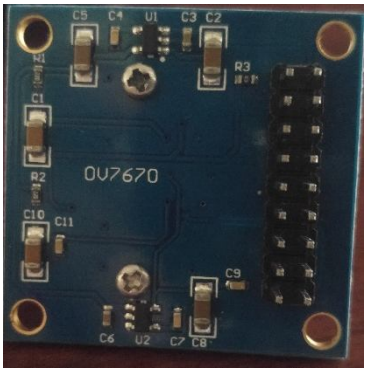

Figura 50. Pines de cámara OV7670

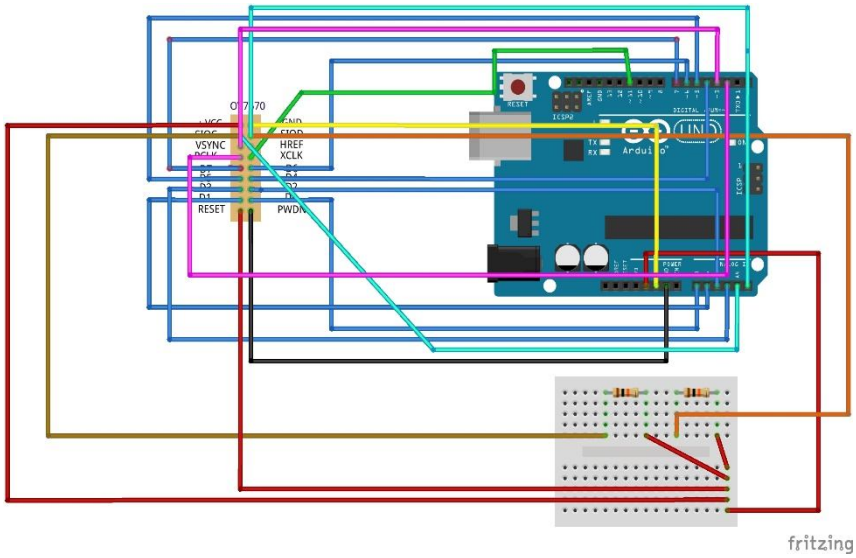

Figura 51. Esquema hardware de la conexión del módulo OV7670

En la Figura 49 se ve el módulo OV7670 conectado a la placa Arduino, pero en el prototipado no se aprecia bien como está cableada debido a la cantidad de conexiones que necesita, por eso, la Figura 50 muestra la disposición de los pines en la parte trasera de la cámara, y la Figura 51 el cableado con la etiqueta de cada pin para saber cómo ha ido conectado exactamente a la placa Arduino Yún.

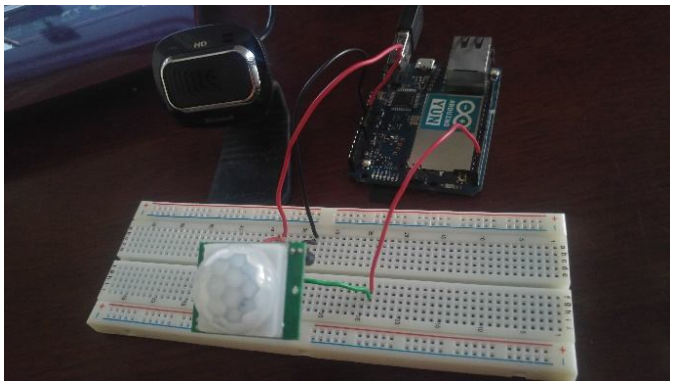

Figura 52. Conexión sensor PIR

En la Figura 52 se muestra el cableado del sensor PIR conectarlo a Arduino. El cable rojo de la izquierda va conectado a la salida de 3.3V de la placa (VCC), el negro es la señal GND y el otro cable rojo es la señal de salida del sensor, por lo tanto, va conectado a uno de los pines digitales que configuraremos como entrada.

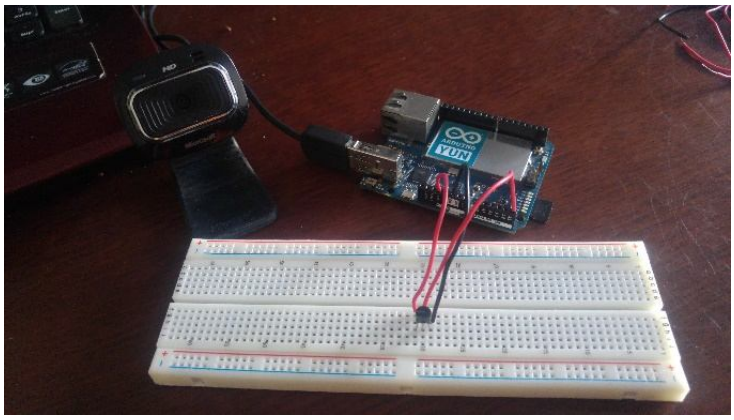

Figura 53. Conexión sensor LM35

Como se observa en la Figura 53, al igual que en la conexión del sensor de proximidad, tendríamos 3 patillas, VCC que conectamos a la alimentación que nos da el Arduino, GND lo uniríamos a la señal GND de la placa, y Vout lo conectamos a uno de los pines para que el Arduino pueda recibir la señal del sensor.

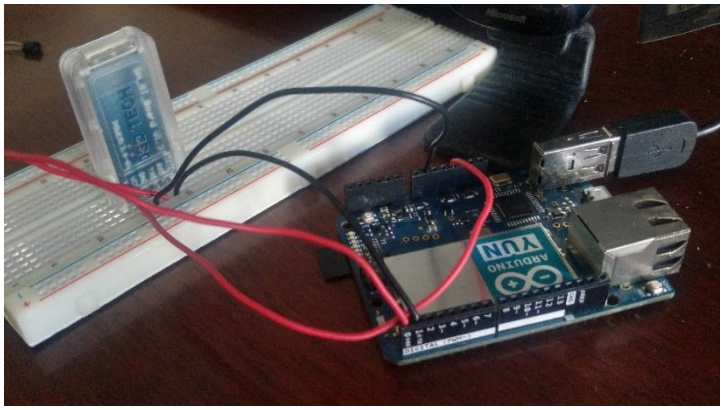

Figura 54. Conexión módulo bluetooth

El módulo bluetooth, Figura 54, en orden de izquierda a derecha, tiene el pin VCC que se conecta a la alimentación de 3.3V que obtenemos de la placa Arduino, el siguiente es masa (GND) que también conectamos al Arduino, después el pin TXD va conectado al pin digital 0 (RXD), y el RXD al pin digital 1 (TXD) del módulo Arduino, para permitir la transmisión de datos entre ambos dispositivo.

Todas las aplicaciones explicadas en el Capítulo 4 han sido probadas mediante los prototipos descritos, comprobándose su correcto funcionamiento. Algunos de los problemas que iban surgiendo se explican en el Capítulo 6 (Conclusión).

Capítulo 6.

Conclusión

En este proyecto se ha desarrollado un prototipo de nodo IoT basado en la plataforma Arduino Yún, con cuatro aplicaciones implementadas para su funcionamiento tanto desde una interfaz web como desde una APP para móviles Android, conectando con el nodo vía internet o bluetooth. Para ello se ha realizado un estudio sobre los componentes de la placa Arduino Yún y su funcionamiento interno, la configuración y las técnicas necesarias para la conexión de cámaras y sensores, su programación, y ejecución.

Así, se ha alcanzado el objetivo global planteado: el desarrollo de un prototipo de nodo IoT que permita el acceso a distancia a sensores y cámaras "low-cost" instalados en el hogar del usuario, de una forma barata y sencilla, disponiéndose de un prototipo plenamente operativo, como se ha comprobado en el laboratorio.

La configuración básica de la placa Arduino ha supuesto la conexión a internet y la instalación de los programas necesarios en el PC para poder acceder al interior del sistema a través del microprocesador Linux que incluye, así como la incorporación de una tarjeta SD para ampliar la memoria del módulo Arduino. Esto es necesario para dotar al sistema de la capacidad de instalar todos los paquetes necesarios en Linux y, además, poder alojar los archivos del servidor web Arduino.

No obstante, a lo largo del desarrollo del proyecto se han encontrado problemas diversos, casi siempre relacionados con la falta de documentación por parte de los fabricantes, desarrolladores de software o proveedores de servicios de internet.

Así, durante el proceso de configuración surgieron problemas con la instalación de algunos de los paquetes necesarios debido a la falta de documentación oficial disponible, teniendo que recurrir a menudo a foros o blogs de internet o a "prueba y error". Por ejemplo, el paquete Dropbox instalaba en el sistema la versión más actual, pero no cumplía los requisitos necesarios, debido a la falta de documentación para consultar, se tuvo que avanzar a base de prueba y error, y se ha tenido que restaurar el sistema a sus valores de fábrica (VER ANEXO: Primeros Pasos en Arduino Yún) y repetir el proceso numerosas veces hasta tenerlo completamente preparado para la ejecución de las aplicaciones.

También se encontraron dificultades a la hora de conectar el sistema a una red que necesitara de usuario y contraseña para poder acceder a ella, como

por ejemplo la red WiFi de la EINA, ya que desde la configuración interna del dispositivo Arduino solo permite la conexión a redes que necesiten tan solo la introducción de una contraseña. Hasta este momento no se ha encontrado solución, pero se seguirá estudiando el problema para poder hacer el sistema más accesible.

La implementación de las aplicaciones se ha llevado a cabo mediante programación en el entorno de desarrollo de Arduino. Esto ha supuesto la necesidad de conocer y entender muchas de las librerías que incluye el software, tanto las de propósito general como las dedicadas únicamente para el modelo Arduino Yún. Por ejemplo, la librería "Bridge" permite acceder a los pines digitales y analógicos con órdenes desde el navegador con la IP local, teniendo la placa Arduino conectada a la misma red que el PC. Como en algunos casos los ejemplos que incluyen las librerías no son suficientemente explicativos, de nuevo se ha tenido que consultar información en internet (foros, blogs), especialmente, en los casos de la creación y escritura de archivos de texto, obtener el tiempo actual mediante la conexión a internet, y recibir y procesar las órdenes a través del cliente web, siendo estas las más problemáticas a la hora de entender el funcionamiento y conseguir utilizarlas satisfactoriamente.

Por último, se abordó el desarrollo tanto de la aplicación para móviles con sistema operativo Android como de las páginas web. Para implementar la versión de prueba de la App se ha tenido que estudiar la aplicación web "*AppInventor*" para la programación por bloques que ofrece el MIT (*Massachussets Institute of Technology*), pero debido a la falta de información y ejemplos en el propio entorno de programación sobre cómo utilizar los componentes incluidos, surgieron dificultades al necesitar las funcionalidades más complejas, como la conexión mediante internet, a través de la cual se envían las órdenes al cliente web programado en Arduino. O como la conexión bluetooth, con la que acceder al nodo IoT sin necesidad de internet. Debido a dichas dificultades se ha desarrollado únicamente una versión demostrativa. Además, el diseño de las páginas web ha supuesto el estudio de los lenguajes de programación HTML5, para la creación de los elementos y los atributos de la web semántica, y a causa de la dificultad de programar páginas web dinámicas también se tuvieron que utilizar otros lenguajes como complemento, en este caso Python y JavaScript.

En definitiva, el proyecto en su parte básica es plenamente operativo, pero un mejor acabado excede con mucho el tiempo y esfuerzo que conlleva un TFG valorado en 12 ECTS. Como desarrollos futuros se plantean las siguientes posibilidades:

- La realización de un estudio de consumo energético para conectar el sistema a una batería externa y poder dotarlo de mayor portabilidad, dando la posibilidad al usuario de usarlo en mayor número de ocasiones

y en diferentes circunstancias, evitando la necesidad de una conexión USB al ordenador para poder alimentar la placa Arduino Yun, y dependiendo sólo de una conexión wifi que permita al nodo IoT acceder a internet.

- Para poder dotar al sistema de mayor accesibilidad y hacerlo más intuitivo, será necesario un desarrollo definitivo y "más profesional" tanto del interfaz web como de la App para móviles, pudiéndola publicar en un futuro para abrirla al uso por parte de cualquier usuario y dando la posibilidad de registrarse en el sistema e introducir las credenciales tanto del email como para el inicio de sesión en cualquiera de las redes sociales con las que interactúa el nodo IoT.
- Estudiar la manera de implementar la configuración de la conexión wifi de la placa Arduino desde la interfaz de las aplicaciones, facilitando así la tarea y evitando la necesidad de acceder a la configuración interna del módulo Arduino por parte del usuario.

Bibliografía

[1] D. Evans, «Internet de las Cosas: Como la próxima evolución de internet lo cambia todo,» https://www.cisco.com/c/dam/global/es_mx/solutions/executive/assets/pdf/internet-of-things-iot-ibsg.pdf.

[2] «Wired: What is the Internet of Things?,» http://www.wired.co.uk/article/internet-of-things-what-is-explained-iot.

[3] «Technopedia: Internet of Things (IoT),» https://www.techopedia.com/definition/28247/internet-of-things-iot.

[4] «Arduino: ¿What is Arduino?,» https://www.arduino.cc/en/Guide/Introduction.

[5] «Aprendiendo Arduino: ¿Qué es Arduino?,» https://aprendiendoarduino.wordpress.com/2016/12/11/que-es-arduino-2/.

[6] «Arduino: Arduino Yún,» http://arduino.cl/arduino-yun/.

[7] «PanamaHitek: Arduino Yún. Características y Capacidades,» http://panamahitek.com/arduino-yun-caracteristicas-y-capacidades/.

[8] «Microchip: ATmega32U4,» https://www.microchip.com/wwwproducts/en/ATmega32u4#datasheet-toggle.

[9] A. Corporation, «Microchip: ATmega32U4,» http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Datasheet.pdf.

[10] I. Atheros Communication, «Atheros AR9331: Datasheet,» https://www.openhacks.com/uploadsproductos/ar9331_datasheet.pdf.

[11] «Comunicación de Arduino por Puerto Serie,» https://www.luisllamas.es/arduino-puerto-serie/.

[12] «Puertos y Buses: I2C y UART,» https://geekytheory.com/puertos-y-buses-1-i2c-y-uart.

[13] «Aprendiendo Arduino: Comunicaciones con Arduino,» https://aprendiendoarduino.wordpress.com/2014/11/18/tema-6-comunicaciones-con-arduino-4/.

[14] «Características hardware de la nueva Arduino Yún,» https://www.mexchip.com/2013/09/revision-de-las-caracteristicas-hardware-de-la-nueva-arduino-yun/.

[15] «Aprendiendo Arduino: WiFi en Arduino,» https://aprendiendoarduino.wordpress.com/2016/11/12/wifi-en-arduino/.

[16] «Datasheet LM35,» http://www.electronicoscaldas.com/datasheet/LM35_TI.pdf.

[17] A. L. Systems, «PIR Motion Sensor,» https://cdn-learn.adafruit.com/downloads/pdf/pir-passive-infrared-proximity-motion-sensor.pdf.

[18] «¿Qué es un sensor PIR?,» https://www.luisllamas.es/detector-de-movimiento-con-arduino-y-sensor-pir/.

[19] «Aprendiendio Arduino: Bluetotth en Arduino,» https://aprendiendoarduino.wordpress.com/2016/11/13/bluetooth-en-arduino/.

[20] «Mjpeg-streamer,» https://github.com/jacksonliam/mjpg-streamer.

[21] «Video Streaming Live!,» https://geekytheory.com/video-streaming-live-con-raspberrypi-y-playstation-eye.

[22] «CMOS OV7670 Datasheet,» https://www.openhacks.com/uploadsproductos/ov7670_cmos_camera_module_revc_ds.pdf.

[23] «Aprendiendo Arduino: Primeros Pasos Programación Arduino,» https://aprendiendoarduino.wordpress.com/category/ide/.

[24] «Hostinger: ¿Cómo funciona el SSH?,» https://www.hostinger.es/tutoriales/que-es-ssh.

[25] «VozIdea: ¿Qué es PuTTY y para qué sirve?,» http://www.vozidea.com/que-es-putty-y-para-que-sirve.

[26] «¿Qué es Python?,» https://desarrolloweb.com/articulos/1325.php.

[27] Z. A. Shaw, Aprenda a programar con Python, Anaya Multimedia, 2014.

[28] «Definición de HTML,» https://www.definicionabc.com/tecnologia/html.php.

[29] «Acerca de HTML: ¿Qué es HTML y para que sirve?,» http://www.acercadehtml.com/manual-html/que-es-html.html.

[30] M. Fletcher y J. W. Lowery, HTML5 para desarrolladores, Anaya Multimedia, 2012.

[31] J. Eguíluz Pérez, «Introducción a JavaScript,» https://www.jesusda.com/docs/ebooks/introduccion_javascript.pdf.

[32] «Digital Learning: ¿Qué es AJAX?,» https://www.digitallearning.es/blog/que-es-ajax/.

[33] «Arduino: Arduino Yún Rev 2,» https://store.arduino.cc/arduino-yun-rev-2.

[34] «MIT App Inventor: Library & Support,» http://appinventor.mit.edu/explore/library.html.

[35] L. LLamas, «Comunicación de arduino con puerto serie,» https://www.luisllamas.es/arduino-puerto-serie/.

[36] «Arduino YUN Schematics,» https://www.arduino.cc/en/uploads/Main/YUN-V04(20150114).pdf.

[37] «USB Basic Support,» http://wiki.openwrt.org/doc/howto/usb.essentials.

[38] «Arduino: Guide to Setup Streaming Web Cam on the YUN,» http://forum.arduino.cc/index.php?topic=188690.0;topicseen.

[39] «Install UVC Webcam,» http://www.ibuyopenwrt.com/index.php/8-yun-compatible/60-uvc-webcam.

[40] «Install Non UVC Cam,» http://www.ibuyopenwrt.com/index.php/8-yun-compatible/61-non-uvc-camera-gspca-camera.

[41] «Tutorial de Arduino, Bluetooth y Android: Crear App con MIT App Inventor,» https://robologs.net/2015/10/29/tutorial-de-arduino-bluetooth-y-android-2-crear-una-app-con-mit-inventor/.

[42] «Send Mail with Gmail and Ssmtp,» https://www.nixtutor.com/linux/send-mail-with-gmail-and-ssmtp/.

[43] «Arduino: Reflashing the OpenWrt-Yun image on the Yún,» https://www.arduino.cc/en/Tutorial/YunUBootReflash.

[44] «Arduino: How to expand the Yún disk space,» https://www.arduino.cc/en/Tutorial/ExpandingYunDiskSpace.