

Trabajo de IARS

Dashboard en IBM Cloud



IBM Cloud

Estudios: Máster en Sistemas Inteligentes en Energía y Transporte

Asignatura: Infraestructura Avanzada de Redes de Sensores

Realizado por : Luis Manuel García-Baquero Corredera <luikygaba@gmail.com>

Curso: 2018/2019

Índice

1. Introducción	3
2. Node-RED en Raspberry Pi	3
2.1 Sensor de Temperatura	3
2.2 Aire Acondicionado	5
3. Integración en IBM Cloud	6
3.1 Internet Of Things Platform	6
3.2 Db2	9
3.3 Cloudant	12
3.4 Node-RED	12
3.4.1 Flujo de sensorización	14
3.4.2 Flujo de actuadores	16
4. Dashboard	18
4.1 Sensorización	19
4.2 AC	20
4.3 Dispositivos	20
5. Conclusiones	21

1. Introducción

En esta memoria se va a presentar la realización del trabajo final de la asignatura IARS, en el cual hay que hacer la integración de los datos de temperatura generados en las prácticas con la plataforma de IBM Cloud. El objetivo es poder visualizar un historial de los datos recibidos, un indicador del último dato recibido y un nuevo mecanismo para encender y apagar la recepción de datos como se hizo con el botón de las prácticas.

Para afrontarlo, se va a implementar una instancia de Node-RED en Bluemix en la que se van a desarrollar los flujos necesarios para la integración. La visualización y el accionamiento pedidos van a integrarse en un dashboard de Node-RED.

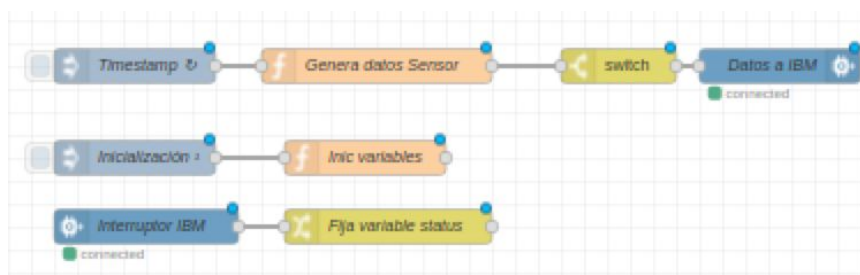
Como añadido a los requisitos iniciales del problema, se va a apoyar el sistema en una base de datos Db2 de IBM Cloud con diversas tablas y se van a implementar algunas características adicionales como es el envío de avisos con correo electrónico. También se va a incorporar un dispositivo actuador, un aire acondicionado, para ver cómo interactúa con el sensor de temperatura y que se acerque más a un escenario real de automatización con IoT.

2. Node-RED en Raspberry Pi

En la Raspberry se inicializa una instancia de Node-RED con dos flujos: uno para el sensor de temperatura y otro para el aire acondicionado. Entre ellos comparten dos variables globales: la temperatura del habitáculo y la velocidad del compresor del aire acondicionado.

2.1 Sensor de Temperatura

La implementación del sensor de temperatura es muy parecida a la de las prácticas, sustituyendo los nodos de entrada y salida por los propios de IBM y el algoritmo de simulación de temperatura:



En primer lugar encontramos el nodo de Timestamp, que inyecta el timestamp del momento concreto cada 5 segundos, enviándolo al generador de datos del sensor que funciona así:

```
1 // Obtiene la temperatura actual
2 temp = global.get('temp');
3 // Actualiza la temperatura
4 global.set('temp',temp+0.1-global.get('velo_aire'));
5
6 // Pone el timestamp en el formato y zona correctos
7 var d = Math.round(new Date(msg.payload).getTime()/1000)+2*3600;
8 d = new Date(d*1000).toISOString();
9
10 // ID del sensor de temperatura
11 var sensor_id = 31;
12
13 // Guarda los datos
14 msg.payload = {Temperature: temp,
15               Timestamp: d,
16               Units: '°C',
17               Sensor_Id: sensor_id,
18               reading_Id: msg.payload+sensor_id,
19               magnitud: 'Temperatura'
20 };
21 return msg;
```

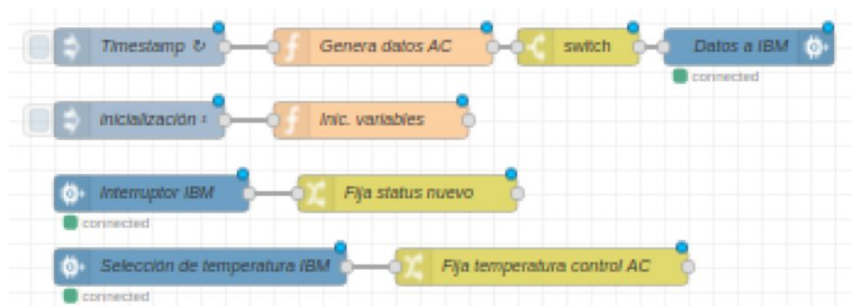
Como se ha comentado anteriormente, se carga la temperatura del habitáculo que está almacenada en una variable global. Después se actualiza la temperatura, la cual aumenta 0.1°C cada 5 segundos (se supone que en el exterior hace más calor), pero puede disminuir si está encendido el aire acondicionado. A continuación se pone el timestamp en un formato correcto y se fija el identificador del sensor, para terminar guardando todos los datos en la carga del mensaje.

Los datos generados en este nodo son enviados a la plataforma de IBM mediante el último nodo de la derecha si la lectura de datos se encuentra activada, teniendo la variable de flujo 'Status' igual a 1. Esta variable 'Status' es controlada desde un accionamiento en la plataforma de IBM, cuyo comando se recibe en el primer nodo de la tercera fila, 'Interruptor de IBM'. Éste modifica el valor de la variable 'Status' cuando llega un comando desde la plataforma de IBM. Finalmente, la segunda fila se encarga de inicializar la temperatura a 35°C (día caluroso de verano) y con estado de encendido.

Los nodos de entrada y salida de IBM se configuran asignándoles una entidad de dispositivo creada en la plataforma de IoT de IBM para representar el sensor de temperatura, introduciendo la organización a la que pertenece, el ID de dispositivo y la clave de acceso generada. Más adelante se explicará esta configuración en el punto sobre la plataforma IoT.

2.2 Aire Acondicionado

La estructura de este flujo es similar a la anterior, ya que también se basa en la generación de datos, el envío de estos a la plataforma de IBM y la recepción de otros comandos provenientes de ésta. En este caso los comandos a recibir posibles son para encender y apagar el aire acondicionado, y también otro que regula la temperatura de control del dispositivo, que es la que va a intentar mantener. Por lo tanto, en este apartado van a explicarse los cambios que haya respecto al flujo de temperatura. Con todo esto, el flujo queda así:



La generación de datos en este flujo es distinta, además de producirse cada 3 segundos en vez de 5 (necesario para lograr un buen control):

```

1 // Obtiene la temperatura del habitáculo,
2 // el estado de encendido del AC y la temperatura
3 // de control
4 var temp = global.get('temp');
5 var st = flow.get('st');
6 var control = flow.get('control_aire');
7
8 // Fija el ID de actuador
9 var actuador_id = 42;
10
11 // Cálculo de la velocidad del compresor necesaria
12 // para alcanzar la temperatura de control y
13 // cálculo de consumo energético
14 * if (st=='0'){
15     global.set('velo_aire',0);
16     consumo = 0;
17 * } else if(st=='1' & (temp-control)>10){
18     global.set('velo_aire',1.5);
19     consumo = 1.5 * 3/3600;
20 * } else if(st=='1' & (temp-control)>7.5){
21     global.set('velo_aire',1);
22     consumo = 1.2 * 3/3600;
23 * } else if(st=='1' & (temp-control)>3){

```

Como se puede ver, la velocidad de compresor del aire acondicionado depende de la diferencia entre la temperatura del habitáculo y la temperatura de control fijada en el dispositivo, comenzando con mayor velocidad para paliar la alta diferencia y disminuyendo conforme se va acercando al objetivo. También se aprecia cómo se almacena la velocidad del compresor

como variable global para que pueda hacer su efecto en la actualización de la temperatura del habitáculo en el flujo del sensor de temperatura. Al final, la estructura de datos que se saca es:

```
msg.payload = {Consum: consumo,  
Timestamp: d,  
Units: 'kWh',  
Actuator_Id: actuator_id,  
Temp_control: control,  
Action_Id: msg.payload+actuator_id,  
magnitud: 'Energia'  
};
```

En este caso, la inicialización hace que el aire acondicionado comience estando apagado, con la variable de flujo 'Status' igual a 0, y la temperatura de control igual a 24°C, una medida estándar. Finalmente, la última diferencia es que ahora llega un comando nuevo de IBM, a través del nodo de la última fila. Ésta es una entrada numérica que fija la temperatura de control del aire acondicionado, la cual se almacena en la variable de flujo 'control_aire'.

3. Integración en IBM Cloud

En este proyecto va a trabajarse en el entorno de IBM Cloud, el servicio que ofrece IBM en la nube para desarrollar y alojar todo tipo de soluciones. Ésta ofrece servicios de IaaS (Infrastructure as a Service), PaaS (Platform as a Service) y SaaS (Software as a Service), pudiendo hacer uso de ellos para tareas de computación, almacenamiento, gestión de dispositivos, alojamiento de servidores...

Para las tareas concretas que se van a realizar, se van a usar herramientas de Cloud Foundry, un servicio PaaS de código abierto perteneciente a IBM. Ésta se ejecuta en un servicio de Kubernetes, reduciendo la complejidad, ya que proporciona a los equipos de desarrollo un conjunto completo de herramientas familiares con una gestión unificada, ofreciendo la posibilidad de desplegarse a nivel público, privado o privado empresarial.

Se van a utilizar tres servicios de los que ofrece Cloud Foundry: Internet of Things Platform, Node-RED, Cloudant y Db2.

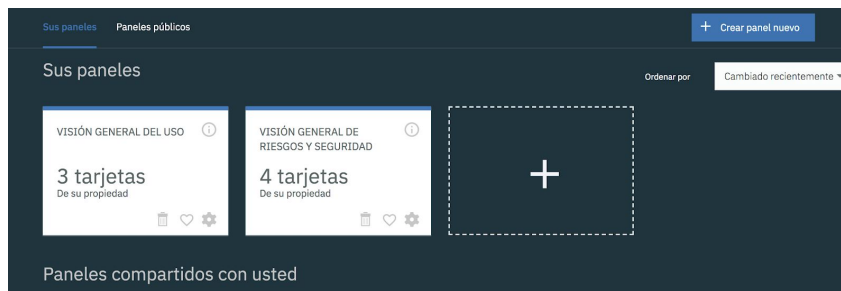
3.1 Internet Of Things Platform



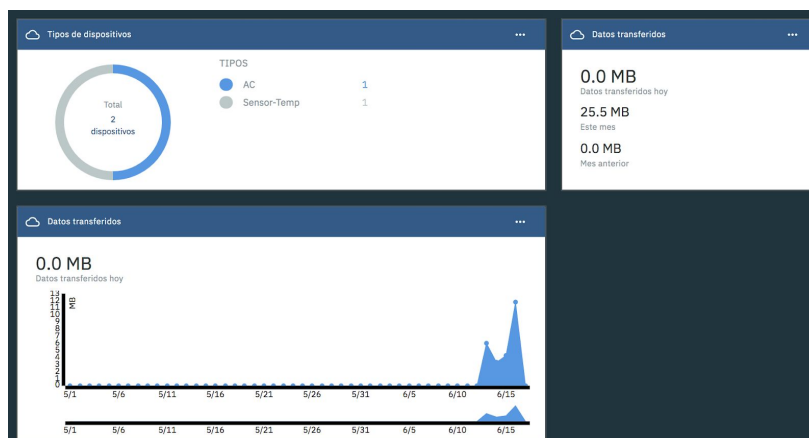
Internet Of Things
IBM

Como su propio nombre indica, ésta es la plataforma de IoT de IBM. Está orientada a integrar infraestructuras de IoT con todas las demás soluciones de IBM Cloud, como las bases de datos, Node-RED, servicios de analítica predictiva... Tiene tres pestañas principales que son:

Paneles



En esta pestaña podemos crear paneles de visualización de informaciones relacionadas con el uso de la plataforma de IoT, o incluso los propios datos enviados por dispositivos. Por ejemplo, en el primer panel, 'Visión general del uso', se puede ver información sobre la transferencia de datos o el tipo de dispositivos que hay en nuestra plataforma:



En un principio se pensó usar uno de estos paneles para cumplir el objetivo de visualización de la práctica, pero la imposibilidad de colocar botones o campos para introducir datos en el propio dashboard hizo que se prefiriese el uso de una dashboard de Node-RED alojada en la nube.

Dispositivos

Examinar

Acción

Tipos de dispositivo

Añadir dispositivo

Examinar dispositivos

Todos los dispositivos

Diagnosticar

Esta tabla muestra un resumen de todos los dispositivos que se han añadido. Se puede filtrar, organizar y buscar en ella utilizando distintos criterios. Para empezar, puede añadir dispositivos utilizando la API o el botón Añadir dispositivo.

Buscar por ID de dispos...

Simulador de dispositivo

	ID de dispositivo	Estado	Tipo de dispositivo	ID de clase
>	31	Conectado	Sensor-Temp	Dispositivo
>	42	Conectado	AC	Dispositivo

Items per page 10 | 1-2 of 2 items

1 of 1 pages

En esta pestaña se gestiona el registro de dispositivos de nuestra infraestructura, pudiendo introducir nuevos elementos, ver y editar los ya registrados, diagnosticar posibles errores o incluso simular algún dispositivo para hacer pruebas.

Para este proyecto se han tenido que registrar dos dispositivos, uno para el sensor de temperatura y otro para el aire acondicionado. Ambos comparten organización, con nombre ‘op8k2b’, y uno tiene de ID 31 y el otro 42. También hay que introducir qué tipo de dispositivo es cada uno, siendo aquí ‘Sensor-Temp’ el primero y ‘AC’ el segundo. Al crear cada uno de estos dispositivos se crea una clave que se introduce junto a los otros datos en los nodos de entrada y salida de IBM, como se mencionó anteriormente. Un ejemplo de los datos resultantes de la creación de un dispositivo sería así:

ID de Organización

op8k2b

Tipo de dispositivo

Sensor_temperatura

ID de dispositivo

31

Método de autenticación

use-token-auth

Señal de autenticación

2itEeM+!XaZj*P&nS+

Aplicaciones

Examinar

Aplicaciones de IBM Cloud

Generar clave de API

Examinar claves de API

Escriba la descripción de la aplicación a bus

Esta tabla muestra un resumen de las claves de API que se han añadido para la organización. Es posible aplicarle filtros, organizarla a conveniencia o realizar búsquedas en ella utilizando distintos criterios. Para empezar, puede añadir claves de API pulsando Generar clave de API o utilizando la API. Para obtener más información sobre cómo añadir claves de API, consulte [Conexión de clave de API](#).

Clave	Descripción	Rol	Caducidad	
1 resultado				
a-op8k2b-ule9r78dxe	RaspberryPi	Aplicación estándar	-	

Esta pestaña permite obtener claves de API que permitan la creación de aplicaciones desde los servicios de la IBM Cloud con los propios dispositivos que se conectan a la plataforma.

En este trabajo hacía falta una aplicación para la conexión del Node-RED de Bluemix con el de la Raspberry, por lo que, como se ve en la imagen de la pestaña, se ha creado una aplicación llamada 'RaspberryPi' de carácter 'estándar'. En un principio se le asignó el carácter 'dispositivo' pero resultó que sólo permitía la comunicación del dispositivo a la Cloud, mientras que el 'estándar' permite que la conexión sea bidireccional. Como se verá posteriormente, en los nodos de entrada y salida de IBM en la Node-RED de su propia Cloud requieren que se introduzca la clave API con su token de autenticación. Un ejemplo de datos generados al crear una aplicación es:

Detalles generados		Información de la clave de API	
Clave API	a-op8k2b-ule9r78dxe 	Descripción	Sensor_temperatura
Señal de autenticación	B2pO8ABY*2YNs?OhM2 	Rol	Aplicación estándar
		Caducidad	Nunca

Para terminar con la plataforma IoT, cabe mencionar que se ha elegido el plan Lite que ofrece gratuitamente IBM. Este plan limita la transferencia de datos a 200 MB, permite tener hasta 500 enlaces de aplicación y 500 dispositivos conectados.

3.2 Db2



DB2 on Cloud

IBM

IBM Cloud Db2 es una base de datos SQL de alto rendimiento alojada en la nube. Tiene una escalabilidad bastante flexible, pudiendo aumentar los nodos de CPU o el almacenamiento de forma independiente en función de las necesidades instantáneas de los servicios. Además se trata de un servicio de alta disponibilidad, con un tiempo en servicio de funcionamiento del 99.99%, siendo así muy robusta a posibles paradas no planificadas del sistema.

En este proyecto se ha utilizado como base de datos para almacenar los datos recibidos en el Node-RED de Bluemix y para usarla como respaldo para hacer consultas que alimentan las visualizaciones del dashboard. En total se han generado 5 tablas que agrupan todos los datos de interés para el problema abordado:

SENSORS_INFO

En esta tabla se almacena la información de todos los sensores que conforman la infraestructura, teniendo de cada uno sus coordenadas geográficas, qué tipo es y una descripción general:

SENSOR_ID	SENSOR_TYPE	SENSOR_LAT	SENSOR_LON	SENSOR_INFO
31	Temperature	37.375671	-6.001995	Aula 2.2 BIS

La consulta de creación de esta tabla es:

```
CREATE TABLE SENSORS_INFO (  
  SENSOR_ID INT PRIMARY KEY NOT NULL,  
  SENSOR_TYPE VARCHAR(100) NOT NULL,  
  SENSOR_LAT FLOAT NOT NULL,  
  SENSOR_LON FLOAT NOT NULL,  
  SENSOR_INFO VARCHAR(100) NOT NULL)
```

SENSOR_DATA

Aquí se guardan todas las lecturas de sensor recibidas en la plataforma de IBM, contando con el identificador único de lectura, el timestamp, el valor de la lectura realizada, sus unidades físicas, para relacionarlo con la tabla de magnitudes físicas, y el identificador de sensor, para relacionarlo con la tabla de información de sensores:

READ_ID	READ_DATE	READ_VALUE	READ_UNITS	SENSOR_ID
1560645013304	2019-06-16 00:30:13.0	22.6	°C	31
1560644973149	2019-06-16 00:29:33.0	21.94	°C	31
1560644978170	2019-06-16 00:29:38.0	28.25	°C	31

Para crear esta tabla se usó:

```
CREATE TABLE SENSOR_DATA (  
  READ_ID BIGINT PRIMARY KEY NOT NULL,  
  READ_DATE TIMESTAMP NOT NULL,  
  READ_VALUE FLOAT NOT NULL,  
  READ_UNITS VARCHAR(100) NOT NULL,  
  SENSOR_ID INTEGER NOT NULL)
```

PHYSICAL_PROPERTIES

Esta tabla sirve para guardar todas las magnitudes físicas que se miden o aparecen en la infraestructura de IoT, y tiene como variables el identificador de propiedad, su nombre, sus unidades y una descripción de estas últimas:

PROPERTY_ID	PROPERTY_NAME	PROPERTY_UNITS	UNITS_INFO
1	Temperatura	°C	Grados Celsius
2	Energía	kWh	Kilovatios hora

La creación de esta tabla se hizo con la consulta:

```
CREATE TABLE PHYSICAL_PROPERTIES (  
    PROPERTY_ID INTEGER PRIMARY KEY NOT NULL,  
    PROPERTY_NAME VARCHAR(100) NOT NULL,  
    PROPERTY_UNITS VARCHAR(100) NOT NULL,  
    UNITS_INFO VARCHAR(100) NOT NULL);
```

ENERGY_CONSUMPTION

Esta tabla está orientada a almacenar el consumo energético de los actuadores de la infraestructura, el cual se calculaba para el caso del aire acondicionado en el Node-RED de la Raspberry, tomándolo como la potencia que consume el dispositivo por el lapso de 3 segundos. Por lo tanto se guarda el identificador de lectura, el timestamp, la energía consumida en los 3 segundos precedentes, sus unidades y el identificador de actuador, para relacionar esta tabla con la tabla de información de actuadores:

ACTION_ID	ACTION_DATE	CONS_VALUE	CONS_UNITS	ACTUATOR_ID
1560690169903	2019-06-16 15:02:50.0	0.0	kWh	42
1560690172908	2019-06-16 15:02:53.0	0.0	kWh	42
1560690175912	2019-06-16 15:02:56.0	0.0	kWh	42

La tabla se creó con:

```
CREATE TABLE ENERGY_CONSUMPTION (  
    ACTION_ID BIGINT PRIMARY KEY NOT NULL,  
    ACTION_DATE TIMESTAMP NOT NULL,  
    CONS_VALUE FLOAT NOT NULL,  
    CONS_UNITS VARCHAR(100) NOT NULL,  
    ACTUATOR_ID INT NOT NULL);
```

ACTUATORS_INFO

Finalmente, la última tabla es bastante similar a la 'sensors_info', pero esta vez guardando la información de los actuadores:

ACTUATOR_ID	ACTUATOR_TYPE	ACTUATOR_LAT	ACTUATOR_LON	ACTUATOR_INFO
42	AC	37.375671	-6.001995	Aula 2.2 BIS





Y su consulta de creación es:

```
CREATE TABLE ACTUATORS_INFO (
  ACTUATOR_ID INT PRIMARY KEY NOT NULL,
  ACTUATOR_TYPE VARCHAR(100) NOT NULL,
  ACTUATOR_LAT FLOAT NOT NULL,
  ACTUATOR_LON FLOAT NOT NULL,
  ACTUATOR_INFO VARCHAR(100) NOT NULL);
```

Al igual que con la plataforma IoT, cabe mencionar que se ha utilizado el plan gratuito 'Lite' de este servicio, el cual está limitado a 200 MB de almacenamiento y 5 conexiones simultáneas.

3.3 Cloudant

IBM Cloudant es una base de datos NoSQL, distribuida optimizada para gestionar cargas de trabajo pesadas, típicas de las aplicaciones móviles y web de gran tamaño y con un gran crecimiento. En este proyecto se ha utilizado ya que el servicio de Node-RED lleva asociada una instancia de Cloudant para almacenar los documentos relativos a éste:

id	key	value
<input type="checkbox"/>  IARSapp/credential	IARSapp/credential	{ "rev": "4-c95cb596d8ff7391a0b...
<input type="checkbox"/>  IARSapp/flow	IARSapp/flow	{ "rev": "229-4ffac89da84be0399...
<input type="checkbox"/>  IARSapp/settings	IARSapp/settings	{ "rev": "6-d218733121583eef29...
<input type="checkbox"/>  _design/library	_design/library	{ "rev": "1-003b750938ae583f15...

3.4 Node-RED

IBM Cloud ofrece la posibilidad de inicializar una instancia de Node-RED en su nube, alojando todos los recursos necesarios para su correcto funcionamiento (como los documentos de Cloudant vistos en el subapartado anterior).

Node-RED

Flow-based programming for the Internet of Things

Para inicializar este servicio no hace falta más que ir a la página principal del producto, definir un nombre para la aplicación, registrar un usuario y contraseña específicos para la aplicación de Node-RED y finalmente decidir si se hace público o privado. En este caso se ha elegido exponerlo al público, pero sin capacidad de editar, para que los profesores puedan entrar con sus navegadores y se pueda mostrar con facilidad el dashboard en clase.

Al usar el plan 'Lite', la instancia de Node-RED tiene la gran limitación de tener como máximo 256 MB de memoria RAM. Esto ha dado problemas en el desarrollo del trabajo debido a la instalación nueva de los nodos de dashboard de Node-RED, que fueron instalados desde la paleta de gestión de la UI de Node-RED. Si se cierra la aplicación, o se reinicia por alguna causa externa como una caída del servidor, al volverse a iniciar deben instalarse todos los paquetes no básicos presentes en el flujo. Esto provocó que se instalasen todos los nodos de dashboard a la vez, sobrepasando el límite de RAM y conllevando la desconexión del sistema.

De cara a solucionarlo, se ha activado la opción de entrega continua de esta aplicación, lo cual habilita una herramienta Git para poder manipular los ficheros de la aplicación. Dentro del Git se ha modificado el archivo package.json de Node-RED, introduciendo en las dependencias el 'node-red-dashboard' con versión 2.15.4, para que sea instalado de forma secuencial junto al resto de dependencias al inicializarse la aplicación y evitar así pasar el límite de RAM.

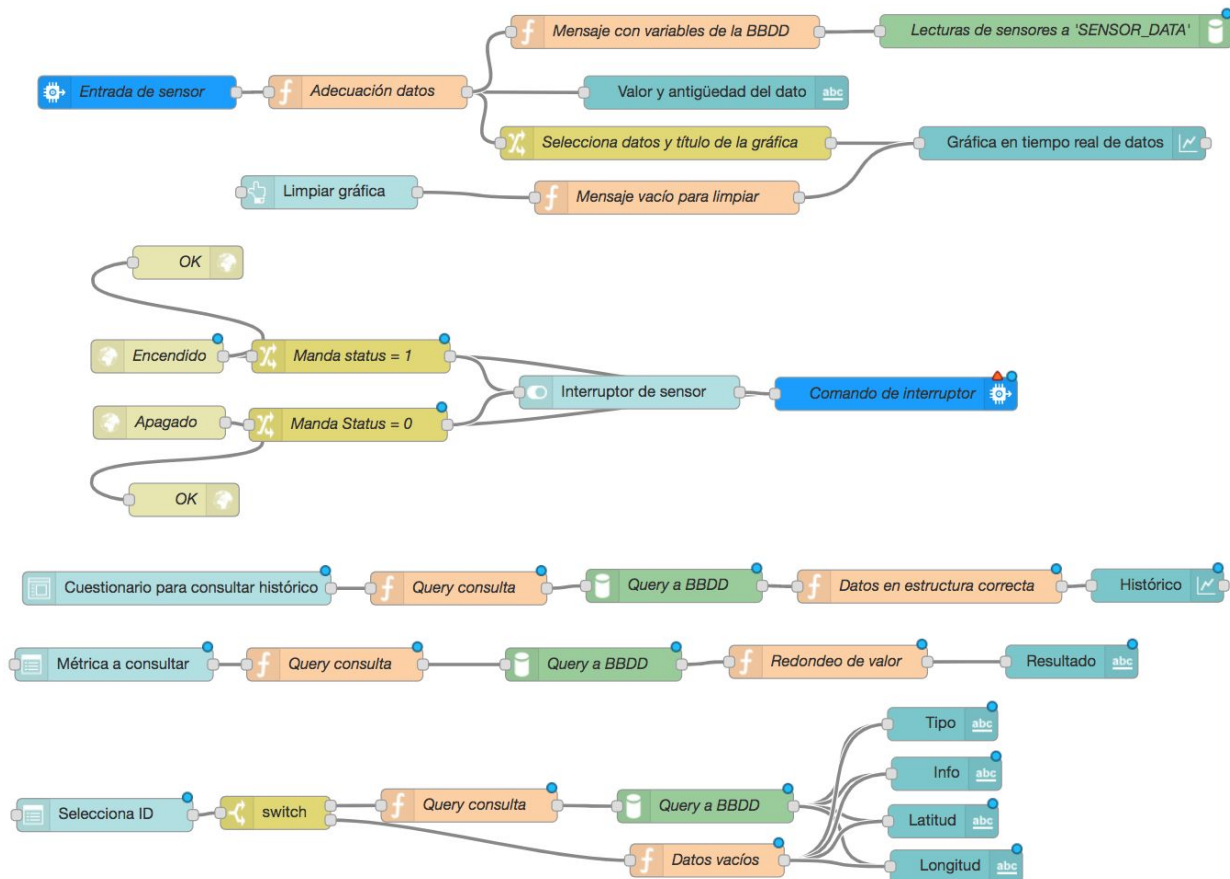
Una vez ya dentro de la UI de Node-RED, puede verse que es exactamente igual que las instancias ya iniciadas con anterioridad en el ordenador y la Raspberry con la salvedad de que ésta trae instalados muchos nodos propios de IBM para hacer uso de sus servicios.

Dicho esto, se procede a desarrollar los flujos necesarios para hacer funcionar la integración de los datos de la Raspberry con las herramientas de IBM Cloud y la creación del Dashboard donde se pueda visualizar la información y mandar órdenes a la Raspberry. Con estas premisas se van a crear dos flujos: uno para la integración de sensores y otro para la del aire acondicionado.

Se puede visualizar la instancia de Node-RED en: <https://iarsapp.eu-gb.mybluemix.net/red>

3.4.1 Flujo de sensorización

Para los sensores, el flujo de Node-RED queda así:



En este flujo se ven tres zonas principales: una formada por las dos primeras ramas, que tratan los datos de sensor recibidos en IBM, otra compuesta por las dos siguientes ramas, que actúan de interruptor de la recepción de datos, y la última formada por las tres últimas ramas, que están orientadas a componer el resto de la dashboard.

En la primera zona se reciben los datos por un nodo de entrada de IBM, el cual está configurado para leer los datos provenientes del sensor de temperatura de la Raspberry, con la clave API y configuración de dispositivo introducidas en la plataforma IoT. A continuación le pasa una función que pone buen formato a la fecha y el valor leído y lo envía a tres ramificaciones. La primera vincula los datos a almacenar en la base de datos con los nombres de las variables de la tabla 'SENSOR_DATA', enviándolos posteriormente a ésta. La segunda crea un campo de texto en la dashboard con el valor instantáneo del dato y su timestamp correspondiente. La

última rama pasa el valor leído en el sensor y su magnitud a una gráfica de líneas para ver la evolución en tiempo real. También se ve en la parte inferior que se ha introducido un botón para limpiar la gráfica de tiempo real, en caso de que esté ya muy sucia.

La segunda zona está orientada a hacer de interruptor para la recepción de los datos en la Raspberry, constando de dos opciones: por trama HTTP, como se hizo en las prácticas, y mediante un botón deslizante presente en el dashboard. La trama HTTP se coge de los directorios '/sens/on' y '/sens/off', respectivamente para encender y apagar. Además, si se acciona el interruptor por HTTP, también modifica el interruptor de la dashboard para que no se quede desincronizado. Finalmente, el comando se envía por un nodo de salida de IBM

La última zona tiene como objetivo hacer consultas a la base de datos para obtener el histórico de datos leídos y información del sensor. La primera rama crea un cuestionario en la dashboard donde se elige una fecha y hora de comienzo, una fecha y hora de fin y un ID de sensor, para mandar una query a la base de datos solicitando el histórico de los datos de ese dispositivo en ese periodo concreto, el cual se representa en una gráfica de líneas:

```
1 // Carga las variables que forman la query
2 var fini = Math.round(new Date(msg.payload['Fecha_ini']).getTime()/1000) + 2*3600
3 var ffin = Math.round(new Date(msg.payload['Fecha_fin']).getTime()/1000) + 2*3600
4 var senid = msg.payload['sensor_Id']
5 var hini = msg.payload['hora_ini']
6 var hfin = msg.payload['hora_fin']
7
8 // Pone en buen formato la fecha
9 fini = new Date(fini*1000).toISOString();
10 ffin = new Date(ffin*1000).toISOString();
11
12 fini = fini.split('T')[0]
13 ffin = ffin.split('T')[0]
14
15 // Modifica los valores de las variables de flujo para ser usadas en la consulta
16 // de métrica
17 flow.set(['fini', 'ffin', 'hini', 'hfin', 'senid'],
18         [fini, ffin, hini, hfin, senid])
19
20 // Crea la query
21 var query = ""
22
23 query = "SELECT READ_DATE, READ_VALUE, "+
24         "PROPERTY_NAME FROM SENSOR_DATA AS A "+
25         "LEFT JOIN PHYSICAL_PROPERTIES AS B "+
26         "ON A.READ_UNITS = B.PROPERTY_UNITS "+
27         "WHERE READ_DATE >= '"+fini+"' "+
28         hini.toString()+" :00:00' AND "+
29         "READ_DATE <= '"+ffin+"' "+
30         hfin.toString()+" :00:00' AND "+
31         "SENSOR_ID = '"+senid.toString()+"' "+
32         "ORDER BY READ_DATE;"
33
34 // Guarda la query
35 msg.payload = query
36 return msg;
```

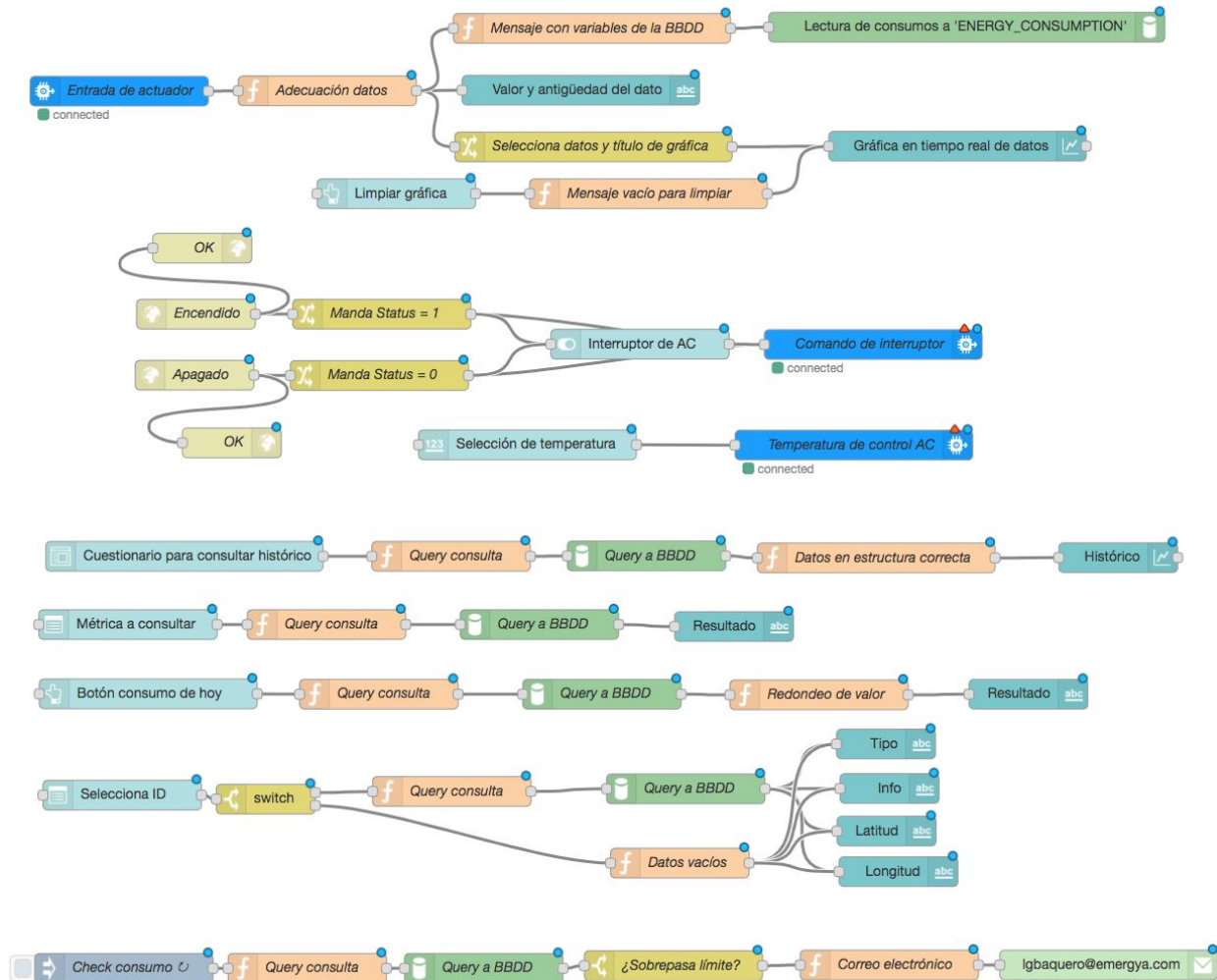
En medio de la rama hace falta aplicar una función que lee los datos devueltos por la base de datos, que se encuentran estructurados en varios campos anidados, y los convierte en la estructura de datos que es capaz de interpretar el nodo de la gráfica:

```
1 // Inicializa variables necesarias
2 var series = ["Value"];
3 var labels = ["Labels"];
4 var data = [];
5 var i, len, string;
6
7 // Bucle que recorre el objeto obtenido
8 // de la base de datos
9 for (i = 1, len = msg.payload.length, string = ""; i < len; i++) {
10 // Actualiza data con el timestamp y valor de lectura de cada fila
11 data.push({"x":new Date(Math.round(new Date(msg.payload[i].READ_DATE).getTime()/1000)-2*3600)*1000), "y":msg.payload[i].READ_VALUE});
12 }
13
14 data = [data];
15
16 // Guarda el nombre de la magnitud para la gráfica
17 msg.topic = msg.payload[1].PROPERTY_NAME;
18
19 // Guarda los datos en el mensaje
20 msg.payload = [{series, data, labels}];
21 return msg;
```

La siguiente rama, partiendo de las opciones elegidas en el cuestionario anterior que han sido almacenadas en variables de flujo, crea un desplegable para elegir una métrica (valor máximo, mínimo y medio) que será consultada a la base de datos para el periodo elegido anteriormente. Finalmente, la última rama crea otro desplegable que permite elegir un dispositivo entre los existentes, para consultar en la base de datos la tabla 'SENSORS_INFO' y sacar así la visualización de 4 campos de texto, con los datos presentes en dicha tabla.

3.4.2 Flujo de actuadores

En este caso, el flujo es en buena parte parecido al del flujo de sensores, con alguna modificación en lo que respecta a los datos visualizables en el dashboard, la introducción de un control adicional para elegir la temperatura de control del aire acondicionado, y la inclusión de un aviso por correo electrónico cuando el consumo eléctrico del día ha sido excesivo. Se va a intentar no repetir la explicación de elementos ya vistos en el flujo anterior, quedando este flujo con la siguiente estructura:



En este flujo pueden diferenciarse cuatro zonas principales, siendo las tres primeras equivalentes a las del flujo de sensores, más una cuarta zona para el envío de avisos por correo electrónico. La primera zona, por ejemplo, es prácticamente igual, cambiando que los campos de los datos recibidos y las variables de la tabla 'ENERGY_CONSUMPTION' son distintos, debiendo modificarlos dentro de los propios nodos.

En la segunda zona, los directorios para encender y apagar ahora son '/ac/on' y '/ac/off'. Además, se ha incluido un seleccionable numérico para elegir la temperatura de control del aire acondicionado, la cual será usada por éste para variar la velocidad de su compresor para acercar la temperatura del habitáculo a ésta.

En la tercera zona se ha incluido la opción de consultar en la base de datos cuál ha sido el consumo eléctrico a lo largo del día en cuestión, de cara a monitorizar si está habiendo un consumo excesivo del aire acondicionado.

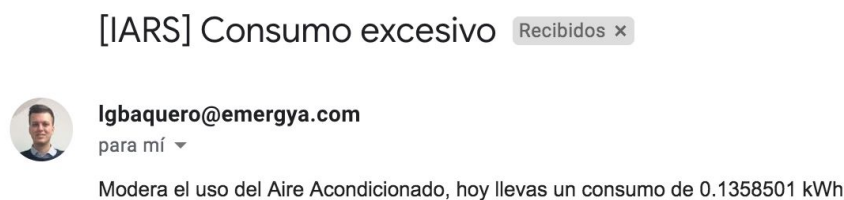
Vinculada directamente a la anterior, la cuarta zona es un sistema de notificaciones por correo electrónico para regular el consumo energético excesivo. Cada 30 minutos inyecta un mensaje para consultar en la base de datos el consumo acumulado en el día en cuestión y, si sobrepasa un límite que se ha puesto en 3 kWh, envía un correo electrónico al usuario avisándole, el cual se redacta en el nodo de 'Correo electrónico':

```

1 // Redondea el valor a 7 decimales para que sea legible
2 msg.payload.CONSUMO = Math.round(msg.payload.CONSUMO*10000000)/10000000
3
4 // Redacta el correo electrónico
5 msg.payload = "Modera el uso del Aire "+
6               "Acondicionado, hoy llevas un "+
7               "consumo de "+msg.payload.CONSUMO.toString()+
8               " kWh";
9
10 // Asunto del correo electrónico
11 msg.topic = "[IARS] Consumo excesivo"
12
13 return msg;












```

Para ello se ha configurado el servidor smtp de mi cuenta de Gmail para que pueda recibir este tipo de correos. Un ejemplo de correo enviado sería:



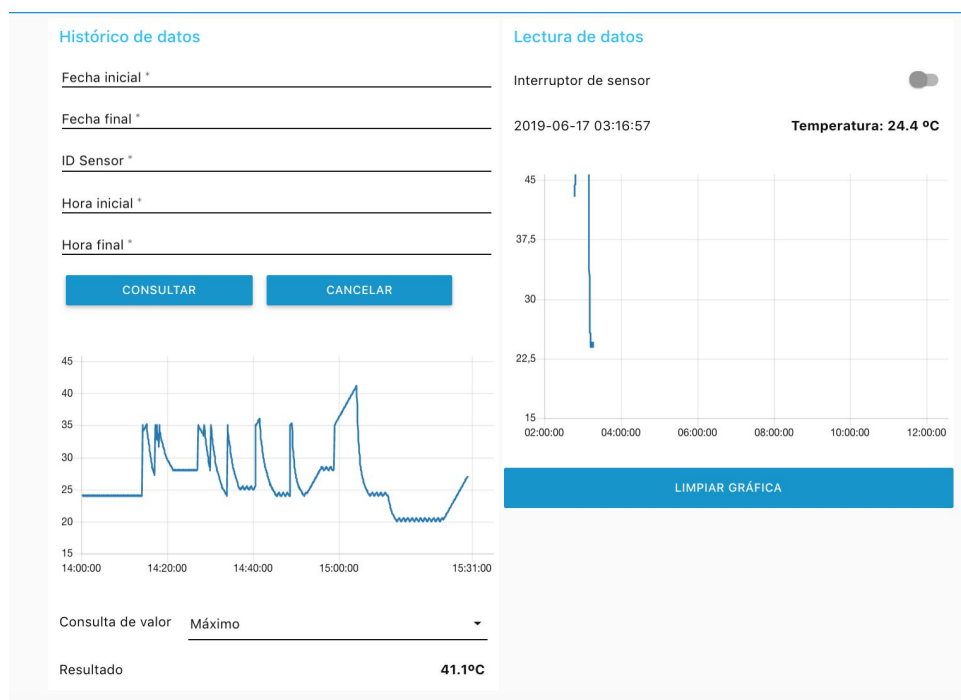
4. Dashboard

Una vez se tiene implementado todo el proceso de integración entre la Raspberry y Node-RED, se tiene terminada la Dashboard, la cual se puede visitar en <https://iarsapp.eu-gb.mybluemix.net/ui>. Ésta tiene la siguiente estructura:

▼  Sensorización
>  Histórico de datos
>  Lectura de datos
▼  AC
>  Histórico de consumo
>  Control de AC
>  Lectura de consumo
>  Consumo de hoy
▼  Dispositivos
>  Actuadores
>  Sensores

Como se ve, hay tres paneles principales, que son Sensorización, AC y Dispositivos:

4.1 Sensorización

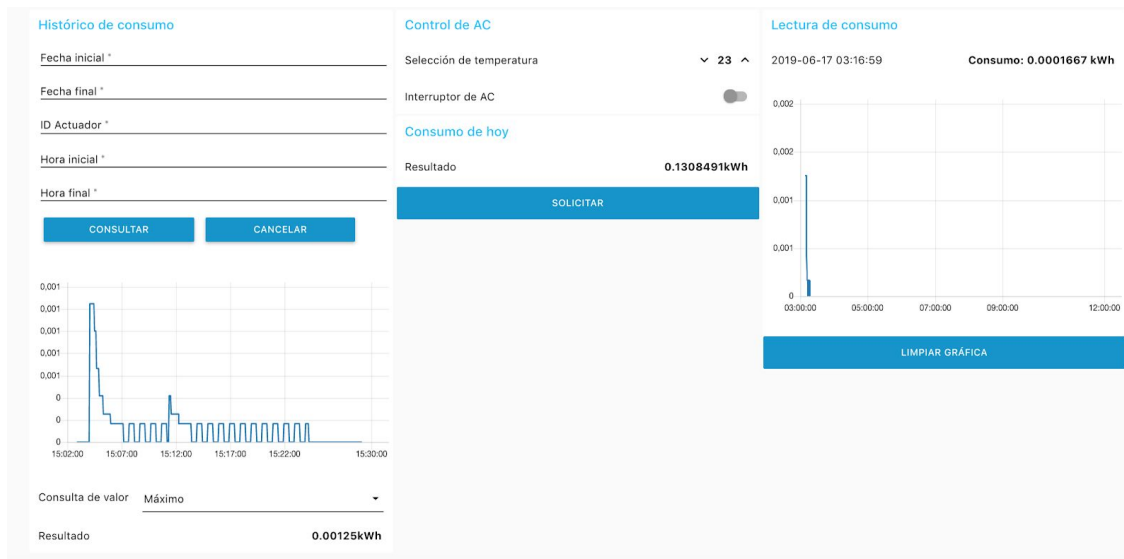


En este panel se tiene acceso a los datos emitidos por los sensores de la red, y cuenta con dos grupos de objetos. El primero, llamado 'Histórico de datos', nos da acceso a la base de datos para hacer consultas de históricos de datos. En primer lugar aparece el cuestionario que nos pide la información para consultar la base de datos y justo debajo aparece el resultado de la consulta en forma de serie temporal. También nos permite elegir una métrica entre valores máximo, mínimo y medio del periodo con un desplegable, mostrando debajo el resultado.

A la derecha tenemos el grupo de 'Lectura de datos'. Primero aparece el interruptor deslizable para encender o apagar la recepción de datos en la Raspberry. Finalmente, tenemos la gráfica en tiempo real de los datos recibidos por ésta, además del botón para limpiarla.

En todo momento, el nombre de la magnitud del dato leído y sus unidades son obtenidos de los datos recibidos de la Raspberry o de los obtenidos de la base de datos, en función del dispositivo del que se trate.

4.2 AC



Es el panel equivalente al de sensorización pero para la visualización y control del aire acondicionado. Esta vez las gráficas de series temporales no corresponden a las magnitudes leídas por el sensor, sino al consumo que ha tenido el aire acondicionado de forma instantánea y en el histórico de datos de la base de datos (en kWh). Además, en la consulta de valor también podemos seleccionar el consumo acumulado del periodo concreto ya que es de utilidad.

En la parte central del panel aparecen el grupo de control y el de consumo diario. En el grupo de control, además del interruptor para apagar y encender el dispositivo, también aparece un selector de la temperatura de control. Y en el grupo de 'Consumo de hoy' se puede pulsar el botón que aparece en la imagen para consultar cuánto se lleva consumido en el día concreto.

4.3 Dispositivos

Sensores		Actuadores	
Selecciona ID	31	Selecciona ID	42
Tipo	Temperature	Tipo	AC
Info	Aula 2.2 BIS	Info	Aula 2.2 BIS
Latitud	37.375671	Latitud	37.375671
Longitud	-6.001995	Longitud	-6.001995

Finalmente, en este panel se tiene acceso a los datos de las tablas 'SENSORS_INFO' y 'ACTUATORS_INFO'. Con los dos desplegados se puede seleccionar el ID del dispositivo que queramos consultar, mostrándonos debajo el tipo de dispositivo que es, la descripción y sus coordenadas geográficas.

5. Conclusiones

En el trabajo presentado se ha abordado de forma completa cómo hacer una integración de una red con sensores y un dispositivo actuador (aire acondicionado), con los datos enviados desde el servidor (la Raspberry) a una plataforma en Cloud.

Se ha obtenido una visión general de las posibilidades que ofrece la Cloud de IBM para trabajar en proyectos de IoT, ya sea para la gestión de dispositivos y el almacenamiento, como para poder crear aplicaciones funcionales que permitan visualizar los datos y accionar los elementos de la infraestructura física.