Application level of IoT networks -- Part 2

Rafik Zitouni

ECE Paris

rafik.zitouni@ece.fr

ECE PARIS
ÉCOLE D'INGÉNIEURS

Outline

- IPV6 and 6LowPAN
- CoAP protocol
- Quality of Service (QoS)
  - MQTT
  - DDS, AMQP, XMPP .etc

MQTT Message Queuing Telemetry Transport

Characteristics (**OASIS standard 2013)** http://public.dhe.ibm.com/software/dw/webservices/ws-mqtt/mqtt-v3r1.html)

- Publish / Subscribe message pattern → messaging distribution, applications decoupling;

- Topics: series of topic levels separated by **'/'**

Example: house/alalarm/status
house/camera/capture

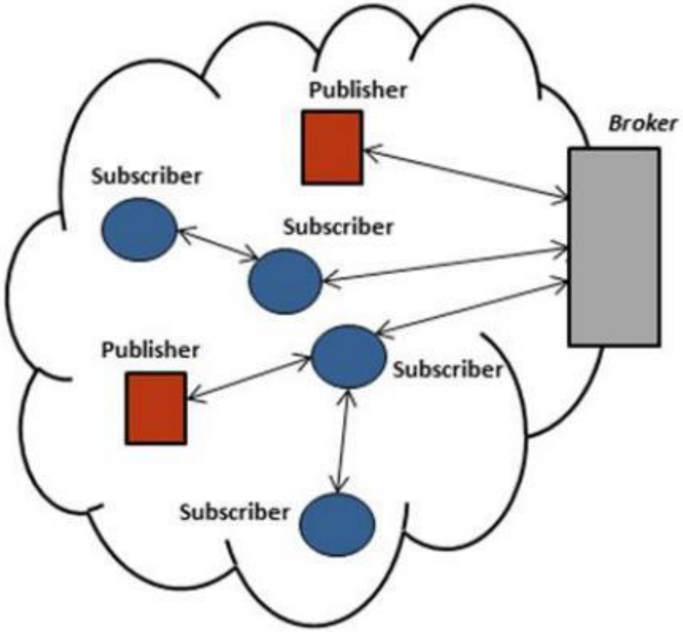
- Messages:

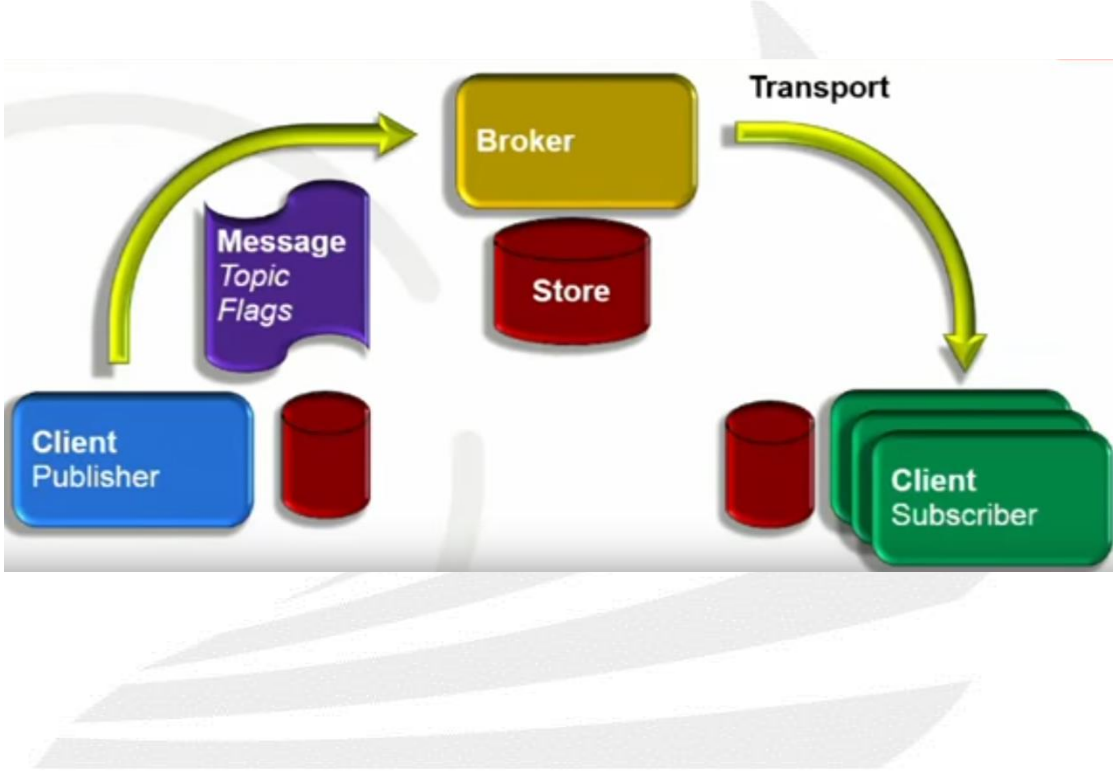
Examples: FFFF1111000, 456456.jpg

- **3 QoS levels:** At Most Once, At Least Once, Exactly Once;
- Small Transport Overhead, minimal messages exchanges;
- Assumes the use of the **TCP/IP** protocol stack;
- **Data agnostic**

MQTT Message Queuing Telemetry Transport

MQTT base components

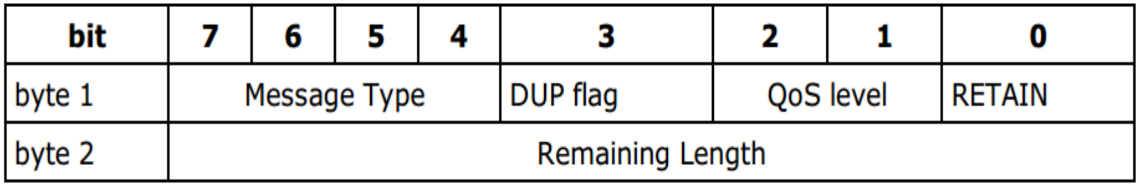




MQTT Message Queuing Telemetry Transport

Message Format

- **Fixed Header**
  - Big Endian Byte Order



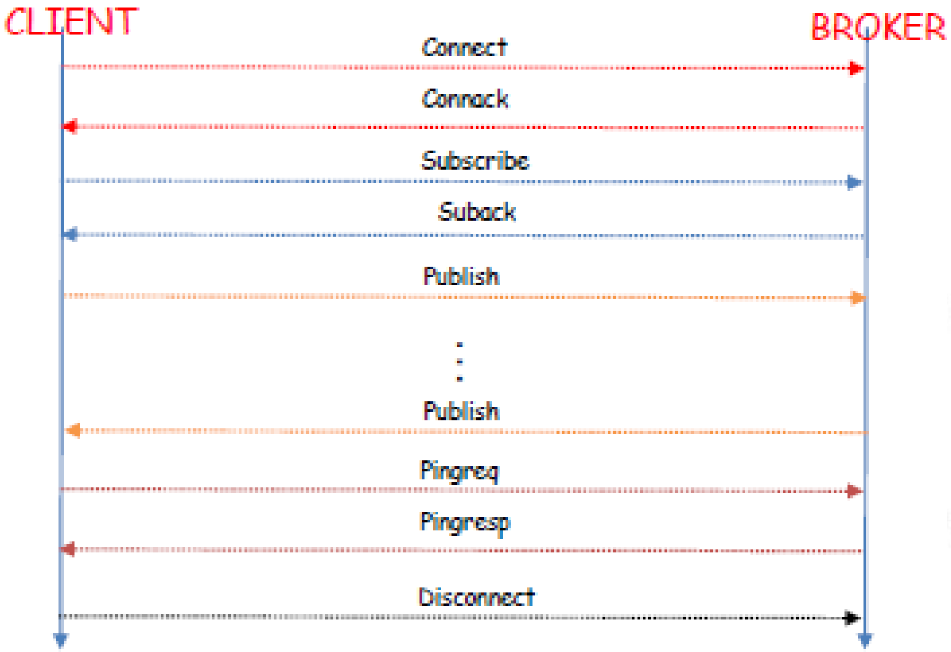
bit	7	6	5	4	3	2	1	0
byte 1	Message Type				DUP flag	QoS level		RETAIN
byte 2	Remaining Length							

- **Variable Header**
  - Topic
- **Payload**
  - Byte Buffer
    - Json, XML

MQTT Message Queuing Telemetry Transport

Message Types

1) CONNECT
2) CONNACK
3) PUBLISH
4) PUBACK
5) PUBREC
6) PUBREL
7) PUBCOMP
8) SUBSCRIBE
9) SUBACK
10) UNSUBSCRIBE
11) UNSUBACK
12) PINGREQ
13) PINGRESP
14) DISCONNECT



MQTT Message Queuing Telemetry Transport

Subscription Types

- **“Non Durable” subscription**
  - CONNECT
  - SUBSCRIBE → TOPIC
  - …
  - UNSUBSCRIBE → TOPIC
  - DISCONNECT

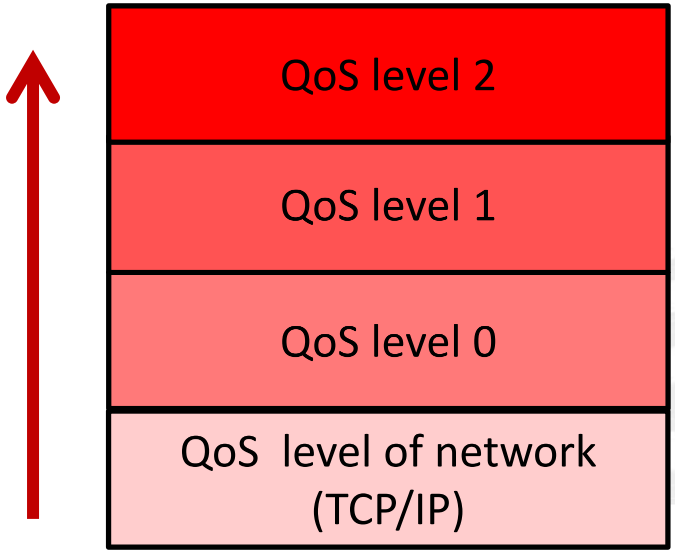
- **“Durable” subscription**
  - CONNECT
  - SUBSCRIBE → TOPIC
  - DISCONNECT
  - …
  - CONNECT
  - Receive all messages published to the TOPIC since the last DISCONNECT
  - …….

MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

- What is QoS ?
  - Low Quality means Low traffic and lower reliability
  - High Quality means More traffic and reliable

- 3 Services levels defined



Increasing level of QoS

QoS level 2

Exactly-once delivery

QoS level 1

At least once delivery

QoS level 0

At most-once delivery (=best effort)

QoS level of network (TCP/IP)

best effort

MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

QoS 0 messaging (Fire and forget)

- Typical Application:

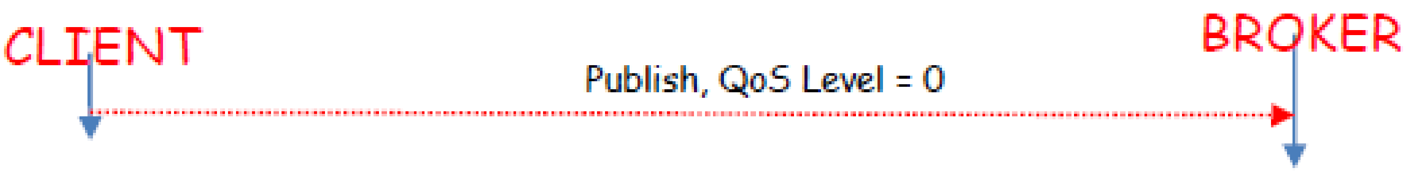
  Publisher: GPS data source publishes GPS coordinates every second

  Subscriber: Navigation System

Messages loss data doesn’t cause problems

GPS coordinates are note constantly available anyways

We don’t considerate autonomous cars positioning



MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

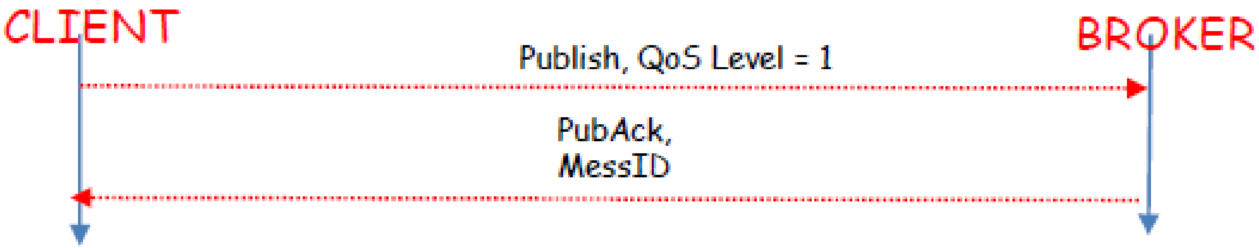
QoS 1 messaging

- Reception of the message is acknowledged
- Simple mechanism ensures at least once delivery

➢ Typical Application:

For tour tracking, an end unit (On Board Unit) sends collection of GPS data to datacenters

Messages loss is not accepted



MQTT Message Queuing Telemetry Transport

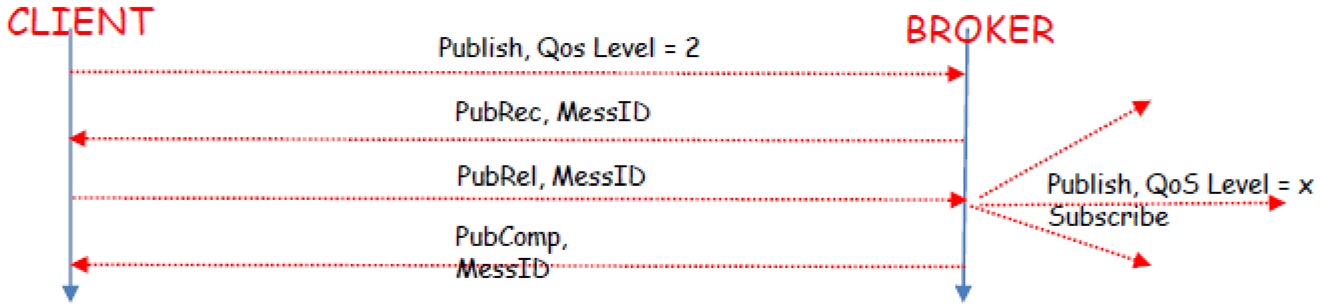
Quality of Service (QoS)

QoS 2 messaging

- Reception of the message is acknowledged
- Double message exchange ensures exactly one delivery of a message

➢ Typical Application:

Oil pipeline surveillance, one message by event

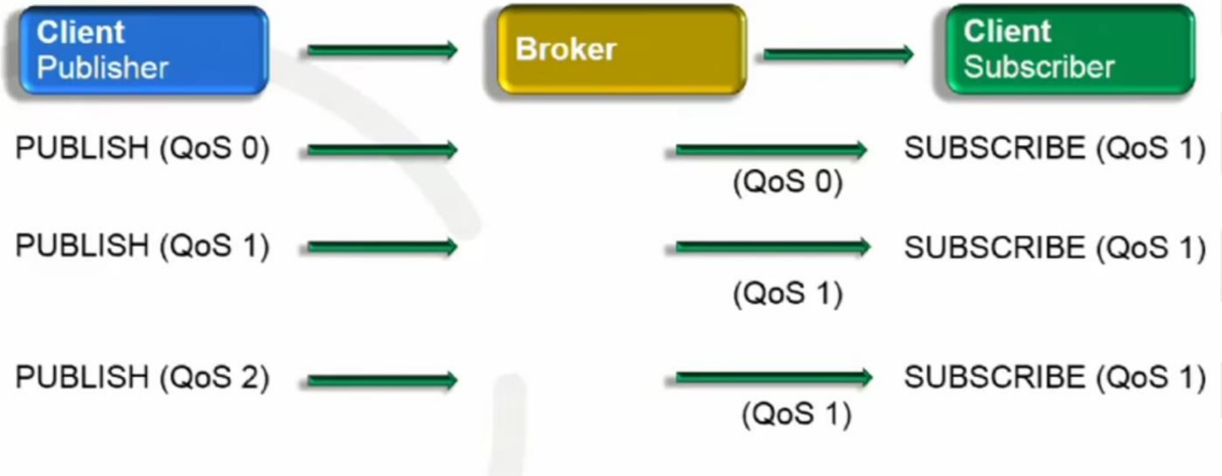


Cost of QoS ?

MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

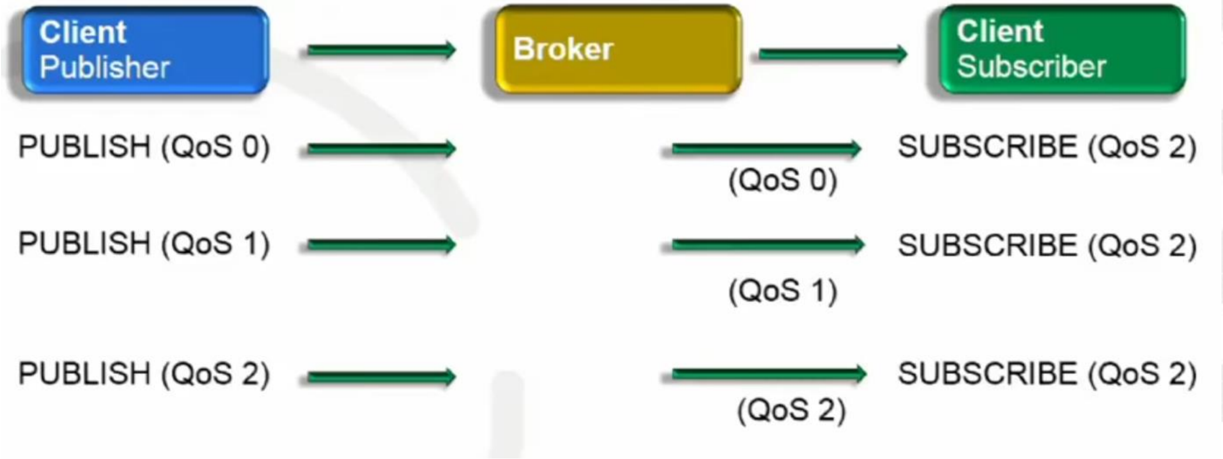
- If QoS of PUBLISH and SUBSCRIBE are different for a particular topic
  - QoS of PUBLISH defines **the best possible QoS** level for the end-to-end delivery
  - QoS of SUBSCRIBE defines the **maximum** level to be used on the “second leg”
  - “Second leg” QoS level of PUBLISH request may be downgraded **(but never upgraded)**



MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

- If QoS of PUBLISH and SUBSCRIBE are different for a particular topic
  - QoS of PUBLISH defines **the best possible QoS** level for the end-to-end delivery
  - QoS of SUBSCRIBE defines the **maximum** level to be used on the "second leg"
  - "Second leg" QoS level of PUBLISH request may be downgraded **(but never upgraded)**



MQTT Message Queuing Telemetry Transport

Quality of Service (QoS)

Last Will

- **Will message** is published by the Borker in case of a communication problem with a particular client
  - Broken Network Connection
  - Client fails to response to broker messages
- **Will message** will not be sent on regular DISCONNECT

MQTT Message Queuing Telemetry Transport

Topics

How a client speaks to others by subscribing to a particular topic?

- Hierarchical identifiers for messages
  - UTF strings
  - At least 1 character

- Each message has exactly one topic
- Topics can be structured with a separator “/” to form a topic tree

Example:
house/alarm/status → status of your alarm on/off/activated
house/alarm/zone
House/+/temperature → All temperature for all /house/xxxx/ topics
house/room1/# → Multi-level topics, only at the end

MQTT Message Queuing Telemetry Transport

MQTT implementations

MQTT Client Implementations:

- *WebSphere MQ Telemetry Client (C,Java)*
- *Eclipse Paho(C,Java,Python, Lua)*

MQTT Broker Implementations:

- *WebSphere MQ Broker (C, Java);*
- *Really Small Message Broker, RSMB (C);*
- ***Mosquitto(JMS)**;*

Utility for MQTT:

- *Eclipse Paho(Eclipse);*
- *WMQTT (Java application);*

Related Technology Proposals

- **MoquetteMQTT**: creation of a simple and small self contained Java Implementation of a client broker;

MQTT Message Queuing Telemetry Transport

MQTT implementations

Example:

Install mosquitto package in your ubuntu VM:

```
$ sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
$ sudo apt-get update
$ sudo apt-add-repository ppa:mosquitto-dev/mosquitto-ppa
```

run on two terminals the following publish and subscribe actions:

1) Ensure that your Mosquetto broker is in run.

```
 $ sudo lsof -i TCP:1883
```

2) Start the command line subscriber:

```
 $ mosquitto_sub -v -t 'test/topic'
```

3) Publish test message with the command line publisher:

```
 $ mosquitto_pub -t 'test/topic' -m 'helloWorld'
```

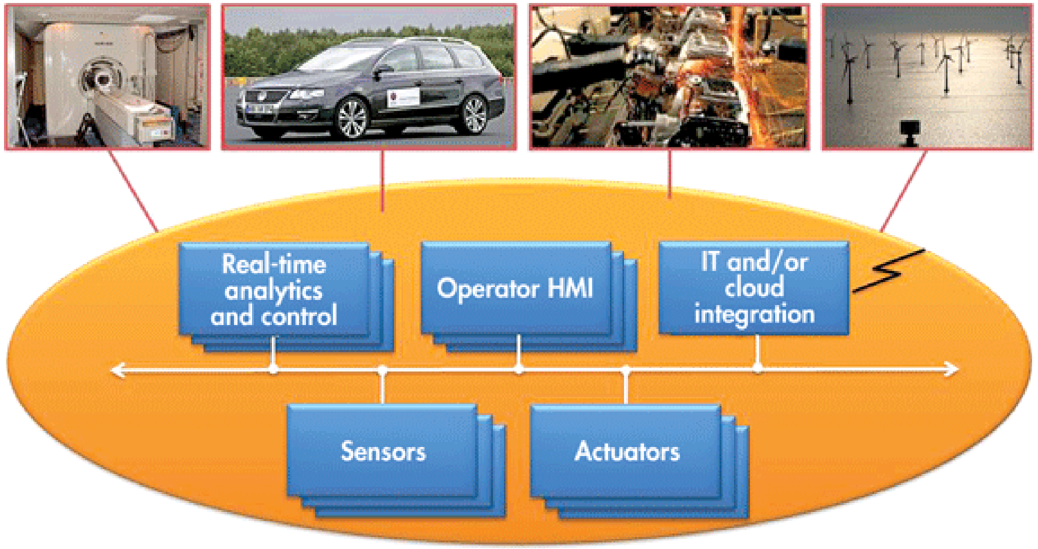
CoAP vs MQTT

CoAP	MQTT
- One to many communication protocol - Document centric (http like) - UDP - Adapted to state transfer and not purely event-based - Tunnelling or Port Forwarding can be used to allow CoAP in NAT environments (IPv4). with IPv6 no problems. - Reliability mechanism is based on NON/CON messages	- Many-to-Many Communication Protocol - Data centric - Adapted to long lived outgoing TCP connection to a broker - Tunnelling or Port Forwarding can be used to allow CoAP in NAT environments (IPv4). with IPv6 no problems. - 3 QoS levels

DDS Data Distribution Service

DDS addresses Data distribution requirements of mission-critical systems. Very similar with MQTT, often confused (**Thales**).

- Publish/Subscribe message pattern as MQTT
- Optimized design for embedded or constrained devices as MQTT



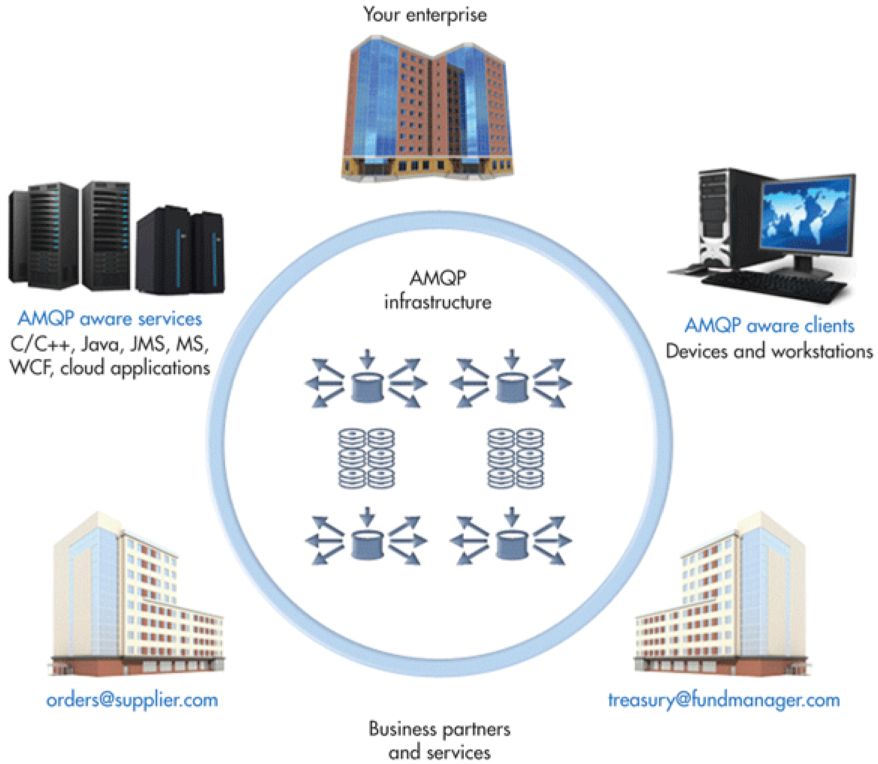
Typical Applications

- Financial trading, air-traffic control, smart grid management.

AMQP Advanced Message Queuing Protocol

AMQP suitable for queues.
It sends transactional messages between servers. As a message-centric middleware coming from the banking industry, it can process thousands of reliable queued transactions.

- No Lose of Messages (TCP)
- 3QoS Level, as MQTT
- A single node can be a client or a server, while in MQTT this is impossible.

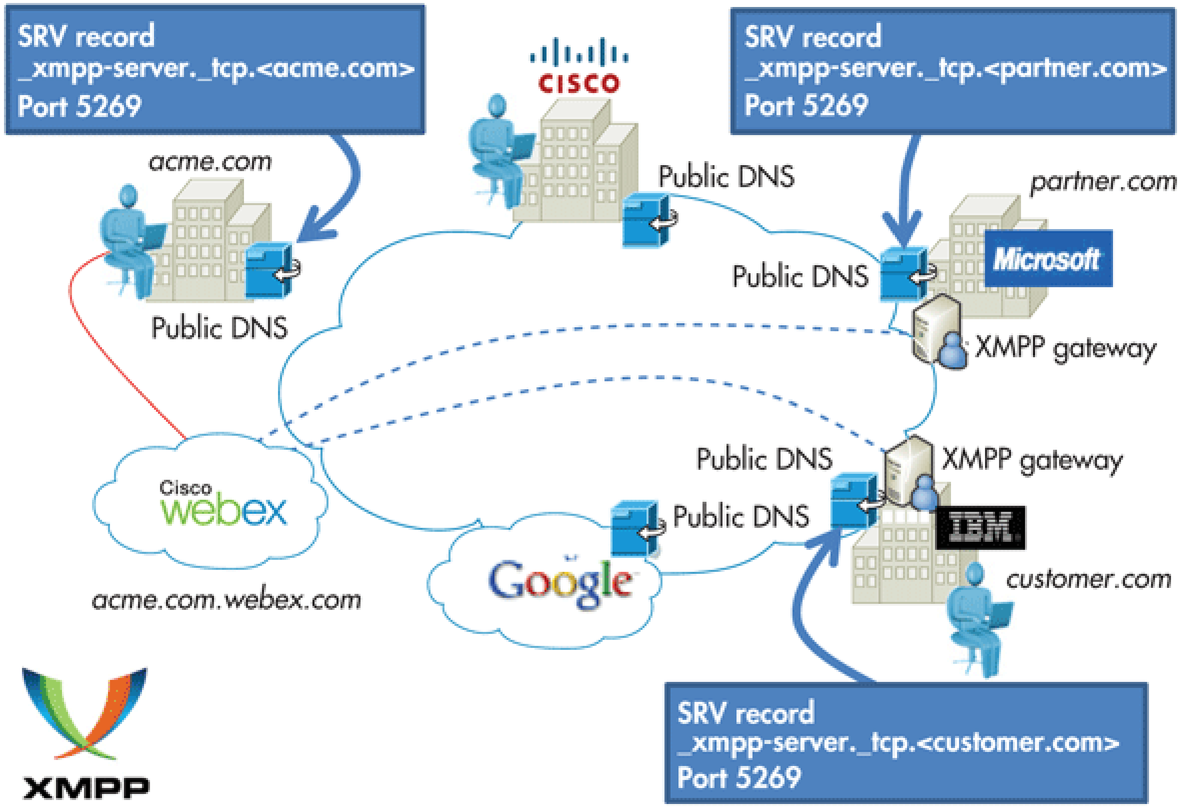


XMPP Extensible Messaging And Presence Protocol

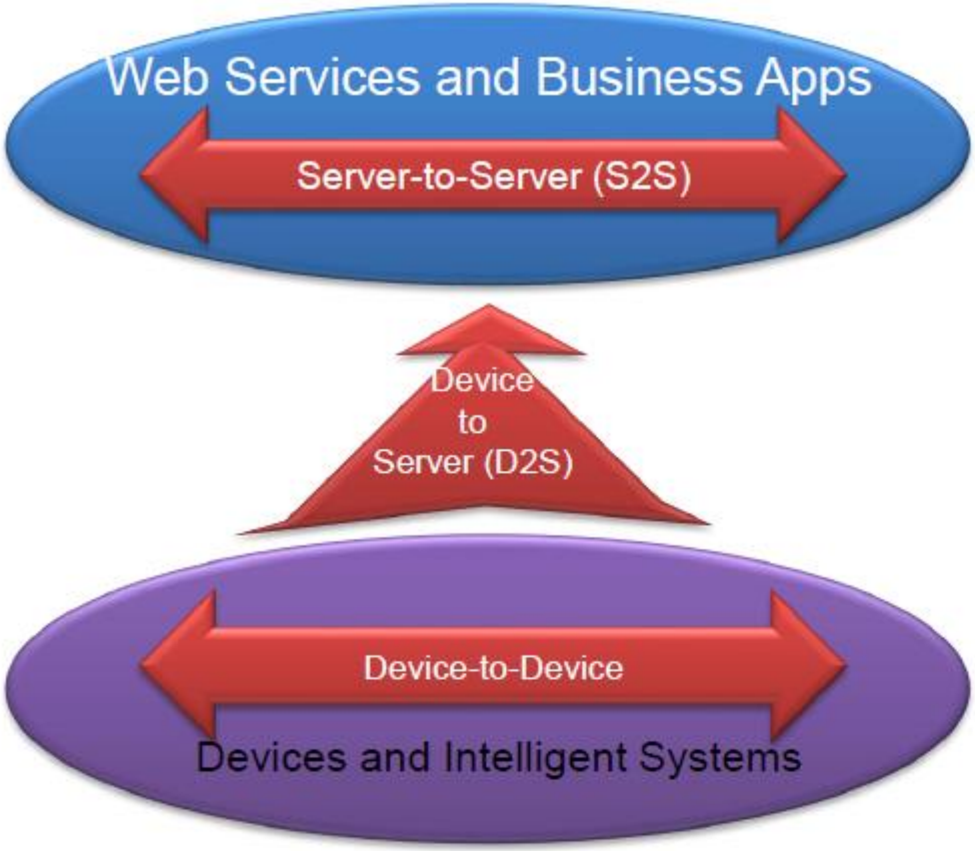
XMPP

It is a way to connect people with people through text messages.

- XMPP uses the XML text format as its native type.
- Its key strength is aname@domain.com addressing scheme that helps connect the needles in the Internet.



Levels of service and protocol


Web Services and Business Apps
Server-to-Server (S2S)
Device
to
Server (D2S)
Device-to-Device
Devices and Intelligent Systems

- S2S
  - Biz intelligence
  - Centralized/ESB
  - ~100ms
  - **MQ/AMQP** ?
- D2S
  - Collect data
  - ~10ms
  - MQTT/CoAP/**XMPP ?**
- D2D
  - Control, distribute
  - Databus
  - ~.01ms
  - **DDS ?**

References

Constrained Application Protocol (CoAP) Tutorial
https://www.youtube.com/watch?v=4bSr5x5gKvA

Home Automation with Node.js and MQTT
https://www.youtube.com/watch?v=80DxfDmoZUI

Using MQTT in Real-World M2M Communication
https://www.youtube.com/watch?v=r6HEQVhgnP8

http://electronicdesign.com/iot/understanding-protocols-behind-internet-things

Contiki 6LoWPAN Quick Guide with nucleo boards (X-NUCLEO-IDS01A4, X-NUCLEO-IDS01A5)

Lauree Magistrali, " Application Layer Solutions for the Internet of Things"

Zach Shelby, "ARM IoT Tutorial"