MQTT

MQTT is a machine-to-machine (M2M)/"Internet of Things" connectivity protocol.

WHAT? WHERE? WHY? POWER? TRAFFIC? RESPONSE?

LabWare Internal April 2019

By John M McIntosh

MQTT – Old – proven – robust - simple

Created by Dr. Andy Stanford-Clark of IBM and Arlen Nipper of Arcom -- now Eurotech -- in 1999 as a cost-effective, reliable way to connect monitoring devices used in the oil and gas industries to remote enterprise servers.

IE North Sea oil industry.

MQTT – Standard

OASIS is a nonprofit consortium that drives the development, convergence and adoption of open standards for the global information society.

OASIS is pleased to announce that MQTT Version 3.1.1 from the OASIS Message Queuing Telemetry Transport (MQTT) TC has been approved by the membership as an OASIS Standard. Now working on Version 4.x aka 5

MQTT – Power/Traffic & FaceBook?

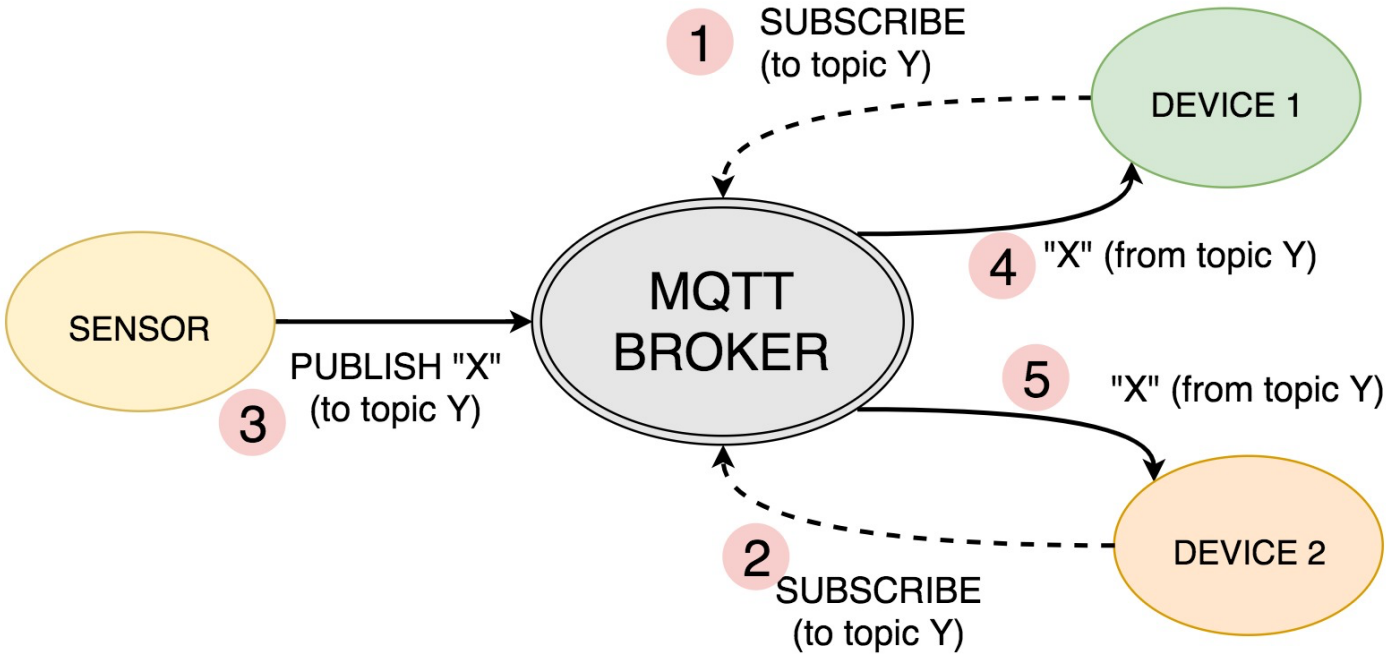
Ask Dr Google.

Up to 22% more energy efficient & **15% faster than HTTP.**

Does not depend on if the connection type is 3G or WiFi.

It is the transport layer for FaceBooks Apps, see github.

MQTT – Simplistic (Subscribe & Publish: 1 to many)



https://www.norwegiancreations.com/2017/07/mqtt-what-is-it-and-how-can-you-use-it/

MQTT – QOS – 3 levels

Persistent store - Restartable

0 Send and forget, I don’t care. (might not get delivered)

1 Send and ensure delivery happens at least once.

2 Send and ensure delivery happens only once.

If I send with ensured delivery (2) with persistent store, then the application can do send and not take on the responsibility of ensure the message was received and processed.

Kinda like a database COMMIT.

MQTT – Hey various Smalltalk Projects

Snatched Tim Rowledge's work for Scratch on the PI.

Client, got that, mmm **Broker?** Nope.

Fine, refactor and build as a client & data broker solution in pure VSE Smalltalk.

Data Broker in Smalltalk? Why?

A standalone MQTT Data broker would require customer verification validation step.

But we can add the MQTT data broker to a VSE image as an SLL upgrade!
No questions asked.

MQTT- Simple? Oh sure a month or 4…

Yes, the standard is simple? Just grind thru the specs, keep a spreadsheet…

#	MQTT Standard and our solution & work notes
[MQTT-1.5.3-1]	The character data in a UTF-8 encoded string MUST be well-formed UTF-8 as defined by the Unicode specification [Unicode] and restated in RFC 3629 [RFC3629]. In particular this data MUST NOT include encodings of code points between U+D800 and U+DFFF. If a Server or Client receives a Control Packet containing ill-formed UTF-8 it MUST close the Network Connection.
[MQTT-1.5.3-2]	check for subscribe, unsubscribe topic A UTF-8 encoded string MUST NOT include an encoding of the null character U+0000. If a receiver (Server or Client) receives a Control Packet containing U+0000 it MUST close the Network Connection.
[MQTT-1.5.3-3]	A UTF-8 encoded sequence 0xEF 0xBB 0xBF is always to be interpreted to mean U+FEFF ("ZERO WIDTH NO-BREAK SPACE") wherever it appears in a string and MUST NOT be skipped over or stripped off by a packet receiver.
[MQTT-2.2.2-1]	Where a flag bit is marked as “Reserved” in Table 2.2 - Flag Bits, it is reserved for future use and MUST be set to the value listed in that table.
[MQTT-2.2.2-2]	If invalid flags are received, the receiver MUST close the Network Connection.
[MQTT-2.3.1-1]	**MQTTTransportLayer>>newPackID** SUBSCRIBE, UNSUBSCRIBE, and PUBLISH (in cases where QoS > 0) Control Packets MUST contain a non-zero 16-bit Packet Identifier.
[MQTT-2.3.1-2]	**MQTTTransportLayer>>badPacketID** & **newPacketID** Each time a Client sends a new packet of one of these types it MUST assign it a currently unused Packet Identifier.
[MQTT-2.3.1-3]	**MQTTPacketPublish>>acknowledgement, MQTTPendingPubCompJob>>resendFor:ifNeededAtTime: MQTTPacketPubRel>>acknowledgement, MQTTTransportlayerServer>>handleSubscribePacket:** If a Client re-sends a particular Control Packet, then it MUST use the same Packet Identifier in subsequent re-sends of that packet. The Packet Identifier becomes available for reuse after the Client has processed the corresponding acknowledgement packet. In the case of a QoS 1 PUBLISH this is the corresponding PUBACK; in the case of QO2 it is PUBCOMP. For SUBSCRIBE or UNSUBSCRIBE it is the corresponding SUBACK or UNSUBACK.
[MQTT-2.3.1-4]	**MQTTTransportLayerServer>>handlePublishResponse:** The same conditions [MQTT-2.3.1-3] apply to a Server when it sends a PUBLISH with QoS >0.
[MQTT-2.3.1-5]	**MQTTServerInterface>>** **handlePublishedPacket: aPublishPacket** **usingTransport: aTransport**

MQTT – Testing

Ugh, those IBM MQTT compliance test scripts are rather nasty.

https://test.mosquitto.org

Want to test, oh sure, **24 GB a day of wonder data**.

Subscribe to wild card topic *, broadcast to a Linux Cmd line client or three… Run for weekend, anything explode?

Shrug, that looks like personal data? Sure why not make a public test server your production server for your iOS ap, who would know?

March 2018 – Finished? Ugh no -> 2019

Nope, as alluded to yesterday.

Subscribe and Publish seem simple, but another layer is needed for 1 to 1 peer or client/server abstraction.

Our solution:

EndPoint.

EndPoint - Hiding complexity & Receive

EndPoint is **behavior**.

Datum & tag. Datum is opaque, tag is meta-data.

>>receiveMessage: aByteArray tag: aTagString
"Async incoming message, *aByteArray* should be process here"
^ByteArray new "possible response here"

Note: Any exception thrown is bundled up as a stack trace and returned to the other EndPoint as Error message via the:

>>receiveError: aByteArray tag: aTagString

MQTT Sending Data

Three different cases:

>>**send:** aByteArray tag: aTagString “Async send, no reponse”

>>**sendSync:** aByteArray tag: aTagString “Sync send, will get response or error data”

>>**sendAsyncResponse:** aByteArray tag: aTagString
“Possible response via receiveMessage or receiveError”

Exploit a Promise class to do the right thing for **send**: or **sendAsyncResponse:**
Even **sendSync:**

Promise

Sure halt the JavaScript thread.

Get the data on the MQTT thread.

Signal the Semaphore

Resume the JavaScript thread.

Allow timeout for failure, IE response never receive in time.

Simple? A painful month of self doubt & go, no go crisis

MQTT – example - iPhone to VSE Image

Stop in VSE receive message logic.

Make the iPhone sendSync: time out.

See the XCode relationship

See the JavaScript relationship

See the VSE relationship.