DOMÒTICA APLICADA A UN LOCAL

OSCAR SOLANS CABALLER

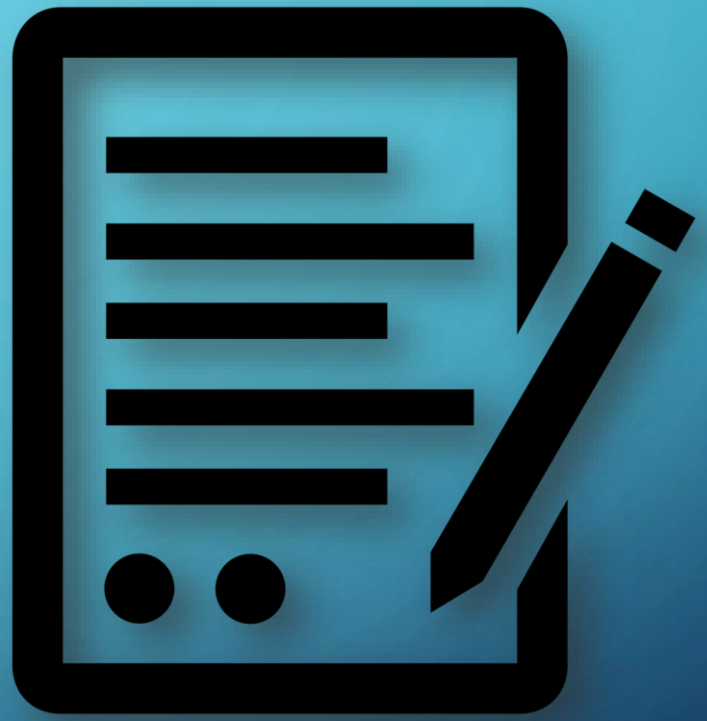
GRAU DE TECNOLOGIES DE LA TELECOMUNICACIÓ

ARDUINO

GENER 2019

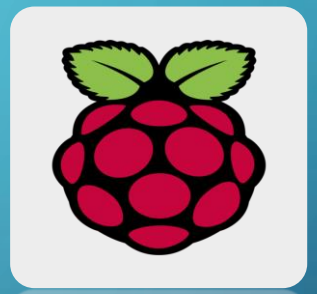
ÍNDEX DE CONTINGUTS

- Introducció
- Objectius
- Disseny del sistema
  - Funcionament bàsic
  - Tecnologies utilitzades
- Implementació
- Exemple d'ús del sistema (vídeo)
- Conclusions



INTRODUCCIÓ

- Sistema domòtic per a un local comercial.
- Interacció de manera telemàtica.
- Visualització telemàtica del sistema.
- Creació d'una maqueta de proves.

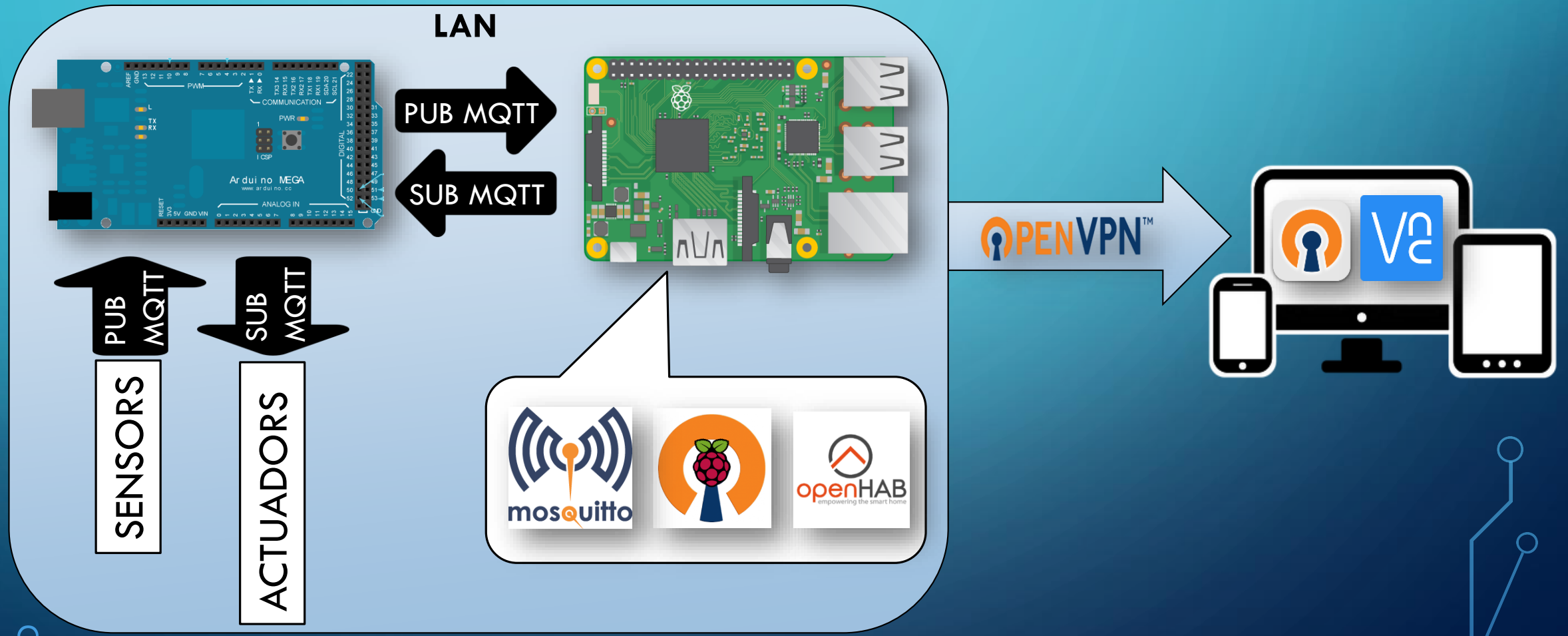


OBJECTIUS

- Comunicar els nodes amb protocol MQTT.
- Interactuar amb el sistema mitjançant OpenHAB.
- Creació d'una xarxa VPN per accedir al sistema.
- Ús d'interrupcions per alliberar Arduino de processos i per estalvi d'energia.
- Gestió de les diferents alarmes i la seva notificació.
- Control remot dels actuadors.

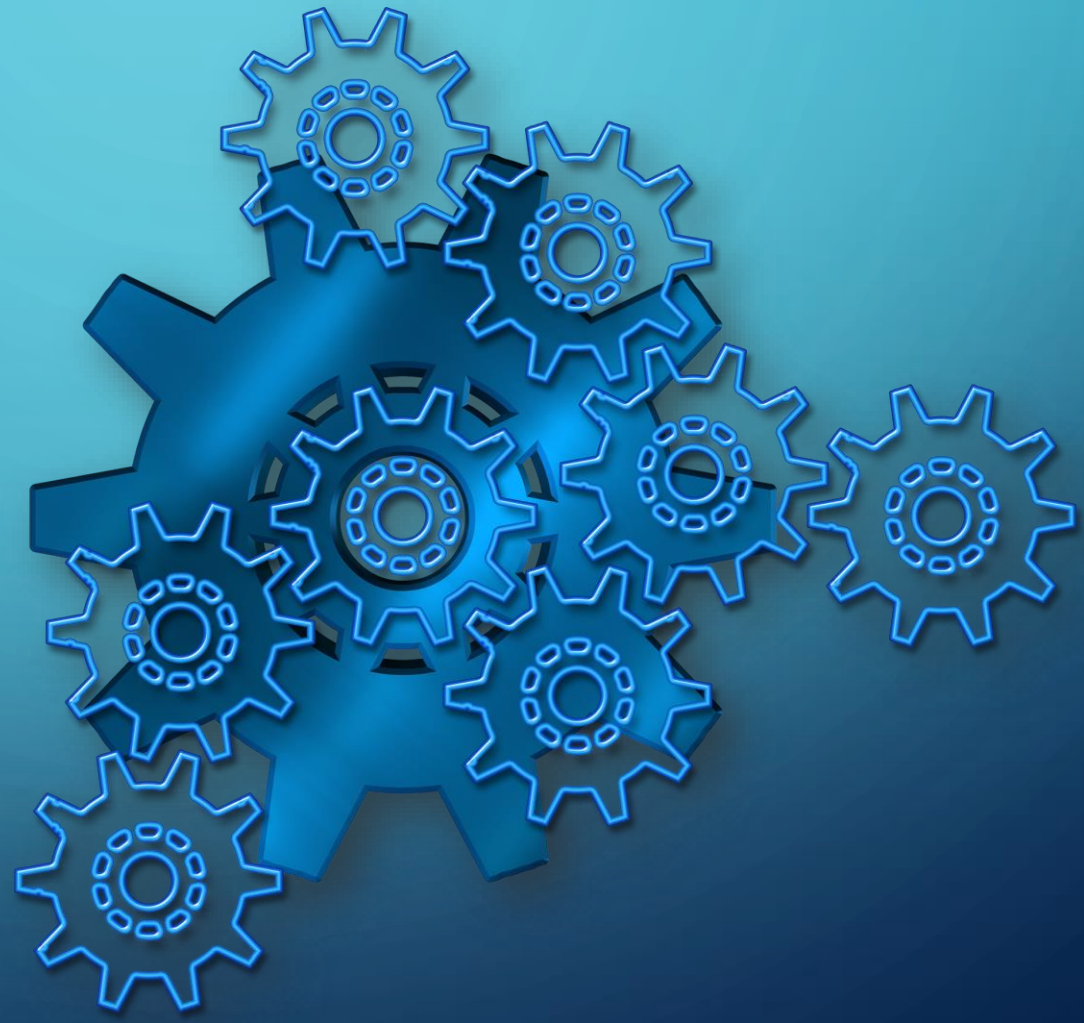
DISSENY DEL SISTEMA

FUNCIONAMENT BÀSIC



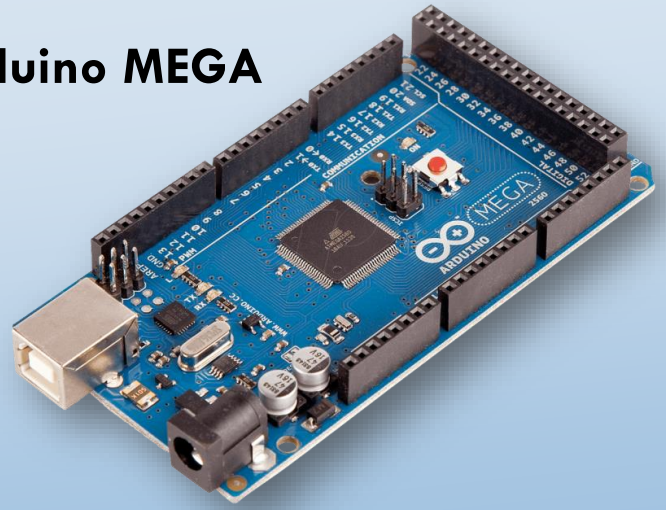
TECNOLOGIES UTILITZADES

- Arduino
- Sensors i actuadors
- Raspberry Pi
- MQTT
- OpenHAB

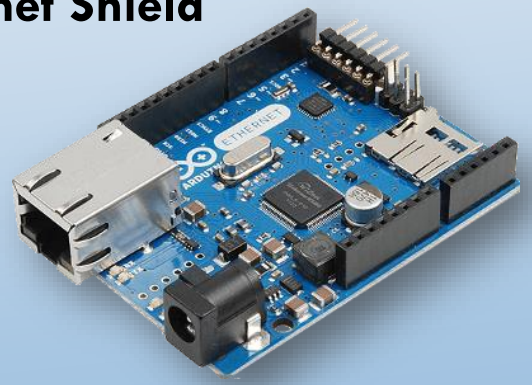


- Arduino

Arduino MEGA



Ethernet Shield



Controlador encarregat de rebre la informació de cadascun dels sensors i enviar ordres als actuadors.

Transmet la informació sobre la xarxa LAN a través de l'Ethernet Shield.

Executa tant el codi propi com les ordres rebudes d'OpenHAB.

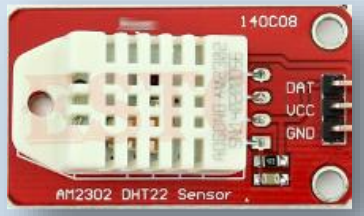
Client MQTT.

• Sensors i actuadors

SENSORS

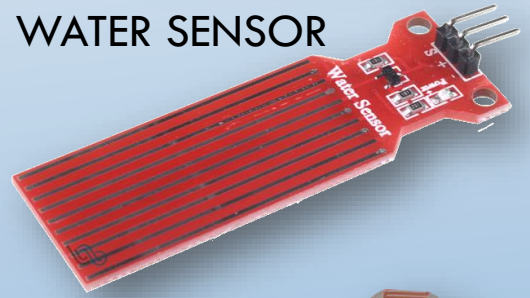
HC-SR04

DHT22



PIR

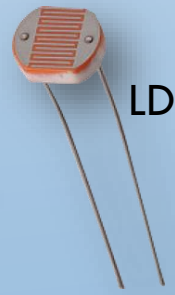
WATER SENSOR



ACS712

MQ7

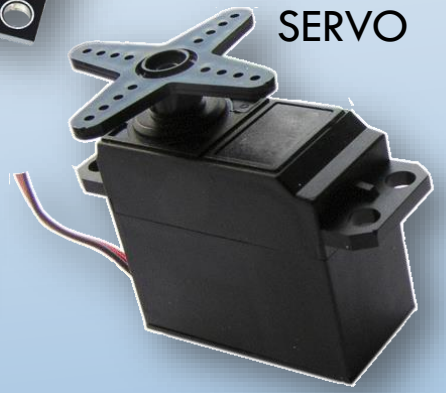
LDR55



ACTUADORS

RELÉ

SERVO



LED

BUZZER

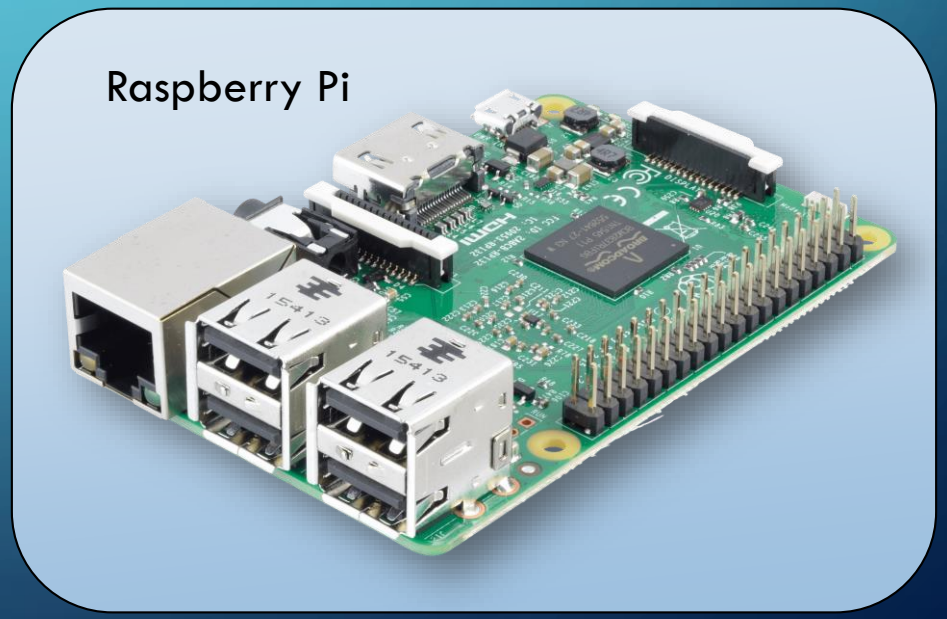
- Raspberry Pi

  Node central del sistema.

  Sistema sobre el qual s'executaran OpenHAB, PiVPN i CameraSuite.

  Encarregat d'enviar ordres d'actuació a Arduino.

  MQTT: Broker Mosquitto.



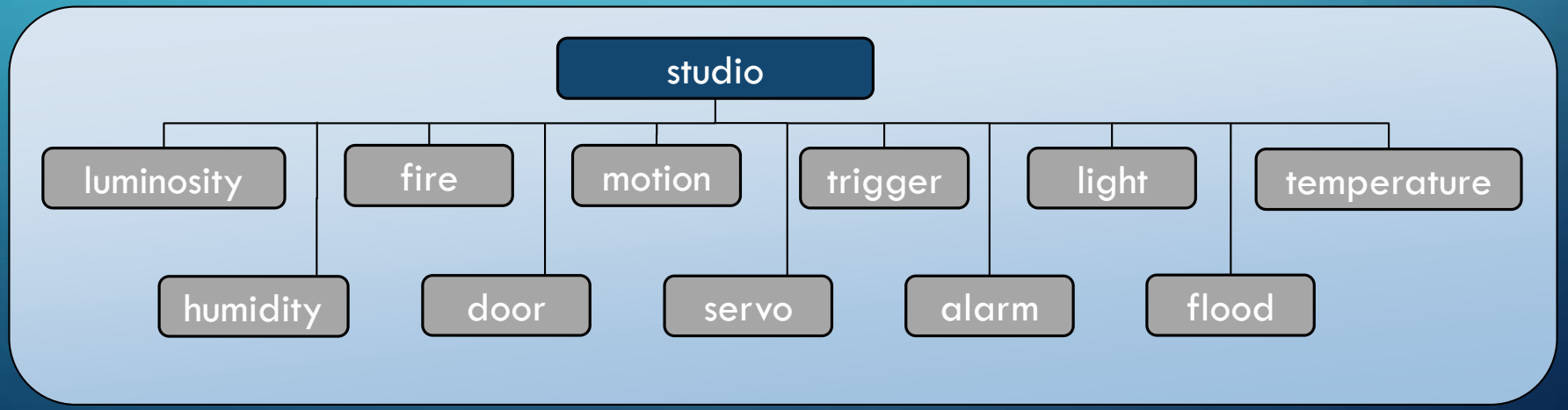
• MQTT

Llenguatge M2M que consumeix poc ample de banda.

Sistema de comunicació basat en TOPICS (i. ex: studio/fire).

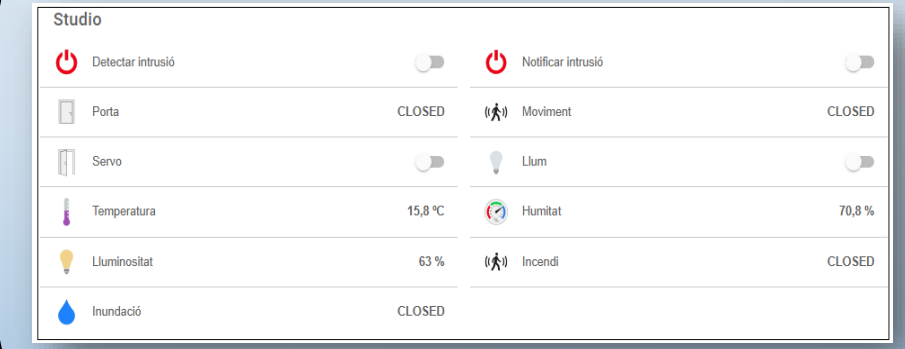
Funcions PUBLISH i SUBSCRIBE per enviar o rebre un missatge.

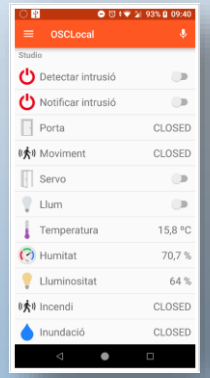
Topologia d'estrella.



•OpenHAB

BasicUI





APP

HABPanel

Software desenvolupat en Java (multiplataforma).

Gran escalabilitat: adaptació a tot tipus de necessitats.

Representa els elements físics en ITEMS dins un entorn virtual anomenat SITEMAP.

Permet automatitzar accions segons regles configurades.

Proporciona diferents interfícies per interactuar amb el sistema.

IMPLEMENTACIÓ

• Arduino (I)

Inici de l'sketch

```
#include <Ethernet.h>
#include <PubSubClient.h>
#include <DHT.h>
#include <Servo.h>

byte mac[] = { 0xDE, 0xED, 0xBF, 0xAF, 0xFF, 0xED };
IPAddress ip(192, 168, 1, 230);
IPAddress subnet(255, 255, 255, 0);
IPAddress gateway(192, 168, 1, 1);

IPAddress mqtt_server(192,168,1,10);

EthernetClient ethClient;
PubSubClient client(ethClient);

#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
const int LDRPin = A15;
const int DHTPin = 49;
const int PIRPin= 33;
const int ECHOPin = 43;
const int TRIGGERPin = 42;
const int BUZZPin = 39;
const int MQPin = 21;
const int LEDPin = 31;
const int WATERPin = 20;
const int RELAYPin = 41;
const int ACS712Pin = A11;
const int SERVOPin = 37;
DHT dht(DHTPin, DHTTYPE);
Servo servoPorta;

unsigned long prevMillis1 = 0;
unsigned long prevMillis2 = 0;
unsigned long time_now_1 = 0;
unsigned long time_now_2 = 0;
float lum = 0;
char lumStr[3];
char humStr[3];
char tempStr[3];
int pirState = 2;
int trigger;
int hcsrState = 2;
volatile int mqState = 2;
int servoState = 2;
volatile int waterState =2;
int pos = 85;
int lightState;
int relayState;
int sensibility = 185; // use 185 for 5A Module
double voltage = 0;
double vRMS = 0;
double ampRMS = 0;
long duration;
long distanceCm;
char waterStr;
```

• Arduino (II)

Def. funcions

```
void reconnect(){
    ...}
void lightRead(){
    ...}
void dhtRead(){
    ...}
void pirRead(){
    ...}
void hcsr04Read(){
    ...}
void mqRead(){
    ...}
void waterRead(){
    ...}
void getVPP(){
    ...}
void light(){
    ...}
void callback(){
    ...}
```

Setup

```
void setup() {
  Serial.begin(9600);
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
  servoPorta.attach(SERVOPin);
  pinMode(LDRPin, INPUT);
  pinMode(DHTPin, INPUT);
  pinMode(PIRPin, INPUT);
  pinMode(ACS712Pin, INPUT);
  pinMode(TRIGGERPin, OUTPUT);
  pinMode(LEDPin, OUTPUT);
  pinMode(ECHOPin, INPUT);
  pinMode(BUZZPin, OUTPUT);
  pinMode(RELAYPin, OUTPUT);
  pinMode(MQPin, INPUT_PULLUP);
  pinMode(WATERPin, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(WATERPin), waterRead, CHANGE);
  attachInterrupt(digitalPinToInterrupt(MQPin), mqRead, CHANGE);
}
```

Loop

```
void loop() {
  if (!client.connected()) {
    Serial.print("Connectant ...\n");
    Ethernet.begin(mac, ip, gateway, subnet);
    reconnect();
    client.subscribe("studio/#");
  }else{
      unsigned long time_now_1 = millis();
      if ((unsigned long)(time_now_1 - prevMillis1) >= 30000){
        lightRead();
        dhtRead();
        prevMillis1 = millis();
        }
      light();
      if(trigger == 1){
         pirRead();
         hcsr04Read();
        }
    client.loop();
  }
  delay(100);
}
```

OpenHAB (I)

Sitemap

```
sitemap local label="OSCLocal" {
    Frame label="Studio" {

        Switch item=MotionTrig label="Detectar intrusió"

        Switch item=Alarm label="Notificar intrusió"

        Default item=Door label="Porta"

        Default item=Motion label="Moviment"

        Switch item=Servo label="Servo"

        Switch item=Light label="Llum"

        Default item=Temperature label="Temperatura"

        Default item=Humidity label="Humitat"

        Default item=Luminosity label="Lluminositat"

        Default item=Fire label="Incendi"

        Default item=Flood label="Inundació"


    }
}
```

Alarms

```
rule "Moviment detectat"
when
    Item Motion received update
then
    if ((Motion.state == OPEN) && (Alarm.state == ON)){
    sendNotification("[redacted]@gmail.com", "MOVIMENT DETECTAT!!!");
    }

end
```

```
rule "Inundació"
when
    Item Flood received update
then
    if (Flood.state == OPEN){
    sendNotification("[redacted]@gmail.com", "INUNDACIÓ!!!");
    }

end
```

```
rule "Porta oberta"
when
    Item Door received update
then
    if ((Door.state == OPEN) && (Alarm.state == ON)){
    sendNotification("[redacted]@gmail.com", "PORTA OBERTA!!!");
    }

end
```

```
rule "Detecció d'incendi"
when
    Item Fire received update
then
    if (Fire.state == OPEN){
    sendNotification("[redacted]@gmail.com", "INCENDI DETECTAT!!!");
    }

end
```

• OpenHAB (II)

Items

```
Group     Home                "OSCLocal"              <house>

Group     Studio              "Estudio"               <pantry>        (Home)


Switch    Light         "Luz"                   <light>         (Studio, gLight)         ["Lighting"]     {mqtt=">[mqtt:studio/light:command:ON:1],>[mqtt:s
Contact   Door          "Porta [%s]"               <door>          (Studio, gDoor)                           {mqtt="<[mqtt:studio/door:state:default]"}
Contact   Motion        "Sensor de moviment[%s]"   <motion>        (Studio, gMotion)                        {mqtt="<[mqtt:studio/movement:state:default]"}
Switch    MotionTrig        "Trigger"                  <switch>        (Studio, gMotion)         ["Switchable"]   {mqtt=">[mqtt:studio/trigger:command:ON:1
Number    Temperature   "Temperatura [%.1f ºC]"           <temperature>   (Studio, gTemperature)                    {mqtt="<[mqtt:studio/temperature:state:
Number    Humidity      "Humitat [%.1f %%]"              <humidity>      (Studio, gHumidity)                      {mqtt="<[mqtt:studio/humidity:state:def
Number    Luminosity      "Lluminositat [%.0f %%]"             <light>      (Studio, gLuminosity)                        {mqtt="<[mqtt:studio/luminosity:s
Contact   Fire        "Sensor de foc[%s]"   <motion>        (Studio, gMotion)                          {mqtt="<[mqtt:studio/fire:state:default]"}
Switch    Alarm     "Alarma"    (gAlarm) {mqtt=">[mqtt:studio/alarm:command:ON:1],>[mqtt:studio/alarm:command:OFF:0],<[mqtt:studio/alarm:state:ON:1],<[mqtt:
Contact   Flood       "Sensor d'aigua[MAP(flood.map):%s]"   <water>        (Studio, gWater)                        {mqtt="<[mqtt:studio/flood:state:defaul
Switch    Servo         "Servo Porta"               <door>   (Studio, gMotion)         ["Switchable"]   {mqtt=">[mqtt:studio/servo:command:ON:1],>[mqtt:stu

Group:Switch:OR(ON, OFF)         gLight         "Luz"                  <light>         (Home)
Group:Contact:OR(OPEN, CLOSED)   gMotion          "Porta"                <door>          (Home)
Group:Contact:OR(OPEN, CLOSED)   gMotion        "Sensor de moviment"   <motion>        (Home)
Group:Switch:OR(ON, OFF)         gMotion         "Trigger"              <motion>        (Home)
Group:Number:AVG                 gTemperature   "Temperatura"          <temperature>   (Home)
Group:Number:AVG                 gHumidity      "Humitat"              <humidity>      (Home)
Group:Contact:OR(OPEN, CLOSED)   gMotion        "Sensor de foc"        <motion>        (Home)
Group:Switch:OR(ON, OFF)     gMotion            "Alarma"               <motion>        (Home)
Group:Contact:OR(OPEN, CLOSED)   gMotion          "Servo Porta"                <door>          (Home)
```

EXEMPLE D'ÚS DEL SISTEMA (VÍDEO)

CONCLUSIONS

CONCLUSIONS

- Els components triats permeten elaborar un sistema robust i fiable.
- S'han assolit els terminis previstos en l'etapa de planificació.
- El sistema ofereix una gran escalabilitat i per tant, una gran capacitat d'adaptació a nous entorns.
- La velocitat de comunicació que permet MQTT degut al seu baix consum d'ample de banda atorga al sistema un temps de resposta molt baix.

MOLTES GRÀCIES