

UNIVERSIDADE DE BRASÍLIA - BRASIL

DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

ENGENHARIA MECATRÔNICA



---

Implementação de uma "rede de  
sensores" utilizando o protocolo  
MQTT

---

**Aluno:** Rafael Ramos de Matos - 15/0145683

**E-mail:** ramosunb@gmail.com

**Professor:** Marcos F. Caetano

**Disciplina:** Transmissão de Dados

**Turma:** A - 2/2019

Brasília, 1 de Dezembro de 2019

## Resumo

This article presents the result of implementing the MQTT protocol using the python2.7 programming language. The algorithm basically consists of a message aggregator (BROKER) and two clients (PUBLISHER and SUBSCRIBER) that can receive and send messages to BROKER.

# 1 Objetivo

O objetivo deste trabalho é, usando o conteúdo ministrado em sala sobre o funcionamento de redes de computadores e transmissão de dados, implementar um algoritmo que visa a construção de uma arquitetura cliente servidor, com o auxílio protocolo TCP/IP, que forneça toda a infraestrutura do protocolo MQTT.

# 2 Introdução

## 2.1 Contextualização

No mundo, com um crescimento exponencial de dispositivos móveis e embarcados de baixa, surgiu a necessidade de uma conexão que demande um baixo consumo energético. Para atender essa demanda, apareceu o protocolo MQTT que foi inventado e desenvolvido inicialmente pela IBM no final dos anos 90 com o objetivo de uso industrial no ramo de petróleo.

O MQTT é um protocolo de mensagens com baixo custo computacional para sensores e pequenos dispositivos móveis usando, em uma camada mais baixa, de forma otimizada o protocolo da camada de transporte TCP/IP. Esse protocolo de comunicação tem a estrutura básica cliente servidor, porém ele implementa uma ferramenta de desacoplamento de dispositivos chamada de *broker* que faz um gerenciamento de mensagem e de clientes (*publisher* e *subscriber*).

A figura 1, mostrada logo abaixo, apresenta, de forma gráfica, a estrutura básica do protocolo MQTT.

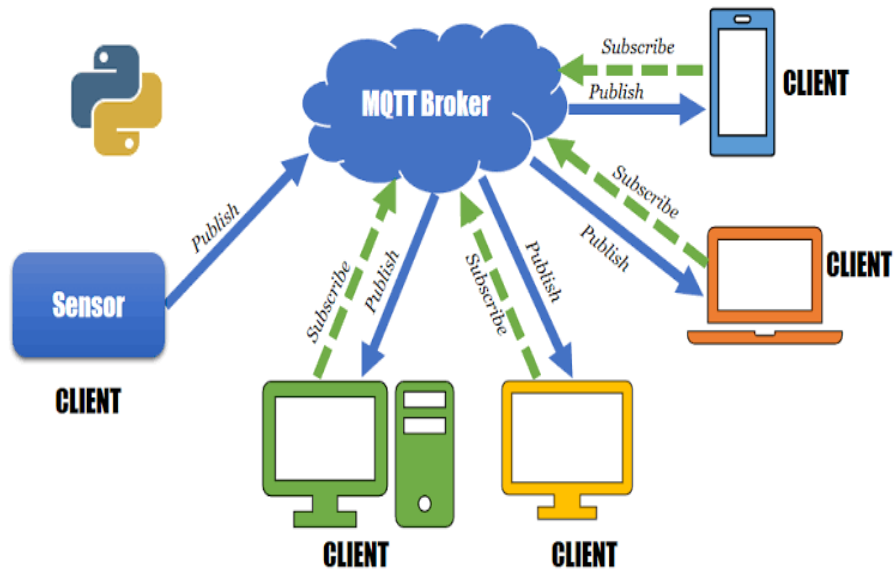


Figura 1: Estrutura do protocolo MQTT

## 2.2 Comparação entre MQTT e HTTP

HTTP é um protocolo que permite a obtenção de recursos, tais como documentos HTML. É a base de qualquer troca de dados na Web e um protocolo cliente-servidor, o que significa que as requisições são iniciadas pelo destinatário, geralmente um navegador da Web. Além disso, vale ressaltar que, o protocolo HTTP permite um fluxo intenso de dados, uma característica que já não se encontra no protocolo MQTT.

O protocolo HTTP possui um mecanismo de gerenciamento de mensagens e de clientes análogo ao *Broker* chamado *Internet*. A figura 2, mostrada logo abaixo, apresenta, de forma gráfica a estrutura básica do protocolo HTTP.

Para fins de comparação, a tabela 1 mostra as características de cada protocolo para a demanda deste relatório.

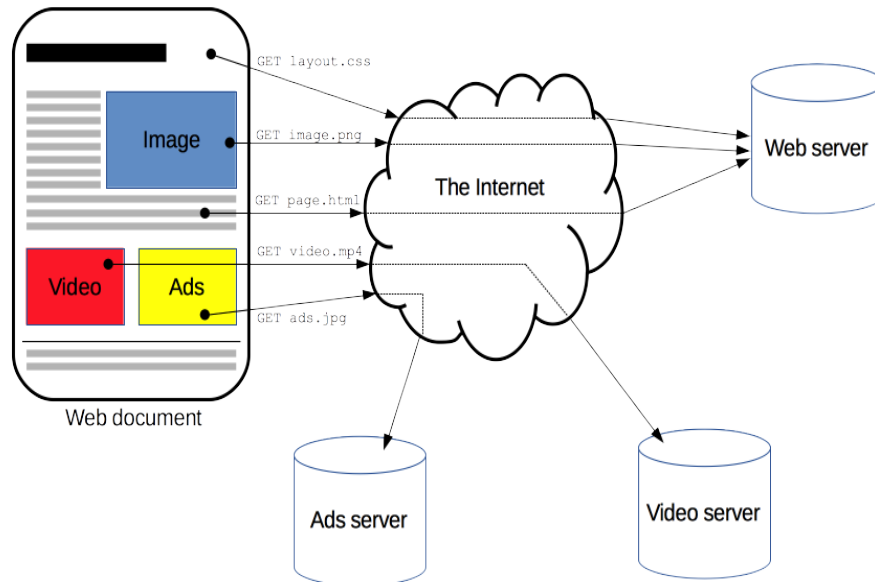


Figura 2: Estrutura do protocolo HTTP

Protocolo	Baixa potência	Fluxo de informação	TCP/IP	Embarcados
MQTT	SIM	ALTO	SIM	SIM
HTTP	NÃO	BAIXO	SIM	NÃO

Tabela 1: tempos das tarefas

## 3 Arquitetura do Sistema

### 3.1 Funcionamento Básico

O funcionamento básico da estrutura da protocolo MQTT é mostrada na figura 1. Para uma funcionalidade básica do protocolo MQTT precisa-se construir 3 entidades, *Broker*, *Publisher* e *Subscriber*.

O *broker* é a entidade que cuida do gerenciamento de todos os clientes (*Subscriber* e *Publisher*) que estão conectados e que queiram se conectar ou desconectar. Além disso, o *broker* é responsável pelo gerenciamento de mensagem durante toda a aplicação, autorizando quem pode ou não receber determinada mensagem e determinando que pode ou não publicar, no seu

banco de dados, uma determinada mensagem.

*Subscriber* e *Publisher* são as entidades periféricas que são gerenciada pelo *Broker*. O *Publisher* é a entidade de onde vem as mensagens, pode ser um sensor de temperatura, pressão, luminosidade, etc. Já o *Subscriber* é a entidade que recebe os dados que pode ou não gerar uma ação baseada no dado recebido.

### 3.2 Explicação da Arquitetura Produzida

A figura 3, mostrada a seguir, expõe o funcionamento da troca de mensagens utilizando o protocolo MQTT com suas entidades básicas. Para fins de exemplificação, a figura 3 trás um *broker* e dois clientes ( um *subscriber* à direita e um *publisher* à esquerda).

Quando o *broker* não está em funcionamento, não existe comunicação entre *Subscriber* e *Publisher*, então, para a comunicação acontecer o *broker* precisa está ativo. Para iniciar a comunicação, os clientes fazem um requisição de conexão com a mensagem *CONNECT*, se não existir nenhum empecilho, como já existir um cliente com o mesmo *client\_id*, o *broker* retorna um *CONNACK*. Depois disso, o cliente faz um requisição se pode receber ou publicar dados no *broker* com a mensagem *SUBCIBE*, se o acesso for permitido o *broker* retorna um *SUBACK*. A patir desse momento começa a troca de informação entre *broker* e os diversos *Subscriber* presentes.

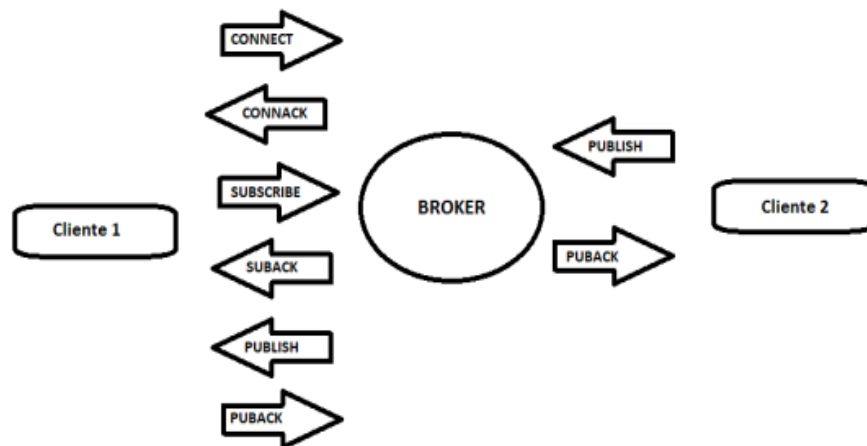


Figura 3: Estrutura do Protocolo MQTT

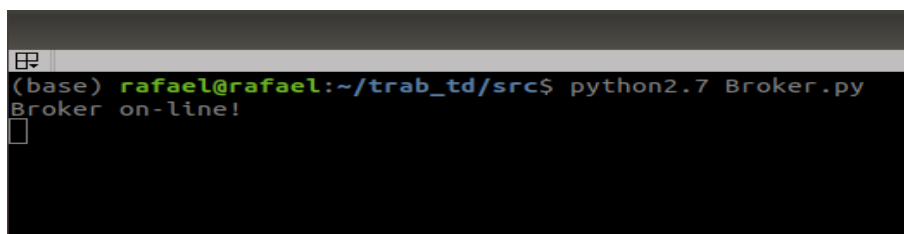
## 4 Instruções para compilação/execução do código

Como o trabalho foi implementado em python2.7 não se faz necessário sua compilação, porém é necessário que a máquina que irá rodar o código tenha o interpretador python2.7.

Para a implementação da tarefa proposta para este trabalho, a implementação tem 3 (três) códigos que são: *Broker.py*, *Publisher.py* e *Subscriber.py*, cada um com suas respectivas funcionalidades como descrito na seção 3.2.

Para executar a aplicação, abra o terminal do linux, começando pelo *Broker.py*, e digite **python2.7 Broker.py** e aperte ENTER.

Após esse comando irá aparecer uma mensagem como a mostrada na figura 4 abaixo.

A terminal window with a dark background. The prompt is '(base) raphael@rafael:~/trab\_td/src\$'. The command 'python2.7 Broker.py' has been entered and executed. The output is 'Broker on-line!' followed by a cursor on the next line.

```
(base) raphael@rafael:~/trab_td/src$ python2.7 Broker.py
Broker on-line!
█
```

Figura 4: Confirmação de que o broker está ativo

Neste momento o broker está esperando uma requisição de inscrição.

Para iniciar o *Publisher.py*, abra outra seção no terminal e digite o comando **python2.7 Publisher.py** e aperte ENTER.

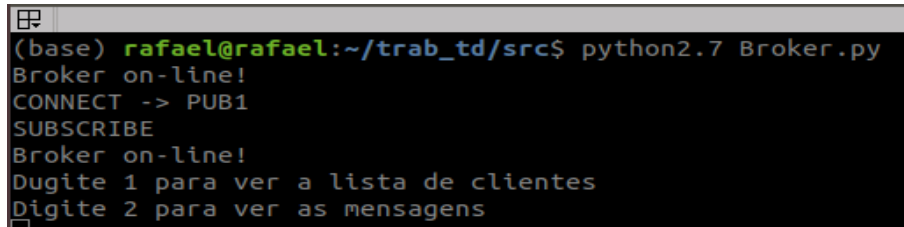
Após esse comando irá aparecer uma mensagem como a mostrada na figura 5 abaixo.

A terminal window with a dark background. The prompt is '(base) raphael@rafael:~/trab\_td/src\$'. The command 'python2.7 Publisher.py' has been entered and executed. The output is 'Subscriber on-line', 'CONNACK -> PUB1', and 'PUBACK' followed by a cursor on the next line.

```
(base) raphael@rafael:~/trab_td/src$ python2.7 Publisher.py
Subscriber on-line
CONNACK -> PUB1
PUBACK
█
```

Figura 5: Confirmação de que o Publisher está ativo

Na seção do terminal, que está rodando o *Broker*, irá aparecer as mensagens de confirmação de inscrição e de validação para publicação e um menu. A figura 6 mostra essa mensagem.

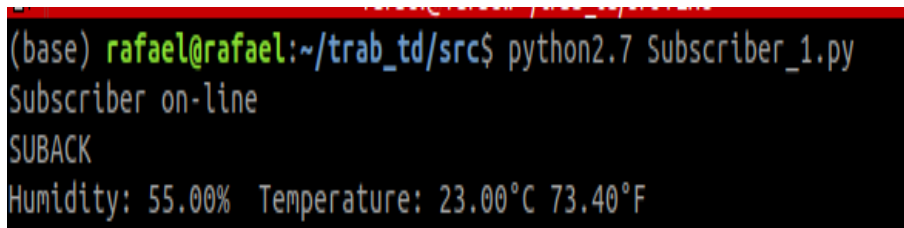
A terminal window with a dark background and light green text. The prompt is (base) rafaelf@rafael:~/trab\_td/src\$. The command python2.7 Broker.py has been executed. The output is: Broker on-line!, CONNECT -> PUB1, SUBSCRIBE, Broker on-line!, Digite 1 para ver a lista de clientes, Digite 2 para ver as mensagens.

```
(base) rafaelf@rafael:~/trab_td/src$ python2.7 Broker.py
Broker on-line!
CONNECT -> PUB1
SUBSCRIBE
Broker on-line!
Digite 1 para ver a lista de clientes
Digite 2 para ver as mensagens
```

Figura 6: Confirmação de inscrição mais menu

Para iniciar o *Subscriber.py*, abra outra seção no terminal e digite o comando **python2.7 Subscriber.py** e aperte ENTER.

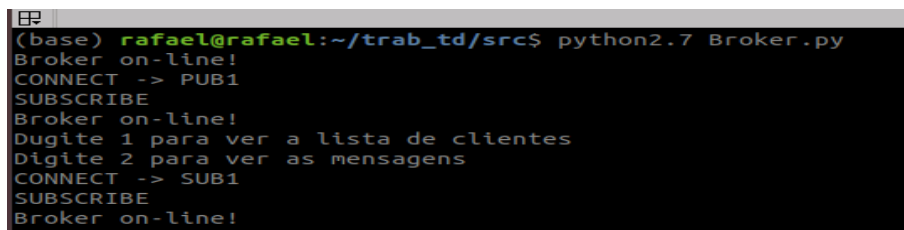
Após esse comando irá aparecer um uma mensagem como a mostrada na figura 7 abaixo.

A terminal window with a dark background and light green text. The prompt is (base) rafaelf@rafael:~/trab\_td/src\$. The command python2.7 Subscriber\_1.py has been executed. The output is: Subscriber on-line, SUBACK, Humidity: 55.00% Temperature: 23.00°C 73.40°F.

```
(base) rafaelf@rafael:~/trab_td/src$ python2.7 Subscriber_1.py
Subscriber on-line
SUBACK
Humidity: 55.00% Temperature: 23.00°C 73.40°F
```

Figura 7: Confirmação de que o Subscriber está ativo

Na seção do terminal, que está rodando o *Broker*, irá aparecer as de confirmação de inscrição e do Subscriber. A figura 8 mostra essa mensagem.

A terminal window with a dark background and light green text. The prompt is (base) rafaelf@rafael:~/trab\_td/src\$. The command python2.7 Broker.py has been executed. The output is: Broker on-line!, CONNECT -> PUB1, SUBSCRIBE, Broker on-line!, Digite 1 para ver a lista de clientes, Digite 2 para ver as mensagens, CONNECT -> SUB1, SUBSCRIBE, Broker on-line!, DISCONNECT SUB1.

```
(base) rafaelf@rafael:~/trab_td/src$ python2.7 Broker.py
Broker on-line!
CONNECT -> PUB1
SUBSCRIBE
Broker on-line!
Digite 1 para ver a lista de clientes
Digite 2 para ver as mensagens
CONNECT -> SUB1
SUBSCRIBE
Broker on-line!
DISCONNECT SUB1
```

Figura 8: Confirmação de inscrição do Subscriber no Broker

No menu do broker pode-se, se assim quiser, ver a lista de clientes ou a mensagem que pode ser transmitida.

## 5 Conclusão

Apesar de já existir muito coisa pronta sobre o protocolo de comunicação MQTT, optou-se por usar apenas o protocolo TCP/IP para sua implementação. O que podemos concluir, com base na implementação, é que o protocolo MQTT não é tão trivial de ser implementado, pois envolve muitos agentes operando ao mesmo tempo.

A principal dificuldade, foi conciliar todos os agentes usando a mesma porta de comunicação TCP/IP, tanto para verificar o *status* da comunicação quanto para fluxo de dados.

## Referências

- [1] KUROSE and ROSS (2016) *Redes de computadores e a internet*. Editora PEARSON, sexta edição.
- [2] <https://www.embarcados.com.br/mqtt-protocolos-para-iot/>
- [3] <https://docs.python.org/2.7/>
- [4] <http://mqtt.org/>
- [5] <https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Overview>