

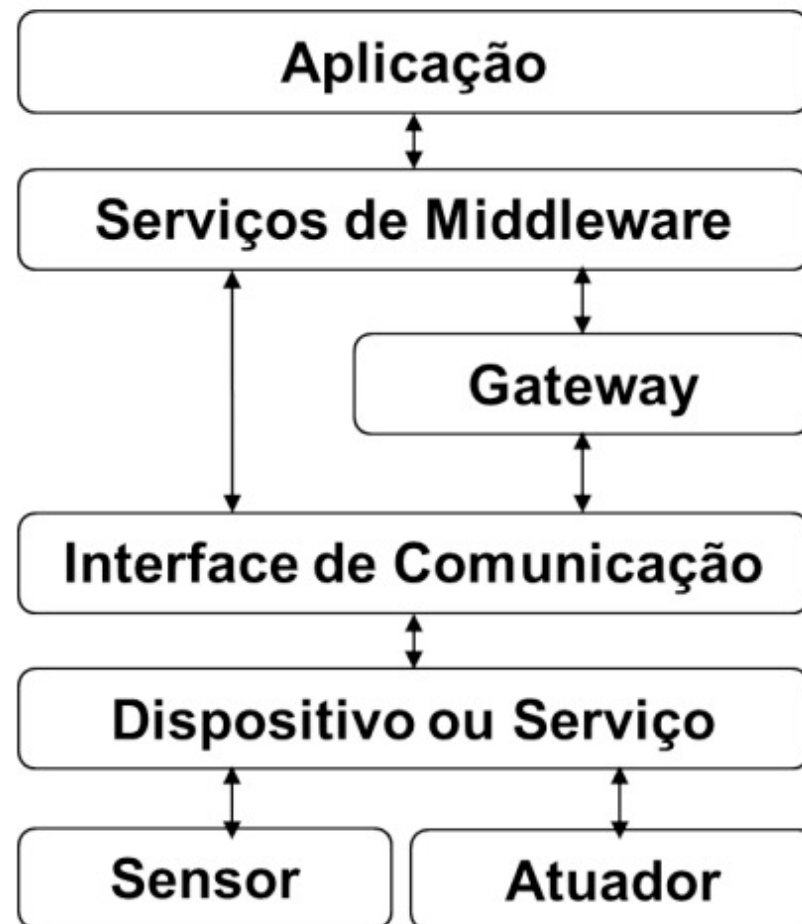
PROTOCOLOS DE COMUNICAÇÃO PARA IOT

MQTT

Sistemas Embarcados – Prof. Marcos Chaves

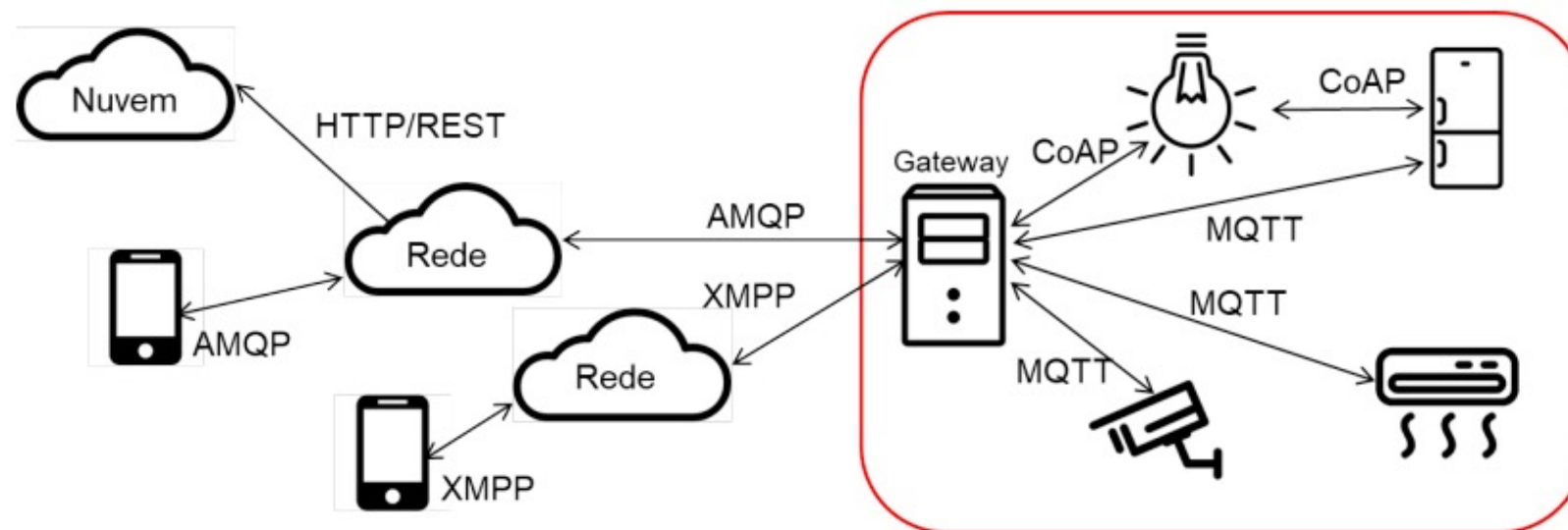
Engenharia de Controle e Automação/IFSP Catanduva

Organização de elementos na IoT



Múltiplos protocolos de aplicação

Vários Protocolos de Aplicação estão estabelecidos e alguns padronizados, com destaque para o MQTT, CoAP, AMQP e XMPP. Cada um tem características indicadas para determinadas aplicações em IoT, dependendo dos recursos e modos de operação dos dispositivos ou serviços a serem integrados. Ainda, a tecnologia WebSocket e a abordagem REST sobre HTTP/TCP



Interação dos protocolos

Sistemas de acionamento e monitoramento normalmente utilizam dois estilos de interação: requisição-resposta (request-response) e publica-subscribe (publish-subscribe)

Request-Response:

Direta (Requisição-Resposta) Protocolos de Aplicação que utilizam o estilo request-response são estruturados para oferecer interação síncrona e direta entre dois processos.

Publish-Subscribe:

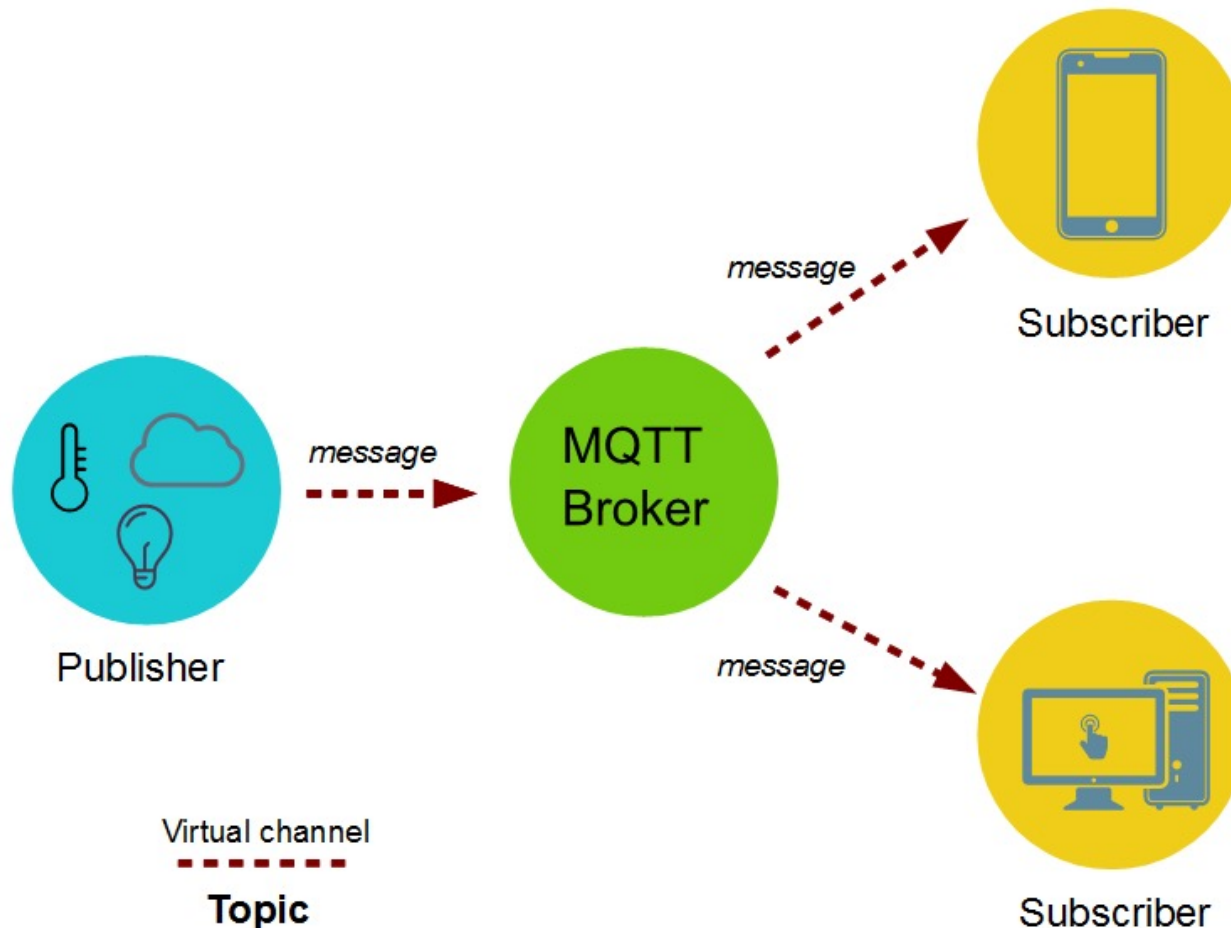
Indireta (Publica-Subscreve) O estilo de interação indireta, publica-subscribe, também chamado de interação baseada em eventos (event-based), é apropriado para comunicação assíncrona, quando os elementos que precisam interagir, não podem ou não querem estabelecer uma comunicação direta para trocar mensagens, ou não podem aguardar bloqueados até a interação ocorrer.

WebSockets

O WebSocket é um mecanismo similar ao socket “tradicional”, proposto para oferecer suporte para troca de mensagens bidirecionais, full-duplex, sobre TCP, entre um navegador Web e um servidor HTTP remoto, que suportem HTML5

MQTT

- MQTT(Message Queuing Telemetry Transport) é um protocolo de comunicação máquina para máquina (M2M - Machine to Machine)



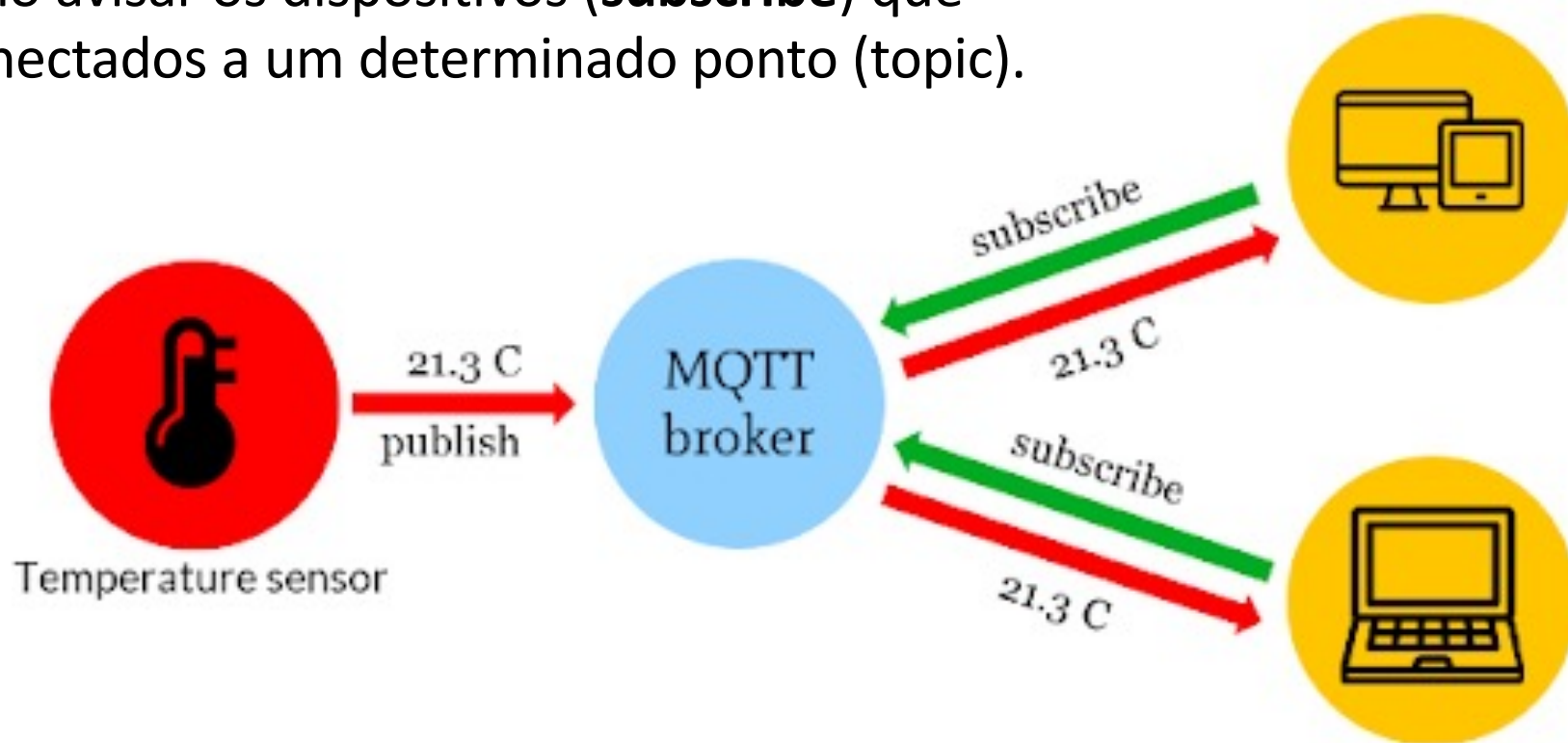
Comunicação
baseada em
**Publish e
Subscriber**

Breve Histórico

MQTT é um protocolo de mensagens de publicação/assinatura que funciona sobre o protocolo TCP/IP. A primeira versão do protocolo foi desenvolvida por Andy Stanford-Clark da IBM e Arlen Nipper da Cirrus Link em 1999. O MQTT torna-se mais rápido em enviar solicitações HTTP para dispositivos IoT pois as mensagens MQTT podem ter apenas 2 bytes, enquanto HTTP requerem cabeçalhos e outras informações as quais outros dispositivos podem não se importar. Além disso, se você tiver vários dispositivos aguardando uma solicitação com HTTP, precisará enviar uma ação POST para cada cliente. Com o MQTT, quando um servidor recebe as informações de um cliente, ele automaticamente distribui essas informações para cada um dos clientes interessados.

Componentes do MQTT

De forma geral, para a utilização do MQTT, é necessário um **broker**, o qual funciona como um intermediário (algo parecido com um servidor) mas que consegue interpretar envios (**publish**) de valores bem como avisar os dispositivos (**subscribe**) que estão conectados a um determinado ponto (topic).



Schematic data flow from sensor (machine) to devise (machine)

Componentes do MQTT

Broker - O broker é o servidor que distribui as informações aos clientes interessados conectados ao servidor.

Client - O dispositivo que se conecta ao broker para enviar ou receber informações.

Topic - O nome sobre o qual a mensagem se refere. Os clientes publicam, assinam ou fazem as duas coisas em um tópico.

Publish - Clientes que enviam informações ao corretor para distribuir aos clientes interessados com base no nome do tópico.

Subscribe - Os clientes informam ao Broker em quais tópicos estão interessados. Quando um cliente se inscreve em um tópico, qualquer mensagem publicada no Broker é distribuída aos assinantes desse tópico. Os clientes também podem cancelar a assinatura para parar de receber mensagens do broker sobre esse tópico.

Qualidade de serviço MQTT

QoS - Qualidade de Serviço. Cada conexão pode especificar uma qualidade de serviço para o broker com um valor inteiro variando de 0 a 2. A QoS não afeta o manuseio das transmissões de dados TCP, apenas entre os clientes MQTT.

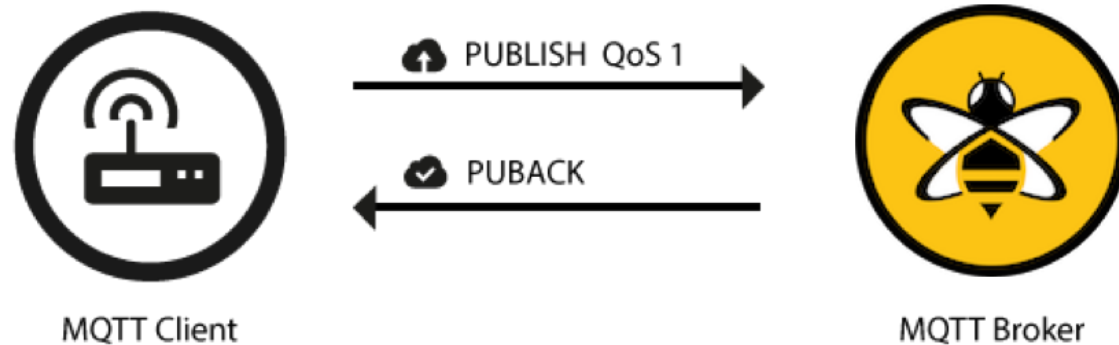
- **0** especifica no máximo uma vez, ou uma vez e apenas uma vez sem exigir uma confirmação de entrega. Isso é muitas vezes referido como fogo e esquecer.
- **1** especifica pelo menos uma vez. A mensagem é enviada várias vezes até que uma confirmação seja recebida, também conhecida como entrega confirmada.
- **2** especifica exatamente uma vez. Os clientes remetente e destinatário usam um *handshake* de dois níveis para garantir que apenas uma cópia da mensagem seja recebida, conhecida como entrega garantida.

Qualidade de serviço MQTT

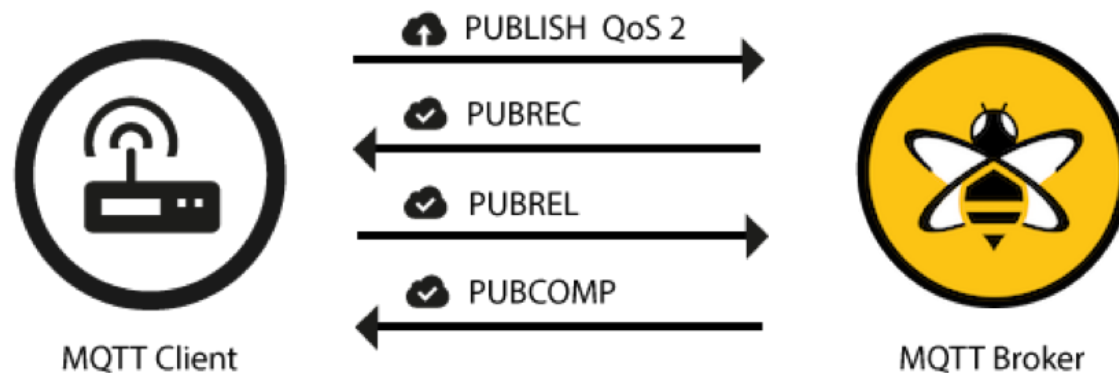
QoS 0 - No máximo uma vez

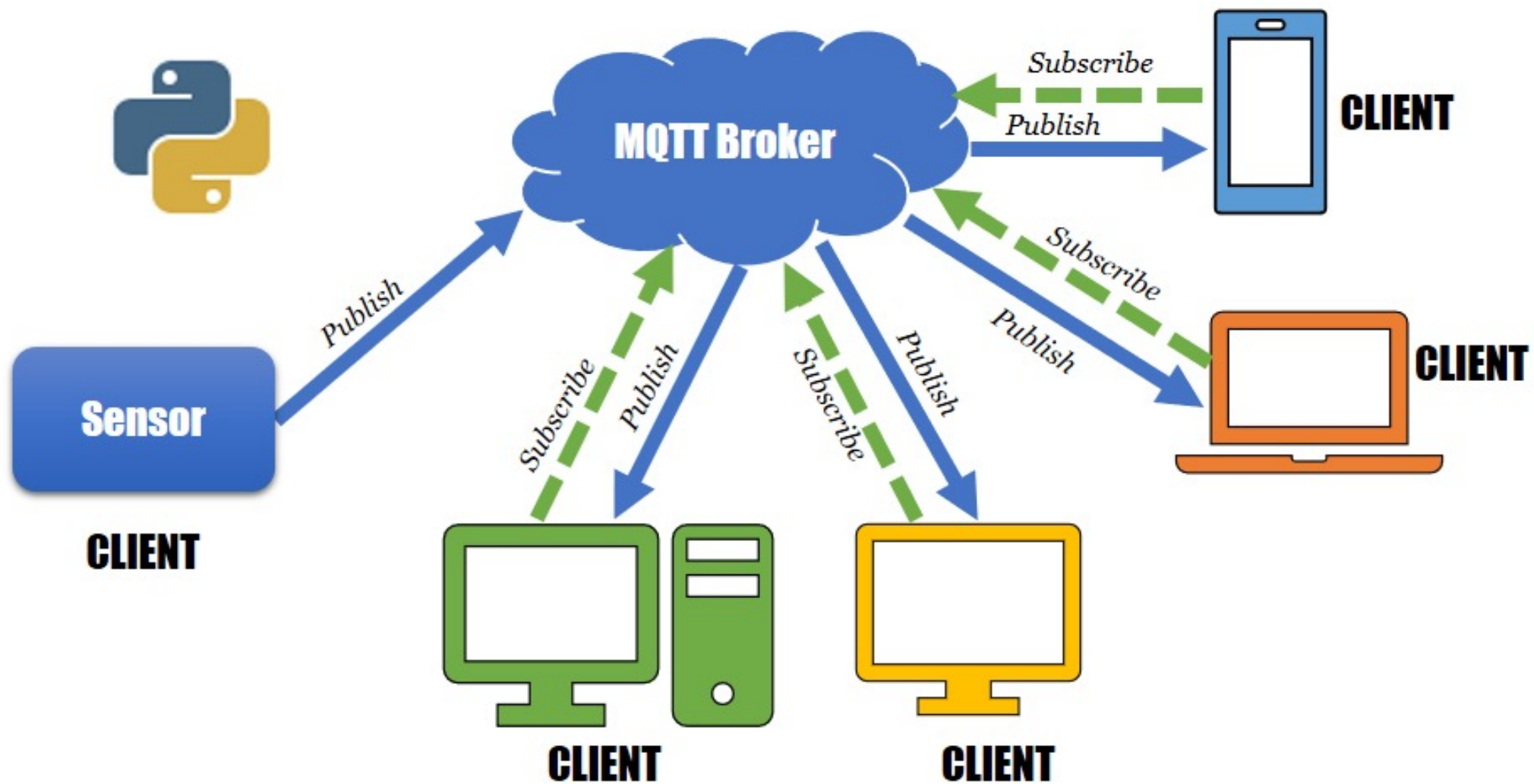


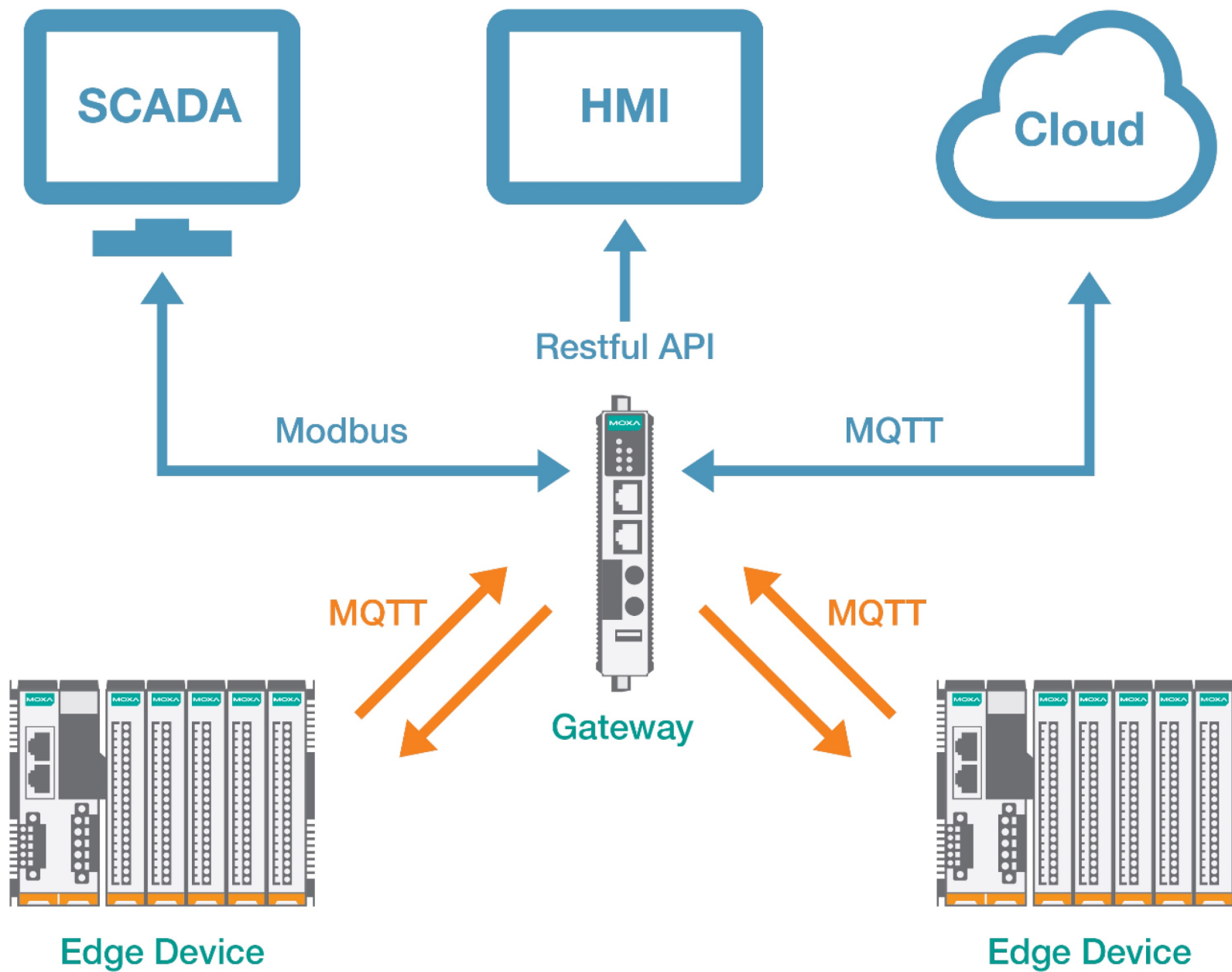
QoS 1 - Pelo menos uma vez



QoS 2 - Exatamente uma vez





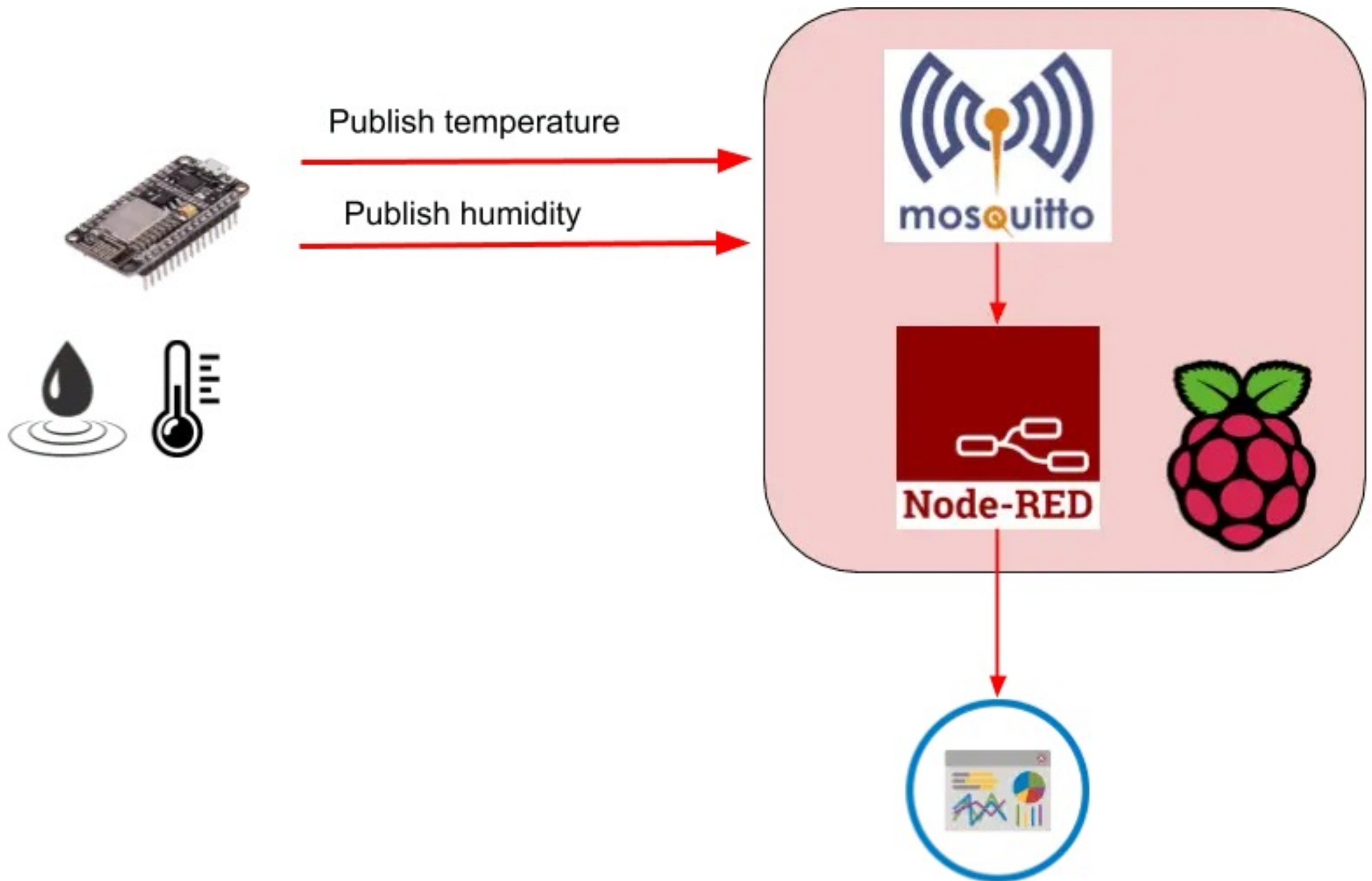


Aplicação do protocolo MQTT na indústria

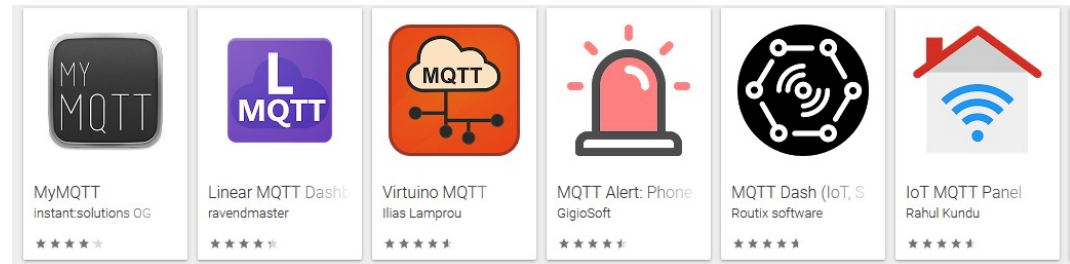
Brokers

Broker	Server	Ports	Websocket
Mosquitto	iot.eclipse.org	1883 / 8883	n/a
HiveMQ	broker.hivemq.com	1883	8000
Mosquitto	test.mosquitto.org	1883 / 8883 / 8884	8080 / 8081
mosca	test.mosca.io	1883	80
HiveMQ	broker.mqttdashboard.com	1883	

Aplicação ESP32 + MQTT + Node-Red



Utilizando Smartphone como cliente MQTT



Existem diversos aplicativos para celular, utilizar o protocolo mqtt para suas aplicações IOT. O aplicativo android: [MQTT Dash](#) é muito simples de operar.


Referências:

<http://smartcampus.ctd.ifsp.edu.br/blog/index.php?IDselecionado=48>

<https://www.curtocircuito.com.br/blog/introducao-ao-mqtt>

<https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/mqtt/#pubsub>

http://www.hivemq.com/demos/websocket-client/



HIVEMQ

Websockets Client Showcase

Connection

Host

broker.mqttdashboard.com

Port

8000

ClientID

clientId-x7wXpv9a7k

Disconnect

Username

mchavesferreira

Password

.....

Keep Alive

60

SSL

☐

Clean Session

☒

Last-Will Topic

Last-Will QoS

0

Last-Will Retain

☐

Last-Will Message

Publish

Topic

movel/botao

QoS

0

Retain

☐

Publish

Message

0

Subscriptions

Add New Topic Subscription

Qos: 0

movel/botao

X

Qos: 0

movel/accelera

X