



Módulo 6:

Protocolo MQTT

Agenda

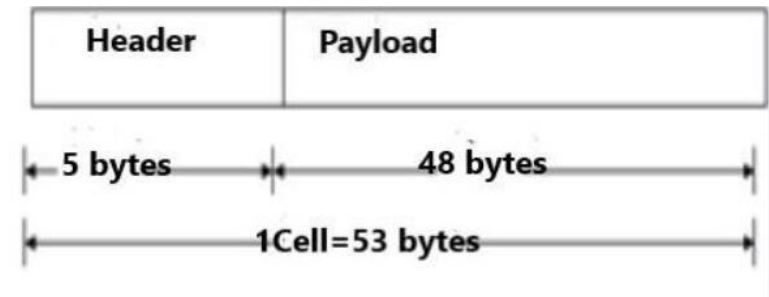
- Recordatorio: Protocolos de Internet
- Informacion general
- Caracteristicas
- Laboratorio 6



Recordatorio: Protocolos de Internet

¿Qué hace un protocolo de Internet?

- Definen las rutas y la forma de entrega de los mensajes
- Definen Header y payload
 - Header es un identificador
 - Payload es los datos a enviar
- Modelo OSI
 - Existen muchas tareas que los protocolos deben manejar
 - Ruteado, control de flujo, sistema de prioridad
 - OSI divide estas tareas en capas de abstracción



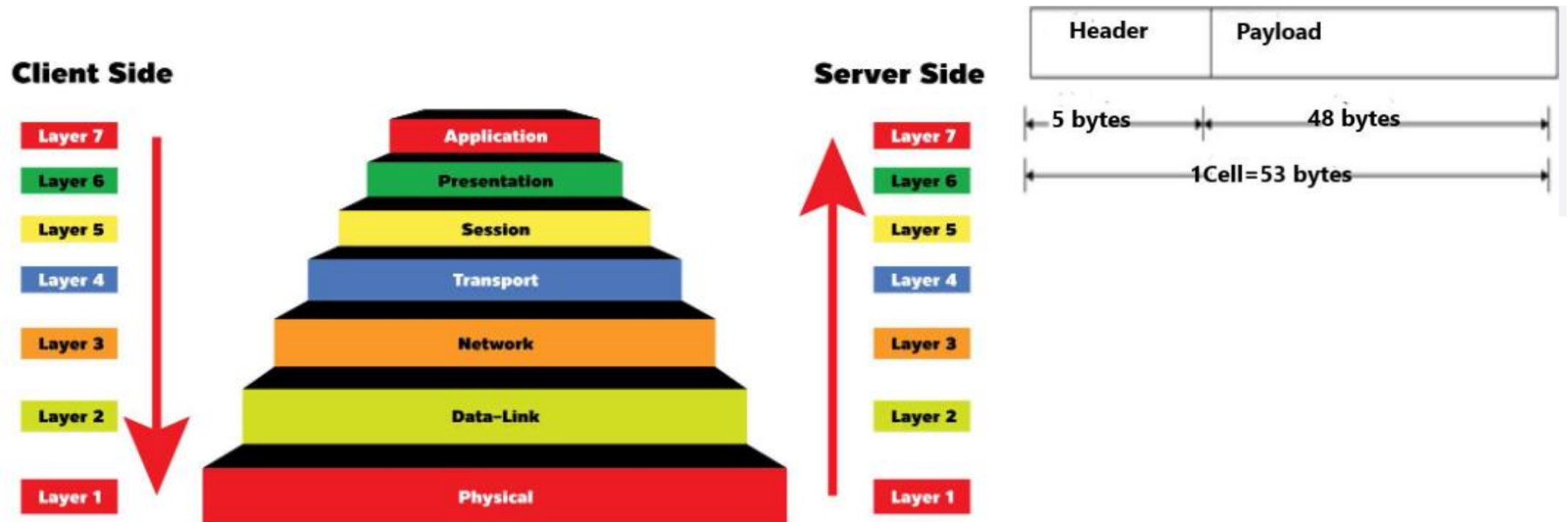
OSI MODEL





Recordatorio: Protocolos de Internet

OSI MODEL



Informacion General

MQTT: Message Queuing Telemetry Transport (Telemetria de mensajes de cola)

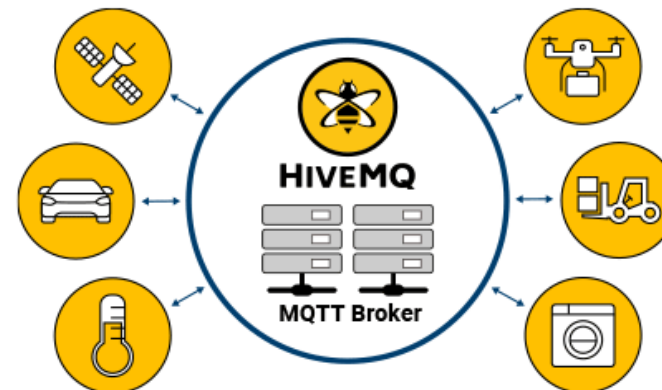
- Inventado en 1999 por IBM
- Publicado en 2010
- Se convirtio en un estandar oficial de OASIS en 2014
- Version actual: MQTT Version 5.0 <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>



MQTT: capa de aplicacion

Tipos de Capas de Aplicacion

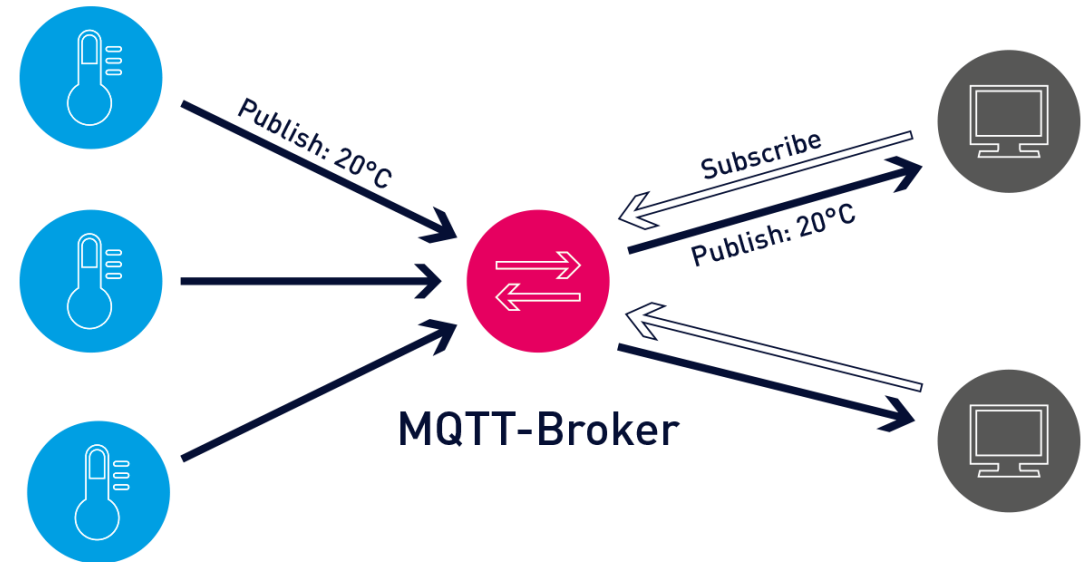
- Cliente/Servidor (navegador web)
 - HTTP
 - CoAP
- **Publicacion/Suscripcion** (twitter)
 - MQTT



MQTT: Características

Protocolo de transporte de mensajes tipo publicacion/suscripcion

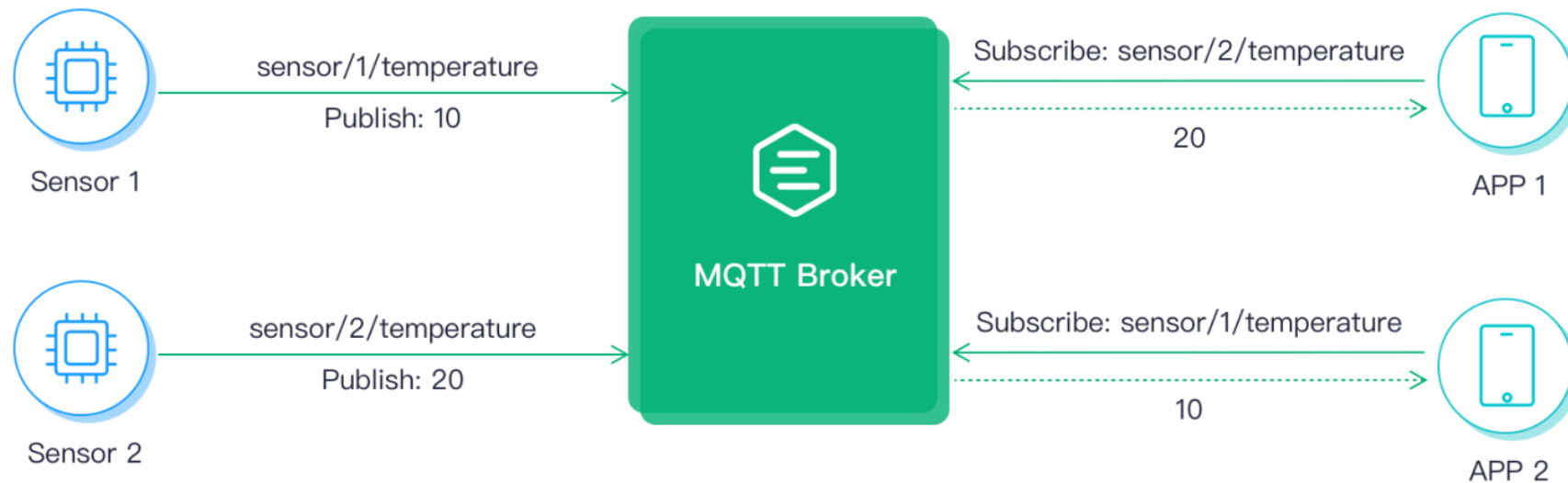
- Simple de implementar (especialmente en el lado del sensor)
- Soporte QoS (Quality of Service)
- Ligero y eficiente en ancho de banda
- Flexible, indiferente al formato de los datos
 - Importante en el campo de IoT
- Conciencia de sesion
 - Mantiene informacion acerca del estado del cliente



MQTT: Patron de comunicacion

Publicacion/suscripcion

- Los clientes no se conocen entre si
- Paradigma uno-a-muchos
- Todos los clientes pueden publicar y suscribirse



MQTT: Temas (topics)

Cadenas que se utilizan para etiquetar y categorizar los mensajes

topic level
separator
↓
myhome / groundfloor / livingroom / temperature
└───┬───┘ └───┬───┘
topic level topic level



- myhome / groundfloor / livingroom / temperature
- myhome / groundfloor / kitchen / temperature
- myhome / groundfloor / kitchen / **brightness**
- myhome / **firstfloor** / kitchen / temperature
- myhome / groundfloor / kitchen / **fridge** / temperature

- single-level
wildcard
↓
myhome / groundfloor / + / temperature
|
only one level

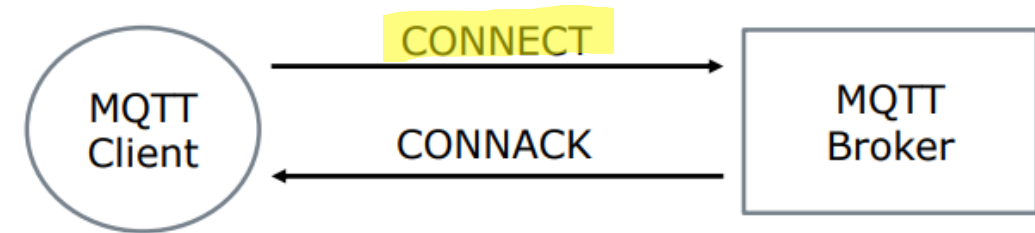
- multi-level
wildcard
↓
myhome / groundfloor / #
- only at the end
 - multiple topic levels

MQTT: Conexiones

Cada cliente MQTT abre una conexión al Broker MQTT

- Mensaje **CONNECT** (conectar)

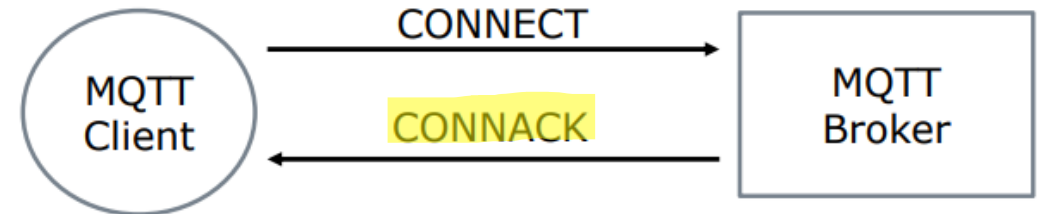
- clientId "clienteEmbedIoT"
- cleanSession true
- username (opcional) "claseiot"
- password (opcional) "1234"
- lastWillTopic (opcional) "claseiot/temperatura"
- lastWillQoS (opcional) 1
- lastWillMessage (opcional) "desconexión del servidor"
- keepAlive 30



MQTT: Conexiones

Cada cliente MQTT abre una conexion al Broker MQTT

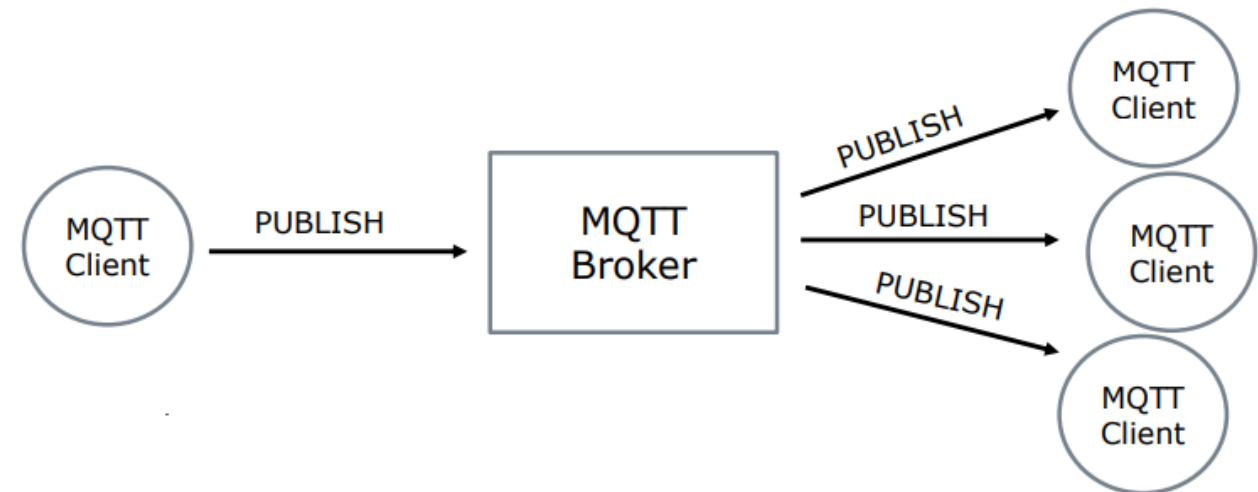
- Mensaje **CONNACK** (confirmacion de conneccion)
 - sessionPresent true
 - returnCode 0-5
 - 0: todo OK
 - 1: version no aceptada
 - 2: Id rechazada
 - 3: servidor no disponible
 - 4: usuario o password incorrecto
 - 5: no autorizado



MQTT: Publicacion

Mensaje PUBLISH

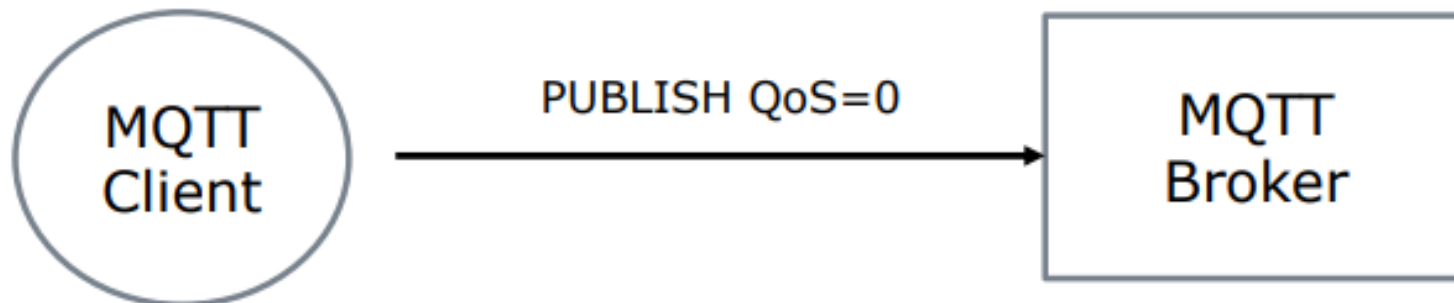
- packetId 2
- topicName "claseiot/temperatura"
- QoS 1
- retainFlag false
- payload "temperatura:30"
- dupFlag false



MQTT: Calidad de servicio

En ingles Quality of Service (QoS) → QoS: 0, QoS: 1, QoS: 2

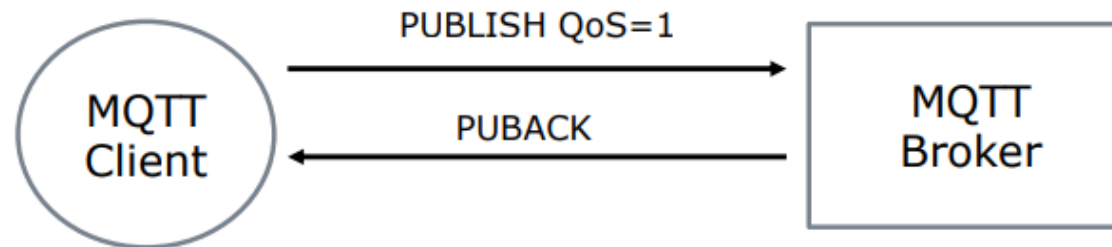
- QoS 0: “envia y olvida”
 - El mensaje se publica al broker y luego se reenvia a los suscriptores sin ninguna confirmacion
 - Sin garantia de entrega
 - Entrega rapida
 - Usado en escenarios donde se acepta la perdida ocasional de mensajes
 - Ejemplo: sensores a alta frecuencia, camaras y video



MQTT: Calidad de servicio

En ingles: Quality of Service (QoS) → QoS: 0, QoS: 1, QoS: 2

- QoS 1: “entregado por lo menos una vez”
 - El mensaje se publica al broker y el broker envia una confirmacion de recepcion
 - Si no se recibe la confirmacion, el mensaje se reenvia
 - Entrega garantizada
 - Utilizado en escenarios donde se requiera una entrega garantizada y se tolere la duplicacion ocasional de mensajes
 - Ejemplo: sensores de temperatura

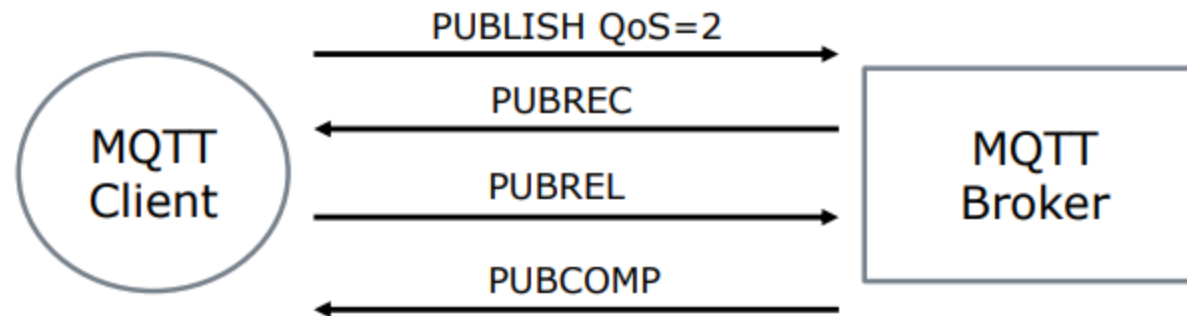


- Mensaje PUBACK
 - packetID: 2

MQTT: Calidad de servicio

En ingles: Quality of Service (QoS) → QoS: 0, QoS: 1, QoS: 2

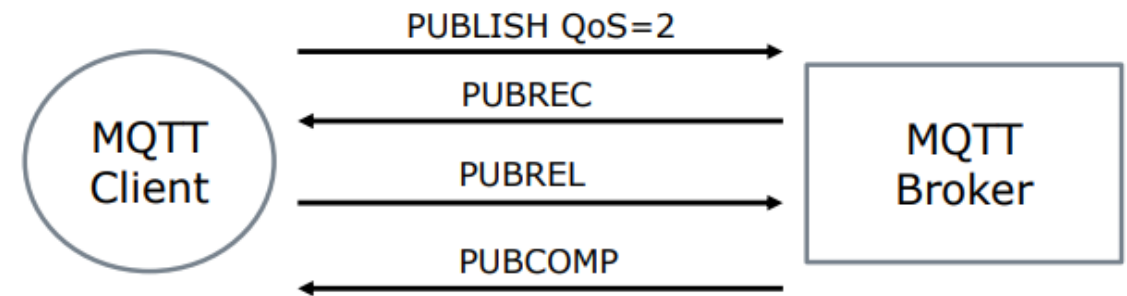
- QoS 2: “entregado exactamente una vez”
 - Garantiza la entrega exacta de un solo mensaje
 - Evita duplicacion y perdida de mensajes
 - Se utiliza en escenarios criticos donde la perdida de mensajes no es aceptable
 - Ejemplos: sistemas de control (plantas industriales), transacciones financieras (pago con el celular)



MQTT: Calidad de servicio

En ingles: Quality of Service (QoS) → QoS: 0, QoS: 1, **QoS: 2**

- QoS 2: consiste en 4 pasos
 - **PUBLISH:**
 - De cliente a broker,
 - Contiene el mensaje con nuestra informacion
 - **PUBREC** (publish received):
 - De broker a cliente
 - El broker guarda el packetID para evitar duplicados
 - **PUBREL** (publish released):
 - De cliente a broker
 - El cliente descarta el paquete inicial y confirma el PUBREC
 - **PUBCOMP** (publish completed):
 - De broker a cliente
 - El broker confirma la recepcion del PUBREL y completa la operacion



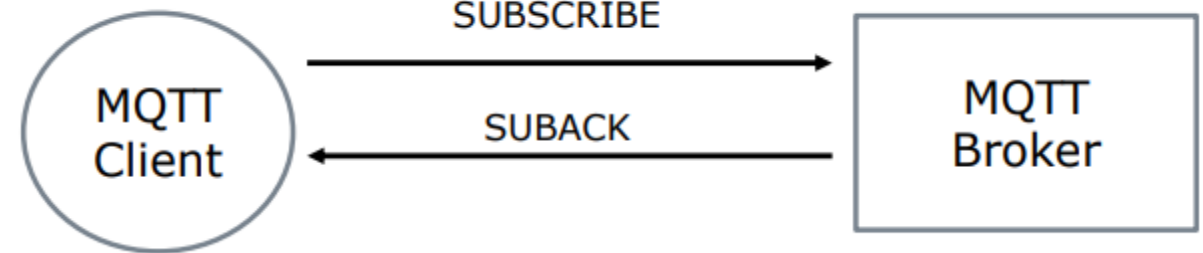
PUBREC, PUBREL y PUBCOMP
• packetID: 2

MQTT: Suscripcion

Mensaje SUSCRIBE

- packetId 2
- QoS1 0
- Topic1 "habitacion/temperatura/1"
- QoS2 1
- Topic2 "cocina/temperatura/2"
-

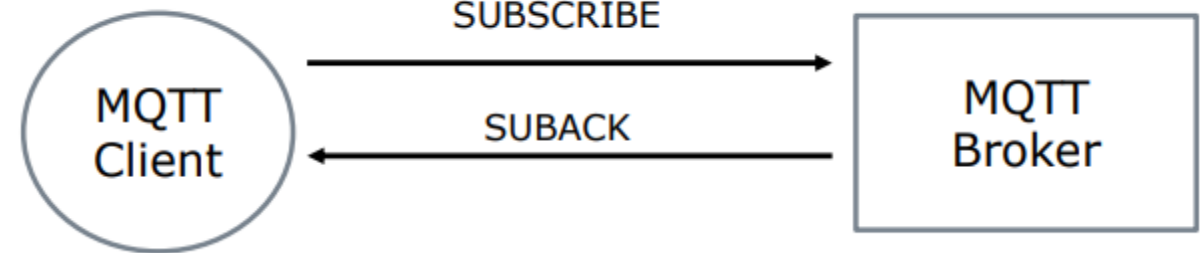
Una pareja de QoS/Topic
por suscripcion



MQTT: Suscripcion

Mensaje SUBACK

- packetId 2
- returnCode1 "habitacion/temperatura/1"
- returnCode2 "cocina/temperatura/2"



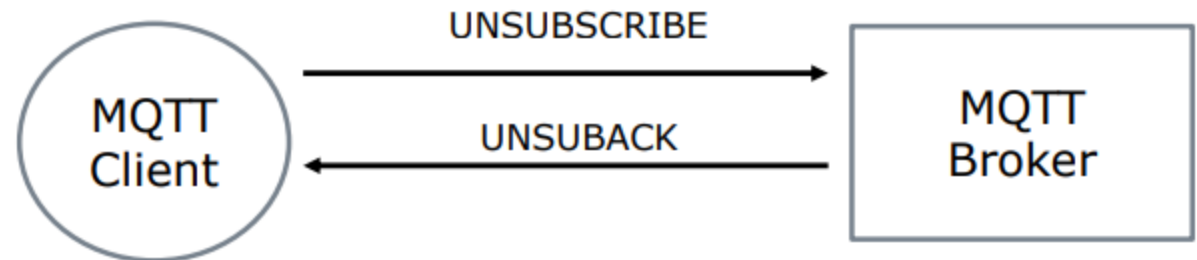
MQTT: Cancelar la suscripcion

Mensaje UNSUBSCRIBE

- packetId 2
- returnCode1 "habitacion/temperatura/1"
- returnCode2 "cocina/temperatura/2"

Mensaje UNSUBACK

- packetId 2



MQTT: Sesiones persistentes

- En el modo de operación predeterminado
 - Cuando el cliente se desconecta, todos los estados relacionados con el cliente en el broker se eliminan
 - (lista de suscripciones, mensajes pendientes con niveles de calidad de servicio (QoS), etc.).
- En sesiones persistentes, tanto el cliente como el broker mantienen una sesión:
 - En el Broker:
 - Existencia de una sesión, incluso si no hay suscripciones.
 - Todas las suscripciones.
 - Todos los mensajes en el flujo con niveles de calidad de servicio (QoS) 1 o 2 que no han sido confirmados por el cliente.
 - Todos los nuevos mensajes QoS 1 o 2 que el cliente se perdió mientras estuvo desconectado.
 - Todos los mensajes QoS 2 recibidos que aún no han sido confirmados al cliente.
 - En el Cliente:
 - Todos los mensajes en un flujo con QoS 1 o 2 que no han sido confirmados por el broker.
 - Todos los mensajes QoS 2 recibidos que aún no han sido confirmados al broker.

MQTT: Mensajes retenidos

Problema: publicar y suscribir son procesos asincronos

- Un cliente que se suscribe a un tema
 - puede no recibir ningún mensaje en ese tema hasta que algún otro cliente lo publique.
- Los mensajes retenidos son mensajes PUBLISH con el indicador de retención
 - (retained flag) establecido en uno.
- El broker almacena localmente el mensaje retenido
 - y lo envía a cualquier otro cliente que se suscriba a un patrón de tema que coincida con el del mensaje retenido.

MQTT: Mensaje de ultima voluntad

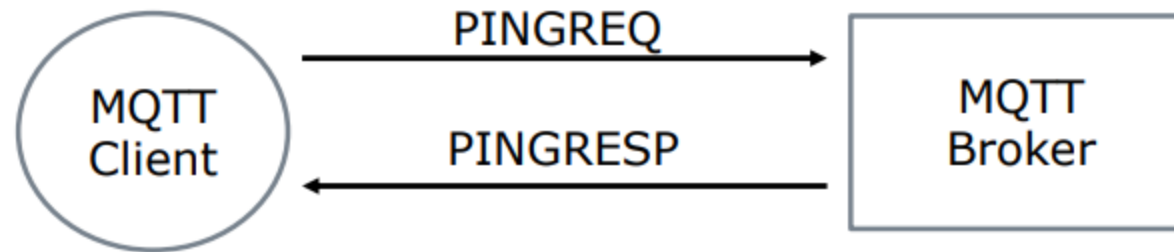
En ingles: Last Will and Testament (LWT)

- Notifica a otros clientes sobre una desconexión brusca por parte de un cliente específico.
- Cada cliente puede especificar su mensaje de testamento al conectarse a un broker.
- El broker almacenará el mensaje hasta que detecte la desconexión brusca del cliente.
- El broker enviará el mensaje a todos los clientes suscritos en el tema específico.
- El mensaje de LWT almacenado se descartará si un cliente se desconecta de manera adecuada enviando un mensaje DISCONNECT.



MQTT: Keepalive (mantenerse activo)

- Es responsabilidad del cliente mantener activa la conexión MQTT.
- Al vencer el tiempo de mantenimiento activo (keepalive), si no ha ocurrido ninguna otra interacción con el broker, el cliente "pingea" al broker, que le responde con un "ping" de vuelta.



- PINGREQ y PINGRESP tienen payload nula