**Contenido**

[Introducción Microsoft Azure IoT 2](#_Toc29223578)

[Internet of Things (IoT). 2](#_Toc29223579)

[Arquitectura Azure IoT 2](#_Toc29223580)

[Core Subsystem. 2](#_Toc29223581)

[Servicios de IoT 4](#_Toc29223582)

[NodeMcu 6](#_Toc29223583)

[Pinout NodeMcu 6](#_Toc29223584)

[Configuración entorno de desarrollo. 8](#_Toc29223585)

[Crear un nuevo Proyecto. 8](#_Toc29223586)

[Subiendo el código al ESP8266 8](#_Toc29223587)

[Ejercicio 01 Probando el entorno de programación. 8](#_Toc29223588)

[Ejercicio 02 HC-SR04 9](#_Toc29223589)

[Ejercicio 03 Conectando Azure IoT Hub 10](#_Toc29223590)

# Introducción Microsoft Azure IoT

## Internet of Things (IoT).

Internet de las cosas (IoT) es la conexión de millones de dispositivos inteligentes y sensores conectados a Internet. Estos dispositivos y sensores conectados recopilan y comparten datos para que muchas organizaciones las usen y evalúen. Estas organizaciones incluyen empresas, ciudades, gobiernos, hospitales y personas. IoT ha sido posible, en parte, debido a la llegada de procesadores asequibles y a las redes inalámbricas. Los objetos previamente inanimados, como picaportes o lámparas, ahora pueden contar con un sensor inteligente que puede recopilar y transferir datos a una red.

## Arquitectura Azure IoT

Las aplicaciones de IoT se pueden describir como **Cosas** (o dispositivos), enviando datos o eventos que se usan para generar **Insights**, que se usan para generar **Acciones** para ayudar a mejorar un negocio o proceso.

Un ejemplo es un motor (**una cosa**), que envía datos de presión y temperatura que se utilizan para evaluar si el motor está funcionando como se esperaba (**una idea**), que se utiliza para priorizar proactivamente el programa de mantenimiento del motor (**una acción**).



Figura Arquitectura Azure Iot

## Core Subsystem.

En el núcleo, una aplicación IoT consta de los siguientes subsistemas:

1. Dispositivos y/o Edge Gateway: Los dispositivos se pueden conectar directa o indirectamente a través de un gateway (IoT Edge).tanto los dispositivos como los gateways pueden implementar capacidades de inteligencia de edge. Esto permite la agregación y reducción de datos de telemetría raw antes de enviar al backend y brinda la capacidad local de toma de decisiones en el edge.
2. Cloud Gateway (IoT hub): Una puerta de enlace en la nube permite la comunicación remota desde y hacia dispositivos o dispositivos periféricos, que potencialmente residen en varios sitios diferentes. Se podrá acceder a una puerta de enlace en la nube a través de Internet pública, o una VPN, o conexiones de red privadas en los centros de datos de Azure.
3. Procesadores de Flujo: A medida que los datos se ingieren al backend de IoT, es importante comprender cómo puede variar el flujo del procesamiento de datos. Dependiendo de los escenarios y las aplicaciones, los registros de datos pueden fluir a través de diferentes etapas, combinados en diferente orden y, a menudo, procesados por tareas paralelas concurrentes. Estas etapas se pueden clasificar en cuatro categorías: almacenamiento, enrutamiento, análisis y acción / visualización.
   1. El almacenamiento incluye cachés en memoria, colas temporales y archivos permanentes.
   2. El enrutamiento permite el envío de registros de datos a uno o más puntos finales de almacenamiento, procesos de análisis y acciones.
   3. El análisis se utiliza para ejecutar registros de datos de entrada a través de un conjunto de condiciones y puede producir diferentes registros de datos de salida.
   4. Los registros de datos de entrada originales y los registros de salida de análisis generalmente se almacenan y están disponibles para mostrar, y pueden desencadenar acciones como correos electrónicos, mensajes instantáneos, tickets de incidentes, tareas de CRM, comandos de dispositivos, etc.
4. Interfaz de usuario: La interfaz de usuario (UI) de la solución generalmente incluye un sitio web y un informe, pero también puede incluir servicios web y una aplicación móvil o de escritorio.

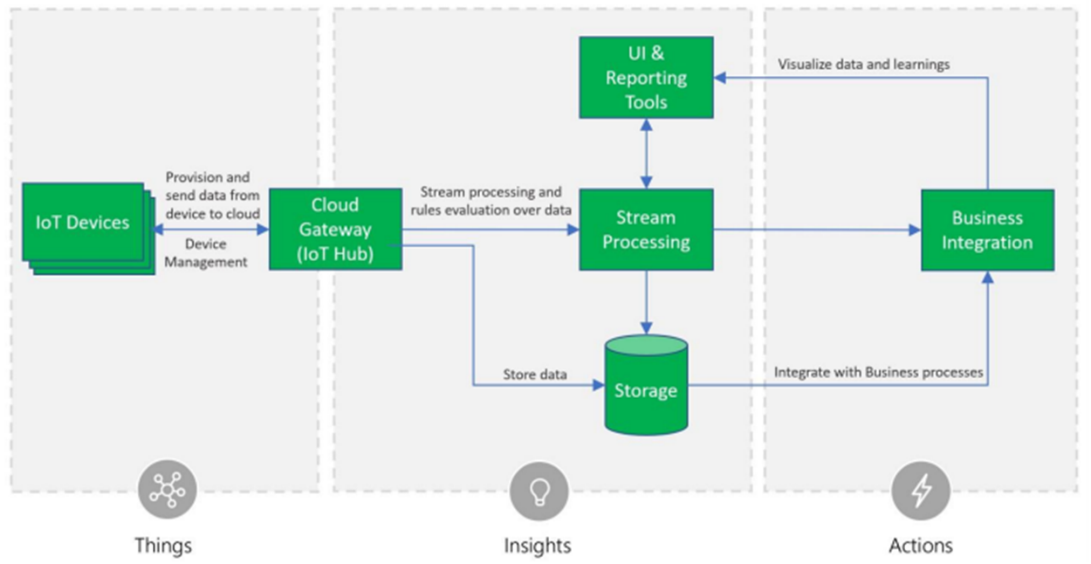


Figura Core Subsystem

## Servicios de IoT

Hay varios servicios relacionados con IoT en Azure y puede resultar confuso averiguar cuál de ellos desea usar. Algunos, como IoT Central y los aceleradores de soluciones IoT, proporcionan plantillas para ayudarle a crear su propia solución y empezar a trabajar rápidamente. También puede desarrollar completamente sus propias soluciones mediante otros servicios disponibles, todo depende de cuánta ayuda y cuánto control desee. Esta es una lista de los servicios disponibles, así como sus usos posibles.

IoT Central: Se trata de una plataforma de aplicaciones de IoT que simplifica la creación de soluciones de IoT y ayuda a reducir la carga y el costo de las operaciones de administración de IoT y el desarrollo. Para empezar, seleccione una plantilla para el tipo de dispositivo y cree y pruebe una aplicación básica de IoT Central que utilizarán los operadores del dispositivo. La aplicación de IoT Central también le permitirá supervisar los dispositivos y aprovisionar nuevos dispositivos. Este servicio es para soluciones sencillas que no requieren personalización profunda del servicio.

Aceleradores de soluciones IoT: se trata de una colección de soluciones PaaS que puede usar para acelerar el desarrollo de una solución IoT. Comience con una solución IoT proporcionada y personalice completamente esa solución según sus requisitos. Necesita conocimientos de Java o .NET para personalizar el back-end y conocimientos de JavaScript para personalizar la visualización.

IoT Hub: este servicio le permite conectarse desde los dispositivos a un centro de IoT y supervisar y controlar miles de millones de dispositivos IoT. Es especialmente útil si necesita comunicación bidireccional entre los dispositivos IoT y el back-end. Se trata del servicio subyacente de IoT Central y los aceleradores de soluciones IoT.

IoT Hub Device Provisioning Service: se trata de un servicio auxiliar de IoT Hub que puede usar para aprovisionar dispositivos en el centro de IoT de forma segura. Con este servicio, se pueden aprovisionar fácilmente millones de dispositivos con rapidez, en lugar de aprovisionarlos de uno en uno.

IoT Edge: este servicio se basa en IoT Hub. Se puede usar para analizar los datos en los dispositivos IoT en lugar de en la nube. Al mover partes de la carga de trabajo al perímetro, es necesario enviar menos mensajes a la nube.

Azure Digital Twins: este servicio le permite crear modelos integrales del entorno físico. Puede modelar las relaciones y las interacciones entre personas, espacios y dispositivos. Por ejemplo, puede predecir las necesidades de mantenimiento de una fábrica, analizar los requisitos de energía en tiempo real de una red eléctrica u optimizar el uso del espacio disponible en una oficina.

Time Series Insights: este servicio le permite almacenar, visualizar y consultar grandes cantidades de datos de series temporales generados por los dispositivos IoT. Puede usar este servicio con IoT Hub.

Azure Maps: este servicio proporciona información geográfica para aplicaciones web y móviles. Hay un conjunto completo de API REST, así como un control JavaScript basado en web que se puede usar para crear aplicaciones flexibles que trabajan en aplicaciones de escritorio o móviles para dispositivos Apple y Windows.

# NodeMcu

El ESP8266 es un chip Wi-Fi de bajo coste que funciona mediante el protocolo TCP/IP. Incluye un microcontrolador (Tensilica Xtensa LX106) para manejar dicho protocolo y el software necesario para la conexión 802.11. Además la mayoría de modelos dispone de entradas/salidas digitales de propósito general (GPIO), así como una entrada analógica (ADC de 10bit).

Su punto fuerte es el disponer de una conexión Wi-Fi en un microcontrolador cuyo precio ronda los 3€. Además se puede programar directamente con el entorno de Arduino con lo que es el chip perfecto para desarrollar nuestras aplicaciones de IoT.

Existen varios modelos del ESP8266 que se diferencian en la cantidad de GPIOs disponibles y la forma/tecnología de la antena Wi-Fi. En nuestro caso, nos centraremos en el Modelo ESP-12-E por ser el que tiene un uso mas extendido, así como una mayor comunidad.

NodeMCU es un proyecto Open-Source para el desarrollo de un modelo sencillo de integrar la IoT (Internet of Things), para ello se desarrollan modelos hardware y software que facilite el desarrollo de programas y aplicaciones basados en WiFi. Es uno de los proyectos más interesantes y completos que han surgido alrededor del módulo ESP8266.

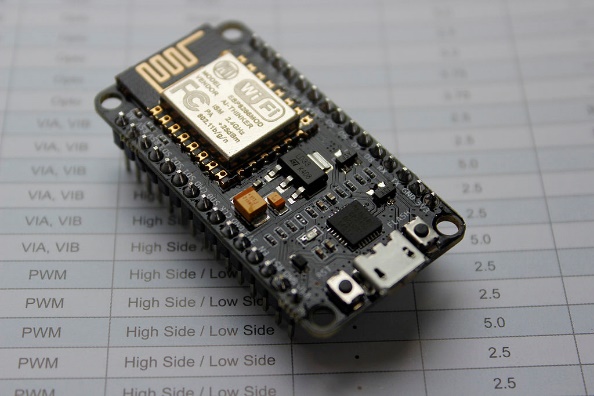


Figura NodeMCU

## Pinout NodeMcu

* Digital I/O Pins (DIO): 16
* Analog Input Pins (ADC): 1
* UARTs: 1
* SPIs: 1
* I2Cs: 1
* Wi-Fi: IEEE 802.11 b/g/n

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Pin | GPIO | Input | Output | Comentarios |
| D0 | GPIO16 | No interrupciones | No PWM  No I2C | HIGH durante boot Resistencia Pull-Down Conectar a RST para Wake-Up |
| D1 | GPIO5 | OK | OK | SCL (I2C) (frecuentemente) |
| D2 | GPIO4 | OK | OK | SDA (I2c) (frecuentemente) |
| D3 | GPIO0 | Pulled Up | OK | Boot falla si pulled LOW Conectado a botón FLASH |
| D4 | GPIO2 | Pulled Up | OK | HIGH durante boot Boot falla si pulled LOW Built-in LED TX1 |
| D5 | GPIO14 | OK | OK | SLCK (SPI) |
| D6 | GPIO12 | OK | OK | MISO (SPI) |
| D7 | GPIO13 | OK | OK | MOSI (SPI) |
| D8 | GPIO15 | Pulled GND | OK | CS (SPI) LOW durante boot Boot falla si pulled HIGH No tiene Pull-Up |
| RX | GPIO3 | OK | RX | HIGH durante boot No usable si se usa UART |
| TX | GPIO1 | TX | OK | HIGH durante boot Boot falla si pulled LOW  Debug output en boot No usable si se usa UART |
| A0 | **ADC0** | Analog Input | NO |  |

Tabla PinOut NodeMcu

## Configuración entorno de desarrollo.

* Descargar el IDE de Arduino e instalarlo en nuestra Computadora.
* Configurar el gestor de URL´s de tarjetas adicionales,
  + <https://arduino.esp8266.com/stable/package_esp8266com_index.json>
* Ir al gestor de tarjetas e instalar la tarjeta esp8266.
* Ingresar a Visual Studio Code e instalar las siguientes extensiones:
  + Arduino.
  + c/c++.
* Abrir la paleta de comandos e ingresar a la configuraciones de usuario.
* Buscar “Arduino”, colocar la ruta de instalación de Arduino IDE.

## Crear un nuevo Proyecto.

* En VS Code creamos un nuevo directorio llamada “ejercicio-01”.
* Creamos un nuevo archivo y lo guardamos con el nombre “ejercicio01.ino”.
* En la paleta de comandos tecleamos “Arduino:initialize”.
  + Seleccionamos la tarjeta “NodeMCU 1.0”.
* Configuramos los parámetros de la tarjeta “Board Config”:
  + Flash size: 4MB (FS 3MB OTA ~512KB).

## Subiendo el código al ESP8266

* Seleccionamos el puerto serial “Select Serial Port”.
* Subimos el código al Esp8266 “Upload”.

## Ejercicio 01 Probando el entorno de programación.

#include "Arduino.h"

#include "Esp.h"

void setup(){

    pinMode(2,OUTPUT);

}

void loop(){

    digitalWrite(2,HIGH);

    delay(500);

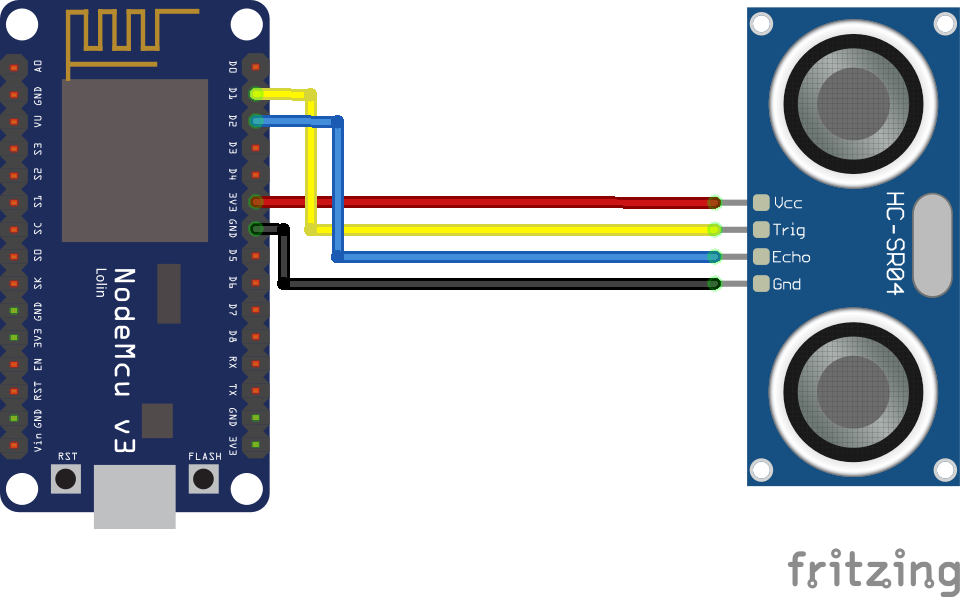
    digitalWrite(2, LOW);

    delay(100);

}

## Ejercicio 02 HC-SR04

Creamos el siguiente circuito.



#include "Arduino.h"

#include "Esp.h"

#define TRIGGER 5

#define ECHO 4

void setup()

{

    Serial.begin(9600);

    pinMode(TRIGGER, OUTPUT);

    pinMode(ECHO, INPUT);

}

void loop()

{

    long duration, distance;

    digitalWrite(TRIGGER, LOW);

    delayMicroseconds(2);

    digitalWrite(TRIGGER, HIGH);

    delayMicroseconds(10);

    digitalWrite(TRIGGER, LOW);

    duration = pulseIn(ECHO, HIGH);

    distance = (duration / 2) / 29.1;

    Serial.print("Distancia: ");

    Serial.print(distance);

    Serial.println(" cm");

    delay(1000);

}

## Ejercicio 03 Conectando Azure IoT Hub

* Creamos un Grupo de Recursos
* Creamos un IoT HUB
* Creamos un Dispositivo
* Instalar las siguientes Librerías.
  + AzureIoTHub
  + AzureIoTUtility
  + AzureIoTProtocol\_MQTT
  + ArduinoJson