Data Science for

INTERNET OF THINGS

(IoT) Applications

21CSE429T DATA SCIENCE FOR INTERNET OF THINGS

Syllabus
Unit-1 - Defining IoT Analytics and Challenges 9 Hrs *The situation - Defining IoT analytics - Defining analytics - Defining the Internet of Things - The concept of constrained - IoT analytics challenges - The data volume - Problems with* *Data quality - Analytics challenges - Business value concerns*
Unit-2 - IoT Devices, Networking Protocols and Standards for Internet of Things 9 Hrs *IoT Devices-Healthcare-Manufacturing-Transportation and logistics-Retail-Oil and gas- - Home automation or monitoring - Wearables - Sensor types-IoT Data Link Protocols-Network Layer Routing Protocols - Network Layer-Encapsulation Protocols -Session Layer Protocols-IoT Management Protocols-Security in IoT Protocols-IoT Challenges*
Unit-3 - IoT Sensing, Mobile and Cognitive Systems 9 Hrs *Sensing Technologies for Internet of Things - IoT Interactions with GPS, Clouds and Smart Machines - Radio Frequency Identification (RFID) - Sensors, Wireless Sensor Networks and GPS Systems - Cognitive Computing Technologies and Prototype Systems – Problems*

21CSE429T DATA SCIENCE FOR INTERNET OF THINGS

Syllabus

Unit-4 - Smart Applications IoT with Data Analytics 9 Hrs

Defragmenting Intelligent Transportation: A Practical Case Study -Connected and Autonomous Vehicles-Transit Hub: A Smart Decision Support System for Public Transit Operations –
Smart Home Services Using the Internet of Things

Unit-5 - Case Studies in IoT Healthcare 9 Hrs

Big Data Analytics for Healthcare and Cognitive Learning - Machine Learning for Big Data in Healthcare Applications - Healthcare Problems and Machine Learning Tools - IoT-based Healthcare Systems and Applications, Emotional Insights via Wearables- Structural Health Monitoring-Home Healthcare and Remote Patient Monitoring

1. Analytics for the Internet of Things (IoT) by Andrew Minteer, Released July 2017, Publisher(s): Packt Publishing, ISBN: 9781787120730.
2. Big-Data Analytics for Cloud, IoT and Cognitive Computing, Kai Hwang, Min Chen, ISBN: 978-1-119-24729-6 March 2017.
3. Internet of Things and Data Analytics Handbook,Hwaiyu Geng (Editor) - ISBN: 978-1-119- 17364-9 January 2017
4. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton and Jerome Henry, IoT Fundamentals: Networking Technologies, Protocols and Use Cases for Internet of Things, Cisco Press, 2017
5. Arshdeep Bahga, Vijay Madisetti, Internet of Things – A hands-on approach, Universities Press, 2015

21CSE429T DATA SCIENCE FOR INTERNET OF THINGS

Text Books
1. Analytics for the Internet of Things (IoT) by Andrew Minteer, Released July 2017, Publisher(s): Packt Publishing, ISBN: 9781787120730. *2. Big-Data Analytics for Cloud, IoT and Cognitive Computing, Kai Hwang, Min Chen, ISBN: 978-1-119-24729-6 March 2017.* *3. Internet of Things and Data Analytics Handbook,Hwaiyu Geng (Editor) - ISBN: 978-1-119- 17364-9 January 2017* *4. David Hanes, Gonzalo Salgueiro, Patrick Grossetete, Rob Barton and Jerome Henry, IoT Fundamentals: Networking Technologies, Protocols and Use Cases for Internet of Things, Cisco Press, 2017* *5. Arshdeep Bahga, Vijay Madisetti, Internet of Things – A hands-on approach, Universities Press, 2015*
Unit-1 - Defining IoT Analytics and Challenges - T1 Unit-2 - IoT Devices, Networking Protocols and Standards for Internet of Things - T1 Unit-3 - IoT Sensing, Mobile and Cognitive Systems - T2 Unit 4 - Smart Applications IoT with Data Analytics - T3 Unit-5 - Case Studies in IoT Healthcare - T2, T3

What is IoT?

- IoT is a system of interrelated devices connected to the internet to transform or receive data from one to the other.

- **Examples - Smartphones, Smart Tablets, Smart Watches, Smart Tvs, Etc.**

- **Smart home - AC, Doorbell, Geaser, Smoke Detectors, and Security Alarms**

What is Data Science?

- **Data Science** is the field of study that involves extracting knowledge and insights from noisy data and turning those insights into meaningful actions that a business/organization can take.

- Plays a crucial role in Internet of Things by extracting knowledge and meaningful insights from huge data collected from various interconnected devices.

Difference between Traditional Data and IoT Data

Traditional Data	IoT Data
created by human.	***data is created by the machine***
Collecting and organized data is a straightforward process	is continuously pushed and seems never-ending.
Data Analysis is initiated whenever required	continuous arriving data must be quickly analyzed and instant decisions must be taken

- If the data processing is slowed down in IoT, the overall value of the data might be essentially reduced.

Data Science Techniques used in IoT applications:

1.Cognitive Computing:

Advanced techniques such as NLP, Pattern Matching and Machine Learning can be used to gain meaningful insights from huge amounts of data which cannot be done using conventional methods.

2.Real-Time Processing:

The data coming from IoT devices is continous and non stop. The actions must be taken immediately based on the incoming data and a delay can cause huge loss and also a second or a minute is a huge thing.

3.Deep Learning:

Various techniques of Deep Learning such as Neural Networks can be used to process unstructured data either in the form of text,image,audio or a video.Image Processing and Speech Processing, Natural Language Processing and Autonomous Decison-Making techniques can be utilized in IoT applications.

Data Science Techniques used in IoT applications:

4.Big Data Analysis:

Big Data to a huge arrangement of data that no traditional management tool of data can deal with and permits admittance to huge data sets in real time. They can be used to gather insights from IoT data that would be very difficult to deal with traditional methods.

5.Edge Computing:

refers to processing or utilizing data at the edge of the network or closer to where its being generated which adds value to real-time processing in IoT applications. This also improves the response time and minimizes bandwith.

IoT Applications Empowered by Data Science

1.Predictive Maintenance:

The **algorithms of Data Science** can help in analyzing the sensor data of various devices connected to IoT to predict failures and maintenance needs. This can help organizations to take precautionary measures in advance further reducing the downtime and increasing the lifespan of resources.

2.Retail Analytics:

Data science can help retailers to analyze customer feedback and experiences thereby increase sales and reduce costs. Various algorithms can be used to analyze customer behavior and develop pricing strategies that generates more sales.

3.Healthcare: Smart devices

such as wearable fitness trackers and healthcare monitoring systems can collect patients health related data. This data can be anaylzed using data science models and can be used by healthcare professionals to montior vital signs and predict diseases and early intervention.

IoT Applications Empowered by Data Science

4.Traffic Management:

Data Science can be applied to data obtained from sensors embedded in infrastructure to manage traffic flow, improve energy consumption and also improve city services.

5.Automation:

Data science can be used in smart homes to automate tasks such as electricity supplies and security based on the users preferences and patterns

Challenges of IoT Applications in Data Science

1.Data Storage and Analysis :

IoT Devices carry with huge amounts of data which is very expensive and challenging to store and process large volumes of data. Hence, effective solutions are needed to store and process lots of data.

2.knowledge discovery and computational complexities:

As the data is massive, figuring out and understanding useful information is very hard because of its size and complexity. An action that could be taken to solve this would be putting away the data that is acquired from the working frameworks.

3.Data Analysis and Visualization:

It is often challenging to apply data science methods in a secure way and also it is difficult to present the data in a meaningful way.

Challenges of IoT Applications in Data Science

4.Too Much Data :

Too much of data becomes overwhelming and difficult to filter out. Errors in data entry operations could lead to poor data quality. However, much categorized industry principles needs to be utilized to overcome this.

5.Balancing Scale and Speed :

Data from IoT devices can become challenging to analyze in large scale environments like cloud. The cloud may not be suitable for real time processing scenarios.

What is IoT Analytics?

- The objective of IoT analytics is to gain value from large volumes of data generated by devices connected via the Internet of Things (IoT).
- Organizations use IIoT to collect and analyze data from **pipelines, weather stations, sensors on manufacturing equipment, smart meters, delivery trucks, and other machinery.**
- IoT analytics is also used in **retail, data center management, healthcare.**
- IoT data is a subset of big data, and is constantly growing in **volume, variety and velocity (the 3Vs model).** It consists of heterogeneous streams that need to be transformed and combined to produce current, comprehensive and accurate information for business analysis and reporting.
- Many IoT devices **were not developed for compatibility with other IoT devices** and systems**. IoT data in**tegration is thus complex, as is the analytics that relies on it.

Types of IoT Analytics

- **Descriptive analytics on IoT data**
- **Diagnostic analytics on IoT data**
- **Predictive analytics on IoT data**
- **Prescriptive analytics on IoT data**

Descriptive analytics on IoT data

- Focuses on what's happening, by **monitoring the status of IoT devices, machines, products and assets**. Determines if things are going as planned, and notifies if anomalies occur. Descriptive analytics is generally implemented **as dashboards that show current and historical sensor data, key performance indicators (KPIs), statistics and alerts.**

- Addresses questions such as:

- Are there any anomalies that demand attention?
- What's the utilization and throughput of this machine?
- How are consumers using our products?
- Where do my assets reside?
- How many components are we creating with this tool?
- How much energy is this machine using?

Diagnostic analytics on IoT data

- Answers the question: **why is something happening?** Analyzes IoT data to identify core problems and to fix or improve a service, product or process.
- Diagnostic capabilities are typically extensions to dashboards that permit users to drill into data, compare it, and visualize correlations and trends in an ad-hoc manner.
- Addresses questions such as:

- Why is this machine producing more defective parts than other machines?
- Why is this machine consuming excessive energy?
- Why aren't we producing enough parts with this tool?
- Why are we getting a lot of product returns from American customers?

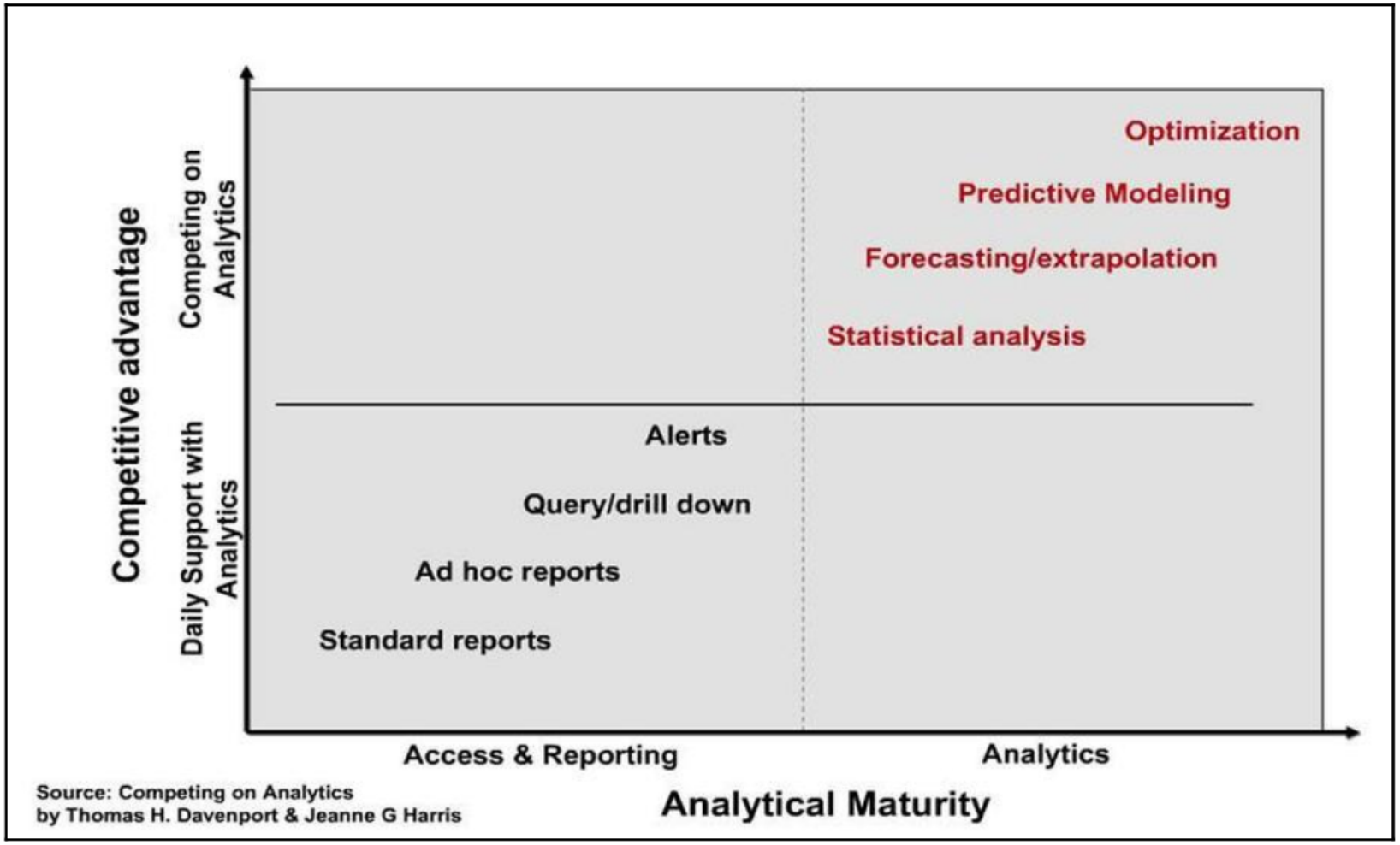
Predictive Analytics

- Predictive analytics incorporates machine learning capabilities to assess the likelihood of a future event happening.
- Machine learning models are trained with vast amounts of historical data that allow it to identify trends and the probabilities of certain things leading to certain outcomes.
- It applies this knowledge to the real-time data coming in from IoT devices to effectively predict the future. These types of insights give organizations the time to act proactively to change the predicted outcome if it’s not what is desired.

Prescriptive Analytics

- One of the more advanced analytics capabilities is prescriptive.
- Prescriptive analytics gives additional insights into what actions you can take to impact the results of descriptive, diagnostic, or predictive analytics.
- It helps organizations better understand how they can prevent failures, improve effectiveness, avoid or increase outputs and more.

IoT Analytics



IoT analytics challenges

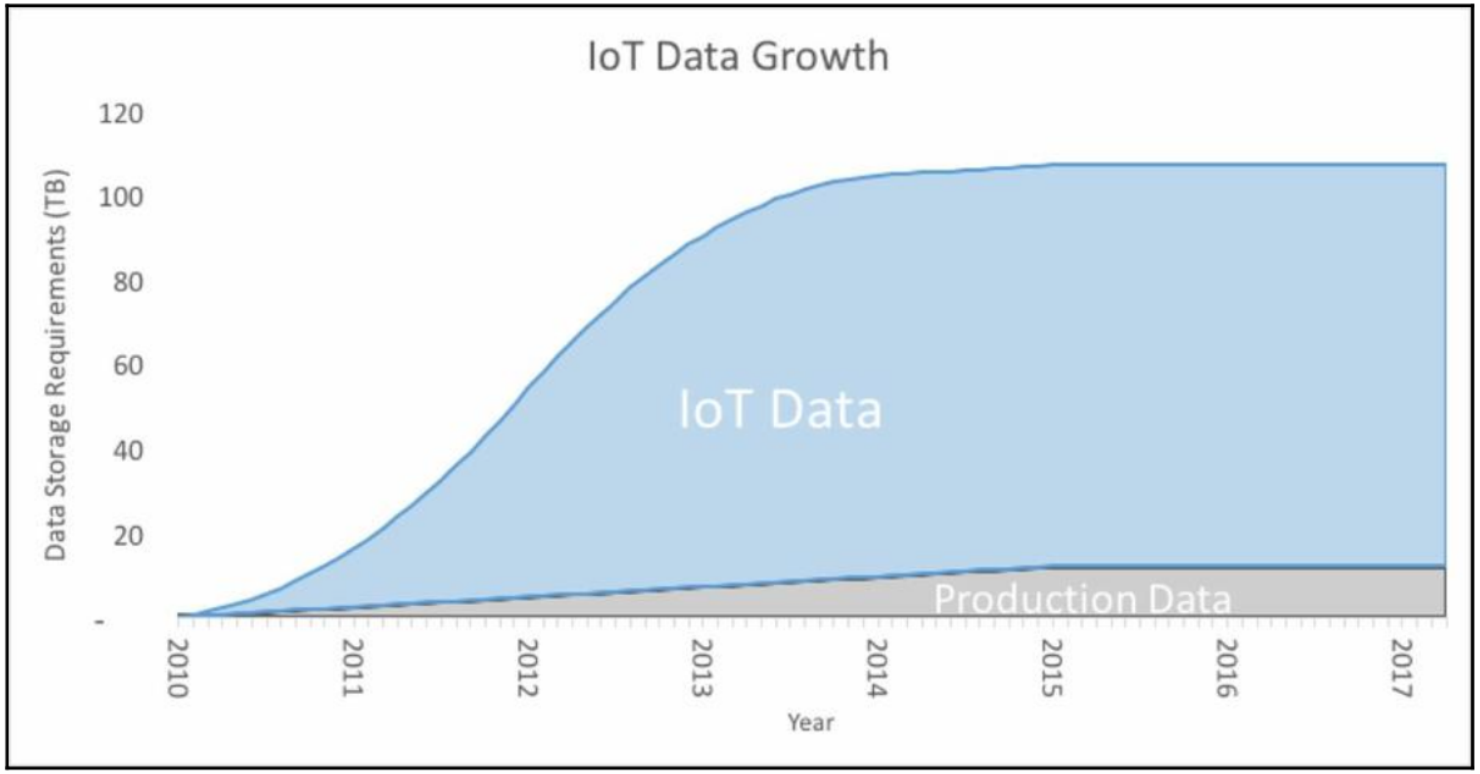
- The data was created by devices operating remotely, sometimes in widely varying environmental conditions that can change from day to day. The devices are often distributed widely geographically.
- The data is communicated over long distances, often across different networking technologies. It is very common for data to first transmit across a wireless network, then through a type of gateway device to be sent over the public internet-which itself includes multiple different types of networking technology working together.
- **The data volume**
- A company can easily have thousands to millions of IoT devices with several sensors on each unit, each sensor reporting values on a regular basis. The inflow of data can grow quite large very quickly. Since IoT devices send data on an ongoing basis, the volume of data in total can increase much faster than many companies are used to.

IoT analytics challenges

- **Data generated/created remotely in different environment**
- **Communicated across different network technologies**
- **Data Volume**
  - Storage – multiple servers
  - Computing power
- **Time**
  - Time is very tightly tied to geographical position and the date on the calendar. Coordinated Universal Time (UTC)and Greenwich Mean Time (GMT)
  - GMT is a time zone, while UTC is a time standard. UTC does not observe **Daylight Savings Time** (DST)
  - Absolute time and Local Time
  - captures a time value that may not work across multiple time zones and locations.
  - clock synchronization
- **Problems with space**
  - multiple geographic locations. Different areas of the world have different environmental conditions. Temperature variations can affect sensor accuracy
  - IoT devices are solar powered - battery charge can affect the frequency of data reporting
  - political considerations related to the location of the IoT device.
- **Data quality**
  - missing or inconsistent data - outside influences, such as environment conditions
  - not be consistent across locations - differences in reporting frequency or formatting of values. It can result in lost or mangled data.
  - dead batteries result in missing values.

IoT analytics challenges - Data Volume

- A company that manufactures small monitoring devices, which produces 12,000 devices a year, starting in 2010 when the product was launched. Each one is tested at the end of assembly and the values reported by the sensors on the device are kept for analysis for five years.
- imagine the device also had internet connectivity to track sensor values, and each one remains connected for two years. Since the data inflow continues well after the devices are built, data growth is exponential until it stabilizes when older devices stop reporting values. This looks more like the blue area in the following chart:



IoT analytics challenges

- **Problems with time**
- Time is very tightly tied to geographical position and the date on the calendar. The international standard way of tracking a common time is using **Coordinated Universal Time** (**UTC**). UTC is geographically tied to 00 longitude, which passes through Greenwich, England, in the UK. Although it is tied to the location, it is actually not the same as **Greenwich Mean Time** (**GMT**). GMT is a time zone, while UTC is a time standard. UTC does not observe **Daylight Savings Time** (DST):

- When data used for analytics is recorded at headquarters or a manufacturing plant, everything happens at the same place and time zone. IoT devices are spread out across the globe. Events that happen at the absolute same time do not happen at the same local time. How time is recorded affects the integrity of the resulting analytics.
- When IoT devices communicate sensor data, time may be captured using the local time. It can dramatically affect analytics results if it is not clear whether local time or UTC was recorded. For example, consider an analyst working at a company that makes parking spot occupancy detection sensors. She is tasked with creating predictive models to estimate future parking lot fill rates. The time of day is likely to be a very predictive data point. It makes a big difference to her on how this time is recorded. Even determining if it is night or day at the sensor location will be difficult.
- This may not be apparent to the engineer creating the device. His task is to design a device that determines if the spot is open or not. He may not appreciate the importance of writing code that captures a time value that can be aggregated across multiple time zones and locations.

IoT analytics challenges

- **Problems with time**
- There can also be issues with clock synchronization. Devices set their internal clock to be in sync with the time standard being used. If it is local time, it could be using the wrong time zone due to a configuration error. It could also get out of sync due to a communication problem with the time standard source.
- If local time is being used, daylight savings time can cause problems. How will the events that happen between 1 a.m. and 2 a.m. on the day autumn daylight savings is adjusted be recorded since that hour happens twice? Laws that determine which days mark daylight savings time can change, as they did in Turkey when DST was scrapped in September 2016. If the device is locked into a set date range at the time of manufacture, the time would be incorrect for several days out of the year after the DST dates change.
- How daylight savings time changes is different from country to country. In the United States, daylight savings time is changed at 02:00 local time in each time zone. In the European Union, it is coordinated so that all EU countries change at 01:00 GMT for all time zones at once. This keeps time zones always an hour apart at the expense of it changing at different local times for each time zone.
- When time is recorded for an event, such as a parking spot being vacated, it is essential for analytics that the time is as close to the actual occurrence as possible. In practice, though, the time available for analytics can be the time the event occurred, the time the IoT device sent the data, the time the data was received, or the time the data was added to your data warehouse.

IoT analytics challenges

- **Problems with space**
- IoT devices are located in multiple geographic locations. Different areas of the world have different environmental conditions. Temperature variations can affect sensor accuracy. You could have less accurate readings in Calgary, Canada than in Cancun, Mexico, if cold impacts your device.
- Elevation can affect equipment such as diesel engines. If location and elevation is not taken into consideration, you may falsely conclude from IoT sensor readings that a Denver-based fleet of delivery trucks is poorly managing fuel economy compared to a fleet in Indiana. Lots of mountain roads can burn up some fuel!
- Remote locations may have weaker network access. The higher data loss could cause data values for those locations to be underrepresented in the resulting analytics.
- Many IoT devices are solar powered. The available battery charge can affect the frequency of data reporting. A device in Portland, Oregon, where it is often cloudy and rainy will be more impacted than the same device in Phoenix, Arizona, where it is mostly sunny.
- There are also political considerations related to the location of the IoT device. Privacy laws in Europe affect how the data from devices can be stored and what type of analytics is acceptable. You may be required to anonymize the data from certain countries, which can affect what you can do with analytics.

IoT analytics challenges

- **Data quality**
- Constrained devices means lossy networks. For analytics, it often results in either missing or inconsistent data. The missing data is often not random. As mentioned previously, it can be impacted by the location. Devices run on a software, called firmware, which may not be consistent across locations. This could mean differences in reporting frequency or formatting of values. It can result in lost or mangled data.
- Data messages from IoT devices often require the destination to know how to interpret the message being sent. Software bugs can lead to garbled messages and data records.
- Messages lost in translation or never sent due to dead batteries result in missing values. The conservation of power often means not all values available on the device are sent at the same time. The resulting datasets often have missing values, as the device sends some values consistently every time it reports and sends some other values less frequently.

Analytics challenges

- Analytics often requires deciding on whether to fill in or ignore the missing values. Either choice may lead to a dataset that is not a representative of reality.
- As an example of how this can affect results, consider the case of inaccurate political poll results in recent years. Many experts believe it is now in near crisis due to the shift of much of the world to mobile numbers as their only phone number. For pollsters, it is cheaper and easier to reach people on landline numbers. This can lead to the over representation of people with landlines. These people tend to be both older and wealthier than mobile-only respondents.
- The response rate has also dropped from near 80% in the 1970s to about 8% (if you are lucky) today. This makes it more difficult (and expensive) to obtain a representative sample leading to many embarrassingly wrong poll predictions.
- There can also be outside influences, such as environment conditions, that are not captured in the data. Winter storms can lead to power failures affecting devices that are able to report back data. You may end up drawing conclusions based on a non-representative sample of data without realizing it. This can affect the results of IoT analytics – and it will not be clear why.

IoT Devices and Networking Protocols

IoT devices

- **Healthcare**
  - vital signs monitored by low-power wireless sensors where the data can be analyzed remotely.
  - **Eg -** Proteus Digital Health- a start up – developed pill-sized ingestible sensors. A digestible sensor pill that, in combination with a sensor patch worn on skin, monitors when and how often patients are taking their pills.
- **Manufacturing**
  - **Industrial Internet of Things** (**IIoT**) or **Industry 4.0** . A wide variety of manufacturing industries are implementing IoT devices and sensors.
- **Transportation and logistics**
  - **Telematics devices -** GPS tracking devices that logistics providers can attach to their vehicles to monitor and analyze routes, fuel economy, and detect accidents.
  - Devices can monitor driver behavior and send reports of speeding or harsh braking to the home office. They also read any fault codes reported by the vehicle and transmit the information instantly to fleet managers, so problems can be addressed before a major breakdown occurs.

Geotab GO7 telematics device. Source: Geotab

IoT devices

- **Retail**
  - **Radio Frequency Identification** (**RFID**) tags and sensors to optimize stocking and monitor movements inside warehouses.
  - Eg >- Extreme Networks is partnering with the New England Patriots and their home stadium to monitor visitor traffic patterns. The stadium uses the data to understand where customers gather and price advertisements and product placements appropriately using traffic heat maps generated from the IoT sensor data.
- **Oil and gas**
  - In combination with **edge analytics** on sensor data, companies can do things such as optimize the oil well plunger lift cycle. The analytic processing occurs at multiple locations at the boundary of the network or the *edge*, versus occurring at a centralized location.
- **Home automation or monitoring**
  - Eg - Ecobee Smart Wi-Fi thermostats connect with battery operated wireless sensors placed in different rooms throughout a home. It tracks not only Heating Ventilation and Air Conditioning (HVAC) but temperatures reported by each sensor. It can not only balance temperatures in the rooms, but it can also notify you when the furnace is not behaving properly.
- **Wearables**
  - Along with cell phones, there is a plethora of devices such as the Fitbit step tracking wristband, Garmin heart rate monitoring chest band, and Nike smart shoes. IoT- enabled umbrella that sends weather alerts to your phone

Connectivity protocols

When the available power is limited

The following protocols are specifically designed to address **low power constraints.** They are usually associated with **lower complexity and bit transfer rates** in the supported IoT devices.

- **Bluetooth Low Energy (also called Bluetooth Smart)**
- **6LoWPAN**
- **ZigBee**
- **NFC**
- **Sigfox**

When power is not a problem

These protocols are more about **speed and high bit rates** . Power constraints are less of a concern. They usually require more complexity in the supported IoT devices.

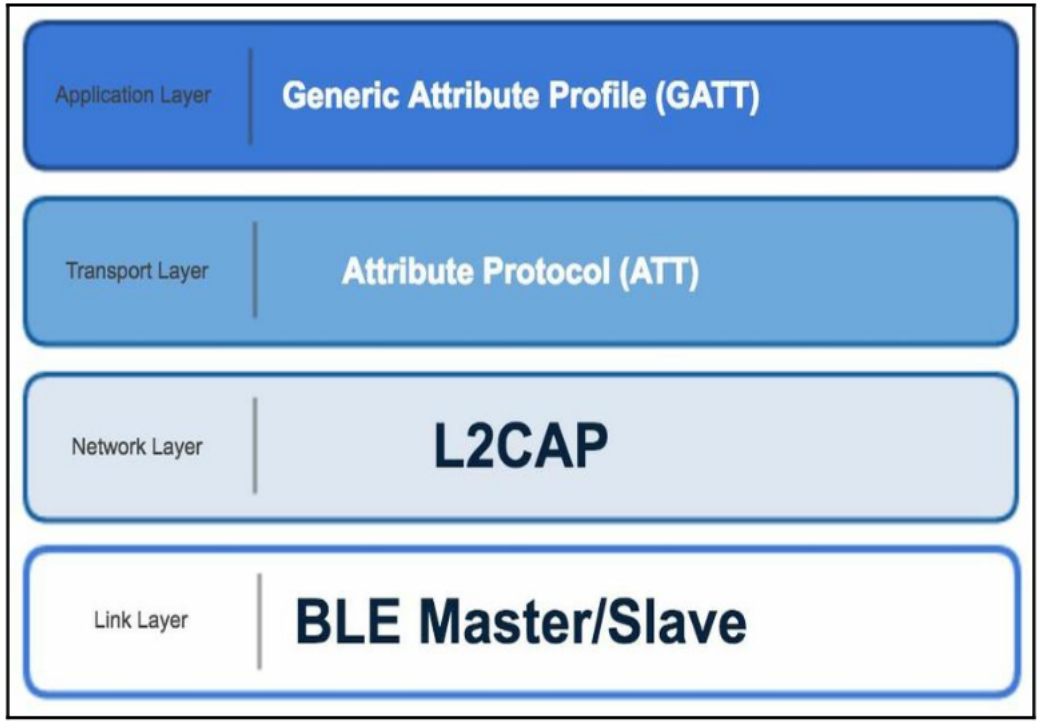
- **Wi-Fi**
- **Cellular (4G/LTE)**

Connectivity protocols (when the available power is limited)

- designed to address low power constraints.
- They are usually associated with lower complexity and bit transfer rates in the supported IoT devices.

Bluetooth Low Energy (also called Bluetooth Smart)

- BLE device is either a master or a slave in the network.
- A master acts as an advertiser in the connection while the slave acts as receiver. A master can have several slaves connected to it while a slave can link to only one master.
- The network group involving the slaves connected to a master is called a **piconet**.
- the master device passes data over the internet using one of the data messaging protocols through a gateway device that is both part of the bluetooth network and also linked to the internet.
- Gateway translates data reported from BLE nodes into the appropriate format for data messaging over the internet.
- It is commonly an unconstrained device that does not need to worry much about power.



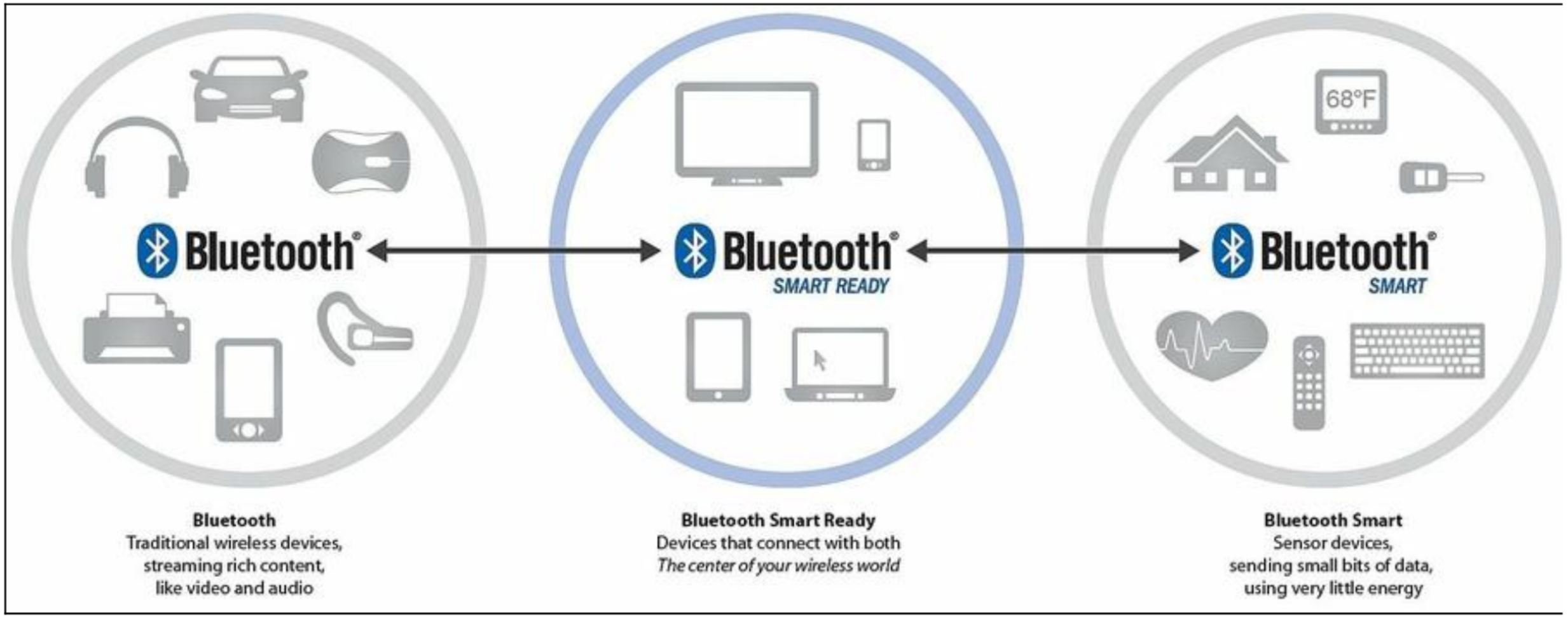
Logical Link Control and Adaptation Protocol (L2CAP)

- L2CAP offers segmentation and reassembly services for large packets to be transmitted across Bluetooth links and also allows for the multiplexing of higher-layer protocols and services.

Functions of L2CAP Layer

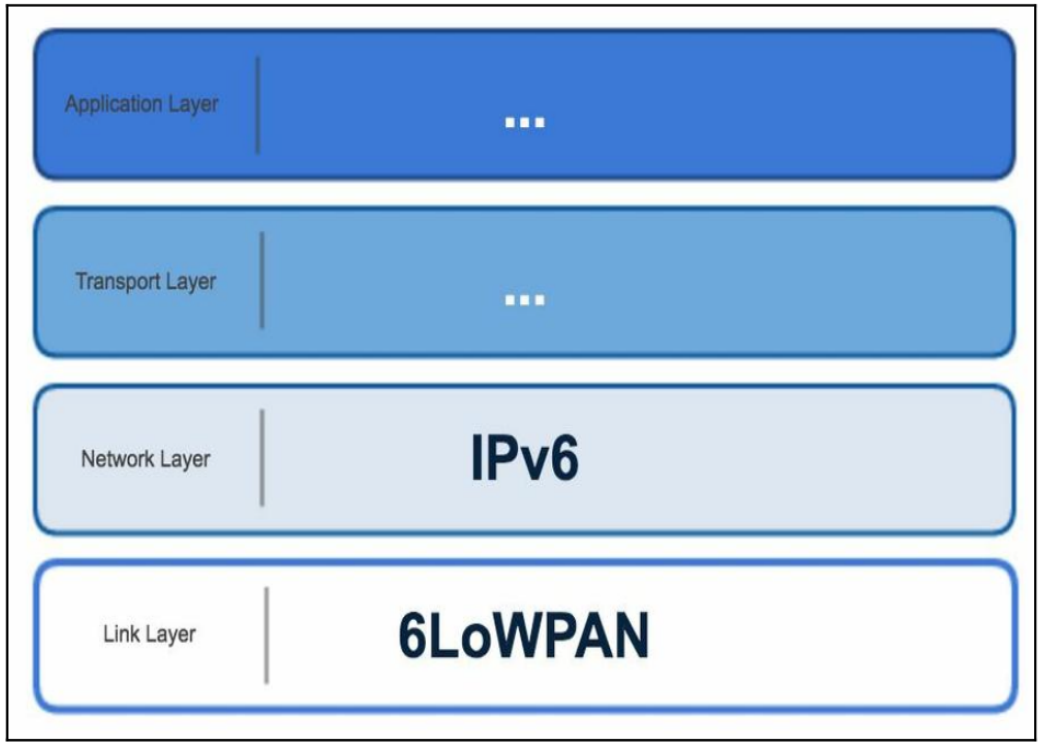
1. It accepts packets of up to 64 kB from upper layers and breaks them down into smaller frames for transmission. These frames are then reassembled into packets again at the receiving end.
2. L2CAP manages the multiplexing and demultiplexing of multiple packet sources. Once a packet has been reassembled, the L2CAP layer determines which upper-layer protocol it should be passed to, such as RF communication or telephony.
3. L2CAP also handles quality of service requirements, both during link establishment and normal operation. This includes negotiating maximum payload size to prevent large-packet devices from overwhelming small-packet devices. This is important because not all devices can handle the maximum packet size of 64 kB. The L2CAP layer corresponds with the 802 Data Link Layer which is typically responsible for transmission, framing, and error control over a particular link. As such, L2CAP overlaps the link controller task and the control end of the baseband, including error checking and correction.

Bluetooth Low Energy (also called Bluetooth Smart)



6LoWPAN

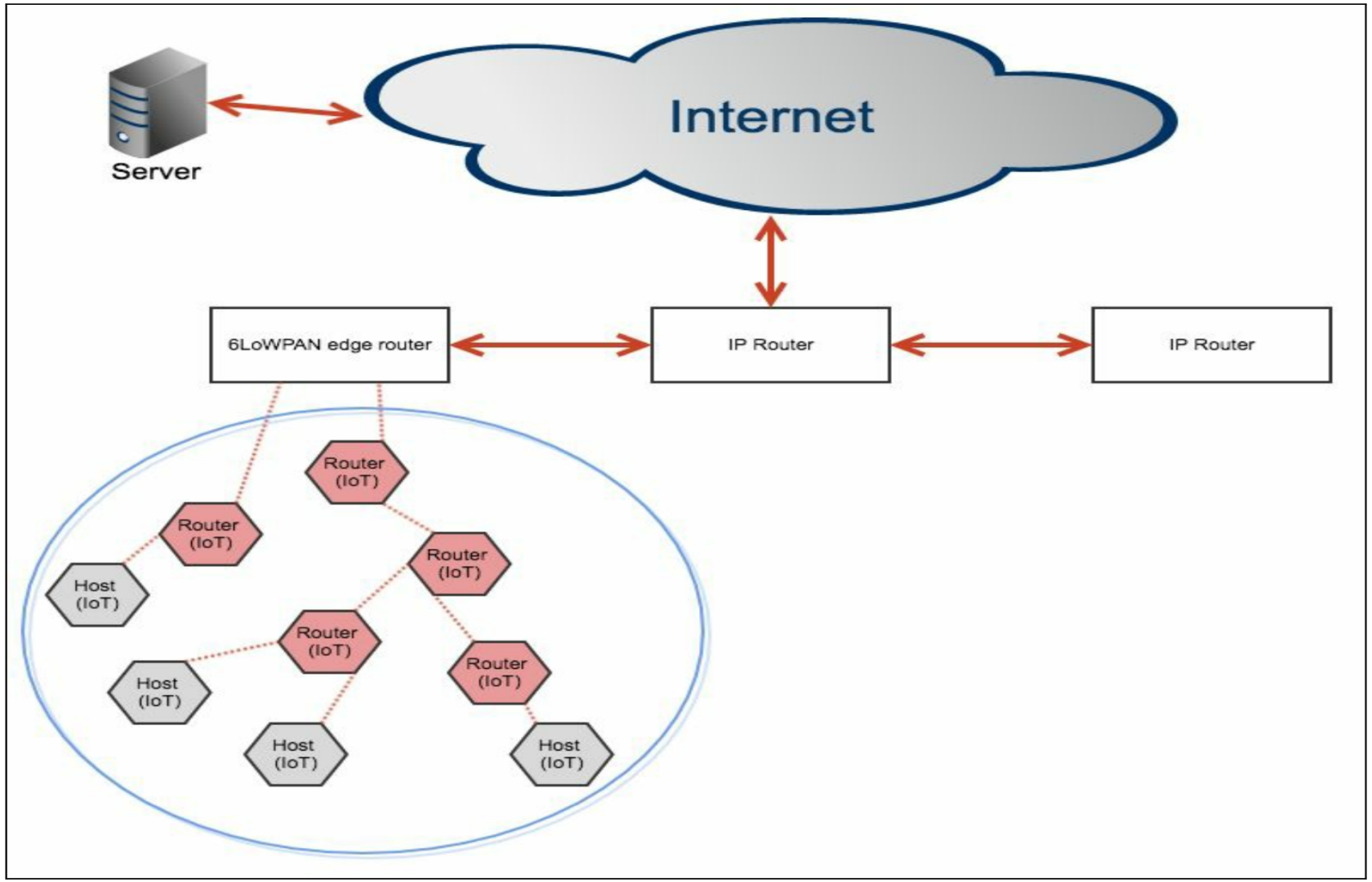
- **IPv6 over Low-Power Wireless Personal Area Networks** (**6LoWPANs**) works as an adaptation layer in the data link portion of the protocol stack. It adapts wireless standard IEEE 802.15.4 devices to communicate using IPv6.
- 6LoWPAN optimizes the IPv6 datagram transmission over low-power and lossy networks.
  - One of the ways it does this is through **header compression**. This reduces the addressing information in the data packet down to a few bytes.
- Security can be implemented with **Advanced Encryption Standard** (**AES**)-128 encryption at the link layer. **Transport Layer Security** (**TLS**) encryption can also be used at the transport layer.
- Mesh style networks are supported. Devices inside the network use stateless auto configuration where they generate their own IPv6 addresses.



6LoWPAN

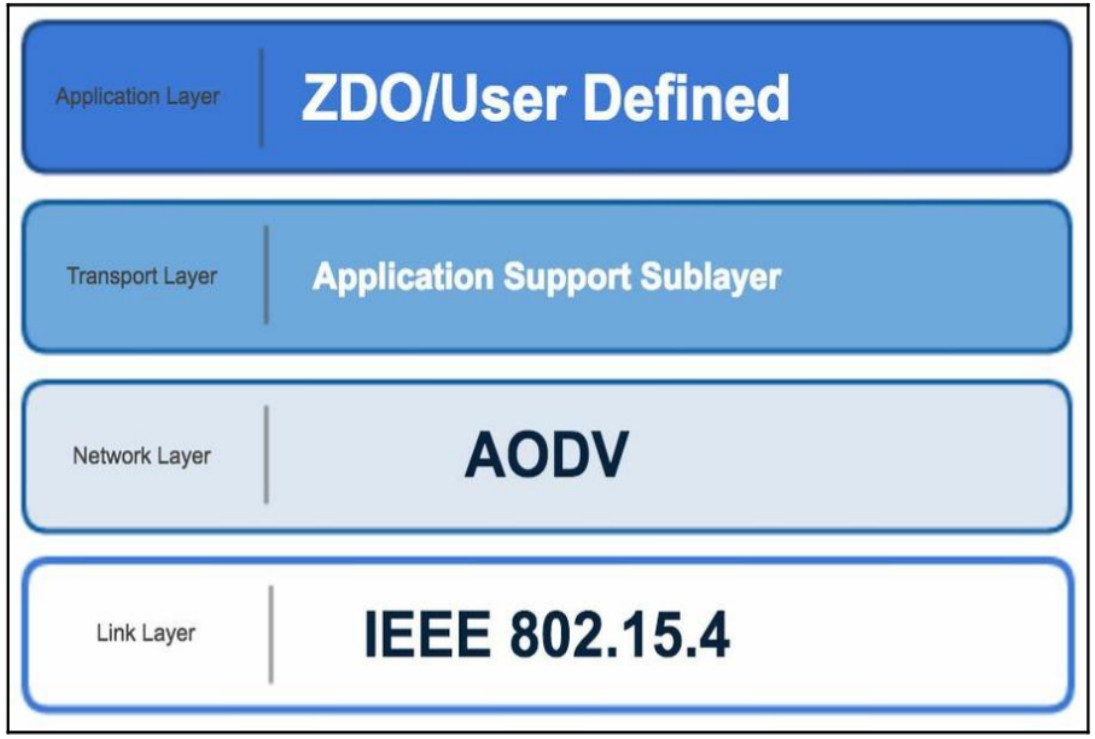
- The 6LoWPAN mesh network is connected to the IPv6 internet through an edge router. The edger router has three jobs:
  - Local data exchanges between devices in the 6LoWPAN
  - Data exchange between the internet and 6LoWPAN devices
  - The creation and maintenance of the 6LoWPAN wireless network
- 6LoWPAN networks are connected to other networks using common IP-based routers. This connectivity can be through any type of linkage such as Wi-Fi, Ethernet, or cellular.
- This makes it simpler to connect to the wider internet than other connectivity options, which require *stateful* gateway devices in order to communicate to IP networks. Stateful gateway devices are more complicated as they need to remember the communication state and status for each IoT device.
- There are two types of device inside a 6LoWPAN network, **hosts** and **routers**. Routers can forward datagrams to other devices. Hosts are simply endpoints. They can operate in sleepy states, periodically waking up and checking in with their parent device (router) for new data.

6LoWPAN


Server
Internet
6LoWPAN edge router
IP Router
IP Router
Router (IoT)
Router (IoT)
Router (IoT)
Host (IoT)
Router (IoT)
Router (IoT)
Host (IoT)
Host (IoT)
Host (IoT)

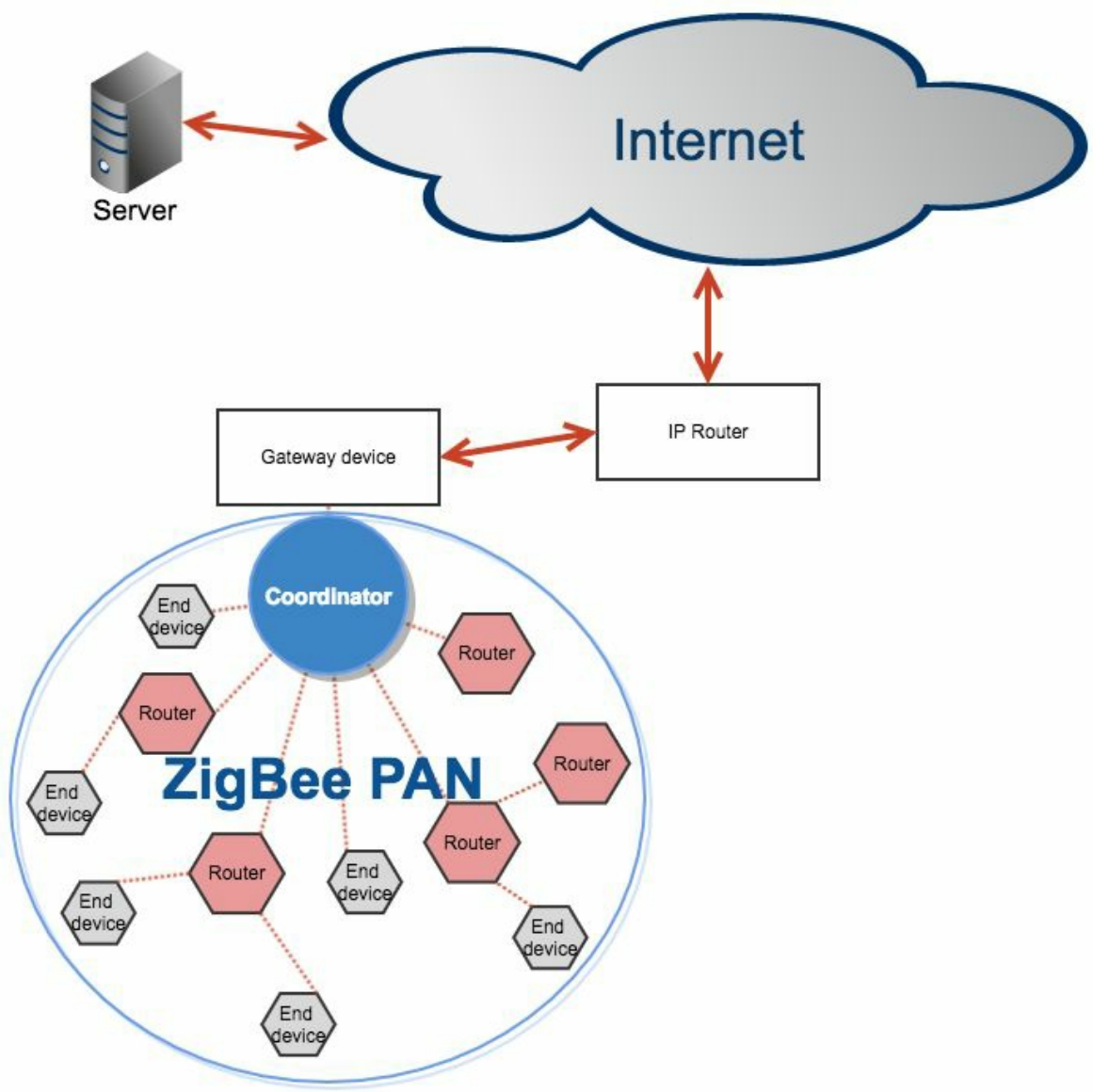
ZigBee

- It covers multiple layers of the protocol stack. It is the network protocol, the transport layer, and an application framework that is the underpinnings of a customer developed application layer.
- It has security builtin as a requirement and works using small, low power digital radios.
- It is a self- organizing wireless mesh-style network.
- The range is short, up to 100 meters line of sight, but it can travel long distances by passing data through a mesh network of other ZigBee devices.
- ZigBee was first released as a standard in 2003. It is intended to be simpler and cheaper than other wireless protocols.
- Data rates are low and devices tend to be simple. ZigBee standards require at least a two-year battery life for a device to qualify.
- The data transmission is periodic and intermittent.
- There are different categories of ZigBee device. A fully functioning device can act as a network coordinator and operates in both star and mesh topologies. A reduced functionality device can only communicate with a network coordinator and works in a star topology.



ZigBee

- These devices form a **Personal Area Network** (**PAN**). Network interaction times can be very fast. A device can join a network in 30 ms and change from sleeping to active in 15 ms.
- There are three types of ZigBee devices in a network:
- A **coordinator device** that initiates network formation. There is one and only of these for each network. It may also act as a router once the network is formed. It serves as the trust center and is the repository for security keys.
- A **router device** is optional. It can join with a coordinator or another router device and acts as a multi-hop intermediary in routing of messages.
- An **end device** has just enough functionality to communicate with its coordinator. It is optional and cannot participate in the routing of messages.



ZigBee

- The ZigBee networking routing protocol is reactive. It establishes a route path on demand instead of proactively such as IP.
- The network protocol used is **Ad hoc On-Demand Distant Vector** (**AODV**) which finds a destination by broadcasting to all its neighbors, which then send a requests to all their neighbors, until the destination is found.
- Networks are self-healing. ZigBee networks are secured by 128 bit AES encryption. There is often a gateway device that is part of the ZigBee network and also linked to the internet. It translates data into the appropriate format for messaging over the internet. It can also convert ZigBee network routing protocol into IP routing protocol.
- The gateway is commonly an unconstrained device that does not need to worry much about power.

ZigBee

- **Advantages of ZigBee**
  - **Reliable and robust**: Networks are self-forming and self-healing.
  - **Low power requirements**: Devices can operate for years without needing a new battery. Solar power could last even longer.
  - **Global**: The wireless protocol it is built on is available globally.
  - **Security**: Encryption is part of the standard instead of an optional add on.

- **Disadvantages of ZigBee**
  - **Not free**: It costs $3,500 USD at the time of writing to license the standard
  - **Low data rates**: It is not designed to support higher data rates
  - **Star network is limited**: The coordinator device supports up to 65,000 devices

Common use cases

Home automation
Utilities monitoring and control Smart lighting
Building automation
Security systems
Medical data collection

NFC

- **Near Field Communication** (**NFC**) allows two devices to communicate when brought within very short distances of each other - about 4 cm. It is a set of communication protocols that offer low speeds and simple setup. There is a variety of NFC protocol stacks depending on the type and use case.
- NFC uses electromagnetic induction between two loop style antennas when devices exchange information. Data rates are around 100-400 kbit/s. It operates over radio frequency band 13.56 MHz, which is unlicensed and globally available.
- There is usually a full NFC device, such as a smartphone, and an NFC tag, such as a chip reader. All full NFC devices can operate in three modes:
- **NFC peer-to-peer** : Devices exchange information on the fly.
  NFC card emulation : Smartphones can act like smart cards in this mode, allowing mobile payments or ticketing.
  NFC reader/writer : Full NFC devices can read information stored in inexpensive NFC tags, such as the type used on posters or pamphlets. You have probably used this at a conference to scan your attendee badge.

NFC

- NFC tags are usually read-only, but it is possible for them to be writeable. There is an initiator device and a target device in NFC communications. The initiator device creates an RF field that can power a target device. This means NFC tags do not need batteries in what is called passive mode. Unpowered NFC tag devices are very inexpensive, just a few pennies.
- There is also an active mode where each device can communicate with the other by alternatively generating its own field to read data, then turning it off to receive data. This mode typically requires both the devices to have a power supply.
- For data records that eventually end up in an analytics dataset, the full NFC device would act as a gateway, translate the message, and then communicate over a different network protocol stack. In the case of a smartphone, it would take the NFC tag data and create its own data message to send over 4G/LTE for a destination on the internet.

Common use cases

NFC is commonly used in the following applications:
Smartphone payments:
Apple pay
Host Card Emulation (HCE) - Android supported
Smart posters
Identification cards
Event logging
Triggering a programmable action

Sigfox

- Sigfox is a company based in France that has developed a cellular style, ultra low power, long-range networking technology. It is intended for small messages that are sent infrequently. Data rates are very low at 100 bits per second (tops). The signals used can travel up to 25 miles, and a base station can handle up to a million nodes. Similar to cellular, it is not intended to be for private networks, as it requires coverage in the area to use it.
- Sigfox devices can be very low cost due to low complexity and power requirements. Power requirements could be as low as one thousandth of equivalent cellular communications. It has been used in inexpensive, but crucial, things such as smoke detectors and a fitness tracking collar for dogs (seriously).
- The current coverage includes the entirety of France and Spain, several other pockets in Europe, and some metropolitan areas in North America.

Connectivity protocols (when power is not a problem)

These protocols are more about speed and high bit rates. Power constraints are less of a concern. They usually require more complexity in the supported IoT devices.

Wi-Fi

- Wi-Fi is any wireless network that follows IEEE 802.11 specifications. It is ubiquitous and global. Provided you have not been hiding in a cave for the last decade, you know it well. Many IoT devices use it to connect to the internet, so it is useful to review how it works.
- To connect to a **Wi-Fi Local Area Network** (**WLAN**), a device needs a network interface controller. This may be a separate controller card or simply integrated as part of the chipset inside the device. Communication uses Ethernet-style data packets over pre-identified radio communication bands. Packet delivery is not guaranteed and uses best effort delivery mechanism. Network protocol layers higher up in the stack, such as TCP or UDP, can offer guarantees of data delivery on top of the Wi-Fi standards.

Wi-Fi


Application Layer
...
Transport Layer
...
Network Layer
IP
Link Layer
LLC/MAC

Wi-Fi

- IoT devices connect to a Wi-Fi access point that itself is (usually) connected to a wired Ethernet network that has internet access.
- Wi-Fi offers very fast data rates but requires more power to support the constant back and forth communications. The universality of it makes it attractive for IoT devices. New standards such as 802.11ah are designed for low power scenarios. You should expect Wi-Fi to continue to be a heavily used connectivity option for IoT.
- **Common use cases**
- Wi-Fi is commonly used in the following applications:
- Home automation
- Amazon dash button
- Home security
- Manufacturing plant machine monitoring

Cellular (4G/LTE)

- Cellular data operates over radio waves. The fourth generation networking technology, 4G, is an IP- based networking architecture handling both voice and data. **Long Term Evolution** (**LTE**) is a 4G technology. It handles data rates up to 300 Mbit/s.
- LTE can handle fast moving mobile devices (such as mobile phones or telematics units). It can handle devices moving out of range of one network and into another without disruption. Due to higher power needs, it usually requires either constant power or frequent battery charges. LTE supports different categories of services.

Cellular (4G/LTE)

Cellular (4G/LTE)

- A newer category, **LTE Category 0** (**LTE Cat 0**), is designed for IoT requirements. It has lower data rates, capped at 1 Mbit/s maximum. LTE Cat 0 also reduces bandwidth ranges, which allows for significant complexity reduction. It is still able to operate on all existing LTE networks and supports low power constrained IoT devices.
- Cellular coverage is not available in all areas, though. There are still some places where you will not find a signal. For analytics, this means that if your IoT device moves around, it could venture into an area without a signal. Data could be lost in this case, unless it is buffered until the signal is recovered.

Common use cases

Cellular is commonly used in the follow applications:

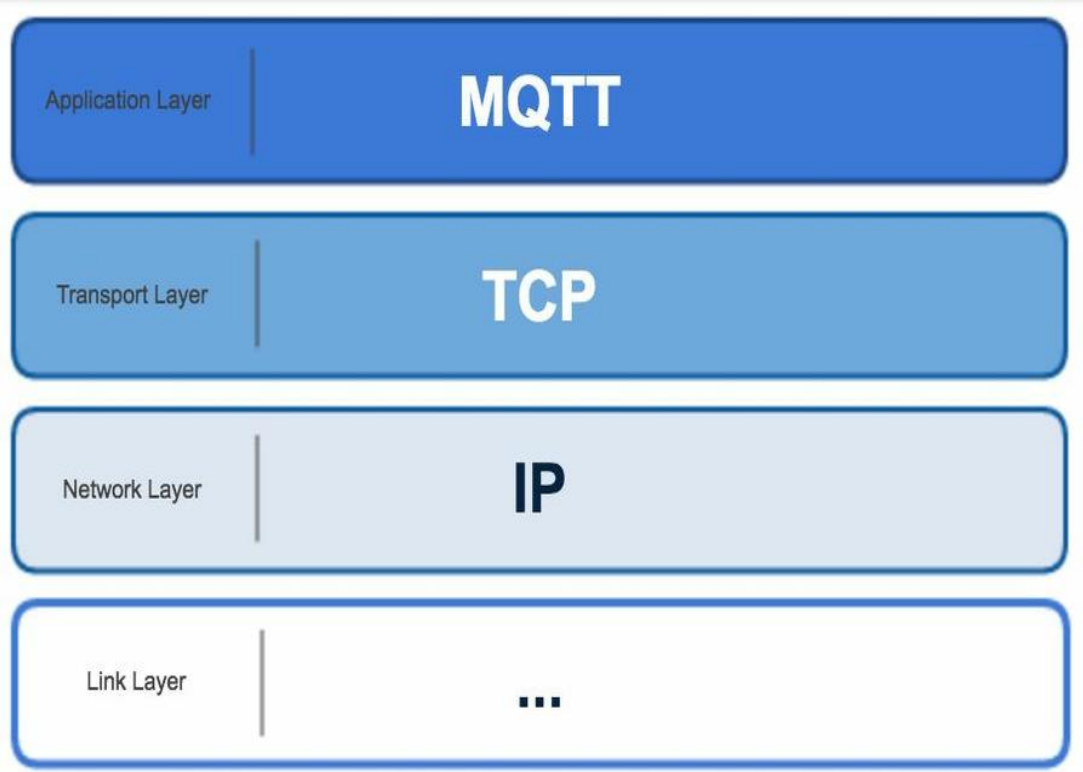
- Freight tracking
- Vehicle connectivity (telematics)
- Smartphones (obviously)
- Remote equipment monitoring

IoT networking data messaging protocols

- There are many strategies that IoT networked devices use to transfer data messages. Although connectivity and data messaging can sometimes blend together, we will discuss them separately for simplicity.
- **Message Queue Telemetry Transport (MQTT)**
- **Hyper-Text Transport Protocol (HTTP)**
- **Constrained Application Protocol (CoAP)**

Message Queue Telemetry Transport (MQTT)

- MQTT is the most common data messaging protocol associated with IoT.
- It is supported by all the major cloud infrastructure providers (AWS, Microsoft, and Google). And it is most likely the protocol that is being used to deliver your data.
- It was designed for minimal power loss and minimal bandwidth requirements.
- It originated to support remote oil and gas use cases over satellite communication networks.
- It translated well into the broader IoT world as it developed in recent years.



Message Queue Telemetry Transport (MQTT)

- Message Queuing Telemetry Transport, or MQTT, is a communications protocol designed for Internet of Things devices with extremely high latency and restricted low bandwidth.
- Message Queuing Telemetry Transport is a perfect protocol for machine-to-machine (M2M) communication since it is designed specifically for low-bandwidth, high-latency settings.
- **MQTT** is a simple, lightweight messaging protocol used to establish communication between multiple devices.
- It is a TCP-based protocol relying on the publish-subscribe model. This communication protocol is suitable for transmitting data between resource-constrained devices having low bandwidth and low power requirements

Message Queue Telemetry Transport (MQTT)

Publish-Subscribe Model

- This model involves multiple clients interacting with each other, without having any direct connection established between them.
- All clients communicate with other clients only via a third party known as a Broker.

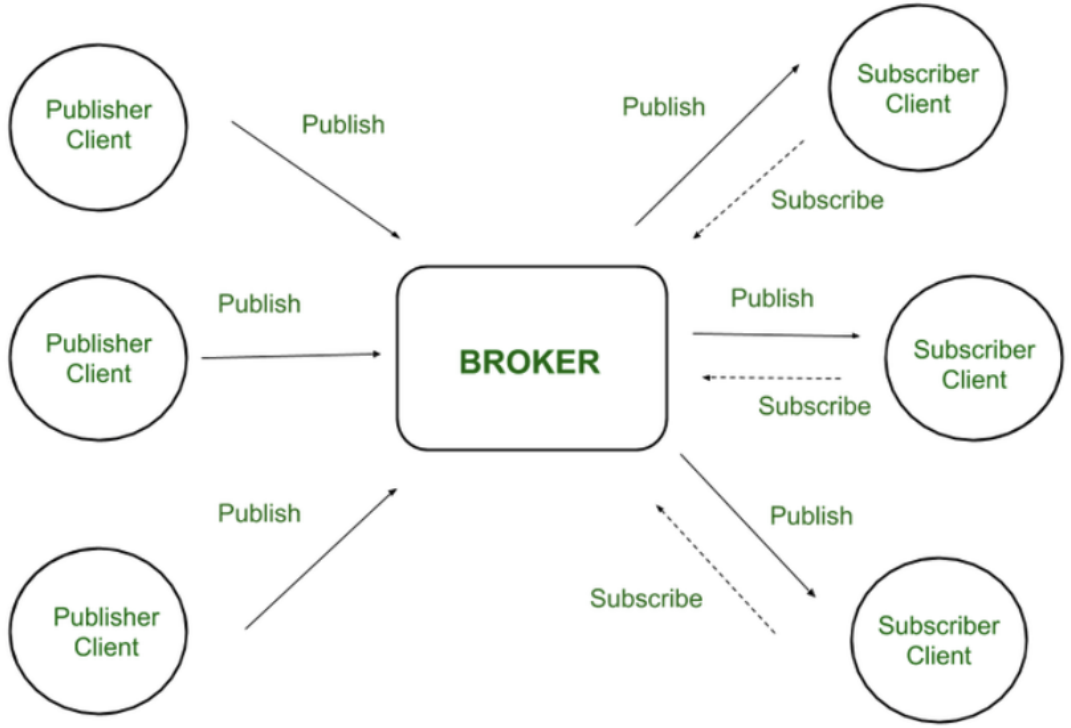
MQTT Client and Broker

- Clients publish messages on different topics to brokers.
- The broker is the central server that receives these messages and filters them based on their topics.
- It then sends these messages to respective clients that have subscribed to those different topics.
- The heart of any publish/subscribe protocol is the MQTT broker. A broker can handle up to thousands of concurrently connected MQTT customers, depending on how it is implemented.
- All communications must be received by the broker, who will then sort them, ascertain who subscribed to each one, and deliver the messages to the clients who have subscribed.
- All persistent customers' sessions, including missed messages and subscriptions, are likewise kept by the Broker.

Message Queue Telemetry Transport (MQTT)

Publish-Subscribe Model

- Client that has subscribed to a specific topic receives all messages published on that topic.
- Here the broker is central hub that receives messages, filters them, and distributes them to appropriate clients, such that both message publishers, as well as subscribers, are clients.



Publish-Subscribe Model

Working of MQTT

- MQTT's publish/subscribe (pub/sub) communication style, which aims to maximise available bandwidth, is an alternative to conventional client-server architecture that communicates directly with an endpoint.
- Client who transmits the message (the publisher) and the client or clients who receive it (the subscribers) are not connected in the pub/sub paradigm. Third parties–the brokers–manage the relationships between the publishers and subscribers because they don't communicate with one another directly.
- Publishers and subscribers, which denote whether a client is publishing messages or has subscribed to receive messages, are examples of MQTT clients. The same MQTT client can be used to accomplish these two features. A publish occurs when a client or device want to submit data to a server or broker.
- The term "subscribe" refers to the reversal of the procedure. Several clients can connect to a broker under the pub/sub paradigm and subscribe to subjects that interest them.

Working of MQTT

- When a broker and a subscribing client lose contact, the broker will store messages in a buffer and send them to the subscriber whenever the broker is back up and running. The broker has the right to cut off communication with subscribers and send them a cached message containing publisher instructions if the publishing client abruptly disconnects from the broker.
- “Publishers send the messages, subscribers receive the messages they are interested in, and brokers pass the messages from the publishers to the subscribers.
- MQTT clients, such as publishers and subscribers, can only speak with MQTT brokers. Any device or programme that runs a MQTT library can be a MQTT client, ranging from microcontrollers like the Arduino to entire application servers housed in the cloud.

Message Queue Telemetry Transport (MQTT)

- Differences between MQTT queue (topics) and traditional message queues are as follows:
  Topics do not have to be created ahead of time : They are flexible and can be created on the fly. Traditional message queues have to be created and named explicitly before they can receive messages.
  Topics do not keep all messages until they are consumed : Traditional queues keep everything until it has been consumed by a subscriber. Topics just replace old messages with new ones.
- **Topics can have multiple subscribers receiving the same message** : Traditional message queues are designed for a message to be consumed by one client.
- **Messages do not require a subscriber** : Traditional message queues require someone to consume them. Otherwise, it will eventually fill up and be unable to accept more items.
- MQTT is intended to be very light overhead. A device can switch on periodically, connect to the broker, send a message, then go back to sleep without worrying about if the subscriber is able to receive it right away.
- However, most implementations keep a persistent connection. Communication tends to be near real time in practice.

Characterstics of MQTT

- **Lightweight:**
- **Publish-Subscribe Model:**
- **Quality of Service (QoS) Levels:** QoS levels range from 0 to 2, providing various stages of reliability and message transport guarantees, relying at the utility necessities.
- **Retained Messages:** MQTT lets in agents to store retained messages on topics, making sure that new subscribers acquire the maximum latest message posted on a subject right now after subscribing. This characteristic is beneficial for fame updates and configuration settings.
- **Last Will and Testament (LWT):** MQTT clients can specify a Last Will and Testament message to be posted by way of the broker in the occasion of an sudden consumer disconnect. This function affords a mechanism for detecting patron failures and dealing with them gracefully.
- **Security:** MQTT helps various protection mechanisms, consisting of Transport Layer Security (TLS) encryption and authentication mechanisms which include username/password and consumer certificates. These capabilities make certain the confidentiality, integrity, and authenticity of messages exchanged over MQTT connections.

Advantages of MQTT

- This model is not restricted to one-to-one communication between clients. Although the publisher client sends a single message on specific topic, broker sends multiple messages to all different clients subscribed to that topic.
- Similarly, messages sent by multiple such publisher clients on multiple different topics will be sent to all multiple clients subscribed to those topics. Hence one-to-many, many-to-one, as well as many-to-many communication is possible using this model.
- Also, clients can publish data and at the same time receive data due to this two-way communication protocol. Hence MQTT is considered to be bi-directional protocol. The default unencrypted MQTT port used for data transmission is 1883. The encrypted port for secure transmission is 8883.

- Lightweight protocol that is quick to create and allows for efficient data transport
- Minimal data packet usage, resulting in low network usage
- Effective data dispersion
- The effective use of remote sensing and control
- Prompt and effective message delivery
- Minimises power consumption, which is beneficial for the linked devices, and maximises network capacity.
- Data transmission is quick, efficient, and lightweight because MQTT messages have small code footprint. These control messages have a fixed header of size 2 bytes and payload message up to size 256 megabytes.

Disadvantages of MQTT

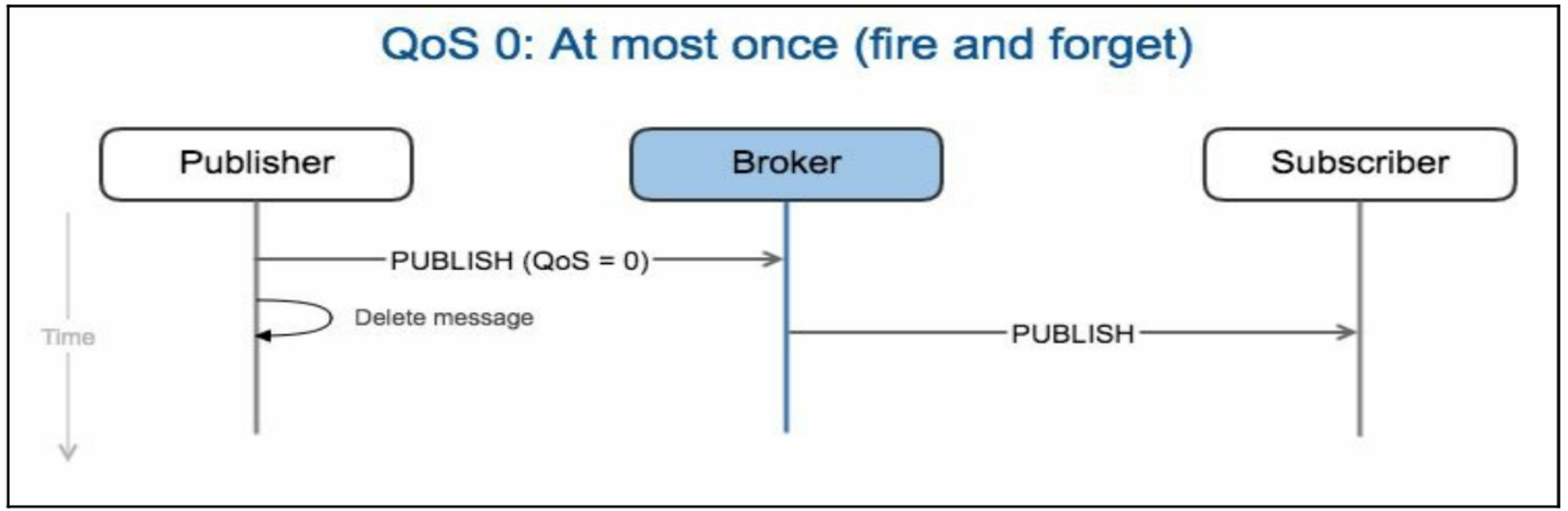
- **It operates over TCP** : TCP was designed for devices that had more memory and processing power than many of the lightweight, power constrained IoT devices have available to them. TCP requires more handshaking to set up communication links before any messages can be exchanged. This increases wake-up and communication times, which affects the long-term battery consumption. TCP connected devices tend to keep sockets open for each other with a persistent session. This adds to power and memory requirements.
- **Centralized broker can limit scale** : The broker can affect scalability as there is additional overhead for each device connected to it. The network can only grow as large as the local broker hub can support it. This puts a limit on expansion for each hub and spoke group.
- **Broker single point of failure** : It can also be a single point of failure in the network. A common situation is a broker device that is plugged into a wall socket with several publishing devices that are battery powered. In the event of a power failure, the publishing devices would keep operating but the broker would be offline. The network would be useless until the power is resumed.
- **Security**: MQTT is unencrypted by default. This makes it natively unsecured and requires you to take additional steps and absorb some overhead to make sure TLS/SSL is implemented. If not, any communication over MQTT, including username and password, is open to hackers.
- Building an internationally **scalable** MQTT network is challenging.

QoS levels

- QoS is the MQTT terminology for message delivery reliability requirements.
- The trade-off is between bandwidth used and reliability of message delivery. More back and forth messaging is required to ensure higher reliability, which increases bandwidth requirements for the same message packet. MQTT gives you some options to choose between.
- *QoS is set separately for both the ends of the communication. There is a QoS that is set for the publisher to broker messaging connection and a separate QoS that is set for the broker to subscriber connection. Often, QoS is the same for both, but not always.*

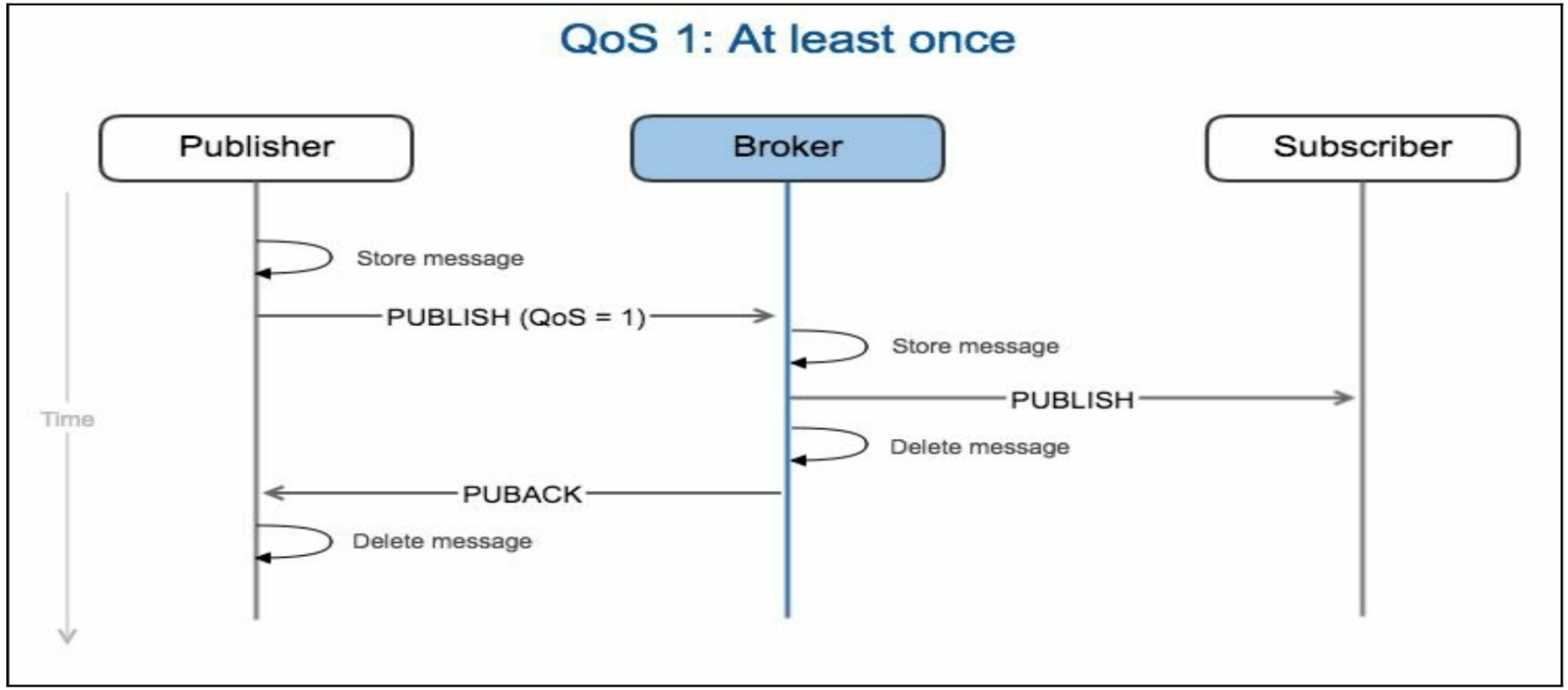
QoS 0 - Fire and forget – Minimal Level

- Messages are sent to the broker without confirmation that it has been sent on by the broker to subscribers. It still has all the guarantees of the TCP protocol, as do all forms of standard MQTT.
- there is minimal communication overhead, which translates into minimal power requirements. Once sent to the broker, the message is deleted on the device (publisher). The broker immediately sends on to subscribers that have an open connection. Unlike the other QoS levels, the message is not stored for offline subscribers.
- QoS 0 is often used when there is a stable connection and disruption is unlikely, like in the case of a wired connection. It is also used when power constraints are more important than message delivery. resulting data is acceptable even with some messages lost, or the frequency of messages is high enough



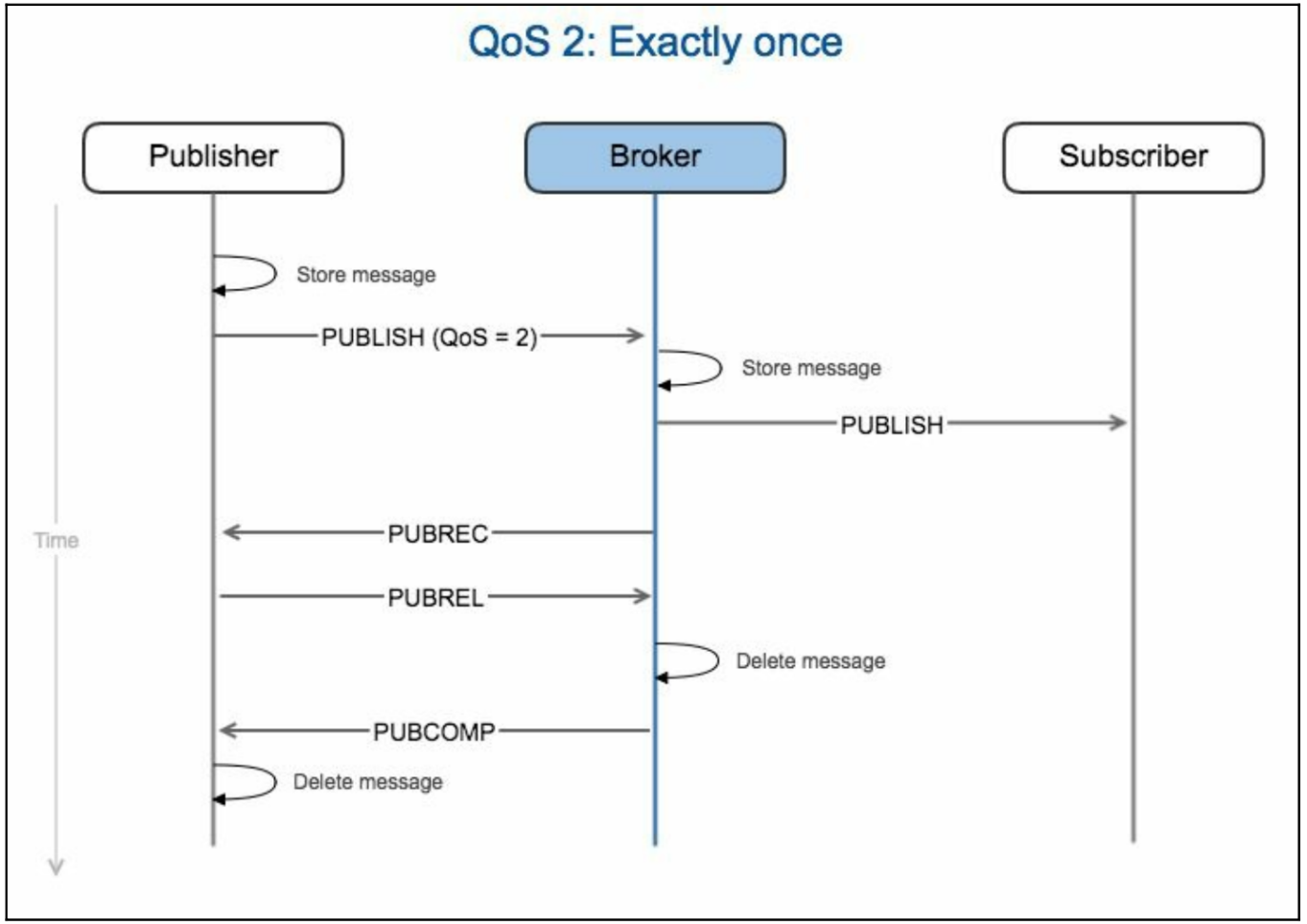
QoS 1 - Deliver once - Commonly Used

- **Deliver once -** I don't care if you send it 10 times, just make sure they get it.
- It guarantees the message is delivered at least once. Duplicate messages need to be handled either on the subscriber side or in some post processing. There is a higher communication overhead than QoS 0 but only half as much back and forth as QoS 2.
- QoS 1 is used when delivery needs to be guaranteed and the use case can handle duplicate messages



QoS 2 - Deliver Exactly once

- QoS 2 guarantees the message is delivered and delivered only once.
- The state of the communication needs to be stored by both the sides until full confirmation of delivery is communicated back to the publisher.
- This requires both a higher level of power on the device and a larger memory footprint in order to store the more complex state. It also takes a bit longer to complete QoS 2 communications.
- QoS 2 is used when delivery needs to be guaranteed and the end application or subscribing client is unable to handle duplicate messages.



MQTT

- **Last Will and Testament (LWT)**
- This creatively named feature helps MQTT deal with lossy networks and enhances scalability. It is a message that can be stored on the broker in case the publisher drops off the network unexpectedly.
- It stores state and purpose including the commands it published and its subscriptions. When it drops out, the broker notifies all subscribers of the LWT. When it returns, the broker sends it back to its prior state.

- **Tips for analytics**
- Make sure the time the device records the data is tracked along with the time the data is received. You can monitor delivery times to diagnose issues and keep a close eye on information lag times. For example, if you notice the delivery time steadily increases just before you get a data loss, it is probably something in the networking and not the device that is causing the issue.
- Since MQTT enables subscribers and publishers to be offline if necessary, the time the message is received could be very different from when it was originated. This can affect the predictive modeling accuracy if the time received becomes significantly distant from when the event being analyzed actually occurred. If connections are persistent, this should be a non issue.

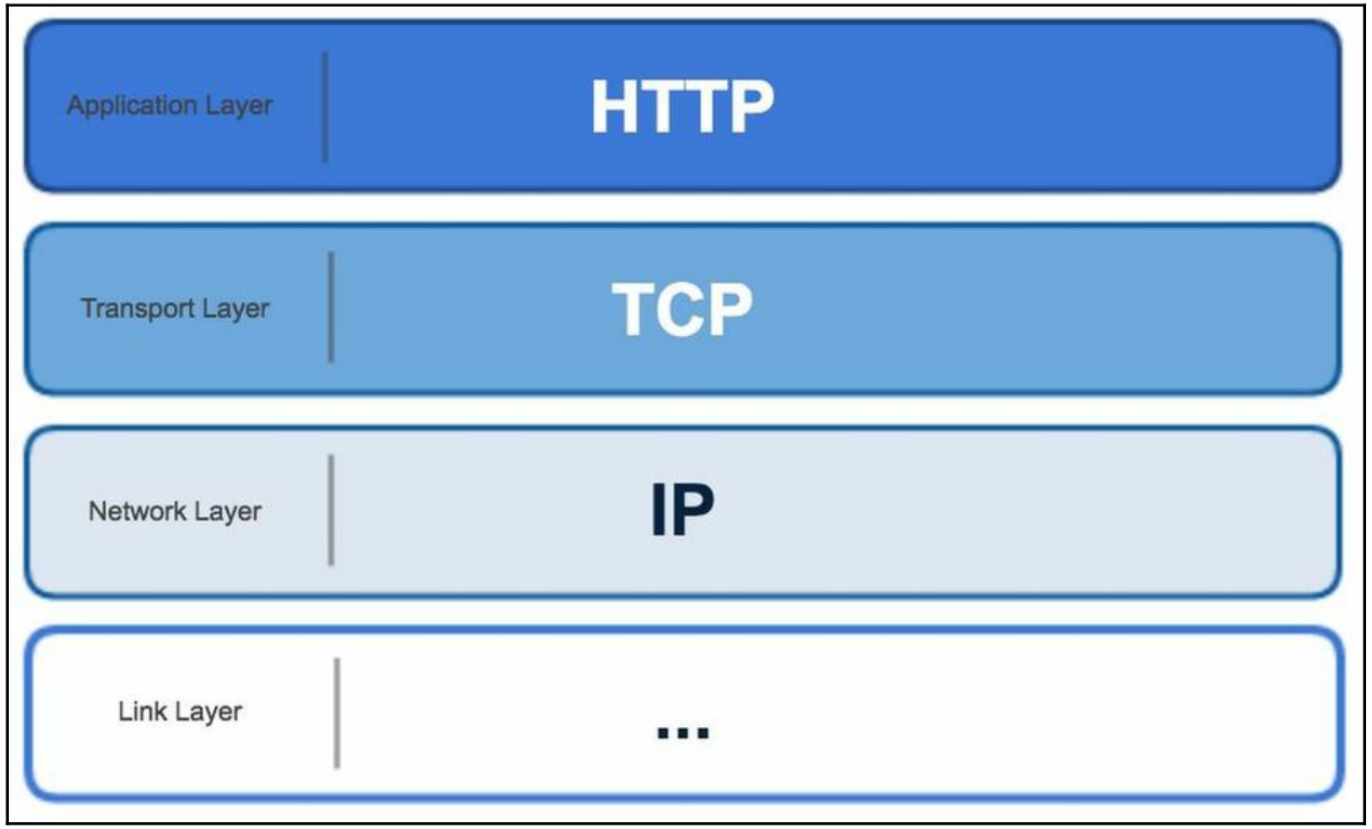
MQTT

Common use cases

- Casino gambling devices
- Home healthcare
- Automotive telematics
- Oil and gas remote monitoring
- Inventory tracking
- Cargo trailer tracking
- Monitoring of energy usage

Hyper-Text Transport Protocol (HTTP)

- Web pages use HTTP protocol to exchange data.
- HTTP is a character-based protocol.
- It operates at the application layer and conforms to RESTful principles.



Representational State Transfer (REST) principles

- REST allows internet-enabled devices to inter-operate with each other using a uniform **Application Programming Interface** (**API**).
- A device or server that complies to this is called RESTful. It assumes interactions are client/server-based and are stateless.
- The following table shows the key commands supported by the HTTP REST API Interface:

HTTP REST API
HTTP POST: Adds a new resource
HTTP PUT: Updates a specific resource by replacing it with an updated version
HTTP GET: Reads a specified resource
HTTP PATCH: Partially updates a resource in a collection
HTTP DELETE: Deletes a resource

HTTP and IoT

- In many cases, HTTP is not ideal for IoT applications.
- The latency is not predictable, and it often depends on polling to detect state changes.
- It is a text-based protocol, which means message size tends to be large. This adds power needs and complexity overhead, for IoT devices to communicate using it.
- However, it is a mature standard and in wide use. This makes it an established and well supported interface.
- Connections are established from a client device to a server device and remain open until the communication is completed.
- Delivery is guaranteed and data message receipts are acknowledged at every step of the process.
- It operates on top of TCP and is reliable.

Advantages of HTTP

- **Reliability** : Message delivery is guaranteed and acknowledged.
- **Ubiquity** : HTTP is used all over the place and is very easy to implement.
- **Ease of implementation** : If you can connect to the internet, you can use HTTP anywhere in the world. No special hardware or software is required.

Disadvantages of HTTP

- **Higher power needs** : Communications have frequent back and forth interactions, connections need to be sustained, and plain text results in larger message sizes. All of this requires more power to support.
- **The IoT device complexity** : The device needs enough memory and CPU power to support the TCP protocol and high level HTTP RESTful APIs.

Constrained Application Protocol (CoAP)

- CoAP is point-to-point communication over **User Datagram Protocol** (**UDP**).
- It has low overhead requirements and was specifically developed for low-power devices operating over the internet.
- It was designed to work on microcontrollers with as low as 10 KiB of RAM and needs only 100 KiB for the operating codes.

Constrained Application Protocol (CoAP)

- CoAP was created by the **Internet Engineering Task Force** (**IETF**) to address the needs of power constrained IoT devices. It is also supported by the Eclipse foundation as an open standard.
- CoAP follows a client/server architecture and is a one-to-one communication convention. It does allow for some multicast capabilities, which are early in development at the time of this writing.
- CoAP is similar to HTTP in that it is a document transfer protocol and interoperates with the RESTful web. It is different from HTTP, in that it was designed for applications where networks are low- powered and often lossy.
- CoAP operates on UDP protocol where data packets are much smaller than HTTP, which uses larger TCP packets. Clients and servers communicate without established connections. Information is transferred using self-contained datagrams. Datagrams are the core of the UDP protocol and each datagram contains all necessary information for routing without any reliance on previous exchanges between client and server.
- It may help to think of a datagram like communicating with someone over mail (old school snail mail). The envelope has a delivery address and a return address and the payload would be the letter inside, but this is hidden from the mail man. You send regular letters to your cousin in Poland. She sends regular letters to you.
- You have no guarantee that all your letters will be delivered or delivered in the same order that you sent them. It is your responsibility, once letters are received, to open them up, put them in the right order, and figure out if there are any missing. If you find one to be missing, you would handle mailing your cousin to request a resend of it.
- CoAP works in the same manner where the sending and receiving applications play the role of you and your cousin. Applications need to be programmed appropriately to handle the required level of delivery reliability.

CoAP - Advantages

- **Reduced power requirements** : It operates over UDP, which requires minimal overhead for communications. It also allows faster wake up times and extended sleepy states. Taken together, this means batteries last longer for IoT devices.
- **Smaller packet size** : Another advantage of UDP is small packet sizes. This leads to faster communication cycles. Again, this allows batteries to last longer.
- **Security** : Like MQTT, this is on both the advantage and disadvantage lists. When **Datagram Transport Layer Security** (**DTLS**) is employed over UDP, communication is encrypted and secure. Even though there is some additional overhead required to implement this, you can and should use it.
- **Asynchronous communication option** : Clients can request to observe a device by setting a flag. The server (IoT device) can then stream state changes to the client as they happen. Either side can cancel the observe request.
- **IPv6 based** : It was designed from the beginning to support IPv6. This also allows for a multicasting option.
- **Resource discovery** : Servers can provide a list of resources and media types. The client can then review and discover what is available.

CoAP - Disadvantages

- **Message unreliability** : UDP does not guarantee the delivery of datagrams. CoAP adds a method to request a confirmation acknowledgement to confirm the message was received. This does not verify that it was received in its entirety and decoded properly.
- **Standards are still maturing** : CoAP is still evolving, although there is a lot of market momentum behind it. It is likely to mature quickly as use becomes more widespread.
- **NAT issues** : **Network Address Translation** (**NAT**) devices are commonly used in enterprise networks and cloud environments. CoAP can have problems communicating with devices behind a NAT since the IP can be dynamic over time.

- **Security** : Like MQTT, CoAP is unencrypted by default. This makes it natively unsecure and you need to take additional steps to make sure communication is not open to hackers.