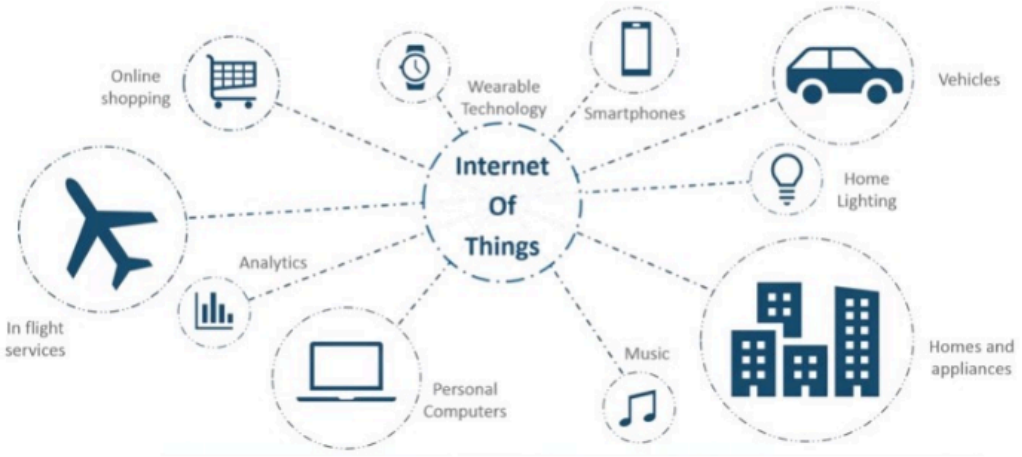
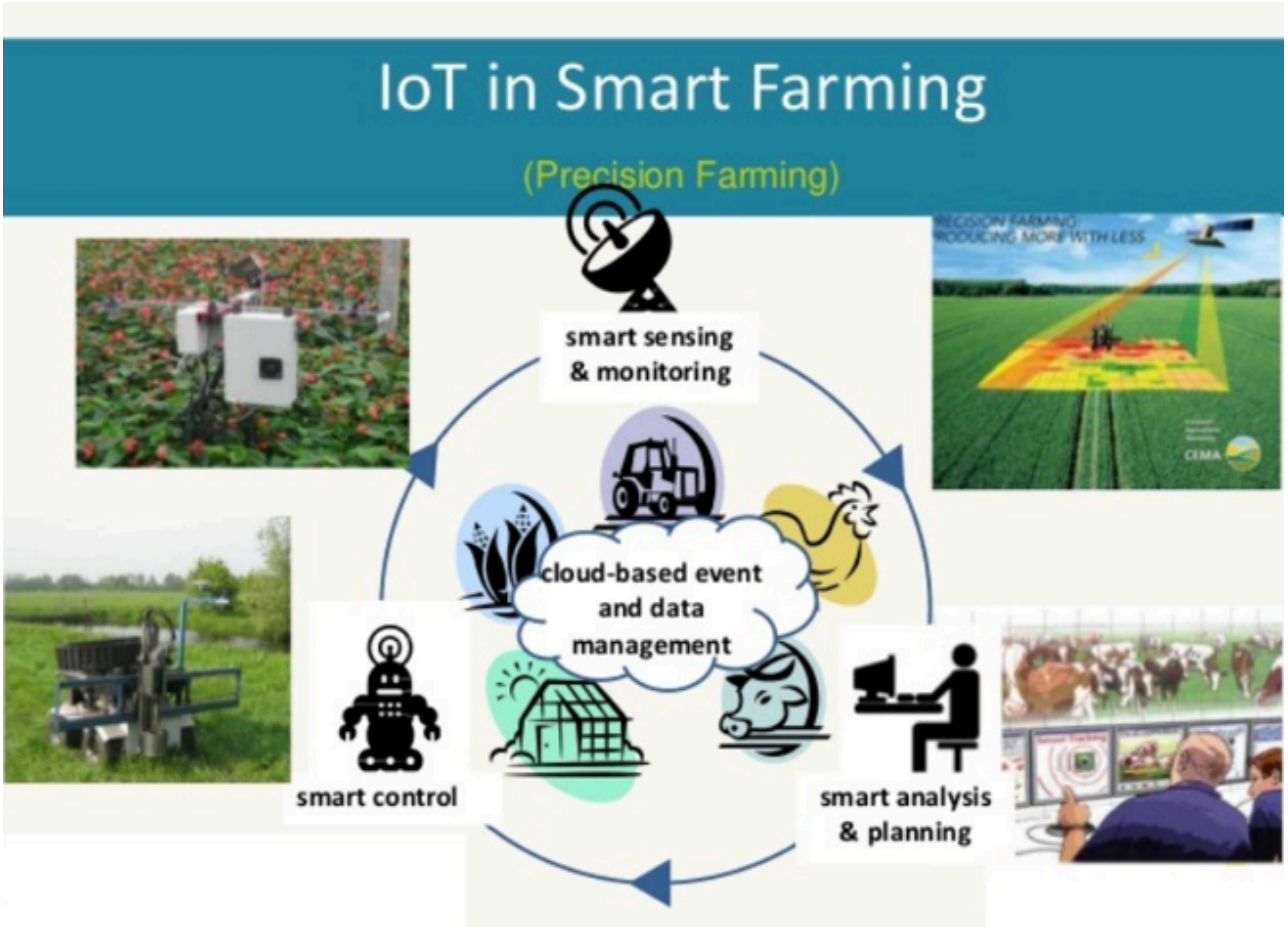
ALL DIAGRAM IOT

Consolidated List of Diagram, Placeholders with Headings

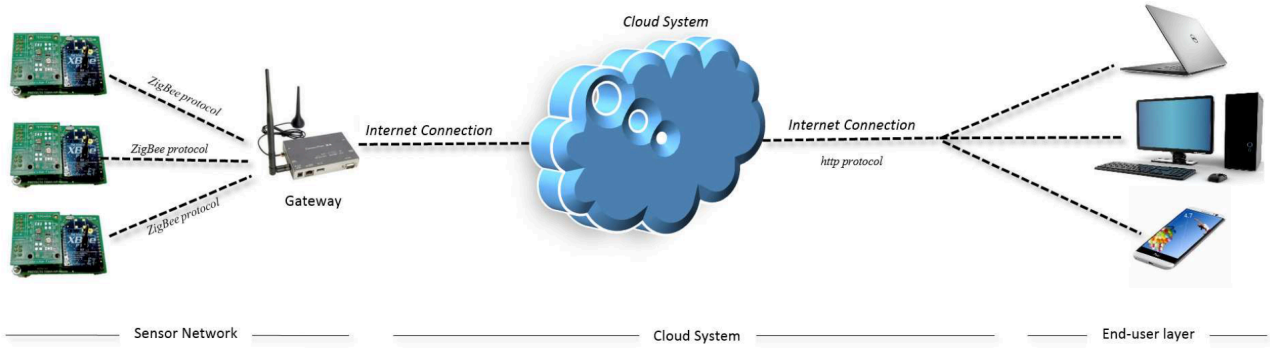
Unit 1: Introduction to IoT

- **1. What is IoT (Internet of Things)?**
  - 

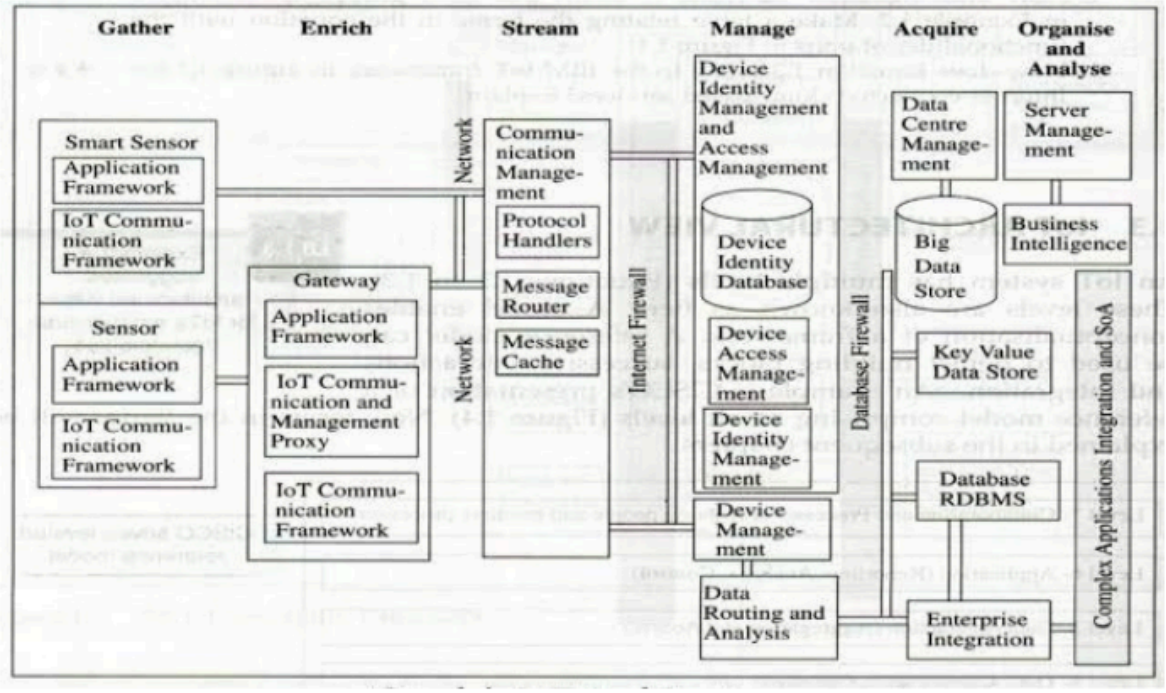
    (General IoT concept diagram)
- **2. Why IoT Matters?** (Things that do both - Farming Example)
  - 

- **3. History of IoT** (LG Smart Fridge)

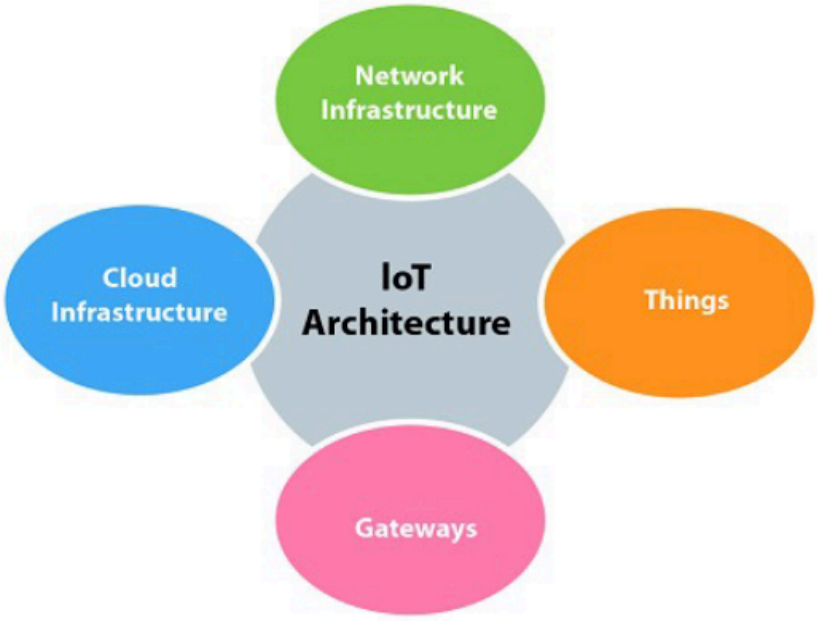
- **4. How does an IoT System Actually Work?**



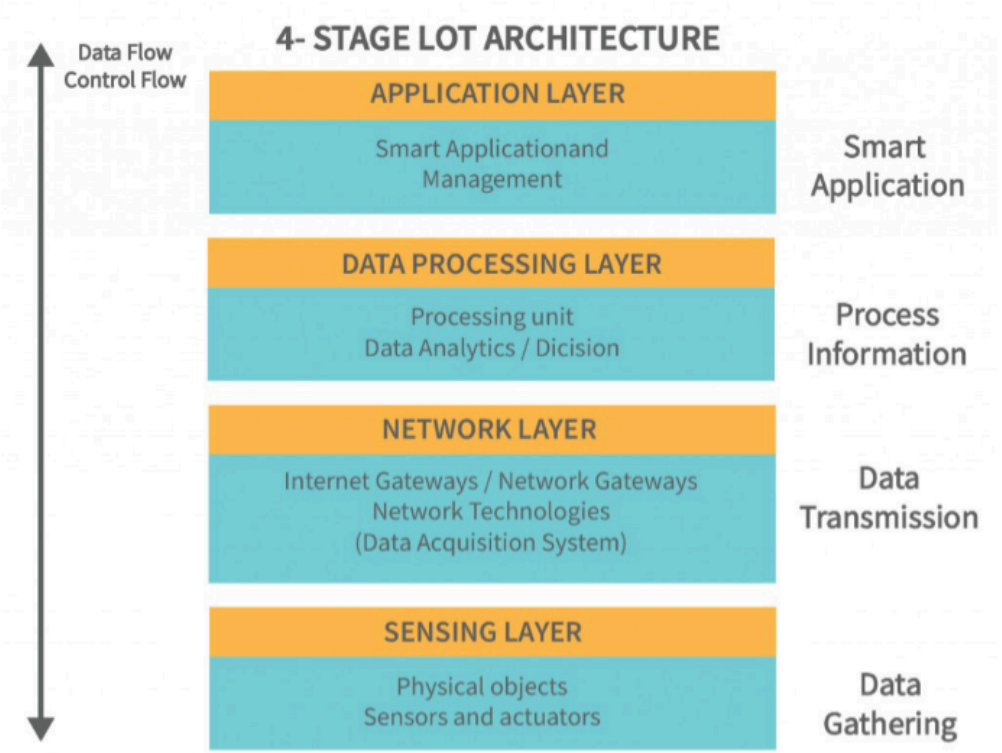
- **8. IoT Conceptual Framework** (Oracle's Suggested Architecture)



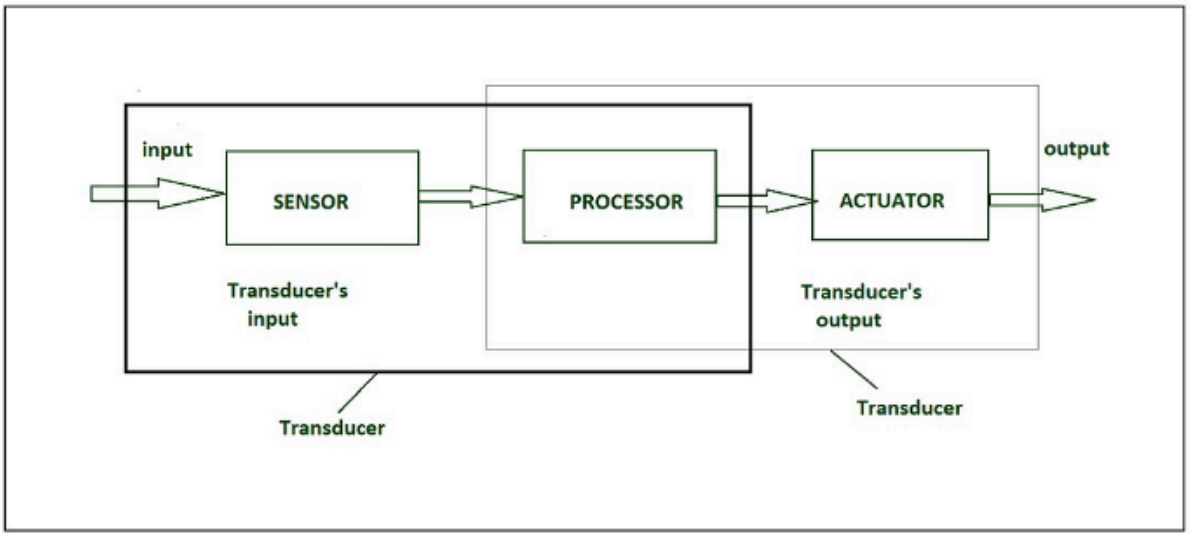
- **9. What is IoT Architecture?** (General components)



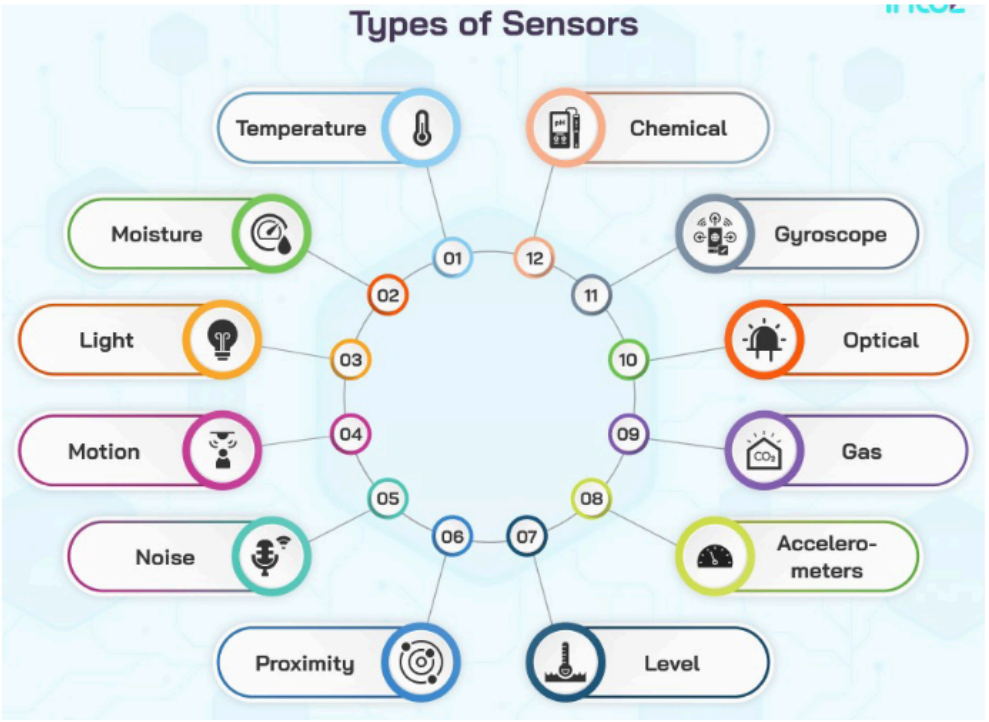
- **10. Different Layers of IoT Architecture (Standard 4-Layer Model)**



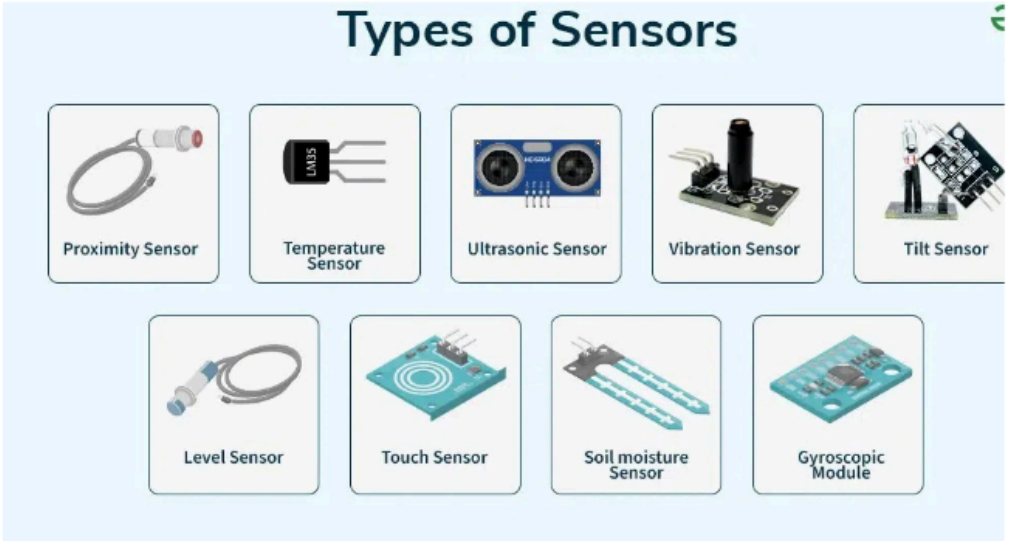
- **11. Sensors** (Sensor-Processor-Actuator flow)



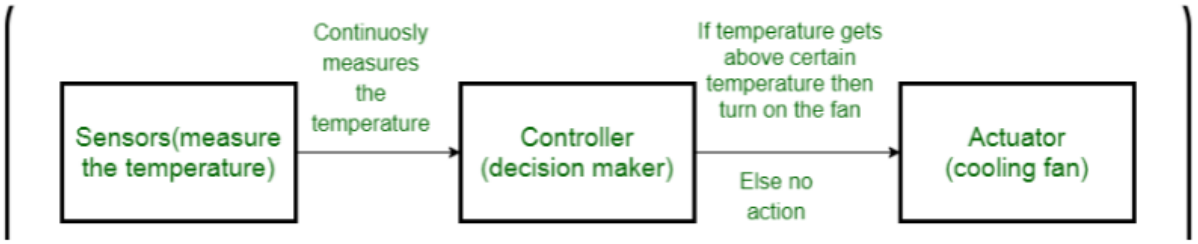
- **11. Sensors** (Types of Sensors - circular diagram)



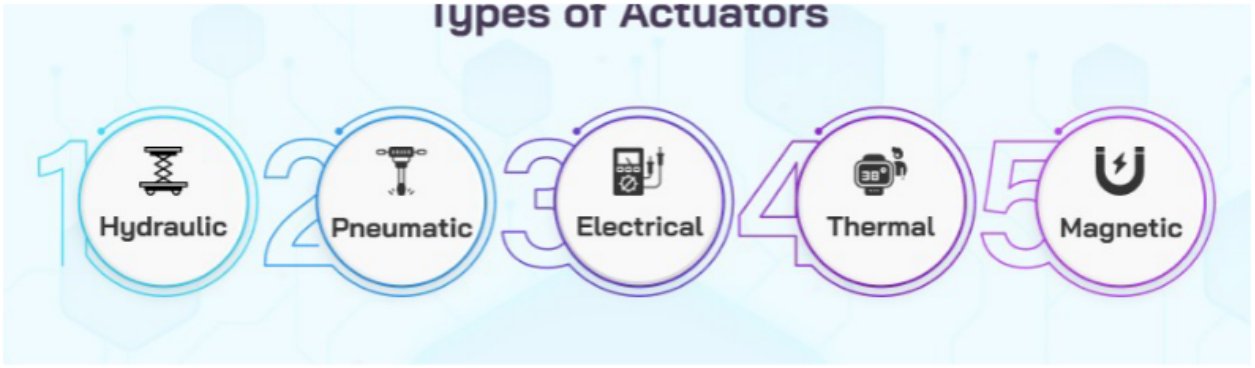
- **11. Sensors** (Types of Sensors - grid diagram)



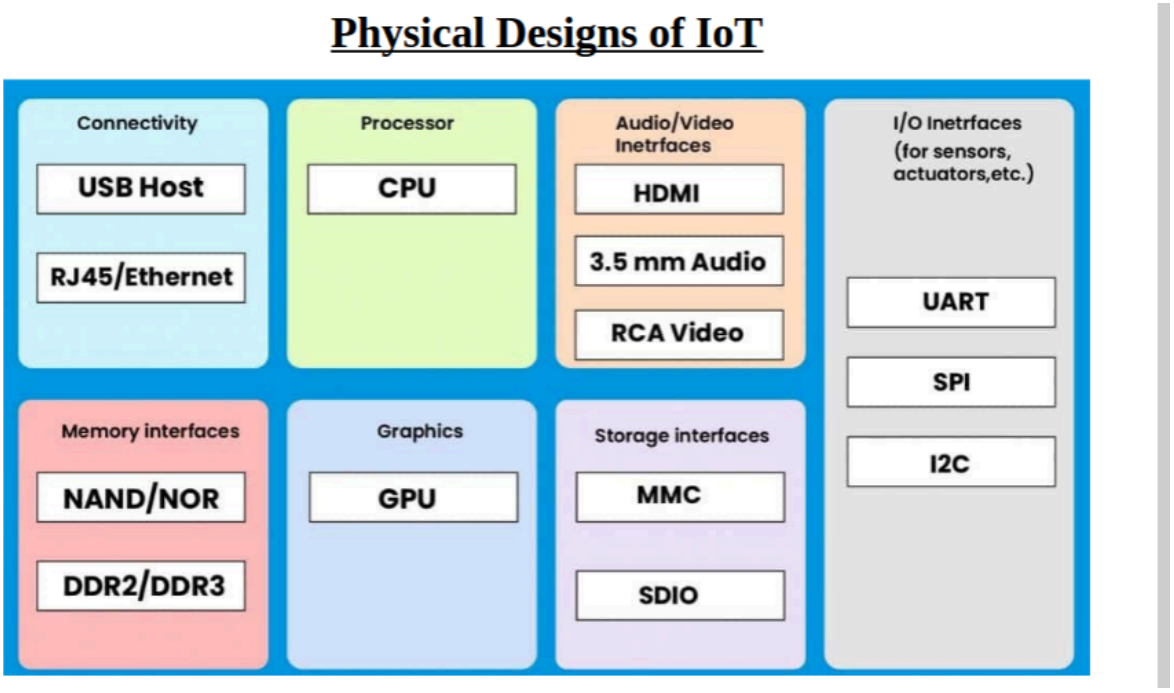
- **12. Actuators** (Actuator diagram: Sensors -> Controller -> Actuator)



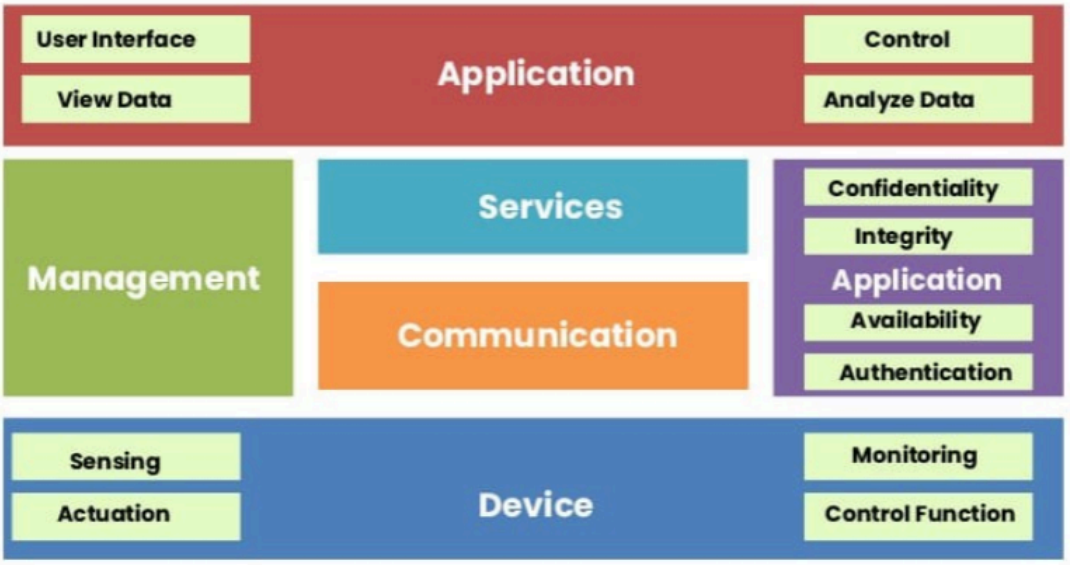
- **12. Actuators** (Types of Actuators - circular icons)



- **13. Physical Design of IoT (Generic Components)**



- **14. Logical Design of IoT** (Functional Blocks)

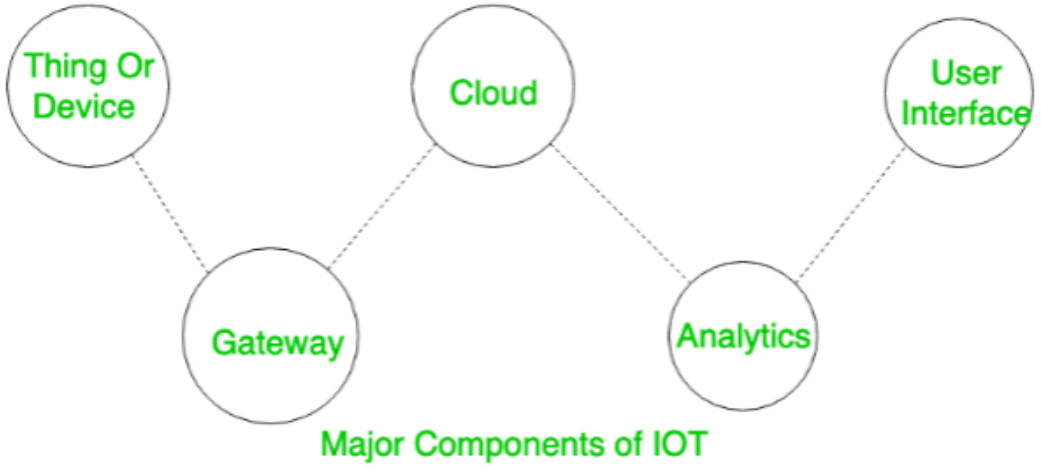


- **15. Difference Between Physical and Logical Design of IoT**

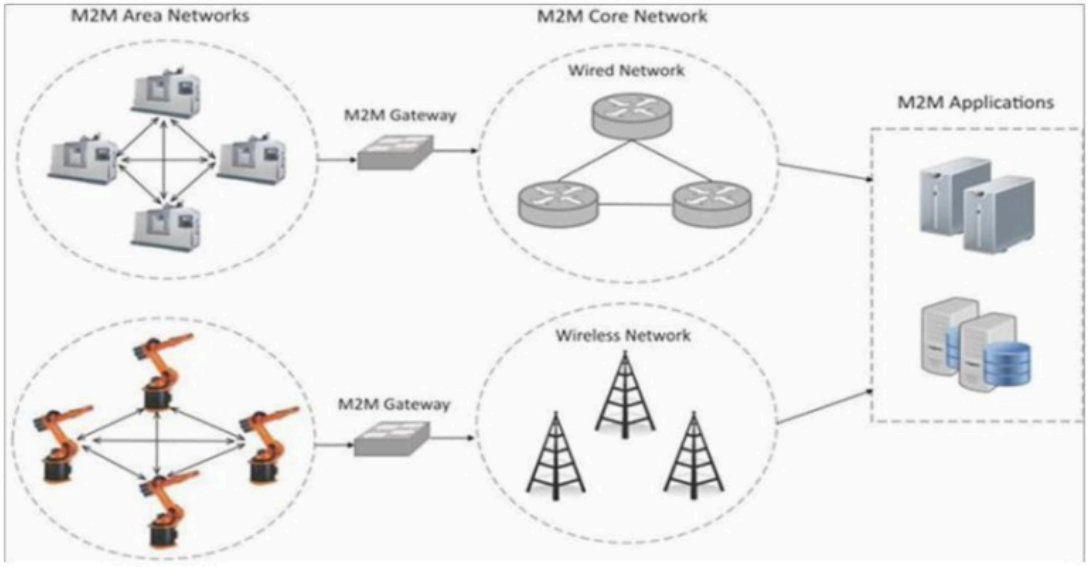
Difference Between Physical and Logical Design of IoT

Physical Design	Logical Design
Physical design is highly detailed.	Logical design is a high-level design and doesn't provide any detail.
Physical design is more graphical than textual; however, it can comprise both.	Logical design can be textual, graphic, or both.
A physical design focuses on specific solutions explaining how they are assembled or configured.	A logical design focuses on satisfying the design factors, including risks, requirements, constraints, and assumptions.

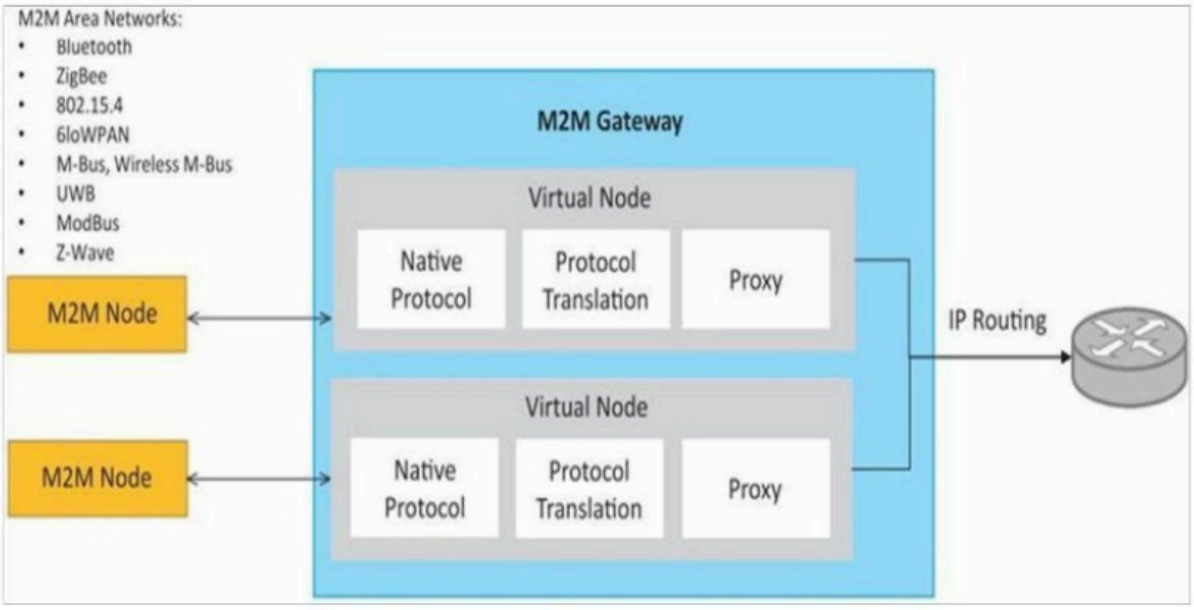
- **16. Major Components of IoT (Simplified View)**



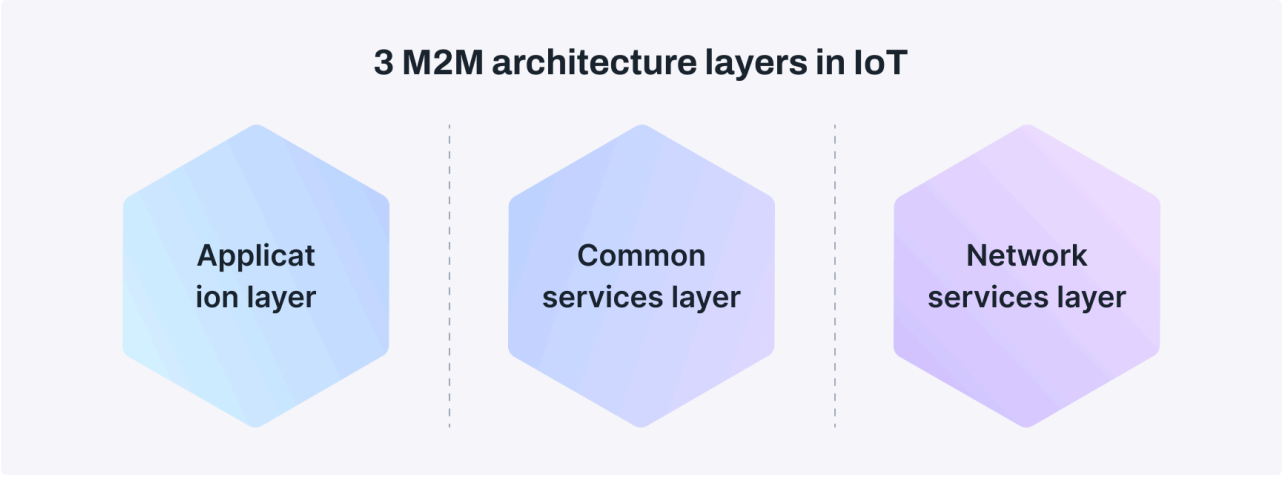
- **19. M2M (Machine-to-Machine) Communication** (M2M System Architecture)



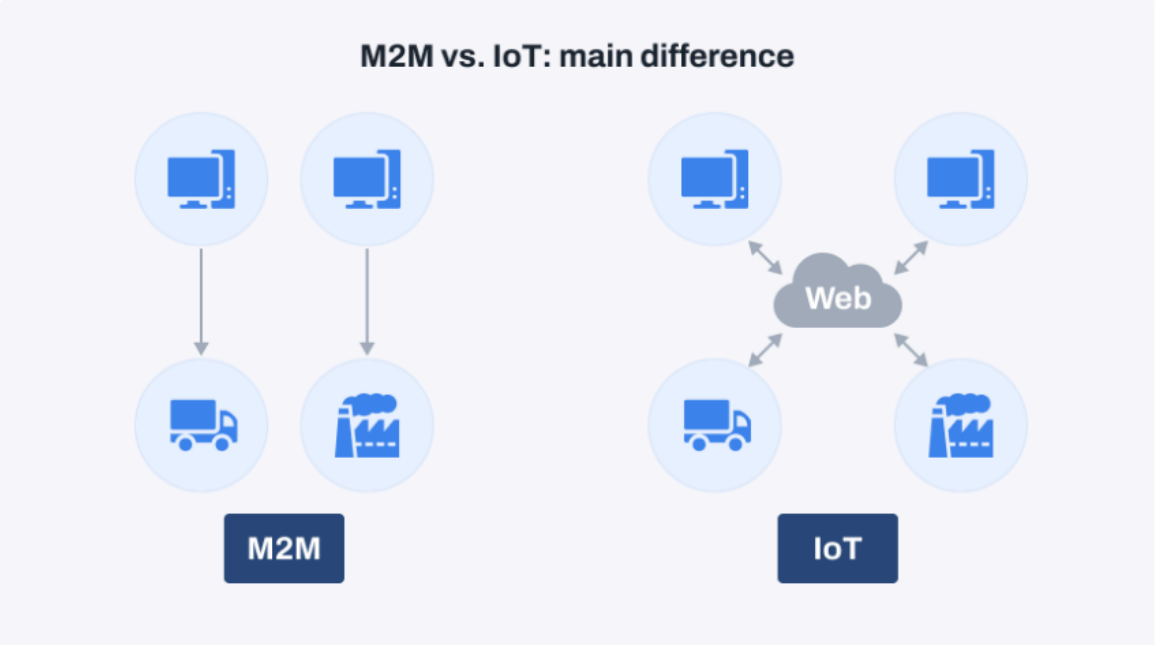
- **19. M2M (Machine-to-Machine) Communication** (M2M Gateway block diagram)



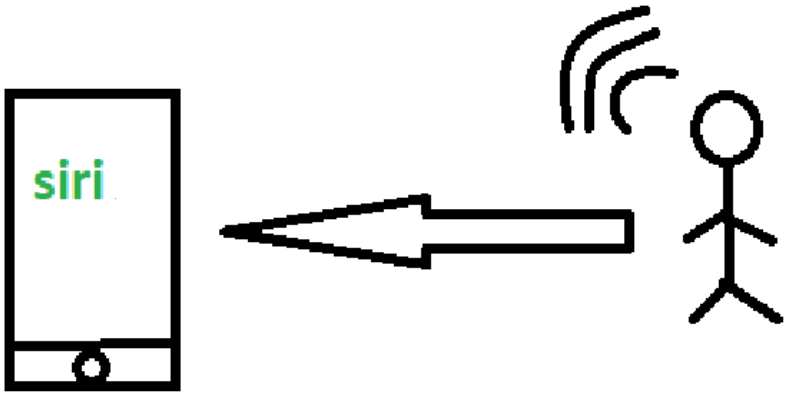
- **19. M2M (Machine-to-Machine) Communication** (M2M Architecture in IoT - 3 layers)



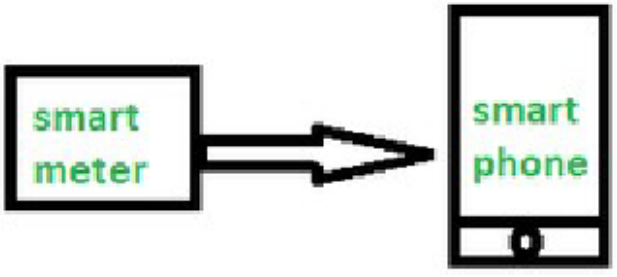
- **19. M2M (Machine-to-Machine) Communication** (Differences between M2M and IoT - visual)



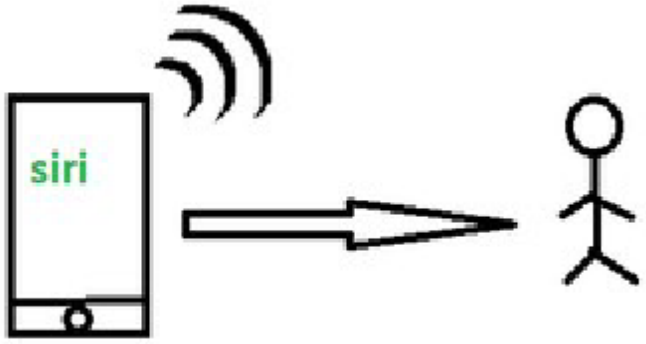
- **21. Types of Communications in IoT** (H2M - Siri example)



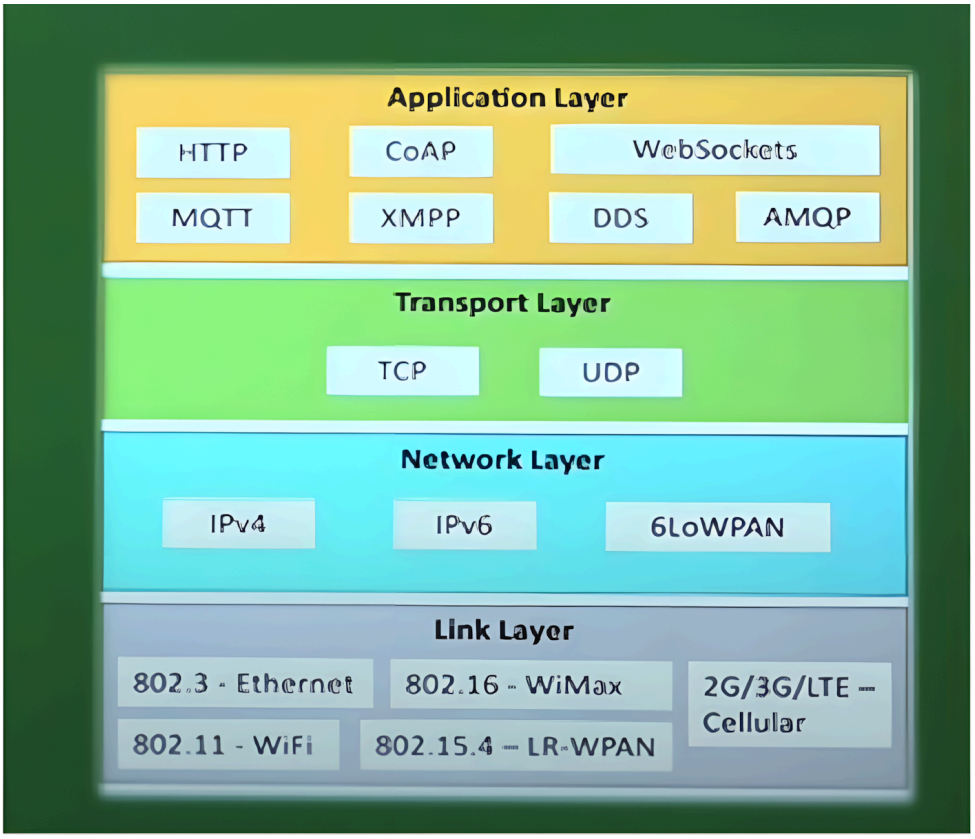
- **21. Types of Communications in IoT** (M2M - Smart meter to phone)



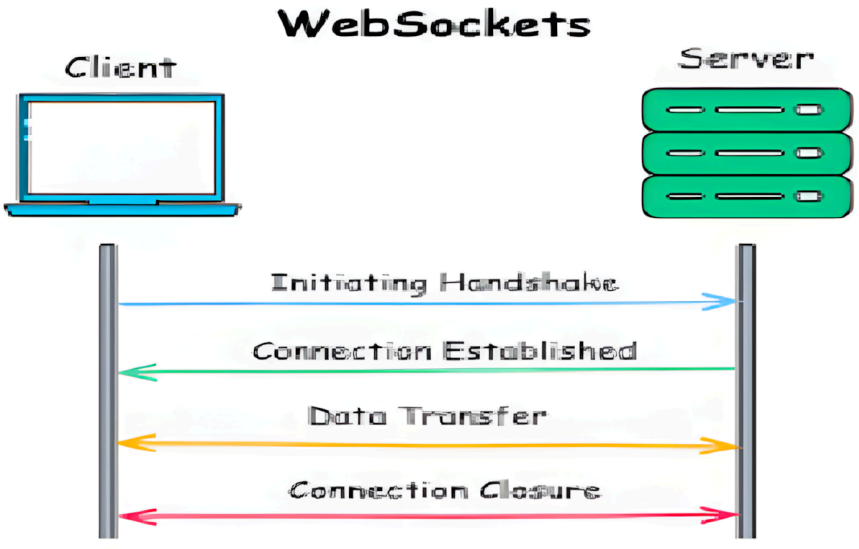
- **21. Types of Communications in IoT** (M2H - Siri example, machine initiating)



- **22. IoT Protocols** (Protocol Stack Overview)



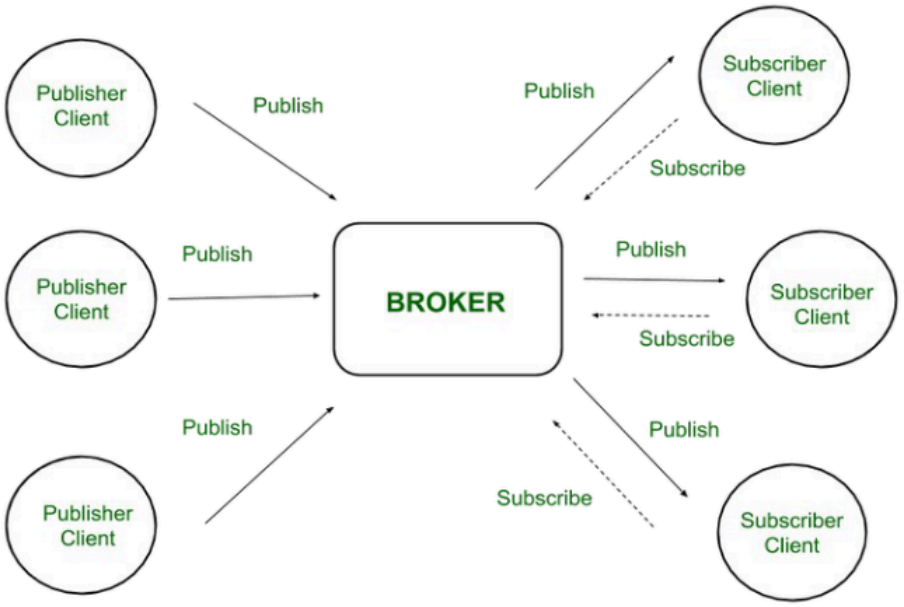
- **22. IoT Protocols** (WebSocket working)



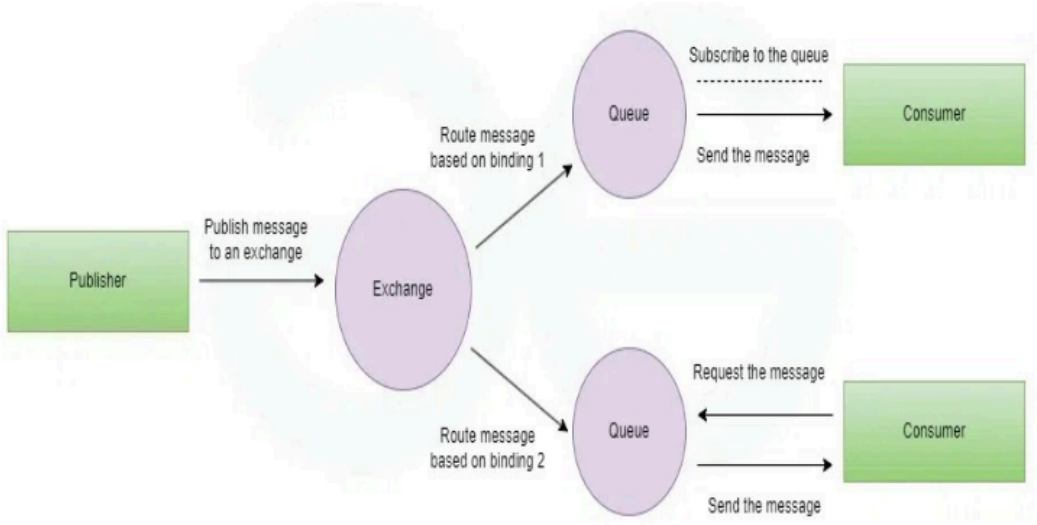
- **22. IoT Protocols** (WebSocket - Client-Server Handshake)

- WebSocket uses a TCP connection to create a persistent connection between a client and a server
- The client and server use an HTTP/HTTPS handshake to establish a connection
- The client sends an Upgrade: websocket header, and the server responds with 101 Switching Protocols
- The WebSocket protocol supports text and binary data formats

- **22. IoT Protocols** (MQTT Publish/Subscribe Architecture)



- **22. IoT Protocols** (AMQP Components diagram)

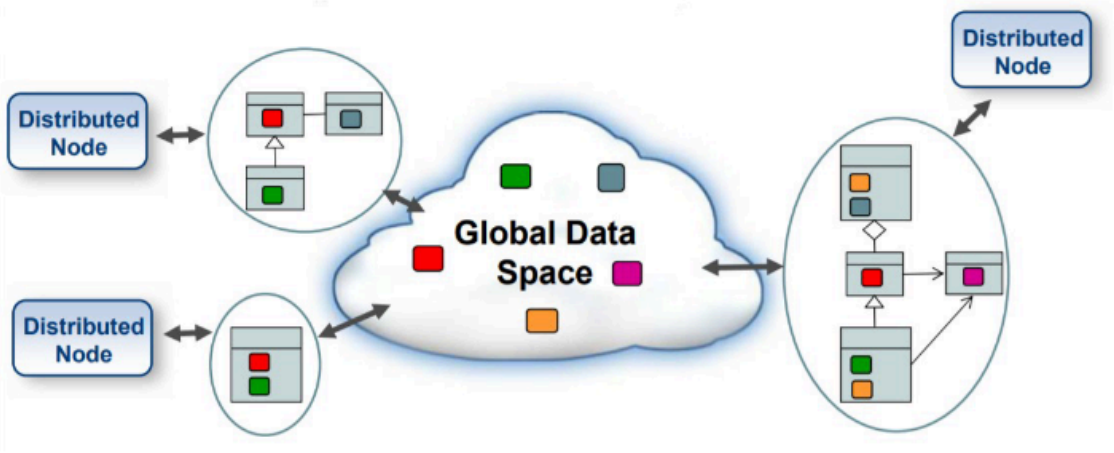


- **22. IoT Protocols** (XMPP - Not an explicit diagram, but context given)

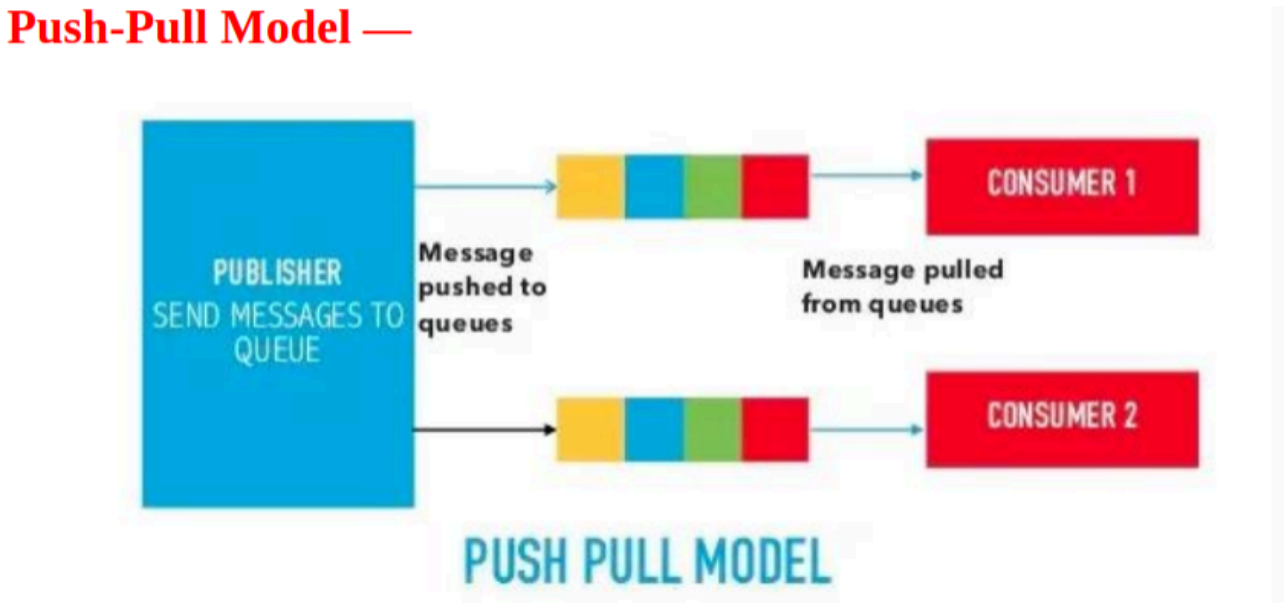
- **X :** It means eXtensible. XMPP is an open-source project which can be changed or extended according to the need.
- **M :** XMPP is designed for sending messages in real time. It has very efficient push mechanism compared to other protocols.
- **P :** It determines whether you are online/offline/busy. It indicates the state.
- **P :** XMPP is a protocol, that is, a set of standards that allow systems to communicate with each other.

(XMPP logo/concept)

- **22. IoT Protocols** (DDS Global Data Space diagram)

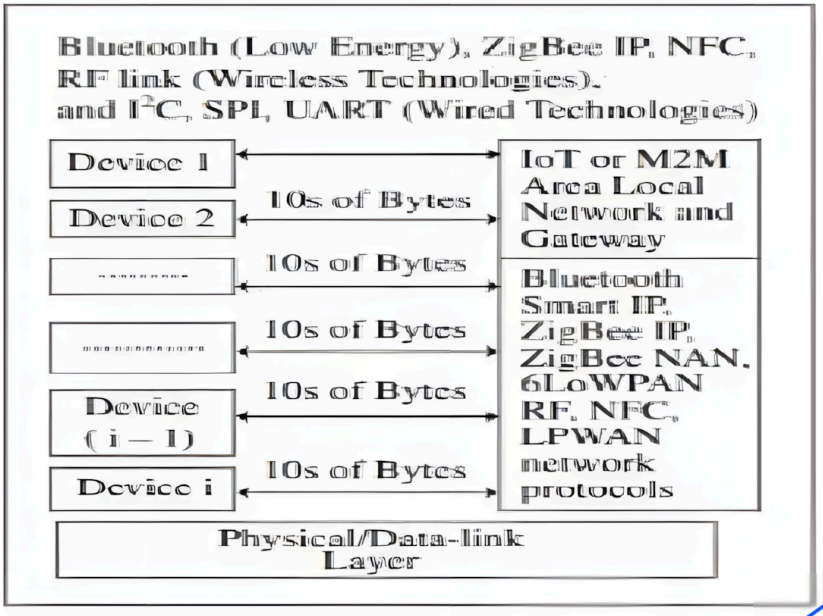


- **22. IoT Protocols** (Push-Pull Model)

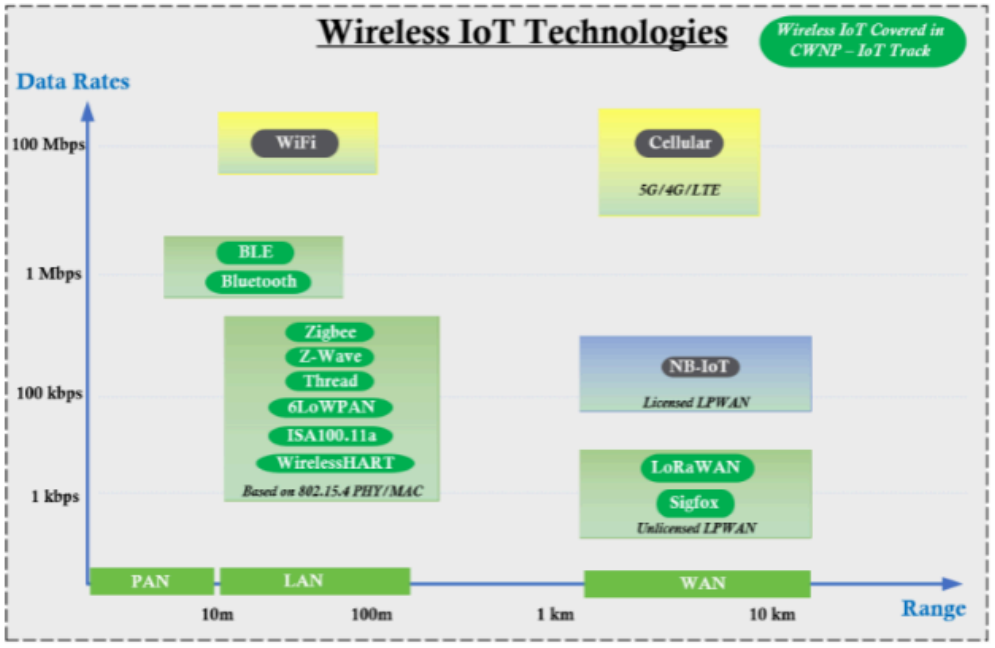


- **22. IoT Protocols** (Exclusive Pair Model)

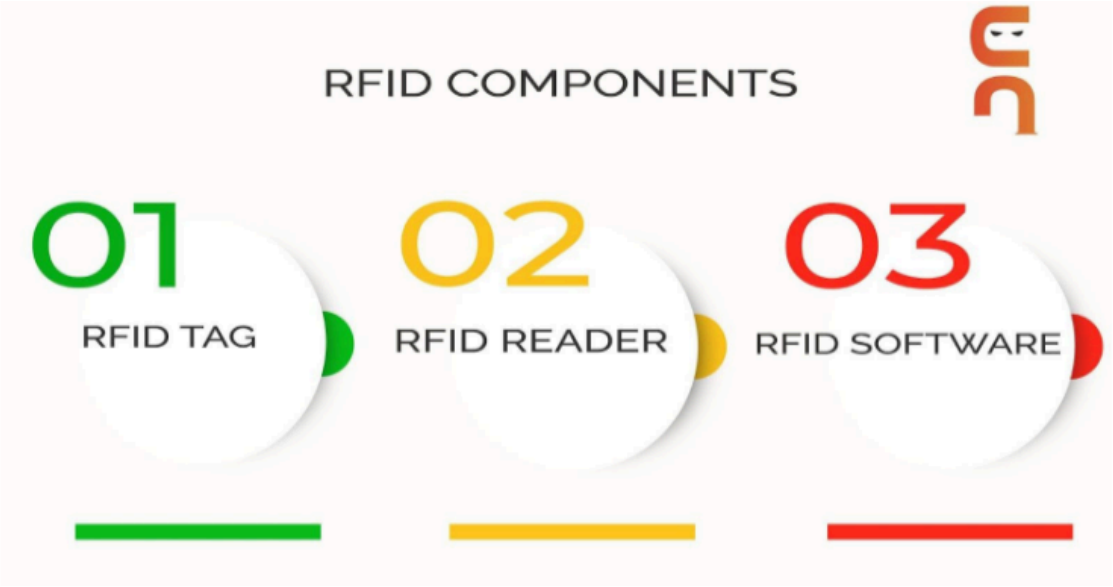
- **23. Communication Technologies** (Device to Gateway - General)



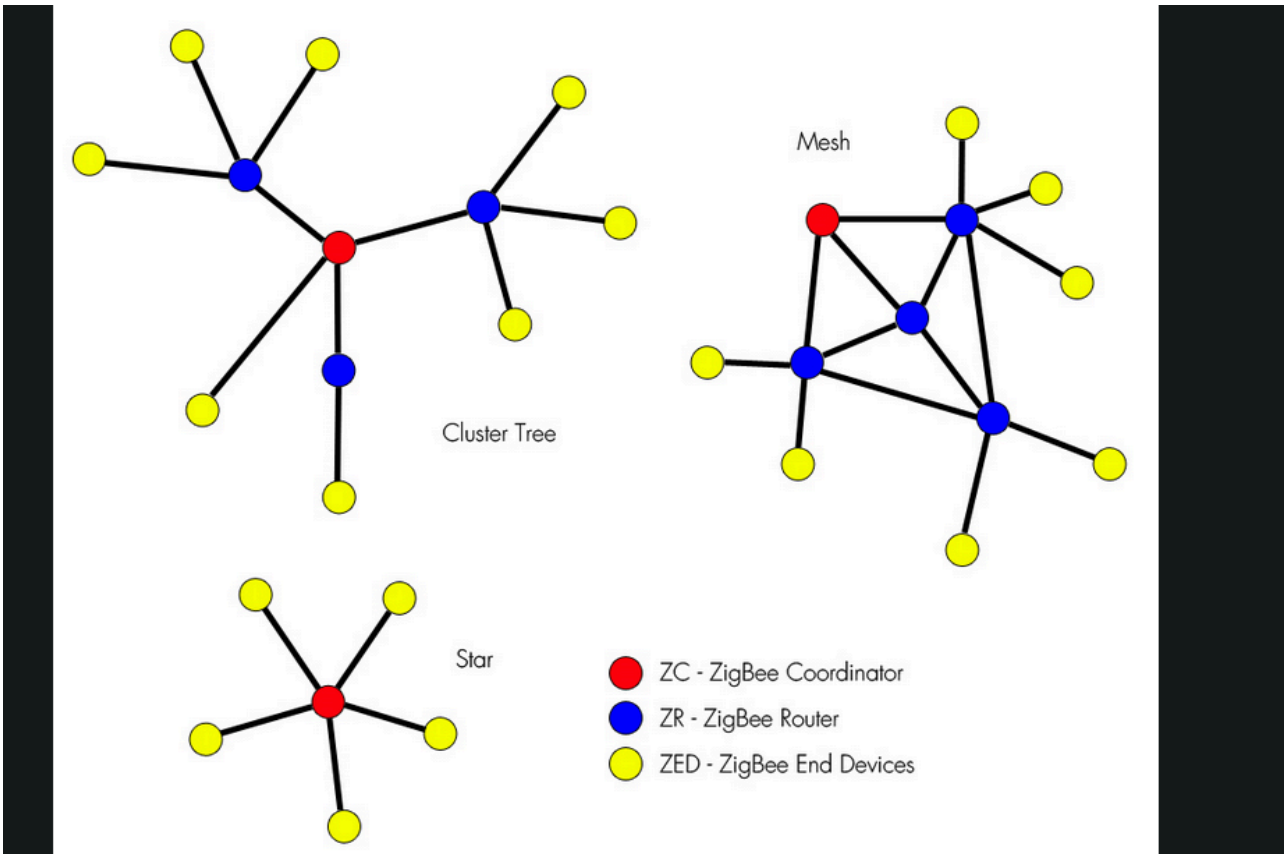
- **23. Communication Technologies** (Wireless IoT Technologies - Data Rates vs Range graph)



- **23. Communication Technologies** (RFID Components)



- **23. Communication Technologies** (Zigbee network topologies)

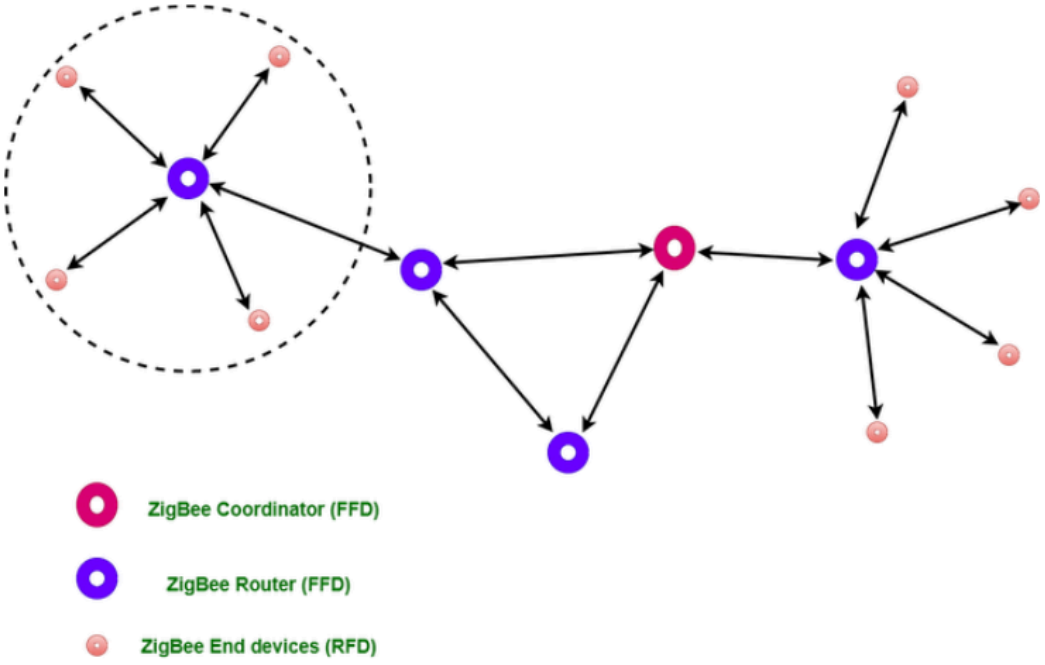
- 

- **23. Communication Technologies** (WiFi vs Bluetooth image)

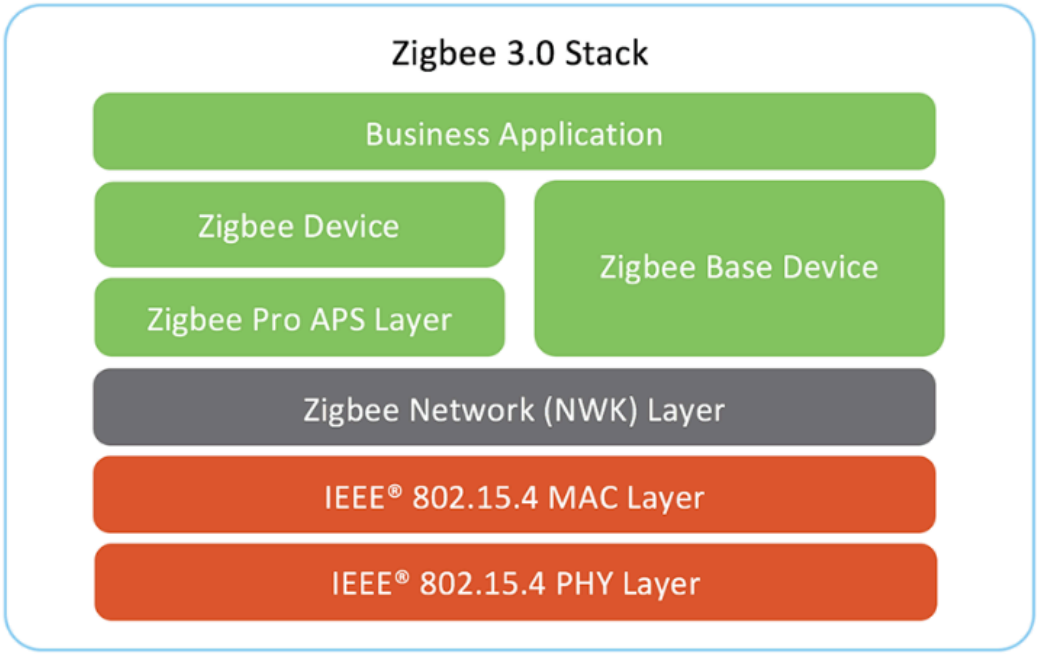
  Types of ZigBee Devices:

  - **Zigbee Coordinator Device:** It communicates with routers. This device is used for connecting the devices.
  - **Zigbee Router:** It is used for passing the data between devices.
  - **Zigbee End Device:** It is the device that is going to be controlled.

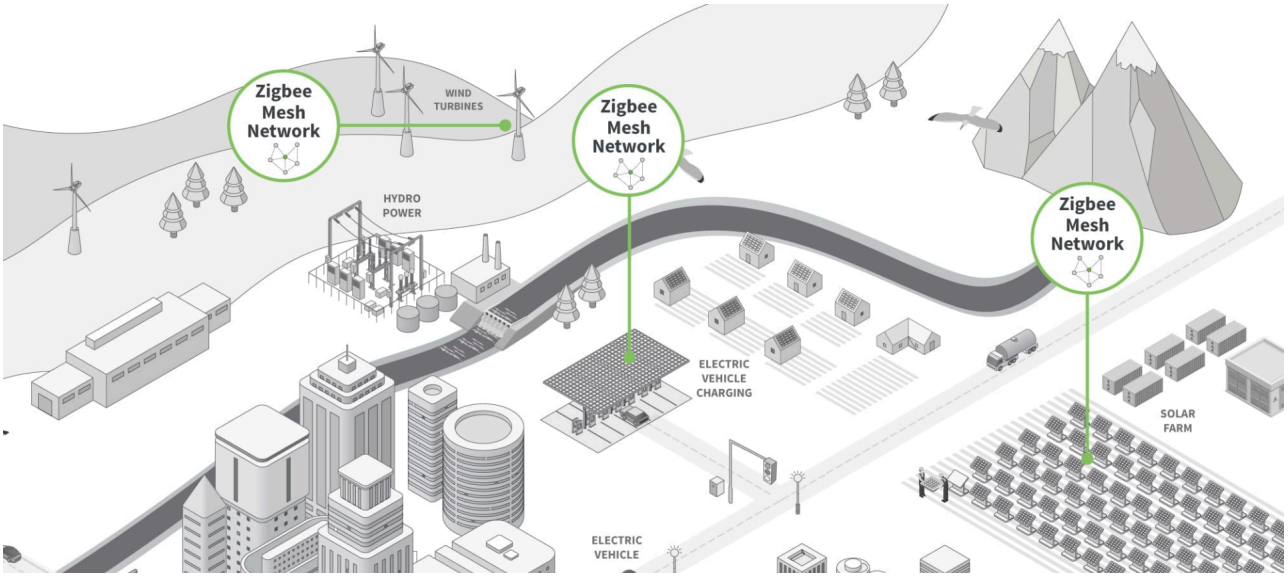
- **23. Communication Technologies** (ZigBee devices in network)

- **23. Communication Technologies** (Zigbee 3.0 Stack)

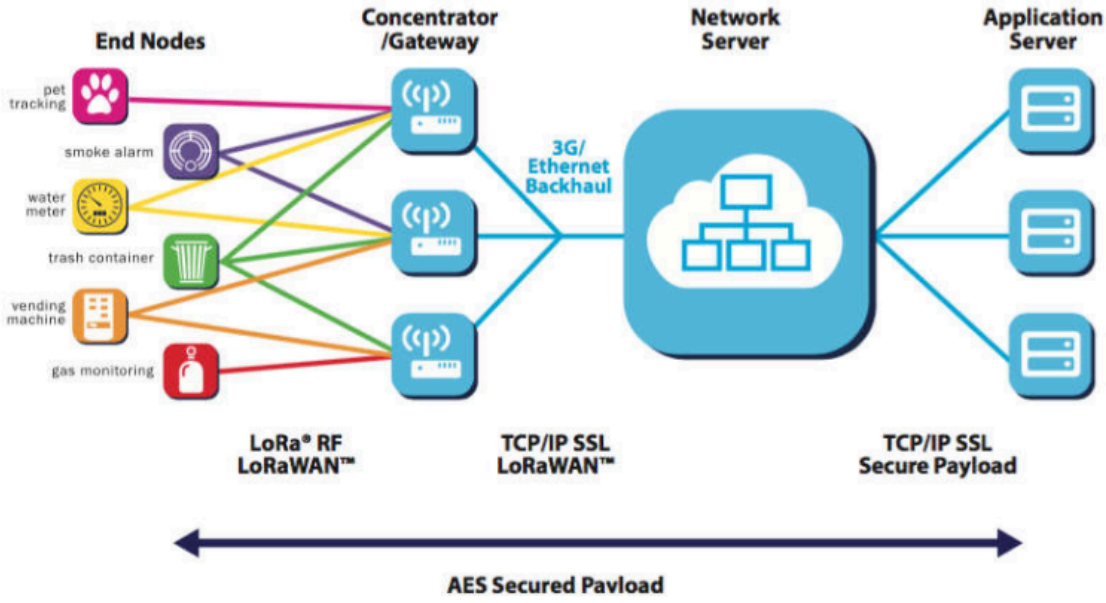


- **23. Communication Technologies** (Zigbee Mesh Network applications visual)

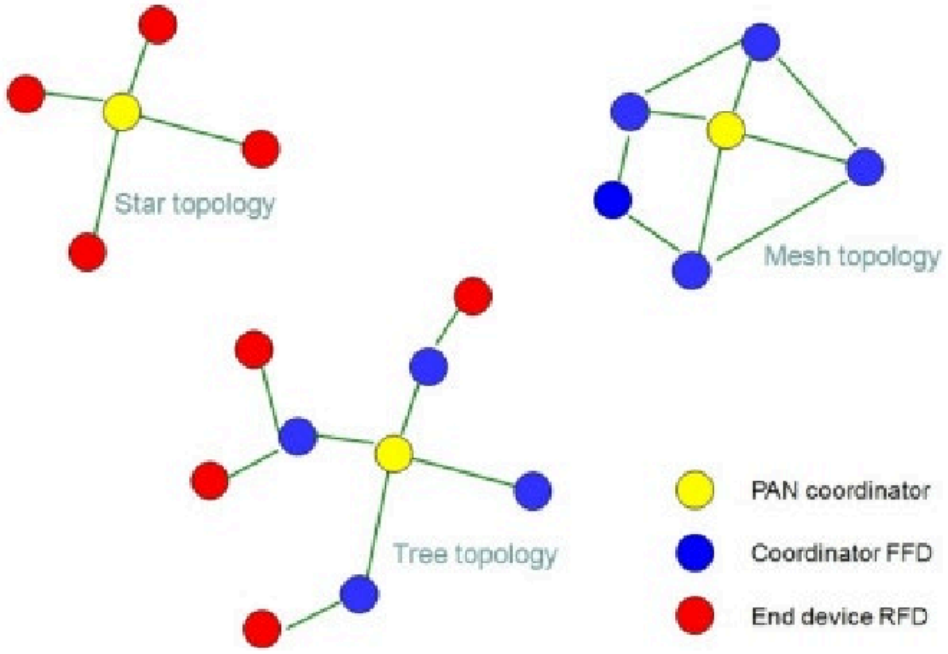


- **23. Communication Technologies** (LoRaWAN Pros)

- **23. Communication Technologies** (LoRaWAN Architecture)



- **23. Communication Technologies** (IEEE 802.15.4 topologies)



- **23. Communication Technologies** (IEEE 802.15.4 Pros)

Pros:

- Very long range (up to 15-20 kilometers in rural areas).
- Extremely low power consumption, enabling years of operation on a single battery.
- Suitable for environments with limited infrastructure (e.g., rural areas).
- Good scalability, capable of supporting thousands of devices.

- **23. Communication Technologies** (IEEE 802.15.4 Cons)

Cons:

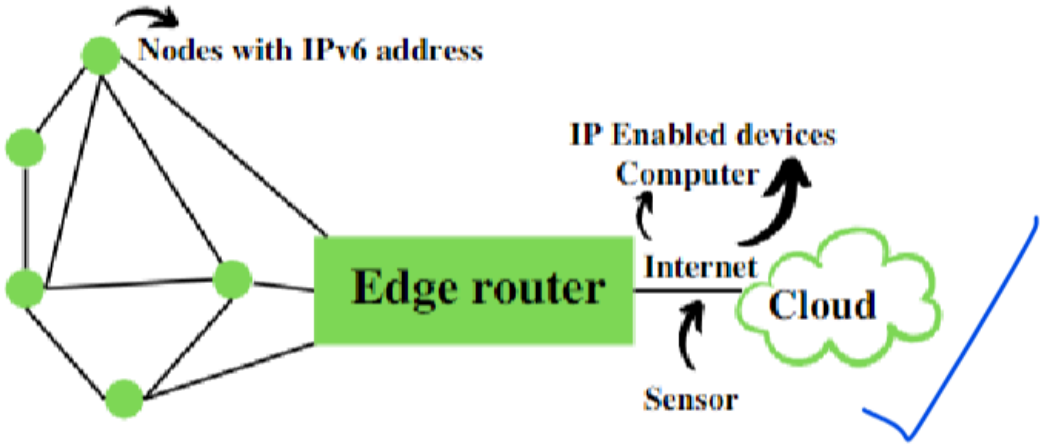
- Low data rate (up to 50 kbps), making it unsuitable for high-bandwidth applications.
- Typically, it requires a gateway for internet access.
- Limited interoperability with other networks.

- **23. Communication Technologies** (IEEE 802.15.4 Applications)

Applications:

- Agricultural IoT (e.g., soil moisture sensors, livestock monitoring).
- Smart cities (e.g., parking sensors, waste management).
- Environmental monitoring (e.g., air quality sensors, flood detection)

- **23. Communication Technologies** (6LoWPAN - Nodes, Edge router, Cloud)

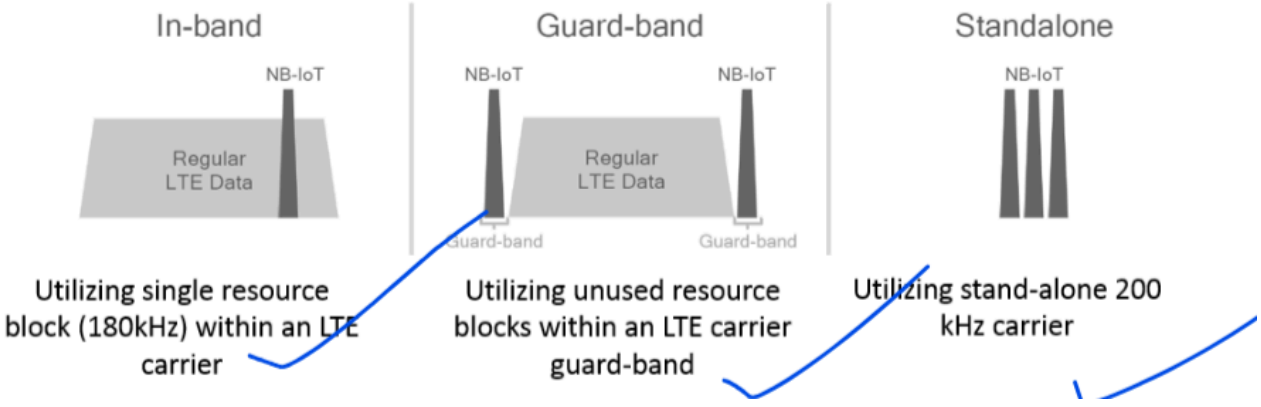


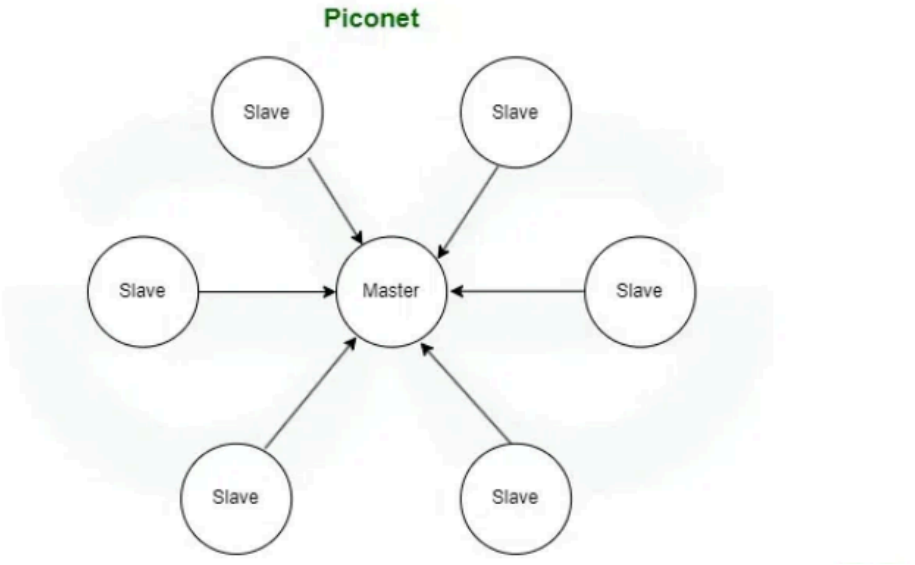
- **23. Communication Technologies** (6LoWPAN Advantages)

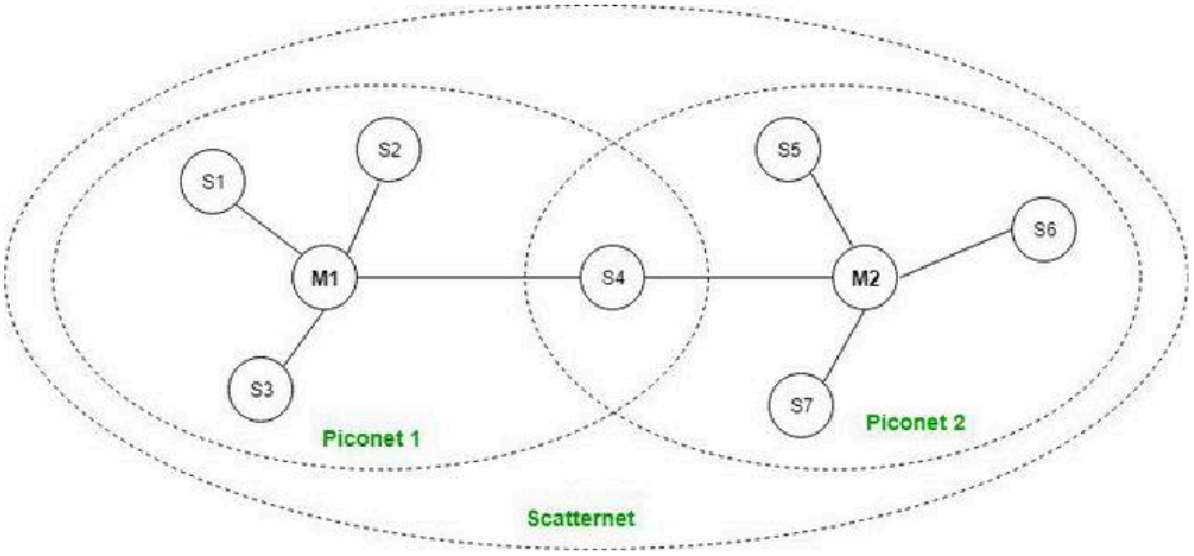
Basic Requirements of 6LoWPAN

- The device should be having sleep mode in order to support the battery saving.
- Minimal memory requirement.
- Routing overhead should be lowered.

- **23. Communication Technologies** (NB-IoT Deployment visuals)



- **23. Communication Technologies** (Bluetooth Piconet)
  - 

- **23. Communication Technologies** (Bluetooth Scatternet)
  - 

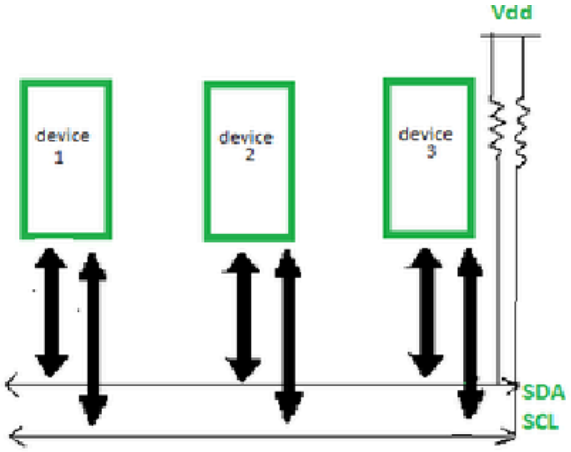
- **23. Communication Technologies** (Bluetooth Features)
  - **Applications of Bluetooth**
    - It can be used in wireless headsets, wireless PANs, and LANs.
    - It can connect a digital camera wireless to a mobile phone.
    - It can transfer data in terms of videos, songs, photographs, or files from one cell phone to another cell phone or computer.
    - It is used in the sectors of Medical healthcare, sports and fitness, Military.
- **23. Communication Technologies** (Bluetooth Advantages/Disadvantages)

Types of Bluetooth

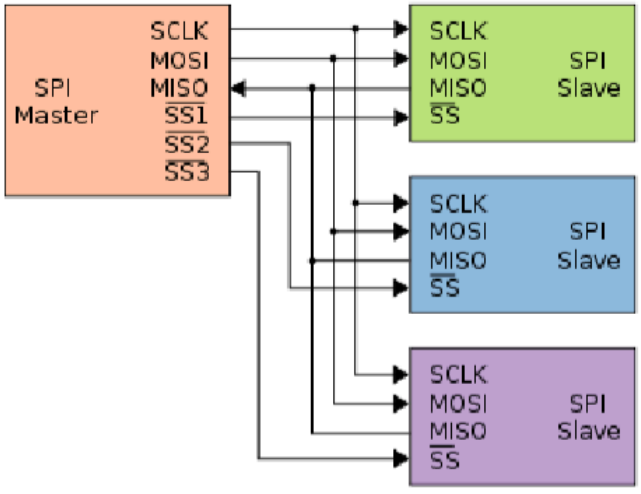
Various types of Bluetooth are available in the market nowadays. Let us look at them.

- **In-Car Headset:** One can make calls from the car speaker system without the use of mobile phones.
- **Stereo Headset:** To listen to music in car or in music players at home.
- **Webcam:** One can link the camera with the help of Bluetooth with their laptop or phone.

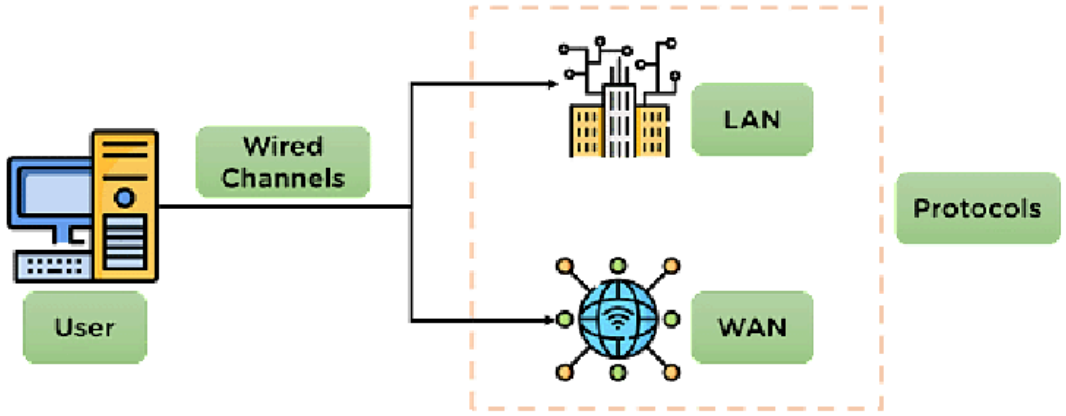
- **23. Communication Technologies** (I2C data transmission)



- **23. Communication Technologies** (SPI communication)



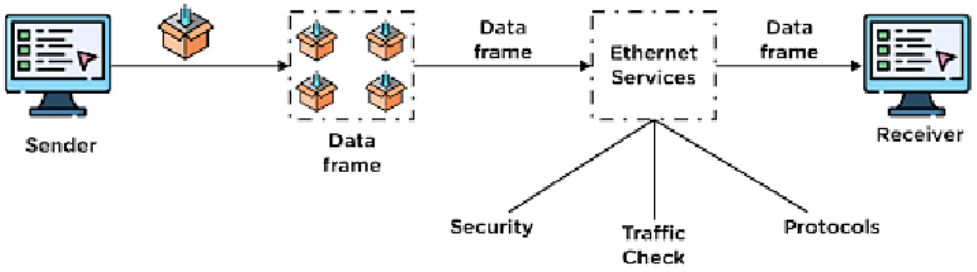
- **23. Communication Technologies** (Ethernet LAN diagram)



- **23. Communication Technologies** (Ethernet Frame/Packet flow)

  Ethernet divides the transmission of data into two parts: packets and frames.

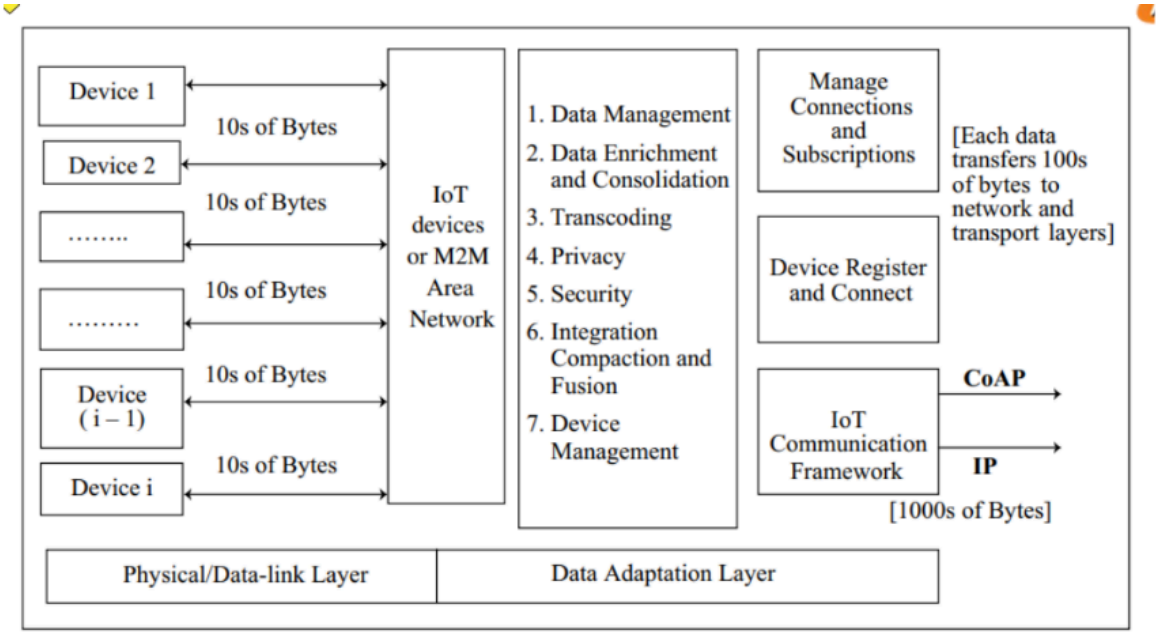
  - Packet–Refers to a unit of data in the network.
  - Frame–Refers to the collection of data packets being transmitted.



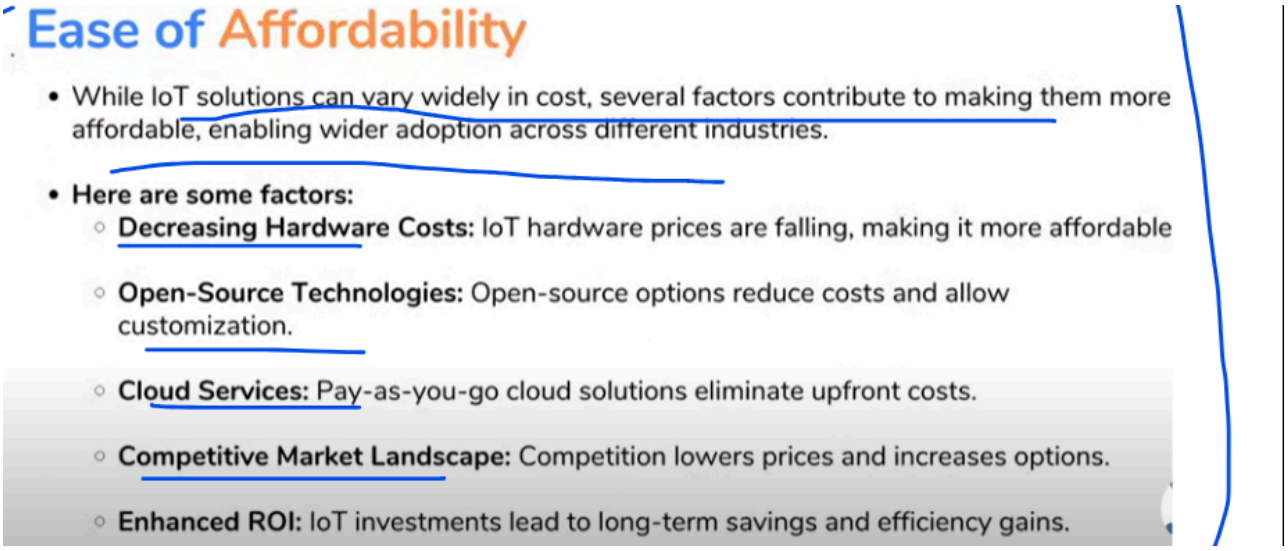
- **23. Communication Technologies** (UART - Device A Tx/Rx)

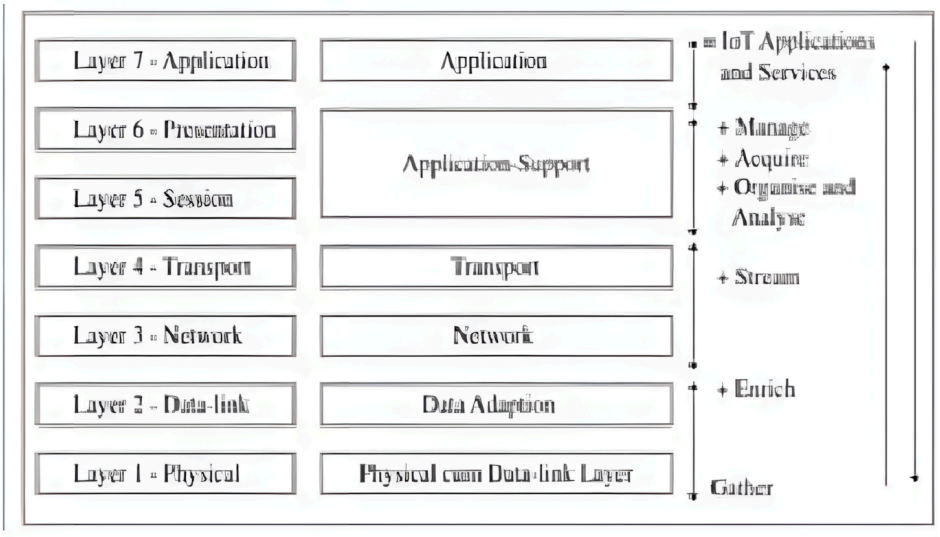
- **23. Communication Technologies** (USB cable image)

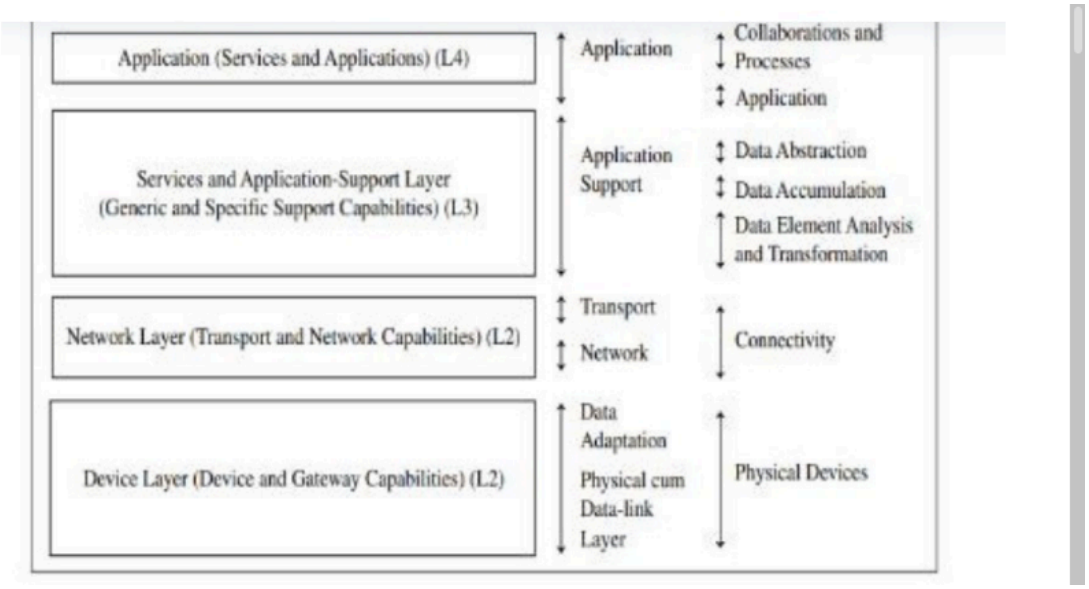
- **24. Data Enrichment and Consolidation Gateway** (Block diagram)



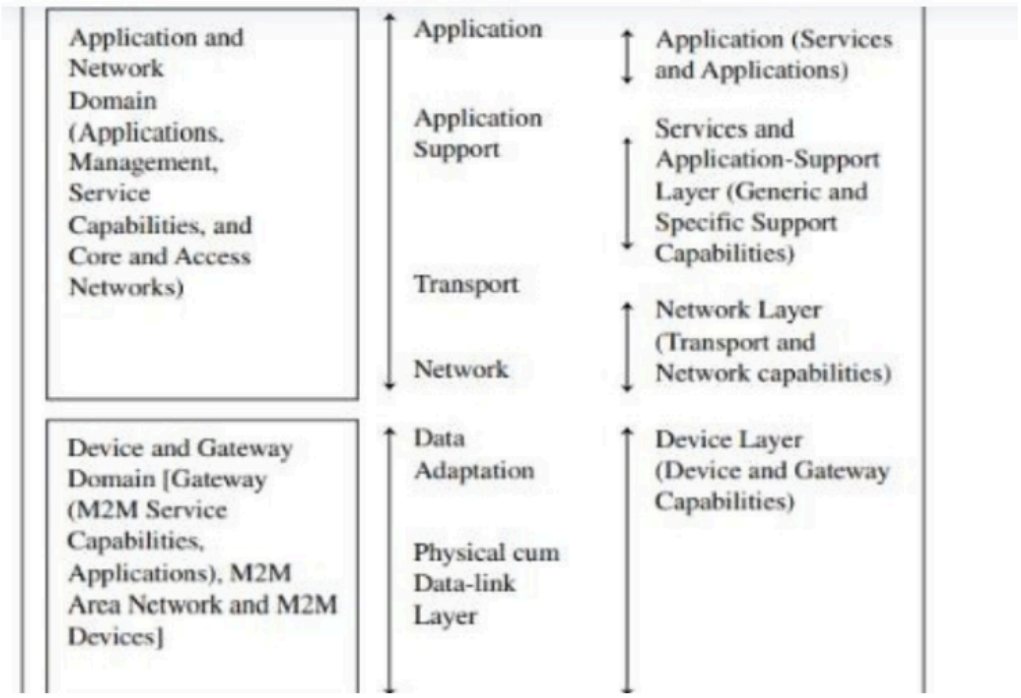
- **24. Data Enrichment and Consolidation Gateway** (Functions visual)
  - Data consolidation is the process of collecting and organizing data from various IoT devices into a single, unified system.
- **24. Data Enrichment and Consolidation Gateway** (Data Consolidation steps visual)
  - **Steps in Data Consolidation**
    - **Data Collection:** Gather data from different IoT devices and sources.
    - **Data Storage:** Use cloud storage to hold large volumes of IoT data securely.
    - **Data Organization:** Structure data in a way that makes it easy to retrieve and analyze, such as organizing by timestamp, sensor type, or location.
    - **Data Compression:** Use compression techniques to reduce data volume while maintaining its integrity.
- **25. Device Management Gateway** (Ease of Designing visual)
  - **Ease of Designing**
    - The design process for IoT systems can be complex due to the integration of various hardware and software components.
    - However, advancements in technology and frameworks have simplified this process significantly.
    - **Here are some factors:**
      - **Modular Components:** Simplifies integration and customization.
      - **Development Platforms and Tools:** Streamlines development and deployment.
      - **Standard Protocols:** Ensures device interoperability.
      - **Libraries and Frameworks:** Accelerates prototyping.
      - **Community Support and Resources:** Aids troubleshooting and learning.
- **25. Device Management Gateway** (Ease of Affordability visual)

- 

- **26. IoT M2M Systems Layers and Design Standardization** (IETF six-layer model vs Oracle)
  - 

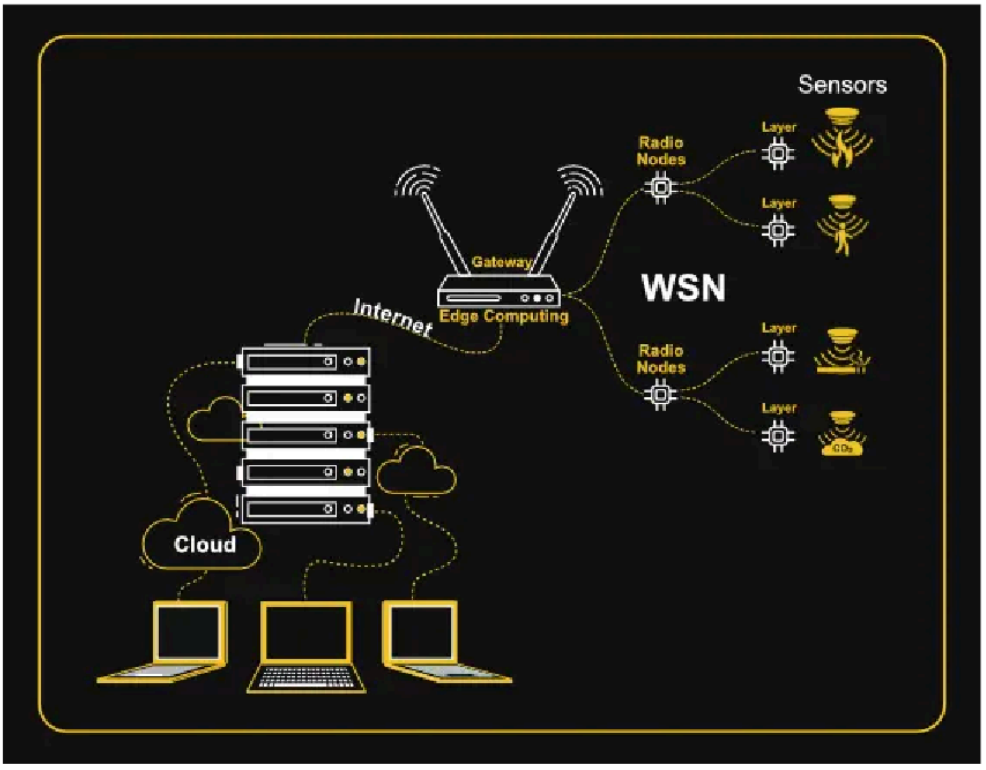
- **26. IoT M2M Systems Layers and Design Standardization** (ITU-T RM1 vs OSI vs CISCO RM2)
  - 

- **26. IoT M2M Systems Layers and Design Standardization** (ETSI M2M domains vs OSI vs ITU-T)



Unit 2: IoT Networking, Platforms, Sensors, and Actuators

- **1. Wireless Sensor Network (WSN)** (WSN Architecture)



- **2. Participatory Sensing (PS)** (Phases of PS process)

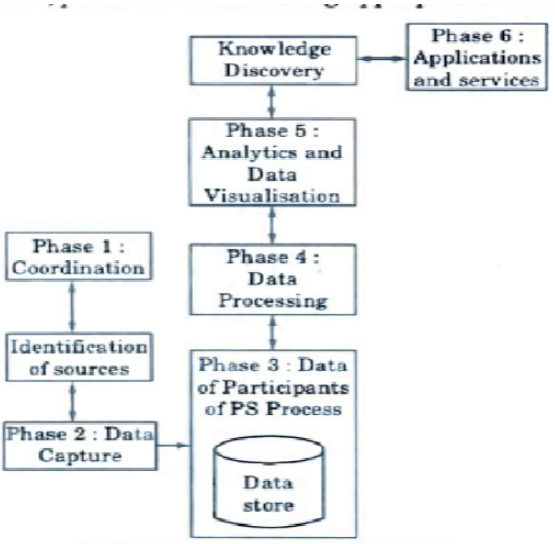
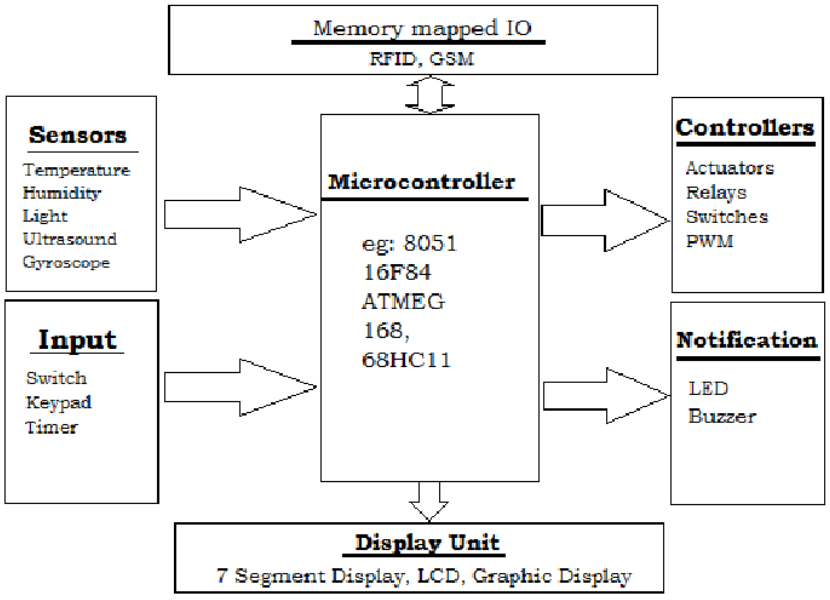


Fig. 2.13.1. Phases of PS process.

- **3. Embedded Devices (System) in IoT** (Embedded Devices System diagram)



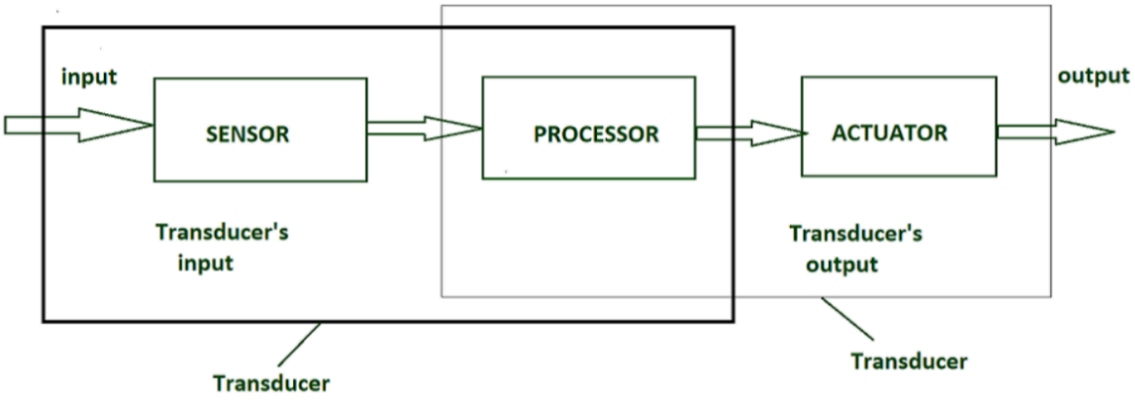
- **4. Sensors** (Introduction to Sensors - highlighted text)

  - A Sensor is a characteristic of any device or material to detect the presence of a particular physical quantity.
  - The output of the sensor is a signal, which is converted to human readable form.

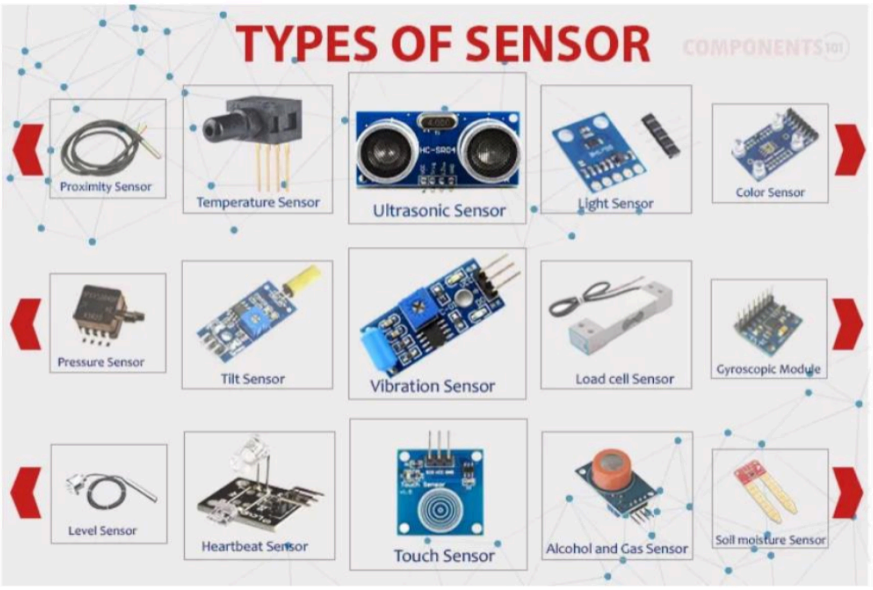
- **4. Sensors** (Transducer definition - highlighted text)

  - **Transducer** : It converts the signal from one physical form to another physical form. it is also called energy converter. For example, microphone converts sound to electrical signal . It is based on the principle of conservation of energy.

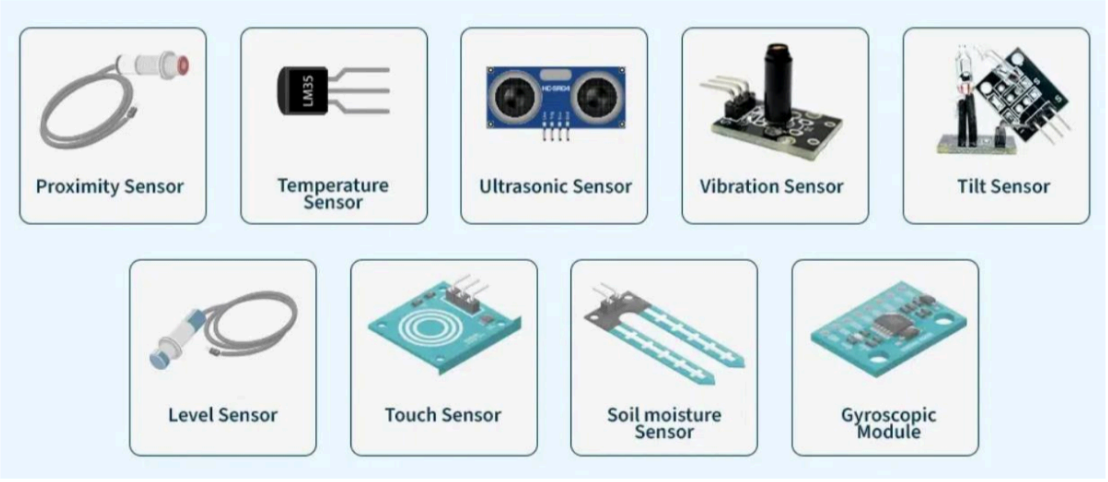
- **4. Sensors** (Sensor-Processor-Actuator flow)



- **4. Sensors** (Types of Sensor Components 101 - image grid)



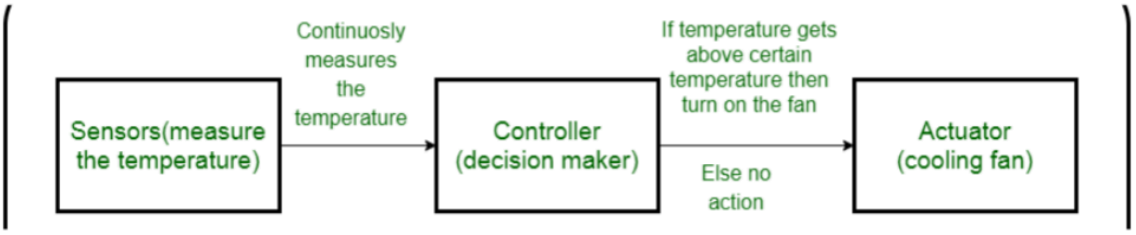
- **4. Sensors** (Types of Sensors - simpler image grid)



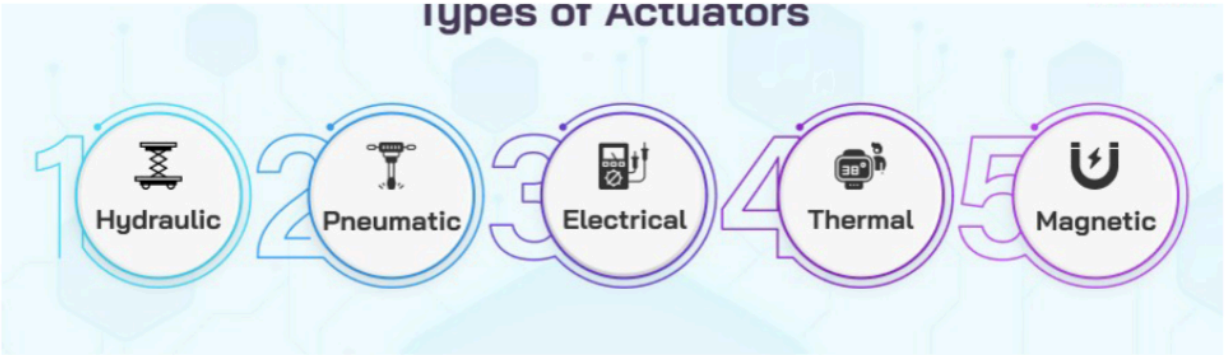
- **4. Sensors** (Table comparing Digital vs Analog Sensors)

Feature	Digital Sensors	Analog Sensors
Signal Type	Digital (discrete)	Analog (continuous)
Signal Output	Discrete digital signal (usually binary)	Continuous analog voltage or current
Accuracy	Higher accuracy, less susceptible to noise	Generally less accurate due to noise and drift
Complexity	More complex due to signal processing	Simpler design and circuitry
Interference	Less prone to electromagnetic interference (EMI)	More prone to EMI

- **5. Actuators** (IoT device components including Actuators)



- **5. Actuators** (Types of Actuators - circular icons)



- **8. Overview of IoT Hardware Platforms** (Arduino Uno Board Image)

- **8. Overview of IoT Hardware Platforms** (Arduino Uno Architecture)

Architecture of Arduino board :

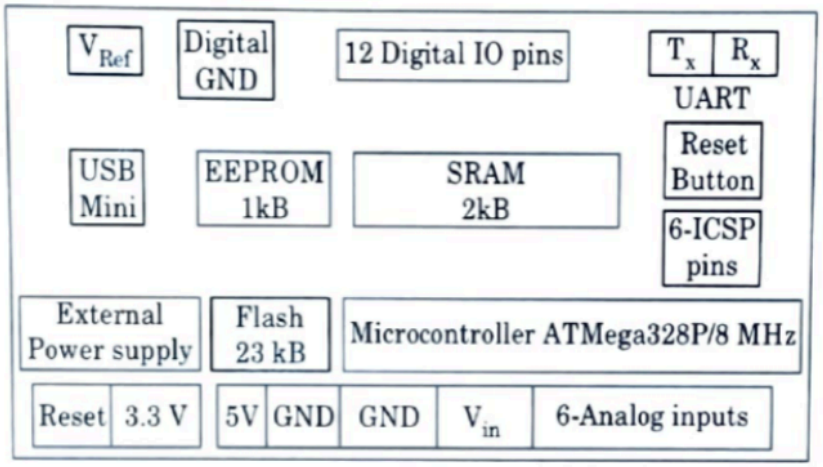


Fig. 2.17.1.

- **8. Overview of IoT Hardware Platforms** (Raspberry Pi Board Image)

- **8. Overview of IoT Hardware Platforms** (Raspberry Pi Architecture)

Architecture of Raspberry Pi board :

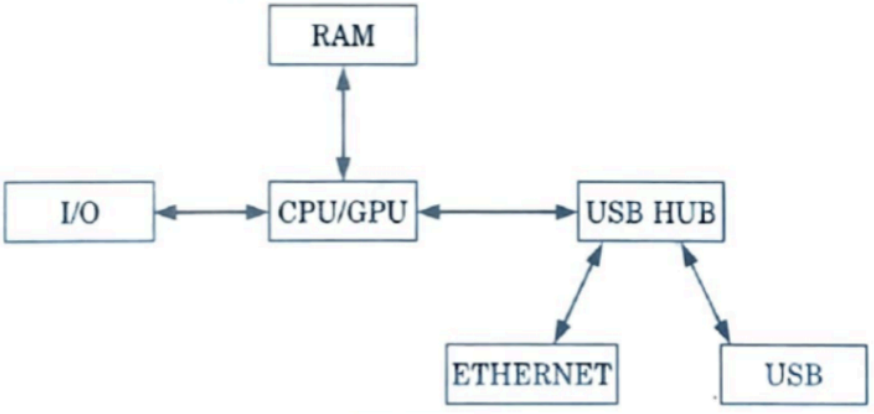
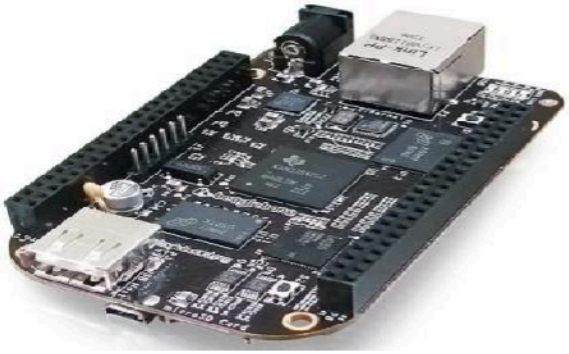
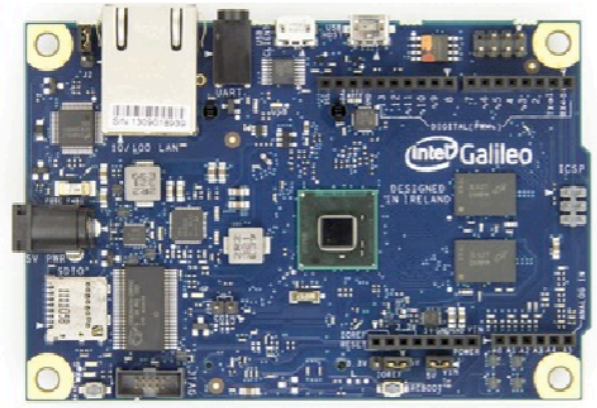


Fig. 2.20.1.

- **8. Overview of IoT Hardware Platforms** (BeagleBone Board Image)

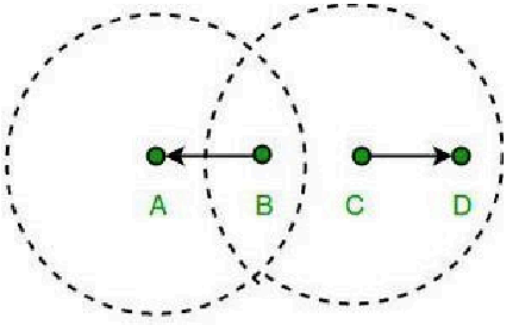


- **8. Overview of IoT Hardware Platforms** (Intel Galileo Board Image)

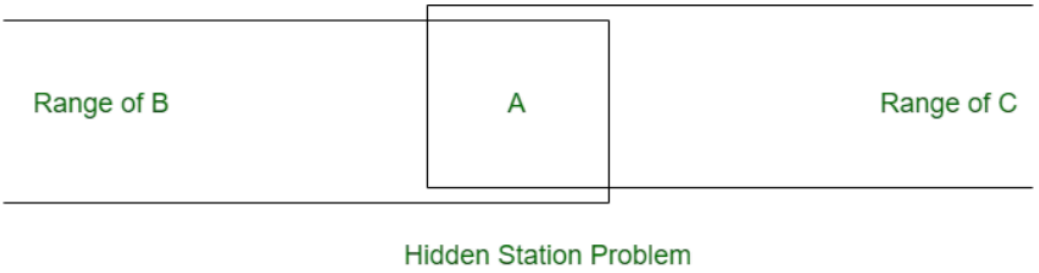


Unit 3: Network & Communication Aspects in IoT

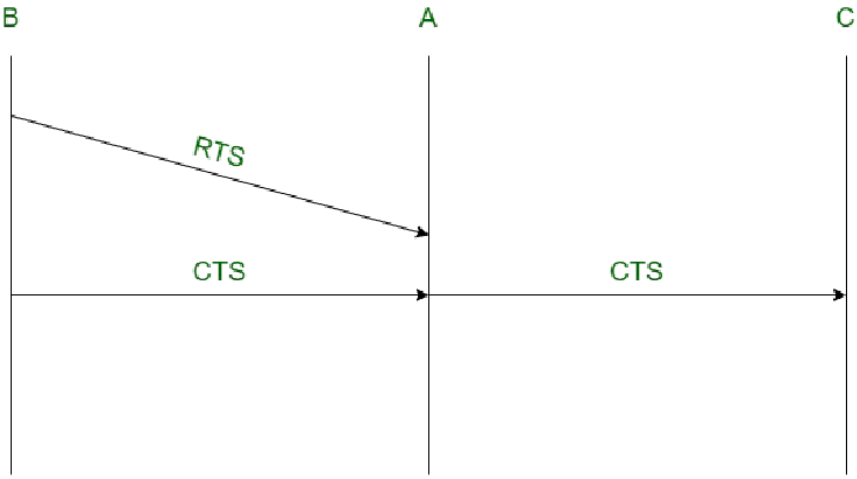
- **2.c. Exposed Terminal Problem** (Diagram)



- **2.d. Hidden Station Problem (HSP)** (Diagram)

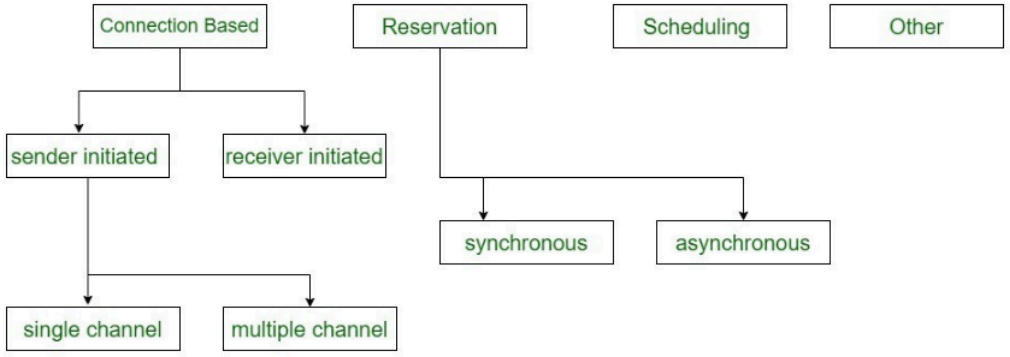


- **2.d. Hidden Station Problem (HSP)** (Handshaking to prevent HSP)

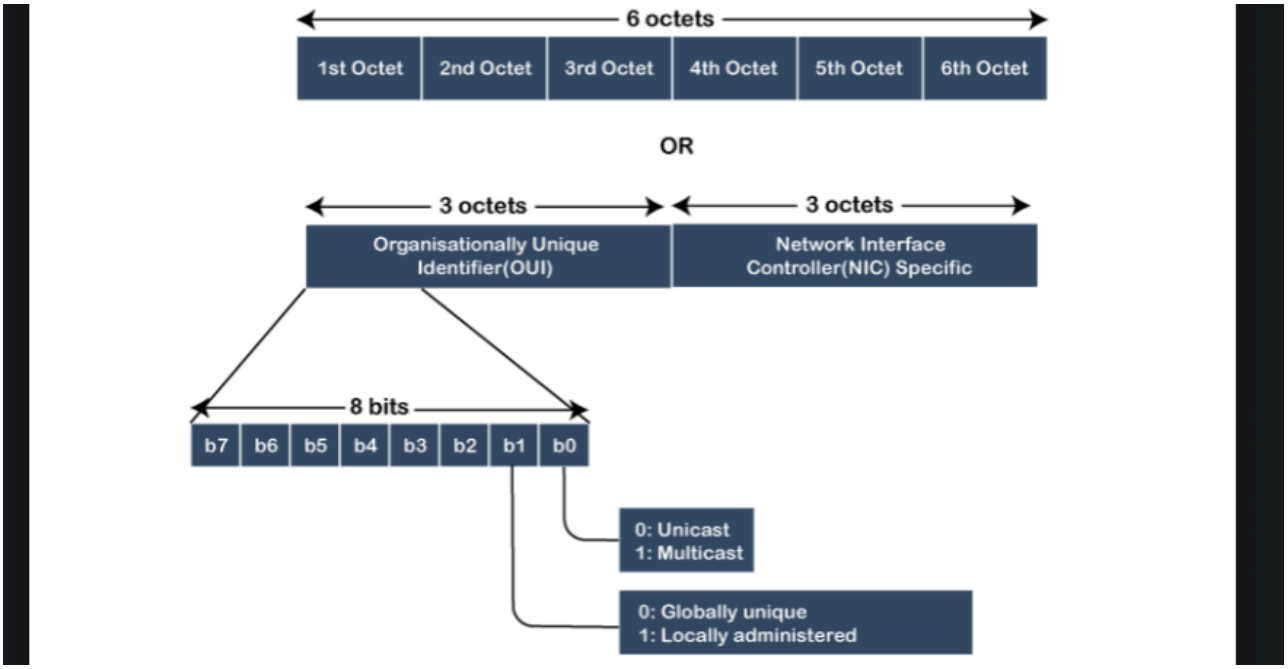


Use of handshaking to prevent hidden station problem

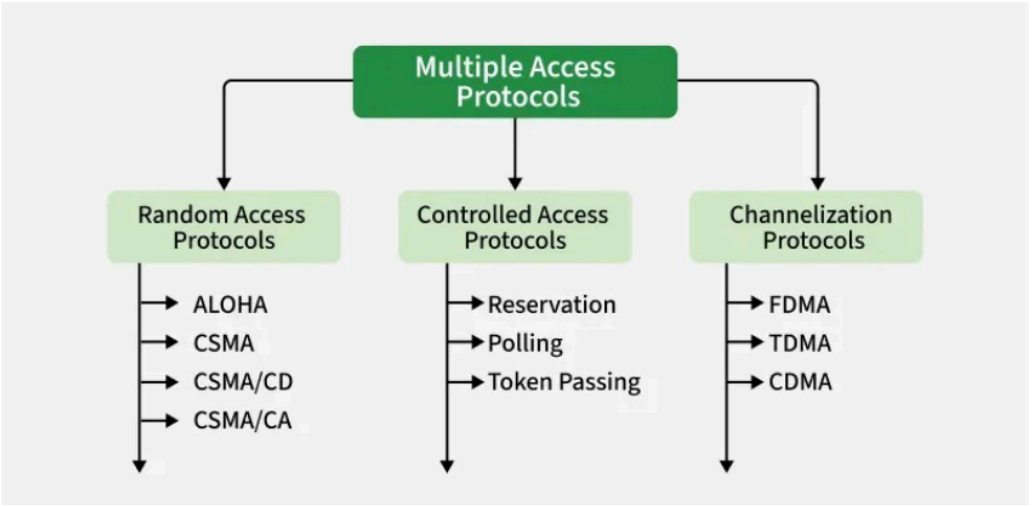
- **3. Classification of MAC Protocols** (High-level classification)



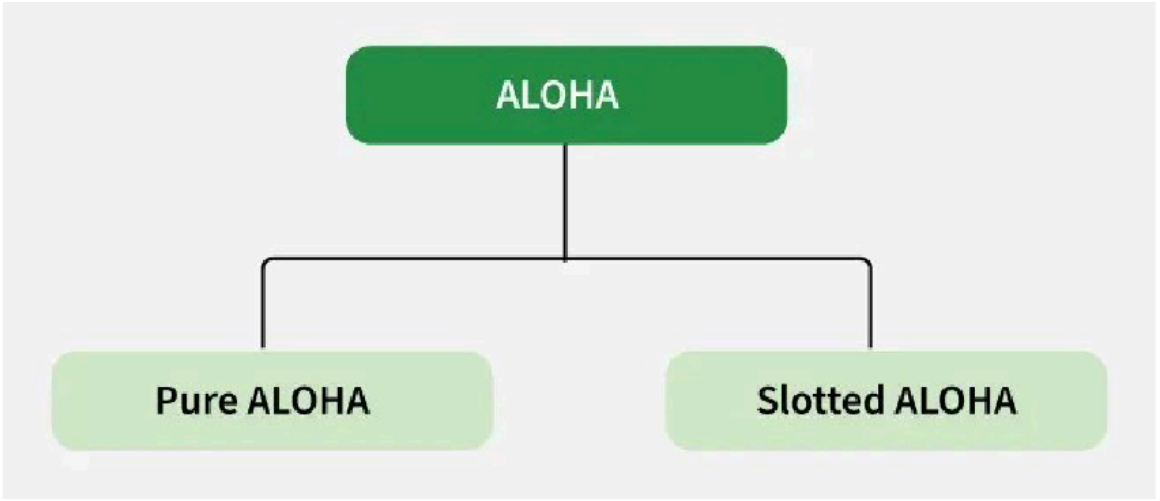
- **4.3. Format of a MAC Address** (MAC Address Structure)



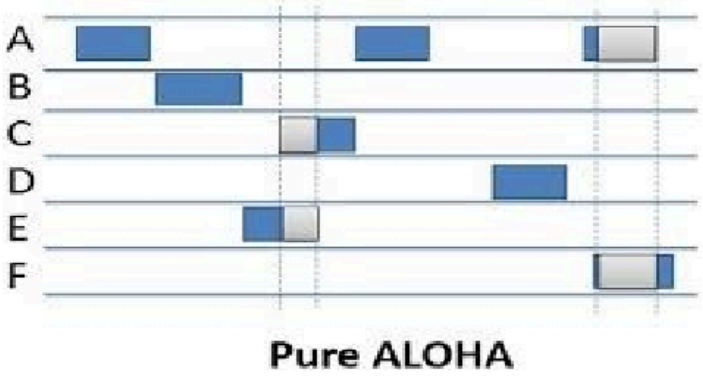
- **6. Subdivisions of Multiple Access Protocols** (Random, Controlled, Channelization)



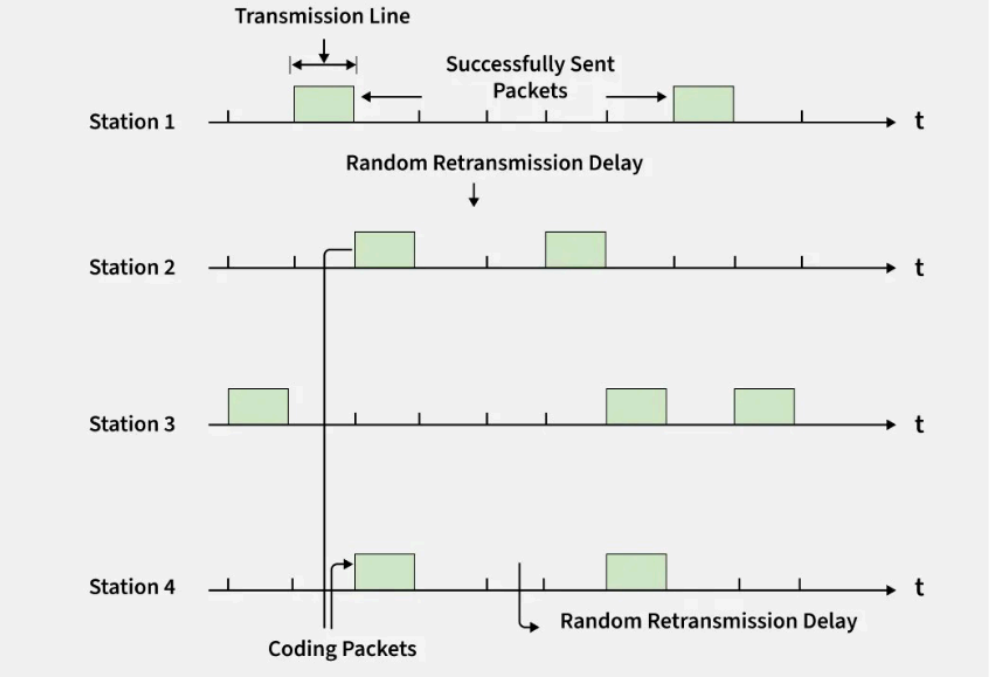
- **6.1.A. ALOHA** (Pure ALOHA, Slotted ALOHA types)



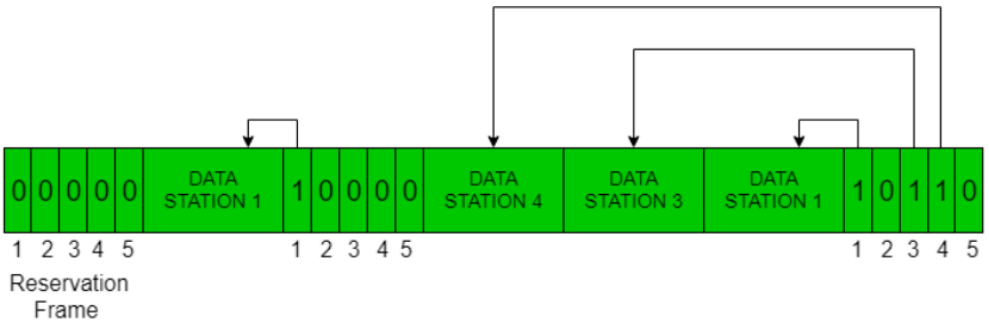
- **6.1.A. Pure ALOHA** (Timing diagram)



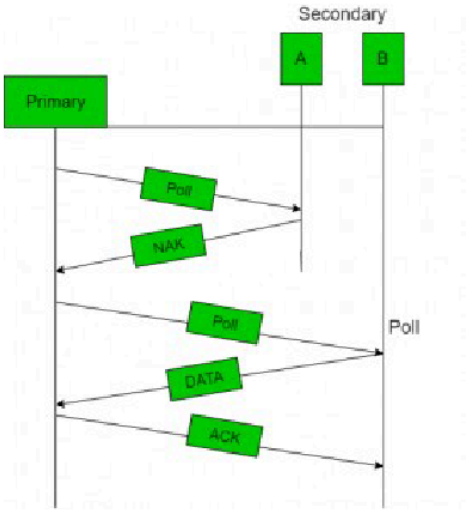
- **6.1.A. Slotted ALOHA** (Transmission line diagram)



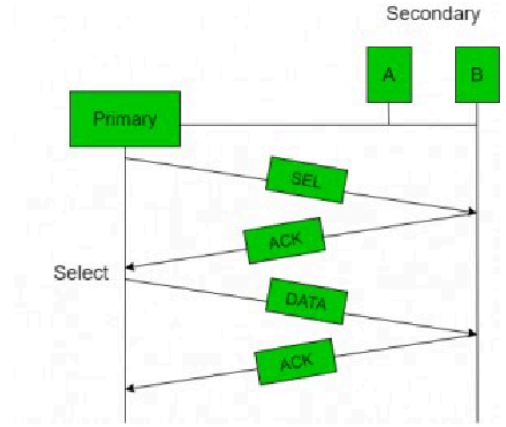
- **6.2.A. Reservation** (Reservation frame example)



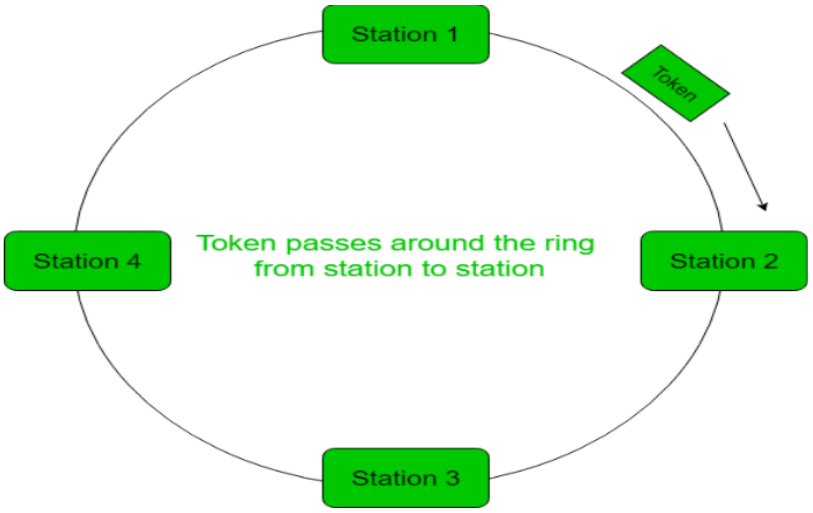
- **6.2.B. Polling** (Polling with NAK)



- **6.2.B. Polling** (Polling with Data)



- **6.2.C. Token Passing** (Token Passing in a ring)



- **7. Wireless Sensor Network (WSN)** (Base Station in WSN System)

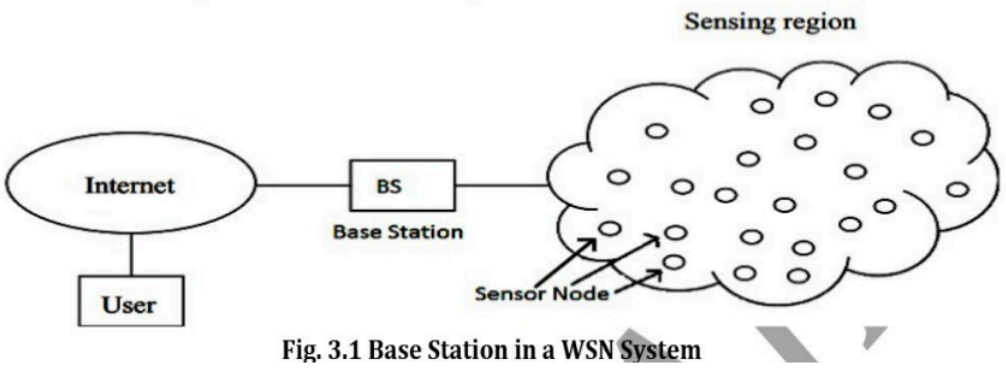
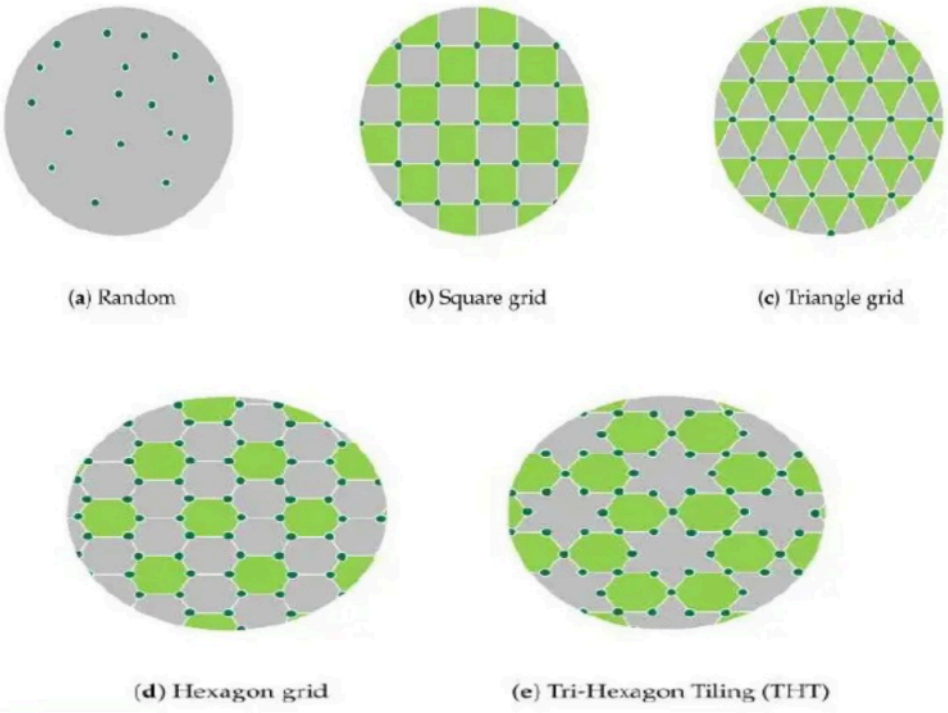
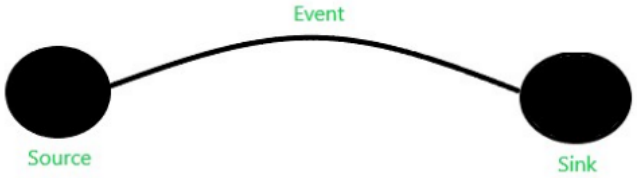


Fig. 3.1 Base Station in a WSN System

- **9.1. Sensor Deployment in WSN** (Regular Deployment Patterns)

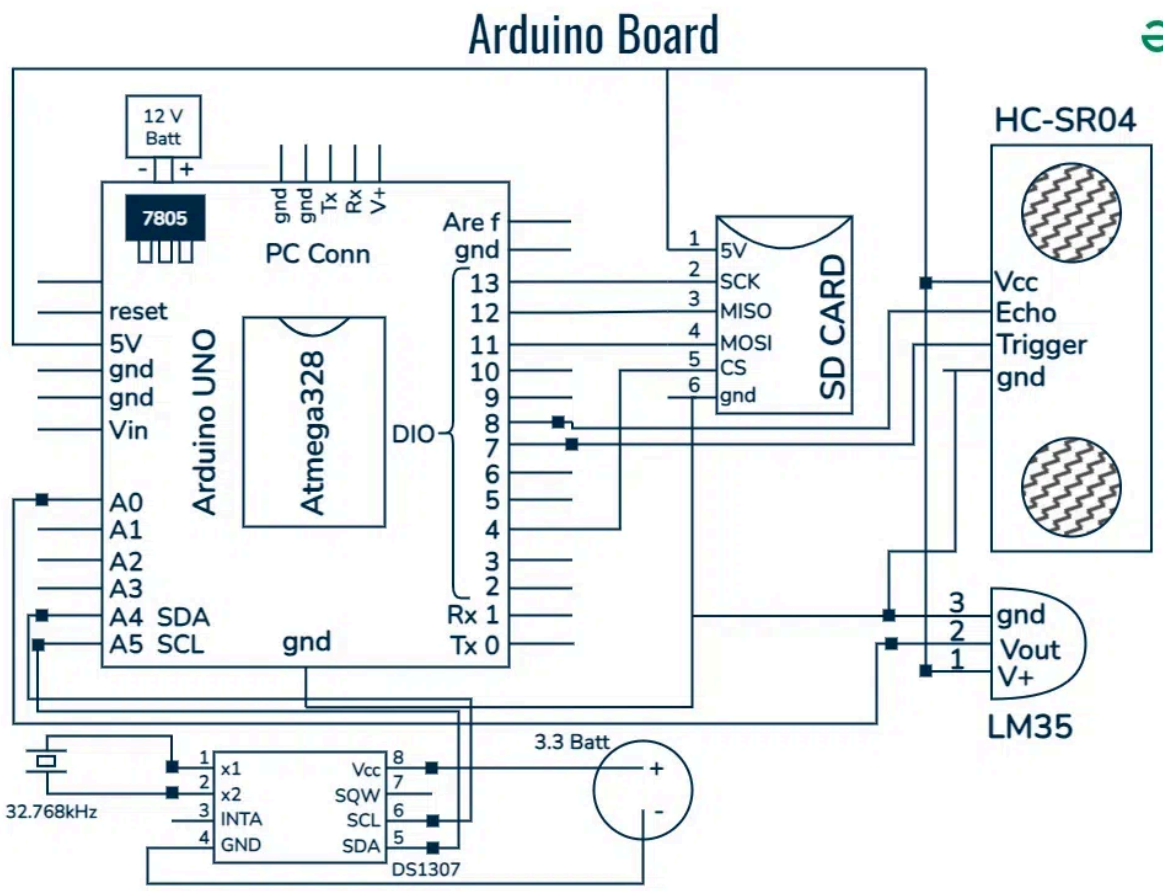


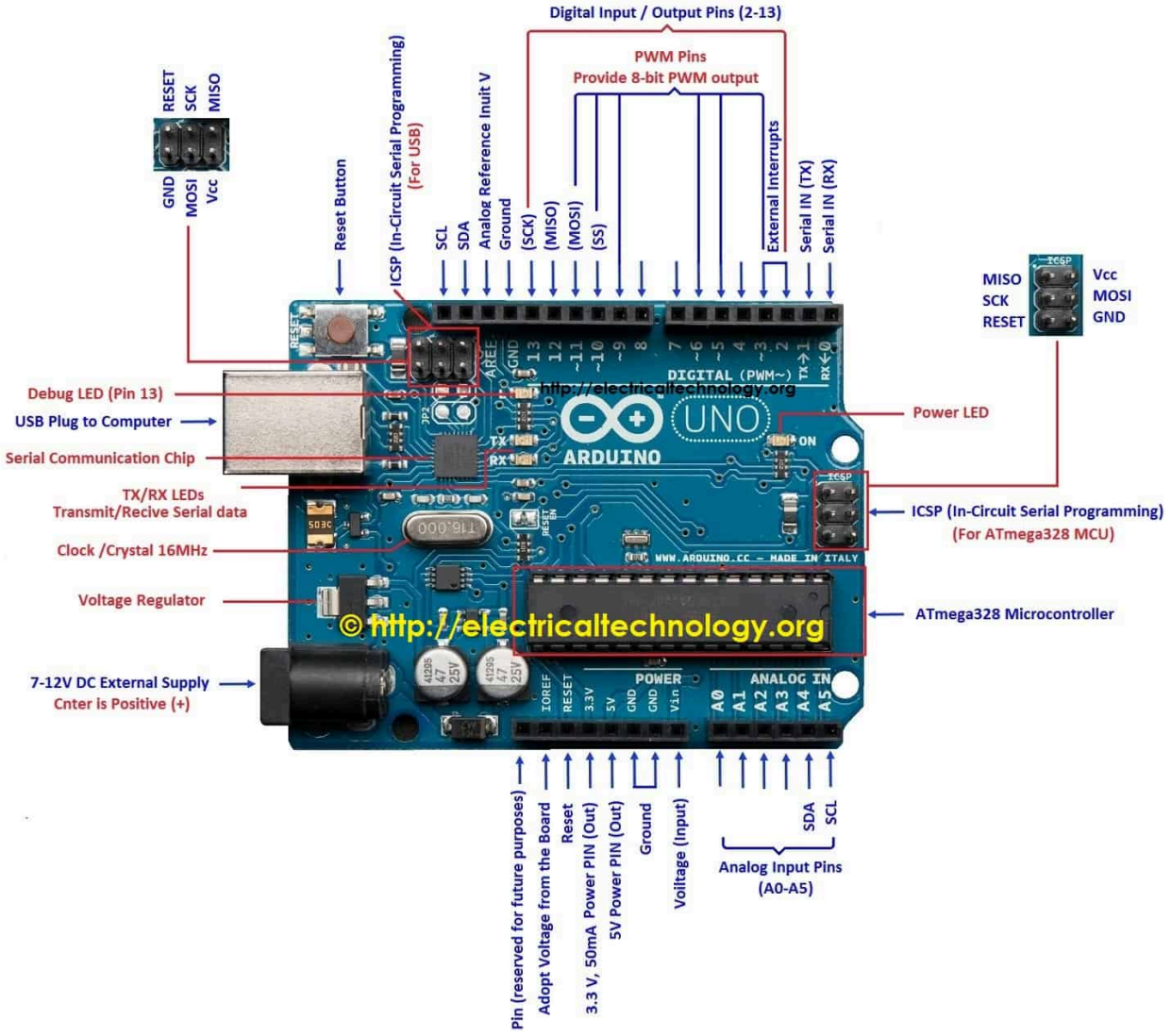
- **10. Data Dissemination in WSNs** (Source-Event-Sink diagram)



Unit 4: Arduino Platform, Anatomy, and IDE

- **UNIT 4 Heading Image** (Arduino platform boards & anatomy Arduino IDE title slide)

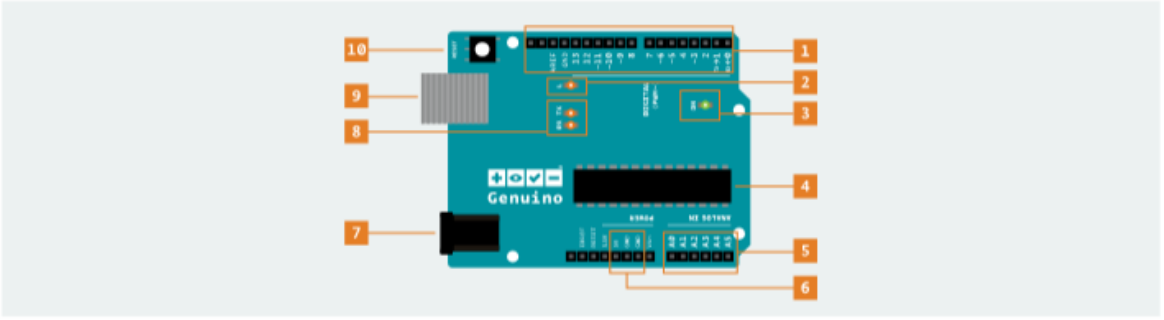

Arduino Board
12 V
Batt
7805
gnd
gnd
Tx
Rx
V+
PC Conn
reset
5V
gnd
gnd
Vin
A0
A1
A2
A3
A4 SDA
A5 SCL
Arduino UNO
Atmega328
gnd
Are f
gnd
13
12
11
10
9
8
7
6
5
4
3
2
Rx 1
Tx 0
DIO
1 5V
2 SCK
3 MISO
4 MOSI
5 CS
6 gnd
SD CARD
HC-SR04
Vcc
Echo
Trigger
gnd
3 gnd
2 Vout
1 V+
LM35
3.3 Batt
+
-
32.768kHz
1 x1
2 x2
3 INTA
4 GND
Vcc 8
SQW 7
SCL 6
SDA 5
DS1307



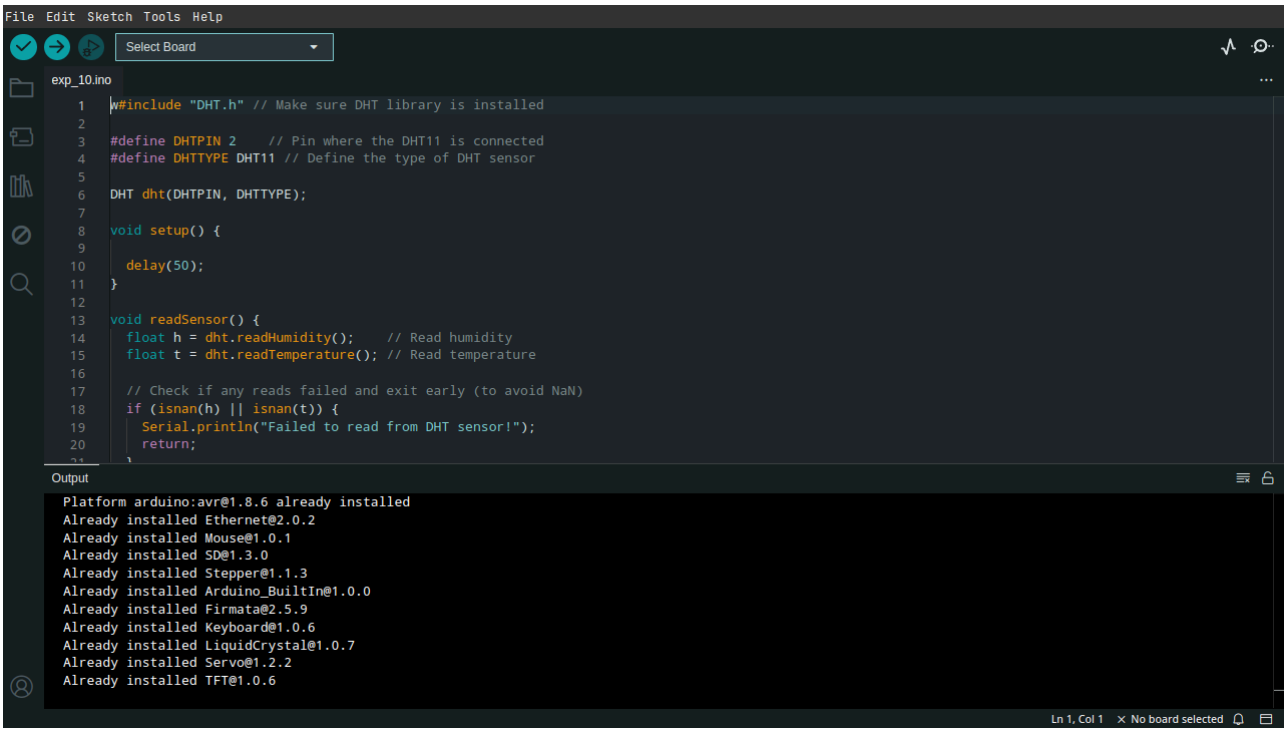
Arduino Programming:
What is Auruino and How to Program it

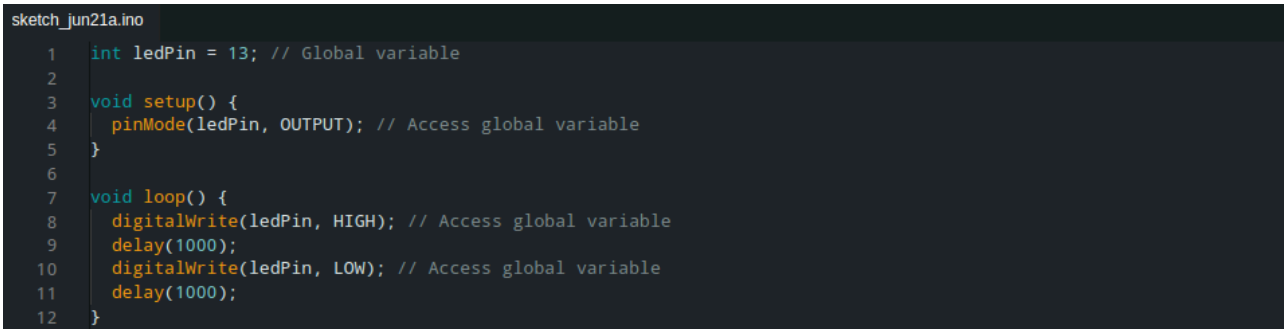
http://electricaltechnology.org

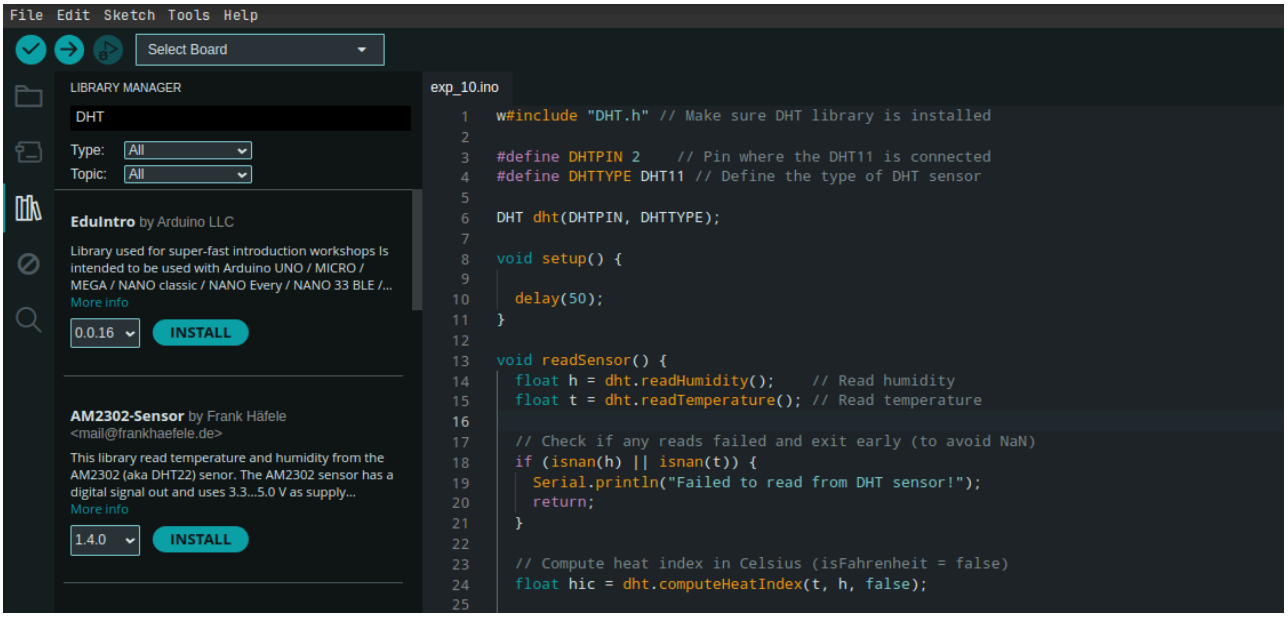
- **3. Arduino UNO Board Anatomy** (Annotated Arduino Uno Board Image)



- **4. Arduino IDE** (Screenshot of Arduino IDE interface)

- 

- **6. Variables and Scope in Arduino** (Global vs Local variable example)
  - 

- **7. Function Libraries in Arduino** (Servo library usage example)
  - 

- **9. Decision Making Statements (if, else) in Arduino** (if statement example)

Arithmetic Operators

+ (addition)

= (assignment operator)

/ (division)

* (multiplication)

% (remainder)

- (subtraction)

Pointer Access Operators

* (dereference operator)

& (reference operator)

(This is for if/if-else combined in the slide)

- **9. Decision Making Statements (if, else) in Arduino** (if-else if-else & switch-case examples)

Comparison Operators

== (equal to)

> (greater than)

>= (greater than or equal to)

< (less than)

<= (less than or equal to)

!= (not equal to)

Bitwise Operators

<< (bitshift left)

>> (bitshift right)

& (bitwise AND)

~ (bitwise NOT)

| (bitwise OR)

^ (bitwise XOR)

(This is for if-else if-else and switch-case combined in the slide)

- **10. Operators in Arduino** (Table of Operators)

Boolean Operators

&& (logical AND)

! (logical NOT)

|| (logical OR)

Compound Operators

+= (compound addition)

&= (compound bitwise AND)

|= (compound bitwise OR)

^= (compound bitwise XOR)

/= (compound division)

*= (compound multiplication)

%= (compound remainder)

-= (compound subtraction)

-- (decrement)

++ (increment)

- **10. Operators in Arduino** (Additions in Arduino code example)

```
1   void setup() {
2     // Start the serial communication
3     Serial.begin(9600);
4
5     // Declare and initialize two numbers
6     int num1 = 5;  // First number
7     int num2 = 10; // Second number
8
9     // Calculate the sum
10    int sum = num1 + num2;
```

- **11. Example: Blink LED in Arduino** (Blink LED code example)
  - (This image was requested in Q9 solution: `![Blink LED in Arduino](:/placeholder_blink_led.png)` - but the provided screenshot is actually from Unit 4, page 13 which is `Programming the Arduino for IoT - Blink LED in Arduino`)
  - The image on page 13 of the Unit 4 notes is directly relevant here.

DHT CIRCUIT