Aula 08 - Protocolos para IoT

</ />

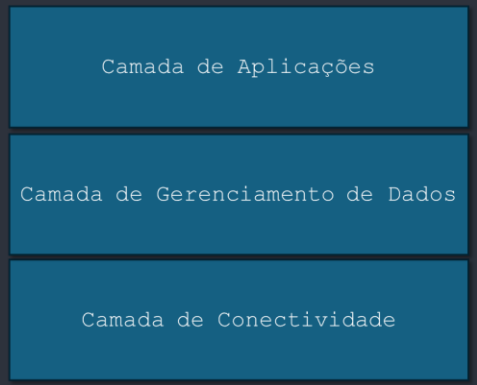
Prof. Dr. Jordan P. Sausen

UNIVALI

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Introdução

- Comunicação entre hardware (sensor e microcontrolador) 👉 Camada 1
- Dispositivos IoT são conectados à Internet usando o protocolo IP 👉 Camada 3
- Aplicações IoT: protocolos de camada de aplicação 👉 Camada 7



1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 1

</Introdução

</ Como fazer a escolha do protocolo de aplicação? />

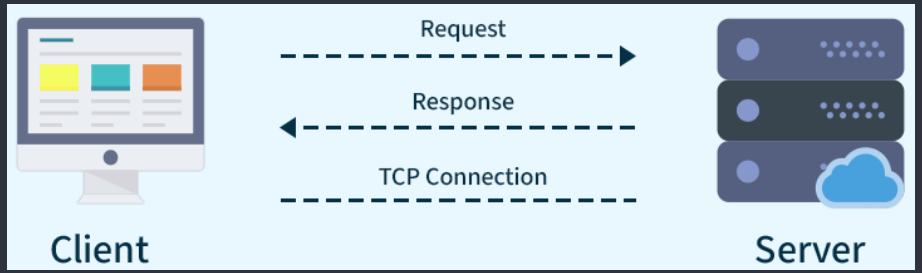
- Protocolos tradicionais: solicitação-resposta (cliente-servidor)
  - HTTP e HTTPS
- Modelo comum em IoT: publicação-assinatura (pub/sub)
  - MQTT, CoAP, AMQP

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo HTTP

HyperText Transfer Protocol

- Conjunto de regras de comunicação entre cliente e servidor
- Tecnologia que alimenta a comunicação de rede
- Quando você visita um site, o navegador envia uma solicitação HTTP ao servidor Web, que responde com uma resposta HTTP



- O servidor Web e o navegador trocam dados de texto simples

1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo HTTP

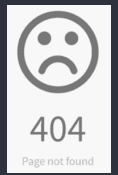
HyperText Transfer Protocol

HTTP GET. Leitura de dados

HTTP POST. Criar ou atualizar informações

HTTP PUT. Substituição de dados

HTTP DELETE. Apagar informações

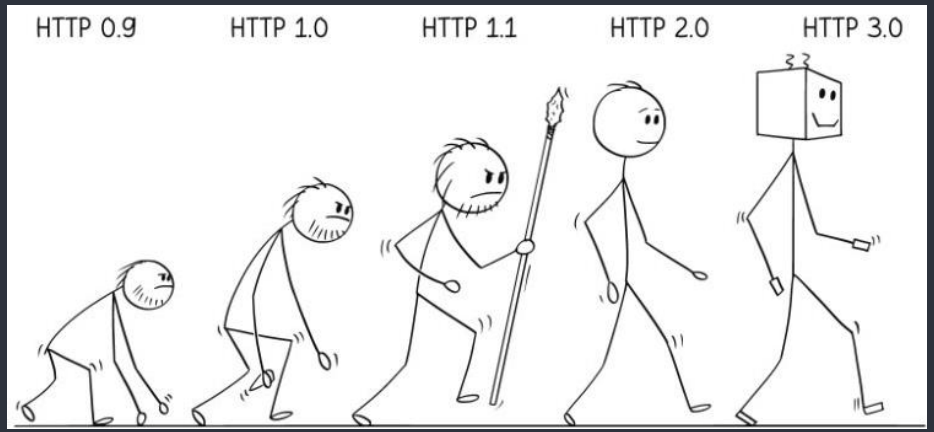


200 - OK
400 - Solicitação inválida
404 - Recurso não encontrado

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo HTTP

HyperText Transfer Protocol



HTTP/2. Dados binários

HTTP/3. Transmissões em tempo real

HTTPS. Segurança de dados

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo HTTPS

HyperText Transfer Protocol Secure

UNIVALI

Camada adicional de segurança para comunicação que combina solicitações e respostas HTTP com a tecnologia SSL e TLS:

1. Você visita um site HTTPS digitando o formato de URL https://na barra de endereço do seu navegador.
2. O navegador tenta verificar a autenticidade do site solicitando o certificado SSL do servidor.
3. O servidor envia o certificado SSL que contém uma chave pública como resposta.
4. O certificado SSL do site comprova a identidade do servidor. Quando o navegador estiver satisfeito, ele usará a chave pública para criptografar e enviar uma mensagem que contém uma chave de sessão secreta.
5. O servidor web usa sua chave privada para descriptografar a mensagem e recuperar a chave de sessão. Em seguida, ele criptografa a chave da sessão e envia uma mensagem de confirmação ao navegador.
6. Agora, o navegador e o servidor da Web mudam para usar a mesma chave de sessão para trocar mensagens com segurança.

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</HTTP vs HTTPS

	HTTP HyperText Transfer Protocol	HTTPS HyperText Transfer Protocol Secure
PROTOCOLOS.	HTTP/1 e 2 usam TCP/IP. HTTP/3 usa QUIC	HTTP/2 com SSL/TLS
PORTA.	Porta padrão 80	Porta padrão 443
USO.	Sites antigos de texto	Todos os sites modernos
SEGURANÇA.	Nenhum recurso adicional	Certificados SSL para criptografia de chave pública
BENEFÍCIOS.	Tornou possível a comunicação pela internet	Melhora autoridade, confiança e classificação dos mecanismos de pesquisa

</Protocolo HTTPS

HyperText Transfer Protocol Secure

UNIVALI

Por que escolher HTTPS em vez do HTTP?

Segurança

Criptografia

Performance e análise

Velocidade

Autoridade

Preferencia do usuário

</Protocolo HTTP

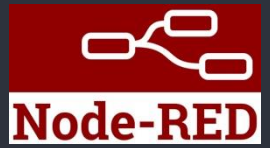
HyperText Transfer Protocol

- O HTTP pode ser usado como um protocolo de camada de aplicação no design de uma aplicação IoT.
- No entanto, devido ao consumo extensivo de energia do HTTP, ele não é eficiente para muitas aplicações IoT.
- Ele é repleto de cabeçalhos e regras que aumentam o tamanho dos pacotes e, portanto, não é adequado para dispositivos IoT que operam em redes com recursos limitados.
- O HTTP suporta um modelo de comunicação um-para-um, no qual há um cliente, um servidor e uma solicitação por vez

1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1

</Protocolo HTTP com

HyperText Transfer Protocol



UNIVALI

TE (20 minutos)

- Baixar e instalar em http://nodejs.org
- Verificar a versão do Node-RED instalada usando o comando `node-red --version` em um terminal do Windows/OS
- Instalar npm usando o comando `npm install -g -unsafe-perm node-red` no mesmo terminal do Windows/OS
- Iniciar o Node-RED usando o comando `node-red -v' no mesmo terminal do Windows/OS
- Copiar porta do terminal Ex: http://127.0.0.1:1880/
- Colar no navegador

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo HTTP com Node-RED

HyperText Transfer Protocol

UNIVALI

TE (30 minutos)

- OBJETIVO: Aprender a fazer chamadas a uma API usando Node-RED para obter dados meteorológicos 👉 https://www.weatherbit.io/api/weather-current
- Com o Node-RED aberto, adicione um *timestamp*, uma Requisição HTTP e um *debug*
- Configure o nó para realizar uma chamada HTTP GET para a API do WeatherBit e exibir a resposta no formato JSON
- Implemente o fluxo e verifique a resposta para a sua cidade

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Mensagens pub/sub

publisher/subscriber

- Modelo de comunicação assíncrona
- Facilita a criação de aplicações funcionais e complexas na nuvem
- Desacoplamento de aplicações em blocos menores e independentes 👉 serviços
- Notificações instantâneas de eventos para sistemas distribuídos
- Suporte a comunicação escalável e confiável entre módulos de software independentes

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Mensagens pub/sub

publisher/subscriber

{*}

Mensagens

strings de texto, vídeo,
dados de sensores, áudio, etc

{*}

Tópicos

canal entre emissores e
receptores

{*}

Assinantes

destinatário da mensagem

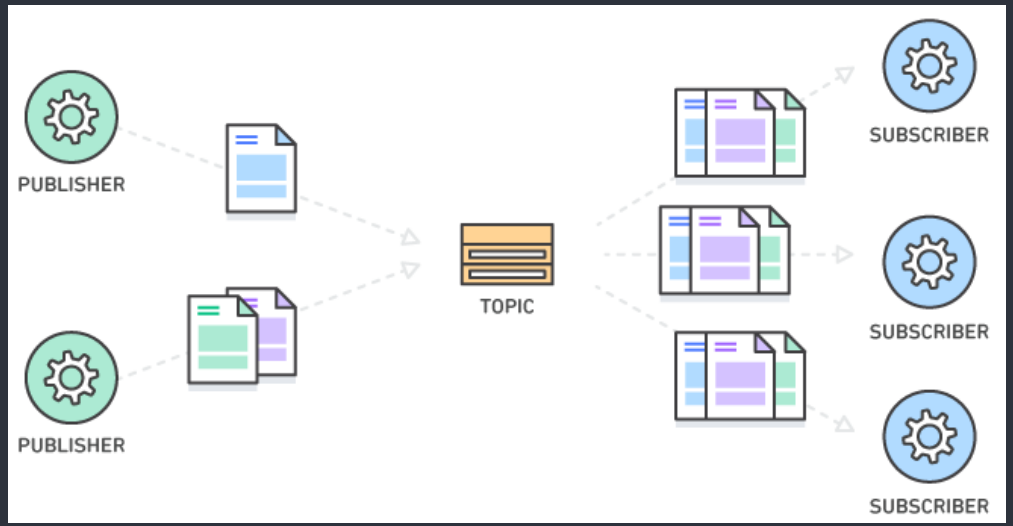
{*}

Publicador

envio de mensagens

</Mensagens pub/sub

publisher/subscriber



1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo MQTT

Message Queuing Telemetry Transport

UNIVALI

- Baseado em padrões ou conjunto de regras
- Limitação de recursos e largura de banda limitada 👉 dispositivos a bateria
- Fácil de implementar e pode comunicar dados IoT com eficiência
- Suporte a mensagens entre dispositivos para a nuvem
- Suporte a mensagens entre nuvem para o dispositivo
- Um software cliente MQTT completo pode caber em microcontroladores com 32 KB de memória flash e 2 KB de RAM

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo MQTT

UNIVALI

MQTT em Internet das Coisas

Leve e eficiente
Recursos mínimos

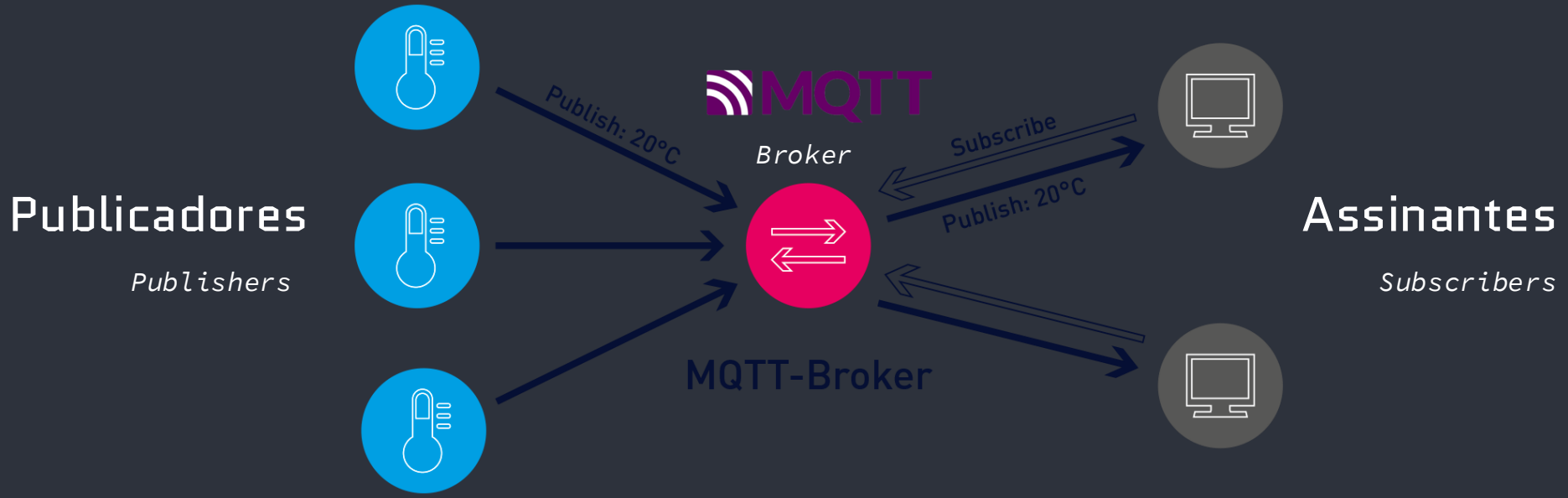
Escalável
Pouco código, pouca energia

Confiável e seguro
QoS, criptografia e autenticação

Bom suporte
Python e outros

</Protocolo MQTT

UNIVALI



</Protocolo MQTT

UNIVALI

Desacoplamento entre publicador/assinante

</Protocolo MQTT

UNIVALI

Cliente. Qualquer dispositivo que executa uma biblioteca MQTT

{publisher} — Envia mensagens

{subscriber} — Recebe mensagens

Agente. Backend que coordena mensagens entre os diferentes clientes

- Receber e filtrar mensagens, identificar clientes, encaminhas mensagens
- Autorizar e autenticar clientes MQTT
- Transmitir mensagens a outros sistemas parta análise posterior
- Solucionar mensagens e sessões perdidas

Conexão. Comunicação entre clientes e agentes

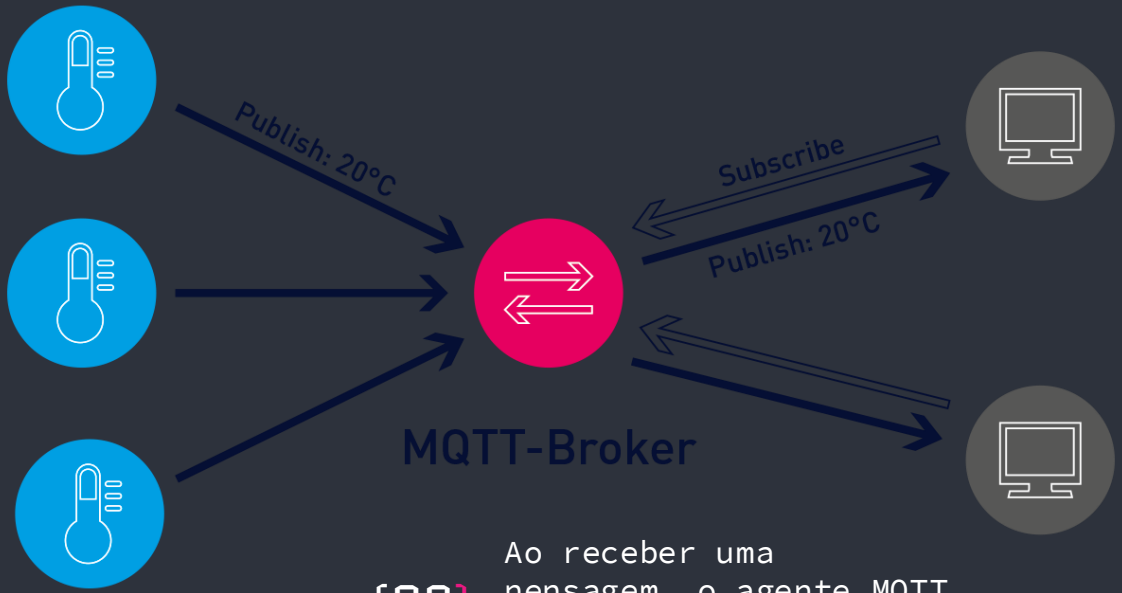
</CONNECT/> *</CONNACK/>*

1 0 1 1 0 1 1 0 1 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 0 1 1 1 0 1 1 0 1 1 1 1 1 0 1

</Protocolo MQTT - Como funciona?

{01}

Um cliente MQTT estabelece uma conexão



{02}

Após conectado, o cliente pode publicar/assinar mensagens

{03}

Ao receber uma nensagem, o agente MQTT a encaminha aos assinantes interessados

</Protocolo MQTT - Como funciona?

Tópico MQTT

.palavras-chave
usadas como filtro

.organizadas de
maneira hierárquica

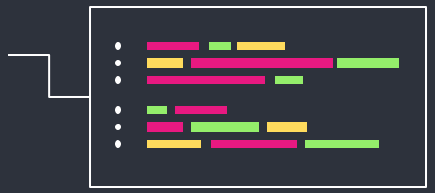
</ *ourhome/groundfloor/livingroom/light*
</ *ourhome/firstfloor/kitchen/temperature*

</Protocolo MQTT - Como funciona?

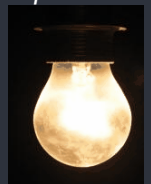
Publicação MQTT

.mensagens contendo o tópico e os dados

.formato de dados: texto, binário, arquivos XML ou JSON



</ *livingroom/light → lampada ON*

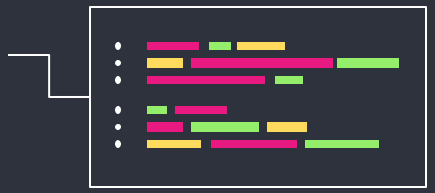


</Protocolo MQTT - Como funciona?

Assinatura MQTT

.subscribe para receber mensagem do tópico de interesse

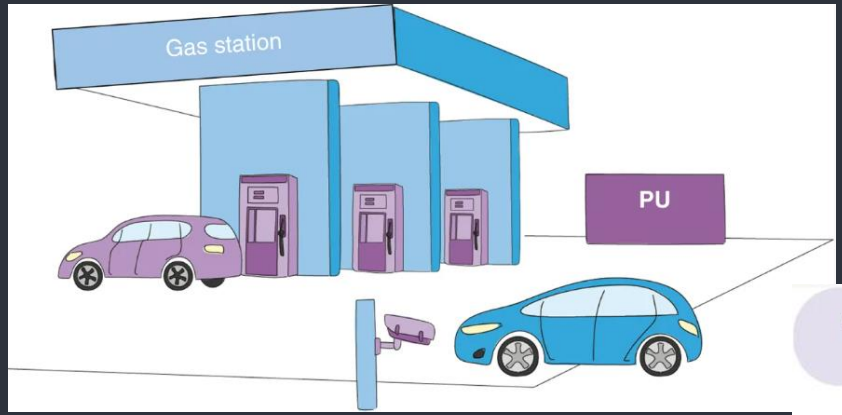
.identificador e lista de assinaturas

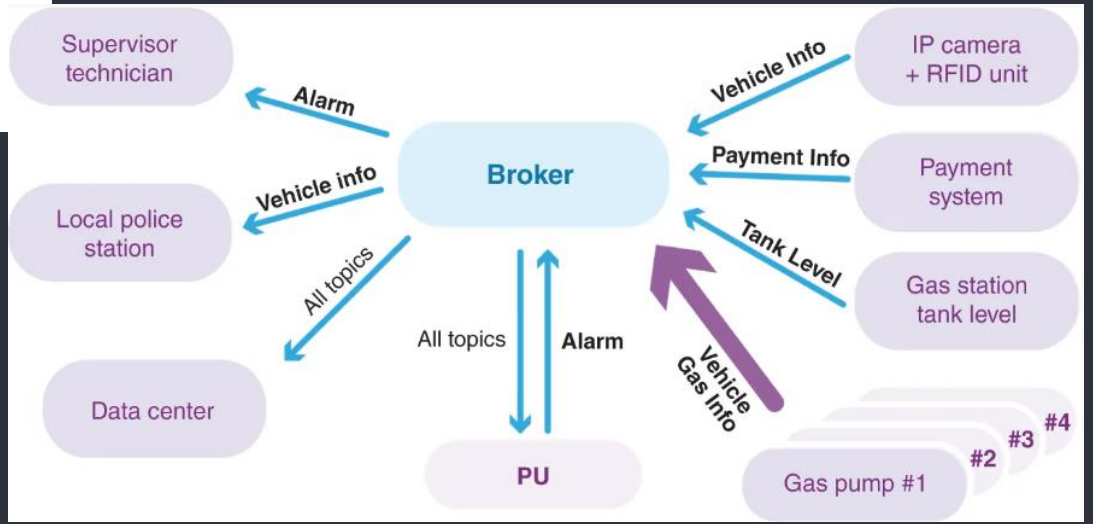


</ *Tópico → sub_light → count_ON*

</Protocolo MQTT

UNIVALI

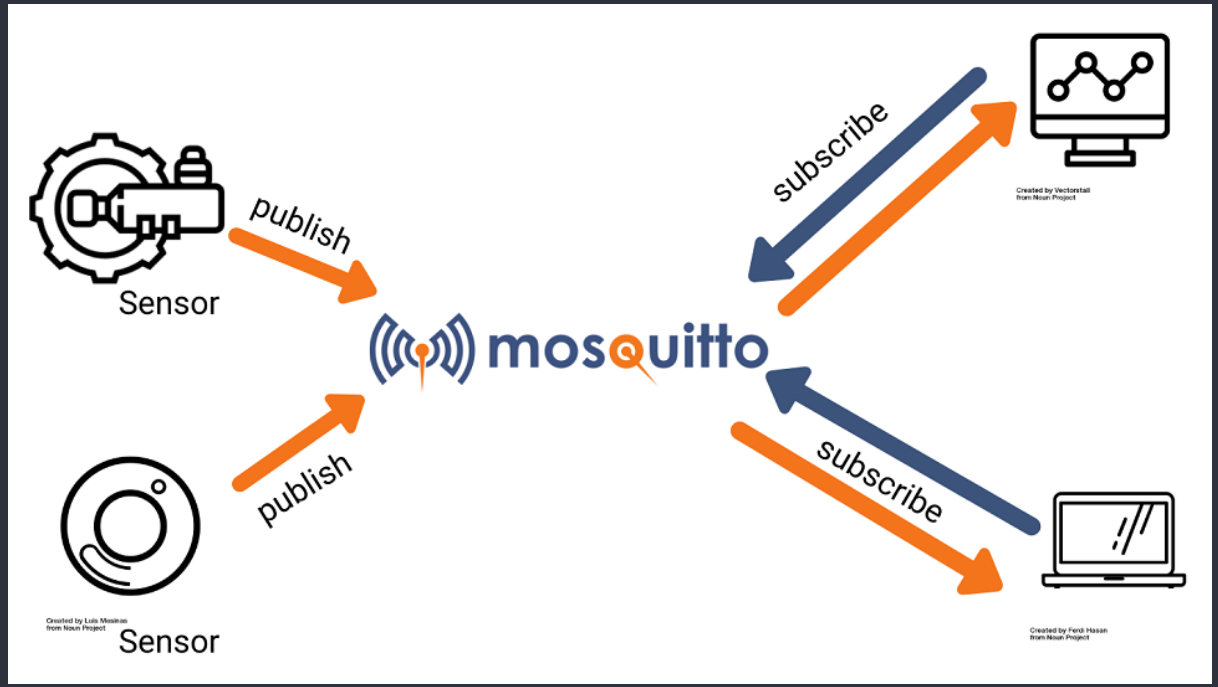




“sensors/building2/floor3/room4/temperature/heater”

- Tópico 1: "sensors/building2/floor3/room4/temperature/+"
- Tópico 2: "sensors/building2/+/room4/temperature/+"
- Tópico 3: "sensors/building2/floor3/room4/#"
- Tópico 4: "sensors/building2/#"

</Protocolo MQTT com



https://mosquitto.org/download/

</Protocolo MQTT com

UNIVALI

TE (20 minutos)

- Baixar e instalar https://mosquitto.org/download/ ou rodar via Docker
- Verificar se o Mosquitto está rodando corretamente usando o comando `mosquitto -v` em um terminal do Windows/OS

- Abrir outro terminal e executar o comando para criar um subscriber:

```
mosquitto_sub -h localhost -p 1883 -t “topico1”
```

Parâmetros:
mosquitto_sub: cliente subscriber do Mosquitto, usado para se inscrever em um ou mais tópicos e receber mensagens publicadas nesses tópicos.
-h localhost: -h especifica o endereço do host onde o broker MQTT está rodando (local), que indica que o broker MQTT está rodando na mesma máquina em que o comando está sendo executado. Se o broker estivesse em outra máquina, aqui você colocaria o IP ou o nome de domínio dessa máquina.
-p 1883: -p especifica a porta que o cliente usa para se conectar ao broker MQTT. 1883 é a porta padrão MQTT.
-t ‘topico1’: -t especifica o tópico ao qual o cliente deseja se inscrever. “topico1” é o nome do tópico. Tópicos são usados para organizar as mensagens no MQTT e podem ser definidos conforme necessário. O subscriber receberá todas as mensagens publicadas nesse tópico específico.

</Protocolo MQTT com

UNIVALI

TE (10 minutos)

- Abrir outro terminal e executar o comando para criar um publisher:
  mosquitto_pub -h localhost -p 1883 -t ‘topico1’ -m ‘Teste IoT Mestrado’

Parâmetros:
mosquitto_pub: cliente subscriber do Mosquitto, usado para se inscrever em um ou mais tópicos e receber mensagens publicadas nesses tópicos.
-m ‘Teste IoT Mestrado’: -t especifica o tópico ao qual o cliente deseja se inscrever. ‘topico1’ é o nome do tópico. Tópicos são usados para organizar as mensagens no MQTT e podem ser definidos conforme necessário. O subscriber receberá todas as mensagens publicadas nesse tópico específico.

</Protocolo CoAP

Constrained Application Protocol

- Protocolo REST muito leve que o torna adequado para dispositivos IoT com recursos limitados.
- O CoAP utiliza comandos GET, PUT, POST e DELETE similares, mas não exatamente iguais, ao HTTP (TCP).
- CoAP e HTTP têm muitas características semelhantes, com a diferença de que o CoAP é otimizado para IoT (UDP)

CoAP GET. Leitura de dados

CoAP POST. Criar ou atualizar informações

CoAP PUT. Substituição de dados

CoAP DELETE. Apagar informações

</Protocolo CoAP

Constrained Application Protocol

UNIVALI

- Em vez de tópicos MQTT, o CoAP usa o Identificador de Recurso Universal (URI).
- Pode-se encontrar uma espécie de semelhança entre URIs CoAP e tópicos MQTT.
- Por exemplo, um sensor de temperatura que publica suas informações de sensor para um servidor, utilizando os protocolos CoAP ou MQTT, pode usar os seguintes formatos, respectivamente:

Sensor CoAP publicando para o servidor CoAP:
URI: “coap://devices/sensors/temperature”

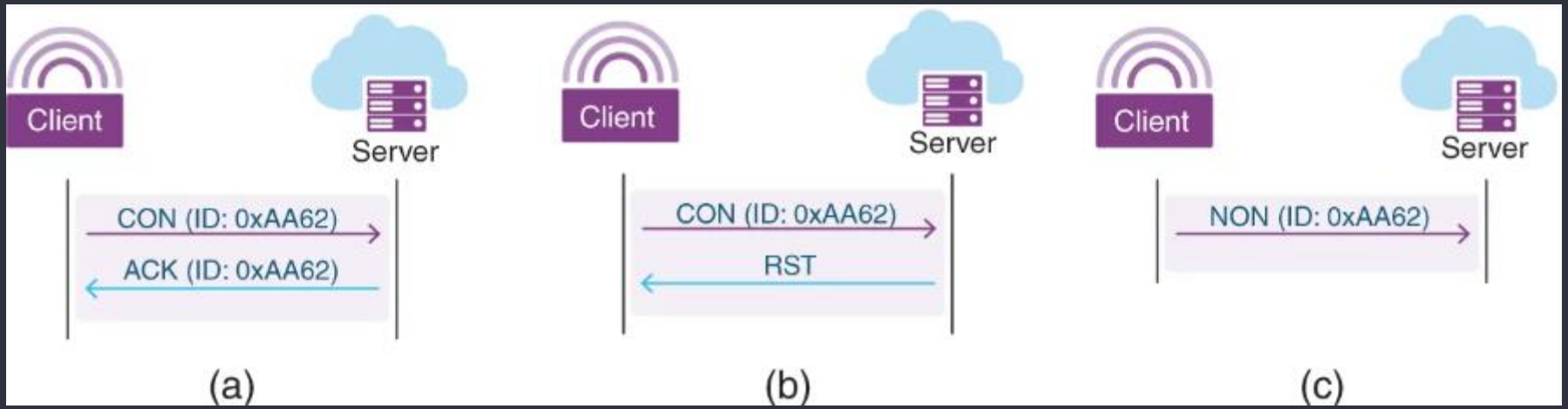
Cliente MQTT publicando para um broker MQTT:
tópico: “/devices/sensors/temperature”

</Protocolo CoAP

Constrained Application Protocol

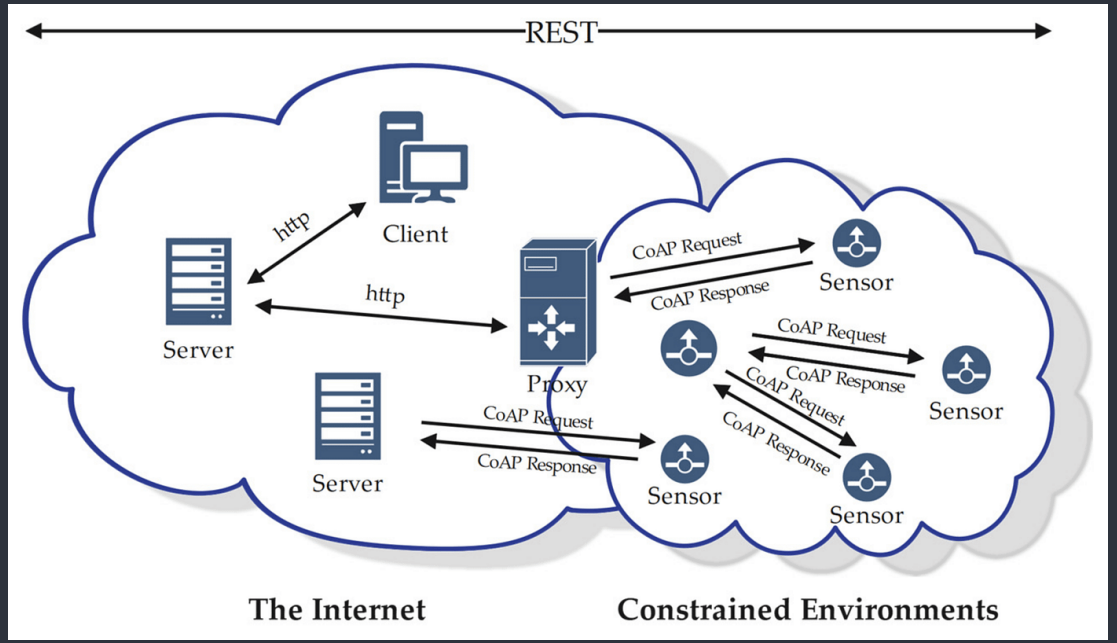
</Protocolo CoAP

Constrained Application Protocol



</Protocolo CoAP

Constrained Application Protocol



Capacidade multicast → um para muitos

</Protocolo MQTT com Node-RED

UNIVALI

ATIVIDADE COMPLEMENTAR

- Instalar MQTT Explorer https://mqtt-explorer.com/
- Seguir instruções do documento compartilhado

</Internet das Coisas (IoT)

Thanks!

jordan@univali.br