



Control remoto de herramientas de *hardware* libre con Node.js y Firmata*

Remote control of free hardware tools with Node.js and Firmata

Cristian Enrique Gamboa **
Carlos William Sánchez ***
Jhon Edward Ordóñez ****

Rec: Abril 2017

Acep: Octubre 2017

Resumen

Uno de los principales retos en la elaboración de proyectos de carácter informático, específicamente en el área de Internet de las Cosas (IoT, por sus siglas en inglés), es la integración de *software* y *hardware*, debido a la barrera de la integración latente en dicha labor, junto a los posibles temores por la incursión en estas tendencias. En ese orden, se han creado iniciativas que permiten una ágil interacción entre los componentes tipo lógico, y los de tipo físico. Una de estas iniciativas, es la plataforma de hardware *Arduino*, la cual es de gran utilidad en desarrollos de automatización y monitoreo remoto. En este trabajo, se propone una forma de controlar diferentes actuadores electrónicos conectados a un dispositivo *Arduino*, siendo esta una de las herramientas de *hardware* libre de mayor uso en los procesos de automatización, mediante la implementación de métodos de programación *web*, permitiendo al usuario una incursión en los conceptos de la robótica y la automatización. Al final, se presentan dos casos de uso en los que se evidencia una comunicación simple entre el servidor, alojado en un mini ordenador comercial diseñado para solucionar problemas de espacio y portabilidad, *Raspberry Pi*, y la tarjeta *Arduino* y sus actuadores. Como consecuencia, se establecen las bases para la elaboración de proyectos, relacionados con la robótica, implementando monitoreo remoto, es decir, se muestra la incursión en los procesos de automatización, con el uso de *hardware* y *software* libre, mostrando de manera práctica los conceptos de *IoT*.

Palabras claves: control, integración, robótica, automatización, Internet de las Cosas.

* Este trabajo fue financiado dentro del proyecto de investigación titulado "Diseño y fabricación de una plataforma robótica para la inspección de ductos de alcantarillado para tuberías con diámetros iguales o superiores a 1 metro" con código asignado FI-010106, de la Fundación Universitaria Católica Lumen Gentium.

** Cristian Enrique Gamboa, Ingeniero en Sistemas, semillero docente programa de ingeniería en sistemas. Grupo KIMSA, Facultad de Ingeniería, Fundación Universitaria Católica Lumen Gentium. cgamboa001@unicatolica.edu.co

*** Carlos William Sánchez, MSc. en Física, profesor tiempo completo programas de ingeniería Industrial y de Sistemas. Grupo KIMSA, Facultad de Ingeniería, Fundación Universitaria Católica Lumen Gentium. Coordinado Semillero Métodos Físicos Aplicados a la Industria (MeFAI). e-mail: csanchez@unicatolica.edu.co

**** John Edward Ordoñez. PhD. en Física, profesor tiempo completo programas de ingeniería Industrial y de Sistemas. Grupo KIMSA, Facultad de Ingeniería, Fundación Universitaria Católica Lumen Gentium. Director Grupo de Investigación KIMSA. e-mail: jordonez@unicatolica.edu.co

Abstract

One of the main challenges in the elaboration of informatics projects, specifically in IoT is the integration of software and hardware, given a latent barrier existing in this task and the uncertainties for getting into tendencies. In the process, initiatives have been developed that allow efficient interactions between logical and physical components. One of this is the Arduino hardware platform, which is very useful in developments related to automatization and remote control. In this work, a new way to control different electronic devices connected to an Arduino dispositive is proposed, being this one of the most used free hardware in automatization processes, by the implementation of web programming methods, that allow the user to get immersed in the concepts of robotics and automatization. Two cases of application are presented showing simple communication between the server, resident in mini commander designed to solve space problems and portability, Raspery PI and the Arduino card. In this way, bases for the elaboration of projects related to robotics with remote control are established, through the using of free hardware and software and showing in a practical way, the IoT concepts.

Keywords: control, robotic integration, automatization, internet of things.

Introducción

En el campo de la ingeniería, es común escuchar el termino *software* libre, el cual hace referencia a su utilización en una forma subjetiva, en cuanto adaptación, mejora, estudio, redistribución, entre otros. De manera análoga surge el hardware libre, que como su contraparte, pertenece a una filosofía de acceso y distribución libre enmarcada en parámetros claros de derechos de autor. En resumen, el hardware libre hace referencia a un conjunto de dispositivos electrónicos cuyo diseño y especificaciones son de acceso público, es decir, es estos pueden ser modificados y distribuidos a disposición del usuario (E. de Negocios, 2009; Torres, J., Vila-Viña, D. y Lazalde, A, 2015).

Actualmente, en los campos de la robótica y la automatización, es muy frecuente emplear estos tipos de dispositivos, debido a que poseen características o especificaciones que permiten al usuario solucionar problemas a costos menores que las alternativas comerciales, además de disponer de forma gratuita, diagramas de diseño y conexión, sin restricciones de distribución. Con el uso de estos dispositivos se tiene una mayor flexibilidad y adaptabilidad de los mismos de acuerdo al proyecto que se desea desarrollar.

Arduino es una iniciativa de *hardware* libre nacida en Italia con la motivación de acercar profesionales y novatos de diferentes ramas a la automatización industrial. Actualmente empleada para el desarrollo de diferentes proyectos de robótica, domótica, automatización, entre otros (Golliav y Oli, 2016; Torrente, 2013). Esta iniciativa ha impulsado

el desarrollo de una familia de tarjetas de circuito programable, las cuales permiten la ejecución de programas cargados desde un ordenador a través del puerto serial universal (USB) (Torrente, 2013; Torres, Vila-Viña y Lazalde, 2015). Esta plataforma puede ser programada en un lenguaje *Arduino*, que es basado en el lenguaje *Processing*, el cual posee una sintaxis similar al lenguaje C++ (Torrente, 2013; Reas y Fry, 2015).

En este trabajo se mostrará una de las formas de programación de estas tarjetas de forma remota. Este proceso, se hace a partir del lenguaje *JavaScript*, el cual como es conocido, es un lenguaje de programación *web* que se ejecuta del lado del cliente (Flanagan, 2007). Además, se utilizará un entorno de tiempo de ejecución multiplataforma de código abierto para capa del servidor, *node.js*, el cual consta de un conjunto de funcionalidades en forma de archivos y directorios que actúan como base para la creación de aplicaciones *web*, de una forma más eficiente (Jordisan, 2014), junto con la librería *Firmata*, la cual permite la comunicación entre un mini ordenador y la tarjeta *Arduino*. Se establece la comunicación, utilizando la *web* como medio entre un computador y el mini ordenador *Raspberry Pi*. En el primero se visualizará la página *web* para el envío de órdenes, y en el segundo, se alojará el servidor que va a procesarlas. También, se muestra como realizar una comunicación, a través del puerto serial, entre el mini ordenador y la tarjeta *Arduino*, encargada de transmitir las ordenes a los diferentes periféricos co-

nectados a este. Todo esto permite obtener control y registro sobre diferentes dispositivos de *hardware*, a partir de medios *web*.

El documento se divide en dos partes. En la primera se describirá el fundamento teórico, haciendo una descripción de los conceptos y teorías utilizadas para construir las bases, y en la segunda, se mostrará la aplicación práctica, donde se describirá un proceso en el que se aplican, de manera sencilla, los conceptos tratados.

Fundamentos teóricos

Tarjeta de circuito programable

Una tarjeta de circuito programable es un dispositivo, que incorpora un microcontrolador, sobre el cual se escribe el código a ejecutar, y un conjunto de entradas analógicas (puertos) y digitales, las cuales permiten la conexión de otros dispositivos a manera de periféricos mediante una serie de ranuras extensoras, conocidas como *shields*, compatibles con la tarjeta, y que permiten operar diferentes actuadores como: motores, sensores, entre otros. Dichos actuadores le permiten a la tarjeta de circuito programable interactuar con el mundo exterior (Torrente, 2013).

Una de las tarjetas que se ha hecho más populares entre los expertos, y los no tan expertos, es la tarjeta *Arduino*, esto debido a su fácil uso, precio y facilidad de adaptación, ya que es parte del *hardware* libre disponible. Pero este tipo de herramientas compiten de cerca con un gran conjunto de tarjetas, como lo son la Galileo (Richardson, 2014), desarrollada por Intel, la Launchpad MSP430, hecha por *Texas instruments* (Agus, 2014), *Picaxe* que usa un procesador PIC (Anderson, 2017), Parallax Propeller (Uelmen, 2014), entre otras, todas alternativas de pago. Por otro lado, el *pcDuino* (Kurniawan, 2004) es también una alternativa gratuita, que no ha visto la luz con tanto arraigo como su alternativa, *Arduino*, debido a que aún es un poco compleja su implementación.

Dentro de los conceptos de filosofía libre, se asocia la iniciativa *Arduino* como un sistema, debido a que sus partes de *software* y *hardware* están entrelazadas (Loi y Goilav, 2016). Con el tiempo los desarrolladores de *Arduino* han extendido la familia de tarjetas programables, aumentando sus posibilidades en cuanto a *hardware*, y/o diseñando dispositivos más portables. A continuación, se relacionan algunas de las versiones de esta tarjeta debido a que es la que se usó en la implementación de los casos de uso, junto con sus especialidades (Torrente, 2013):

- **Arduino Uno:** es la tarjeta estándar de *Arduino*, y principalmente se compone de un microcontrolador Atmega328p. Además, posee 16 entradas digitales y 6 entradas analógicas.
- **Arduino Mega:** esta tarjeta cuenta con un microcontrolador Atmega2560, brindando más memoria que su versión anterior, además de un mayor número de entradas analógicas y digitales.
- **Arduino Ethernet:** Esta tarjeta es similar al *Arduino Uno*, pero además incluye un puerto de conexión a la red [4].

Debido a la filosofía que manejan sus desarrolladores, *Arduino* ha promovido la creación de una gama de dispositivos que abarcan, cada vez más, diferentes necesidades de investigación e innovación. En la Figura 1 se ilustra una de las versiones de *Arduino* más utilizada, *Arduino Uno* en su versión R3.

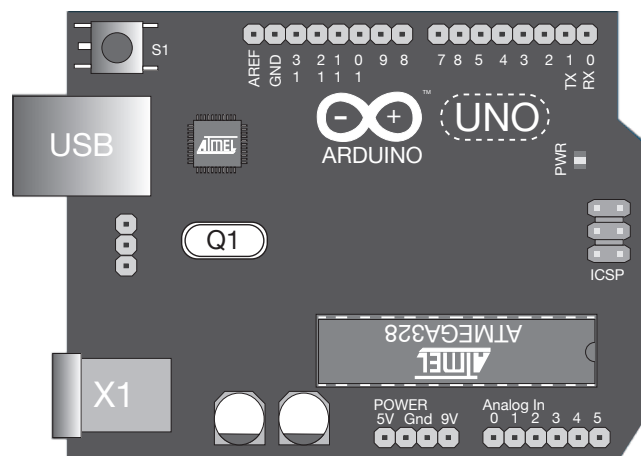


Figura 1. Arduino Uno R3

Fuente: Elaboración propia (2017).

Lenguaje de programación de Arduino

El lenguaje de programación utilizado para la codificación de la plataforma *Arduino*, posee una orientación similar al lenguaje *Processing*, haciendo alusión a la facilidad de uso. Dicho lenguaje posee un entorno amigable, y permite la rápida creación de aplicaciones interactivas, sin necesidad de poseer grandes conocimientos técnicos en programación. También este lenguaje está basado en *Java*, en cuanto a su sintaxis, lo que permite el despliegue de su entorno de desarrollo en diferentes sistemas operativos (Cuartas, 2014; Purdum, 2013). Los *scripts* creados con este lenguaje, son conocidos como *Sketch*, que significa “bosquejo” o “esquema”, y hace referencia a que permite generar bocetos de aplicaciones que, al ser estructuradas, pueden constituir implementaciones más grandes (Cuartas, 2014).

Firmata

En ocasiones se puede presentar el caso en que, para la creación de proyectos con *Arduino*, se deba establecer comunicación entre diferentes lenguajes, lo que puede llegar a resultar engorroso. Para estas situaciones, se presentan herramientas como *Firmata*. Esta, es una librería de *Arduino*, que permite la comunicación entre la placa y los programas escritos en un lenguaje de programación diferente al *Arduino*, a través de una conexión por cable serial. Esto permite que el equipo externo pueda enviar y recibir datos a la tarjeta, controlando los actuadores conectados a ella (Breseman y Mulder, 2014; Torrente, 2013).

Cliente/ Servidor

Cliente / Servidor es una arquitectura de red, en la que participan diferentes equipos o procesos, que pueden actuar de dos maneras: cliente o servidor. En la primera se solicitan servicios, y en la segunda, se suministran (Mora, 2002). Esta arquitectura presenta las siguientes características:

- Una parte cliente, que se encarga de la interacción con los usuarios, y una parte servidor, que mantiene interacción con los recursos y servicios.
- El cliente y el servidor poseen diferentes niveles de recursos, como almacenamiento, memoria, velocidad de procesamiento, etc.

- Los procesos de cliente y servidor pueden ser ejecutados en un mismo procesador, o en distintos.

Protocolo HTTP

Uno de los protocolos más usados en los servicios *web* es el HTTP (*Hypertext Transfer Protocol*), el cual es un protocolo de comunicaciones, que pertenece a la capa 7 (capa de aplicación) del modelo OSI (*Open System Interconnection*), y permite tareas de solicitud y respuesta (Born, 2001; Mora, 2002). El servidor desarrollado implementa este protocolo, y permite la comunicación con la placa *Arduino* del caso descrito en este documento, recibiendo solicitudes desde un computador externo, y retornando una página *web*, desde la cual se puede establecer control y registro sobre los actuadores.

Node.js

Esta tecnología primero se debe tratar desde el lenguaje *JavaScript*, el cual es un lenguaje de programación creado para el desarrollo de aplicaciones Cliente / Servidor, el cual puede ser incrustado entre etiquetas HTML, y permite obtener más dinamismo en las páginas *web* (Flanagan, 2007). *JavaScript* es un lenguaje que normalmente funciona del lado del cliente. En la actualidad, y debido a su facilidad de aprendizaje, está siendo utilizado para la creación de aplicaciones del lado del servidor, en donde comúnmente se usaban lenguajes como PHP o Ruby (Flanagan, 2007). Se puede decir que el lenguaje *JavaScript* brinda dinamismo, en forma de efectos de rápido despliegue, a las páginas *web*. Con este lenguaje se pueden llevar a cabo actividades como cambio de texto, presentación de imágenes, realización de cálculos, entre otros.

Por otro lado, *Node.js* es un intérprete que permite la creación de aplicaciones que funcionan de lado del servidor, a partir de *JavaScript*. Este hace una renovación a la forma como se venía trabajando el tema de las conexiones a los servidores, utilizando el paradigma de programación orientada a eventos (Abernethy, 2011). Una de las mayores ventajas de esta tecnología es la ejecución de tareas de manera asíncrona, lo cual disminuye el tiempo de ejecución de los procesos (Abernethy, 2011). Teniendo en cuenta que los procesos explicados en este escrito requieren de conexiones a

un determinado servidor, el uso de *Node.js* resulta de gran ayuda. Esta tecnología gestiona las conexiones a través del manejo de hilos o subprocesos, lo que disminuye la posibilidad de generar cuellos de botella, garantizando mayor disponibilidad de los servicios (Abernethy, 2011).

Caso de práctica

En esta sección se llevará a cabo una sencilla práctica, que consiste en la elaboración de un sistema capaz de obtener una orden, a partir de un entorno *web*, para llevarla a la tarjeta *Arduino*. En este caso, la orden será el encendido de un *led*, estableciendo el requerimiento desde una página *web*.

A continuación se hará una descripción con ayuda de los conceptos anteriormente descritos. En la Figura 2 se muestra una descripción del funcionamiento lógico del sistema base que ha sido utilizado, el cual se compone de un servidor HTTP y de una página *web*, que permite la comunicación con el mismo. Dicho sistema funciona empleando un servidor que está a la espera de solicitudes por parte de entes externos. Dependiendo del caso, puede desplegar una página *web* a través de la cual se puede ejercer control sobre los actuadores, o permitir el paso de órdenes hacia el *Arduino*.

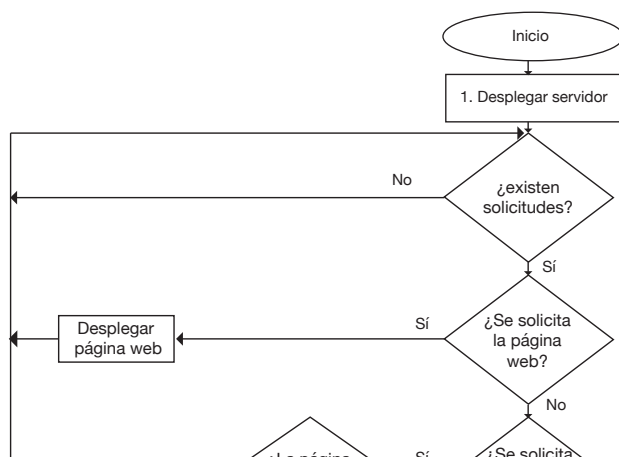


Figura 2. Diagrama de funcionamiento lógico del sistema
Fuente: elaboración propia (2017).

Para la realización de esta práctica, se utilizó un miniordenador o *Mini PC Raspberry Pi*. Este tipo de dispositivo es un tipo de ordenador, diseñado para

solucionar problemas de espacio y portabilidad, que puede ser usado en proyectos de diferentes campos (Richardson y Wallace, 2016; Varonas, 2011). Uno de los miniordenadores que combina potencia y precio, y que además se sustenta en una gran comunidad de colaboradores es el *Raspberry Pi* (Espeso, 2015). La morfología de este miniordenador, en su versión 2 - modelo B, puede ser apreciada en la Figura 3. Para su funcionamiento *Raspberry Pi* usa el sistema operativo *Linux Raspbian*, en el que se implementó un servidor *web* desde cero, lo que permitió una mayor profundización en los conceptos de servidores. De tal forma que el servidor fue instalado en el miniordenador, y se pudo acceder a él de manera remota.

Los materiales usados en esta implementación son:

- Placa *Arduino* Uno R3
- Cable de *Arduino*
- Mini Pc *Raspberry Pi*
- Cargador Micro USB
- Cable UTP, con conectores rj45 en conexión estándar.

Esta práctica consiste en enviar una sencilla orden entre la página y el servidor, que como se mencionó anteriormente, se comunican utilizando herramientas *web*. Por esta razón, se cargará el servidor HTTP en el mini ordenador *Raspberry Pi*, y se envían órdenes a este de manera remota. Todo esto para que el puerto trece (13) del *Arduino* cambie su estado de cero (0) a uno (1), lo que generará el encendido de un *led* conectado a dicho terminal. Se ha decidido emplear este *led* debido a que viene integrado en la tarjeta.

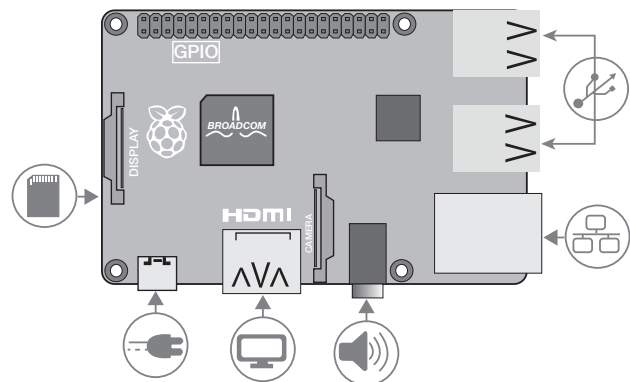


Figura. *Raspberry Pi* 2 modelo B
Fuente: elaboración propia (2017).

Es necesario tener en cuenta que un cable UTP (*Unshielded Twisted Pair*) es un tipo de cable de par trenzado, muy utilizado en redes telefónicas y Redes de Área Local (LAN), el cual permite la transmisión de información entre diferentes equipos (Vazquez, Baeza y Candelas, 2010). En la Figura 4 se puede observar el esquema eléctrico del montaje realizado para ambas prácticas. En este caso, para el uso de periféricos solo se tendrá en cuenta el *led*, y el resto de ellos, como el motor, la pantalla, entre otros, hacen referencia a la diversidad de actuadores que pueden ser usados. En ella se presenta el diagrama de conexión, el cual funciona empleando una conexión LAN, entre el *modem* y la *Raspberry*, además de una conexión serial con el *Arduino*. También se muestra un equipo desde el cual se accede a la página *web* y se ejerce control sobre los actuadores. Para esto se creó un servidor HTTP en el miniordenador *Raspberry Pi*, encargado de recibir solicitudes desde la página, ejecutar eventos y enviar órdenes a la tarjeta *Arduino*, la cual utiliza la librería *Firmata* para entender el lenguaje en el que se envían las ordenes.

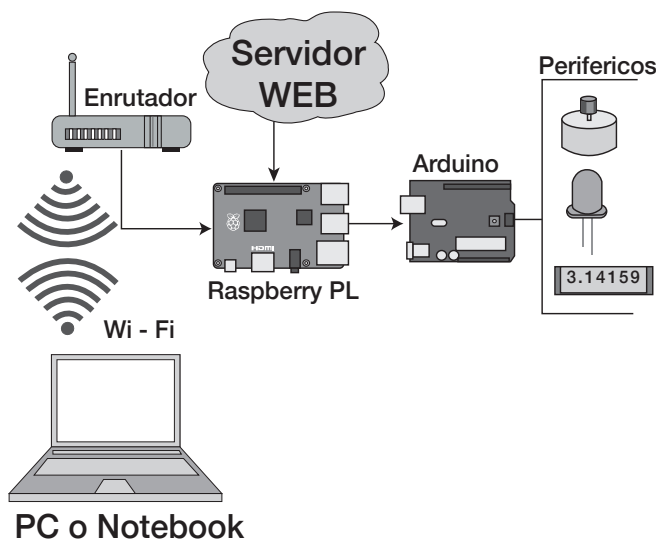


Figura 4. Esquema de la conexión entre el computador, el *Raspberry Pi* y la tarjeta *Arduino*

Fuente: elaboración propia (2017).

Los fragmentos de código 1 y 2 hacen parte del servidor y la página *web*, respectivamente, y fueron tomados como base para la construcción del sistema. Estos muestran los factores importantes para el envío y captura de datos.

Algoritmo 1: Fragmento de código de la página *web*. Fragmento tomado de Danasf (2014, secc. 2)

```
$( '#ledOn' ).on( 'click', function() {
  socket.emit( 'encender' );
} );
```

Algoritmo 2: Fragmento de código del servidor. Fragmento tomado de Danasf (2014, secc. 2)

```
socket.on( 'encender', function( data ) {
  console.log( " Encendiendo led" );
  if( board.isReady ) { led.on(); }
} );
```

Cabe resaltar que la línea dos (2) del código 1 permite emitir la orden de encender el *led*, mientras que la línea número tres (3) del código 2 es la que se encarga de verificar si la placa *Arduino*, citada como *board*, esta lista, y luego envía la orden de encendido. El uso de este tipo de palabras, como *board*, es posible a través del uso de una librería de *JavaScript*, conocida como *Jhonny-Five*, que permite la programación de *hardware*, a través de código escrito en un lenguaje de alto nivel (Media, Waldron y Szymczykowski, 2015). También fue necesario utilizar una librería de *JavaScript*, *Socket.io*, que permite la comunicación entre las entidades cliente y servidor, utilizadas en esta práctica (Andrears, 2014).

Práctica propuesta

Teniendo en cuenta todos los conceptos mencionados anteriormente, y la práctica en la cual se han utilizado dichos conceptos, se plantea la siguiente actividad como un ejercicio adecuado para la ejercitación de dichos conceptos.

La práctica consiste en la construcción de una casa inteligente sencilla, utilizando la tarjeta *Arduino* como elemento que ejerce control sobre los diferentes actuadores, y una tarjeta *Raspberry Pi*, que se encargue de ejecutar el servidor, el cual permite el paso de ordenes hacia la parte de control. También se necesitará un *modem* para permitir que el miniordenador acceda a la red, como se muestra en la Figura 4. Por último, se debe tener en cuenta que la página *web*, a través del cual el usuario envía las ordenes al PC, puede ser accedido a través de cualquier equipo que tenga acceso a un navegador, y que se encuentre ligado a la misma red que el miniordenador. Recordemos que un actuador es cualquier dispo-

sitivo que le permite al *Arduino* tener comunicación con el mundo externo, y que la iniciativa *Arduino* promueve el uso de tarjetas, o *shields*, que le permiten un adecuado control de los mismos. Los elementos necesarios para la elaboración de este ejercicio serían los siguientes:

- *Arduino mega*
- *Raspberry Pi*
- *Modem ethernet*
- Motores paso a paso
- *Motorshield*
- Bombillos led
- Zumbadores
- Sensores ultrasónicos
- Sensores de temperatura
- Sensores de humedad
- Sensores de CO₂
- *Display lcd 16 x 2*
- *Arduino compatible Active Speaker Buzzer Module*

A continuación, se presenta la descripción de algunos de los elementos que no habían sido tratados en el escrito. Dichos elementos, son los siguientes:

- ***Motorshield***: es un dispositivo de la iniciativa *Arduino*, que permite el control de motores sin comprometer el resto del sistema, debido a las peticiones de corriente realizadas por estos actuadores. Para llevar a cabo esta función, el equipo toma una señal, y permitiendo el paso de corriente de una fuente externa, la redirige hacia el motor. Este dispositivo permite el control de motores paso a paso, y motores de corriente continua facilitando el manejo de su velocidad y dirección (Loi y Goilav, 2016).
- ***Arduino Compatible active speaker buzzer module***: es un módulo de *Arduino* que permite generar sonidos, como un zumbador. Esto, junto con las luces, permitirá generar alertas (Loi y Goilav, 2016; Torrente, 2013)
- ***Display lcd 16x2***: es una pantalla plana, que posee un arreglo bidimensional, formada por un determinado número de píxeles. Esta será utilizada para mostrar mensajes de operación de cada uno de los dispositivos (Loi y Goilav, 2016).

Por último, es necesario establecer las posibles posiciones para los elementos utilizados. El posicionamiento puede observarse a continuación:

- En cada una de las habitaciones se recomienda ubicar sensores ultrasónicos para controlar el encendido de la luz. La idea es que los sensores permitirán identificar si hay personas presentes en la habitación, o pasillo, si se considera.
- En todas las habitaciones, incluyendo la cocina, se posicionarán sensores de temperatura, con el fin de saber si se presenta alguna precipitación o incendio.
- Tras el portón de la entrada se ubicarán motores paso a paso, los cuales permitirán el cerrado a distancia.
- Los bombillos *led* serán utilizados para construir arreglos que permitan iluminar los pasillos, sobre los cuales no se puede tener una buena visibilidad.
- Los sensores de CO₂ serán ubicados en las áreas de estudio de la casa.
- Los sensores de humedad serán posicionados en los baños, y las habitaciones donde se lava ropa. Estos nos permitirán saber cuándo se presentan condiciones que promueven la formación de moho.

Se debe tener en cuenta que el CO₂, dióxido de carbono, es un agente que, bajo ciertas concentraciones, puede generar algunos inconvenientes para el ser humano. Por esta razón, es recomendable mantener un monitoreo constante sobre la calidad del aire. Se debe tener en cuenta que, dado la naturaleza de este gas, el ser humano puede sentirlo cuando se presentan alteraciones negativas sobre los mismos. Entre dicha alteraciones se encuentran: malestar y una posible disminución en la concentración. Todo esto sucede cuando se sobrepasa un cierto porcentaje fijado por el programador.

Una vez estén posicionados todos los elementos, se iniciará con el proceso de conexión, a nivel de *hardware*, y posteriores pruebas. Para esto se recomienda llevar a cabo los siguientes pasos:

1. Conecte todos los actuadores a la tarjeta *Arduino Mega*.
2. Elabore un programa, sobre la plataforma *Arduino*, que le permita conocer el estado de cada actua-

dor, algo similar a una rutina de diagnóstico. Esto le servirá para identificar si todo está funcionando de manera adecuada.

3. Utilice el USB para evidenciar el funcionamiento de los actuadores.

Como se puede observar, el sistema se construye de manera modular. Por esto, se le recomienda, que, como siguiente paso, realice la programación del servidor en el miniordenador *Raspberry Pi*. Para esto siga los siguientes pasos:

1. Conecte el Arduino al miniordenador.
2. Escriba el servidor.
3. Introduzca el código de *Firmata* sobre el *Arduino*.
4. Realice pruebas de conexión y envío de datos entre el *Arduino* y la *Raspberry*.

En este punto ya es posible hacer uso de los actuadores distribuidos en las operaciones de automatización de la casa, pero es necesario construir una interfaz amigable con el usuario. Por lo cual se procederá a la construcción de la página *web*, para lo cual se recomienda hacer lo siguiente:

1. Construya el sitio *web*, utilizando *JavaScript* y *HTML*.
2. Configure el servidor de tal forma que este pueda responder a las peticiones *http* cuando se solicite la página *web*.
3. Despliegue las pruebas sobre todo el sistema.

Por último, es posible posicionar el *Display LCD* y los zumbadores en un lugar estratégico, con el fin de notificar acerca de los procesos de los actuadores. El primero mostrará los mensajes de operación, y el segundo alertará cuando se presenten precipitaciones. Recuerde que la *Raspberry* debe ir conectada al *Modem Ethernet*, y solo aquellos equipos que estén vinculado a la red, generada por este último, podrán acceder a la página *web*.

Conclusión

El uso de miniordenadores, como la tarjeta *Raspberry Pi*, y las tarjetas de circuito programable, como *Arduino*, permiten una ágil creación de proyectos en los cuales se requiere implementar *software* y

hardware. Estos equipos brindan portabilidad y flexibilidad a proyectos de diferente naturaleza. Por este motivo, el tamaño de los proyectos realizados con los conceptos descritos en este proyecto depende de la imaginación del lector. Aunque la práctica presentada en este artículo representa una implementación sencilla, se constituye en una alternativa para la generación de proyectos en los que se requiere realizar comunicación entre *software* y *hardware*, los cuales pueden ser implementados para el manejo de diferentes actuadores, como: luces, motores, pantallas, entre otros.

Los conceptos presentados tienen amplias aplicaciones en domótica o automatización industrial, debido a que pueden ser aplicados al monitoreo y control remoto de sistemas robustos mediante interfaces adecuados para cada aplicación, teniendo en cuenta que una de las ideas fundamentales de este escrito es el establecimiento de la comunicación a través de un medio flexible y eficiente, como la *web*.

Referencias bibliográficas

- Abernethy, M. (2011). *¿simplemente qué es node.js? un servidor listo para codificar*. IBM developer Works. California.
- Andrearrs, S. (2014). Socket.io: Comunicación bidireccional en tiempo real para javascript. *hipertextual*. Consultado el 10 de Junio de 2018. Recuperado de: <https://hipertextual.com/archivo/2014/08/socketio-javascript/>
- Anderson, K. (2017). *Picaxe Project Handbook: A Guide to using Picaxe Microcontrollers V1.Pt1*. (Volume One). Publicado independientemente.
- Agus, K. (2014). *MSP430 LaunchPad Programming*. PE Press. Boston.
- Breseman, K. y Mulder, P. (2016). *Node.js for embedded systems: using web technologies to build connected devices*. O'Reilly Media. California.
- Born, G. (2001). *Kompendium HTML: con XHTML, DHTML, CSS, XML, XSL y WML*. Markt + Technik. Berlin.
- Cuartas, J. (2014). *Digitopolis 1: Diseño de aplicaciones interactivas para creativos y comunicadores*. Fundación Universitaria Los Libertadores. Bogotá.

- Danasf, A. (2014). *Javascript robotics and browser-based arduino control based-Arduino-cont/?ALLSTEPS*. Consultado el 10 de Junio de 2018. Recuperado de <http://www.instructables.com/id/Javascript-robotics-and-browser-based-Arduino-cont/>
- E. D. Negocios (2009). *La oportunidad del software libre: capacidades, derechos e innovacion*. Recuperado de <https://tiscar.com/2009/07/27/la-oportunidad-del-software-libre-capacidades-derechos-e-innovacion/>
- Espeso, P (2015). Nueva Raspberry PI vs sus competidores: buscando el mejor mini ordenador barato. *Xataka*. Consultado el 10 de Junio de 2018. Recuperado de <https://www.xataka.com/componentes/nueva-raspberry-pi-vs-sus-competidores-buscando-el-mejor-mini-ordenador-barato>
- Flanagan, D. (2007). *JavaScript: la guia definitiva*. Editorial Anaya Multimedia. México.
- Jordisan, S. (2014). ¿ Que son los frameworks de desarrollo y por que usarlos?. Recuperado de <https://jordisan.net/blog/2006/que-es-un-framework/>
- Kurniawan, A. (2004). *Getting Started with pcDuino3*. PE Press. Boston.
- Loi, G. y Goilav, N. (2016). *Arduino: Aprender a desarrollar para crear objetos inteligentes*. Ediciones Eni. España. Recuperado de <https://www.ediciones-eni.com/libro/arduino-aprender-a-desarrollar-para-crear-objetos-inteligentes-9782409000447>
- Media, B., Waldron, R. y Szymczykowski, P. (2015). *JavaScript robotics: building nodebots with Johnny-Five, Raspberry Pi, Arduino, and BeagleBone*. Maker Media, Inc. Boston.
- Mora, S. (2002). *Programacion de aplicaciones web: historia, principios basicos y clientes web*. Editorial Club Universitario. Alicante.
- Purdum, J. (2013). *Beginning C for Arduino: learn C Programming for the Arduino*. Apress. Ohio.
- Reas, C. y Fry, B. (2015). *Processing: a programming handbook for visual designers and artist*. MIT Press. Massachusetts.
- Richardson, M. (2014). *Make: getting started with Intel Galileo: electronic projects with the Quark-Powered Arduino-Compatible Board*. Maker Media Inc. Boston, United States.
- Richardson, M. y Wallace, S. (2016). *Getting started with Raspberry PI: an introduction to the fastest-selling computer in the world*. Maker Media Inc. Boston.
- Torres, J., Vila-Viña, D. y Lazalde, A. (2015). *Hardware libre. Recomendaciones para el fenomeno de innovacion ciudadana*. Ministerio Coordinador del Conocimiento y el Talento Humano. Recuperado de <http://floksoctety.org/docs/Espanol/4/4.1.pdf>