



Universidad
de La Laguna

Sistema de vigilancia de la calidad ambiental para entornos de trabajo

Monitoring system of environmental quality in working environments

Sergio Medina Toledo

Dpto. Ingeniería informática y de sistemas

Escuela Técnica Superior de Ingeniería Informática

Trabajo de Fin de Grado

La Laguna, 9 de julio de 2014

D. **Jesus Miguel Torres Jorge**, con N.I.F. 43.826.207-Y profesor Titular de Universidad adscrito al Departamento de Ingeniería informática y de sistemas de la Universidad de La Laguna

C E R T I F I C A

Que la presente memoria titulada:

“Sistema de vigilancia de la calidad ambiental para entornos de trabajo.”

ha sido realizada bajo su dirección por D. **Sergio Medina Toledo**, con N.I.F. 45.898.780-A.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 9 de julio de 2014



Agradecimientos

A Jesus Miguel Torres Jorge por haber sido mi tutor de proyecto y haberme ayudado en todo lo posible.
A Pedro González Yanes por haberle propuesto a Jesus este proyecto tan completo e interesante.
A mi familia por estar siempre ahí apoyándome.
Y por ultimo a todo el profesorado que me ha enseñado durante estos 4 años.

Resumen

En la actualidad gracias al abaratamiento de los microcontroladores, SBC y muchas otras tecnologías (GPS, sensores, pantallas ..) han empezado a proliferar distintos dispositivos que dotan de "inteligencia" a objetos y los conectan a internet, el llamado Internet de las Cosas, lo que en este proyecto se plantea es la creación de uno de estos dispositivos que se encargue de monitorizar la calidad ambiental de entornos de trabajo, mediante la medición de ruidos en habitaciones de trabajo y a la vez actuando en base a las mediciones para disminuir el ruido.

Palabras clave: Microcontrolador, SBC, Internet de las Cosas, Calidad ambiental, Ruido

Abstract

Today thanks to cheaper microcontrollers, SBC and many other technologies (GPS, sensors, displays ..) have begun to proliferate different devices that provide "intelligence" to objects and connect it to the Internet, called Internet of Things , which is posed in this project is the creation of one of these devices to be in charge of monitoring the environmental quality in work environments, by measuring noise in workrooms and acting based on measurements for noise reduction.

Keywords: *Microcontroler, SBC, Internet of Things, Environmental quality, Noise*

Índice general

1. Introducción	1
2. Antecedentes y estado actual del tema	2
2.1. Inteligencia ambiental	2
2.2. Internet de las cosas	2
2.3. Sistemas similares	3
2.4. Single-Board Computers (SBC)	3
3. El Hardware	5
3.1. Electric imp	5
3.2. APRIL developement board	8
3.3. Los Microfonos	8
3.4. El Zumbador	9
3.5. La Batería	10
3.6. Regulador de Voltaje	10
3.7. La Barra de Leds, Led RGB	11
3.8. Esquema de conexiones	12
3.9. Problemas Encontrados	13
4. El Software	15
4.1. Plataforma electric imp	15
4.2. El imp	15
4.3. El agente	16
4.4. El Servicio Web	16
4.5. Control Panel	16
4.6. API	18
4.7. Problemas Encontrados	18
5. Conclusiones y trabajos futuros	19
6. Summary and Conclusions	21
7. Presupuesto	23
7.1. Tabla de Presupuesto y Coste Total	23

A. Código del imp	25
B. Código Agente	28
C. Repositorio de código del Control Panel	30
Bibliografía	30

Índice de figuras

3.1. Sistema del imp	6
3.2. Electric Imp	7
3.3. IDE del imp y el agente	7
3.4. Intrucciones de Blinkup	8
3.5. APRIL developement board	8
3.6. Microfono	9
3.7. Zumbador piezoelectrico	9
3.8. Bateria LiPo 2500mAh	10
3.9. Cargador de Bateria LiPo	10
3.10. Regulador de Voltaje	11
3.11. Barra de Leds	11
3.12. Led RGB	12
3.13. Esquema de conexiones	13
3.14. Cortocircuito en micrófono para desactivar AGC	14
4.1. Formula de calculo de decibelios	16
4.2. Panel de Control	17
4.3. Grafica de db en tiempo real	17
4.4. Grafica de db en un intervalo de tiempo definido por el usuario	18

Índice de tablas

7.1. Tabla de Presupuesto y Coste Total	23
---	----

Capítulo 1

Introducción

En bibliotecas y otros entornos de trabajo es cada vez más común el uso de dispositivos destinados a monitorizar las condiciones ambientales; especialmente: temperatura, humedad, ruido, etc. Esto permite automatizar acciones destinadas a mejorar dichas condiciones o establecer alertas destinadas al personal responsable. El principal objetivo de este proyecto es desarrollar uno de esos sistemas de vigilancia utilizando SBC, productos destinados al mercado del Internet de las cosas y otros elementos de bajo coste nacidos al calor de los movimientos MAKER y DIY. Más concretamente un sistema de monitorización de calidad ambiental mediante la medición de ruidos en habitaciones de trabajo y a la vez actuando en base a las mediciones para disminuir el ruido y avisar a los responsables de la sala.

Capítulo 2

Antecedentes y estado actual del tema

2.1. Inteligencia ambiental

La inteligencia ambiental se refiere a entornos electrónicos que son sensibles y responden a la presencia de personas. La inteligencia ambiental es una visión de el futuro de la electrónica de consumo, las telecomunicaciones y la informática que se desarrolló originalmente a finales de 1990 para el marco temporal 2010-2020. En un mundo de la inteligencia ambiental, los dispositivos funcionan en conjunto para apoyar a las personas en la realización de sus actividades de la vida diaria y sus tareas de una manera fácil y natural utilizando la información e inteligencia que se oculta en la red que conecta estos dispositivos (véase la Internet de las cosas). A medida que estos dispositivos cada vez son más pequeños, están más conectados y más integrados en nuestro entorno, la tecnología desaparece de nuestro entorno hasta que sólo la interfaz de usuario sigue siendo perceptible por los usuarios. El paradigma de la inteligencia ambiental se basa en la computación ubicua, perfilado, sensibilidad al contexto, y el diseño de interacción ordenador-humano.

2.2. Internet de las cosas

El concepto de Internet de las cosas fue propuesto por Kevin Ashton en el Auto-ID Center del MIT en 1999, donde se realizaban investigaciones en el campo de la identificación por radiofrecuencia en red (RFID) y tecnologías de sensores.

Por ejemplo, si los libros, termostatos, refrigeradores, la paquetería, lámparas, botiquines, partes automotrices, etc. estuvieran conectados a Internet y equipados con dispositivos de identificación, no existirían, en teoría, cosas fuera de stock o medicinas faltantes o caducadas, sabríamos exactamente la ubicación y cómo se consumen y compran productos en todo el mundo; el extravío sería cosa del pasado y sabríamos qué está encendido o apagado en todo momento.

El Internet de las cosas debería codificar de 50 a 100.000 millones de objetos y seguir el movimiento de estos; se calcula que todo ser humano está rodeado de por lo menos 1.000

a 5.000 objetos. Según la empresa Gartner, en 2020 habrá en el mundo aproximadamente 26 mil millones de dispositivos con un sistema de adaptación al Internet de las cosas. Abi Research, por otro lado, asegura que para el mismo año existirán 30 mil millones de dispositivos inalámbricos conectados al Internet. Con la próxima generación de aplicaciones de Internet (protocolo IPv6) se podrían identificar todos los objetos, algo que no se puede hacer con IPv4, el sistema actualmente en uso. Este sistema sería capaz de identificar instantáneamente por medio de un código a cualquier tipo de objeto.

La empresa estadounidense Cisco, que está desarrollando en gran medida la iniciativa del Internet de las cosas, ha creado un "contador de conexiones" dinámico que le permite estimar el número de cosas conectadas desde julio de 2013 hasta el 2020. El concepto de que los dispositivos se conectan a la red a través de señales de radio de baja potencia es el campo de estudio más activo del Internet de las Cosas. Este hecho se explica porque las señales de este tipo no necesitan ni Wi-Fi ni Bluetooth. Sin embargo, se están investigando distintas alternativas que necesitan menos energía y que resultan más baratas, bajo el nombre de "Chirp Networks".

Actualmente, el término Internet de las cosas se usa con una denotación de conexión avanzada de dispositivos, sistemas y servicios que va más allá del tradicional M2M (máquina a máquina) y cubre una amplia variedad de protocolos, dominios y aplicaciones.

2.3. Sistemas similares

Existe un sistema similar al planteado en esta memoria el SoundEar, este sistema advierte a los trabajadores a través de señales luminosas con leds, es configurable un nivel máximo de decibelios, usa un cable de alimentación, el sistema de logeo de ruido se vende aparte y no reporta en tiempo real el ruido si no que lo guarda en una memoria y luego hay que pasarlo al PC con un cable para analizar los datos de mediciones posteriormente. Su precio asciende con el módulo de logeo de ruido a 705 euros, como comprobaremos más adelante el sistema planteado en esta memoria es mucho más barato y tiene más funcionalidades que el anteriormente mencionado.

2.4. Single-Board Computers (SBC)

La proliferación en el mercado de pequeños ordenadores de una única placa (Single-Board Computers o SBC), capaces de ofrecer la potencia de PCs de hace 10 o 15 años con una fracción de su consumo y por unas decenas de euros, ha facilitado la aparición de todo tipo de proyectos que buscan la integración de la informática en el entorno de las personas (IoT). Sistemas de sensado, registro y/o monitorización de entornos físicos o aparatos tales como cámaras IP, que hoy tienen un coste muy alto, pueden ser reemplazados por este tipo de ordenadores, por mucho menos dinero y con mucha más potencia de cálculo, lo que nos ofrece nuevas posibilidades.

Tomar medidas de ruido, humedad, temperatura o gases, grabar imágenes o detectar presencia y accionar alarmas, indicadores, puertas o controlar sistemas de aire acondicionado, son algunas de las cosas que SBCs como Arduino, RaspberryPi, Beagleboard, etc.

pueden hacer de forma autónoma o gestionada de manera centralizada.

Las SBC que se plantearon usar en este proyecto fueron :

Arduino : Esta formado por un microcontrolador Atmel AVR de la familia ATmega, dependiendo del modelo tiene más o menos puertos y es una plataforma de software y hardware libre. Arduino es programable a través de un IDE propio y se programa en C o en un lenguaje propio basado en Processing. De forma nativa no soporta comunicaciones ethernet o wifi aunque existen módulos para proporcionar esta características a Arduino. Al ser una plataforma de software y hardware libre tiene una gran comunidad y existe una gran cantidad de documentación y librerías para todo tipo de dispositivos que se pueden conectar a Arduino.

Raspberry pi : Es una SBC basada en el SOC Broadcom BCM2835 que contiene una CPU ARM1176JZF-S a 700 MHz un procesador gráfico (GPU) VideoCore IV, y 512 MB de memoria RAM, usa una tarjeta SD para el almacenamiento permanente, tiene 2 puertos USB, una salida de vídeo HDMI otra RCA (PAL y NTSC), un puerto ethernet, 24 puertos de entrada salida (GPIO) y Linux como SO. La Raspberry pi se puede programar con casi cualquier lenguaje mientras exista compilador o interprete para Linux.

Electric Imp : Es un microcontrolador con conectividad wifi encapsulado en una tarjeta SD con 6 puertos de entrada salida, tiene nube propia desde donde se programa a través de un IDE web, se usa squirrel como lenguaje de programación, desde la misma nube se realizan todas las comunicaciones con el imp mediante un .objeto” programable al que llaman agente. Es una plataforma cerrada y propietaria de Electric Imp, Inc.

Capítulo 3

El Hardware

Para desarrollar este proyecto era necesario cierto hardware que se encargara de el sensado de el ruido por lo que se barajó entre los siguientes SBC, Arduino, Raspberry pi y Electric imp. Al final se escogió el Electric imp ya que es el que más se ajusta a las necesidades del proyecto, porque tiene wifi y los puertos de entrada salida son suficientes para el proyecto.

Ya teniendo como elección el imp pasaremos a profundizar en él y los sensores y dispositivos conectados al mismo.

3.1. Electric imp

El electric imp es un microcontrolador propiedad de Electric Imp, Inc encapsulado en un formato de tarjeta SD con 6 puertos de entrada salida y conectividad wifi. El imp se licencia de dos formas, como kit de desarrollo, el cual es el usado en este proyecto y te permite usar un imp y una cuenta en la nube, está enfocada a pruebas, hobbies o makers, o como modelo comercial enfocado en usar el imp en algún producto comercial con soporte por parte de la empresa y precio especial por unidades (para este modelo hay que contactar directamente con Electric Imp, Inc).

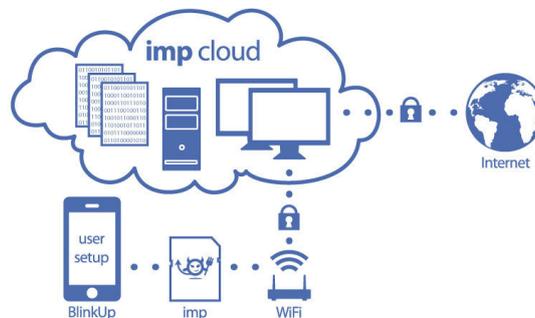


Figura 3.1: Sistema del imp

El modo de funcionamiento y desarrollo del imp es bastante novedoso ya que no se conecta al pc para programarlo ni para configurarlo, tampoco se hace a través de LAN por wifi, se configura (únicamente la wifi) usando una tecnología propietaria de Electric Imp Inc llamada BlinkUp, la cual consiste en que a través de un sensor de luminosidad que tiene el imp en la parte superior de la tarjeta SD y una App móvil donde pones el nombre de la wifi y su contraseña se manda la información introducida en la App por medio de parpadeos de la pantalla del smartphone pegada al sensor de luminosidad mencionado anteriormente, con esto conseguimos que el imp se conecte a su nube, esta conexión la realiza a través de un túnel TLS. En la nube del imp tenemos varias herramientas entre ellas esta un IDE con el que podemos programar el imp remotamente desde cualquier parte y cuando el imp se conecte se descarga automáticamente el programa y empieza a ejecutarlo. También tenemos en la nube el Agente que viene a ser el programa que esta en la nube y que se encarga de procesar toda la información que el imp le manda y a la vez es capaz de mandar información al imp, el agente viene a ser el puente de comunicaciones entre el imp y el exterior, el agente también se puede programar desde el mismo IDE que el imp. El agente pone a disposición de los usuarios una API que se basa en HTTP y que el usuario ha de usar para crear su propia API de comunicación con el Imp.



Figura 3.2: Electric Imp



Figura 3.5: APRIL development board

3.3. Los Micrófonos

Para que el imp pueda sentir el sonido hacen falta micrófonos por lo que se escogieron dos micrófonos omnidireccionales con amplificador y AGC (Automatic Gain Control) incluidos de adafruit (Link Micrófono), el micrófono, en base al pin GAIN, tiene tres modos de ganancia 40db en voltaje, 50db en tierra y 60db al aire, en este proyecto se ha usado en el modo de 60db para tener la mayor ganancia, y para tener mediciones reales se tuvo que desactivar el AGC haciendo un corto entre dos patas del chip integrado en la placa del micrófono.

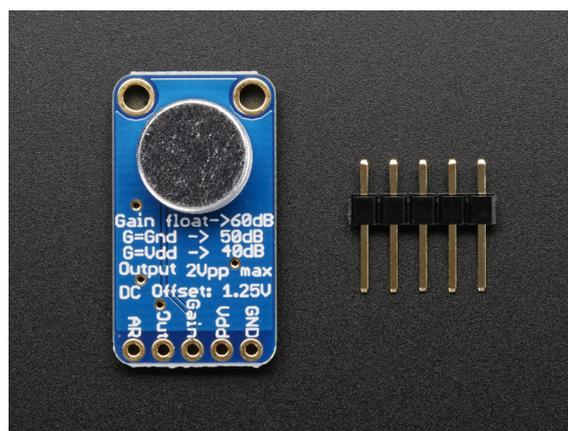


Figura 3.6: Micrófono

3.4. El Zumbador

Para que el imp pueda "mandar a callar" a las personas en la sala hace falta un zumbador que emita un pitido en señal de que se están sobrepasando los decibelios permitidos, para ello se escogió el siguiente (Link Zumbador) zumbador piezoeléctrico el cual es capaz de emitir sonidos bastante altos, agudos y molestos por lo que cumple bastante bien su función.

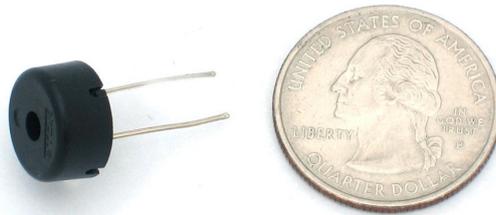


Figura 3.7: Zumbador piezoeléctrico

3.5. La Batería

Uno de los requisitos principales del proyecto era que fuera como un reloj literalmente, que se colgara en la pared sin uso de cables para alimentación o comunicación por lo que hace falta una batería, pero no una batería cualquiera ya que tiene que durar bastante ya que es un incordio tener que quitar el aparato de la pared para sacar la batería todos los días, cargarla y poner el aparato de nuevo en la pared, por lo que se escogió una batería de LiPo de 2500mAh (link batería) también hace falta un cargador para la batería por lo que se escogió un cargador usb bastante sencillo y pequeño (link cargador)



Figura 3.8: Batería LiPo 2500mAh

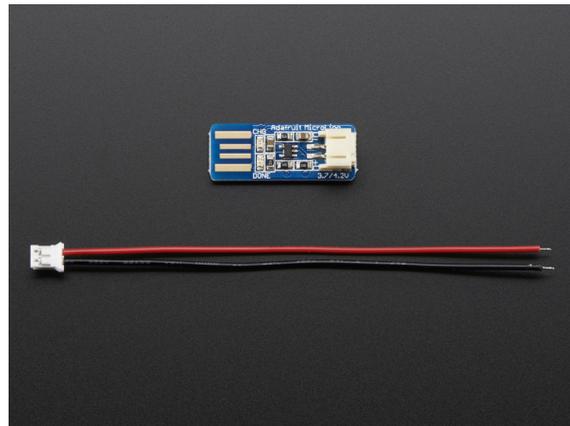


Figura 3.9: Cargador de Batería LiPo

3.6. Regulador de Voltaje

Al ser la batería de 3.7v y el imp usar 3.3v hace falta un regulador de voltaje por lo que se escogió el siguiente (link regulador)



Figura 3.10: Regulador de Voltaje

3.7. La Barra de Leds, Led RGB

A la hora de mostrar el nivel de decibelios a los trabajadores de un sala hace falta algún método que sea fácilmente entendible por lo que se escogió una barra de leds que representaría una barra de decibelios, la barra de leds se controla por i2c y no existe driver para el imp, por lo que se tuvo que crear el driver para la barra de leds, pero la barra de leds no se encendía por lo que se tuvo que mirar con el osciloscopio si se estaba realizando correctamente la comunicación a través de i2c y al principio se hacía mal pero se corrigió y aun así no se encendía la barra de leds, por lo que se decidió medir los voltajes en las patas de la barra de led controlado por el chip con i2c y habían voltajes pero no se encendían los leds, por esto y porque la barra de leds consumía mucha batería sin siquiera encender los leds se decidió usar un led RGB como aviso para los trabajadores.

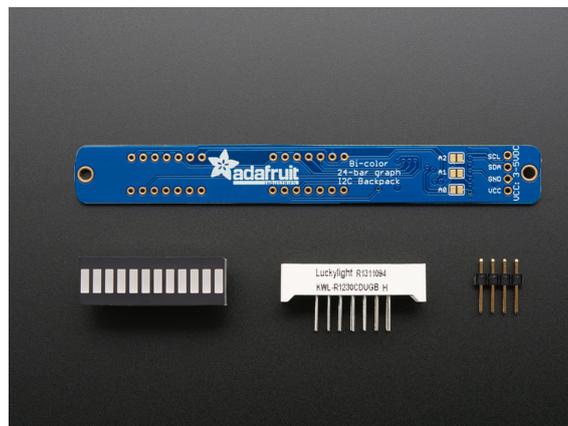


Figura 3.11: Barra de Leds

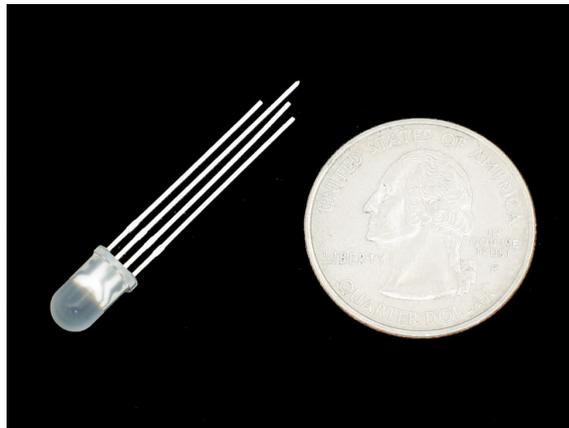


Figura 3.12: Led RGB

3.8. Esquema de conexiones

En el siguiente esquema podemos ver como se interconecta todo el hardware con el imp.

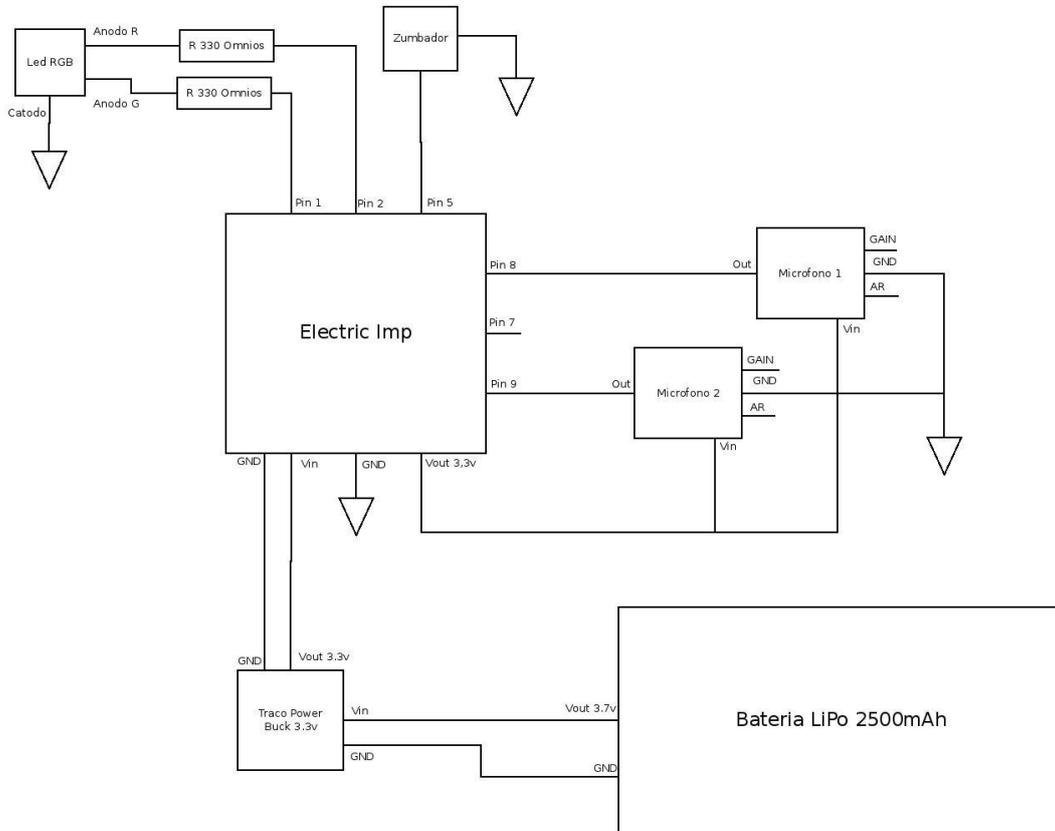


Figura 3.13: Esquema de conexiones

3.9. Problemas Encontrados

El principal problema encontrado a la hora de usar el hardware fue el AGC de los micrófonos que al tenerlo activado las mediciones de decibelios eran incorrectas y tuvo que hacer un corto en el chip para desactivarlo. Y por otro lado la documentación del imp al no ser muy clara en el aspecto del i2c fue difícil desarrollar el driver para la barra de leds así como que aun estando funcionando el protocolo i2c la barra de leds no funcionó.

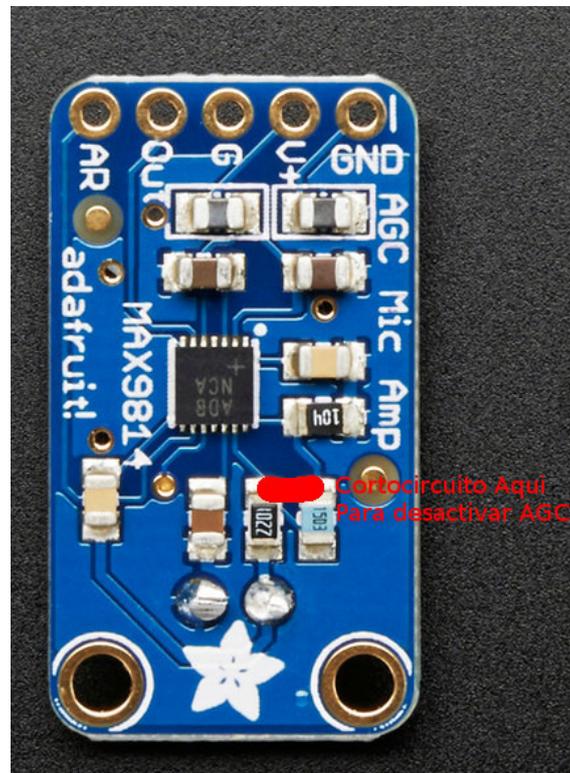


Figura 3.14: Cortocircuito en micrófono para desactivar AGC

Capítulo 4

El Software

El software en el que se apoya este sistema de sensado de sonido está dividido fundamentalmente en dos partes, la plataforma imp y el servicio web que almacena y muestra los datos por lo que en los próximos apartados profundizaremos en estas dos partes empezando por el imp.

4.1. Plataforma electric imp

La plataforma imp tiene dos partes software, el software que corre en el imp y el software que corre en la nube del imp y que se encarga de la comunicación entre el imp y el exterior. La plataforma imp se programa en un lenguaje llamado squirrel con programación orientada a eventos, la propia plataforma te facilita una API para interactuar, en la parte del imp, con el agente y con los pines del imp y en la parte del agente para interactuar con el imp y para comunicarse mediante http con el exterior.

4.2. El imp

El software que he creado para el imp es bastante sencillo, empieza configurando el imp en modo powersave de la interfaz wifi, hace las conexiones más lentas pero aumenta la duración de la batería considerablemente, seguidamente manda la MAC del imp al agente y le pide al agente la configuración, luego empieza un bucle infinito que recibe la configuración desde el agente y comienza un periodo de muestreo de sonido por los micrófonos, este periodo es configurable desde el servicio web, una vez hecho el muestreo calcula el Voltaje rms y con este voltaje calcula la media de decibelios (mediante la formula 1) que se han medido durante el periodo, por ello se recomienda que el periodo de muestreo no sea muy grande ni muy pequeño, una vez calculados los decibelios los manda al agente y si los decibelios son mayores que los permitidos y el imp está activado activa el zumbador y el led RGB en rojo, en caso de estar en el intervalo de decibelios medio enciende el led RGB en naranja, después entra en modo sleep durante un periodo de tiempo configurable y al salir de este modo ejecuta de nuevo el bucle.

$$L_p = 20 \log_{10} \left(\frac{p_{\text{rms}}}{p_{\text{ref}}} \right) \text{ dB}$$

Figura 4.1: Formula de calculo de decibelios

Uno de los aspectos más importantes del proyecto era la duración de la batería y en las primeras versiones del software del imp la batería duraba 33 horas lo que era muy poco por lo que toco optimizar el código y modo de funcionamiento del imp, poner la interfaz wifi del imp en modo ahorro de energía y que cada vez que el imp se quedaba sin eventos que procesar mandarlo a dormir. Con estas optimizaciones llega a durar la batería 52 horas.

4.3. El agente

El software que he creado para el agente no es más que un puente con el exterior que recibe los datos de sensado del imp y se los manda al servicio web junto con la mac del imp como identificador en una petición POST. La otra acción que realiza el agente es que cuando el imp pide la configuración el agente hace una petición GET al servicio web para recuperar la configuración y se la envía al imp.

4.4. El Servicio Web

En el lado del servicio web decidí usar Django para el panel de control , django-rest-framework para el API, postgresql como base de datos, nginx como servidor web, gunicorn como servidor WSGI HTTP y supervisord como controlador de procesos de gunicorn.

4.5. Control Panel

Para hacer el Control Panel modifique el admin de Django para que funcionara como tal, desde el panel de control se pueden añadir habitaciones donde se va a monitorizar el sonido e imps que van a monitorizar el sonido en habitaciones así como la configuración del imp donde puedes modificar los niveles de decibelios permitidos, el tiempo que se pasa el imp muestreando y el que se pasa durmiendo así como si el imp está activado (suena el zumbador si supera los decibelios permitidos). También crea una gráfica por habitación de los decibelios medidos en tiempo real o en el periodo de tiempo que le indiques, para así poder a llegar a ver un día entero y saber cuales son las horas en las cuales más ruido se produce. Para hacer las gráficas se uso Flot y Ajax en conjunto con el API que se explica en el siguiente apartado.

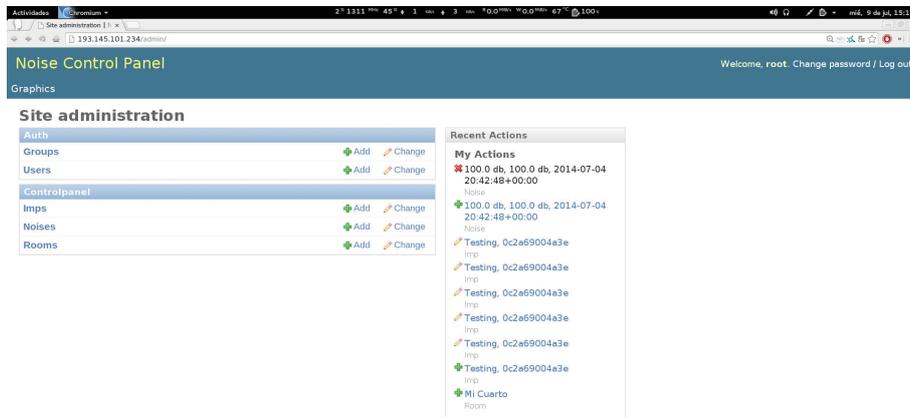


Figura 4.2: Panel de Control

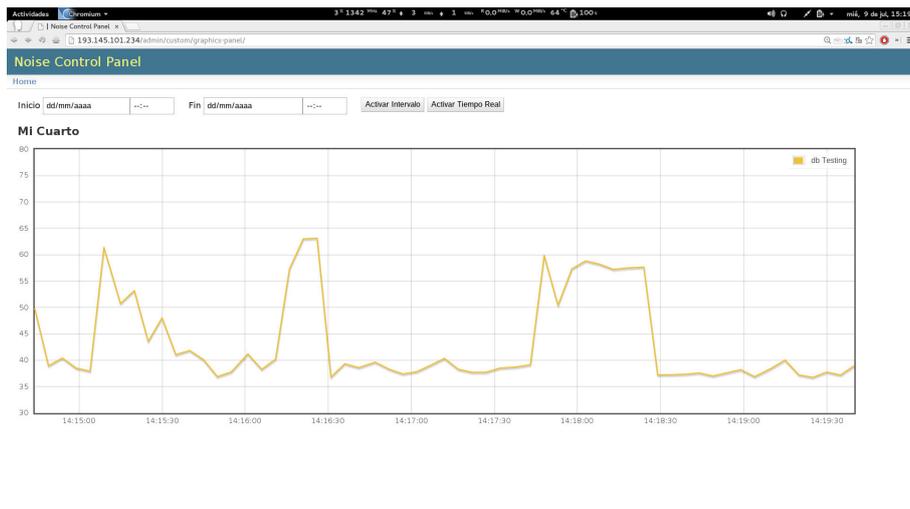


Figura 4.3: Gráfica de db en tiempo real

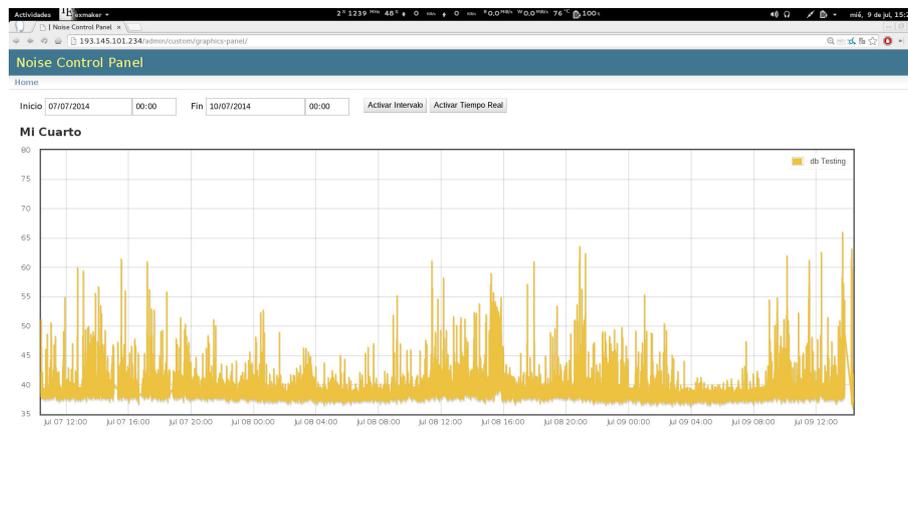


Figura 4.4: Gráfica de db en un intervalo de tiempo definido por el usuario

4.6. API

La API que he creado es una api REST y lo que permite básicamente es recibir datos de mediciones de los imps, recoger los datos que los imps han mandado (Esta funcionalidad del API la usan las gráficas con ajax para luego pintar los datos). Por otro lado permite a los imp consultar su configuración y también permite modificar las configuraciones de los imp, esta feature es para poder modificar las configuraciones desde algún script o aplicación externa fácilmente, por ejemplo desactivar el imp en ciertas horas que si se permite hacer más ruido.

4.7. Problemas Encontrados

El principal problema encontrado en el desarrollo del software ha sido la falta de documentación del imp al ser una plataforma cerrada solo documenta el personal de la empresa y hay muchos aspectos poco documentados y a la vez al se una plataforma nueva hay poca comunidad y es difícil que la comunidad te brinde ayuda , ya que están tan perdidos o mas que tu. El otro problema ha sido la medición de decibelios ya que es una medida que es difícil de interpretar por los humanos al ser una medida logarítmica (un sonido de 20db es 10 veces mas fuerte que un sonido de 10db).

Capítulo 5

Conclusiones y trabajos futuros

Podemos concluir que el trabajo se ha realizado con éxito, que gracias al boom del DIY el movimiento MAKER y los avances tecnológicos es relativamente fácil crear e implementar sistemas como el descrito en esta memoria a un bajo coste, también podemos concluir que las plataformas abiertas siempre tienen mayor comunidad por lo que es más fácil el desarrollo en ellas, y que los decibelios son una medida un tanto compleja.

Como trabajo futuro se podrían implementar otros sistemas de monitorización con otro hardware e integrar fácilmente al panel de control mientras estos sistemas puedan hablar? HTTP también se podría mejorar la seguridad de la API con un sistema de autenticación y por último se podría implementar un hard sleep del imp programable desde el panel de control para no tener que desconectarlo en periodos que no se use pero aun en hard sleep sigue consumiendo batería por lo que es mejor apagarlo completamente.

Capítulo 6

Summary and Conclusions

We can conclude that the work has been successful, thanks to the boom of DIY, the MAKER movement and technological advances is relatively easy create and implement systems like as described in this report for a low cost, we can also conclude that open platforms always have greater community making it easier the development in them, and that decibels are a somewhat complex measure.

As future work could be implemented with other monitoring systems and other hardware and it can be easily integrated control panel while these systems can "talk" HTTP could also improve the security of the API with an authentication system and finally one could implement a hard sleep, programmable from the control panel to avoid having to disconnect but in hard sleep yet still consumes battery so it is better to turn it off completely.

Capítulo 7

Presupuesto

7.1. Tabla de Presupuesto y Coste Total

Articulo	Coste
1 Electric imp (link)	29.95 \$
1 APRIL developement board (link)	12.50 \$
2 Microfonos omnidireccionales 3V (link)	15.9 \$
1 LiPo bateria 3.7v 2500mah (link)	14.95 \$
1 IC cargador de baterias LiPo (link)	5.95 \$
1 Zumbador 3-30V (link)	1.50 \$
1 Regulador de voltaje 3.3V-1A (link)	14.95 \$
TOTAL	95,7 \$

Tabla 7.1: Tabla de Presupuesto y Coste Total

Apéndice A

Código del imp

```
*****
*
* Fichero imp.nut
*
*****
*
* AUTOR : Sergio Medina Toledo
*
*
* FECHA : 4 de julio de 2014
*
*
* DESCRIPCION : Codigo del electric imp
*
*
*****/

SAMPLE_TIME <- 50 // in ms

VOLTAJE <- 3.3
MAX_SAMPLE_ADC <- 65535.0

OFFSET <- 24823.86

SLEEP_TIME <- 5

ACTIVATED <- true

buzzer <- hardware.pin5;
micro1 <- hardware.pin8;
micro2 <- hardware.pin9;
ledRed <- hardware.pin1;
ledGreen <- hardware.pin2;

buzzer.configure(PWM_OUT,1.0/5000.0,0.0);
buzzer.write(0.0);
```

```

micro1.configure(ANALOG_IN);
micro2.configure(ANALOG_IN);

ledRed.configure(DIGITAL_OUT);
ledGreen.configure(DIGITAL_OUT);

dbLowLimit <- 100;
dbMediumLimit <- 100;
dbHighLimit <- 100;

function main()
{
  imp.setpowersave(true);
  agent.send("setMac",imp.getmacaddress());
  agent.send("getConf",0);
  loop();
}

function loop()
{
  agent.send("getConf",0);
  buzzer.write(0.0);

  local sum_sample1 = 0;
  local sum_sample2 = 0;
  local startTime = hardware.millis();
  local number_samples = 0
  // sampling
  while(hardware.millis() < startTime + SAMPLE_TIME)
  {
    sum_sample1 += math.pow((micro1.read() - OFFSET),2)
    sum_sample2 += math.pow((micro2.read() - OFFSET),2)
    number_samples += 1;
  }

  local Vrms1 = math.sqrt(sum_sample1/number_samples) / MAX_SAMPLE_ADC * VOLTAJE;
  local Vrms2 = math.sqrt(sum_sample2/number_samples) / MAX_SAMPLE_ADC * VOLTAJE;

  local db1 = 20.0 * math.log10(Vrms1/0.043) + 37;
  local db2 = 20.0 * math.log10(Vrms2/0.043) + 37;

  local dbData = array(2,0);
  dbData[0] = db1;
  dbData[1] = db2;
  agent.send("db", dbData);

  if ((db1 + db2) / 2 > dbHighLimit && ACTIVATED) {
    buzzer.write(0.5);
  }

  if ((db1 + db2) / 2 > dbHighLimit) {
    ledRed.write(1);
    ledGreen.write(0);
  }
}

```

```
    } else if ((db1 + db2) / 2 > dbMediumLimit) {
        ledRed.write(1);
        ledGreen.write(1);
    } else {
        ledRed.write(0);
        ledGreen.write(0);
    }
}

imp.wakeup(SLEEP_TIME, loop);
}

agent.on("setConf", function(conf) {
    dbLowLimit = conf["dbLowLimit"];
    dbMediumLimit = conf["dbMediumLimit"];
    dbHighLimit = conf["dbHighLimit"];
    SAMPLE_TIME = conf["SAMPLE_TIME"];
    SLEEP_TIME = conf["SLEEP_TIME"];
    ACTIVATED = conf["ACTIVATED"];
})

imp.onidle(function() {
    imp.sleep(SLEEP_TIME - 0.5);
});

main();
```

Apéndice B

Código Agente

```
*****
*
* Fichero imp.nut
*
*****
*
* AUTOR : Sergio Medina Toledo
*
*
* FECHA : 4 de julio de 2014
*
*
* DESCRIPCION :Codigo del agente
*
*
*****/
MAC <- 0

DATA_URL <- "http://193.145.101.234/api/v1/noise/data"

CONF_URL <- "http://193.145.101.234/api/v1/imp/config"

HEADERS <- {"Content-Type": "application/x-www-form-urlencoded"};

device.on("setMac", function(mac){
    MAC = mac;
    server.log("MAC: " + MAC)
})

function HttpPost (url, headers, string) {
    local request = http.post(url, headers, string);
    local response = request.sendsync();
    return response;
}
}
```

```
device.on("db", function(db) {
    HttpPost (DATA_URL, HEADERS, "imp=" + MAC + "&db1=" + db[0] + "&db2=" + db[1] )
});

device.on("getConfig", function(param) {
    local request = http.get(CONF_URL + "?imp=" + MAC + "&format=json", {"":""});
    local response = request.sendsync();
    local conf = http.jsondecode(response.body);
    device.send("setConf",conf);
});
```

Apéndice C

Repositorio de código del Control Panel

<https://gitlab.com/lumasepa/noise-control-panel>

Bibliografía

- [1] Electric Imp. <http://electricimp.com/>.
- [2] Django. <https://www.djangoproject.com/>.
- [3] Adafruit. <https://www.adafruit.com/>.
- [4] Decibelio. <https://en.wikipedia.org/wiki/Decibel>.
- [5] I2C. <https://en.wikipedia.org/wiki/I%C2%B2C>.
- [6] Inteligencia Ambiental. https://en.wikipedia.org/wiki/Ambient_intelligence.