Redes II - Práctica 3
IoT

Carlos Ruiz Pastor

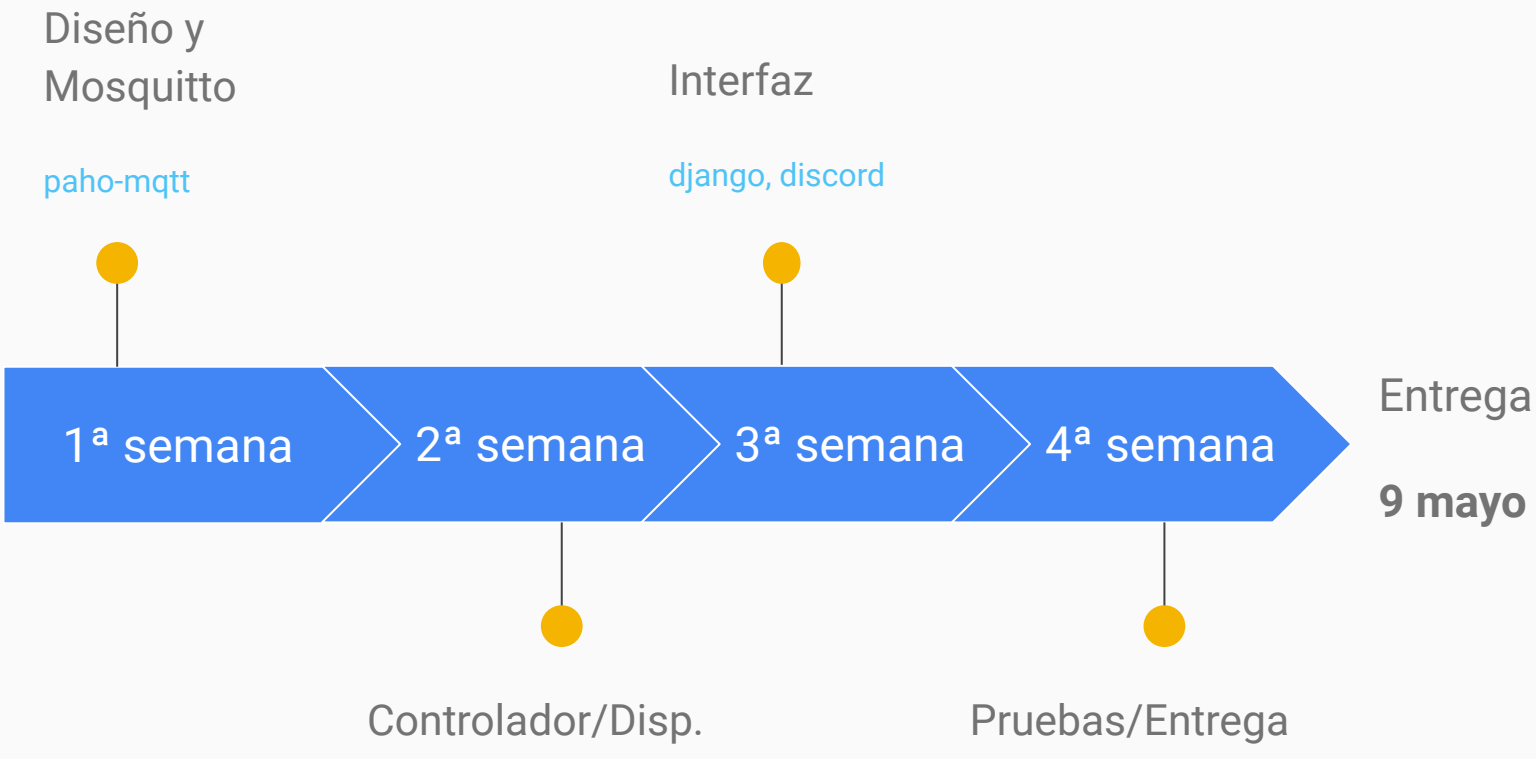
Despacho: B-402
Correo: carlos.ruizp@uam.es

Objetivo

Implementar un sistema domótico para el hogar con dispositivos IoT.

Conocimientos Necesarios

- Control de versiones (Git)
- **Python**
- Broker de mensajes (mqtt)
- Interfaz de usuario
  - Django
  - Discord
- Diseño de código


Diseño y
Mosquitto
paho-mqtt
Interfaz
django, discord
1ª semana
2ª semana
3ª semana
4ª semana
Entrega
9 mayo
Controlador/Disp.
paho-mqtt, argparse,
threading, unittest
Pruebas/Entrega

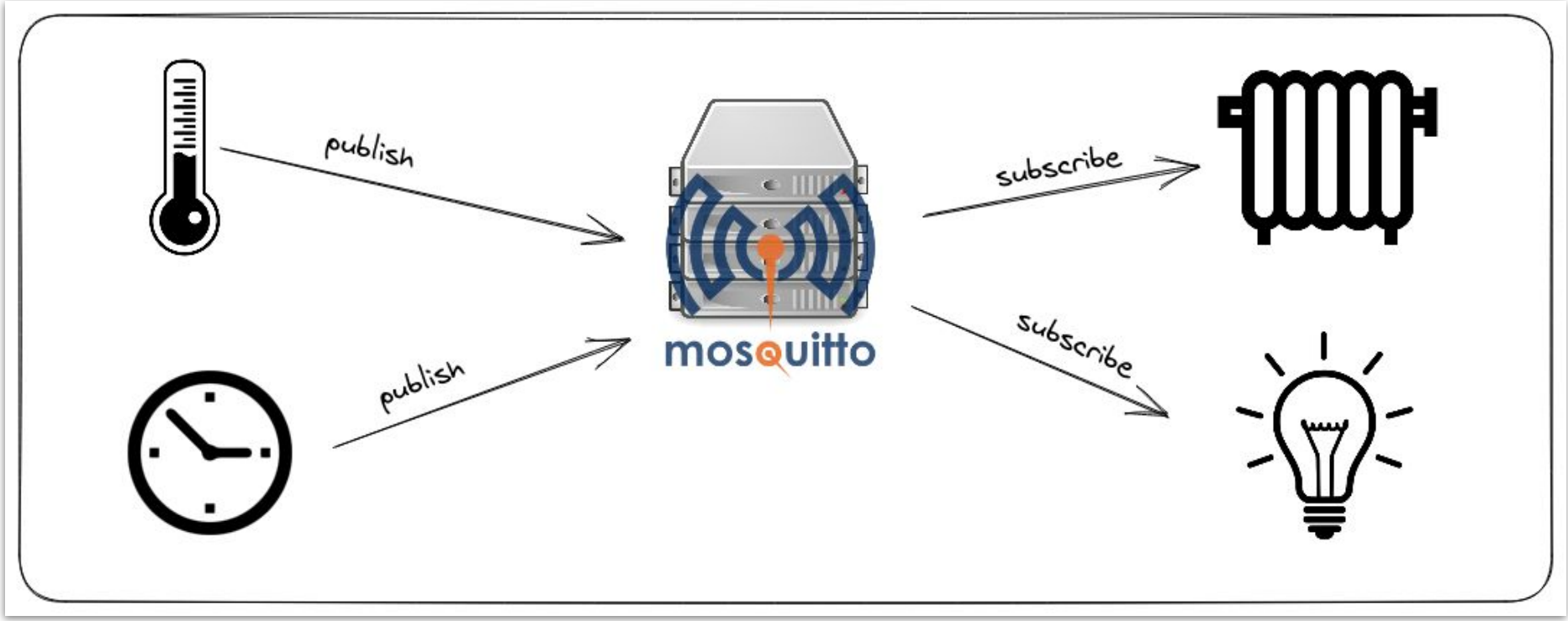
MQTT

¿Qué es MQTT?

MQTT es un protocolo de mensajería basado en el modelo publicador/suscriptor.

Los dispositivos IoT utilizan MQTT para la transmisión de datos, ya que resulta fácil de implementar y puede comunicar datos de manera eficiente.

Esquema de modelo tradicional (HTTP)



Esquema de modelo publicador/suscriptor (MQTT)

HTTP vs MQTT

En HTTP los dispositivos se comunican directamente entre ellos. Con el modelo publicador/suscriptor de MQTT conseguimos:

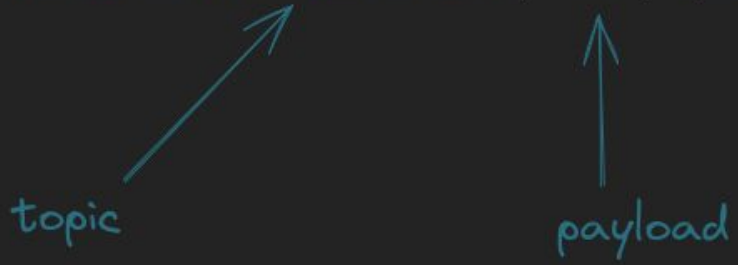
- Desacoplamiento espacial
- Desacoplamiento temporal
- Desacoplamiento de sincronización

Publicar

```
import paho.mqtt.client as mqtt
broker_address="sede.ii.uam.es"
# broker_address="localhost" # use local broker
client = mqtt.Client("P1") #create new instance
client.connect(broker_address) #connect to broker
client.publish("redes2/GGGG/PP/switches/bulb1", "OFF") #publish
```

Publicar

```python
import paho.mqtt.client as mqtt
broker_address="sede.ii.uam.es"
# broker_address="localhost" # use local broker
client = mqtt.Client("P1") #create new instance
client.connect(broker_address) #connect to broker
client.publish("redes2/GGGG/PP/switches/bulb1", "OFF") #publish
```

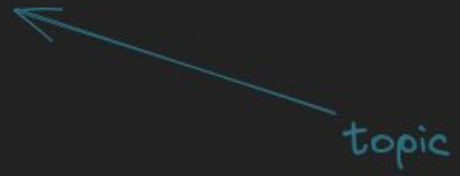


Suscribirse

```
import paho.mqtt.client as mqtt

def on_message(client, userdata, message):
   print("message received ", str(message.payload.decode("utf-8")))
   print("message topic=", message.topic)

broker_address="redes2.ii.uam.es"
client = mqtt.Client("bulb_switch") #create new instance
client.on_message=on_message #attach function to callback
client.connect(broker_address) #connect to broker
client.loop_start() #start the loop
client.subscribe("redes2/GGGG/PP/switches/bulb1")
```

Suscribirse

```python
import paho.mqtt.client as mqtt

def on_message(client, userdata, message):
    print("message received ", str(message.payload.decode("utf-8")))
    print("message topic=", message.topic)

broker_address="redes2.ii.uam.es"
client = mqtt.Client("bulb_switch") #create new instance
client.on_message=on_message #attach function to callback
client.connect(broker_address) #connect to broker
client.loop_start() #start the loop
client.subscribe("redes2/GGGG/PP/switches/bulb1")
```



Semana 1

- Mirar tutorial de MQTT
- ¿Qué clases hay que hacer para el controlador y dispositivos?
  - Ambos son clientes mqtt
- ¿Cómo se comunican? Pensar en topics, payloads
- Implementar sistema sin interfaz con dummy_sensor, dummy_switch
  - Test unitarios para cada clase
  - Script que instancie un sensor, un switch y el controlador

```
$ sudo docker run --rm -p 1883:1883 -p 9002:9002 eclipse-mosquitto
```

Recursos

- Librería paho-mqtt, para utilizar MQTT en Python
- Librería uuid para generar IDs
- Librería argparse, para gestionar los parámetros de la línea de comandos
- Recomiendo **IceCream — Never use print() to debug again**

Páginas recomendadas:

- Steve's Internet Guide
- Python Discord Bot Tutorial – Code a Discord Bot And Host it for Free
- Getting started with Django