FIAP

MBA+

APICON HANDS-on LAB
RASPBERRY PI IoT

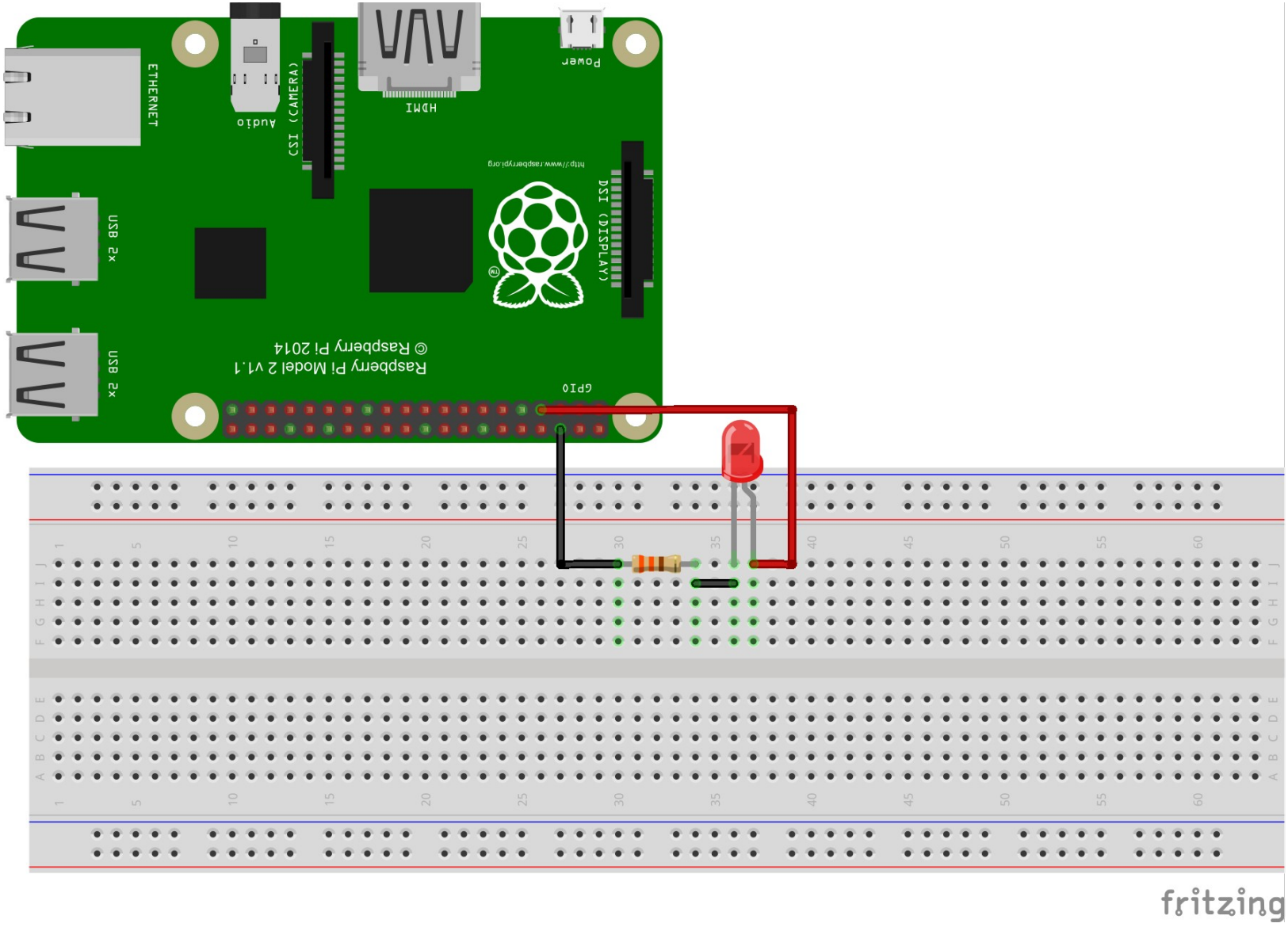
HandsOn Lab

E/S com a Raspberri Pi

Objetivos

1. Primer contato com a Raspberry Pi
   – Configuração inicial da placa
2. **Entrada/Saída**
   – **criação de um circuito**
   – **ativação de um LED (atuador)**

Raspberry – E/S



Raspberry – E/S

FIAP

Rpi.GPIO

- Controla o GPIO

```
1  import RPi.GPIO as GPIO
2  import time
3
4  GPIO.setmode(GPIO.BCM)
5  GPIO.setup(4, GPIO.OUT)
6
7  while(True):
8      print("Led aceso")
9      GPIO.output(4, 1)
10     time.sleep(2)
11     print("Led desligado")
12     GPIO.output(4, 0)
13     time.sleep(2)
```

https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

Protocolos IoT

Características

- Baixo ***overhead***
- Baixo **consumo**
- **Autenticação**
- **Autorização**
- **Criptografia**
- **Latência**

CoAP

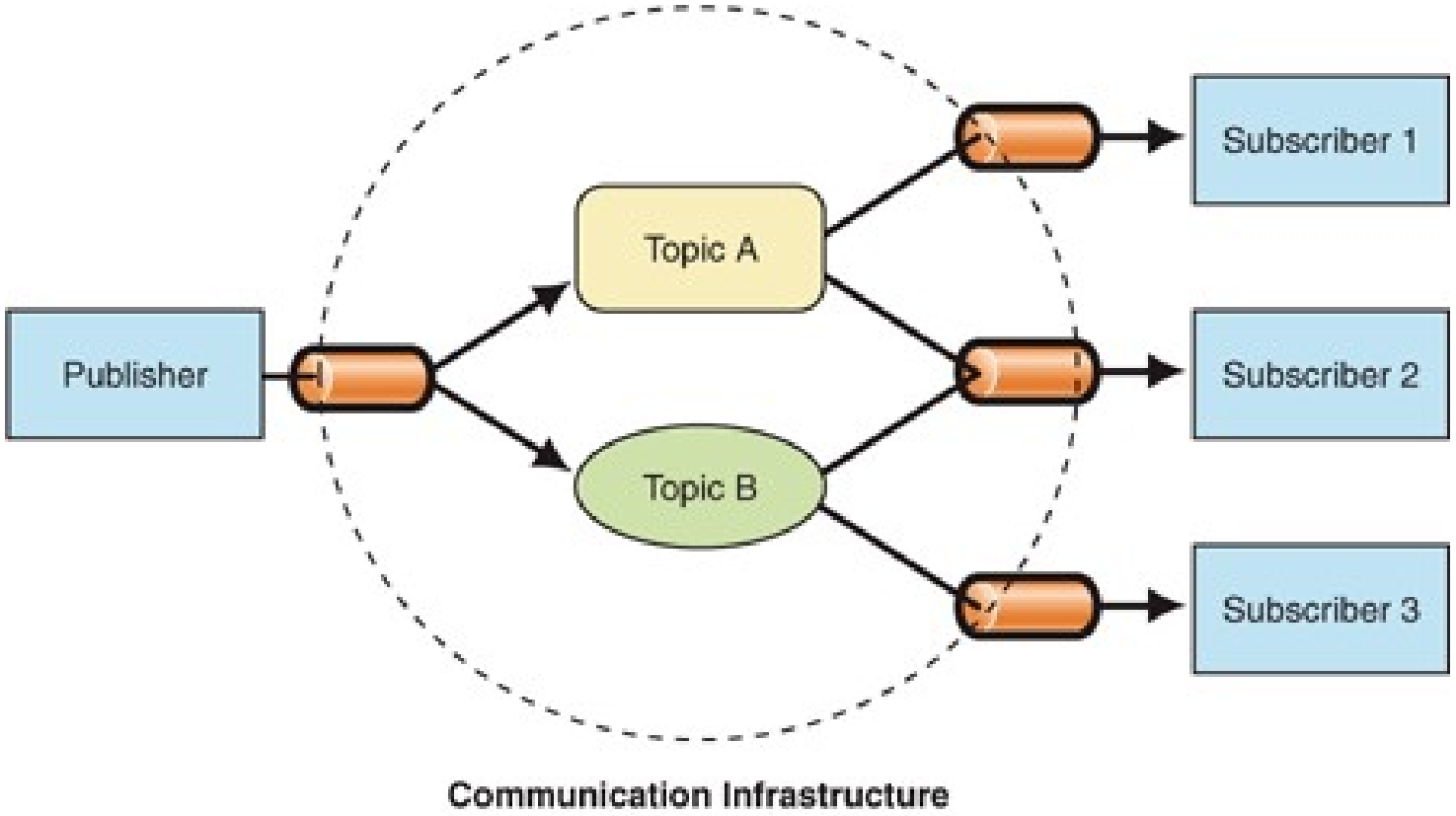
FIAP

Constrained Application Protocol (CoAP)

- Segue o padrão arquitetural **REST** para serviços web
- Aceita vários formatos para o intercâmbio dos dados, como JSON e XML
- Baseado em **UDP**
- Suporta encriptação dos dados
- Foi desenhado para ser **escalável**

Paradigma *publish-subscribe*

Padrão arquitetural usado para a comunicação assíncrona de vários processos



MQTT

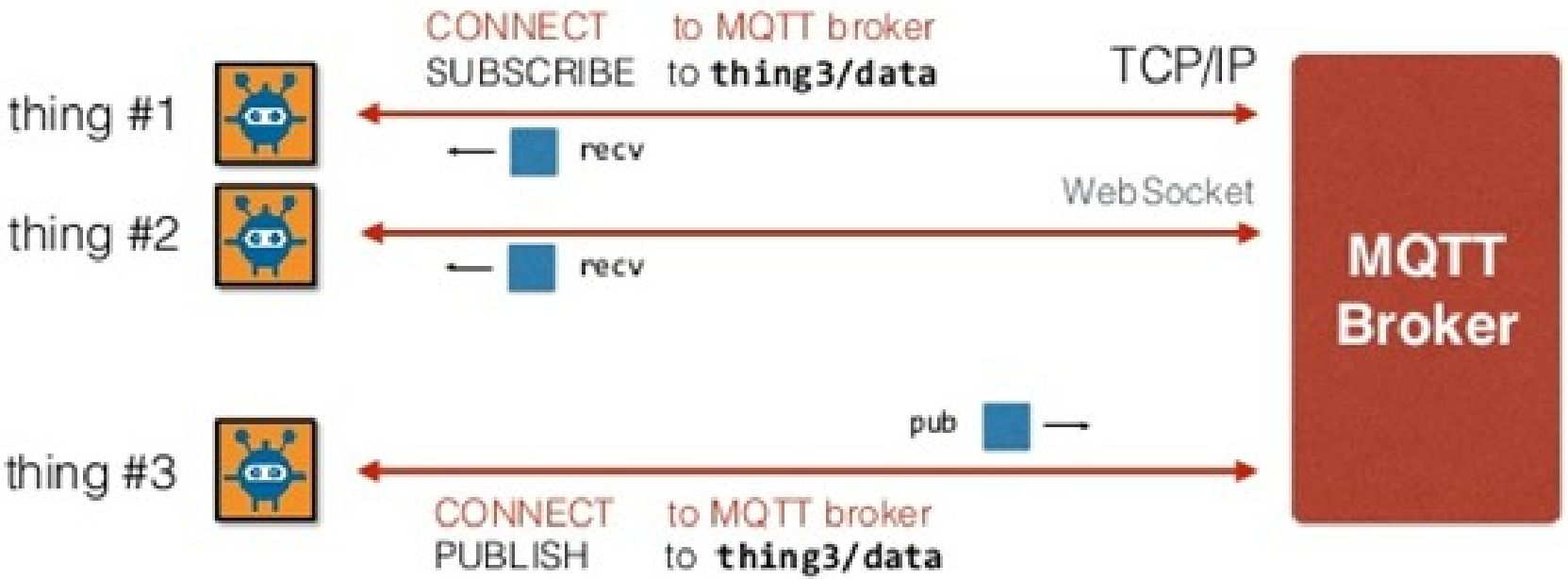
***Message Queue Telemetry Transport* (MQTT)**

- Protocolo leve de aplicação que utiliza **TCP** (porta *1883)* como via de comunicação de dados
- É baseado na publicação de dados em um ***broker*** a partir de um ***publisher***, e na leitura dos dados a partir da inscrição num canal (***subscriber***)
  - Desta forma todas as atualizações serão alertadas ao cliente
- Clientes MQTT podem funcionar, simultaneamente como *publisher* e *subscriber*

FIAP

MQTT

bi-directional, async "push" communication



MQTT

FIAP

Biblioteca

- Eclipse **Paho** project
- Disponível para: C, C++, Java, JavaScript, Python, GO, C#
- Python: *http://www.eclipse.org/paho/clients/python/*
- Java: *https://eclipse.org/paho/clients/java/*

MQTT – *subscribe*


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

# The callback for when the client receives a CONNACK response from the server.
def on_connect(client, userdata, rc):
    print("Connected with result code "+str(rc))
    # Subscribing in on_connect() means that if we lose the connection and
    # reconnect then subscriptions will be renewed.
    client.subscribe("/topic")

# The callback for when a PUBLISH message is received from the server.
def on_message(client, userdata, msg):
    print(msg.topic+" "+str(msg.payload))

client = mqtt.Client("raspberryN", clean_session=True, userdata=None,
                     protocol=MQTTv311, transport="tcp")
client.on_connect = on_connect
client.on_message = on_message
client.username_pq_set("user", "password")

client.connect("iot.eclipse.org", 1883, 60)

# Blocking call that processes network traffic, dispatches callbacks and
# handles reconnecting.
# Other loop*() functions are available that give a threaded interface and a
# manual interface.
client.loop_forever()
```

MQTT – *publish*


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

# Define the payload to be send
payload = "That is just an example message: JSON {'key':'attribute'}"

# Define the topic to publish messages
topic = "/topic"

def send_message(paylaod,topic):
    client.publish(topic, payload, qos=1, retain=False)
    print "Data succesfully published"

# The callback for when the client receives a CONNACK response from the server.
def on_connect(client, userdata, flags, rc):
    print("Connected with result code "+str(rc))

def on_message(client, userdata, msg):
    print "msg arrived"

client = mqtt.Client()
client.on_connect = on_connect
client.on_message = on_message
client.username_pw_set("user", "password")
client.connect( "iot.eclipse.org", 1883, 60)

send_message(payload, topic)

client.disconnect()
```

OMA LWM2M

FIAP

OMA Lightweight M2M (LWM2M)

- Padrão da *Open Mobile Alliance* para M2M e **gerenciamento de dispositivos** IoT
- Estabelece comunicação entre:
  - Servidor LWM2M
  - Cliente LWM2M, localizado no dispositivo
- É frequentemente usado em conjunto com CoAP

CoAP vs MQTT

FIAP

Padrão	CoAP	MQTT
Modelo de comunicação	*request-response*	*publish/subscribe*
RESTful	✓	✗
Camada de transporte	UDP	TCP
Cabeçalho	2009	2.4 / 5
Mensagens	2011	5
Segurança	DLTS	SSL/TLS
QoS	✓	✓

HandsOn Lab

MQTT na Raspberri Pi

Objetivos

1. Primer contato com a Raspberry Pi
   – Configuração inicial da placa
2. Entrada/Saída
   – criação de um circuito
   – ativação de um LED (atuador)
3. Assinar a placa num **canal MQTT**
   /pad/intel-galileo/temperatura
4. Publicar dados no canal MQTT

MQTT

A **conexão** estabelecida com o *broker* MQTT possui a seguinte estrutura:

protocolo://servidor:1883

- **O servidor** utilizado está acessível pelo seguinte endereço: ***10.0.1.3***

Sem **credenciais** para utilizar o *broker* MQTT

Os **canais** MQTT são:

- /temperatura
- /sku
- /luminosidade

Links

The Things Network (TTN) *https://www.thethingsnetwork.org/*

Eclipse Paho project *http://www.eclipse.org/paho/*

Fritzing *http://fritzing.org/home/*

Autodesk Circuits *https://circuits.io/*

Raspberry GPIO

https://sourceforge.net/p/raspberry-gpio-python/wiki/Home/

Sensor Temperatura Grove

http://www.seeedstudio.com/wiki/Grove_-_Temperature_Sensor

PiCamera

https://www.raspberrypi.org/learning/getting-started-with-picamera/worksheet/

FIAP

MBA+