Trabajo de Inserción Profesional

Informe Final

*Título:*

Integración de tecnología para control y automatización de espacio físico*.*

*Alumno:*

*Martin Alejandro Melo.*

*Director:*

*Ing. José Luis Di Biase.*

*Carrera:*

*Tecnicatura en Programación Informática.*



Contenido

[Introducción 4](#_Toc405136788)

[Contexto 4](#_Toc405136789)

[Problemática 4](#_Toc405136790)

[Objetivos 4](#_Toc405136791)

[Resumen 6](#_Toc405136792)

[Desarrollo de la solución propuesta 7](#_Toc405136793)

[Descripción/Contexto General 7](#_Toc405136794)

[Automatización y control de los espacios físicos 7](#_Toc405136795)

[Internet Of Things 8](#_Toc405136796)

[Hardware 9](#_Toc405136797)

[Como nace 9](#_Toc405136798)

[¿Qué es? 10](#_Toc405136799)

[¿Qué resuelve? 11](#_Toc405136800)

[Componente base 11](#_Toc405136801)

[Sensores específicos 12](#_Toc405136802)

[Componentes específicos 12](#_Toc405136803)

[Diagramas de conexión 13](#_Toc405136804)

[Software 13](#_Toc405136805)

[Front-End 14](#_Toc405136806)

[Back-End 15](#_Toc405136807)

[Middleware 15](#_Toc405136808)

[MQTT 16](#_Toc405136809)

[Mosca 16](#_Toc405136810)

[Socket.IO 17](#_Toc405136811)

[Firmware 17](#_Toc405136812)

[PubSubClient 18](#_Toc405136813)

[ArduinoJson 18](#_Toc405136814)

[Forma de comunicación 18](#_Toc405136815)

[Protocolo de comunicación 19](#_Toc405136816)

[Contenidos de los mensajes 22](#_Toc405136817)

[Conclusiones 25](#_Toc405136818)

[Resultados Obtenidos 25](#_Toc405136819)

[Anexos 26](#_Toc405136820)

[Diagramas del Hardware 26](#_Toc405136821)

[M2M 28](#_Toc405136822)

[Licencia MIT 28](#_Toc405136823)

[Referencias 29](#_Toc405136824)

[Referencias Adicionales 29](#_Toc405136825)

# Introducción

## Contexto

En la actualidad es cada vez más común y frecuente ver dispositivos inteligentes en nuestrasvidas. Desde Smartphones, etc.

Una breve introducción al presente y la forma de vida que llevamos.

## Problemática

Dado el contexto denotar que todos estos dispositivos electrónicos pueden estar interconectados através de un sistema informático que los una. Además este sistema informático puede interactuar con ellos para hacerlos realizar ciertas tareas o recolectar información de ellos.

# Objetivos

La empresa en la que actualmente me desempeño programando necesita tener una herramienta que sirva de control y automatización de ciertos espacios físicos (oficina, depósito, zona común, etc.). En este momento estos controles son llevados a cabo por empleados que los realizan diariamente cuando algún superior se lo indica dándole además las instrucciones necesarias para su realización.

Se necesita tener acceso a través de internet a un sitio web con la información de los distintos controles que se tengan los espacios físicos para corroborar estados, efectuar acciones o configurar tareas automatizadas para que se realicen según un calendario.

Los principales controles realizados son los siguientes:

* Iluminación: Se desea poder controlar la iluminación de determinados sectores de forma automática, es decir configurar su encendido y apagado determinando un calendario para el mismo o manualmente.
* Climatización: Esto sería idéntico que para la iluminación ya que determinados lugares del recinto cuentan con diferentes rangos necesarios de humedad y temperatura (por ejemplo: un depósito de documentos en papel o una habitación con servidores).
* Empleados: Se desea poder controlar la llegada y partida de los empleados al recinto através de alguna forma de control como un lector de huellas digital.
* Seguridad: Se desea poder conocer el estado de los distintos puntos de acceso(puertas) al recinto.

Estos 4 controles principales se llevan a cabo diariamente y son de una importancia vital para la empresa.

La elaboración de este trabajo tuvo como objetivo principal desarrollar una herramienta que automatice todos los aspectos mencionados en la sección anterior. Dicha herramienta es intuitiva y de fácil acceso, a su vez esta es distribuida como software libre bajo la licencia MIT.

# Resumen

En la actualidad es cada vez más común y frecuente ver dispositivos inteligentes en nuestrasvidas. Desde Smartphone, etc.

Una breve introducción al presente y la forma de vida que llevamos.

Dado el contexto denotar que todos estos dispositivos electrónicos pueden estar interconectados através de un sistema informático que los una. Además este sistema informático puede interactuar con ellos para hacerlos realizar ciertas tareas o recolectar información de ellos.

La solución propuesta es la de desarrollar una herramienta la cual logre unir la electrónica y la programación para formar una solución capaz de simplificar tareas que día a día se realizan de forma presencial. Como por ejemplo verificar el estado de una puerta, encender o apagarluces, obtener diferentes datos através de sensores que se encuentren en el recinto, etc. Todos estos se va a unir utilizando protocolos de comunicación entre los dispositivos y el programa central con el cual se van a comunicar. El usuario que utilice el sistema se podrá comunicar através de un sitio web(Front-End).

# Desarrollo de la solución propuesta

## Descripción/Contexto General

Este proyecto se basó en los conceptos de Automatización y control de los espacios físicos y también en el concepto de Internet of things para poder obtener un herramienta útil. TODO: METER MAS DATA.

### Automatización y control de los espacios físicos

Es también conocida como Domotica, la cual es un conjunto de sistemas capaces de automatizar una vivienda, aportando servicios de gestión energética, seguridad, bienestar y comunicación, y que pueden estar integrados por medio de redes interiores y exteriores de comunicación, cableadas o inalámbricas, y cuyo control goza de cierta ubicuidad, desde dentro y fuera del hogar. Se podría definir como la integración de la tecnología en el diseño inteligente de un recinto cerrado.

### Internet Of Things



El Internet de las cosas o Internet Of Things es un término que se relacionan mucho con la web. Pero también resulta un concepto un poco abstracto para los no iniciados pese a que ha estado ganando bastante popularidad en el último tiempo. La idea que intenta representar queda bastante bien ilustrada por su nombre, cosas cotidianas que se conectan a Internet, pero que en realidad se trata de mucho más que eso.  
  
Para entender de qué va el internet de las cosas debemos también comprender que sus fundamentos no son en lo absoluto nuevos. Desde hace unos 30 años que se viene trabajando con la idea de hacer un poco más interactivos todos los objetos de uso diario. Planteos como el hogar inteligente o casa del mañana, han evolucionado antes de que nos demos cuenta en el hogar conectado para entrar al internet de las cosas.  
  
El Internet de las cosas potencia objetos que antiguamente se conectaban mediante circuito cerrado, como comunicadores, cámaras, sensores, y demás, y les permite comunicarse globalmente mediante el uso de la red de redes.  
  
Si tuviéramos que dar una definición del internet de las cosas probablemente lo mejor sería decir que se trata de una red que interconecta objetos físicos valiéndose de la red. Los mentados objetos se valen de sistemas embebidos, o lo que es lo mismo, hardware especializado que le permite no solo la conectividad a internet, sino que además programa eventos específicos en función de las tareas que le sean dictadas remotamente (por ejemplo desde nuestro celular mientras trabajamos en la oficina).

## Hardware

El hardware principal el cual se encarga se suministrar información al sistema y a su vez al cual se le podrá pedir realizar distintos tipos de acciones estará basado en la plataforma Arduino.

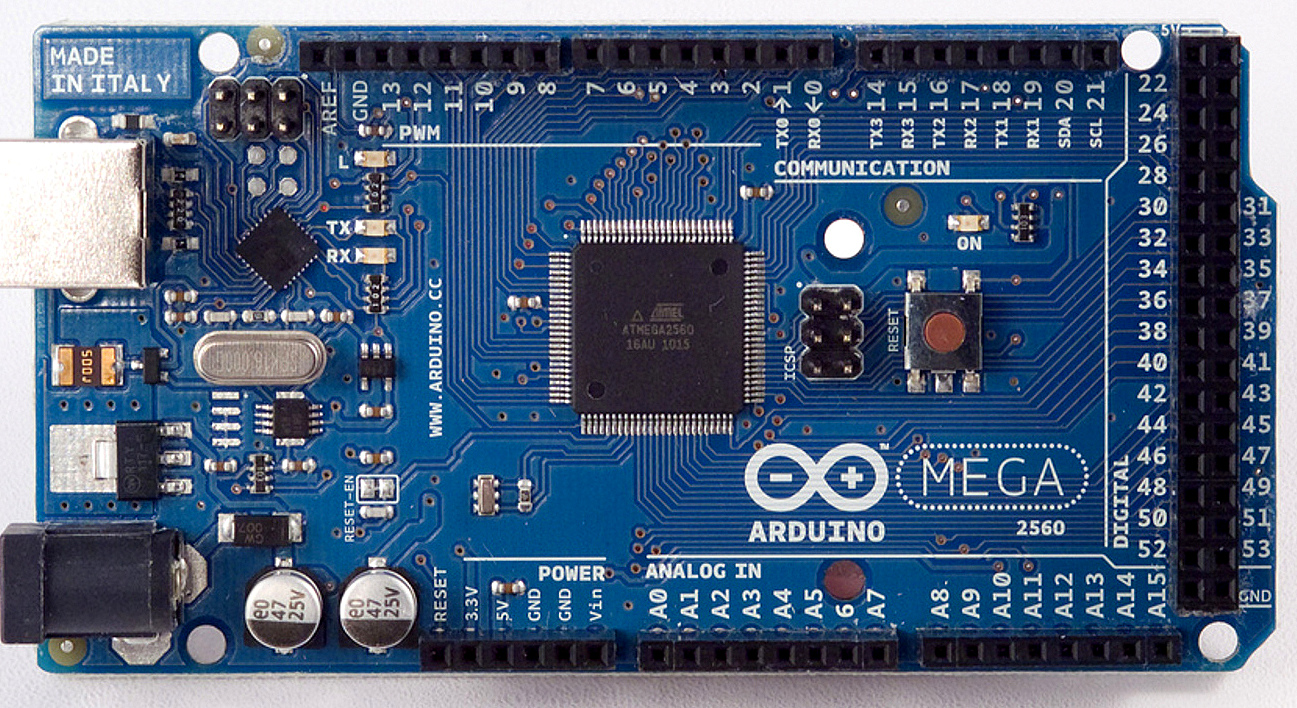
### Como nace

La creación de Arduino se inicio en el año 2005 y fue un proyecto para los estudiantes de un instituto en Ivrea Italia. Este proyecto surgió de la necesidad de contar con un dispositivo de bajo costo y que se pueda utilizar con cualquier sistema operativo.

### ¿Qué es?

*Arduino* es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares.

El hardware consiste en una placa con un microcontrolador [Atmel AVR](http://es.wikipedia.org/wiki/AVR). La siguiente imagen pertenece una placa Arduino Mega2650.



Actualmente existen muchos entornos de desarrollos para crear programas para Arduino, los cuales se encuentran programados en diferentes lenguajes como C, Java, y Python.

La sintaxis del lenguaje de programación Arduino es una versión simplificada de C/C++.

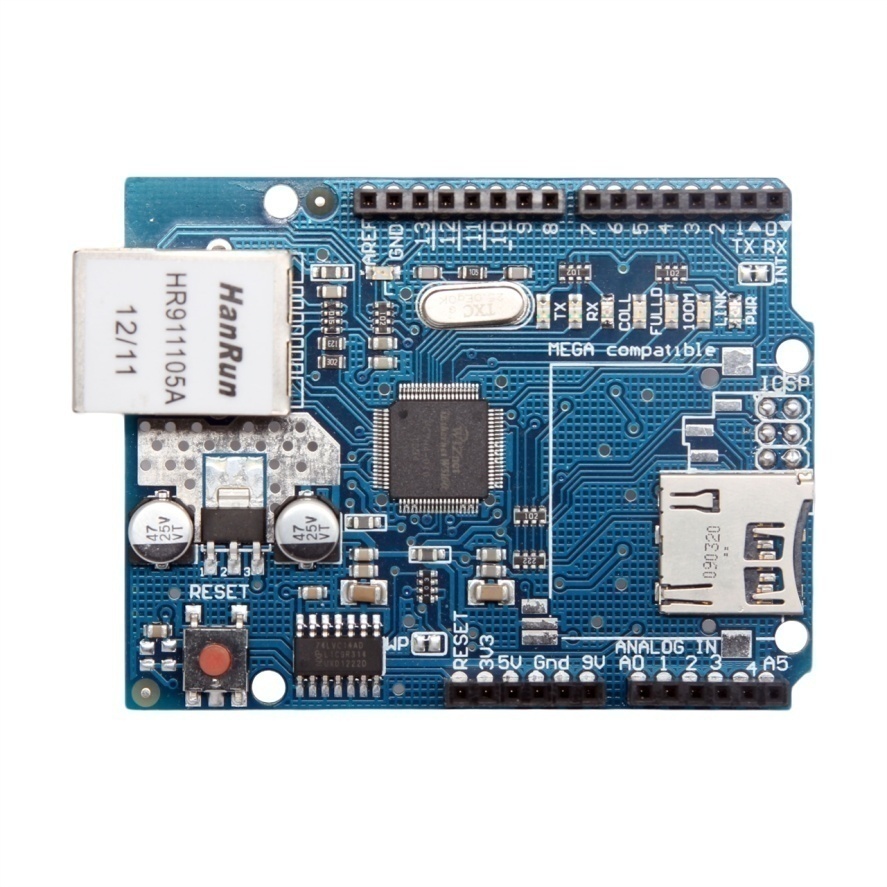
### ¿Qué resuelve?

Se utilizo para crear un dispositivo capaz de poder comunicarse en ambos sentidos con la aplicación que utiliza el usuario y que a su vez se pueda utilizar para efectuar distintas acciones basándose en la información que se le suministre atravez de un protocolo especifico. A su vez, el mismo está conectado a distintos circuitos electrónicos que le proveen efectuar las acciones anteriormente mencionadas y también la obtención de información de los sensores que posea.

Para que el dispositivo funcione y se comunique con la aplicación solo es necesario tener los siguientes componentes base.

### Componente base

Los componentes de base necesarios para que el sistema funcione son:

* Arduino Mega2560.
* Ethernet Shield con chipset Wiznet W5100.

A los componentes anteriores se pueden añadir los sensores específicos los cuales se utilizan para extraer información del ambiente.

### Sensores específicos

El sensor de temperatura utilizado en el circuito impreso es un LM35DZ. Cuya hoja de datos es la siguiente: <http://www.ti.com.cn/cn/lit/ds/symlink/lm35.pdf>

También se utilizaron ciertos componentes específicos para simular sensores de puerta o para controlar el cambio de estado de otros componentes electrónicos como las luces.

### Componentes específicos

Los componentes específicos utilizados son los siguientes:

* ULN2803. Es un array de Darlington que se utiliza para controlar las luces que se encuentran conectadas al circuito de las luces. Su hoja de datos es la siguiente: [http://www.ti.com/lit/ds/symlink/uln2803.pdf](http://www.ti.com/lit/ds/symlink/uln2803a.pdf).
* Botones de estado normal abierto. Cada botón se utiliza para simular apertura de puertas.
* Resistencia 1K ohm. Se utilizan para eliminar el efecto "antena" de los cables que se conectan a las entrada del Arduino, estas resistencia van junto a los botones.

Todos estos componentes son de bajo costo y pueden ser adquiridos en cualquier local donde vendan componentes electrónicos.

### Diagramas de conexión

Las imágenesque se encuentran en el [Anexo](#_Diagramas_del_Hardware) pertenecen al esquema de conexión de los componentes específicos que conforman el hardware.

Estas imágenes se encuentran elaboradas utilizando el programa Fritzing. La primera de ellas pertenece al esquema de conexión de los componentes al microcontrolador. La segunda imagen pertenece al esquema de conexión de los componentes utilizando un protoboard, el cual es un tablero con orificios conectados eléctricamente entre sí, en el cual se pueden insertar componentes electrónicos y cables para el armado y prototipado de circuitos electrónicos y sistemas similares.

## Software

El software que compone a la aplicación se encuentre separado en 3 partes. Estas son : Back-End, Middlewarey Front-End. Cada parte posee un funcionamiento diferente dentro de la aplicación. De forma que la misma se encuentre modularizada y a su vez puedan distribuirse ciertos módulos en diferentes equipos.

Tanto el Back-End como el Front-End se encuentranconstruidasa partir del generador MeanJs. El cual permite generar aplicaciones web utilizando MongoDB, Express, AngularJS y NodeJS. Este generador se utilizo ya que permite reutilizar muchas funcionalidades que eran especificas para esta aplicación, lo cual redujo ampliamente el tiempo de desarrollo.

### Front-End

El Front-End es la parte de la aplicación con la cual el usuario de la misma interactúa en forma directa. En este caso es la pagina web que se utiliza para ver la información y demás.

Este modulo se encuentra programado principalmente utilizando AngularJS, un Framework Javascript. Ya que se necesitaba un interfaz amigable, rápida, liviana en términos de almacenamiento, fácil de desarrollar y opensource .

También se utilizaron las siguientes librerías y frameworks:

* Jquery: Utilizado para realizar acciones de manera más sencilla y reducir el tiempo de desarrollo.
* Bootstrap: Utilizada para darle estilo al sitio.
* socket.io-client: Utilizada para la comunicación con los dispositivos que brindan información a la aplicación.
* malhar-angular-widgets: Utilizada para generar los widgets que contienen la información obtenida de los dispositivos.
* malhar-angular-dashboard: Utilizada para agrupar los Widgets anteriores de una manera mas fácil para reducir el tiempo de diseño.
* font-awesome: Utilizada para graficar iconos.

### Back-End

El Back-End es la parte de la aplicación la cual se encarga de realizar el manejo de la base de datos, y de las distintas tareas que el Front-End le pida realizar.

Este se encuentra programado principalmente utilizando NodeJS, el cual está basado en Javascript. Se utilizo ya para reducir el tiempo de desarrollo y a su vez porque era el tipo de herramienta que se utiliza para este tipo de aplicaciones.

También se utilizan las siguientes librerías y Frameworks:

* MongoDB: Es el motor de la base de datos.
* Mongoose: Utilizado para administrar la base de datos.
* Express: Utilizado para unir el Back-End con el Front-End ya que es el que provee una vía de comunicación entre ambos puntos.
* node-schedule: Utilizada para llevar a cabo las tareas automáticas que se creen en la aplicación.

### Middleware

El Middleware es la parte de la aplicación que permite la comunicación entre los distintos dispositivos y el Front-End. Es decir que por ejemplo cuando el usuario desea realizar el encendido de una luz, el middleware será el encargo de informarle al Dispositivo de esta acción.

En esta aplicación el middleware se encarga únicamente de unir la comunicación entre el Front-End y los distintos dispositivos de control con los que se cuente.

Este se encuentra programado en NodeJS y utiliza los siguientes Frameworks:

* Mqtt.
* Socket.io.
* Mosca.

### MQTT

[Message Queue Telemetry Transport (MQTT)](http://mqtt.org/) es un protocolo de conectividad enfocado a M2M (machine-to-machine) y al IOT (Internet of Things) ya que se ha diseñado para ser un protocolo de mensajería extremadamente ligero basado en TCP.

Es útil para conexiones con sitios remotos donde se posee una alta latencia y el ancho de banda es muy importante.

Una característica muy importante es que al ser un protocolo tan ligero existen clientes y servidores MQTT en diversos lenguajes.

En la aplicación es utilizado como servidor de mensajes para permitir la comunicación entre las distintas aplicaciones que utilicen el mismo protocolo de comunicación

### Mosca

Es un bróker de mensajería, con lo cual es un mecanismo mediador de la comunicación entre aplicaciones y/o dispositivos, permitiendo minimizar el grado de conocimiento mutuo que estos necesitan tener, para poder intercambiar mensajes, implementando así efectivamente su [desacoplamiento](http://es.wikipedia.org/wiki/Dise%C3%B1o_estructurado#Acoplamiento).

El propósito del bróker es recibir los mensajes entrantes desde las aplicaciones y llevar a cabo determinadas acciones con ellas.

En la aplicación se utiliza como servidor de mensajes principal para permitir la comunicación entre todos los servicios.

### Socket.IO

[*Socket.io*](http://socket.io/) es una librería que permite manejar eventos en tiempo real mediante una conexión TCP y todo ello en Javascript. Es realmente potente y permite hacer todo tipo de aplicaciones en tiempo real, como por ejemplos servicios de mensajería/chats. Ya que permite la comunicación bidireccional entre clientes web y servidor. Esta librería está compuesta de 2 partes, la primera de ellas es la que corre en el navegador del cliente y la otra de ellas es la que corre en el servidor de la aplicación. Ambas partes son casi idénticas y ambas se basan en el manejo de eventos.

*Socket.io* utiliza principalmente elprotocolo de Websockets, aunque también permite utilizar polling manteniendo la misma interfaz.

En la aplicación se utilizo Socket.io para crear un servidor de mensajes para permitir la comunicación entre el Front-End y los dispositivos ya que no solo provee la funcionalidad de websockets si no que a su vez también contiene características extra entre las cuales se incluye la posibilidad de realizar broadcast a múltiples sockets, almacenar información asociada a cada cliente y procesar entrada/salida de forma asincrónica..

## Firmware

El firmware que se utiliza en el microcontrolador está programado en el lenguaje C. En el mismo se incluyen las siguientes librerías:

* PubSubClient.
* ArduinoJson.

### PubSubClient

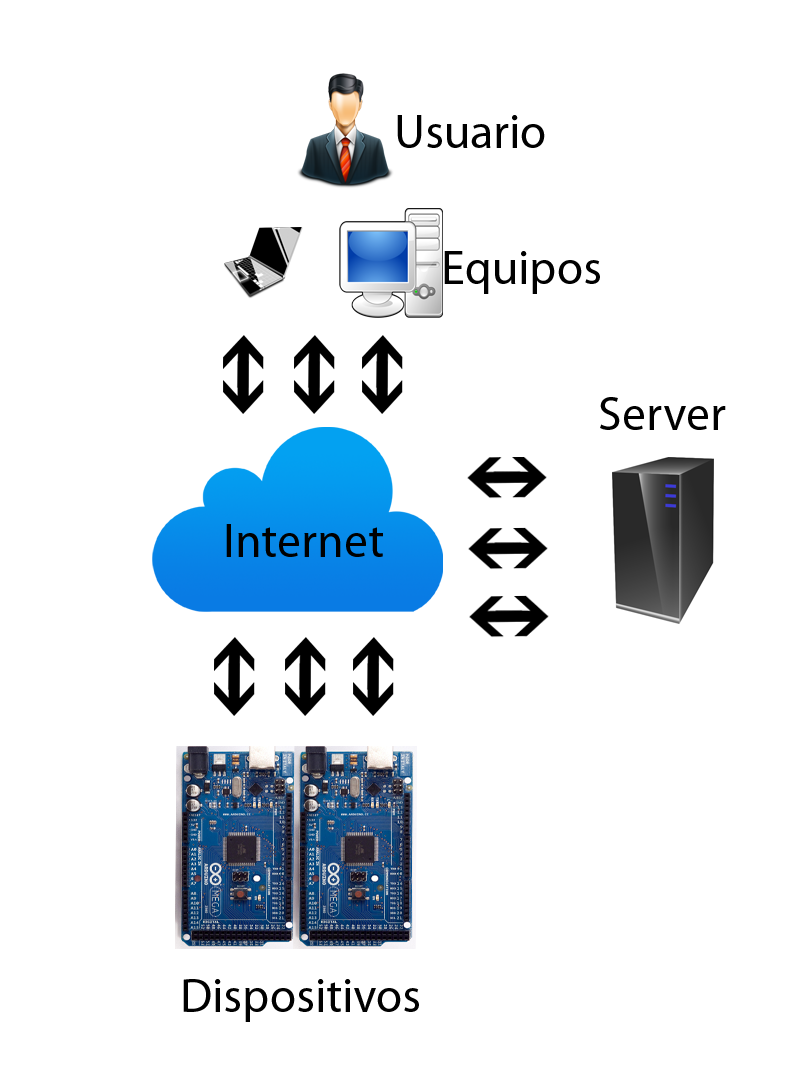
Esta librería permite la suscripción y publicación de mensajes contra un servidor que soporte el protocolo MQTT. Y es utilizado para enviar y recibir información desde los micro controladores.

### ArduinoJson

Permite el parseo y utilización de el texto con formato JSON que se le envía al micro controlador con las instrucciones a realizar.

## Forma de comunicación

Las distintas partes de la aplicación se comunican de la siguiente forma:



### Protocolo de comunicación

El protocolo de comunicación utilizado se basa en el envió de pequeñas porciones de texto con el formato JSON. Que contienen pequeña información de las acciones que se desean realizar.

#### ¿Qué es JSON?

JSON (Javascript Object Notation - Notación de Objetos de Javascript) es un formato ligero de intercambio de datos. Leerlo y escribirlo es simple para humanos, mientras que para las máquinas es simple interpretarlo y generarlo. Está basado en un subconjunto del [Lenguaje de Programación Javascript](http://javascript.crockford.com/), [Standard ECMA-262 3rd Edition - Diciembre 1999](http://www.ecma-international.org/publications/files/ecma-st/ECMA-262.pdf).

JSON es un formato de texto que es completamente independiente del lenguaje pero utiliza convenciones que son ampliamente conocidos por los programadores de la familia de lenguajes C, incluyendo C, C++, C#, Java, Javascript, Perl, Python, y muchos otros. Estas propiedades hacen que JSON sea un lenguaje ideal para el intercambio de datos.

JSON está constituido por dos estructuras:

* Una colección de pares de nombre/valor. En varios lenguajes esto es conocido como un *objeto*, registro, estructura, diccionario, tabla hash, lista de claves o un arreglo asociativo.
* Una lista ordenada de valores. En la mayoría de los lenguajes, esto se implementa como arreglos, vectores, listas o secuencias.

Estas son estructuras universales; virtualmente todos los lenguajes de programación las soportan de una forma u otra. Es razonable que un formato de intercambio de datos que es independiente del lenguaje de programación se base en estas estructuras.

En JSON, se presentan de estas formas:

Un*objeto* es un conjunto desordenado de pares nombre/valor. Un objeto comienza con { (llave de apertura) y termine con } (llave de cierre). Cada nombre es seguido por : (dos puntos) y los pares nombre/valor están separados por , (coma).

Un*arreglo* es una colección de valores. Un arreglo comienza con [ (corchete izquierdo) y termina con ] (corchete derecho). Los valores se separan por , (coma).

Un *valor* puede ser una *cadena de caracteres* con comillas dobles, o un *número*, o true o false o null, o un *objeto* o un *arreglo*. Estas estructuras pueden anidarse.

Una *cadena de caracteres* es una colección de cero o más caracteres Unicode, encerrados entre comillas dobles, usando barras divisorias invertidas como escape. Un carácter está representado por una cadena de caracteres de un único carácter. Una *cadena de caracteres* es parecida a una cadena de caracteres C o Java.

Un *número* es similar a un número C o Java, excepto que no se usan los formatos octales y hexadecimales.

Los espacios en blanco pueden insertarse entre cualquier par de símbolos.

Un pequeño ejemplo de json es el siguiente:

{

"nombre": "cofre simple",

"descripcion": "soy un cofre simple de madera",

"cantidad": 3,

"elementos": [

{

"nombre": "Espada de diamantes",

"id": 276

},

{

"nombre": "Disco Strad",

"id": 2264

},

{

"nombre": "mesa de encantamiento",

"id": 116

}

]

}

### Contenidos de los mensajes

Para esta aplicación se creó una estructura básica y sencilla para poder comunicarse e interactuar con los dispositivos y viceversa.

La estructura para comunicarse con los dispositivo está formada de la siguiente manera:

{

"command": "comando",

"destino": "destino",

"datos": {}

}

Donde el " comando " es el tipo de petición a realizar, pudiendo ser "accion" o "exponerServicios". Siendo la primera la cual informa que el dispositivo va a tener que realizar una accion como por ejemplo encender una luz o enviar información de un sensor, Y la segunda para que el dispositivo envié la información de los servicios que provee el mismo.

El "destino" es para marcar que tipo de petición se desea como por ejemplo si es iluminación, sensores o climatización. Si el comando fue "exponerServicios" este campo sirve para denotar por cual servicio se esta preguntando, para este comando el campo puede tener los siguientes valores: "luz", "sensor" y "todo".

El campo datos es otro json que depende de la sección de destino, ya que la información es diferente para cada sección. Para la sección "Iluminacion" el campo "datos" tendrá la siguiente estructura:

{

"id": "numero de luz",

"estado": "valor"

}

Donde "id" tendrá el numero que pertenece a una luz que el dispositivo tiene asignado. Y "estado" será "on" o "off" siendo el primero para denotar el encendido y el segundo para denotar el apagado de la luz.

Para la sección "sensores" el campo "datos" tendrá la siguiente estructura:

{

"id": "tipo de sensor",

"posicion": "nombre de la posición"

}

Donde "id" pueden ser "temperatura" o "puertas", siendo el primero para pedir información de sensores que son numéricos(por ejemplo un sensor de temperatura ambiental con un rango de valores entre -50 y 150 grados Celsius) y el segundo para sensores que estados de forma booleana, por ejemplo un sensor de apertura donde se desea tener solo 2 valores posibles(true para denotar un estado y false para denotar otro, pudiendo ser el primero para puerta cerrada y el segundo para puerta abierta).

El campo "posicion" se utiliza para denotar el sensor del cual se desea obtener la información.

La respuesta de los dispositivos son únicamente 2, siendo una de ellas para responder a las peticiones de información de un sensor donde el valor de la respuesta es el valor resultado(un numero o un boolean).

La segunda respuesta es a la petición de exponer los servicios. Esta ultima puede tener 3 formas diferentes dependiendo la petición realizada al dispositivo. La primera forma es en respuesta a la petición con el valor de sección "todo" la cual si bien esta implementado no se utiliza, y tendrá la siguiente estructura:

{

"suscripto": "nombre del dispositivo",

"tipos": [

{

"nombre": "luz"

},

{

"nombre": "sensor"

}

]

}

La segunda forma es en respuesta a la petición con el valor de sección "luz" y tendrá la siguiente estructura:

{

"luz": [

{

"nombre": "nombre que identifica la luz",

"topico": "valor numérico correspondiente a la luz"

},

]

}

La tercer forma es en respuesta a la petición con el valor de sección "sensor" y tendrá la siguiente estructura:

{

"sensor": [

{

"nombre": "nombre del sensor",

"topico": "tópico destino",

"tipo": "tipo de datos que se obtiene del sensor"

}

]

}

La información obtenida de las respuesta a la petición de exponer servicios se utiliza para saber cómo interactuar con el dispositivo.

# Conclusiones

### Resultados Obtenidos

# Anexos

## Diagramas del Hardware





## M2M

M2M (Machine to Machine o Máquina a Máquina) es un concepto genérico que se refiere al intercambio de información o comunicación en formato de datos entre dos máquinas remotas.

## Licencia MIT

Esta licencia permite reutilizar el software así licenciado tanto para ser [software libre](http://es.wikipedia.org/wiki/Software_libre) como para ser [software no libre](http://es.wikipedia.org/wiki/Software_no_libre), permitiendo no liberar los cambios realizados al programa original.

También permite licenciar dichos cambios con [licencia BSD](http://es.wikipedia.org/wiki/Licencia_BSD), [GPL](http://es.wikipedia.org/wiki/Licencia_p%C3%BAblica_general_de_GNU), u otra cualquiera que sea compatible (es decir, que cumpla las cláusulas de distribución).

Con esta licencia se tiene software libre. Ejemplos en los que podría interesar su aplicación serían las [licencias duales](http://es.wikipedia.org/wiki/Licencia_dual), si se pretende difundir un [estándar](http://es.wikipedia.org/wiki/Norma_(tecnolog%C3%ADa)) mediante una implementación de referencia, o si simplemente se pretende que el producto sea Libre sin mayores consideraciones.

# Referencias

[1]Paulo Novais, Davy Preuveneers, Juan Manuel Corchado Rodríguez; AmbientIntelligence - Software and Applications: 2nd International Symposium on AmbientIntelligence, Springer, 2011.

[2] Rubén Saavedra Silveira, Automatización de viviendas y edificios, Ediciones CEAC, 2009.

[3] James Gerhart, Home Automation and Wiring, McGraw Hill Professional, 1999.

[4] Jamari Harmon, The Home Automation, Bookpubber, 2014.

[5] David Flanagan,JavaScript: The Definitive Guide, O'Reilly Media, Inc., 2011.

[6] Gary Shelly, Denise Woods, HTML: Introductory Concepts and Techniques, Cengage Learning, 2008.

[7] Brian W. Kernighan, Dennis M. Ritchie, El Lenguaje de Progra C, Pearson Educación, 1991.

## Referencias Adicionales

http://www.wisegeek.com/what-is-domotics.htm

https://angularjs.org/

http://jquery.com/

http://www.json.org/json-es.html

http://nodejs.org/

https://github.com/bblanchon/ArduinoJson

http://knolleary.net/arduino-client-for-mqtt/