

Informe de trabajo práctico final: Estación de alarma

Ingeniería en Computación

Cloud Computing & Cloud Robotics

06/12/2019

Integrantes	
Nombre	Nro Alu
Valentín Cañete	IC: 0596/8 - ATIC:16518/1
Raul Heredia	IC: 0308/5
Orlando Peraza	IC: 0164/9
Ezequiel Humar	IC: 1297/4

Índice

Introducción	1
Sensores	1
Actuadores	2
Configuración de la máquina virtual	2
Instalación de paquetes	3
Configuración de Server	6
Configuración del nodo publicador	7
Diagramas de flujo	7
Raspberry	7
Sensor de humedad	8
Sensor de movimiento	9
Sensor de temperatura	9
Alarma (buzzer)	10
Dashboard - Interfaz	11
DashBoard- NodeRed	12
Sección de Alarma	12
Subsección Humedad	15
Subsección Movimiento	17
Subsección Temperatura	19
Resultados	21

Introducción

Se quiere realizar una alarma para un invernadero/huerta que detecte y de aviso en caso de detectar: movimiento, excesiva humedad en la tierra y excesiva temperatura en el ambiente.

La alarma informa estos eventos cuando está activada mediante sonido, provocado por un buzzer y tres leds de distintos colores. El led rojo se prende cuando se detecta exceso de temperatura en el ambiente; el led verde se prende cuando se detecta exceso de humedad en la tierra y el led amarillo se prende cuando se detecta movimiento.

Sensores

En la siguiente tabla (Tabla 1) podemos ver los sensores utilizados para el proyecto.





Tabla 1. *Sensores utilizados en el sistema*

Sensor de movimiento PIR HC-SR501	Sensor de temperatura DHT-22	Sensor de humedad YL-69
		

Actuadores

En la siguiente tabla (Tabla 2) podemos ver los actuadores utilizados para el proyecto.

Tabla 2. *Actuadores utilizados en el sistema*

Led rojo	Led verde	Led amarillo	Buzzer
			

Además se requiere mostrar la información sensada en una interfaz web. Para esto se deben transmitir estos datos a una computadora que sea la que exhiba los mismos.

Para cumplir con este requerimiento se utilizará el protocolo de red MQTT, el cuál facilita la comunicación en proyectos de internet en las cosas.

Configuración de la máquina virtual

Se usará una máquina virtual con el software Node-RED precargado que se utilizará para la programación del flujo, además el broker Mosquitto se utilizará para realizar la comunicación entre de los sensores (DHT-22, PIR HC-SR501 e YL-69), leyendo los mensajes o envía comandos a los actuadores (LED's y Buzzer).

Esta máquina virtual debe ser configurada para el acceso a *Node-RED*, *Mosquitto* y *SSH* a través del macheo entre cada máquina anfitrión y cada máquina virtual. Para esto hay que dirigirse al apartado de Red en las configuraciones de la máquina virtual -> Avanzadas -> Reenvío de puertos y configurar la IP de anfitrión e invitado para los puertos para realizar la comunicación *SSH*, del *Node-RED*. Donde la IP de anfitrión es la IP de la PC donde se trabajará y la de invitado es la IP de la máquina virtual. (ver Fig. 1)

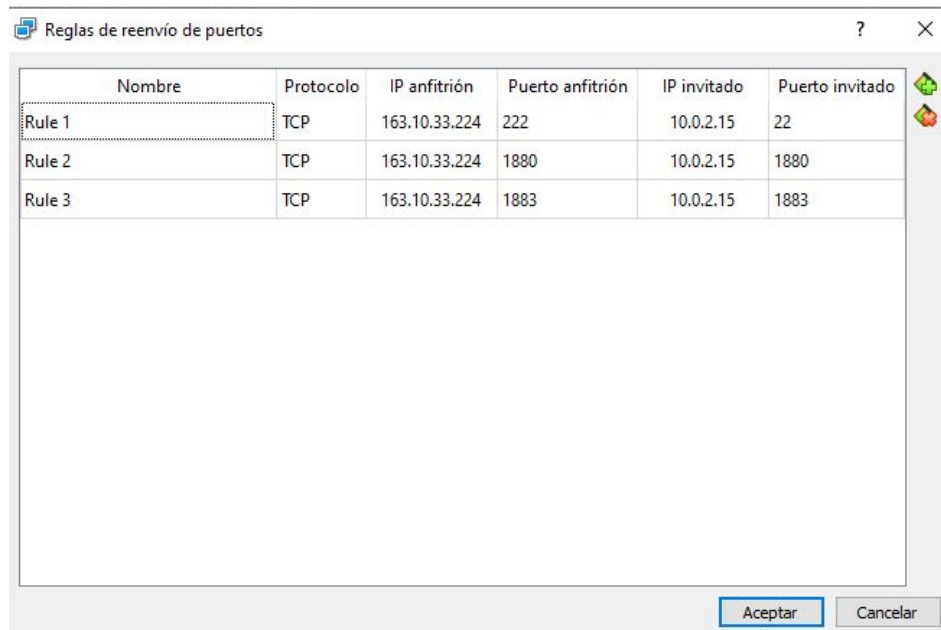


Fig 1. Configuración de reglas de puertos de la Máquina virtual

Instalación de paquetes

Para realizar el proyecto se necesitaron paquetes extras que se instalaron para poder hacer uso de los distintos sensores. En la siguiente imagen (Fig 2) se detallan dichos paquetes:

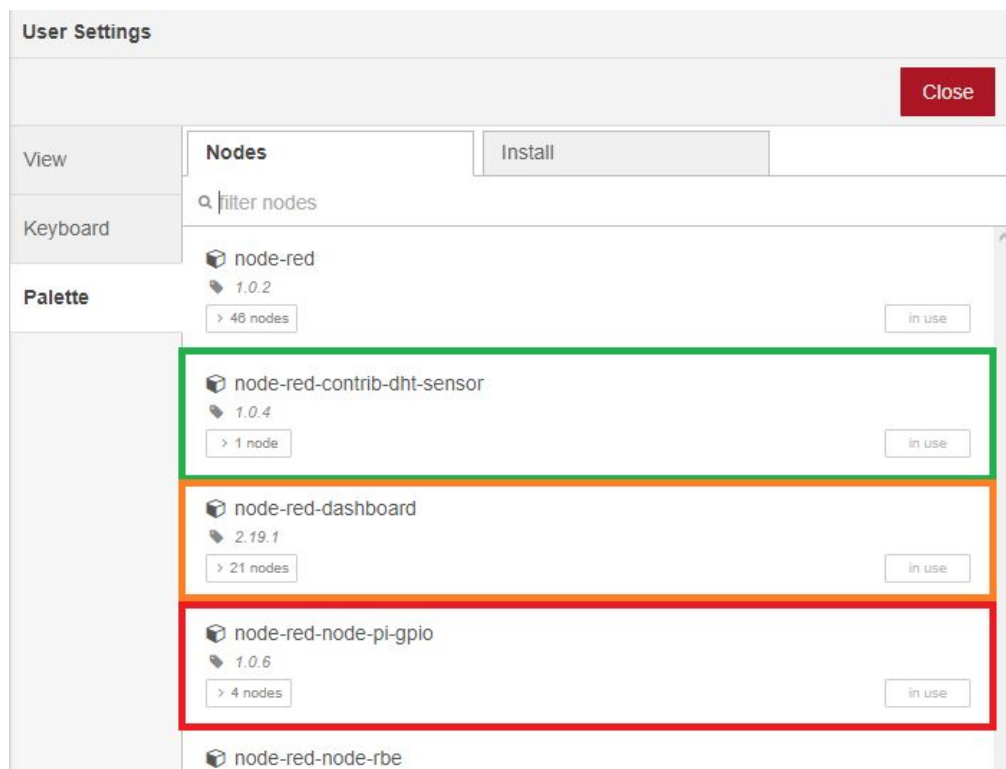


Fig 2. Instalación de Nodos Adicionales

El paquete “**node-red-contrib-dht-sensor**” se instaló para poder hacer uso del sensor de temperatura DHT-22. Los paquetes del sensor se agregan a la paleta de la *Raspberry Pi* como se muestra en la siguiente imagen (Fig 3):

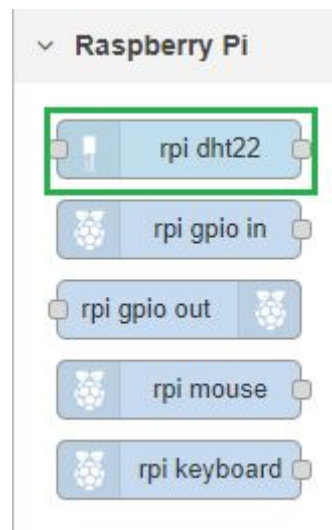


Fig 3. *Nodo sensor DHT22*

El paquete “**node-red-dashboard**” se instaló para poder utilizar algunos componentes del paquete como el componente “gauge”. En la siguiente imagen se muestran algunos de los componentes (Fig. 4):

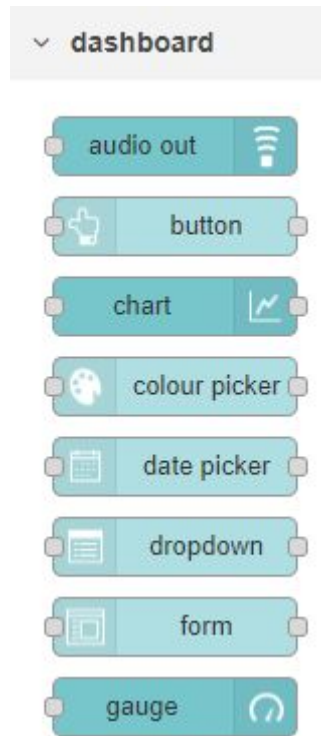


Fig 4. *Nodo Instalado de DashBoard*

El paquete “**node-red-node-pi-gpio**” se instaló para hacer uso de los componentes de la *Raspberry Pi* como por ejemplo sus pines. En la siguiente imagen se muestra la imagen de la paleta con los distintos componentes a utilizar de la *Raspberry Pi*:

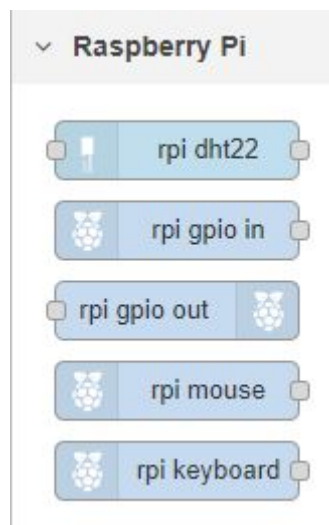


Fig 5. *Nodo Instalado de RaspBerry*

Configuración de Server

El protocolo utilizado para enviar mensajes es mqtt, y se decidió utilizar el broker que se encuentra corriendo en la raspberry. Por lo tanto la configuración del mismo se puede observar a continuación (Fig. 6)

The screenshot shows a web-based configuration interface for an MQTT broker. At the top, there's a breadcrumb trail: "Edit mqtt out node > Edit mqtt-broker node". Below this are three buttons: "Delete", "Cancel", and "Update" (which is highlighted in red). The main section is titled "Properties" with a gear icon and a document icon. It contains several fields and tabs. The "Name" field is set to "broker_raspberry". There are three tabs: "Connection" (selected), "Security", and "Messages". Under the "Connection" tab, the "Server" field is "163.10.53.57" and the "Port" field is "1883". There is an unchecked checkbox for "Enable secure (SSL/TLS) connection". The "Client ID" field is set to "Leave blank for auto generated". There is a "Keep alive time (s)" field set to "60" and a checked checkbox for "Use clean session". At the bottom, there is an unchecked checkbox for "Use legacy MQTT 3.1 support".

Fig 6. Configuración de Broker en raspBerry

Para establecer la comunicación entre los brokers es necesario configurar un Servidor (En nuestro caso configuramos la Raspberry como Servidor). La configuración se realiza una sola vez al incluir el primer broker y en los demás brokers solo se selecciona ese Servidor antes configurado.

Configuración del nodo publicador

Podemos observar en la Fig. 7, la configuración del nodo mqtt de salida:

Edit mqtt out node

Delete Cancel Done

Properties

Server broker_raspberry

Topic humedad

QoS 2 Retain

Name Estado humedicidad

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Fig 7. configuración Nodo MQTT

Como muestra la imagen, se elige el servidor configurado anteriormente y se establece un “Topic” a través del cual se va a realizar la comunicación entre los sensores y actuadores de humedad. Para los demás sensores y actuadores se llevarán a cabo los mismos pasos identificando cada grupo de sensores y actuadores de cada funcionalidad con un “Topic” distinto.

Diagramas de flujo

Raspberry

En la placa Raspberry se encontrarán conectados los sensores y actuadores. A continuación se procede a explicar los diagramas de flujo para leer los datos de los sensores y controlar los actuadores.

Sensor de humedad

Podemos observar en la Fig. 8, la sección de implementación del sensor de humedad:

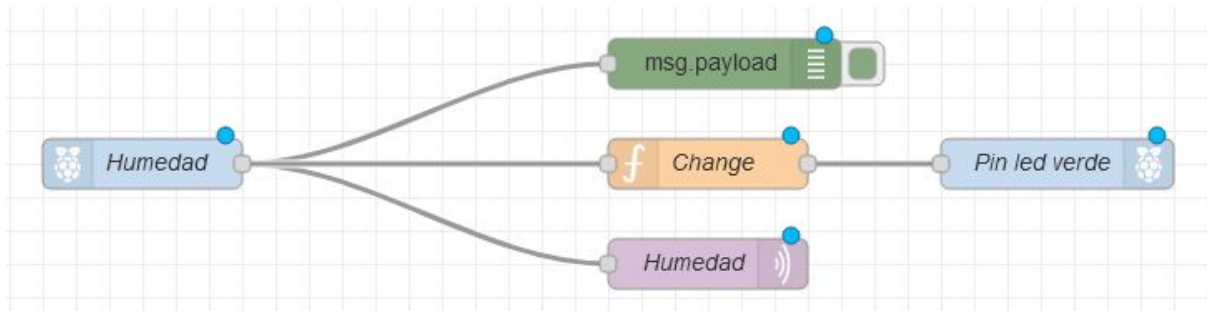


Fig 8. *Nodos en RB para subsección humedad.*

Se recibe por una de las entradas de la placa el estado que indica el sensor de humedad.

El sensor funciona enviando un bit alto cuando detecta baja humedad y un bit bajo cuando detecta alta humedad. El umbral de baja o alta humedad se configura en el mismo sensor mediante un potenciómetro.

El bit que recibe pasa por una función 'Change', la cual invierte el valor recibido (los cero los convierte en uno y los uno los convierte en cero), para luego enviarle el valor a una salida donde se encuentra conectado el LED verde. Por lo tanto, el LED se enciende cuando se detecta alta humedad.

También se publica en el servidor MQTT el valor recibido por el sensor.

El código de la función 'Change' es el siguiente:

```
if (msg.payload === 0) {  
    msg.payload = 1;  
}  
else {  
    msg.payload = 0;  
}  
return msg;
```

Sensor de movimiento

Podemos observar en la Fig. 9, la sección de implementación del sensor de presencia:

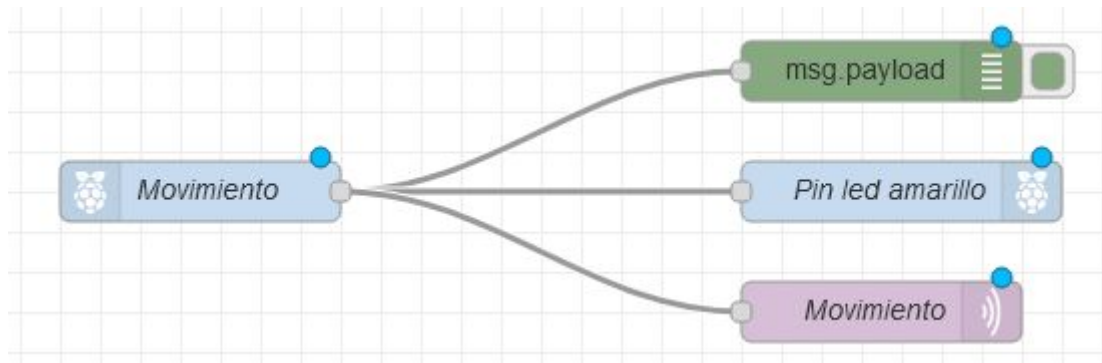


Fig 9. *Nodos de subsección de movimiento en RB*

El sensor de movimiento envía un bit alto cuando detecta movimiento y permanece un periodo de tiempo en ese estado, para luego volver a un estado bajo.

En una entrada se lee el estado del sensor, luego el mismo se lo escribe en una de las salidas, la cual está conectada al LED amarillo. Entonces este led se enciende durante un periodo de tiempo cuando se detecta movimiento.

A su vez se publica el estado del sensor en el servidor MQTT.

Sensor de temperatura

Podemos observar en la Fig. 10, la sección de implementación del sensor de temperatura:

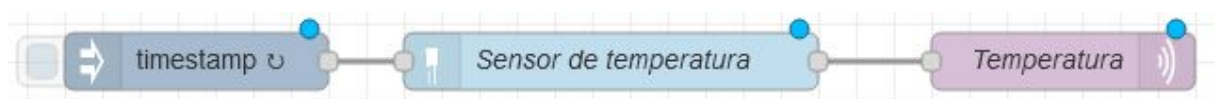


Fig 10. *Nodo de control para captura de datos en sensor de temperatura*

Para utilizar el sensor de temperatura se añade la librería ‘node-red-contrib-dht-sensor’ al proyecto de Node-RED, la cual facilita la lectura de la temperatura sensada.

Por cómo está hecha la librería, la forma de leer los datos del sensor es enviando un dato y luego leer los datos que se reciben. Por lo tanto se le envía al sensor de temperatura un ‘timestamp’, para luego recibir el dato de temperatura y publicarlo al servidor MQTT.

Cuando se considera que la temperatura es excesiva se debe encender el LED rojo.

En la computadora donde se muestran los datos se configura el umbral de temperatura excesiva, por lo tanto esta recibe el dato de la temperatura y si se considera que supera el límite se publica un '1' en el servidor MQTT, o si no se lo supera se publica un '0'.

En la siguiente figura (Fig. 11) se observa el receptor MQTT donde recibe el estado publicado por la computadora. Este estado se envía a la salida donde se conecta el LED rojo, por lo tanto este se encenderá cuando se supere el límite de temperatura o permanecerá apagado en caso de no superarse este límite.

A su vez se publica nuevamente el estado en el servidor MQTT a modo de confirmarle a la computadora el estado del LED.

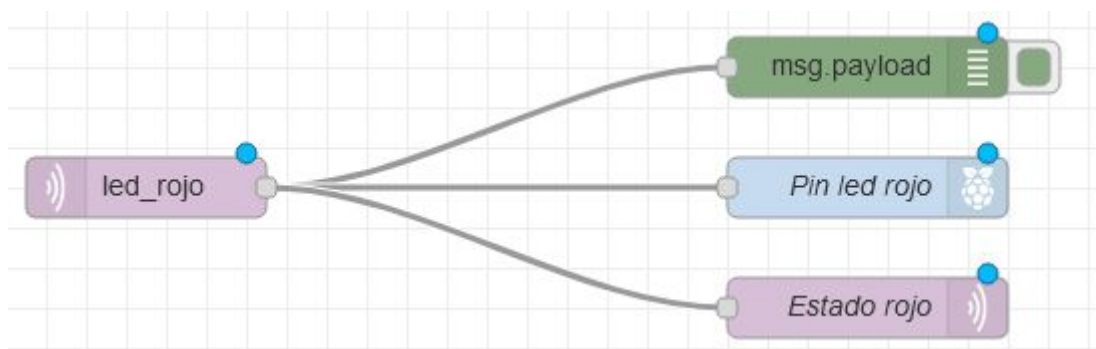


Fig 11. *Nodos de Control de Led Rojo de umbral superado en temperatura.*

Alarma (buzzer)

Podemos observar en la Fig. 12, la sección de implementación del buzzer:

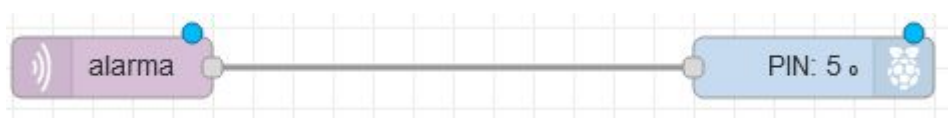


Fig 12. *Sección de nodos de control para Buzzer*

Se recibe por un receptor MQTT el estado en que debe encontrarse la alarma. Si se recibe un '1' la alarma debe sonar, es caso de recibirse un '0' la misma debe encontrarse en silencio.

El estado que se recibe se lo envía a la salida donde se encuentra conectado el buzzer, por lo tanto el mismo sonará en caso de recibirse un '1' y en caso de recibirse un '0' no lo hará.

Dashboard - Interfaz

Para revisar los el estado de los sensores y configurar la alarma se requiere un panel indicador o de control (Fig. 13). Esta interfaz es creada a través del Node-RED que se encuentra corriendo en la máquina virtual.

Su implementación se debe importar la librería “node-red-dashboard” y hacer uso de los nodos provistos por ella.

Dicha interfaz permite:

- Activar y desactivar la alarma: los eventos se informan pero, no se enciende la sirena de alerta.
- Visualizar el estado de la alarma: activada o desactivada.
- Visualizar el estado de la humedad de suelo: seco o mojado.
- Detectar presencia de intruso: hay movimiento o no hay movimiento.
- Controlar un panel de temperatura:
 - indica si hay riesgo o no de incendio de acuerdo a un valor de tolerancia máxima de temperatura seteado por el usuario.
 - indica el valor de temperatura actual
 - permite ingresar el valor de tolerancia máxima de temperatura
 - se indica el valor de tolerancia máxima de temperatura ingresado.

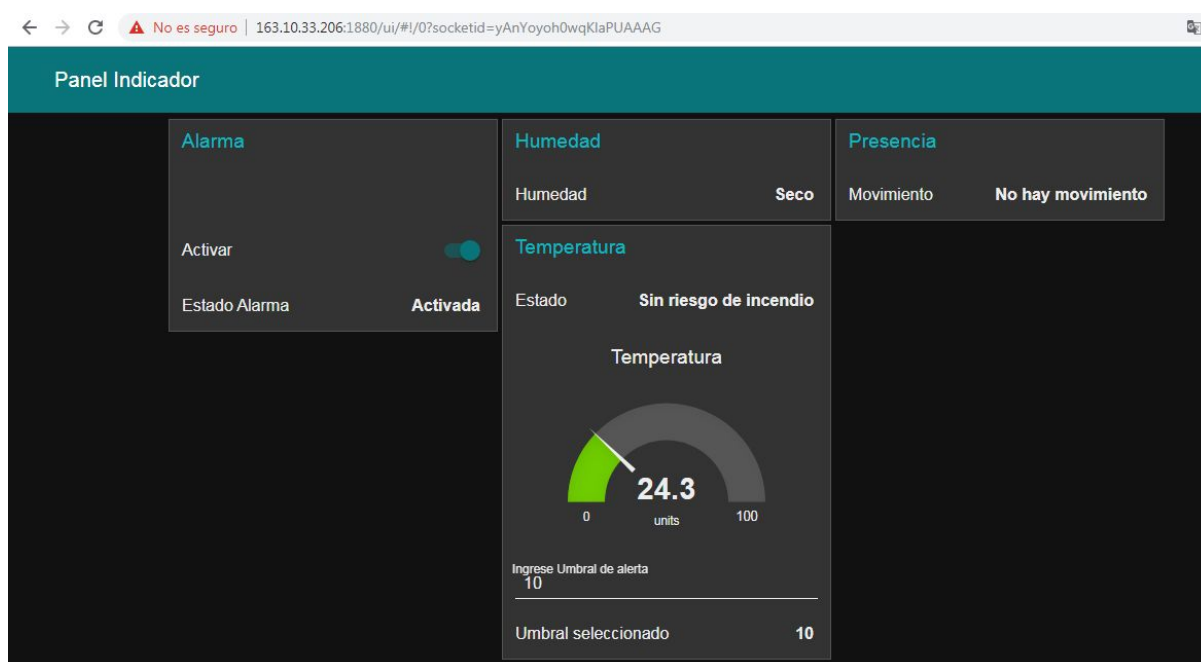


Fig 13. DashBoard del sistema.

DashBoard- NodeRed

Para la Implementación del DashBoard sobre Node Red se decidió implementar todo en un único FLOW el cual tendrá cada una de las secciones Alarma, Humedad, Temperatura y Presencia con sus respectivos nodos.

Sección de Alarma

Esta sección (ver fig 14) le permitirá activar o desactivar la alarma del sistema, el cual seguirá mostrando valores pero si la alarma se encuentra activada, y se supera algún umbral de los subsistemas, encendera una alerta sonora.

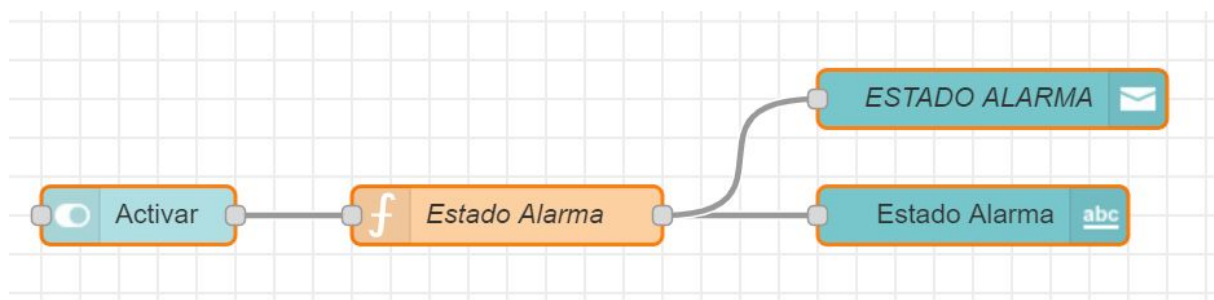


Fig 14. Sesión del DashBoard de la alarma

En esta sección se colocó un nodo Switch (ver fig. 15) el cual le permitirá al usuario activar o desactivar la alarma, luego se introdujo un Nodo Function el cual contiene una variable global, que si la alarma se encuentra encendida se setea dicha variable en 1, para luego dependiendo de la subsección que supere el umbral seteado iniciara la alarma sonora .

Edit switch node

Delete Cancel Done

Properties

Group [Panel Indicador] Alarma

Size auto

Label Activar

Tooltip optional tooltip

Icon Default

→ Pass through msg If payload matches new state: ☐

Indicator Switch icon shows state of the output

When clicked, send:

On Payload true

Off Payload false

Topic

Name

Fig 15. Nodo switch

A su vez se enviará el mensaje “Activada”, para que el usuario identifique el estado en el cual se encuentra la alarma el cual está implementado mediante el nodo Estado Alarma.

A continuación se puede observar el fragmento de código que utiliza la función de alarma.

```
if (msg.payload == 1) {
  msg.payload= "Activada";
  global.set("alarma", 1);
}else{
  msg.payload="Desactivada";
  global.set("alarma", 0);
}
return msg;
```

Como se puede ver en msg.payload se guarda el mensaje y global.set(nom_var_global, valor) es la función que setea un valor en una variable la variable global.

También se incluye un nodo Notification , el cual al activar o desactivar la alarma aparece un mensaje en la esquina superior izquierda (ver fig 16) indicando el estado de la alarma.

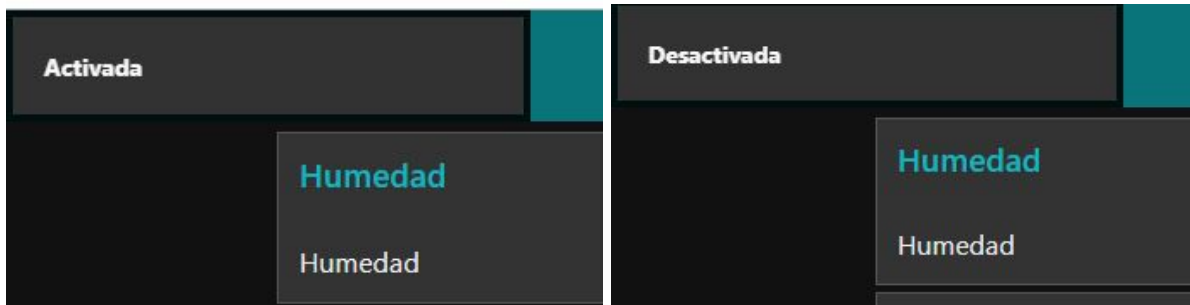


Fig 16. Ventana emergente

A continuación se observa la configuración del nodo de la ventana emergente (Fig 17):

Fig 17. Nodo Notification

Subsección Humedad

En esta subsección se interpretarán los valores recibidos a través de MQTT, datos enviados desde la RB aportados por el sensor de humedad de tierra el cual indicará si el suelo se encuentra mojado o seco.

Estos datos serán mostrados en el DashBoard para así informarle al usuario el estado del suelo.

En la siguiente figura 18 se muestra el FLOW que se encarga de controlar los mensajes para así brindarle información al usuario.



Fig 18. Subsección de nodos del dashboard de humedad

En el FLOW de la subsección de humedad se incluye un nodo MQTT (fig. 19) llamado *Humedad* el cual recibe por protocolo MQTT el valor 0 (cero) o 1(un) desde la RB dependiendo si el sensor de humedad se encuentra mojado en sus terminales.

El Broker utilizado es el que denominamos RASPBERRY con ip 163.10.53.57 y port 1883, y en este caso el tópico a utilizar es *humedad*.

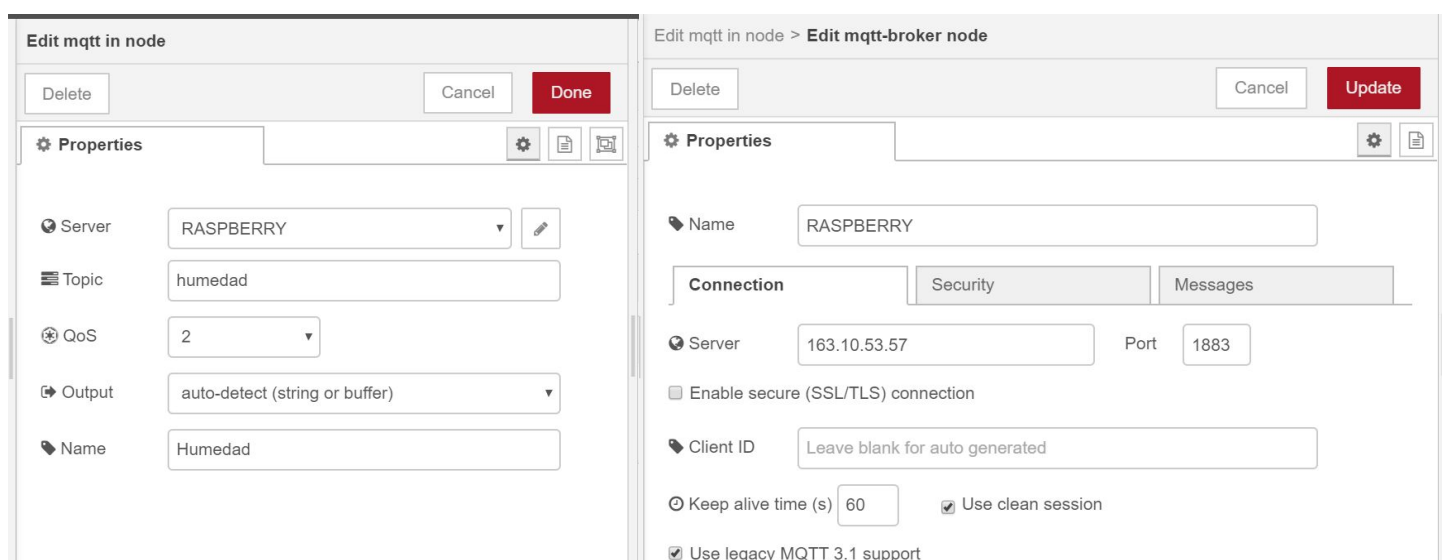


Fig 19. Configuración de Nodo MQTT y broker.

Luego se implementa un nodo Function al cual denominamos Estado Humedad, este retorna un mensaje “Mojado” o “Seco” el cual se mostrará en el DashBoard y así informar al usuario.

```
if (msg.payload == 0) {  
  msg.payload= "Mojado"  
} else {  
  msg.payload="Seco"  
}  
return msg;
```

Luego de obtener el mensaje del estado del sensor de humedad se procede a mostrarlo en el DashBoard para así el usuario poder visualizarlo.

A su vez, la salida de la función anterior, se toma como entrada para otro nodo Function para ésta interpretar el valor y en base a la variable global de la alarma, enviar mediante MQTT un aviso a la RB para hacer sonar la alerta sonora en caso que la alarma se encuentre activada y el suelo se encuentre mojado.

```
var alarma = global.get("alarma");  
if ((msg.payload == "Mojado")&&(alarma==1)) {  
  msg.payload= 1  
} else {  
  msg.payload=0  
}  
return msg;
```

El nodo MQTT implementa al igual que en los casos anteriores el Broker RASPBERRY pero el tópico a usar será “alarma”(fig 20).

Edit mqtt out node

Delete Cancel Done

Properties

Server: RASPBERRY

Topic: alarma

QoS: Retain:

Name: Name

Tip: Leave topic, qos or retain blank if you want to set them via msg properties.

Fig. 20. Tópico para nodo mqtt de Alarma

Subsección Movimiento

La implementación de este sistema es muy similar al apartado de humedad (Fig. 21):

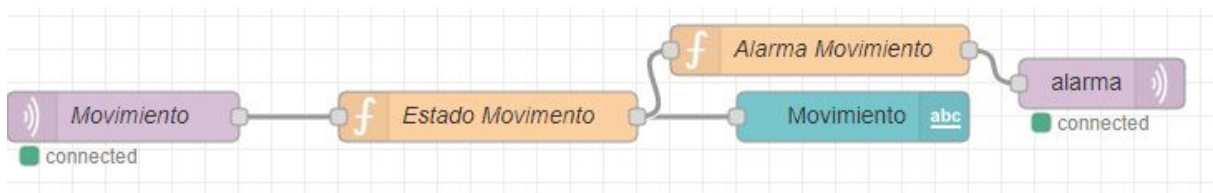


Fig 21. Nodos del DashBoard de la subsección Movimiento.

En esta oportunidad el cliente se suscribe a un nuevo tópico denominado “movimiento” (Fig. 22):

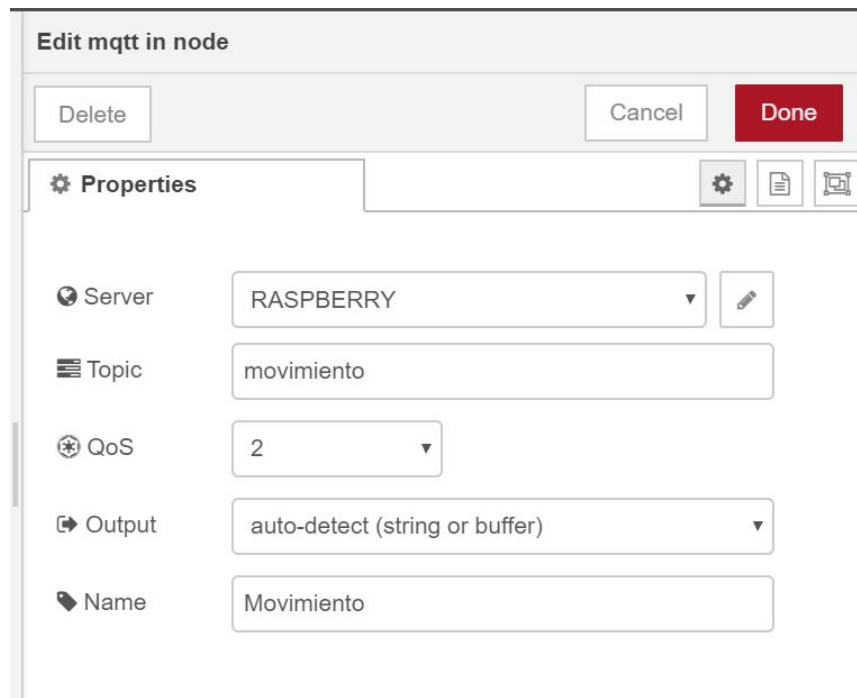


Fig 22. Configuración del nodo mqtt de la subsección movimiento con tópico movimiento.

El pseudocódigo de las funciones es el mismo solo que se reemplazan los textos o indicaciones en la salida:

```
if (msg.payload == 1) {  
  msg.payload= "Se detecto movimiento";  
}else{  
  msg.payload="No hay movimiento";  
}  
return msg;
```

Por último, la respuesta mqtt se realiza de igual manera que en el apartado de la humedad solo que en esta oportunidad se realiza, si la alarma está encendida y si hubo movimiento:

```
var alarma = global.get("alarma");  
if ((msg.payload == "Se detecto  
movimiento")&&(alarma==1)) {  
  msg.payload= 1  
} else {
```

```

    msg.payload=0
  }
  return msg;

```

Subsección Temperatura

La subsección Temperatura, también es muy similar a las dos anteriores solo que en esta oportunidad se le añade un gráfico de gauss para que el usuario pueda ver el valor de temperatura actual, y además de acuerdo a un valor de tolerancia máxima de temperatura (ingresado por el usuario previamente), se le informa al usuario si hay o no peligro de incendio y también envía la orden a la raspberry vía mqtt que encienda un led indicador en caso de alerta de incendio. (Fig. 23)

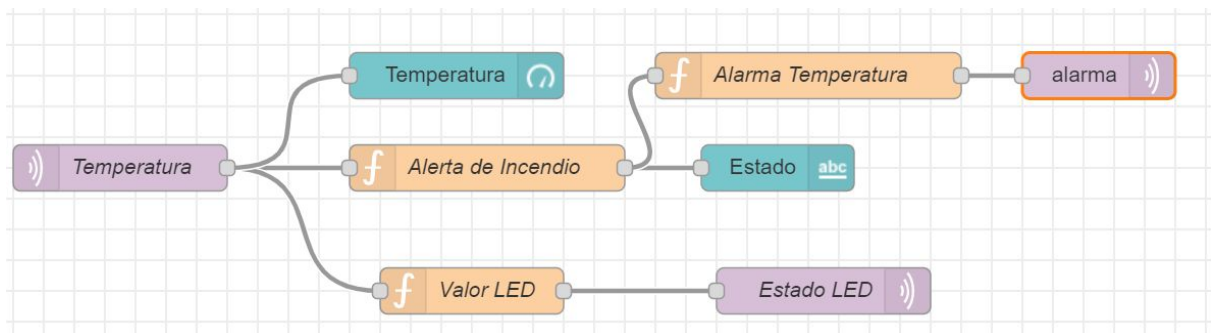


Fig 23. Nodos del DashBoard de la subsección de temperatura

También se le permite al usuario ingresar el valor máximo de tolerancia de temperatura a través de un “text input” y le informa el valor ingresado a través de un “text output” (ver Fig 24), para esto se utilizaron dos nodos correspondientes a la librería del dashboard importada previamente,

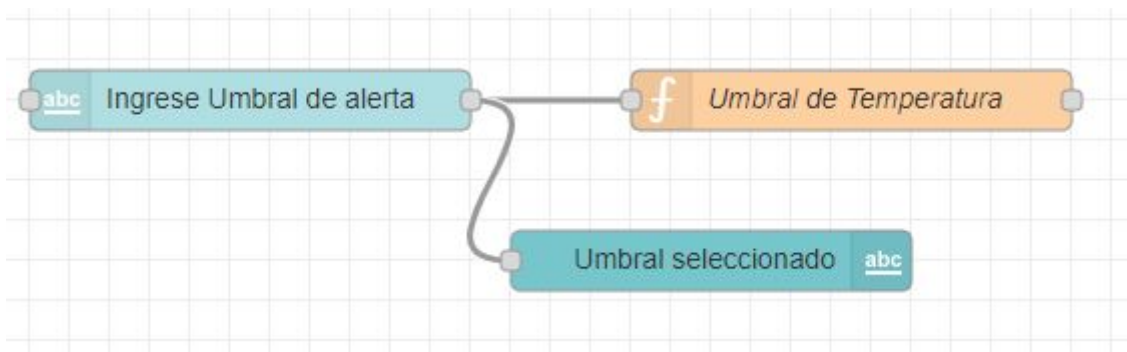


Fig 24. Nodos de Control de umbral para sensado de temperatura.

A continuación se muestran los códigos de las funciones implementadas para la resolución de este apartado:

- La función de “Umbral de Temperatura”, setea el valor de una variable global con la salida del text input:

```
global.set("umbral", msg.payload);
```

- La función “Alerta de Incendio”, que setea el valor de una variable global comparando con el valor de temperatura ingresado por el usuario:

```
var umbral = global.get("umbral");  
if (msg.payload >= umbral) {  
  msg.payload= "ALERTA DE INCENDIO";  
}else{  
  msg.payload="Sin riesgo de  
incendio";  
}  
return msg;
```

- La función “Alarma Temperatura”, que setea el valor de la salida en “1” si hay alerta de incendio y si está activada la alarma, ‘0’ cc:

```
var alarma = global.get("alarma");  
if ((msg.payload == "ALERTA DE  
INCENDIO")&&(alarma==1)) {  
  msg.payload= 1  
} else {  
  msg.payload=0  
}  
return msg;
```

- La función “Valor LED”, que setea el valor de la salida en “1” si hay alerta de incendio y si está activada la alarma; ‘0’ cc:

```
var umbral = global.get("umbral");  
if (msg.payload >= umbral) {  
  msg.payload= 1;  
}else{  
  msg.payload=0;
```

```
}  
return msg;
```

Resultados

A continuación en la figura 25 se muestra el monitor de datos de la alarma en funcionamiento.

En el recuadro rojo se puede observar el switch para activar o desactivar la alarma, y su estado.

En el recuadro azul se observa el estado del sensor de humedad.

En el recuadro naranja se ve estado del sensor de movimiento.

En el recuadro verde se puede ver la temperatura, si hay riesgo de incendio y la opción de configurar el umbral de la temperatura.

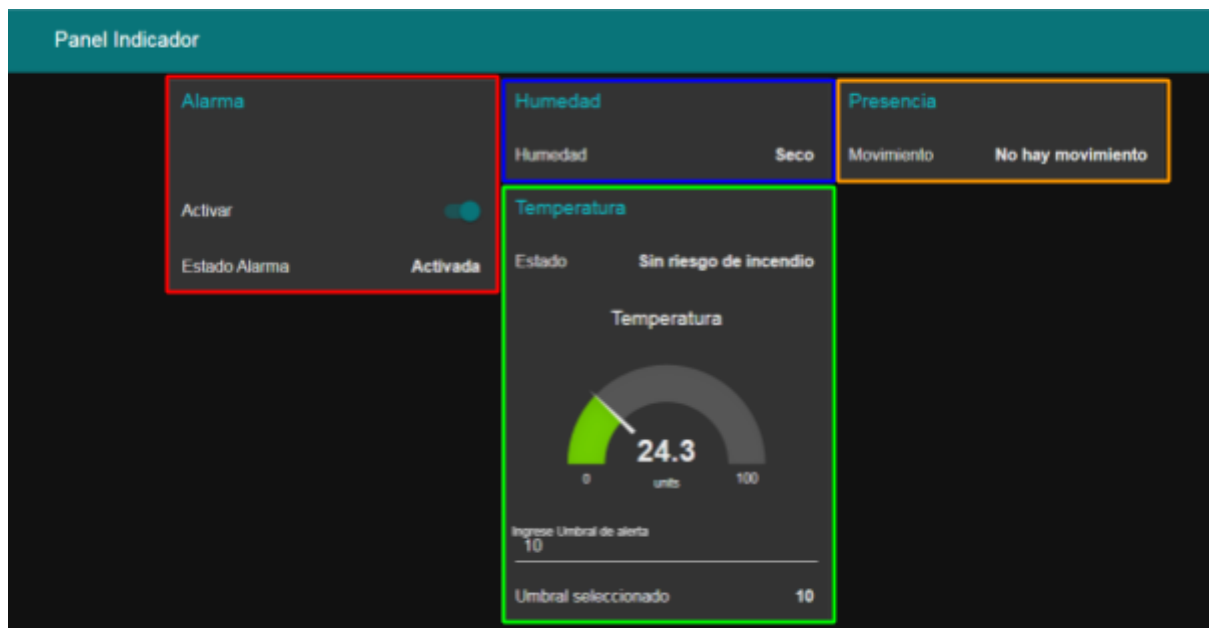


Fig 25. Monitor de datos de alarma.

En la figura 26 se puede ver la placa Raspberry Pi en funcionamiento con todos los sensores y actuadores conectados.

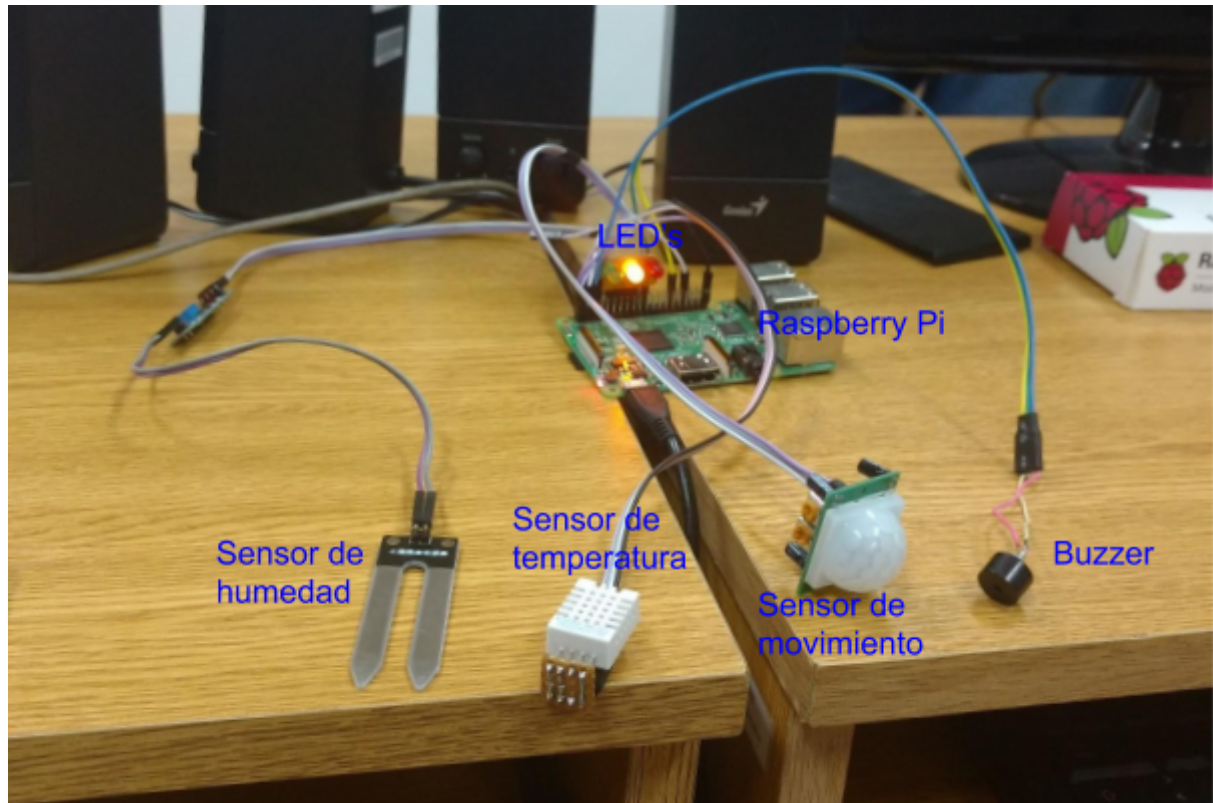


Fig 26. sistema Completo con los respectivos Sensores y Actuadores