

CAPÍTULO I

LOS SISTEMAS EMBEBIDOS

1.1 ¿QUÉ SON LOS SISTEMAS EMBEBIDOS?

Un sistema embebido es una combinación de hardware y software que trabaja junto con algún sistema mecánico o electrónico diseñado para cumplir una función específica. Por lo tanto, se usa para dotar de “inteligencia” a un artefacto.

La diferencia principal entre un sistema embebido y una computadora de propósito general (por ejemplo, una notebook) radica justamente en la especialización que tiene el sistema embebido. Una computadora personal podría utilizarse para varias cosas: para jugar un videojuego, ver videos en Internet, comprar acciones, realizar una simulación o ejecutar algún programa de edición de imágenes. Por el contrario, un sistema embebido solo está diseñado, desde su hardware y software, para cumplir con un conjunto finito de funciones que determinan una tarea específica. Por ejemplo: calentar comida en un microondas, administrar eficientemente el sistema de frenos en un automóvil o ejecutar programas de lavado en un lavarropa.

El hardware de un sistema embebido se compone principalmente por un microprocesador, sensores y actuadores. Los sensores le permiten al microprocesador obtener información del mundo real; el microprocesador toma decisiones basado en el software que ejecuta; y los actuadores realizan acciones con el mundo físico.

Los sensores pueden ser variados. Hay de humo, de temperatura, de sonido, de proximidad, de velocidad, de posición, de tiempo, de inclinación y hasta sensores de ritmo cardíaco y presión atmosférica. Mientras que los actuadores pueden ser sirenas, motores, luces o controles de válvulas, entre otros.

1.2 ¿QUÉ ES ARDUINO?

Arduino, como concepto, es un sistema electrónico de prototipado abierto, basado en software y hardware flexibles. Arduino es simplemente una placa que contiene un microprocesador ATmega, una serie de pines de entrada y salida de propósito general para datos analógicos y digitales, y una conexión USB que permite cargarle programas y establecer una comunicación con una PC. El entorno de desarrollo integrado (IDE), es el software con el que podremos programarlo. Este software sirve para programar otros dispositivos como el ESP8266 o el NodeMCU.

1.3 OPEN SOURCE Y OPEN HARDWARE COMO FILOSOFÍA DE TRABAJO

Arduino, desde su concepción, pertenece al mundo del open source (software libre) y del open hardware (hardware abierto), alineado con la filosofía de trabajo que ambos movimientos proponen. Tanto el hardware como el software abierto fomentan el desarrollo de tecnologías cuyos diseños son públicos y sin secretos, con el fin de que otros hagan uso y modifiquen esa tecnología.

Esto da lugar a que el uso y los costos de las tecnologías sean bajos o nulos, y se facilite el acceso a más cantidad de usuarios; del mismo modo y en sentido inverso, también fomenta que las personas puedan participar y hacer aportes a esas mismas tecnologías, ayudándolas a madurar y mejorar.

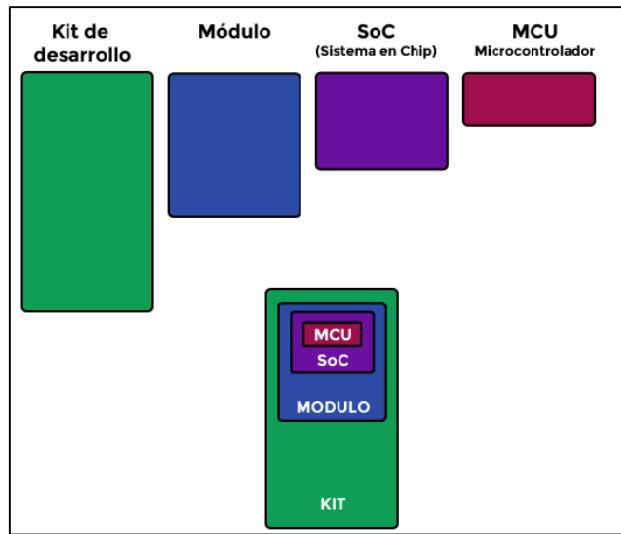
1.4 ¿Qué es NodeMCU?

NodeMCU es una plataforma de desarrollo para Internet de las cosas (IoT) muy similar a Arduino. Te permite crear rápidamente proyectos que se pueden conectar a Internet por Wifi. Cuando fue presentada, no existía la integración de ESP8266 con el entorno de Arduino, no utilizaba un lenguaje compilado sino uno interpretado llamado LUA.

NodeMCU está basado en el SoC (System on Chip) ESP8266, un chip altamente integrado, diseñado para las necesidades de un mundo conectado. Integra un potente procesador con Arquitectura de 32 bits (más potente que el Arduino Due) y conectividad Wifi.

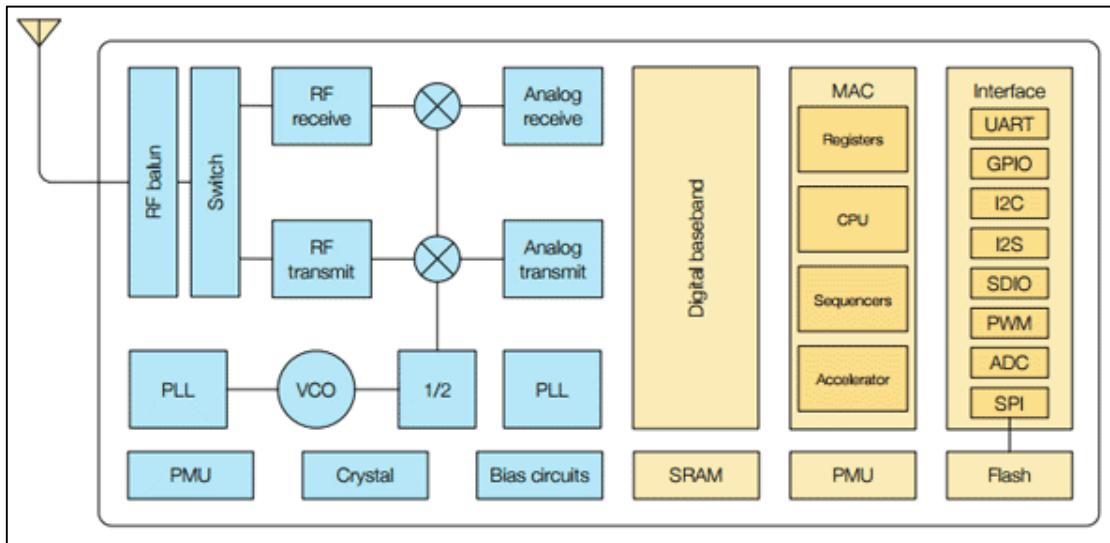
Para el desarrollo de aplicaciones se puede elegir entre los lenguajes Arduino y Lua. Al trabajar con el entorno de desarrollo integrado (IDE) de arduino que es sencillo de utilizar, además de hacer uso de toda la información sobre proyectos y librerías disponibles en internet. La comunidad de usuarios de Arduino es muy activa y da soporte a plataformas como el ESP8266 como el NodeMCU.

NodeMCU es una placa de desarrollo totalmente abierta, a nivel de software y de hardware. Al igual que ocurre con Arduino, en NodeMCU todo está dispuesto para facilitar la programación de un microcontrolador o MCU (del inglés Microcontroller Unit). No hay que confundir microcontrolador con placa de desarrollo. NodeMCU no es un microcontrolador, es una placa o kit de desarrollo que llevan incorporado un chip que se suele llamar SoC (System on a Chip) que dentro tiene un microcontrolador o MCU. El esquema general de este tipo de placas sería el siguiente.



Vamos a partir de la unidad más básica la MCU o microcontrolador. En el NodeMCU este chip se integra dentro del SoC. Como veremos a continuación, a todo este conjunto (SoC) se le conoce como ESP8266.

Por lo tanto, en términos estrictos el ESP8266 no es un microcontrolador. Dentro sí que lleva uno y se llama Tensilica L106 de 32-bit. La MCU se va a encargar de gestionar todas las entradas, salidas y cálculos necesarios para hacer funcionar el programa que hayamos cargado.



El microcontrolador Tensilica L106 funciona con 32-bit lo que viene a decir que puede realizar operaciones con números de ese tamaño (de 0 a 4.294.967.295 o de -2.147.483.648 a 2.147.483.647). Sin embargo, las MCU más comunes son de 8-bit como la que lleva el Arduino UNO el ATmega328P. El microcontrolador Tensilica L106 trabaja a una velocidad de 80MHz aunque puede llegar a los 160MHZ. Esto nos indica la frecuencia con la que la MCU ejecuta las instrucciones.

1.5 Características del NodeMCU: Hardware

NodeMCU integra el ESP8266 en forma de un ESP12E (Uno de los muchos tipos de ESP8266). Este módulo tiene, además del microprocesador, una antena PCB, un LED conectado al GPIO2 y una memoria flash de 16 Mbit (= 4 MB). El ESP12 se alimenta con 3.3 V, pero NodeMCU incluye un regulador de tensión, lo que nos permite alimentarlo por USB con 5V, por ejemplo desde el computador que estamos utilizando para programarlo.

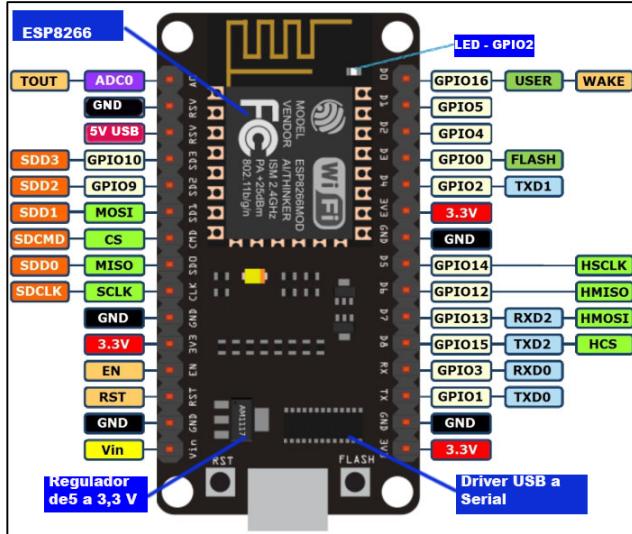
Este regulador tiene un consumo residual de 8mA, lo que hace que esta placa no sea adecuada como módulo definitivo en un proyecto que se requiera un bajo consumo. Aunque esto sea así, el NodeMCU sí se puede utilizar como plataforma de prototipado para aplicaciones de baja energía que luego pasaremos a otras placas con menos componentes y, por tanto, con un consumo de corriente más optimizado.

Cuando se alimenta la placa de desarrollo NodeMCU a través del puerto USB del computador con 5V, internamente tiene un regulador de voltaje que saca 3,3V y 5V (Hablando de la versión 3 del NodeMCU). Los 3,3V se utilizan para alimentar el SoC del NodeMCU y para sacarlo por los 3 pines marcados con ese valor como se muestra en la siguiente figura. Los 5V se utilizan para alimentar otros componentes dentro de la placa y para sacarlos por el pin de 5V.

Hay que tener en cuenta que si alimentamos con 3,3V por alguno de los pines marcados con ese valor, la salida de 5V ya no nos suministrará esos 5V. Recuerde tenerlo en cuenta si en nuestro proyecto hay sensores que se alimentan con 5V.

1.6 Pines de entrada salida

El ESP8266 no tiene memoria flash y por lo tanto necesita de una memoria externa en la que almacenar los programas y datos. Para conectarse a ella, necesita utilizar pines del propio ESP8266.



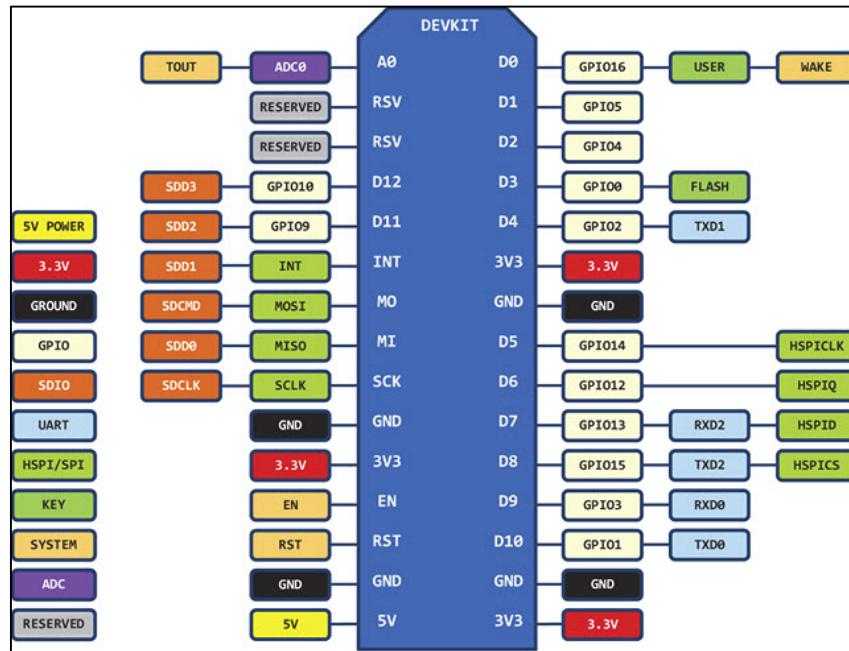
Solo dos pines son accesibles a través de la placa NodeMCU, el pin GPIO9 (D11) y el pin GPIO10 (D12). En la placa aparecerá probablemente como SD2 y SD3. Evita, en la medida de lo posible, utilizar estos pines para conectar sensores y componentes.

Los pines GPIO01 y GPIO03 (D9 y D10) correspondientes al Rx (recibir) y Tx (transmitir) un programa o sketch o para la comunicación entre el NodeMCU y el PC a través del puerto serie. Si se utilizan pueden ocasionar interferencias a la hora de cargar un programa. Estos pines están protegidos con resistencias de 470 Ohm.

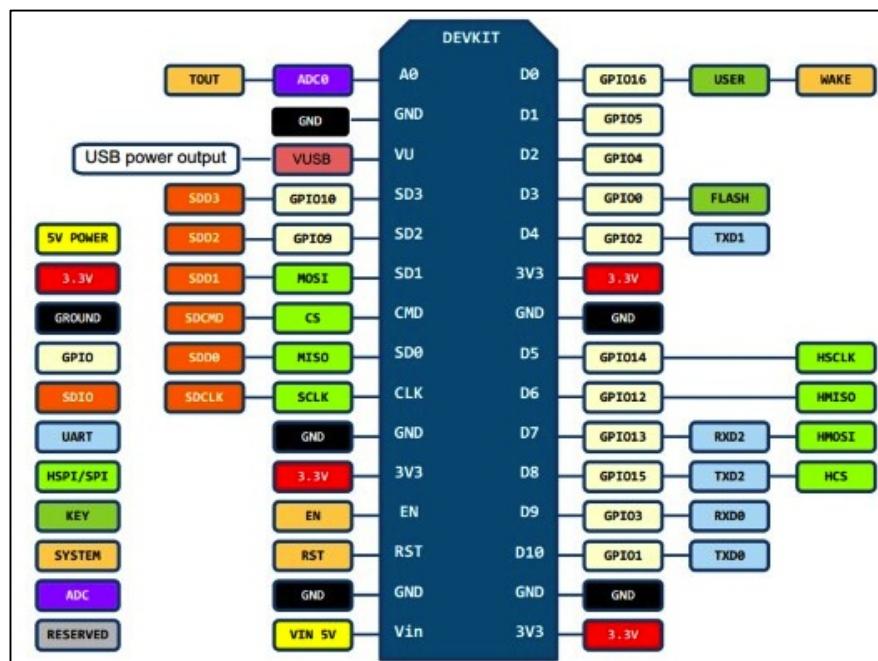
El NodeMCU tiene solo un pin analógico que admite un rango de valores de 0 a 3,3V con una resolución de 10-bit. Esto implica que dentro del código tendremos un valor entre 0 y 1023 que se mapea con el voltaje entre 0 y 3,3V. El ESP8266 tiene un rango de entrada en este puerto de 0 a 1V pero gracias a un divisor de tensión NodeMCU lo extiende de 0 a 3,3V.

La placa tiene 2 botones, uno conectado al pin de RESET y otro al GPIO0, que permite activar el modo de carga de firmware. Normalmente, no es necesario utilizar estos botones si usamos el IDE de Arduino.

El botón de FLASH está conectado al pin GPIO0 que corresponde con el D3 de la NodeMCU. Además tiene una resistencia pull-up lo que hace que en estado de reposo está a estado HIGH (3,3V). Si cargas un programa básico, solo con la función setup() y loop(), y conectas un LED al pin D3, verás como el LED se ilumina.



NodeMCU V1



NodeMCU v3

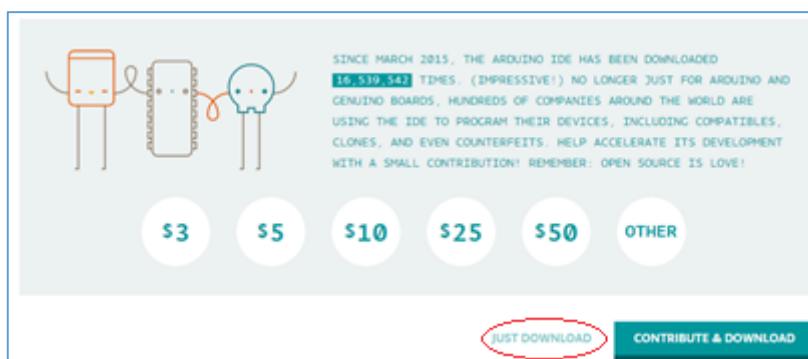
1.7 Instalación del IDE de Arduino.

Para una tarjeta de Arduino, el ESP8266 o el NodeMCU es necesario instalar el entorno de desarrollo o software para programar este dispositivo.

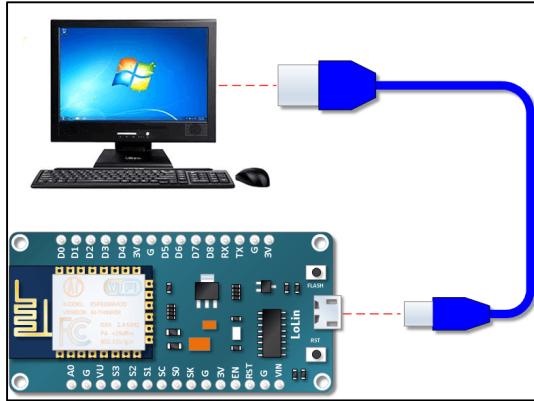
Ingresá a esta página de Arduino (<http://arduino.cc/en/Main/Software>) y elige el sistema operativo donde lo vas a utilizar. Por ejemplo, si lo instalarás en Windows:



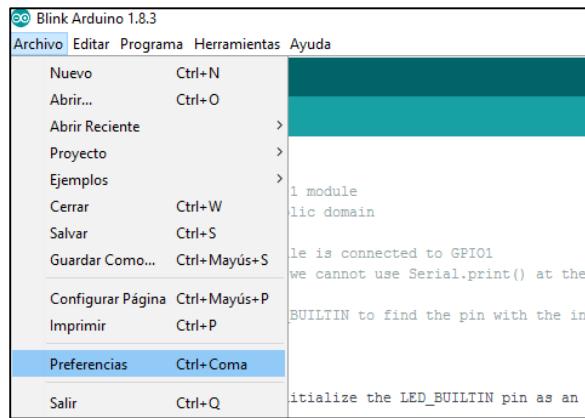
Es posible que en el momento en que instales el IDE (entorno de desarrollo) haya una versión nueva disponible, pero esencialmente los pasos serán los mismos. Descarga el IDE totalmente gratis en JUST DOWNLOAD.



Poé defecto el IDE no trae el ESP8266 por esta razón debemos incluir la librería como se muestra en las siguientes figuras. Para esto abrimos el IDE ya instalado e ir a **Archivo/Preferencias/Gestor de URL's**.



Recuerda conectar y revisar el cable que funcione correctamente.



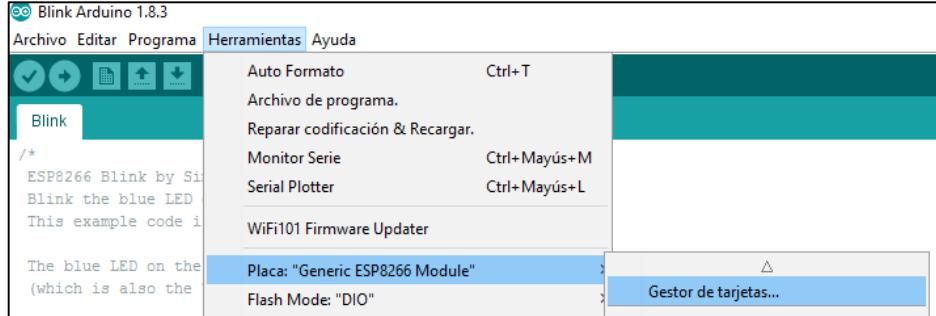
En el Gestor de URL's debes agregar la siguiente dirección:

http://arduino.esp8266.com/stable/package_esp8266com_index.json

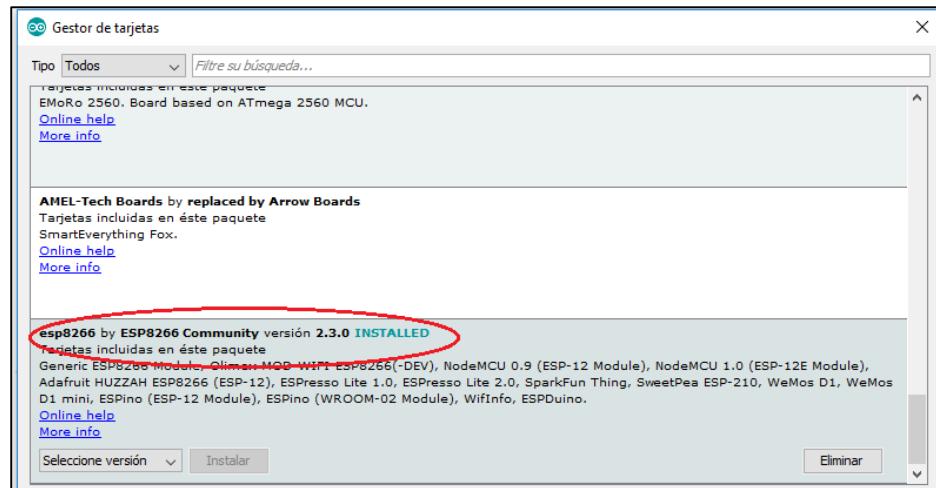


Hacemos click en OK y después abrimos el menú de Herramientas -> Placa -> Gestor de Tarjetas:

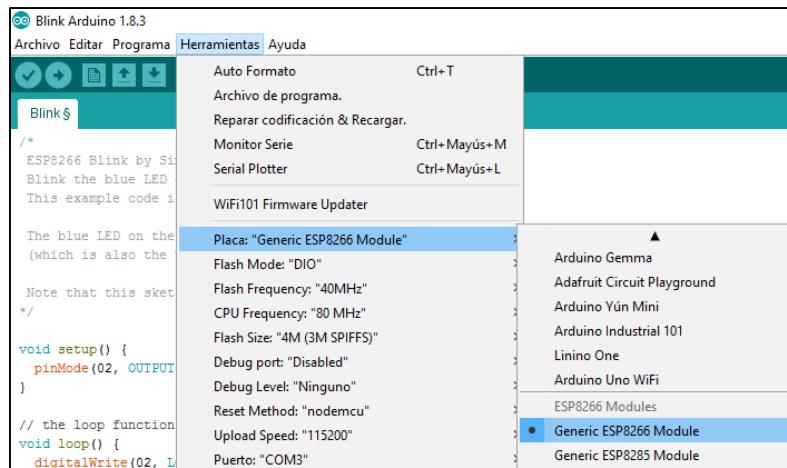
IoT con el NodeMCU - 8



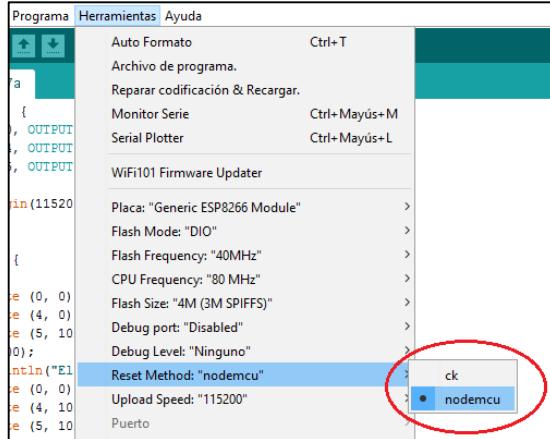
Al final de todo (en la parte de abajo), nos aparece esp8266 by ESP8266 Community. Hacemos click sobre esta opción y click en Instalar:



Ya está instalada la librería y listos para empezar. Verifica y escoge la tarjeta ESP8266 con la que vamos a trabajar.



Por último verifica el modo de reset. Escogemos nodemcu como se ve en la figura.



Para sistemas operativos MAC OS Sierra y Windows 7 el puerto serial algunas veces no reconoce el dispositivo Node MCU. En este caso instale el driver usb en el PC dependiendo del tipo de tarjeta que tenga, por ejemplo para el NodeMCU V3 el circuito integrado conversor de USB a serial es el CH340.

1.8 Practicas iniciales con el NodeMCU.

Puedes iniciar con tus primeras prácticas. Vamos a ayudarnos con el curso en línea de PROMETEC para arduino. Recordemos que el sistema de desarrollo IDE es de arduino y por lo tanto la configuración será exactamente igual. Revisa algunos conceptos de la página de PROMETEC y realiza las primeras prácticas.

Laboratorio 1. Trabajando con el LED y SW integrado.

<http://www.prometec.net/nuestro-primer-programa/>

Laboratorio 2. Primer programa con puertos y la estructura de control if.

<http://www.prometec.net/nuestro-primer-circuito/>

Laboratorio 3. Puertos PWM y la estructura de control for.

<http://www.prometec.net/pines-cuasi-analogicos/>

<http://www.prometec.net/rgb-led/>

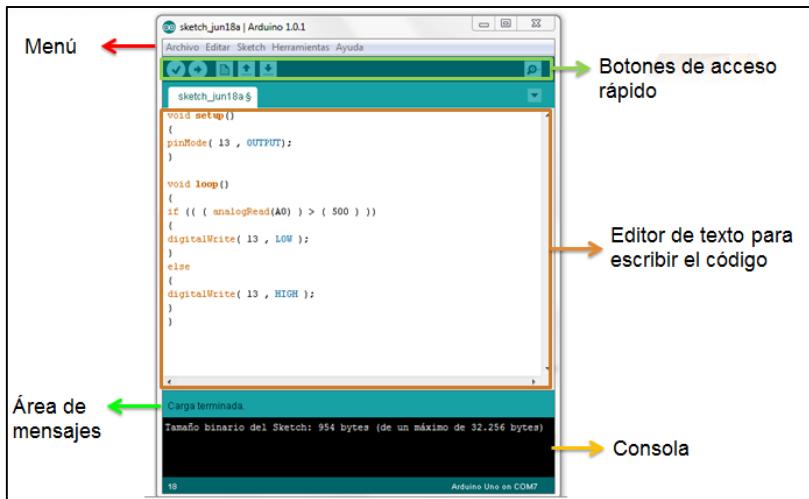
CAPITULO 2

2.0 SISTEMAS EMBEBIDOS EN EL INTERNET DE LAS COSAS.

Arduino proporciona un entorno de programación IDE sencillo y potente para programar el NodeMCU, pero además incluye las herramientas necesarias para compilar el programa y “quemar” el programa ya compilado en la memoria flash del microcontrolador. Además el IDE nos ofrece un sistema de gestión de librerías y placas muy práctico. Como IDE es un software sencillo que carece de funciones avanzadas típicas de otros IDEs, pero suficiente para programar.

2.1 ESTRUCTURA DE UN SKETCH

Un programa de Arduino se denomina sketch o proyecto y tiene la extensión .ino. Para que funcione el sketch, el nombre del fichero debe estar en un directorio con el mismo nombre que el sketch. Las partes principales del IDE de arduino son:



La estructura básica del lenguaje de programación de Arduino es bastante simple y se compone de al menos dos partes. Estas dos partes necesarias, o funciones, encierran bloques que contienen declaraciones, estamentos o instrucciones.

```

void setup() //Primera Parte
{
estamentos;
}
void loop() //Segunda Parte
{
estamentos;
}

```

En donde `setup()` es la parte encargada de recoger la configuración y `loop()` es la que contiene el programa que se ejecutará cíclicamente (de ahí el término `loop` –bucle). Ambas funciones son necesarias para que el programa trabaje.

La función de configuración (`setup`) debe contener la declaración de las variables. Es la primera función a ejecutar en el programa, se ejecuta sólo una vez, y se utiliza para configurar o inicializar `pinMode` (modo de trabajo de las E/S), configuración de la comunicación en serie y otras.

La función bucle (`loop`) siguiente contiene el código que se ejecutara continuamente (lectura de entradas, activación de salidas, etc) Esta función es el núcleo de todos los programas de Arduino y la que realiza la mayor parte del trabajo.

```

/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.

This example code is in the public domain.
*/
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(led, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}

```

introductory comments describe the program

variable declaration section

setup section

loop section

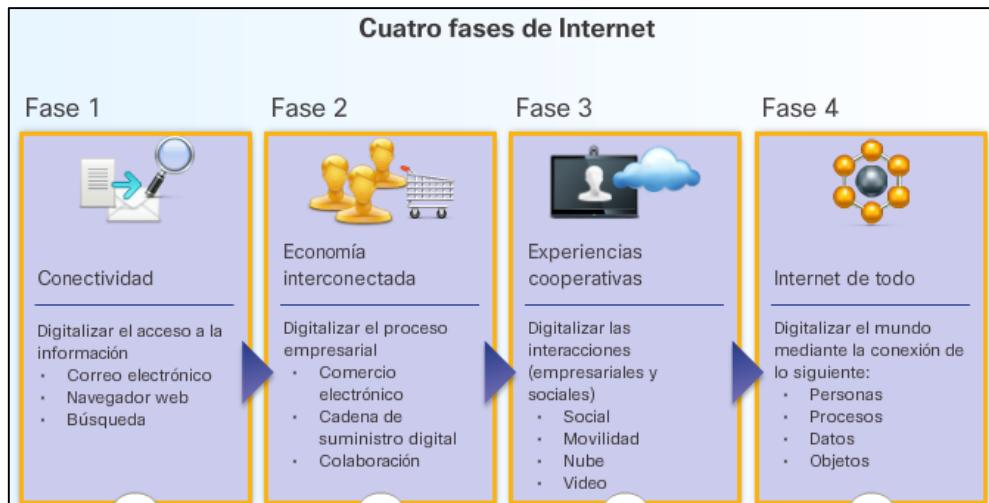
2.2 LOS SISTEMAS EMBEBIDOS Y EL “INTERNET DE LAS COSAS”

Los sistemas embebidos, además de dar la posibilidad de dotar de inteligencia a una gran diversidad de aparatos, están adquiriendo aún más importancia gracias a un concepto conocido como “Internet de las cosas” (IoT por su sigla en inglés, Internet of Things).

Se abre una gran oportunidad para que muchos objetos de uso cotidiano puedan estar dotados de una conexión a Internet para interactuar con el mundo que los rodea y así comunicarse con otros objetos, con diferentes servicios y con sus propietarios y usuarios. En este sentido el IoT representará una gran revolución para el desarrollo de la red ya que los humanos dejarán de ser intermediarios en los canales desde los cuales la red de redes toma sus datos, y permitirá que algunos dispositivos tomen sus decisiones de manera más autónoma.

2.3 EVOLUCIÓN DE INTERNET

La evolución de Internet experimentó cuatro fases distintivas. Cada fase tiene un efecto más profundo en los negocios y en la sociedad que la fase anterior.



La primera fase comenzó hace 20 años y se la denomina “conectividad”. El correo electrónico, la navegación web y la búsqueda de contenido fue solo el principio.

La segunda fase comenzó a fines de la década de los noventa y fue la fase de la “economía interconectada”. Ese fue el comienzo del comercio electrónico y de las cadenas de suministro conectadas digitalmente. Cambió la forma en que hacemos compras y en que las empresas llegan a nuevos mercados.

La tercera fase comenzó a principios de la década de 2000 y se conoce como la fase de las “experiencias cooperativas”. Esta fase se rige por el amplio uso de los medios sociales, la movilidad, los servicios de video y la computación en la nube. Esta fase transformó por completo el mundo laboral.

La fase actual se denomina “Internet de todo” (IdT). En esta fase, se conectan personas, procesos, datos y objetos, lo que transforma la información en acciones que crean nuevas capacidades, experiencias más valiosas y oportunidades sin precedentes.

2.4 PILARES DE INTERNET DE LAS COSAS

IdT incorpora cuatro pilares para lograr que las conexiones en red tengan más importancia y valor que nunca: personas, procesos, datos y objetos.



En la actualidad, la mayoría de las personas se conectan en forma social a través de los dispositivos con conexión a Internet. A medida que IoT evolucione, nos conectaremos de maneras nuevas y valiosas. La ropa y los dispositivos que se llevan puestos ya están cambiando la forma en que nos conectamos.

Los procesos se producen entre todos los demás pilares de IoT. Con los procesos adecuados, las conexiones adquieren más valor. Estas conexiones proporcionan la información correcta, que se entrega a la persona adecuada, en el momento justo y de la manera más pertinente.

Los datos representan la información que generan las personas y los objetos. Cuando se combinan estos datos con el análisis, se proporciona información útil a las personas y a las máquinas. Se toman mejores decisiones y se obtienen mejores resultados.

Los objetos son objetos físicos conectados a Internet y entre sí. Estos dispositivos detectan y recolectan más datos, lo que los hace sensibles al contexto y hace que proporcionen información más empírica para ayudar a las personas y a las máquinas.

2.5 INTERACCIONES DEL IoT

Los pilares interactúan de forma tal que establecen cuatro conexiones principales en el entorno de IoT: personas que se comunican con personas (P2P), máquinas que se comunican con personas (M2P), personas con máquinas (P2M) y máquinas que se comunican con máquinas (M2M).

	PERSONA	MÁQUINA
PERSONA	Comunicaciones Interpersonales P2P Llamadas de voz Mensajería	Comunicaciones de persona a máquina P2M Internet Acceso a contenidos
MÁQUINA	Comunicaciones de máquina a persona M2P Alertas Publicidad	Comunicaciones de máquina a máquina M2M Domótica Telemática

2.6 ¿QUE SON OBJETOS?

Se compone principalmente de varios tipos de computadoras y dispositivos informáticos tradicionales, como equipos de escritorio, computadoras portátiles, smartphones, tablet PC, grandes equipos y clústeres de computadoras. Sin embargo, IdC incluye todos los tipos de objetos, aun los objetos y los dispositivos que no se conectaban tradicionalmente. De hecho, se calcula que, en algún momento del futuro, el 99% de los objetos físicos tendrán conexión.

Estos objetos contienen tecnología integrada para interactuar con servidores internos y con el entorno externo. Además, tienen capacidad de conexión a red y pueden comunicarse mediante una plataforma de red disponible, confiable y segura.

Según la página web Internet World Stats (www.internetworldstats.com), hasta junio de 2012, según las estadísticas había aproximadamente 2400 millones de usuarios de Internet. Esto es solo el 34% de la población mundial total.

En 2012, la cantidad de dispositivos conectados a Internet superó a la población mundial. Esto incluye dispositivos informáticos tradicionales y dispositivos móviles, así como también nuevos dispositivos industriales y de consumo que consideramos "objetos".

Aunque puede parecer que hay demasiados dispositivos conectados a Internet, esto representa menos del 1% de los objetos que podrían conectarse. Entre los dispositivos que actualmente no están conectados, se encuentran los microondas, los despertadores y los sistemas de iluminación.

2.7 SENsoRES

Los sensores son una forma de obtener datos de dispositivos que no son computadoras que convierten los aspectos físicos de nuestro entorno en señales eléctricas que las computadoras pueden procesar. Algunos ejemplos de esto son los sensores de humedad del suelo, los sensores de temperatura del aire, los sensores de radiación y los sensores de movimiento. Todos los tipos de sensores desempeñan una función importante en la conexión de dispositivos que, tradicionalmente, no estaban conectados a IoT.

En sistemas electrónicos, los sensores son los elementos encargados de obtener información. Son llamados técnicamente transductores, y son capaces de convertir cualquier magnitud física, química o biológica en una magnitud eléctrica.

El fenómeno de transducción puede darse de dos formas que se analizan a continuación:

- Activo: la magnitud física a detectar, proporciona la energía necesaria para la generación de la señal eléctrica. Por ejemplo piezoelectrinos o magnéticos.
- Pasivo: cuando la magnitud a detectar se limita a modificar algunos de los parámetros eléctricos característicos del elemento sensor, como ser resistencia o reluctancia.

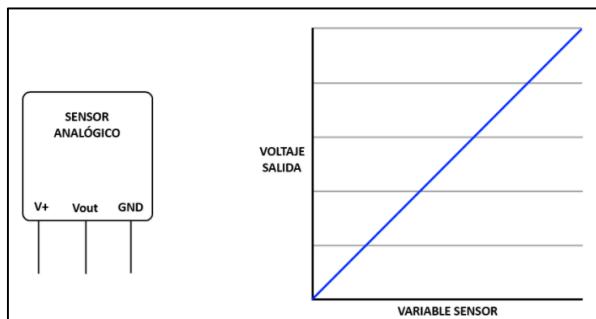
Los sensores se pueden clasificar en función de los datos de salida en:

- Analógicos
- Digitales
- Comunicación por Bus

A la hora de elegir un sensor, debemos leer detenidamente las características y elegir uno que sea compatible con nuestro sistema (tensión y voltaje) y que sea sencillo de usar o nos faciliten una librería sencilla y potente. Recuerde que el NodeMCU tiene un entrada analoga de máximo 3,3V y es transformada a 10 bits.

2.8 Sensores Analógicos y Digitales: Los transductores analógicos proporcionan una señal analógica continua, por ejemplo voltaje o corriente eléctrica. Esta señal puede ser tomada como el valor de la variable física que se mide.

Los sensores analógicos normalmente se componen de tres pins: positivo, masa y salida de voltaje analógica. Esta salida de voltaje es directamente proporcional a la variable de medida del sensor (lineal).



Estas salidas de voltaje analógico se pueden conectar a las entradas analógicas de la tarjeta de desarrollo para poder medir la variable del sensor. Cuando entra un voltaje analógico en un pin analógico del NodeMCU, este hace la conversión de analógico a digital (ADC). Quiere decir que convierte el voltaje de entrada 0-3V en valores enteros comprendidos entre 0-1023.

Para entender este valor, hay que saber que las entradas analógicas del NodeMCU son de 10 bits. Tal como sabemos, un bit puede ser 1 o 0. Recuerda que con 10 bits puedo tener 1024 combinaciones diferentes.

Un sensor analógico es aquel que, como salida, emite una señal comprendida por un campo de valores instantáneos que varían en el tiempo, y son proporcionales a los efectos que se están midiendo; por ejemplo, un termómetro es un dispositivo analógico. La temperatura se mide en grados que pueden tener, en cualquier momento determinado, diferentes valores que son proporcionales a su indicador, o a su "salida" en caso de un dispositivo electrónico.

Un sensor digital en cambio es un dispositivo que puede adoptar únicamente dos valores de salida; 1 -0 encendido o apagado, sí, o no los estados de un sensor digital son absolutos y únicos, y se usan donde se desea verificar estados de "verdad" o "negación" en un sistema automatizado por ejemplo, una caja que es transportada llega al final de un recorrido, y activa un sensor digital; entonces, la señal 0 del sensor en reposo, cambia inmediatamente a 1, dando cuenta al sistema de tal condición.

Los sensores tambíñese pueden clasificar de otras formas como por ejemplo según el tipo de variable medida: Mecánicos – Eléctricos – Magnéticos – Térmicos – Acústicos – Ultrasónicos – Químicos – Ópticos – Radiación – Laser.

2.9 Algunos tipos de sensores.

2.9.1 LDR (Light-Dependent Resistor, resistor dependiente de la luz): Un LDR es un resistor que varía su valor de resistencia eléctrica dependiendo de la cantidad de luz que incide sobre él. Se le llama, también, fotorresistor o fotorresistencia. El valor de resistencia eléctrica de un LDR es bajo cuando hay luz incidiendo en él (en algunos casos puede descender a tan bajo como 50 ohms) y muy alto cuando está a oscuras (puede ser de varios megaohms).



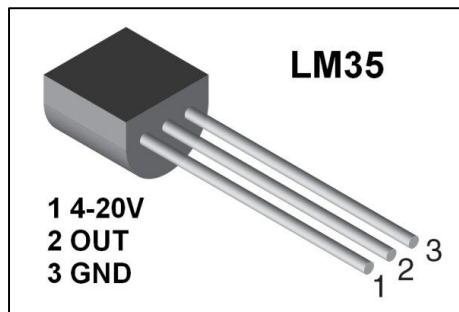
Este sensor es de bajo precio y fácil de adquirir, por eso es muy utilizado en primeras prácticas.

2.9.2 Los fototransistores: no son muy diferentes de un transistor normal, es decir, están compuestos por el mismo material semiconductor, tienen dos junturas y las mismas tres conexiones externas: colector, base y emisor. Por supuesto, siendo un elemento sensible a la luz, la primera diferencia evidente es en su cápsula, que posee una ventana o es totalmente transparente, para dejar que la luz ingrese hasta las junturas de la pastilla semiconductor y produzca el efecto fotoeléctrico.



2.9.3 Sensor de temperatura: Estos sensores se agrupan en cuatro categorías principales: salida de voltaje, salida de corriente, salida de resistencia y salida digital. Con salida de voltaje podemos encontrar los muy comunes LM35 ($^{\circ}\text{C}$) y LM34 ($^{\circ}\text{K}$)

LM35 es un integrado con su propio circuito de control, que proporciona una salida de voltaje proporcional a la temperatura. La salida del LM35 es lineal con la temperatura, incrementando el valor a razón de 10mV por cada grado centígrado. El rango de medición es de -55°C (-550mV) a 150°C (1500 mV). Su precisión a temperatura ambiente es de $0,5^{\circ}\text{C}$.



2.9.4 Sensor de humedad y temperatura.

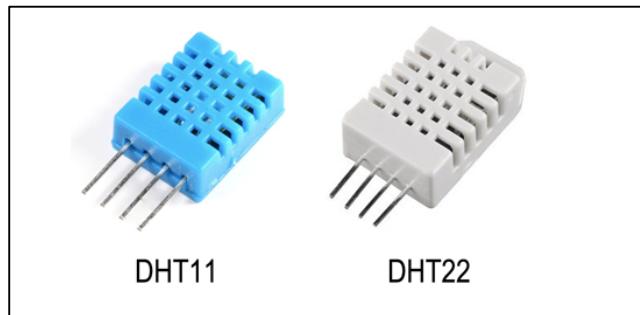
El DHT11 y el DHT22 son dos modelos de una misma familia de sensores, que permiten realizar la medición simultánea de temperatura y humedad, estos sensores disponen de un procesador interno que realiza el proceso de medición, proporcionando la medición mediante una señal digital, por lo que resulta muy sencillo obtener la medición desde el NodeMCU.

El DHT11 presenta una carcasa azul, mientras que en el caso del sensor DHT22 el exterior es blanco. El DHT22 es el modelo superior pero, por contra, tiene un precio superior.

Las características del DHT11 son:

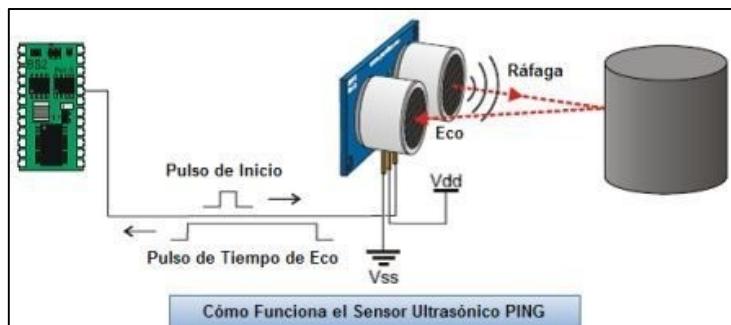
- Medición de temperatura entre 0 a 50, con una precisión de 2°C
- Medición de humedad entre 20 a 80%, con precisión del 5%.
- Frecuencia de muestreo de 1 muestras por segundo (1 Hz)

El DHT11 es un sensor muy limitado que podemos usar con fines de formación, pruebas, o en proyectos que realmente no requieran una medición precisa.



2.9.4 Los medidores ultrasónicos de distancia: que se utilizan en los robots son, básicamente, un sistema de sonar. En el módulo de medición, un emisor lanza un tren de pulsos ultrasónicos y espera el rebote, midiendo el tiempo entre la emisión y el retorno, lo que da como resultado la distancia entre el emisor y el objeto donde se produjo el rebote. Se pueden señalar dos estrategias en estos medidores: los que tienen un emisor y un receptor separados y los que alternan la función (por medio del circuito) sobre un mismo emisor/receptor piezoelectrónico. Este último es el caso de los medidores de distancia incluidos en las cámaras Polaroid con autorango, que se obtienen de desarollo y se usan en la robótica de experimentación personal.

Hay dos sensores característicos que se utilizan en robots: 1. Los módulos de ultrasonido contenidos en las viejas cámaras **Polaroid** con autorango, que se pueden conseguir en el mercado de usados por relativamente poco dinero. 2. Los módulos **SRF** de Devantech, que son capaces de detectar objetos a una distancia de hasta 6 metros, además de conectarse al microcontrolador mediante un bus I2C.

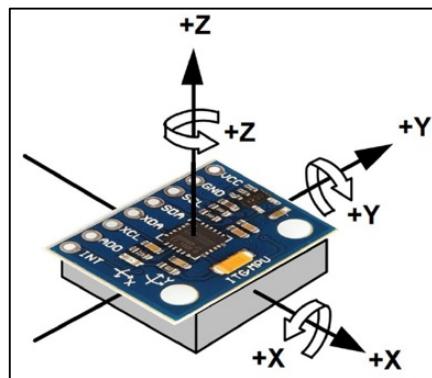


2.9.5 Los sensores de proximidad: que se obtienen en la industria son resultado de la necesidad de contar con indicadores de posición en los que no existe contacto mecánico entre el actuador y el detector. Pueden ser de tipo lineal (detectores de desplazamiento) o de tipo comutador (la comutación entre dos estados indica una posición particular). Hay dos tipos de detectores de proximidad muy utilizados en la industria: inductivos y capacitivos.

Los detectores de proximidad inductivos se basan en el fenómeno de amortiguamiento que se produce en un campo magnético a causa de las corrientes inducidas (corrientes de Foucault) en materiales situados en las cercanías. El material debe ser metálico. Los capacitivos funcionan detectando las variaciones de la capacidad parásita que se origina entre el detector propiamente dicho y el objeto cuya distancia se desea medir. Se emplean para medir distancias a objetos metálicos y no metálicos, como la madera, los líquidos y los materiales plásticos.



2.9.6 Un acelerómetro: es un dispositivo que permite medir el movimiento y las vibraciones a las que está sometido un robot (o una parte de él), en su modo de medición dinámico, y la inclinación (con respecto a la gravedad), en su modo estático. Principalmente existen acelerómetros piezoresistivos, acelerómetros piezoeléctricos y acelerómetros capacitivos.



2.10 Conexión de sensores en el NodeMCU

Los sensores análogos como el LDR o el sensor de temperatura LM35 deben conectarse en la entrada analógicas (A0) del NodeMCU. Solo existe en el NodeMCU, una sola entrada analógica y al igual que un arduino es de 10 bits pero trabaja con un voltaje de referencia interno de 1V. En el NodeMCU la entrada A0 tiene un divisor de voltaje, para adaptar el rango hasta los 3.3V.

Los pines GPIO (de “General Purpose Input/Output”) Entradas y Salidas Digitales es donde conectaremos nuestros sensores para que la placa pueda recibir datos del entorno, y también donde conectaremos los actuadores para que la placa pueda enviarles las órdenes pertinentes, además de poder conectar cualquier otro componente que necesite comunicarse con la placa de alguna manera. Llámese actuador a los dispositivos que brindan la posibilidad de transformar diferentes tipos de energía para generar algún funcionamiento dentro de un sistema automatizado determinado. Usualmente, los actuadores generan una fuerza mecánica a partir de distintos tipos de energía, como puede ser eléctrica, neumática, o hidráulica.

Las salidas analógicas (PWM): En el ESP8266 podemos usar todos sus pines GPIO como salidas de PWM, la resolución del PWM es de 10 bits a diferencia de un arduino que es de 8bits. Generalmente se disponen los pines GPIO para conectar actuadores como motores, luces o dispositivos donde variamos la señal PWM modificando su funcionamiento. Otra diferencia positiva para el ESP8266 es que podemos modificar la frecuencia del PWM, siendo por defecto de 1KHZ.

2.8 Prácticas de manejos de puertos con el NodeMCU.

En esta ocasión las prácticas se realizan con un sensor. Recuerda ayudarte con las prácticas que se encuentran en la página de prometec.

Laboratorio 4. Sensores.

<http://www.prometec.net/actuadores-y-sensores/#>

CAPITULO III

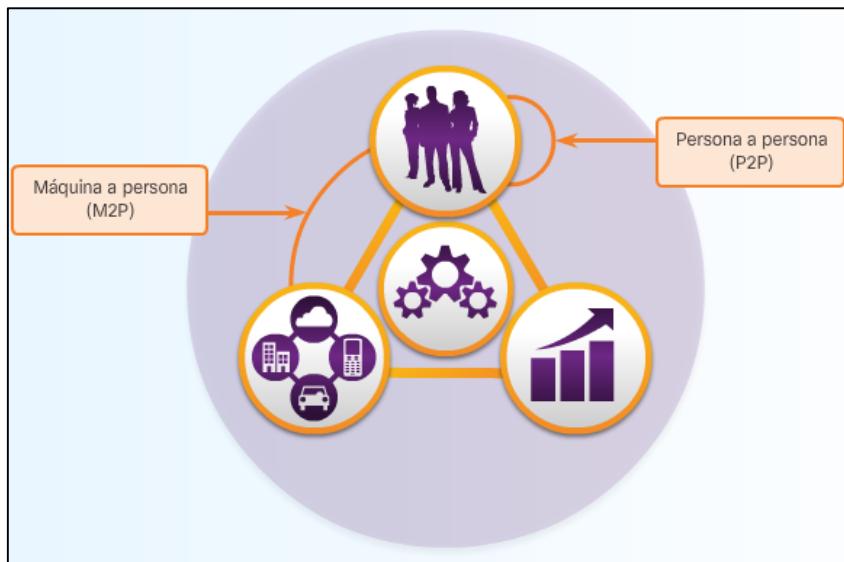
3.1 LAS PERSONAS DEBEN ESTAR CONECTADAS

Las personas son la figura central en cualquier sistema económico: Interactúan como productores y consumidores en un entorno cuyo propósito es mejorar el bienestar satisfaciendo las necesidades humanas. Ya sean las conexiones de persona a persona (P2P), de máquina a persona (M2P) o de máquina a máquina (M2M), todas las conexiones y los datos generados a partir de ellas se utilizan para aumentar el valor para las personas.

3.2 LOS PROCESOS COMO PILARES

Los procesos desempeñan una función fundamental en la manera en que los otros pilares —los objetos, los datos y las personas— operan juntos para ofrecer valor en el mundo conectado de IoT.

Internet revolucionó la manera en que las empresas administran sus cadenas de suministros y la forma en que compran los consumidores. Muy pronto, podremos acceder a detalles de procesos que nunca antes habíamos podido ver. Esto proporcionará oportunidades para hacer que estas interacciones sean más rápidas y simples.



Los procesos facilitan las interacciones entre las personas, los objetos y los datos. En la actualidad, IoT los une mediante la combinación de conexiones de máquina a máquina (M2M), de máquina a persona (M2P) y de persona a persona (P2P)

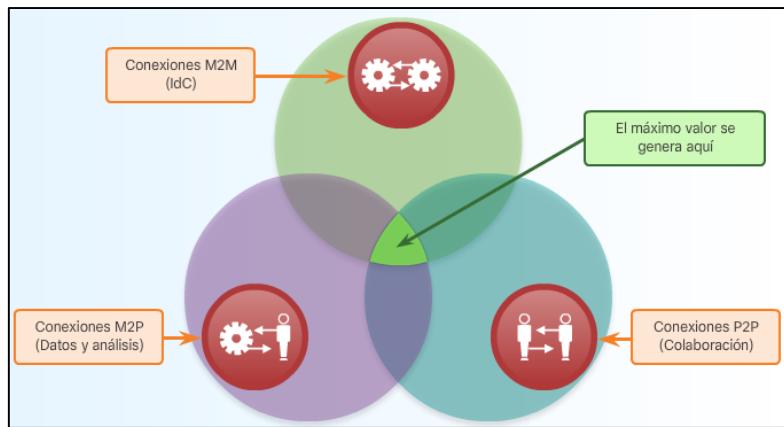
Las conexiones de máquina a máquina (M2M) tienen lugar cuando se transfieren datos de una máquina u “objeto” a otro a través de una red. Las máquinas incluyen sensores, robots, computadoras y dispositivos móviles. Estas conexiones M2M a menudo se denominan “Internet de las cosas”.

Un ejemplo de M2M es un automóvil conectado que emite una señal para informar que un conductor ya casi llega a casa, lo que le indica a la red doméstica que ajuste la temperatura y la iluminación del hogar.

Las conexiones de máquina a persona (M2P) tienen lugar cuando la información se transfiere entre una máquina (como una computadora, un dispositivo móvil o un letrero digital) y una persona, como se muestra en la figura. Cuando una persona obtiene información de una base de datos o realiza un análisis complejo, tiene lugar una conexión M2P. Estas conexiones M2P facilitan el movimiento, la manipulación y la información de datos de máquinas para ayudar a las personas a que tomen decisiones fundadas. Las acciones que las personas realizan según sus razonamientos fundados completan un ciclo de realimentación de IoT.

3.3 CONEXIONES P2P

Las conexiones de persona a persona (P2P) tienen lugar cuando la información se transfiere de una persona a otra. Las conexiones P2P se producen cada vez más a través de video, dispositivos móviles y redes sociales. Con frecuencia, estas conexiones P2P se denominan “colaboración”.



Como se muestra en la ilustración, el valor más alto de IoT se obtiene cuando el proceso facilita la integración de las conexiones M2M, M2P y P2P.

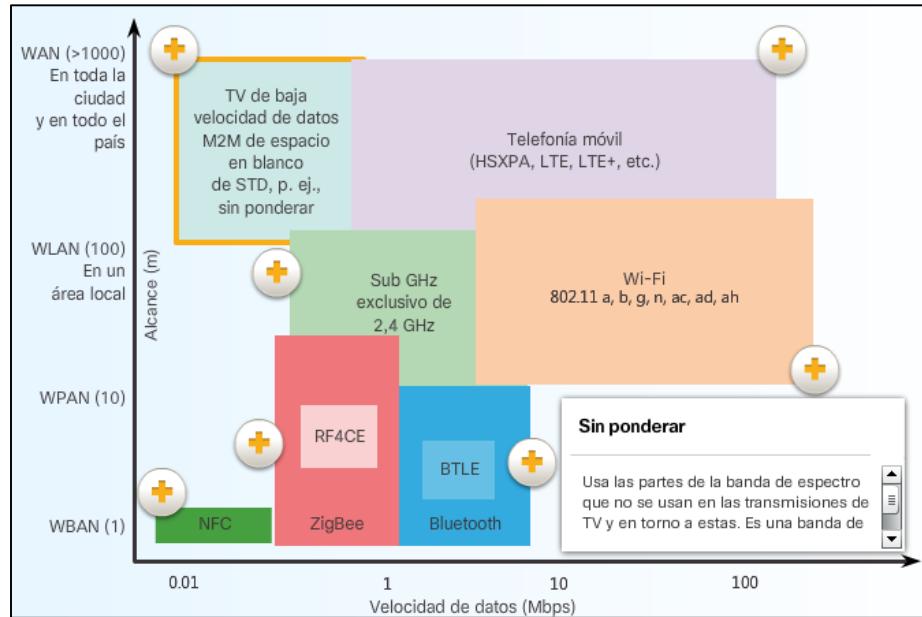
3.4 CONECTIVIDAD ENTRE REDES

En la capa de acceso de red, los dispositivos se pueden conectar a la red en una de dos formas: por cable o de manera inalámbrica.

El protocolo cableado más implementado es el protocolo Ethernet. Ethernet utiliza un paquete de protocolo que permite que los dispositivos de red se comuniquen a través de una conexión LAN cableada. Una LAN Ethernet puede conectar dispositivos con diferentes tipos de medios de cableado.

Actualmente, existen varios protocolos de red inalámbrica disponibles. Las características de estos protocolos varían en gran medida. En la figura siguiente, se proporcionan algunos protocolos inalámbricos comunes y se muestra una representación visual de la ubicación de estos protocolos en el espectro de clasificación. Observe que un protocolo puede abarcar varias clasificaciones.

Además de estos protocolos, hay otros protocolos de capa de acceso de red disponibles en forma inalámbrica y por cable.



3.5 Acceso de red para los objetos actualmente no conectados

Para que los objetos con muy pocos requisitos de energía envíen información a través de la red, existen varios protocolos de comunicación inalámbrica de corto alcance. En algunos casos, estos protocolos no tienen IP habilitado y deben reenviar información a un dispositivo conectado con IP habilitado, como un controlador o una gateway. Por ejemplo, un dispositivo que no usa TCP/IP se puede comunicar con otro dispositivo que no usa este estándar, como el estándar 802.15 del Instituto de Ingenieros Eléctricos y Electrónicos (IEEE). En este estándar se encuentra el conocido Bluetooth y otros como ZigBee.



Bluetooth

El protocolo Bluetooth se suele usar entre dispositivos que están a distancias cortas, como la conexión de un smartphone a auriculares con Bluetooth habilitado, o un teclado inalámbrico con Bluetooth habilitado conectado a un dispositivo informático.

ZigBee

ZigBee es otro ejemplo de un paquete de protocolo 802.15 que usa el emparejamiento entre un origen y un destino específicos. Un ejemplo de esto es entre un sensor para puerta y un sistema de seguridad que envía una alerta cuando se abre la puerta.



NFC

La transmisión de datos en proximidad (NFC) es un estándar para la comunicación entre objetos que están a muy poca distancia, generalmente a pocos centímetros. Por ejemplo, NFC funciona en el punto de venta entre una etiqueta RFID y el lector.

6LoWPAN

6LoWPAN surgió de la necesidad de incluir dispositivos de extremadamente baja energía con capacidades de procesamiento limitadas como parte de IoT; por ejemplo, los medidores inteligentes en una red pequeña.

3.6 WIFI

El wifi es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica. Dentro de estos dispositivos se encuentra la placa o tarjeta de desarrollo NodeMCU.



Wi-Fi es una marca de la Alianza Wi-Fi, la organización comercial que adopta, prueba y certifica que los equipos cumplen con los estándares 802.11 de la IEEE.

Los estándares IEEE 802.11b, IEEE 802.11g e IEEE 802.11n disfrutan de una aceptación internacional trabajando a frecuencias de 2,4 GHz y 5GHz y con velocidades de hasta 11 Mbit/s, 54 Mbit/s y 300 Mbit/s, respectivamente.

En la actualidad ya se maneja también el estándar IEEE 802.11ac, conocido como WIFI 5, que opera en la banda de 5 GHz. La banda de 5 GHz ha sido recientemente habilitada y, además, no existen otras tecnologías (Bluetooth, microondas, ZigBee) que la estén utilizando, por lo tanto existen muy pocas interferencias. Su alcance es algo menor que el de los estándares que trabajan a 2,4 GHz (aproximadamente un 10 %).

Las redes WiFi utilizan la banda ISM (Industrial, Scientific and Medical) que son bandas reservadas internacionalmente para uso no comercial de radiofrecuencia electromagnética en áreas industrial, científica y médica. En la actualidad estas bandas han sido popularizadas por su uso en comunicaciones WLAN (e.g. Wi-Fi) o WPAN (e.g. Bluetooth).

IEEE 802.11ac (también conocido como WiFi 5G o WiFi Gigabit) es una mejora a la norma IEEE 802.11n, se ha desarrollado entre el año 2011 y el 2013, y finalmente aprobada en enero de 2014. El estándar consiste en mejorar las tasas de transferencia hasta 433 Mbit/s por flujo de datos, consiguiendo teóricamente tasas de 1.3 Gbit/s empleando 3 antenas. Opera dentro de la banda de 5 GHz, amplía el ancho de banda hasta 160 MHz (40 MHz en las redes 802.11n), utiliza hasta 8 flujos MIMO e incluye modulación de alta densidad (256 QAM).

Cada red inalámbrica tiene un nombre o SSID (Service Set Identifier) que es una secuencia de 0-32 octetos incluida en todos los paquetes de la red inalámbrica para identificarlos como parte de esa red. El SSID se configura dentro de los dispositivos que se consideran parte de la red, y se transmite en los paquetes. Los receptores ignoran paquetes inalámbricos de redes con un SSID diferente.

Desde junio de 2014 las empresas de microcontroladores iniciaron a implementar sistemas de interconexión WiFi de esta forma Texas Instruments presentó el primer microcontrolador ARM Cortex-M4 con una MCU dedicada Wi-Fi embebida.

Otro ejemplo es el del Arduino MKR1000 que usa el microcontrolador ATSAMD21 (<http://www.atmel.com/devices/ATSAMD21.aspx>) con un módulo wifi y otro de criptoautenticación incluido.

El ESP8266 es un chip Wi-Fi de bajo costo producida por el fabricante chino Espressif Systems, con sede en Shanghai. El chip primero llegó a la atención de los fabricantes occidentales en agosto de 2014 con el módulo ESP-01. Este pequeño módulo permite a los microcontroladores conectarse a una red Wi-Fi y realizar conexiones TCP/IP sencillas.

A finales de octubre de 2014, Espressif lanzó un kit de desarrollo de software (SDK) que permite programar el chip, eliminando la necesidad de un microcontrolador por separado. Desde entonces, ha habido muchos lanzamientos oficiales de SDK.

Otros SDK de código abierto para el ESP8266:

- NodeMCU: un firmware basado en Lua.
- Arduino: un firmware basado en C++. Este núcleo permite que la CPU ESP8266 y sus componentes Wi-Fi sean programados como cualquier otro dispositivo Arduino. El Arduino Core ESP8266 está disponible a través de GitHub: <https://github.com/esp8266/Arduino> y cuyo reference es <https://github.com/esp8266/Arduino/blob/master/doc/reference.md>
- MicroPython: una implementación de Python para dispositivos embebidos a la plataforma ESP8266.
- ESP8266 BASIC: Un intérprete básico de código abierto específicamente diseñado para el Internet de las cosas.
- Mongoose Firmware: Un firmware de código abierto con servicio gratuito en la nube: <https://github.com/cesanta/mongoose-iot>

3.7 El módulo ESP8266

El módulo WIFI ESP8266 incluye toda la electrónica necesaria para la comunicación Radio Frecuencia en la banda WiFi, así como la pila TCP/IP y que se comunica con nosotros a través de un puerto serie.

Dentro de la gran cantidad de usos para este módulo cabe destacar los siguientes:

- Electrodomésticos conectados.
- Automatización del hogar.
- Automatización de la industria.
- Cámaras IP.
- Redes de sensores.
- Wearables.
- IoT (Internet of Things o Internet de las Cosas)
- IIoT (Industrial Internet of Things o Internet de las Cosas para el sector Industrial)

EL ESP8266 no tiene ROM y usa una ROM externa SPI y soporta hasta 16MB y los podemos encontrar en diferentes encapsulados y placas como los de la siguiente figura:



Existen varias placas NodeMCU. Para empezar a comparar las placas NodeMCU que hay en el mercado, en primer lugar, debemos saber que actualmente se está comercializando la SEGUNDA GENERACIÓN (1.0), conocida como V2 ó V3 en función del fabricante. Estas placas integran el procesador ESP8266-12E. La PRIMERA GENERACIÓN (0.9), conocida como V1, integraba el modelo ESP8266-12.

En la actualidad ya está en el mercado la TERCERA GENERACIÓN de placas NodeMcu que integran el nuevo procesador ESP32. El ESP32 no solo es más rápido sino también está diseñado pensando en que sea un microcontrolador para el IoT. En este caso, utiliza un procesador Xtensa Dual-Core LX6 de 32 bits a 160 ó 240 MHz. El usar dos núcleos permite dedicar uno de ellos a la comunicación IP y WiFi y el otro al resto de procesos. Se resuelve así una de las dificultades más importantes que imponía la arquitectura del ESP8266.

Tiene una memoria RAM de 520 kB, accesible por ambos procesadores y puede utilizar memoria RAM externa adicional de hasta 8 MB.

Otras diferencias las podemos apreciar en el siguiente cuadro:

Característica	ESP8266	ESP32
Procesador	Tensilica LX106 32 bit a 80 MHz (hasta 160 MHz)	Tensilica Xtensa LX6 32 bit Dual-Core a 160 MHz (hasta 240 MHz)
Memoria RAM	80 kB (40 kB disponibles)	520 kB
Memoria Flash	Hasta 4 MB	Hasta 16 MB
ROM	No	448 kB
Alimentación	3.0 a 3.6 V	2.2 a 3.6 V
Rango de temperaturas	-40°C a 125°C	-40°C a 125°C
Consumo de corriente	80 mA (promedio). 225 mA máximo	80 mA (promedio). 225 mA máximo
Consumo en modo sueño profundo	20 uA (RTC + memoria RTC)	2.5 uA (10 uA RTC + memoria RTC)
Coprocesador de bajo consumo	No	Sí. Consumo inferior a 150 uA
WiFi	802.11 b/g/n (hasta +20 dBm) WEP, WPA	802.11 b/g/n (hasta +20 dBm) WEP, WPA
Soft-AP	Sí	Sí

3.8 SEGURIDAD EN WIFI

Existen varias alternativas para garantizar la seguridad de estas redes. Las más comunes son la utilización de protocolos de cifrado de datos para los estándares wifi como el WEP, el WPA, o el WPA2 que se encargan de codificar la información transmitida para proteger su confidencialidad, proporcionados por los propios dispositivos inalámbricos. La mayoría de las formas son las siguientes:

- WEP, cifra los datos en su red de forma que sólo el destinatario deseado pueda acceder a ellos. Los cifrados de 64 y 128 bits son dos niveles de seguridad WEP. WEP codifica los datos mediante una “clave” de cifrado antes de enviarlo al aire. Este tipo de cifrado no está recomendado debido a las grandes vulnerabilidades que presenta ya que cualquier cracker puede conseguir sacar la clave, incluso aunque esté bien configurado y la clave utilizada sea compleja.
- WPA: presenta mejoras como generación dinámica de la clave de acceso. Las claves se insertan como dígitos alfanuméricos.
- WPA2 (estándar 802.11i): que es una mejora relativa a WPA. En principio es el protocolo de seguridad más seguro para Wi-Fi en este momento. Sin embargo requieren hardware y software compatibles, ya que los antiguos no lo son. Utiliza el algoritmo de cifrado AES (Advanced Encryption Standard).
- IPSEC (túneles IP) en el caso de las VPN y el conjunto de estándares IEEE 802.1X, que permite la autenticación y autorización de usuarios.
- Filtrado de MAC, de manera que solo se permite acceso a la red a aquellos dispositivos autorizados teniendo en cuenta su dirección MAC. Es lo más recomendable si solo se va a usar con los mismos equipos, y si son pocos.
- Ocultación del punto de acceso: se puede ocultar el el SSID de manera que sea invisible a otros usuarios.

3.9 Potencia en redes WIFI

El dBm (a veces también dBmW o decibelio-milivatio) es una unidad de medida de potencia expresada en decibelios (dB) relativa a un milivatio (mW). Se utiliza en redes WiFi, de radio, microondas y fibra óptica como una medida conveniente de la potencia absoluta a causa de su capacidad para expresar tanto valores muy grandes como muy pequeñas en forma corta.

Los valores aproximados de potencia en una red WiFi en dBm son:

- -40 a -60: señal idónea con tasas de transferencia estables. dependiendo.
- -60: enlace bueno; ajustando TX y basic rates se puede lograr una conexión estable al 80%.
- -70: enlace normal -bajo; es una señal medianamente buena, aunque se pueden sufrir problemas con lluvia y viento.
- -80: es la señal mínima aceptable para establecer la conexión; puede ocurrir caídas, que se traducen en corte de comunicación (pérdida de llamada, perdida de datos, mensajes (sms) corruptos (ilegibles).

3.10 TUNIOT

TUNIOT es un generador de código de bloque para NODEMCU. Usted no necesita ninguna habilidad de codificación para programarlo y hacer su proyecto IoT. La herramienta está disponible en 4 idiomas y está en desarrollo activo. Se basa en la tecnología en bloque. La herramienta genera su código para ser llevado automáticamente al IDE de Arduino

Esta herramienta cuenta con tutoriales desde como instala el IDE de arduino hasta como realizar códigos para poner a trabajar el NodeMCU con proyectos IoT.

En esta página encuentras el tutorial paso a paso para manejarlo:

<http://easycoding.tn/index.php/nodemcu/>

https://www.youtube.com/watch?v=HBSH_x6J1lY&index=2&list=PLfPtpZzK2Z_Qy2ZbbzvWa58cKKOisMUZ1



NODEMCU – ESP8266

THE TOOL

TUNIOT is a bloc code generator for NODEMCU. You don't need any coding skills to program it and make your IoT project. The tool is available on 4 languages and it is in active development. It is based on the blockly Technology. The tool generate C for Arduino code.

You can get start coding by following this [link](#).

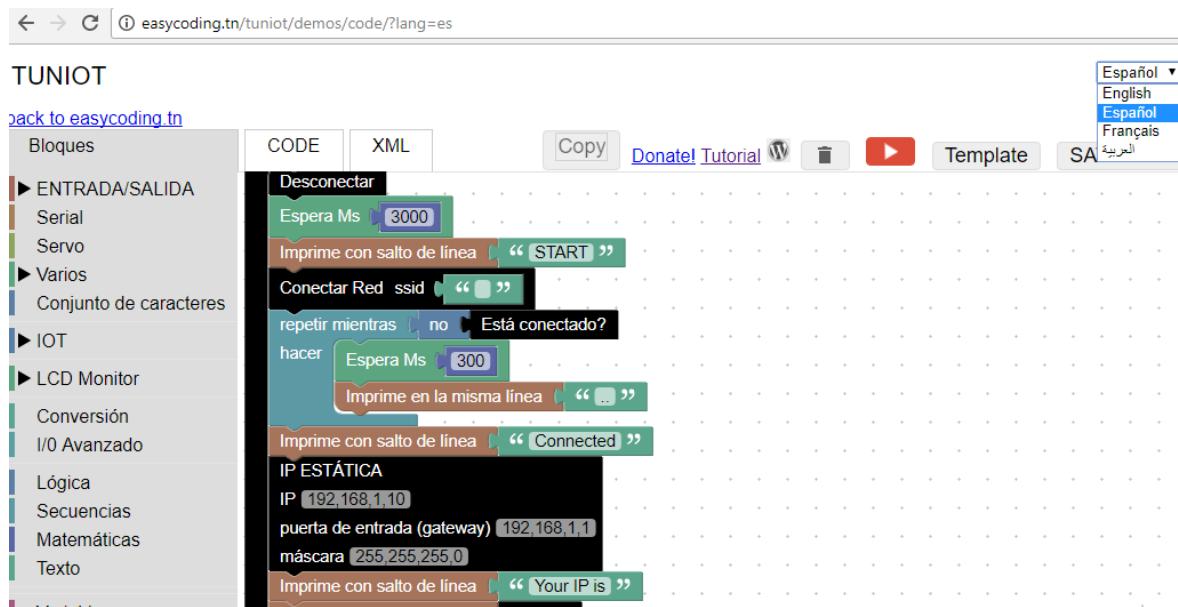
TUTORIALS IN ENGLISH

I made a list of videos available on Youtube. You will be guided step by step. This is an overview of each video with the link.

1. [Software Install](#): In this part, we will see how to install the software required to work with the NODEMCU. We will use the Arduino IDE. And you have to add the ESP8266 board.

Adel Kassah, un profesor de informática en la escuela secundaria en Túnez buscando promover la cultura de programación en su país ha creado el TUNIOT, el cual facilita la programación de la placa NODEMCU o cualquier ESP8266.

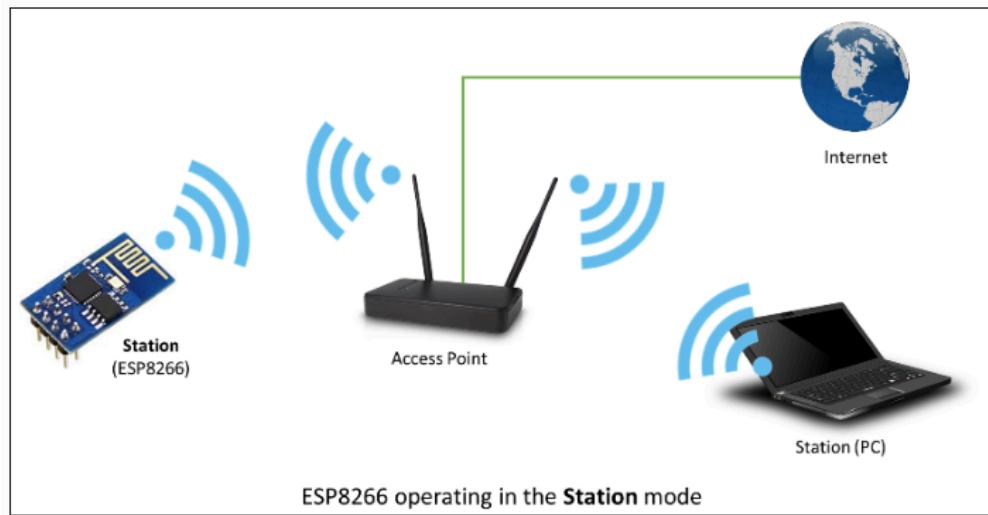
El TUNIOT es una plataforma que sirve para programar en bloques de una forma muy sencilla para después pasarlo al IDE de arduino. Recuerda que este generador de código por bloque es On-line y no necesitas descargar nada pero si debes tener instalado el IDE de arduino con las librerías actualizadas y que reconozca el NodeMCU o ESP8266.



3.11 CONECTANDOSE A UNA RED WIFI MODO ESTACIÓN.

Modo estación (STA): El modo STATION es aquel en el que el dispositivo se conecta a un router para obtener conexión a internet, captando del router los parámetros necesarios para conectarse. El modo Estación (STA) se utiliza para conectar el módulo ESP a una red Wi-Fi establecida por un punto de acceso en el que previamente debemos seleccionar una red visible de nuestro entorno y proporcionar la clave de seguridad.

El comando utilizado en este caso es : **WiFi.begin(ssid, password)**



Recuerde que si configuramos el modulo o placa NodeMCU como cliente tenemos que especificar el identificador de red SSID y la clave o password del Acces point a donde nos deseemos conectar.

WiFi.status() es una función que nos devuelve un entero y que nos muestra el estado de la conexión. Los valores que se pueden obtener son:

- 0 : **WL_IDLE_STATUS** cuando el Wi-Fi está en proceso de cambiar de estado
- 1 : **WL_NO_SSID_AVAIL** en caso de que el SSID configurado no pueda ser alcanzado
- 3 : **WL_CONNECTED** después de establecer una conexión satisfactoriamente
- 4 : **WL_CONNECT_FAILED** si la contraseña es incorrecta
- 6 : **WL_DISCONNECTED** si el módulo no está configurado en el modo de estación
-

Una vez conectados, se puede adquirir por defecto la dirección DHCP que nos coloca el Gateway.

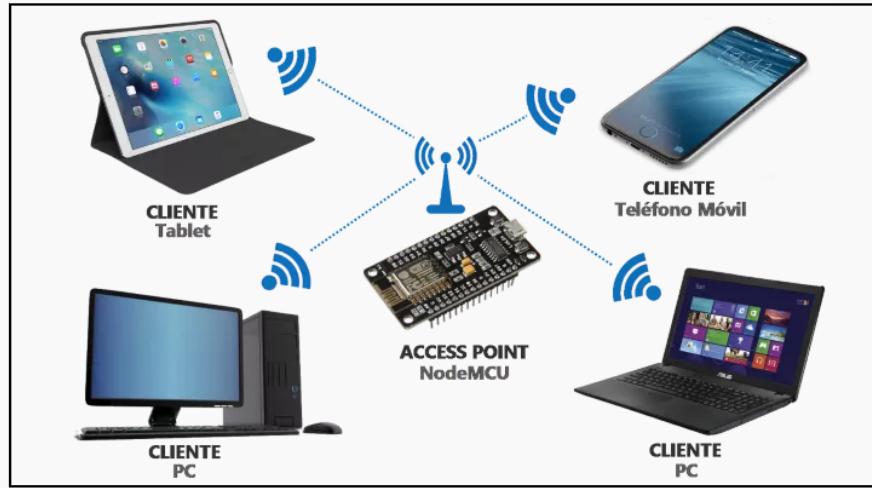
En caso de que se pierda la conexión, ESP8266 se reconectará automáticamente al último punto de acceso utilizado, una vez que esté de nuevo disponible. Lo mismo ocurre en el reinicio del módulo. Esto es posible porque ESP guarda las credenciales del último punto de acceso utilizado en la memoria flash (no volátil).

Para mas información:

<http://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/station-class.html>

3.12 NODEMCU ESP8266 COMO ACCESSSS POINT O PUNTO DE ACCESO

Modo AP: En este modo de operación el modulo NodeMCU puede servir como un punto de acceso y permitir/negar el acceso a la red de datos. Este modo trabaja muy bien incluso tiene varias formas de seguridad de conexión (WPA2 PSK por ejemplo).



En el modo Access Point el NodeMCU difunde un SSID (Service Set Identifier), es decir, el “nombre de red” que se visualiza desde los dispositivos WiFi clientes (salvo que lo ocultemos, por supuesto). La conexión se realiza cuando el NodeMCU autoriza las peticiones de conexión de los clientes.

Por defecto la dirección IP del NodeMCU, que nos sirve para conectar con él desde un navegador Web, es la 192.168.4.1 y asigna a los clientes las siguientes direcciones IP disponibles, 192.168.4.2, 192.168.4.3, etc.

3.13 NODEMCU ESP8266 COMO ESCANER DE REDES

En este caso nuestra placa NodeMCU busca redes WiFi que sea capaz de detectar. De igual forma puedes programar directamente desde el IDE de arduino o hacerlo desde TUNIOT.

Para obtener el número de redes wifi detectadas utilizamos la función en el IDE de arduino **WiFi.scanNetworks()** que devuelve el número de redes encontradas. De igual forma existen comando para visualizar la MAC, la potencia de la red, la MAC del dispositivo detectado y el tipo de seguridad que maneja.

Para obtener información adicional: <http://arduino-esp8266.readthedocs.io/en/latest/esp8266wifi/scan-class.html>

3.14 Comandos WiFi con el IDE de Arduino

WiFi.scanNetworks()

Analiza las redes WiFi disponibles y devuelve el número descubierto. (Modo escaner).

WiFi.begin()

Inicializa la configuración WiFi para conectarse a una red proporciona el estado actual. (Modo estación).

WiFi.softAP()

Para configurar una red como AP (la contraseña debe tener al menos 8 caracteres) WPA2-PSK. (Modo AP).

WiFi.config()

Permite configurar direcciones IP estáticas, cambiar las direcciones DNS y puerta de enlace.

WiFi.SSID()

Obtiene el SSID de la red actual

WiFi.RSSI()

Toma el nivel de señal en potencia dbm en conexión a un Gateway.

WiFi.BSSID()

Toma la MAC del equipo al que está conectado.

WiFi.encryptionType()

Toma el tipo de encriptación utilizado

WiFi.disconnect()

Desconecta de una red WiFi.

WiFi.localIP() si es estación o STA, **WiFi.softAPIP()** si es un punto de acceso o AP.

Toma la IP utilizada localmente.

3.15 COMPARACIÓN DE CÓDIGOS PARA IMPLEMENTACIÓN DE WIFI

ESCANER DE REDES	ACCESS POINT	ESTACIÓN
#include <ESP8266WiFi.h>	#include <ESP8266WiFi.h>	#include <ESP8266WiFi.h>
void setup(){ Serial.begin(9600); Serial.println(); WiFi.disconnect(); delay(100); } void loop (){ Serial.print("Iniciando escaneo ..."); int n = WiFi.scanNetworks(); Serial.println(n); Serial.println("Redes encontradas."); for (int i = 0; i < n; i++) { Serial.println(WiFi.SSID(i)); Serial.println(WiFi.RSSI(i)); delay(500); } Serial.println(); delay(9000); }	void setup(){ Serial.begin(9600); WiFi.softAP("PRUEBA", "ricardo901"); Serial.println((WiFi.softAPIP())); } void loop (){ Serial.println("Estaciones conectadas."); Serial.println((WiFi.softAPgetStationNu m())); delay(3000); }	void setup(){ Serial.begin(9600); delay(1000); WiFi.disconnect(); Serial.println("INICIANDO"); WiFi.begin("D901", "ricardo901"); while((!WiFi.status() == WL_CONNECTED)){ delay(300); Serial.println("...."); } Serial.println("CONECTADO"); Serial.println((WiFi.localIP())); } void loop (){ }

3.16 PRACTICAS DE WIFI CON EL NODEMCU.

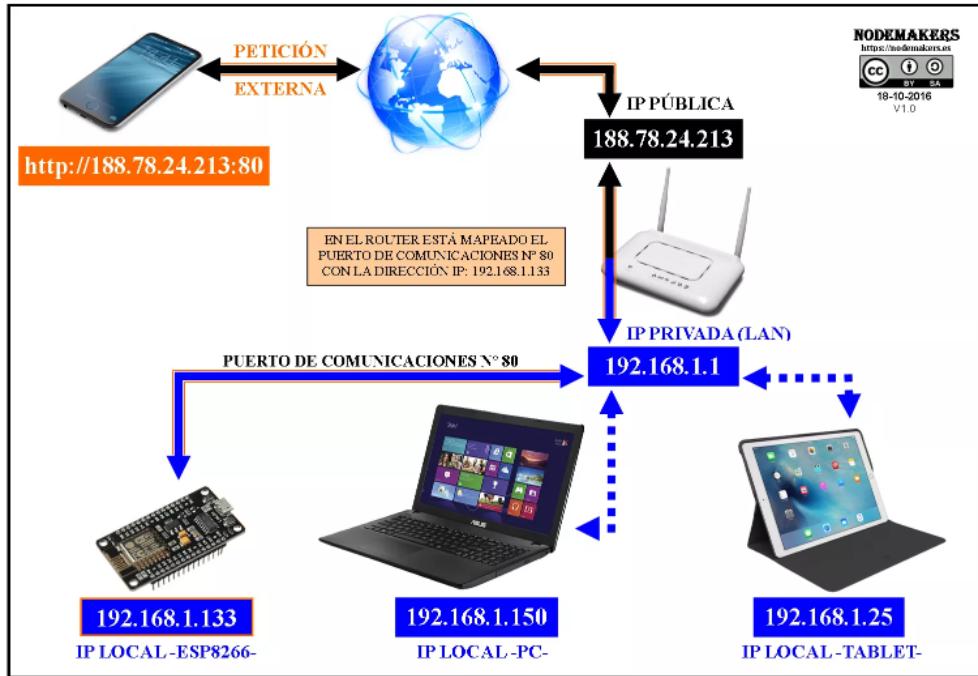
En este capítulo nos enfocaremos a prácticas configuradas para conectarse de alguna forma a WiFi.

Laboratorio 5. Escáner y Estación WIFI.

Laboratorio 6. AP y librería ESP8266WebServer.

CAPITULO IV

4.1 NODEMCU ESP8266 COMO SERVIDOR WEB EN INTERNET



Podemos hacer que el módulo NodeMCU se comporte como un servidor o como clientes, como nuestro pc o móvil, los que inicien la comunicación para obtener o enviarnos datos. Cuando un navegador web (El cliente) quiere acceder a una página web (servidor), envía una petición http (HTTP request) el servidor recibe la petición y le responde con la página web (HTTP response). El encargado de enviar e interpretar esto es el navegador web, el usuario solo escribe la URL y espera a que cargue la página, pero cuando trabajemos con el módulo NodeMCU tenemos que comunicarnos a través de peticiones http.

Recordemos que Hypertext Transfer Protocol o HTTP (en español protocolo de transferencia de hipertexto) es el protocolo de comunicación que permite las transferencias de información en la WWW. Se trata de un protocolo de capa 7 de aplicación.

El comando **WiFiServer server(80);** Crea un servidor que escucha las conexiones entrantes en el puerto especificado. En el caso de un servidor WEB con páginas html el puerto es 80.

server.begin(); Indica al servidor que empiece a escuchar las conexiones entrantes, todo esto funcionando a través del protocolo http.

4.2 Fundamentos de HTML

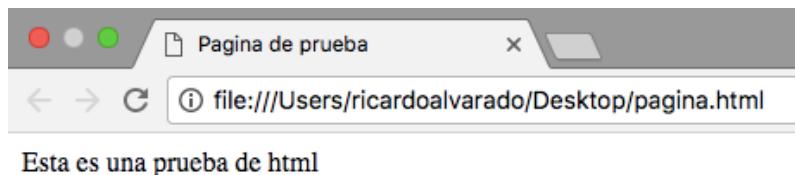
Significa Lenguaje de Marcado para Hipertextos (HyperText Markup Language) es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Es un lenguaje de hipertexto, es decir, un lenguaje que permite escribir texto de forma estructurada, y que está compuesto por etiquetas, que marcan el inicio y el fin de cada elemento del documento.

Un documento hipertexto no sólo se compone de texto, puede contener imágenes, sonido, vídeos, etc., por lo que el resultado puede considerarse como un documento multimedia. Los documentos HTML deben tener la extensión html o htm, para que puedan ser visualizados en los navegadores (programas que permiten visualizar las páginas web).

El cuerpo del documento contiene la información propia del documento, es decir lo que queremos que se visualice, el texto de la página, las imágenes, los formularios, etc. Todos estos elementos tienen que encontrarse entre las etiquetas <body> y </body>, que van justo debajo de la cabecera.

```
<!DOCTYPE html>
<html>
<head>
    <title>Pagina de prueba</title>
</head>
<body>
    Esta es una prueba de html
</body>
</html>
```

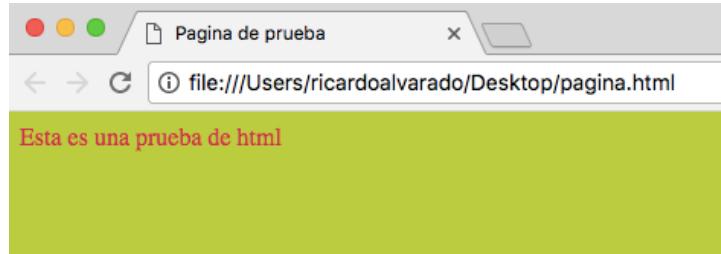
Con este código verás la barra de título del navegador como Pagina de prueba y dentro de la página el contenido: Esta es una prueba de html.



Cada uno de los elementos de la página se encontrará entre una etiqueta de comienzo y su correspondiente etiqueta de cierre, a excepción de algunos elementos que no necesitan etiqueta de cierre, por ejemplo <title> Pagina de prueba </title>.

El color de fondo puede establecerse a través del atributo **bgcolor**, al que es posible asignarle un color representado en hexadecimal o por un nombre predefinido. Ahora podemos cambiar el color de fondo y el color de la letra.

```
<!DOCTYPE html>
<html>
<head>
    <title>Pagina de prueba</title>
</head>
<body bgcolor="BCCC05">
    <font color="CC3350"> Esta es una prueba de html</font>
</body>
</html></html>
```



Para crear una tabla se especifica el número de filas y el número de columnas. Una celda es el resultado de la intersección entre una fila y una columna, por lo que podremos especificar el número de celdas por fila, que equivale a especificar el número de columnas por fila.

Es necesario insertar las etiquetas `<td>` y `</td>` por cada una de las celdas que compongan cada una de las filas de la tabla. Por lo tanto, habrá que insertar esas etiquetas entre las etiquetas `<tr>` y `</tr>`.

Entre las etiquetas `<td>` y `</td>` se podrá especificar el contenido de cada una de las celdas.

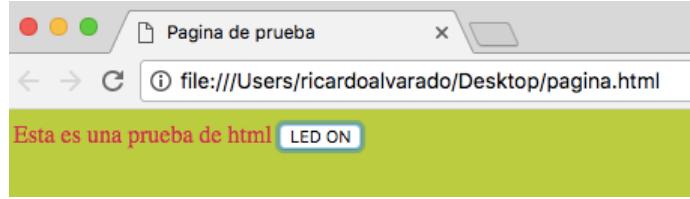
```
<!DOCTYPE html>
<html>
<head>
    <title>Prueba ESP8266</title>
</head>
<body>
MEDIDAS DE TEMPERATURA Y HUMEDAD
<table border="1">
    <tr>
        <td>Temperatura</td>
        <td>Humedad</td>
    </tr>
    <tr>
        <td> XGrados </td>
        <td> % Humedad </td>
    </tr>
</table>
</body>
</html>
```

Con el ejemplo anterior podremos crear una tabla para visualizar los datos de temperatura y humedad tomados con el ESP8266.



Los botones y enlaces son uno de los componentes más utilizados e incluso se podría decir que fundamentales en la programación de cualquier aplicación para un dispositivo móvil. Para utilizarlos simplemente tenemos que añadir la clase CSS `button` a un elemento tipo `button` o a un enlace tipo `a` normal (ambos casos se mostrarán visualmente de la misma forma), por ejemplo añadiendo a nuestro ejemplo anterior:

```
<button class="button">LED ON</button>
```



Los enlaces en HTML se crean mediante la etiqueta `<a>` (su nombre viene del inglés "anchor", literalmente traducido como "ancla"). El atributo más importante de la etiqueta `<a>` es `href`, que se utiliza para indicar la URL a la que apunta el enlace. Cuando el usuario pincha sobre un enlace, el navegador se dirige a la URL del recurso indicado mediante `href`.

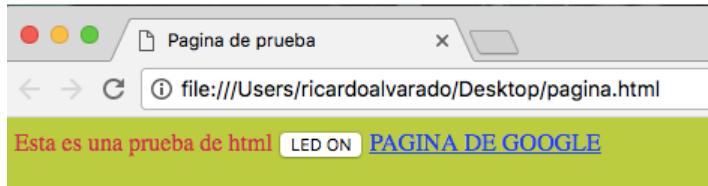
para crear un enlace que apunte a la página principal de Google solamente habría que incluir lo siguiente en un documento HTML:

```
<a href="http://www.google.com">Pagina principal de Google</a>
```

Por ejemplo, añadiendo al código anterior:

```
<!DOCTYPE html>
<html>
<head>
    <title>Pagina de prueba</title>
</head>
<body bgcolor="BCCC05">
    <font color="CC3350"> Esta es una prueba de html</font>
    <button class="button">LED ON</button>
    <a href="http://www.google.com">PAGINA DE GOOGLE</a>

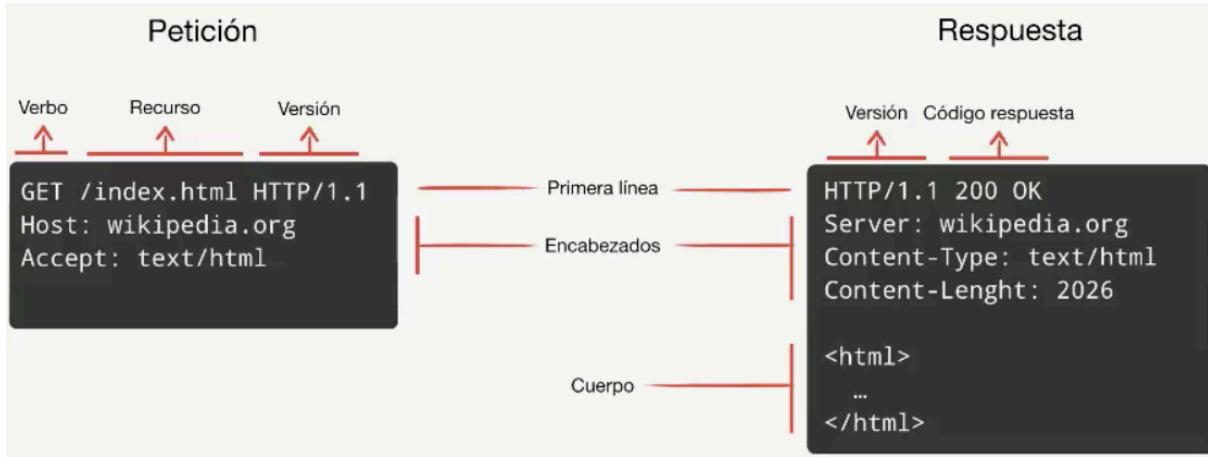
</body>
</html></html>
```



4.3 FUNCIONAMIENTO DE HTTP

HTTP ("HyperText Transfer Protocol") es un lenguaje basado en texto que permite a dos máquinas comunicarse entre sí. HTTP es el término utilizado para describir este lenguaje simple basado en texto. Su estructura básica se divide en un cliente y un servidor. De ahí que debes definir si el NodeMCU será el cliente o el servidor y su código cambiará dependiendo de esto. Recuerda que http es una aplicación y el cliente y servidor puede quedar en un mismo dispositivo.

Los pasos en una conexión http son:



El servidor WEB queda en espera para que un cliente haga una Petición de conexión utilizando el puerto 80. El puerto 80 es el utilizado por el protocolo http y debe ser configurado en el servidor, en este caso en el NodeMCU sería:

ESP8266WebServer server (80);	Servidor a la espera de conexiones
--	------------------------------------

El primer paso en una conexión se hace cuando el cliente hace una petición al servidor a través de un comando GET. Este comando debe ser configurado en el NodeMCU si este es un cliente. Todas las conversaciones en la web comienzan con una petición del usuario o cliente. El cliente envía la petición a un servidor, y luego espera la respuesta.

client_1.print(String("GET /" + " HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection: close\r\n" + "\r\n"));	Cliente envía una petición
---	----------------------------

Segundo paso. Se envía la Respuesta con los datos html del servidor al cliente. La respuesta debe ser configurada en el NodeMCU cuando este es un servidor WEB. La respuesta a la petición GET, el servidor prepara el recurso y lo devuelve en una respuesta HTTP.

WiFiClient client = server.available(); if (!client) { return; } while(!client.available()){ delay(1); } Serial.println((client.readStringUntil('\r'))); client.println("HTTP/1.1 200 OK"); client.println("Content-Type: text/html"); client.println("");	Respuesta del servidor, Primera línea y encabezados
client.println("<!DOCTYPE HTML>"); client.println("<html>"); client.println("BIENVENIDOS"); client.println("</html>");	Cuerpo de la respuesta
client.stop();	Desconexión

La respuesta HTTP contiene el recurso solicitado (en este caso, el contenido HTML de una página web), así como otra información acerca de la respuesta. La primera línea es especialmente importante y contiene el código de estado HTTP (200 en este caso) de la respuesta. En el siguiente cuadro se observa la configuración del NodeMCU como servidor WEB, en modo AP y en modo estación.

ACCESS POINT Y SERVIDOR WEB	ESTACIÓN Y SERVIDOR WEB
<pre>#include <ESP8266WiFi.h> WiFiServer server(80); void setup() { Serial.begin(9600); WiFi.softAP("PRUEBA", "ricardo901"); server.begin(); } void loop(){ WiFiClient client = server.available(); if (!client) { return; } //Chequea que el cliente esté conectado. while(!client.available()){ delay(1); } //Espera que el cliente envíe un dato. Serial.println((client.readStringUntil('\r'))); client.println("HTTP/1.1 200 OK"); client.println("Content-Type: text/html"); client.println(""); client.println("<!DOCTYPE HTML>"); client.println("<html>"); client.println("BIENVENIDOS"); client.println("</html>"); client.stop(); delay(1); }</pre>	<pre>#include <ESP8266WiFi.h> WiFiServer server(80); void setup(){ Serial.begin(9600); delay(1000); WiFi.disconnect(); Serial.println("INICIANDO"); WiFi.begin("D901", "ricardo901"); while((!WiFi.status() == WL_CONNECTED)){ delay(300); Serial.println("..."); } Serial.println("CONECTADO"); server.begin(); Serial.println((WiFi.localIP())); } void loop(){ WiFiClient client = server.available(); if (!client) { return; } while(!client.available()){ delay(1); } Serial.println((client.readStringUntil('\r'))); client.println("HTTP/1.1 200 OK"); client.println("Content-Type: text/html"); client.println(""); client.println("<!DOCTYPE HTML>"); client.println("<html>"); client.println("BIENVENIDOS"); client.println("</html>"); client.stop(); delay(1); }</pre>

4.4 LIBRERÍA “ESP8266WebServer”

La librería ESP8266WebServer se utiliza para simplificar la creación de un servidor web. Para explicar su funcionamiento vamos a realizar un ejemplo que encenderá y apagará un led utilizando un navegador web. El led estará conectado al puerto D0 o GPIO16.

El código completo del programa es el siguiente:

```
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>

const int LED = 16; //Puerto conexión LED
ESP8266WebServer server(80);
```

```

void led_on(){
  digitalWrite(LED,HIGH);
  server.send(200, "text/plain", "LED encendido");
}
void led_off(){
  digitalWrite(LED,LOW);
  server.send(200, "text/plain", "LED apagado");
}
void info() {
  server.send(200, "text/plain", "Hola mundo!");
}
void no_encontrado() {
  server.send(404,"text/plain","Error en la petición");
}
void setup() {

  Serial.begin(115200);           //inicializa el puerto serie
  delay(10);

  pinMode(LED,OUTPUT);          //inicializa el led
  digitalWrite(LED,LOW);         //Inicializa el módulo wifi
  WiFi.mode(WIFI_STA);          //Establece el módulo como cliente wifi
  WiFi.disconnect();             //Se desconecta de cualquier WiFi conectado previamente
  Serial.println();
  Serial.print("Connecting to "); //conecta con la red wifi
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) { // Espera por una conexión WiFi
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());
  digitalWrite(LED,HIGH);

  server.on("/on",led_on);      //definimos los paths
  server.on("/off",led_off);
  server.onNotFound(no_encontrado);

  server.begin();               //inicializa el servidor web
  Serial.println("Servidor HTTP activo");
}
void loop() {
  server.handleClient();
}

```

Creamos un objeto servidor WEB que escuchará el puerto 80:

```
ESP8266WebServer server(80);
```

Con server.on indicamos una función como respuesta del servidor cuando el navegador solicita cierta URL. El ESP8266 nos permite definir comportamientos diferentes para “subdirectorios” de la URL, y así podríamos configurar diferentes respuestas para “/sub-url-1”, “/sub-url-2”, etc.:

```
server.on("/",info);
server.on("/on",led_on);
server.on("/off",led_off);
server.onNotFound(no_encontrado);
```

También podemos introducir el código directamente en vez de llamar a una función. Por ejemplo:

```
server.on("/on",[ ]() {
  digitalWrite(LED,HIGH);
});
```

Cuando el navegador se conecte al ESP8266 se va a ejecutar ese código que hemos puesto dentro de server.on, y que en el caso de que la URL apunte a la raíz (/) es una llamada a la función server.send, que lo que hace es responder al navegador. Para ello, hay que darle tres argumentos:

- Código de respuesta: es un código que se utiliza en el protocolo HTTP para indicar diferentes eventos en las conexiones. Posiblemente te suene el 404 cuando das con un enlace roto, pues el código 200 es menos visible pero más común, es el que se devuelve cuando sí que se encuentra el contenido.
- MIME-type: es un código que indica el tipo de contenido que se devuelve, puede ser texto, imagen...
- Respuesta: El contenido de la respuesta en sí, en este caso tan solo queremos responder un "Hola mundo!" de prueba.

```
void info() {
  server.send(200, "text/plain", "Hola mundo!");
}
```

El método server.onNotFound() definiremos la función que queremos que actúe cuando la petición que recibimos no la tenemos definida. En este caso, vamos a llamar a la función no_encontrado() que hemos definido anteriormente.

Para iniciar el servidor web:

```
server.begin();
```

En loop() escuchamos las conexiones entrantes:

```
server.handleClient();
```

Laboratorio 7. Visualización de datos de sensor en Servidor WEB.

Laboratorio 8. Controlando dispositivos desde servidor WEB.

CAPITULO V

LOS DATOS COMO PILAR DEL IDT

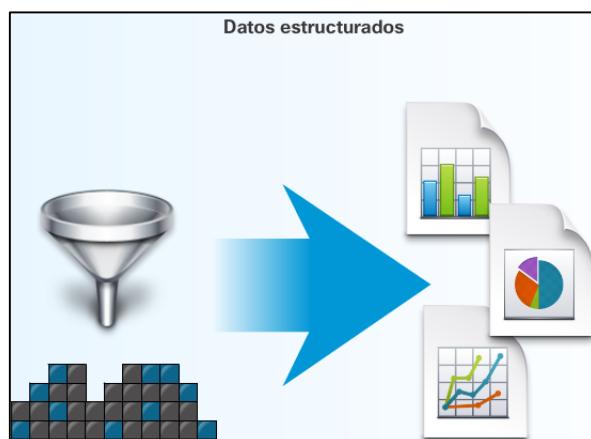
5.1 ¿Qué SON LOS DATOS?

Los datos son un valor asignado a todo lo que nos rodea; están en todas partes. Sin embargo, por sí solos, los datos no tienen sentido. Los datos se vuelven más útiles al interpretarlos, por ejemplo, mediante la correlación o la comparación. Esos datos útiles ahora son información. Cuando se la aplica o se la comprende, esa información se convierte en conocimiento.

En la comunicación electrónica, los datos se representan como unos y ceros. Estos elementos diferenciados se conocen como bits (o dígitos binarios). Todos los datos electrónicos se almacenan en este formato binario digital. Mientras que los seres humanos interpretan palabras e imágenes, las computadoras interpretan patrones de bits.

5.2 DATOS ESTRUCTURADOS

Los datos estructurados son aquellos que se introducen y se mantienen en campos fijos dentro de un archivo o un registro. Las computadoras introducen, clasifican, consultan y analizan datos estructurados con facilidad. Por ejemplo, cuando envía su nombre, dirección y datos de facturación a un sitio web, crea datos estructurados. La estructura obliga al uso de cierto formato para la introducción de los datos, a fin de minimizar los errores y hacer que sea más fácil para la computadora interpretarlos. La figura siguiente representa el almacenamiento de diferentes tipos de datos en ubicaciones específicas para que los programas informáticos puedan ubicarlos.



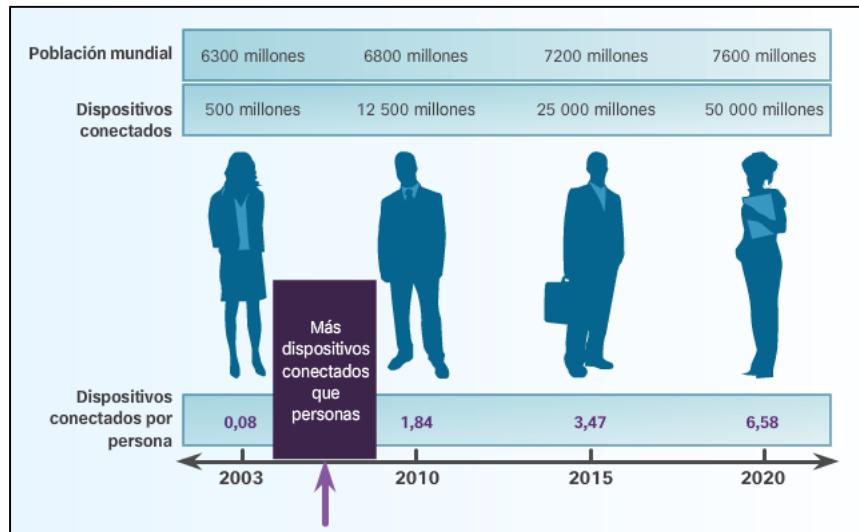
Los datos no estructurados carecen de la organización de los datos estructurados; son datos sin procesar. No tienen la estructura que identifica el valor de los datos. No hay un método fijo para introducir o agrupar los datos no estructurados y, luego, analizarlos. Algunos ejemplos de datos no estructurados incluyen el contenido de fotos y archivos de audio y de video. En la figura siguiente, se muestra La escuela de Atenas, de Rafael. No puede realizarse una búsqueda del contenido, como las figuras y los elementos de la pintura, porque no tiene estructura.



Los datos estructurados y no estructurados son recursos valiosos para las personas, las organizaciones, las industrias y los gobiernos. Al igual que otros recursos, la información recolectada a partir de datos estructurados y no estructurados tiene un valor mensurable. Sin embargo, el valor de esos datos puede aumentar o disminuir, según la forma en que se administren. Incluso los mejores datos pierden valor con el tiempo.

5.3 CONEXIONES

¿Por qué tanta preocupación por los datos? El volumen de datos que, hace una década, se producía en un año, ahora se produce en una semana. Eso equivale a la producción de más de 20 exabytes de datos por semana. A medida que más objetos no conectados se conectan, los datos siguen creciendo exponencialmente.



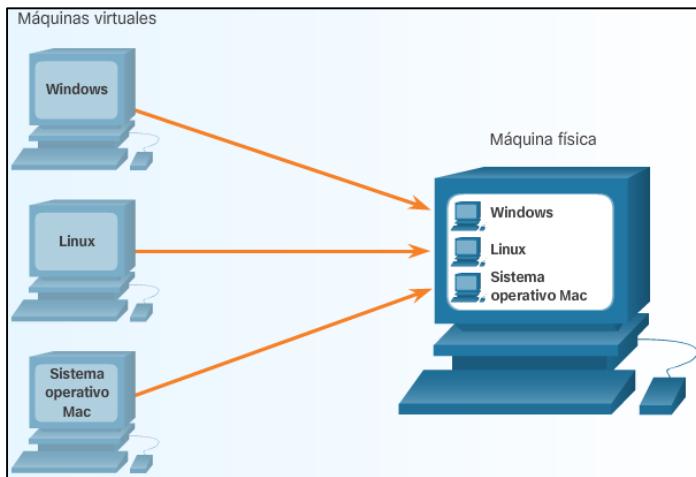
5.4 ADMINISTRACIÓN DE DATOS MASIVOS (BIG DATA)

Nuevos dispositivos se conectan a Internet a diario y generan una gran cantidad de contenido nuevo.

Con esta cantidad de información, las organizaciones deben aprender cómo administrar los datos y también cómo administrar los “datos masivos”. Los datos masivos tienen tres dimensiones principales que es necesario explicar: volumen, variedad y velocidad.

5.5 VIRTUALIZACIÓN

Históricamente, cada computadora tiene su propio sistema operativo, aplicaciones y componentes de hardware exclusivos. Ahora, mediante la emulación de software, es posible ejecutar varios equipos virtuales en una sola computadora física. Esto significa que cada equipo virtual tiene su propio sistema operativo, aplicaciones y componentes de hardware exclusivos. Esto se conoce como “virtualización” en el ámbito de la informática. Como se muestra en la ilustración, cada máquina virtual opera en forma independiente.

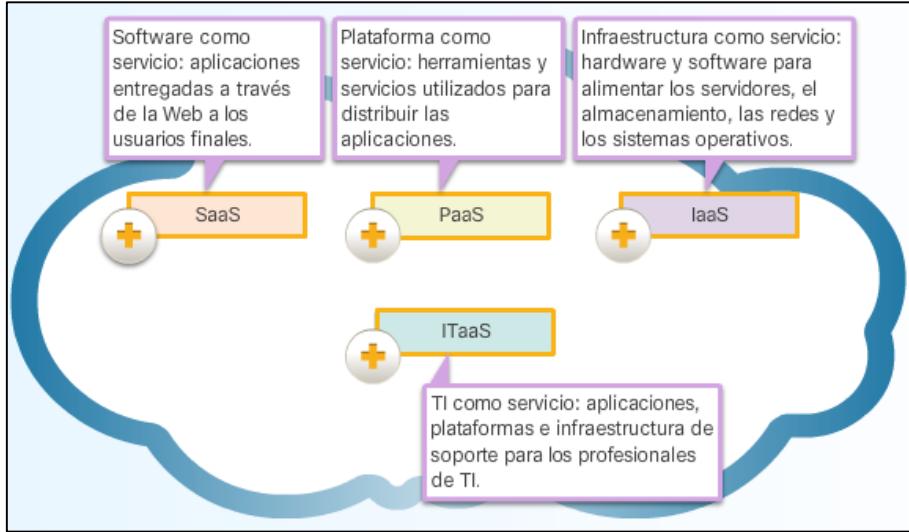


En el mundo empresarial, una sola infraestructura física puede ejecutar varias infraestructuras virtuales. Mediante la virtualización de los servidores y las redes, las empresas pueden reducir los costos operativos y administrativos. Los ahorros operativos pueden provenir de la reducción de la cantidad de máquinas físicas y de las necesidades de energía y refrigeración. Para admitir más aplicaciones, se puede agregar un servidor virtual.

También puede utilizar la virtualización para sus necesidades informáticas personales. Puede probar un nuevo sistema operativo en su computadora sin dañar el sistema actual o puede navegar por Internet de forma segura en la máquina virtual. Si se produce algún problema, la máquina virtual se puede eliminar.

5.6 COMPUTACIÓN EN LA NUBE

La computación en la nube implica una gran cantidad de computadoras conectadas a través de una red. Los proveedores de computación en la nube dependen en gran medida de la virtualización para ofrecer sus servicios. También puede reducir los costos operativos mediante un uso más eficiente de los recursos. Estas empresas proporcionan cuatro categorías distintas de servicios. Haga clic en las categorías de la ilustración para obtener más información.



La computación en la nube es otra forma de administrar y almacenar datos y obtener acceso a ellos.

La computación en la nube permite que los usuarios obtengan acceso a sus datos en cualquier momento y lugar. Si utiliza servicios de correo electrónico basados en Web, es probable que ya esté utilizando ciertas formas de computación en la nube.

La computación en la nube también permite a las organizaciones simplificar sus operaciones de TI mediante el contrato de abonos solo a los servicios necesarios. Con la computación en la nube, las organizaciones también pueden eliminar la necesidad de equipos de TI, mantenimiento, y administración en el sitio. La computación en la nube reduce los costos para las organizaciones: reduce los costos de equipos y de energía, los requisitos físicos de la planta y las necesidades de capacitación del personal de soporte técnico.

5.7 Adquisición de datos

EL NodeMCU es una herramienta perfecta para la recogida de todo tipo de datos en cualquier lugar y condición, comprobación, pre-procesado y normalización de los datos y finalmente transporte de esos datos de una forma rápida y segura al lugar donde se van a almacenar para posteriormente ser procesados y analizados.

La tarjeta de desarrollo ESP8266 o el NodeMCU es capaz de recoger datos, tratarlos en tiempo real (limitado por su capacidad de procesamiento) y comunicarse con el sistema de almacenamiento de datos.

Al ser NodeMCU un elemento barato y con muchas capacidades de comunicación es posible usarlo en la captura de datos distribuidos y desplegar tantos sensores como sea necesario creando una red de sensores fácilmente escalable.

La adquisición de datos (DAQ) es el proceso de medir un fenómeno eléctrico o físico como voltaje, corriente, temperatura, presión o sonido. Un sistema DAQ consiste de sensores, hardware de medidas DAQ y un PC donde almacenar y tratar los datos. Comparados con los sistemas de medidas tradicionales, los sistemas DAQ basados en PC aprovechan la potencia del procesamiento, la productividad, la visualización y las habilidades de conectividad de los PCs estándares en la industria proporcionando una solución de medidas más potente, flexible y rentable.

Un sistema DAQ básico sería un NodeMCU que recoge los datos, los procesa y los guarda en una tarjeta SD o un PC conectado, pero con NodeMCU podemos ir más allá y crear una red de sistemas DAQ

interconectados que procesan los datos que capturan y los mandan a una base de datos o repositorio único o distribuido.

A la hora de recoger datos para su procesamiento debemos responder estas preguntas:

- ¿Qué quieres medir? – Sensores
- ¿Cómo lo quieres conectar? – Comunicaciones /Protocolos
- ¿Dónde vas a almacenar los datos? – Plataformas
- ¿Qué quieres hacer con los datos? – Herramientas de procesado

A modo de resumen, estos son los elementos:

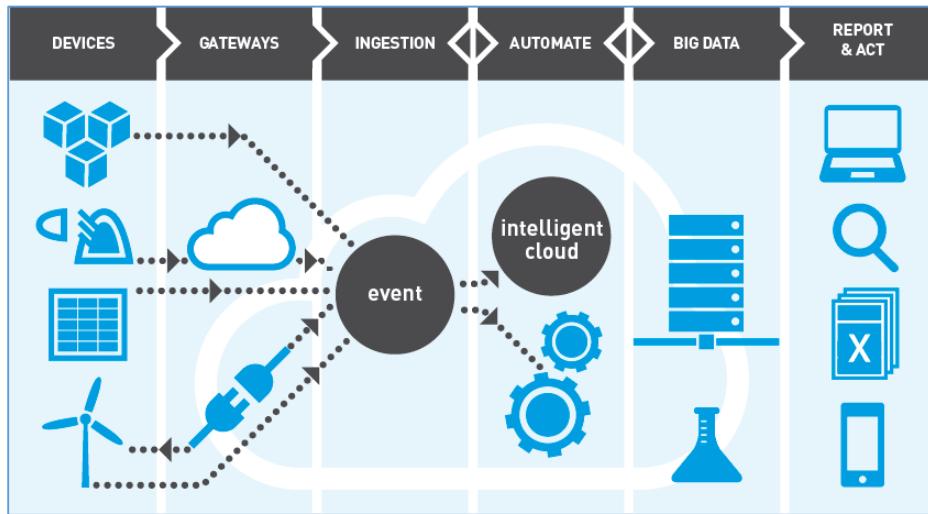
Sensor — MCU — Comunicación — Protocolo — Plataforma — Servicios

Uno de los retos del IoT es mandar datos de cualquier sensor a través de cualquier protocolo a cualquier plataforma de forma inalámbrica y usando la menor energía posible (baterías) y todo esto mediante una comunicación segura (cifrada).

- **Dispositivos Hardware:** (por ejemplo ESP8266), es el dispositivo con el que vamos a medir (sensor) o interactuar (actuador, p.e. bombilla).
HW IoT low cost con ESP8266: <https://en.wikipedia.org/wiki/ESP8266>
- **Conectividad.** Medio de comunicación, como vamos a comunicar el HW, ya sea por red o de forma inalámbrica. Ethernet, Wifi, GPRS, LPWAN, zigbee, bluetooth, ANT+, etc...
Más sobre comunicaciones Arduino:
- **Protocolos de comunicación,** lenguaje para comunicar el HW y el SW. HTTP, fiware, MQTT, API REST,...
- **Plataformas Software,** almacenar y tratar los datos recogidos por nuestros sensores. Pueden ser plataformas de terceros o plataformas propias desarrolladas por nosotros o simplemente guardar en BBDD. Por ejemplo: Carriots, Thingspeak, Temboo, Thinger, etc...
Además todas estas plataformas SW que están en la nube, deben estar soportadas por un HW de servidores, unas BBDD de gran capacidad y una infraestructura segura que los hospede.
- **Servicios,** son los servicios que ofrecen las plataformas como la visualización de los datos recogidos, análisis de los datos, envío de avisos cuando se detecte un evento, interconexión con otras plataformas, etc...
Servicios ofrecidos por la plataforma carriots: <https://www.carriots.com/que-es-carriots>

5.8 Cadena de valor de IoT:

Recoger datos — conectar — almacenar — analizar — mostrar — actuar — predecir

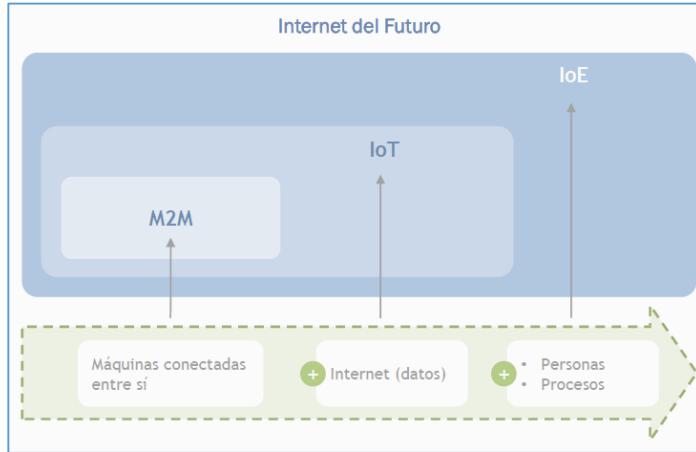


Cuando creamos proyectos IoT con el NodeMCU, tenemos varias maneras de monitorizar los datos que captamos a través de los sensores. La más básica y que requiere tener el dispositivo conectado a un ordenador es a través del monitor serie. Si tenemos algún display como un LCD o una pantalla TFT, ya podremos desconectar el Arduino del ordenador y llevarlo a cualquier sitio. Pero existe una tercera opción, quizás la más idónea, para poder ver los datos y la información, utilizar una plataforma para proyectos del IoT.

5.9 Plataformas IoT

A lo largo del documento se han trabajado los conceptos de Machine To Machine (M2M), Internet of Things (IoT) e Internet of Everything (IoE). A continuación se expone la diferencia entre estos tres términos, todos ellos englobados bajo el concepto global de Internet del Futuro:

- El concepto M2M hace referencia a la interconexión de máquinas y dispositivos, que comparten información.
- IoT comprende la interconexión de las máquinas y dispositivos a la red, incorporando así a M2M las posibilidades que ofrece Internet.
- IoE supone el nivel más avanzado de interconectividad y multiplicidad de agentes y actores, ya que al concepto de IoT se añaden los procesos y las personas, dando como resultado una red de redes que incluye a los dispositivos interconectados a través de Internet, a los usuarios, los procesos y los datos.



Repasando o visto el universo de IoE se compone de cuatro tipos de elementos:

- Las **cosas**, elementos conectados que captan información a través de distintos sensores.
- Las **personas**, que interactúan entre sí (conexiones P2P), o con los dispositivos del sistema (P2M) y para quienes las compañías desarrollan productos y servicios.
- **Datos**, generados, analizados, explotados y compartidos por los distintos agentes del sistema.
- **Procesos**, que permiten la gestión de la información generada en todo el ecosistema.

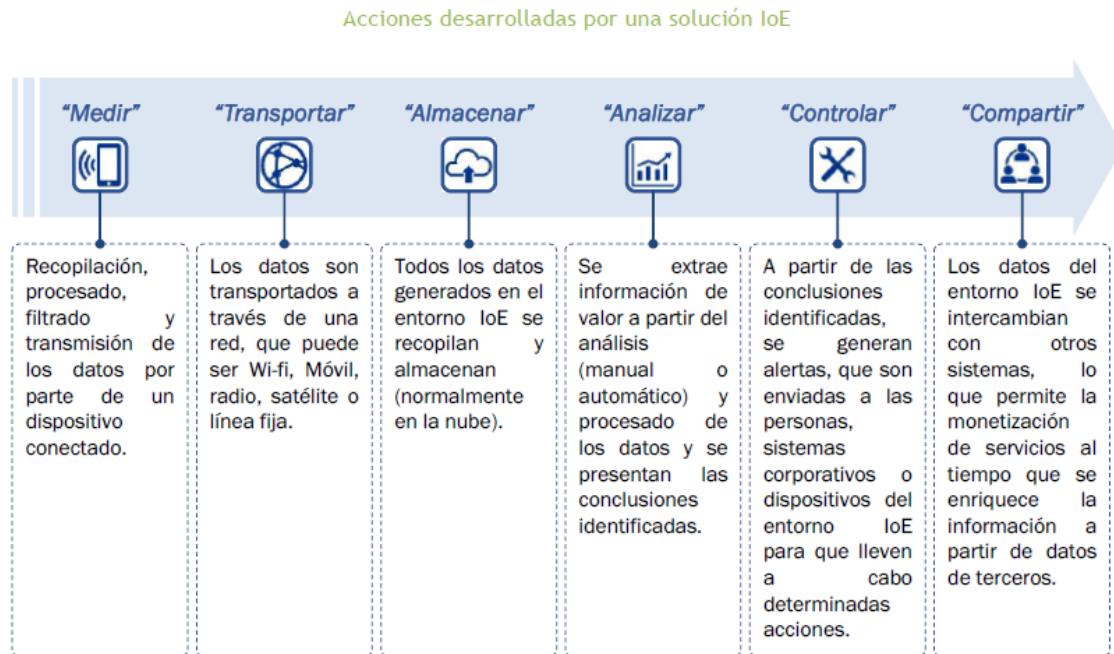


Para que una solución pueda ser realmente considerada en el entorno de IoE debe cumplir con lo que Verizon denomina **la regla de las 3 A's**:

- **“Aware”**. Un dispositivo conectado, que pertenezca a un ecosistema IoE, debe ser consciente de su entorno. Es decir, debe ser capaz de medir lo que sucede a su alrededor, ya sea mediante un sensor de presencia, proximidad, altitud, movimiento, etc. Podría decirse que “para que una solución sea realmente de IoE, debe medir algo”.
- **“Autonomous”**. Los datos recopilados por un dispositivo conectado deben poder ser enviados a otra aplicación en la que sean analizados y procesados de forma automática, ya sea a una hora concreta, o

ante la detección de un evento determinado. Podría decirse que “para que una solución sea realmente de IoE, debe tener conectividad”.

- **“Actionable”.** IoE no consiste únicamente en recopilar datos, sino que además hay que utilizarlos para mejorar los procesos de toma de decisiones, lo que constituye el verdadero valor añadido de IoE. Los datos analizados deben ser integrados en los procesos de negocio, ya sea de forma automática o manual. Podría decirse que “para que una solución pueda considerarse realmente de IoE, los datos almacenados y procesados deben permitir tomar decisiones para realizar acciones, ya sea por la propia solución o por otra aplicación”.



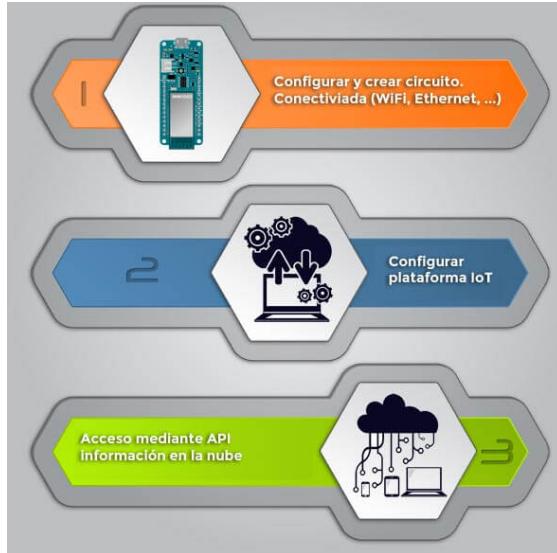
Amazon Web Services (AWS) y Microsoft Azure son considerados líderes en Cloud, siendo AWS aproximadamente el doble de grande que Microsoft Azure quien es su rival más cercano, y 10 veces más grande que sus otros competidores juntos, aunque Azure ha presentado un rápido desarrollo en los últimos años

Además de poder ver la información, algunas de estas plataformas permiten tener un histórico de los datos. Esto entra dentro de lo que se conoce el Big Data (manejo de grandes cantidades de datos e información). Una vez almacenada la información, se pueden aplicar varias técnicas de predicción y de gestión.

Antes de meternos a ver las diferentes plataformas, es interesante hacerse una idea general de lo que sería un sistema basado en plataformas en la nube. En este sistema intervienen tres elementos principales:

- El dispositivo conectado o del IoT
- La plataforma en la nube
- Los dispositivos que consumen la información en la plataforma del IoT

En las plataformas IoT los protocolos de comunicación más utilizados son HTTP, MQTT y CoAP. Además, para comunicarnos, existen diferentes redes como LoRa o SigFox. Son redes WAN para el IoT y una alternativa a los sistemas tradicionales de comunicación.



Tener una API (Application Programming Interface) nos permitirá consultar, modificar y borrar la información desde otros dispositivos.

Al final tenemos que entender que es una capa de comunicación estándar para conectarnos a los datos. Existen diferentes protocolos y estándares. El más utilizado sería a través de servicios web RESTful.

El acceso a dicha API dependerá del software desde donde nos conectemos. Si se trata de una aplicación web, frameworks como jQuery, AngularJS o React, nos facilitan esta tarea enormemente.

Cada día surgen nuevas plataformas para nuestros proyectos IoT y es complicado hacer un análisis de todas ellas.

Se clasifican en:

Las plataformas que están orientadas a startups y empresas pequeñas. En este tipo he incluido todas aquellas que nos permiten su uso de una manera gratuita pero con limitaciones en cuanto al número de mensajes enviados y de dispositivos conectados.

Las plataformas que ofrecen las grandes empresas y corporaciones como Google, Amazon, Microsoft, IBM, etc... Están orientadas sobre todo al sector industrial y a grandes proyectos del IoT, donde se ven involucrados cientos o miles de dispositivos.

El último grupo englobaría las plataformas de código abierto. Son todas aquellas que nos dan acceso al código sin restricciones. Podemos descargarlas e instalarlas en nuestras máquinas de forma local.

- Algunas plataformas gratuitas son:

IoT con el NodeMCU - 50

- Samsung Artik Cloud
- aREST Framework
- Thinger.io
- Arduino Cloud
- ThingSpeak
- Cayenne

5.10 PRACTICAS SOBRE PLATAFORMAS IOT CON EL NODEMCU

Practica 8. Enviando datos al Facebook e-mail con el IFTTT.

Practica 9. Graficando y enviando twiter con el ThingSpeak.

Practica 10. Blink

BIBLIOGRAFÍA Y WEBGRAFÍA.

- Introduction to IoT - CISCO
Conozca cómo Internet de las Cosas (IdC) y la transformación digital de la empresa generan nuevo valor y nuevas oportunidades laborales.
<https://www.netacad.com/es/courses/intro-iot/>
- Generador de código por Adel Kassah.
<http://easycoding.tn/tuniot/demos/code/?lang=es>
- ESPloradores, Ayudas y consejos para procesadores ESP.
<http://www.esploradores.com/>
[admin@esploradores.com.](mailto:admin@esploradores.com)
- Este blog nace con la idea de recopilar información de Arduino y ESPxx/NodeMCU
<http://arduinoamuete.blogspot.com.co/>
- Naylamp Mechatronics, Apoyo en el desarrollo de tus proyectos con los mejores productos y tutoriales
<http://www.naylampmechatronics.com/>
- Tienda de productos
<http://www.prometec.net/indice-tutoriales>
- <https://tinker.yeoman.com.au/>
- <https://programarfacil.com/>
- Si se puede programar, entonces me gusta.
<https://victorventura.es/>
- Arduino en Español
<http://manueldelgadocrespo.blogspot.com.co/>