



# Internet of Things in Banking

Gaetano Ziri  
[gaetano.ziri@aurigaspa.com](mailto:gaetano.ziri@aurigaspa.com)

**AURIGA**  
the banking e-evolution

THE **#NEXTGENBANK**



[www.aurigaspa.com](http://www.aurigaspa.com)



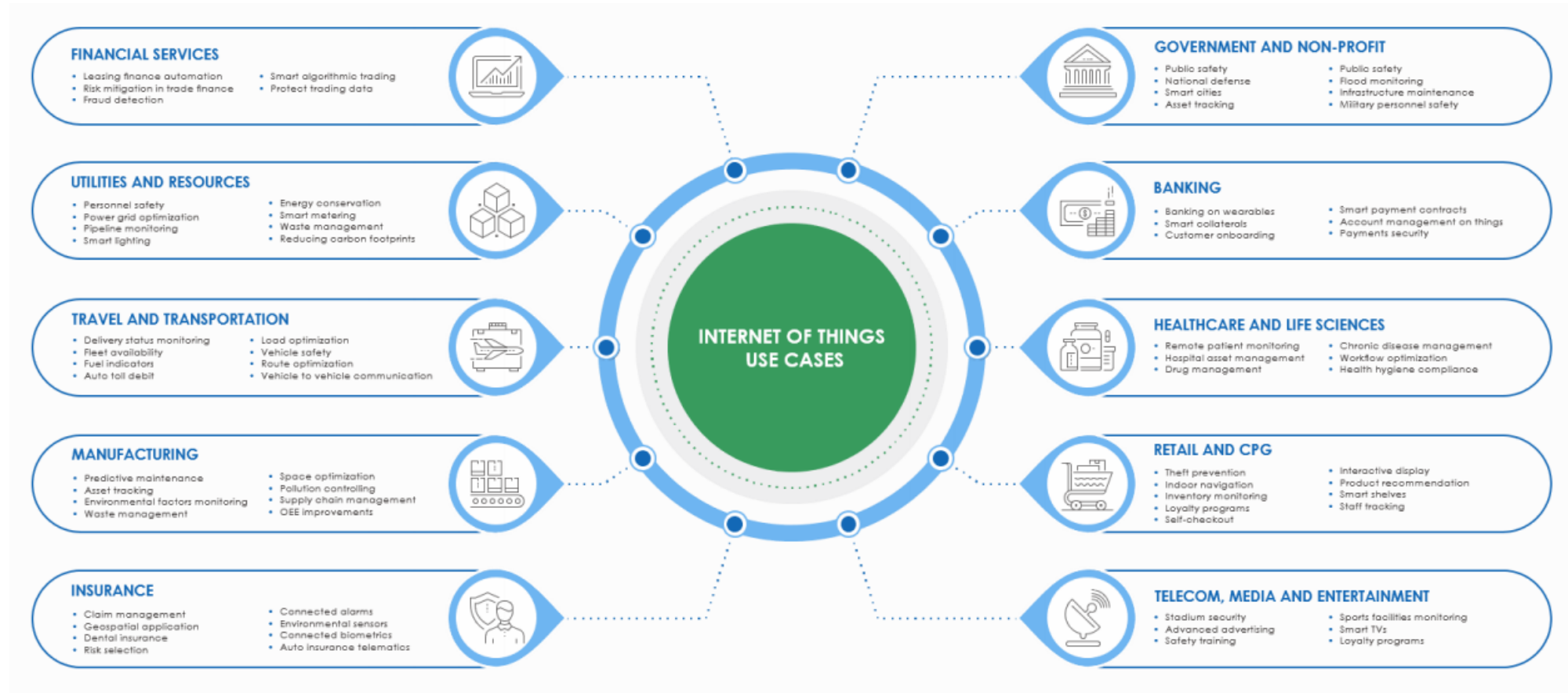
# Agenda

- General overview of IoT
- Examples of IoT Solutions in Banking (IoT for security)
- IoT Architecture
- IoT Architecture in detail (the smart branch)
- Overview about IoT net protocol
- Overview about IoT application protocol
- MQTT, AMQP in detail

- [IEEE] Broadly speaking, the Internet of Things is a system consisting of networks of sensors, actuators, and smart objects whose purpose is to interconnect 'all' things, including everyday and industrial objects, in such a way as to make them intelligent, programmable, and more capable of interacting with humans and each other.
- [NIST] [IETF] [ITU] [OASIS] [ETSI] There are a lot of definitions...

*The Things must communicate **without human interaction.***

# IoT – Use cases



# Auriga IoT - Bank4Me

A dock station that will allow customers to **engage via video conference** with a consultant and independently access all services of their bank. It is a reserved area inside a branch or a closed area outside the branch, accessible 24/7 with identification

- private space
- secure access
- full bank's service set
- advisory & product selling





# Auriga IoT - Bank4Me

## Devices

Banknote cash in/out management module, coin cash in/out management module, video conference monitor, tablet for graphometric signature, A4 scanner and checks, printer, card reader, NFC reader, QR code reader, barcode reader, EPP keypad

## Services

Simple transactional operations - on account and cash, complex transactional operations on account and cash, operations with authorization, sale of digital products, sale of non-digital products, post-sale operations, specialised advice on appointment



# Auriga IoT - ATM Security




# Auriga IoT - ATM Security

- Inking Cassettes
- Gas/Explosive Protection
- Electronic Keylock
- Shutter Protection
- Antiskimming
- Vibrations Sensor
- Anchoring Systems
- ATM Cameras
- GSM/3g
- GPS
- Temperature sensor
- Motion sensor
- Humidity sensor
- Vibrations Sensor
- Light Sensor
- others beacons







How can devices  
work to improve  
bank security?



# Auriga IoT - Attacks on ATMs

**ATM Jackpotting** is a sophisticated crime in which thieves install malicious software and/or hardware at ATMs that forces the machines to dispense huge volumes of cash on demand.

1. Gain physical internal access to an ATM through the top-hat of the terminal.
2. Use an endoscope instrument to locate internal portion of the cash machine.
3. Attach a cord that allows them to sync their device to the ATM's computer.
4. Use a keyboard or device to access the ATM's computer.
5. Install the ATM Malware.
6. Collect cash with money mule.

This is a summarized attack process and is no way an absolute methodology to ATM attacking.



# Auriga IoT - Attacks on ATMs

**Physical attacks on ATMs** are considered risky, as it not only leads to financial losses but also involves the risk to property and life. The physical attack involves solid and gas explosives attacks, along with physical removal of ATM from the site and later using other techniques to gain access to the cash dispenser.

There are **others type of attacks**: Man-in Middle attack, Data Sniffing Attacks, Skimming with Spoofing, etc...



# Auriga IoT – Security scenario

In one branch, the following sensors were arranged:

- Humidity and temperature sensor
- Linear Hall sensor

In particular, they have been included within the ATMs installed in the branch:

- Humidity and temperature sensor
- Motion sensor
- Light Sensor
- GPS Tracker



# Auriga IoT – Security scenario

## ATM Jackpotting

The sensors located on ATM detect that an attacker is trying to open the front panel of the ATM. The server send a command to power off the cash dispenser. The attacker cannot dispense notes form the cash dispenser.

## Physical attacks

An attacker removes the ATM safe and take it away. The GPS tracker device integrated in it will start sending, directly to the server, the position of the safe within 0.1 seconds from the first movement.

# IoT Architecture



THE #NEXTGENBANK



[www.aurigasp.com](http://www.aurigasp.com)





# IoT Architecture requirements

**Reliability:** Mission critical IoT systems must provide a reliability higher than 99.999% to avoid serious implications. This requires the introduction of a high degree of redundancy at every architectural level

**Latency:** Maximum allowed latency for real-time IoT systems should be lower than 100 ms. This requires increased computational power and bandwidth, as well as avoiding potential architectural bottlenecks

**Scalability:** IoT systems often are characterized by exponential growth in terms of data volumes and number of IoT entities involved. This requires a high level of scalability and adaptability



# IoT Architecture requirements

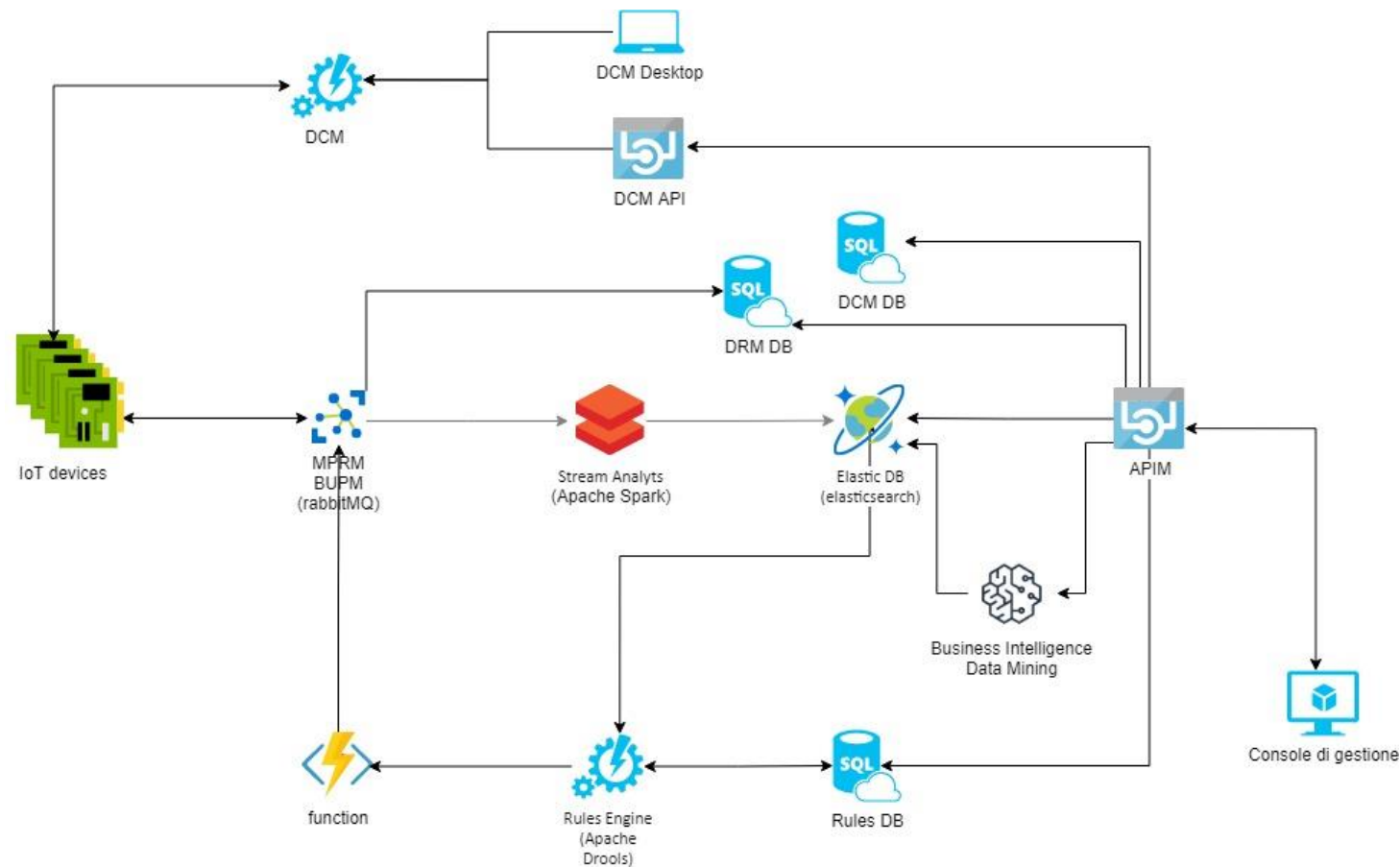
**Security:** As the number of IoT nodes connected to the Internet increases, so does the surface area of the network exposed to attacks. This implies the need for increased security at every architectural level

**Privacy:** The continuous flow of data collected by IoT devices is a potential threat to the privacy of individuals. This implies additional strategies for anonymization and user data protection

**Runtime Integration:** The ability for an IoT system to dynamically integrate heterogeneous components and data that are not known a priori at run time. This requires service discovery and self-configuration capabilities

**Network** IoT networking involves balancing a set of competing requirements

# IoT Architecture in detail





# Hub IoT

The IoT Hub enables highly secure and reliable communications between the Hub and managing IoT devices. IoT Hub provides a backend to which any IoT device can be connected by offering a communication channel with enhanced security for sending and receiving data from IoT devices. Furthermore, the Hub has the task of standardizing the protocols used by the various IoT devices.

- Based on RabbitMQ



# Devices Configuration Module

This component must manage the configuration of the devices present on the periphery using the technologies made available by the manufacturers. In addition to the configuration of the device, this module must be able to manage the updates of the device (OTA) not only for the application functionalities but also for the security updates.



# Stream Analyst

Streaming analytics is the processing and analyzing of data records continuously rather than in batches. Generally, streaming analytics is useful for the types of data sources that send data in small sizes (often in kilobytes) in a continuous flow as the data is generated.

Streaming analytics may include a wide variety of data sources, such as telemetry from connected devices, log files generated by customers using web applications, ecommerce transactions, or information from social networks or geospatial services. It's often used for real-time aggregation and correlation, filtering, or sampling.

- Apache Spark





# Elasticsearch

All data processed and generated by the Stream Analytics module must be saved in order to be used for future analysis and evaluations. Since these data are heterogeneous, it is necessary to use a different solution than traditional relational databases. Elasticsearch is a Lucene-based search server, with Full-Text capability, with support for distributed architectures. All the functionalities are natively exposed through the RESTful interface, while the information is managed as JSON documents

- Elasticsearch



# Rules Engine

Rules engines or inference engines serve as pluggable software components which execute business rules that a business rules approach has externalized or separated from application code. This externalization or separation allows business users to modify the rules without the need for IT intervention. The system as a whole becomes more easily adaptable with such external business rules, but this does not preclude the usual requirements of QA and other testing.

- Drools
- Camunda



# Others components

- IoT devices
- API
- Databases
- Business Intelligence and Data mining
- Web console

# IoT Protocols

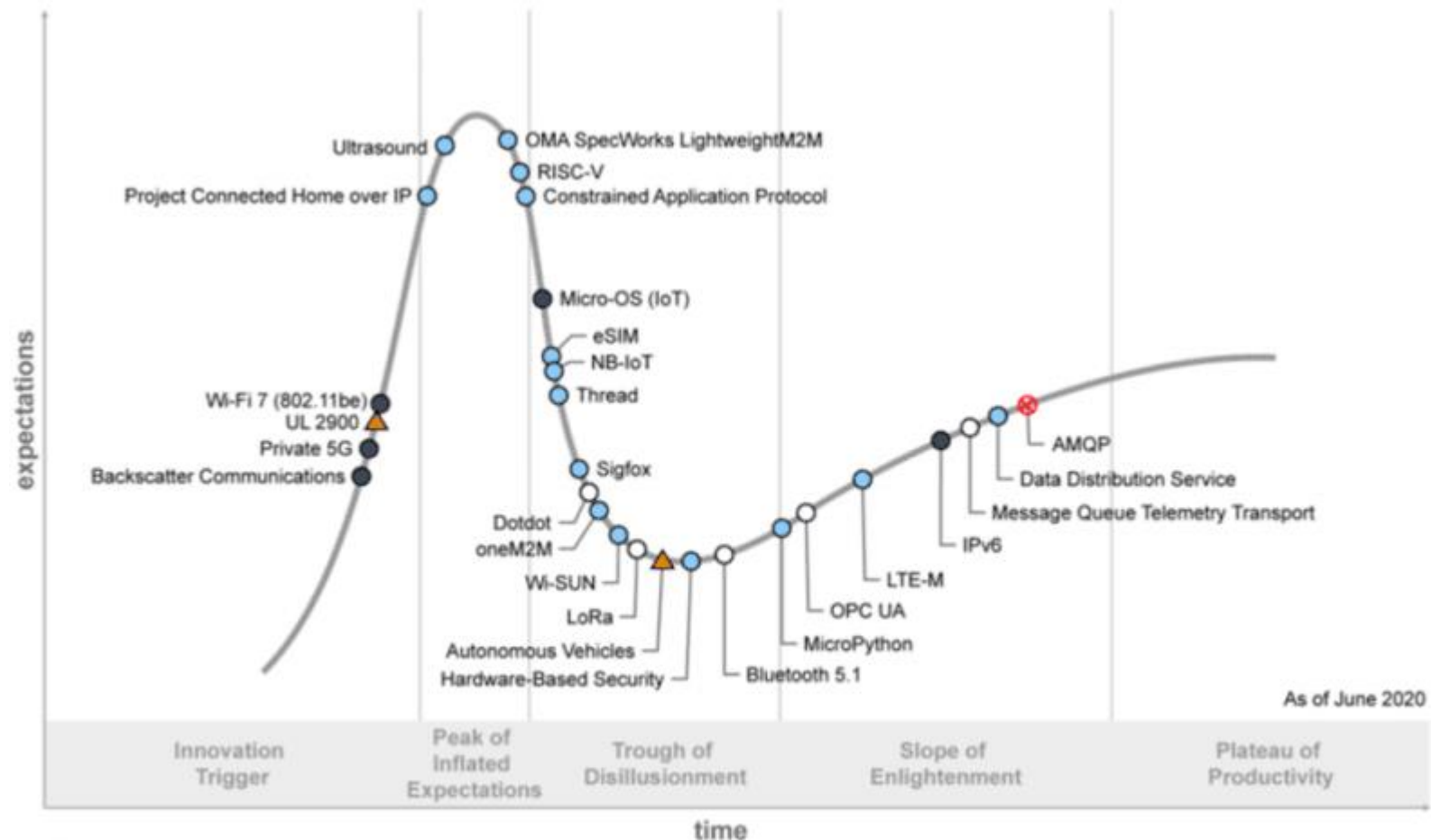


THE #NEXTGENBANK



[www.aurigasp.com](http://www.aurigasp.com)

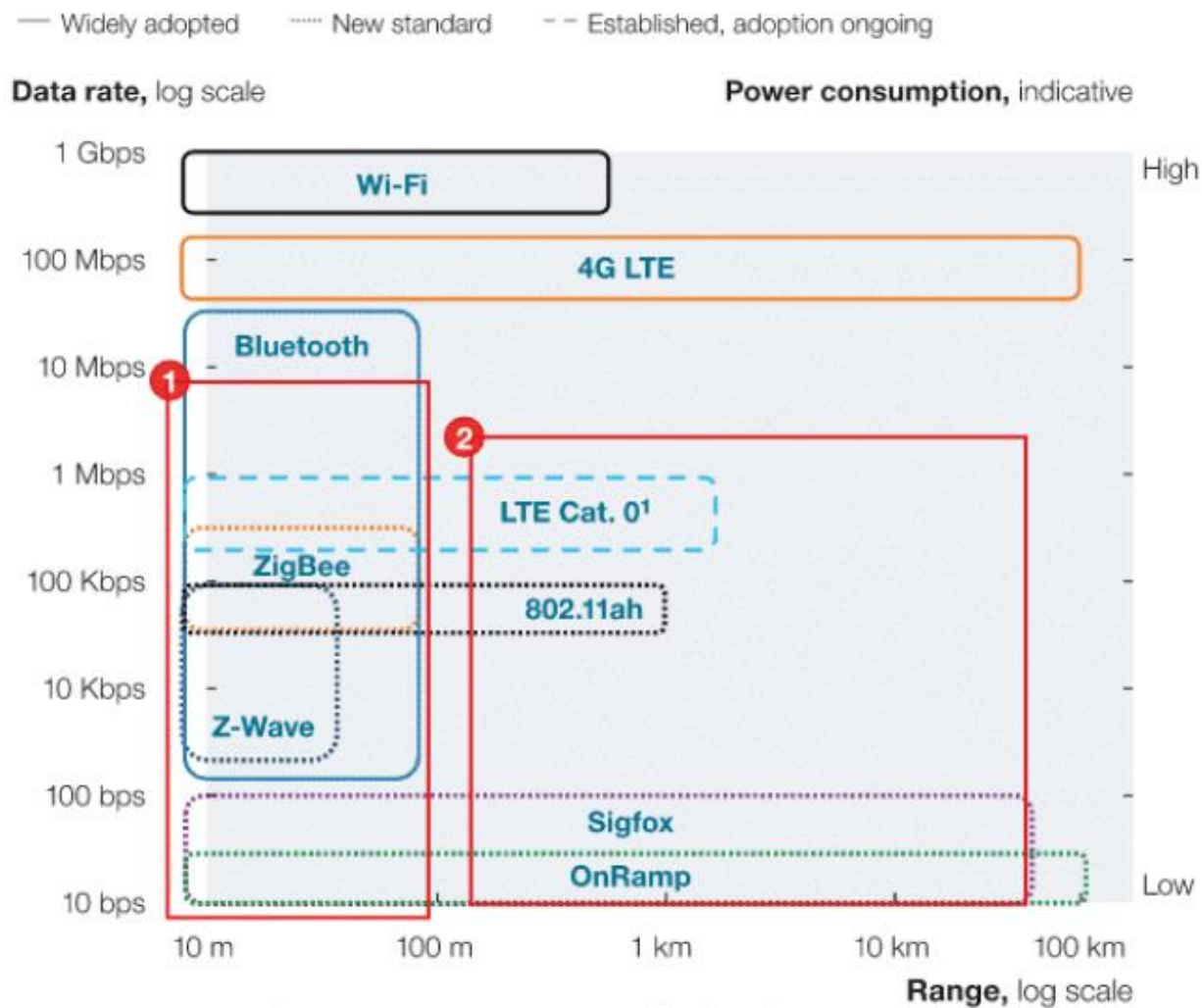
# Hype Cycle for IoT Standards



Plateau will be reached:

○ less than 2 years    ● 2 to 5 years    ● 5 to 10 years    ▲ more than 10 years    ⊗ obsolete before plateau

# Network Protocols













- 1 Many competing standards for low-range, medium-low data rate hinder growth for many IoT applications
- Interoperability missing
  - Consortia wars might be emerging
  - Additional incompatibilities in higher communication layers, eg, 6LoWPAN vs ZigBee

- 2 Standard white space for low-data-rate, low-power, high-range applications such as smart grid
- Wi-Fi and LTE have high power consumption
  - Alternatives with low power and wide range (eg, LTE Cat. 0, 802.11ah, Sigfox, and OnRamp) are in very early stages and compete against each other



# Network Protocols

	Local Area Network Short Range Communication	Low Power Wide Area (LPWAN) Internet of Things	Cellular Network Traditional M2M
	40%	45%	15%
	Well established standards In building	Low power consumption Low cost Positioning	Existing coverage High data rate
	Battery Life Provisioning Network cost & dependencies	High data rate Emerging standards	Autonomy Total cost of ownership
	  		  



# Application Protocols

CoAP, MQTT, AMQP

**AURIGA**  
the banking e-evolution

THE **#NEXTGENBANK**

 [www.aurigaspa.com](http://www.aurigaspa.com)



# HTTP is not suitable for IOT

- **One-to-one communication:** HTTP is designed for communication between 2 systems only at a time. HTTP does not fulfill the need for one to many communication between sensors and the server.
- **Uni-Directional:** HTTP is unidirectional in the sense that only one system(client or server) can send a message to the other at any point in time.
- **Synchronous request-response:** After requesting a resource to the server, the client has to wait for the server to respond. This blocks some system resources like threads, CPU cycles etc.
- **High Power Consumption:** Since HTTP utilizes heavy system resources as explained above, this also leads to heavy power consumption.



# HTTP is not suitable for IOT

- **Not designed for event-based communication:** Most of the IOT applications are event based. The sensor devices measure for some variable like temperature, air quality and contents etc. and might need to take event driven decisions like turning off a switch etc.
- **Scalability:** HTTP connections utilize high system resources especially I/O threads. For every HTTP connection, the client/server has to open an underlying persistent TCP connection as well.

As clear from above, HTTP has severe limitations for IOT applications. Many advanced application-layer protocols(MQTT, AMQP, CoAP) have been developed to overcome these limitations.



# CoAP [RFC 7252]

Constrained Application Protocol (CoAP) is a specialized Internet Application Protocol for constrained devices, as defined in RFC 7252. It enables those constrained devices called "nodes" to communicate with the wider Internet using similar protocols. CoAP is designed for use between devices on the same constrained network (e.g., low-power, lossy networks), between devices and general nodes on the Internet, and between devices on different constrained networks both joined by an internet. CoAP is also being used via other mechanisms, such as SMS on mobile communication networks.

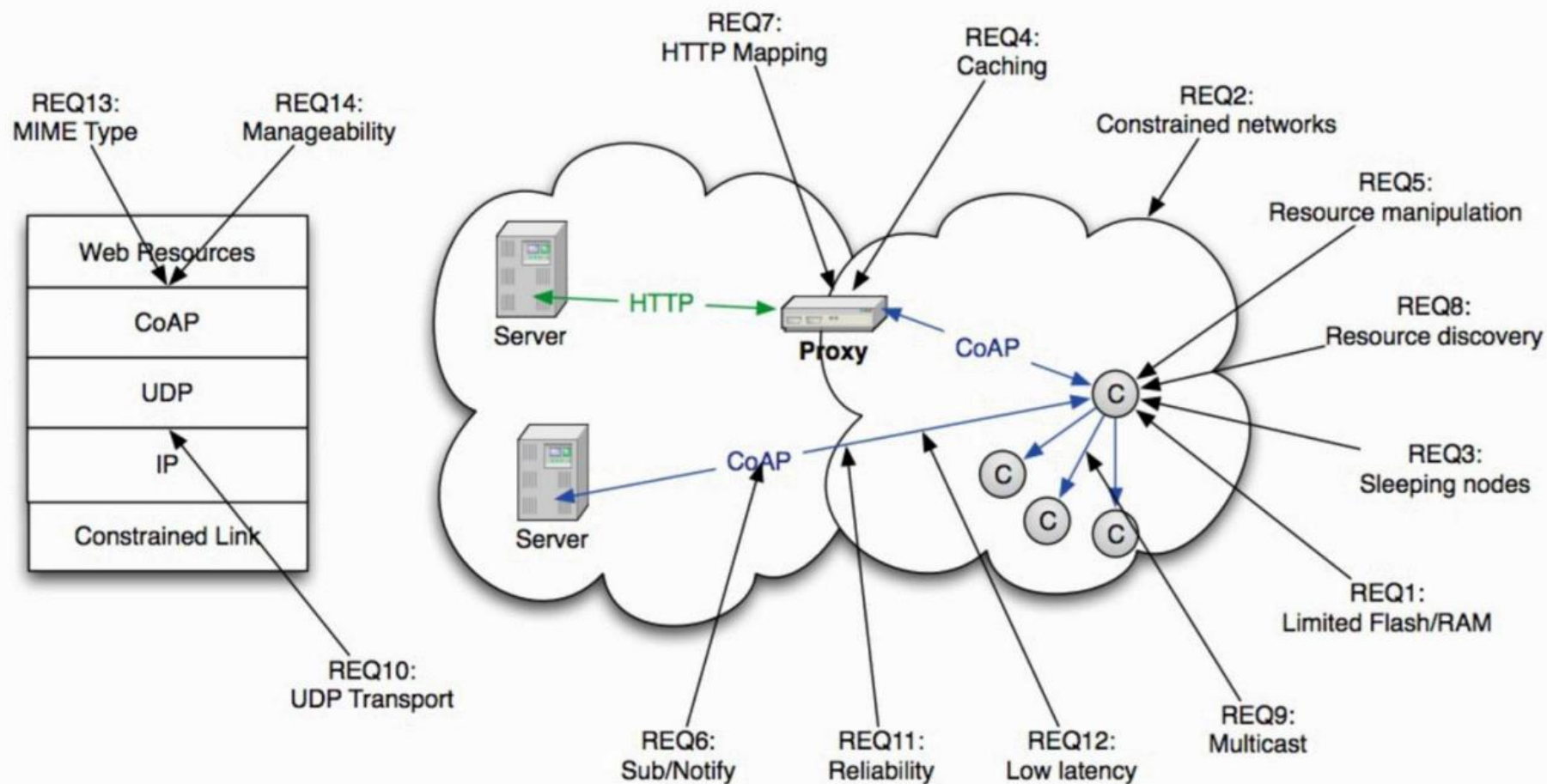
# CoAP [RFC 7252]

- Designed to easily translate to HTTP for simplified integration with the web
- Support UDP, unicast and multicast
- Asynchronous messaging
- Low Header Overhead
- Simple proxying and Caching Capabilities
- Security with Datagram TLS (DTLS)

Code	Descrizione	Corrispondente HTTP
64	2.00 OK	200 OK
65	2.01 Created	201 Created
66	2.02 Deleted	204 No Content
67	2.03 Valid	304 Not Modified
68	2.04 Changed	204 No Content
128	4.00 Bad Request	400 Bad Request
129	4.01 Unauthorized	400 Bad Request
130	4.02 Bad Option	400 Bad Request
131	4.03 Forbidden	403 Forbidden
132	4.04 Not Found	404 Not Found
133	4.05 Method Not Allowed	405 Method Not Allowed
141	4.13 Request Entity Too Large	413 Request Entity Too Large
143	4.15 Unsupported Media Type	415 Unsupported Media Type
160	5.00 Internal Server Error	500 Internal Server Error
161	5.01 Not Implemented	501 Not Implemented
162	5.02 Bad Gateway	502 Bad Gateway
163	5.03 Service Unavailable	503 Service Unavailable
164	5.04 Gateway Timeout	504 Gateway Timeout
165	5.05 Proxying Not Supported	502 Bad Gateway



# CoAP [RFC 7252]

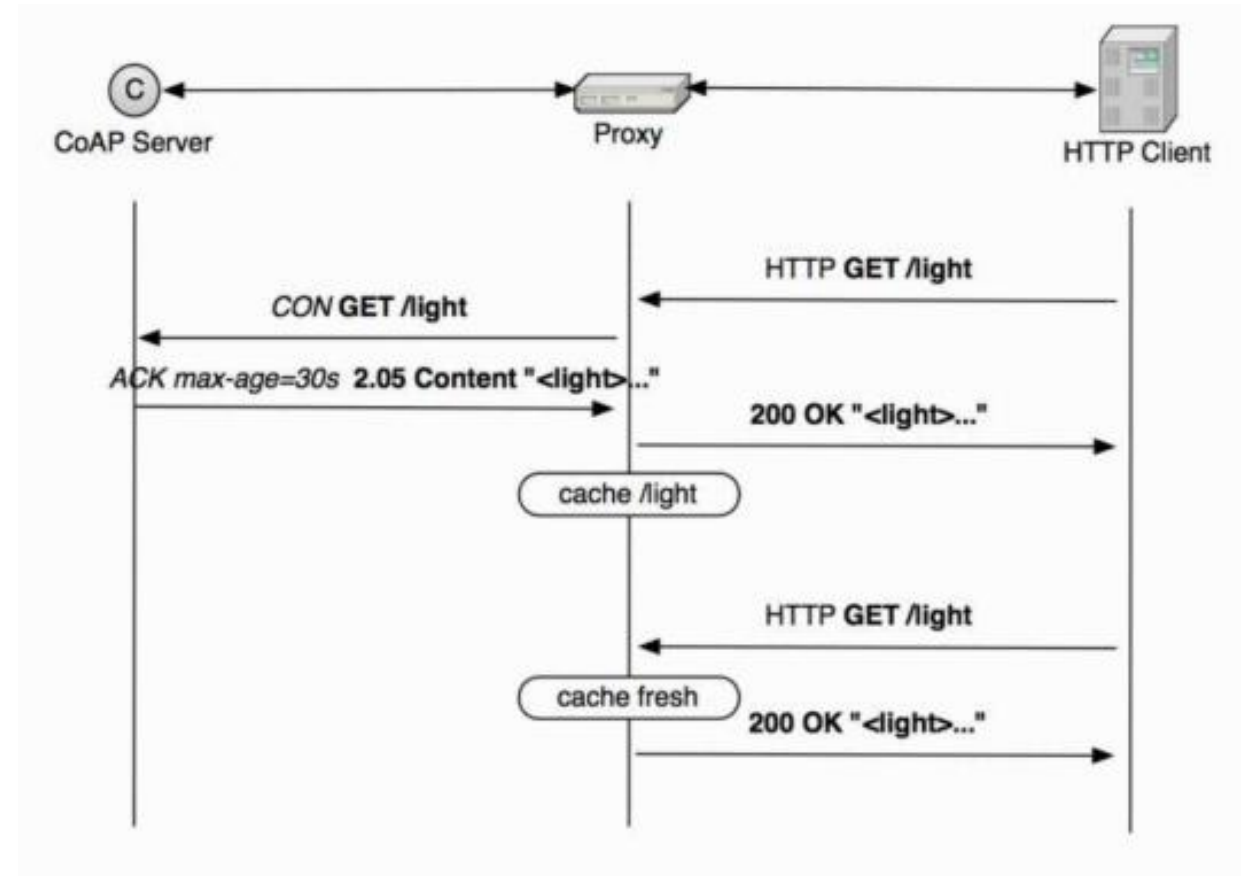


# CoAP [RFC 7252]

The protocol supports the caching of responses in order to efficiently fulfill requests.

Simple caching is enabled using freshness and validity information carried with CoAP responses.

A cache could be located in an endpoint or an intermediary.



# CoAP [RFC 7252]



Based on the **observer pattern**. It is a software design pattern in which an object, named the subject, maintains a list of its dependents, called observers, and notifies them automatically of any state changes, usually by calling one of their methods.

- The server keep a list of observers for each resource;
- The clients are the observers

**Issues:** network congestion caused by notifications, security.



# MQTT [OASIS]

The Message Queuing Telemetry Transport (**MQTT**) is a lightweight, **publish-subscribe** network protocol that transports messages between devices.

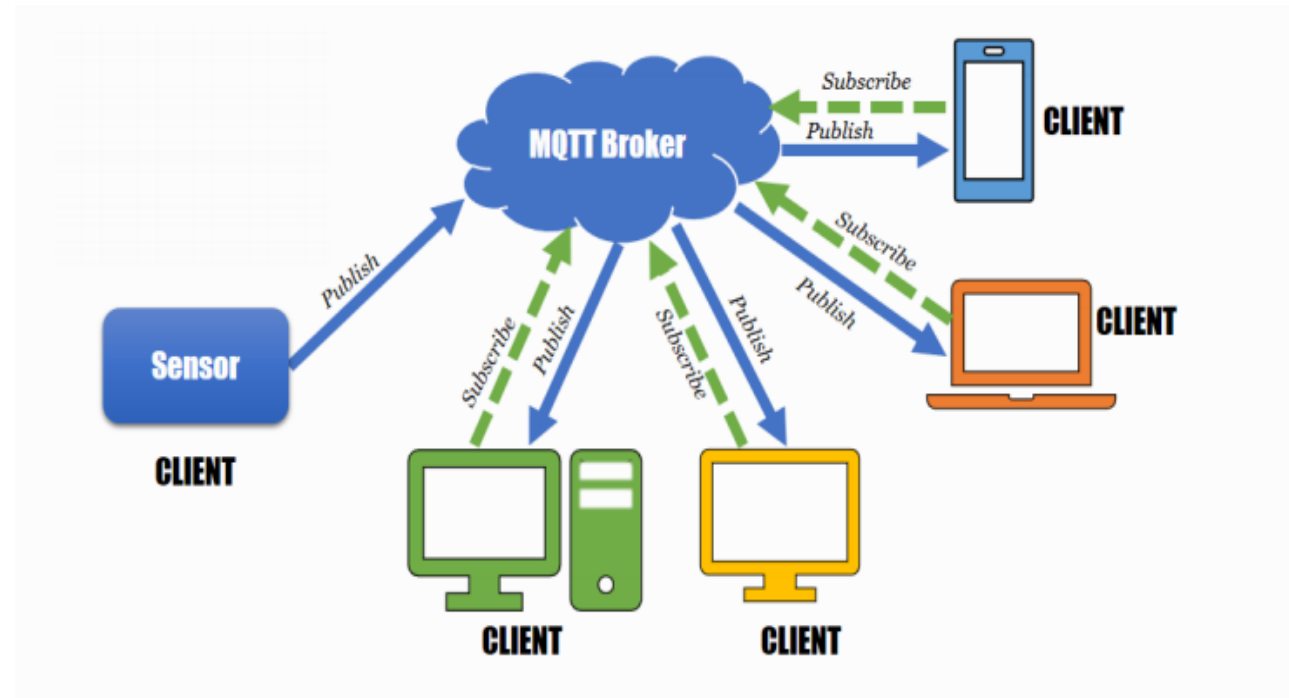
The protocol usually runs **over TCP/IP**; however, any network protocol that provides ordered, lossless, **bi-directional connections** can support MQTT.

It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited.

# MQTT - Publish/Subscribe [OASIS]

An **MQTT broker** is a server that receives all messages from the clients and then routes the messages to the appropriate destination clients.

An **MQTT client** is any device (from a micro controller up to a fully-fledged server) that runs an MQTT library and connects to an MQTT broker over a network





# MQTT – Others Features [OASIS]

- Allow application decoupling;
- Payload agnostic;
- Low Transport Overhead (minimal message exchange);
- Hierarchical Namespace: Information is organized in a hierarchy of topics. The broker distributes the information to any clients that have subscribed to that topic.
- Retain Flag: A retained message is a normal MQTT message with the retained flag set to true. The broker stores the last retained message and the corresponding QoS for the selected topic. Each client that subscribes to a topic pattern that matches the topic of the retained message receives the retained message immediately after they subscribe. The broker stores only one retained message per topic.



# MQTT - QoS [OASIS]

Each connection to the broker can specify a quality of service measure. These are classified in increasing order of overhead:

- At most once - the message is sent only once and the client and broker take no additional steps to acknowledge delivery (fire and forget).
- At least once - the message is re-tried by the sender multiple times until acknowledgement is received (acknowledged delivery).
- Exactly once - the sender and receiver engage in a two-level handshake to ensure only one copy of the message is received (assured delivery).

# MQTT – Message Structure [OASIS]

## Message Type

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

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





# MQTT-SN [OASIS]

MQTT-SN (MQTT for Sensor Networks) is a variation of the main protocol aimed at battery-powered embedded devices on non-TCP/IP networks, such as Zigbee.

After taking over maintenance of the standard from IBM version 3.1.1 with minor changes was released as an OASIS standard on October 29, 2014.

A more substantial upgrade to MQTT version 5, adding several new features was released on March 7, 2019.



# CoAP vs MQTT

CoAP	MQTT
One-to-One Communication Protocol	Many-to-Many Communication Protocol
Document – Centric	Data – Centric
Client/server	Publish/Subscribe
Based on UDP	Based on TCP
QoS Levels based on messages (NON/CON)	3 QoS Levels.



# AMQP

The Advanced Message Queuing Protocol (AMQP) is an open standard application layer protocol for message-oriented middleware. The defining features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security.

It defines a binary wire-level protocol that allows for the reliable exchange of business messages between two parties. AMQP has a layered architecture and the specification is organized as a set of parts that reflects that architecture.



# AMQP vs MQTT

	AMQP	MQTT
<b>Protocol design</b>	Offers a richer set of messaging scenarios defining features: <ul style="list-style-type: none"><li>• Message orientation</li><li>• Queuing</li><li>• Routing (including P2P and pub/sub)</li><li>• Reliability</li><li>• Security</li></ul>	Extremely simple, lightweight pub/sub messaging. Designed for: <ul style="list-style-type: none"><li>• resource-constrained devices</li><li>• low-bandwidth, highlatency or unreliable networks</li><li>• Ensures reliability</li><li>• Some degree of assurance of delivery (3 QoS levels)</li></ul>
<b>Communication type</b>	Asynchronous	Asynchronous



# AMQP vs MQTT

	AMQP	MQTT
<b>Framing of data</b>	<ul style="list-style-type: none"><li>• Buffer-oriented approach (for high-performance servers)</li><li>• Message fragmentation</li></ul>	<ul style="list-style-type: none"><li>• Stream-oriented approach (easier for low-memory clients to write frames)</li><li>• No message fragmentation (difficult to transmit large messages)</li></ul>
<b>Reliable messaging</b>	"Fire & forget, don't try too hard" approach, with finegrained control when order of delivery matters	"Fire & forget, don't try too hard" approach, with 3 QoS levels (at most once, at least once, exactly once)
<b>Message namespace</b>	Multiple namespaces, with different ways of finding messages	Single hierarchical namespace (topic namespace). The namespace is global



# AMQP vs MQTT

	AMQP	MQTT
<b>Queueing</b>	Supports queues of messages	No queues (only retains 1 message per topic)
<b>Transactions support</b>	supports different forms of message acknowledgment and transactions	Does not support any kind of Transaction, supports only basic message acknowledgment
<b>Security</b>	<ul style="list-style-type: none"><li>• Connections use authentication and can be protected using TLS</li><li>• Uses SASL mechanisms, allowing to choose any security strategies without protocol changes</li><li>• Supports proxy security servers and nested firewalls</li></ul>	<ul style="list-style-type: none"><li>• Supports passing user name and password with a message</li><li>• Encryption across the network can be handled with SSL (independently of MQTT itself)</li></ul>



# Auriga

The information provided in this document is the property of Auriga, and any modification or use of all or part of the content of this document without the express written consent of Auriga is strictly prohibited. Failure to reply to a request for consent shall in no case be understood as tacit authorization for the use thereof.

© Auriga S.p.A.

Le informazioni fornite in questo documento sono di proprietà di Auriga. Eventuali modifiche o l'utilizzo di tutto o di una parte del contenuto di questo documento senza il consenso espresso per iscritto da parte di Auriga è severamente proibito. In nessun caso la mancata risposta a una richiesta di consenso deve essere interpretata come il tacito consenso all'uso della stessa.

© Auriga S.p.A.



# Thank You

Gaetani Ziri  
[gaetano.ziri@aurigaspa.com](mailto:gaetano.ziri@aurigaspa.com)

**AURIGA**  
the banking e-evolution

THE **#NEXTGENBANK**

 [www.aurigaspa.com](http://www.aurigaspa.com)