

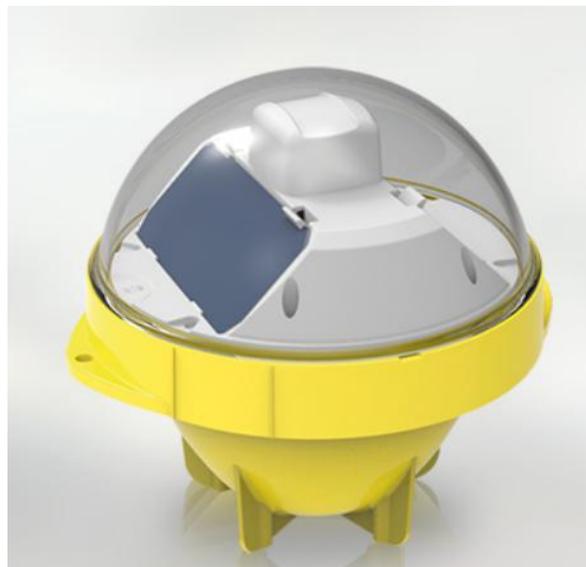
Tecnologías IIoT, hands-on y MVPs

Práctica: MVP para depuradora

Presentación

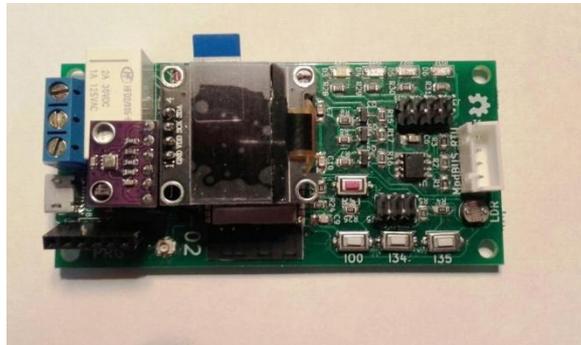
El objetivo principal de esta práctica es desarrollar un mínimo producto viable (MVP) de un sistema de IoT flotante, alimentado por batería de larga autonomía, que permita obtener lecturas del nivel del líquido en tanques, presión, temperatura, humedad relativa, nivel de luz, y en una evolución futura podría incorporar lecturas de sensores en contacto con el líquido como PH y otras informaciones provenientes del líquido de los tanques. El trabajo debe realizarse preferentemente en equipo, pudiendo también ser realizado de forma individual.

En el sector de la pesca hay disponibles diversos productos consistentes en boyas con conexión vía satélite para el seguimiento de bancos de peces y aplicaciones similares. Supondremos que la parte mecánica será resuelta por un equipo ajeno, así como un mecanismo de igualamiento de la presurización del interior con el exterior mediante una válvula de apertura puntual antes de realizar las lecturas de presión, temperatura u otras.



En la figura puede observarse una boya del fabricante Satlink (<https://satlink.es/boyas-inteligentes/satlink-boya-palangre>).

En nuestro caso la tecnología de comunicaciones inalámbricas a utilizar será a utilizar será Wi-Fi estándar, y utilizaremos la placa de prototipaje IoT-02, de hardware abierto, basada en el procesador ESP32:



Objetivos y competencias

Los **objetivos** de esta actividad son los siguientes:

- 1) Conocer la conexión de sensores a microcontroladores de escala mediana con capacidades de conexión inalámbrica.
- 2) Conectar sistemas embebidos a PCs industriales o al *cloud*, almacenando datos de las lecturas en bases de datos orientadas a series temporales como InfluxDB.
- 3) Utilizar sistemas de software para IoT como Node-RED y Grafana.
- 4) Implementar servicios basados en HTTP y MQTT para monitorizar y visualizar los datos de las lecturas mediante dispositivos con navegador.
- 5) Conocer y experimentar con el concepto gemelo digital para prototipaje

Se recomienda haber realizado algún ejercicio previo con la placa IoT-02.

Ejercicio 1

Se dispone de un firmware (ver punto de Recursos disponibles¹) para la placa IoT-02 que permite solicitar vía MQTTs lecturas de de temperatura (T), humedad relativa (RH), presión (P), altura (A), y nivel de luz (L) provenientes del sensor BME-280 incorporado y la LDR integrada en la placa. El archivo de *cabecera IoT-02_mqttTopics.h* define los temas MQTT disponibles en la placa, los cuales permiten solicitar lecturas individuales de cada lectura, así como una lectura conjunta de los sensores en el siguiente formato JSON:

```
{ "T" : 20.0 , "RH" : 45.5 , "P" : 1001.2 , "A" : -8.3 , "L" : 3040 }
```

¹Recursos disponibles:

[http://www.binefa.com/index.php?title=Algunas_pistas_para_la_pr%C3%A1ctica_MVP_para_depuradora_\(curso_2019-2020\)](http://www.binefa.com/index.php?title=Algunas_pistas_para_la_pr%C3%A1ctica_MVP_para_depuradora_(curso_2019-2020))

Programar la placa de electrónica embebida IoT-02, indicando en caso los datos de conexión de vuestro punto de acceso Wi-Fi, y verificad los datos relativos al broker MQTT que vais a utilizar (podéis optar entre broker seguro o broker no seguro):

Broker no seguro Datos del <i>broker</i> : Servidor: vps656540.ovh.net Puerto MQTT: 1883 Puerto Websockets (no seguro): 8080 Usuario: user Contraseña: pass
--

Implementad un programa en Node-RED que realice una petición MQTT a la placa IoT-02 de las lecturas en el formato JSON indicado, y las represente en un Dashboard de Node-RED que tenéis que crear.

Cada uno de vosotros dispone de una instancia de Node-RED que utiliza el puerto cuyo número se obtiene mediante la siguiente fórmula:

$$\text{PuertoUsuario_masterxy} = 1000 * \text{xy} + 880$$

Por ejemplo, el usuario master11 utilizará el puerto 11880.

Para gestionar las instancias de los Node-RED se dispone de la herramienta pm2, que está instalada en el VPS del máster, la cual permite ver la ejecución de las instancias:

```
sudo pm2 status
```

```
$ sudo pm2 status
```

App name	id	version	mode	pid	status	restart	uptime	cpu	mem	user	watching
node-red00	2	N/A	fork	27988	online	0	23h	0%	2.9 MB	root	disabled
node-red10	1	N/A	fork	8391	online	0	20m	0%	2.8 MB	root	disabled
node-red11	3	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red12	4	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red13	5	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red14	6	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red15	7	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red16	8	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red17	9	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red18	10	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red19	11	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red20	12	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red21	13	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red22	14	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red23	15	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red24	16	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red25	17	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red26	18	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red27	19	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red28	20	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled
node-red29	21	N/A	fork	0	stopped	0	0	0%	0 B	root	disabled

Así como arrancarlas, pararlas o reiniciarlas:

```
sudo pm2 start node-redxy  
sudo pm2 stop node-redxy  
sudo pm2 restart node-redxy
```

(*pm2 es una utilidad que se puede instalar en vuestro ordenador personal o en un VPS del cloud para vuestros proyectos: npm install pm2*).

Ejercicio 2

Implementar un proceso que se suscriba al *topic* de publicación de la IoT-02 de forma que las lecturas se almacenen en una base de datos InfluxDB. Para ello puede utilizarse como lenguaje de programación Node-RED ².

En cuanto al InfluxDB, deberéis anteponer el prefijo “m” + xy + “_” al nombre del *measurement* (equivalente a una tabla) de la base de datos.

Representar en el sistema Grafana las lecturas de presión, temperatura, humedad relativa y nivel de luz almacenadas en InfluxDB.

Utilizar los usuarios de Grafana disponibles en el VPS del máster, y nombrar los paneles de control (*dashboards*) según la regla del prefijo “m” + xy + “_”.

² Node-RED es la opción por defecto, pero quien tenga interés puede utilizar alternativamente Python, Node.js o una combinación de éstos. Asimismo, opcionalmente, quien tenga interés, puede probar algún otro tipo de base de datos como Microsoft Azure SQL o MongoDB.

Ejercicio 3

El sensor de presión se puede utilizar para medir la altura en la que está la placa IoT-02, la cual se puede medir calculando la diferencia entre las lecturas entre dos puntos. En un caso real, dispondríamos de dos placas IoT-02, una colocada a la altura del suelo del depósito y la otra colocada en la boya flotante, de manera que la altura se calcularía por la diferencia de las dos lecturas. Como sólo disponemos de una placa, para calcular la altura, deberemos realizar una primera lectura con la placa en el suelo, y utilizarla para el cálculo de la diferencia cuando se esté colocando la placa a diferentes alturas y ver cómo va

evolucionando la lectura de ésta en tiempo real. Alimentar la placa con un *powerbank* de teléfono móvil puede ayudar a realizar el experimento.

La presión de la placa a la altura del suelo va variando con el tiempo, pero supondremos que la variación de ésta mientras dure el experimento, no es significativa.

Se pide que incorporéis en el dashboard del ejercicio 1 la visualización de la altura de la placa IoT-02 respecto al suelo en tiempo real. Valorad la precisión del sistema y proponed un sistema de redondeo para aplicar en depósitos de depuradoras.

Nota sobre la seguridad de Node-RED:

Cada uno de los usuarios masterxy dispone de una carpeta propia `node_modules/node-red`, con el correspondiente fichero de configuración `settings.js`. Como tarea opcional, se pide activéis un acceso con usuario/contraseña para vuestra instancia de Node-RED (<https://nodered.org/docs/user-guide/runtime/securing-node-red>).

Ejercicio 4 (Opcional)

En la dirección <https://wokwi.com/projects/328227183923298899> hay un gemelo digital de un sistema bautizado como VirKoLDR, del cual se ha realizado un prototipo con un [módulo de prototipaje popular basado en ESP32](#), que se acompaña del código fuente que permite la consulta síncrona de la lectura de una LDR. Por otra parte, el código Low-Code disponible en http://vps656540.ovh.net/snap/snap.html#open:test_dt_esp32_ldr.xml permite realizar las consultas con un bloque del tipo “MQTT Request”.

Se pide que:

- 1) Arranquéis el simulador del gemelo digital
- 2) Arranquéis la página low-code (Snap!), ejecutéis los bloques de conexión y luego el bloque de consulta, y comprobéis que se realiza la lectura de forma síncrona.
- 3) Encapsuléis el bloque en un simple bloque tipo *reporter* denominado “getLux”
- 4) Adaptéis la parte de código que consideréis para implementar la misma funcionalidad, y opcionalmente añadir otra(s) similar(es), entre luz, presión, temperatura, humedad ..., de la placa IoT-02 (Considerar la combinación de la dirección MAC y del timer para la formación de la idCall).
- 5) Utilicéis el mismo low-code de pruebas para verificar el sistema real IoT-02.

Nota: Dependiendo de la hora del día, la conexión del simulador puede fallar, puesto que está en modo demostración sin coste

The screenshot displays the WOKWI simulation environment. On the left, the code editor shows a C++ sketch for an ESP32. The code includes headers for WiFi and PubSubClient, defines constants for SSID, password, MQTT server, port, and MAC address, and implements a setup function that initializes serial communication and connects to an MQTT broker. The block editor in the center shows a visual representation of the code logic, including blocks for opening a website, setting the MQTT broker, connecting to the broker, and requesting data from a specific topic. On the right, the simulation window shows a 3D model of the ESP32 board connected to an MQTT broker module. The simulation output at the bottom right displays the following text:

```
10.10.0.2
24:0A:C4:00:01:10
24_0A_C4_00_01_10
Attempting MQTT connection...clientId-676 connected

Topic: topicName/ldr/lux/idCall-1032.3
Callback topic: idCall-1032.3
```