MQTT

Protocoles IP utilisés par l'IoT

- **HTTP (REST)**
- **CoAP (REST)**
- **MQTT**

HTTP

HyperText Transfer Protocol

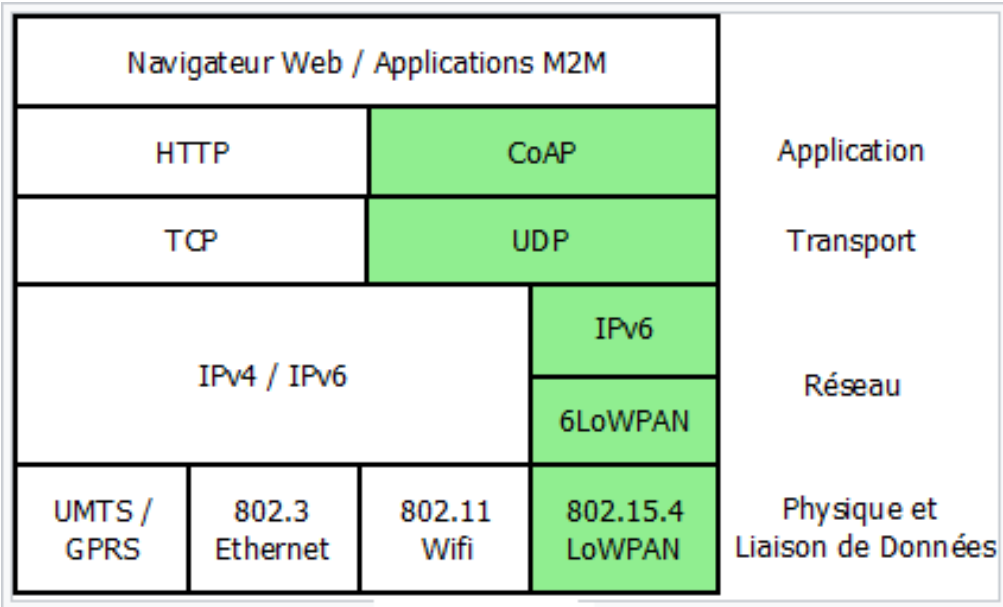
- protocole de communication client-serveur (pull)
- protocole de la couche application
- utilise généralement le protocole TCP comme couche de transport
- pour l'IoT, le style d'architecture REST est utilisé

CoAP

COnstrained Application Protocol

- protocole de communication client-serveur (comme HTTP, pull)
- protocole de la couche application
- utilise le protocole UDP comme couche de transport
- protocole dédié pour l'IoT (REST est utilisé)

CoAP vs HTTP



- CoAP utilise les méthodes similaires au protocole HTTP mais optimisées pour les réseaux de capteurs sans fil. CoAP est RESTful

Architecture REST

REpresentational State Tranfert

- protocole de communication client-serveur
- modèle sans état (sur le server):
  - La requête envoyée vers le serveur contient toutes les informations nécessaires pour la traiter
  - Minimise les ressources systèmes (pas de sessions d'état)
- les ressources sont manipulées à travers des formats de représentation

Architecture REST

- Ressources (identifiant)
  - Identifié par un URI (Uniform Resource Identifier)
  - Par ex. : http://localhost:8080/librairy
- Méthodes (verbes)
  - Manipule la ressource
  - Méthodes HTTP: GET, POST, PUT et DELETE
- Représentation (vue sur l'état)
  - Donne une vue sur l'état de la ressource
  - Informations transférées entre le client et le serveur
  - Par ex. : Text, XML, JSON, etc.,

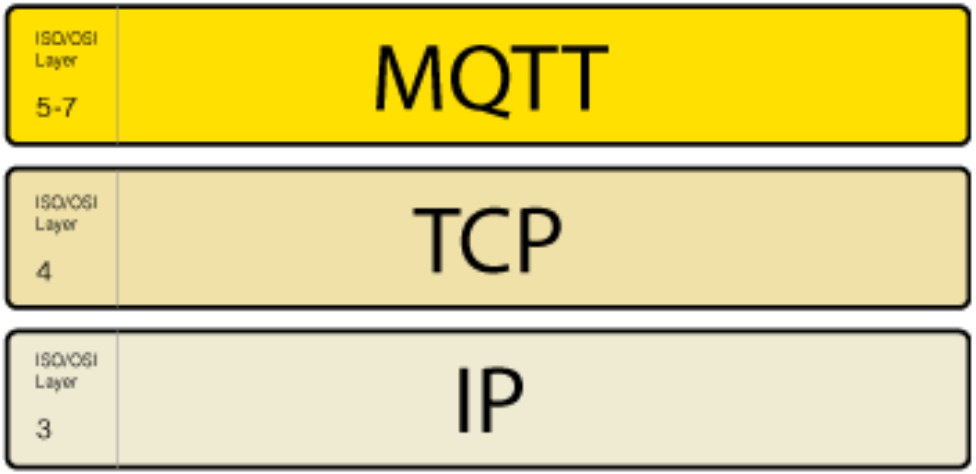
Architecture REST

- **Twitter**:
  https://dev.twitter.com/rest/public
- **Facebook**:
  https://developers.facebook.com/docs/atlas-apis
- **Amazon** offre des services REST (par ex. solution de stockage S3)
  http://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html
- **Google Glass API ("Mirror API")**
  https://youtu.be/JpWmGX55a40
- **Tesla Model S utilise une API REST**
  http://docs.timdorr.apiary.io/#reference/vehicles/state-and-settings
- **Google Maps:**
  https://developers.google.com/maps/web-services/
  http://maps.googleapis.com/maps/api/geocode/json?address=lecco

MQTT

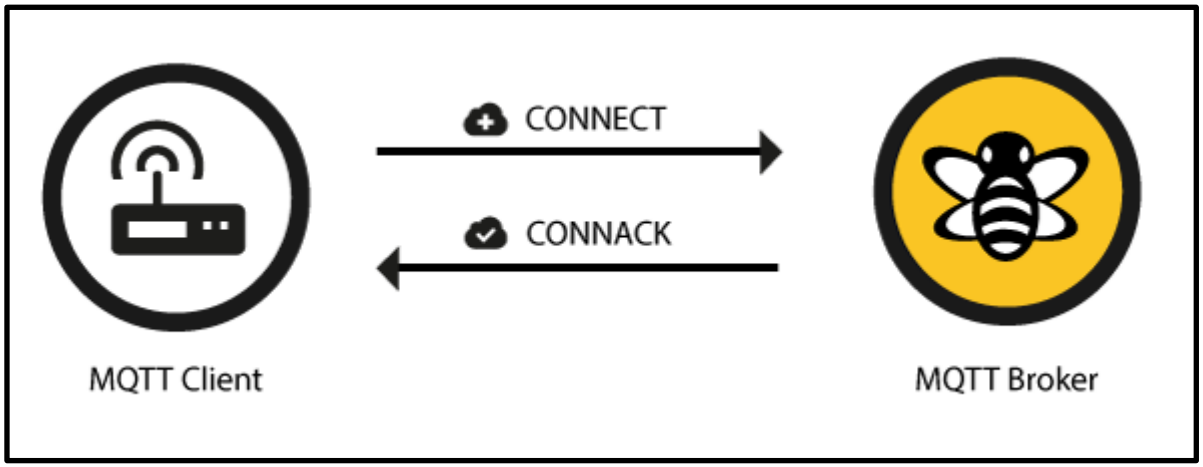
Message Queuing Telemetry Transport

- protocole de communication léger de type «publish-subscribe» pour M2M



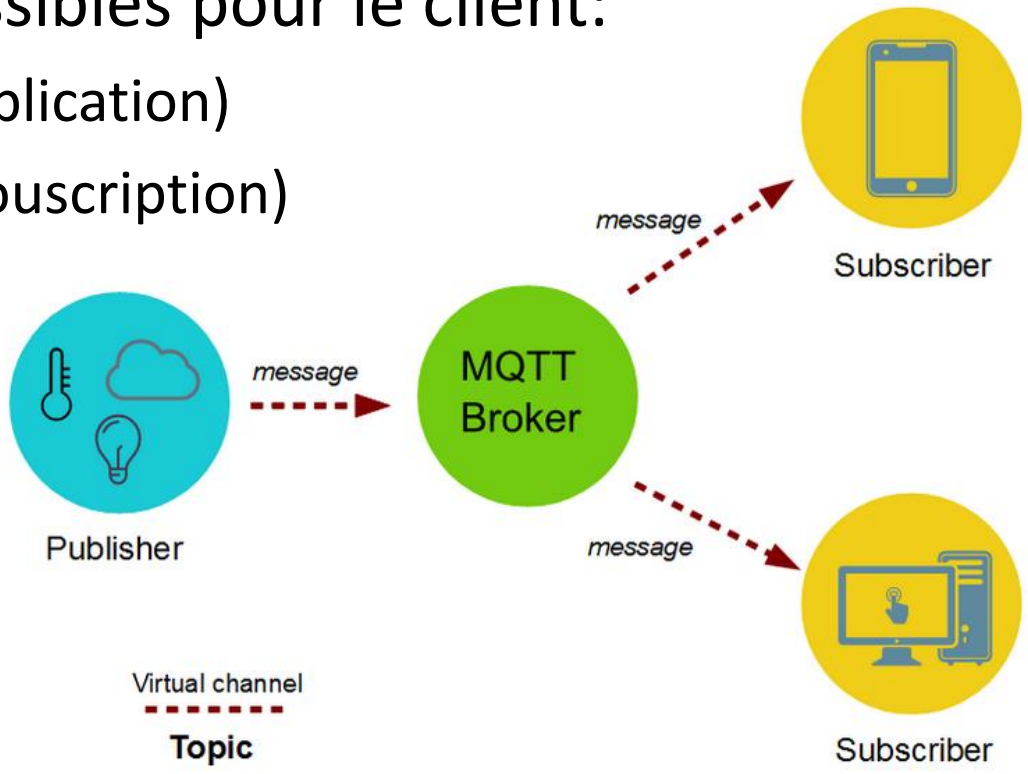
MQTT

- Une connexion MQTT se fait toujours entre un client et un *broker* (push)



MQTT

- Deux rôles possibles pour le client:
  - Publisher (publication)
  - Subscriber (souscription)



  - **Attention, un client MQTT peut être *publisher* et *subscriber***

MQTT - Topics

- Un *topic* est en quelques sortes un chemin d'accès à une ressource

  Quelques exemples :

  sensor/1/temperature

  sensor/1/humidite

  sensor/2/temperature

  sensor/2/humidite

  sensor/1/led

  sensor/2/led

MQTT - actions

CONNECTION

établi une connexion avec le *broker*

DECONNECTION

ferme une connexion avec le *broker*

SUBSCRIBE

demande une souscription au broker pour un *topic*

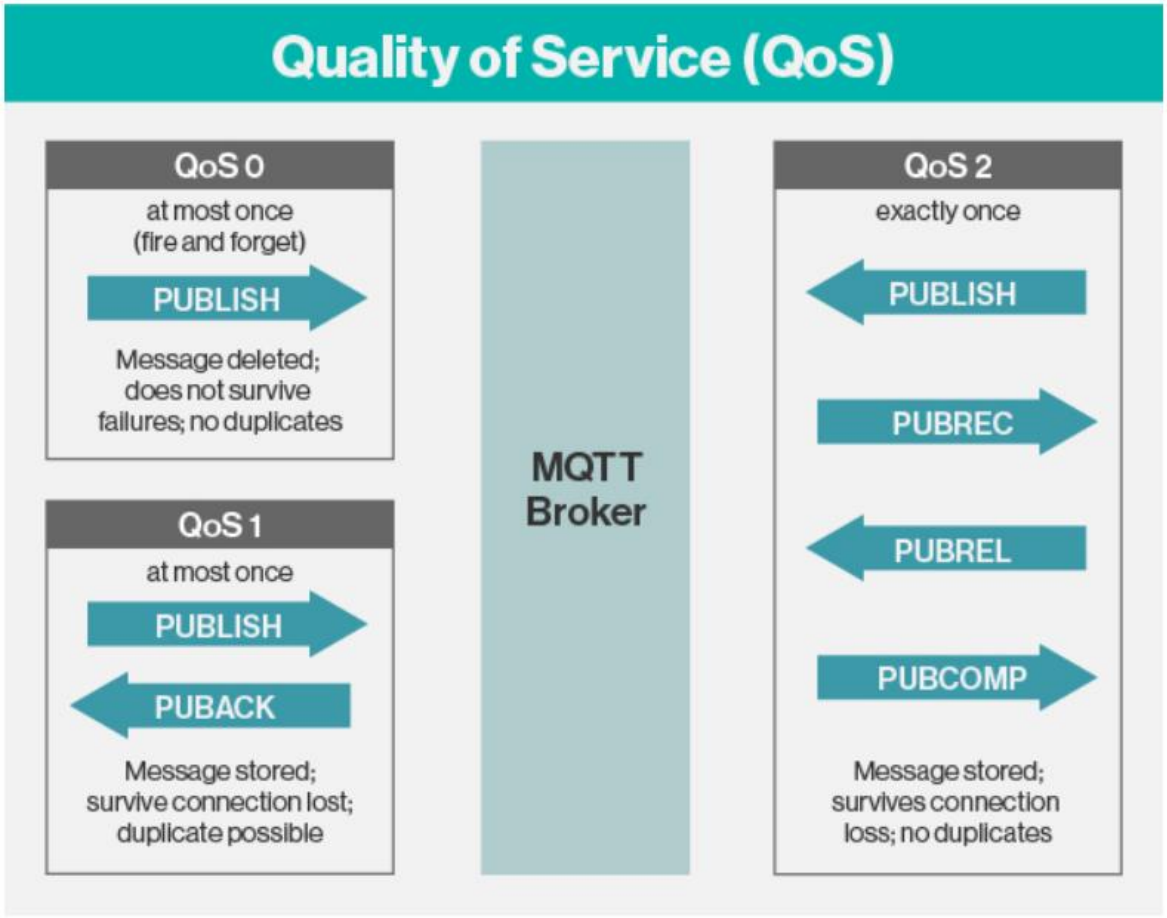
UNSUBSCRIBE

demande une «désouscription» au broker pour un *topic*

PUBLISH

mise à jour d'un *topic* sur le *broker*

MQTT - QoS



MQTT - Messages

MQTT Message	Description
CONNECT	Client request to connect to server
CONNACK	Connect acknowledgement
PUBLISH	Publish message
PUBACK	Publish acknowledgement
PUBREC	Publish received
PUBREL	Publish release
PUBCOMP	Publish complete
SUBSCRIBE	Client subscribe request
SUBACK	Subscribe acknowledgement
UNSUBSCRIBE	Unsubscribe request
UNSUBACK	Unsubscribe acknowledgement
PINGREQ	PING request
PINGRESP	PING response
DISCONNECT	Client is disconnecting

MQTT – Broker (local)

- Mosquitto (https://mosquitto.org/)
- Mosca (https://github.com/mcollina/mosca)
- Emqttd (https://github.com/emqx/emqx)
- Python Test Broker (https://github.com/eclipse/paho.mqtt.testing/tree/master/interoperability)
- VerneMQ (https://vernemq.com/intro/index.html)

MQTT – Broker (cloud)

- AWS (https://aws.amazon.com/fr/)
- flespi (https://flespi.com/mqtt-broker)
- ThingStudio (http://www.thingstud.io/)
- cloudMQTT (https://www.cloudmqtt.com/)
- Erlang/EMQX (https://www.emqx.io/)
- HiveMQ (https://www.hivemq.com/)

Conclusion